









# A novel intelligent approach for man-in-the-middle attacks detection over internet of things environments based on message queuing telemetry transport

Álvaro Michelena<sup>1</sup>  | José Avelaira-Mata<sup>2</sup>  | Esteban Jove<sup>1</sup>  |  
 Martín Bayón-Gutiérrez<sup>3</sup>  | Paulo Novais<sup>4</sup>  | Oscar Fontenla Romero<sup>5</sup>  |  
 José Luis Calvo-Rolle<sup>1</sup>  | Héctor Aláiz-Moretón<sup>2</sup> 

<sup>1</sup>Department of Industrial Engineering, University of A Coruña, CTC, CITIC, A Coruña, Spain

<sup>2</sup>RIASC: Research Institute of Applied Sciences in Cybersecurity, University of León, León, Spain

<sup>3</sup>Department of Electrical and Systems Engineering, University of León, León, Spain

<sup>4</sup>Department of Informatics, ALGORITMI Centre, University of Minho, Braga, Portugal

<sup>5</sup>CITIC Research and Development - Laboratory in Artificial Intelligence (LIDIA), University of A Coruña, A Coruña, Spain

## Correspondence

José Avelaira-Mata, RIASC: Research Institute of Applied Sciences in Cybersecurity, University of León, Vegazana Campus, León, Spain.  
 Email: [jose.aveleira@unileon.es](mailto:jose.aveleira@unileon.es)

## Funding information

European Regional Development Fund; Instituto Nacional de Ciberseguridad; Ministerio de Ciencia, Innovación y Universidades; Universidad de León

## Abstract

One of the most common attacks is man-in-the-middle (MitM) which, due to its complex behaviour, is difficult to detect by traditional cyber-attack detection systems. MitM attacks on internet of things systems take advantage of special features of the protocols and cause system disruptions, making them invisible to legitimate elements. In this work, an intrusion detection system (IDS), where intelligent models can be deployed, is the approach to detect this type of attack considering network alterations. Therefore, this paper presents a novel method to develop the intelligent model used by the IDS, being this method based on a hybrid process. The first stage of the process implements a feature extraction method, while the second one applies different supervised classification techniques, both over a message queuing telemetry transport (MQTT) dataset compiled by authors in previous works. The contribution shows excellent performance for any compared classification methods. Likewise, the best results are obtained using the method with the highest computational cost. Thanks to this, a functional IDS will be able to prevent MQTT attacks.

## KEYWORDS

artificial neural networks, cybersecurity, decision trees, intrusion detection system, K-nearest-neighbours, man-in-the-middle, message queuing telemetry transport, principal component analysis, random forest

## 1 | INTRODUCTION

The internet of things (IoT) is emerging quickly for having a significant impact on everyday life and large industrial systems. IoT systems are able to integrate systems of devices that make it easy to share information between physical devices without interruption. IoT technologies include everyday objects to which new functionalities can be provided by connecting them to the Internet and equipping them with sensors and actuators. They can even become part of small medical and healthcare devices, industrial applications known as Industry 4.0 and smart city infrastructures Malik et al. (2021). In the sector, IoT systems allow connection to control systems. They offer solutions for remote monitoring, making it

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial](https://creativecommons.org/licenses/by-nc/4.0/) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2023 The Authors. *Expert Systems* published by John Wiley & Sons Ltd.

possible to operate equipment in real-time from any client device. This set of new features makes the security of 4.0 industry IoT devices critical Teoh et al., 2021.

IoT devices have special features, as that, they need to be efficient from energetic point of view and work together a small processing systems. Because of this, they are devices with low-computational capacity Nauman et al. (2020). As a large number of IoT devices can be connected to the same network and their processing capabilities are limited, these devices use different types of lightweight protocols. Some application layer protocols often use in IoT devices are message queuing telemetry transport (MQTT), constrained application protocol (CoAP), extensible messaging and presence protocol (XMPP) and data distribution service protocol (DDS) Krejčí et al. (2018). They also use different types of physical protocols such as Bluetooth, BLE, 6Lowpan or ZigBee using IEEE802.15.4 and LoRa C. Sharma and Gondhi (2018). This provides lightweight communications that reduce network saturation and allow real-time communication.

According to Statista (2022), there will be approximately 75.44 billion connected IoT devices by 2025. The large number of IoT devices represents a huge security challenge. Due to their special behaviour for the variety of protocols working together and low-computational capacity. Attackers can use malicious software to infect them and generate distributed denial of service (DDoS) attacks, focused on information systems or websites. One example of this was the biggest recent attack produced with the Mirai botnet, that compromised 400,000 IoT devices Mahbub (2020). In April 2020, another botnet called Dark Nexus was detected M. H. Khalid, Murtaza, and Habbal (2020), which carried out attacks with 1372 bots.

Threat systems can use different vectors of attack to inject malware that turns devices into Bots. These vectors can be, default configurations, insecure gateways or man-in-the-middle attacks, such as MitM attacks in IoT environments using the BLE Pallavi and Narayanan (2019). MitM attacks are also studied in commercial devices Oconnor et al. (2021), and on different popular protocols such as CoAP Al Enany et al. (2021) and MQTT Chen et al. (2020).

In the energy industry, DoS attacks are dangerous and may lead to a large number of customers running out of electric service Ashraf et al. (2021); Khalid, Muyeen, and Peng (2020). Several works present algorithms that are able to predict anomalies in this kind of environment with the aim of avoiding challenging situations Musleh et al. (2019).

Other attacks, based on false data injections, are placated through anomaly detection systems H. M. Khalid and Peng (2016). In terms of MitM attacks, anomaly detection procedures can be, one more time, the way to avoid this threat Eigner et al. (2016). Authors such as Esfahani et al. (2019) developed active security processes based on authentication techniques in order to prevent Men in the Middle attacks.

In order to guarantee security in IoT environments, the implementation of an IDS is the best solution, because it does not change the behaviour and the configuration of existing IoT systems. Rule-based IDS such as Suricata or Snort are complex to implement, since they need to create new rules for the new IoT environments and their vulnerabilities N. V. Sharma, Kavita, Aggarwal, and Sharma et al. (2021). For this reason, anomaly-based IDS can be adopted to detect attacks by monitoring the entire system, comparing anomalous traffic with normal traffic. To improve IDS systems, machine and deep learning models can be trained with large datasets, which are made up of normal and attack frames Yang et al. (2022).

One of the ways to improve IDS systems today, is the use of soft computing techniques, more specifically, machine and deep learning models through the use of datasets. Commonly, the datasets for modelling purposes, contain normal traffic and traffic generated by specific attacks. In this way, the set of frames included in the dataset reflects the behaviour of this type of environment, when threatened Inayat et al. (2022); Khraisat and Alazab (2021); Yang et al. (2022).

Soft computing techniques selected in this paper are used with the aim to make the models lightweight. This means that these models may be deployed into an IDS system that captures traffic in real-time and therefore, they may detect different anomalies.

The most commonly used datasets for IDS systems are the KDD and its enhanced version NKI-KDD Meena and Choudhary (2017). Although, these datasets are not focused on IoT systems traffic, they have been used to detect DDoS and DoS attacks in IoT environments Liu et al. (2020) Fatani et al. (2022). An example of an IEEE 802.11 network dataset is AWID Koliás et al. (2016), which contains attacks on 802.11 systems, with packet dissection for 802.11 traffic but without focusing on IoT environments. Due to the most IoT environments use WiFi connections, this kind of dataset is also used for knowing the behaviour of attacks in IoT environments Rahman et al. (2021). The most common dataset for IoT environments is “IoT-23”, focused on DNS traffic for IoT context which is used for developing IDS models to detect IoT botnets Abdalgawad et al. (2022).

Regarding datasets in IoT environments, there are several MQTT datasets, however they do not focus in specific vulnerabilities associated to the MQTT protocol. The “MQTT-IoT-IDS2020” dataset contains only MQTT traffic from a simulated environment, being useful only for detecting attacks such as the Brute Force Scanning Hindy et al. (2021). The “MQTTSet” dataset is based on simulated environment with IoT-Flock tool, formed only by MQTT traffic with attacks such as Brute-force, malformed data and DoS Ullah and Mahmoud (2021).

There is a MQTT dataset, called the “Bot-IoT”, that contains two attack categories: denial of service (DoS) and distributed denial of service (DDoS). Additionally it has MQTT protocol non-attacking traffic simulating a weather station in order to define an example of a common environment. Thank to this, the dataset can be used to analyse the impacts of DoS attacks in the MQTT protocol, in a real case, such as the attack to weather station presented in Koroniotis et al. (2019). Other works with “Bot-IoT” are reviewed in depth in Peterson et al. (2021) where different techniques are applied to this dataset.

The aim of the authors in this work is to improve the security of IoT systems that use the MQTT protocol. This protocol has specific vulnerabilities due to the fact that it is a lightweight protocol Raikar and Meena (2021). Therefore, attackers can take advantage of these specific attacks, including the MitM attack, which is particularly difficult to detect. This paper focuses on a novel approach implemented like a two-stage hybrid methodology for developing intelligent models working in an IDS for detecting the man-in-the-middle attack on MQTT networks. The first stage is oriented to feature extraction and the second one addresses the implementation of supervised classification techniques.

A high quality dataset is critical when a methodology based on soft computing is applied. As could be seen previously, the existing MQTT dataset do not have included specific vulnerabilities such as Men in the Middle. For this reason, in previous works, authors developed a new dataset capturing real traffic in a real environment constituted by IoT systems that use the MQTT protocol. Therefore the dataset contains the overall traffic of a network Alaiz-Moreton et al. (2019). This dataset was used to achieve good results implementing auto-encoder techniques García-Ordás et al. (2020). In other previous works, a multi-classification approach for Intrusion, DoS and MitM attacks was performed, taking only the most relevant features using previously a entropy function. The final trained model was useful for the detection of DoS attacks, but did not get the same level of success for MitM or Intrusion attacks Aveleira-Mata et al. (2021).

The rest of this manuscript is structured as follows. Section 2, introduces the case study, detailing how the test environment has been deployed, to later describe in depth our dataset, that will be used in this work. Several soft computing methods, such as Principal Component Analysis, K-nearest-Neighbours and Artificial Neural Networks, are presented in Section 3. Section 4 describes the parameter configuration for each one of the soft computing methods presented. Several performance metrics are introduced in Section 5 along with a discussion about the best method of the ones compared. Finally, Section 6 presents our conclusions and a set of possible future work.

## 2 | CASE STUDY

In a Man in-the-Middle attack, the attacker stealthily analyzes the traffic between two parts that expect to be communicating directly with each other. The attacker is situated between the client and the application. The goal of this attack is to collect information, or modify this information by false data Mallik (2019).

In IoT environments the basic purpose of devices is to collect information from device sensors, to interact with the environment with different actuators too.

This makes that the malicious manipulation of IoT data by the attackers may cause real-time and real-life catastrophes Čekerevac et al. (2017).

The MQTT protocol is a light publication/subscription messaging protocol, which works on TCP, designed for M2M (machine to machine) communications. MQTT is based on an extremely lightweight broker-based publish/subscribe messaging protocol, for small code footprints (e.g., 8-bit, 256 KB ram controllers), being an excellent solution for connecting devices in networks with low bandwidth Hintaw et al. (2021). The MQTT protocol system follows a star topology with a central node that functions as a server called *Broker*. The broker is the responsible for managing the network and transmitting the different messages in real-time.

In order to obtain a dataset with real data from an MQTT attack, an MQTT test environment has been deployed. The environment follows a common architecture for IoT networks, composed of: a node.js server, which acts as the MQTT broker, a relay, which acts as an actuator and finally, a HC-04 sensor. Devices are equipped with NodeMCU embedded boards that allow them to communicate in the MQTT network. In addition a set of clients, such as a personal computer (PC) and several smartphones, that interact with actuators and sensors through the application, have been added with the aim of generating network traffic to the Internet.

In the MitM attack over MQTT environment, the attacker is located between the broker and the distance sensor Figure 1 for changing the data and thus, take advantage of the vulnerabilities of the protocol OASIS Open (2015), in this way all the clients subscribed to the distance sensor topic will receive the false information.

MQTT packet structure is composed by a fixed header, with a length of 2 bytes, an optional message-specific variable length header and message payload, as indicated in Table 1.

For carrying out this attack, a Kali Linux distribution with the Ettercap tool has been chosen for scanning the hosts of the network, for late to add the addresses where the attacker has been located, choosing the broker IP and the client IP to place the attacker between broker and client. The next step is to modify the MQTT messages that the sonar sends to the server. For this, the 'nfsqsd' program Gerganov (2018) modifies network traffic using a predefined set of substitution rules. The rules take into account the composition of the MQTT packet to change the payload data.

In the environment, a low-cost TP-Link TL-WR1043N/ND v3 router is used, with the Linux-based open source software OpenWrt OpenWRT (2020). All traffic generated, from browsing the Internet and all traffic interacting with the IoT environment, is stored in a 'Pcap' file by the Router.

Due to the large number of packet fields, the 'pcap' file is dissected, considering the TCP/IP packet fields as well as all the fields belonging to the MQTT protocol, taking the AWID dataset as a reference Chatzoglou et al. (2021), for keeping the following fields:

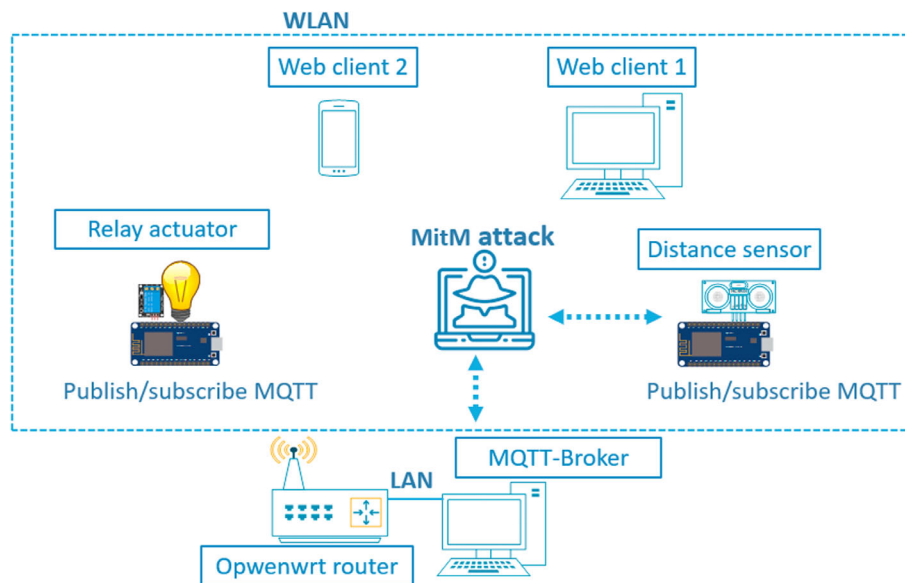


FIGURE 1 MitM attack environment.

TABLE 1 MQTT message

Bit	7	6	5	4	3	2	1	0
Byte 1	Message type				DUP	QoS		Retain
Byte 2	Remaining length							

- Frame level fields: common to all frames, they can be used or not the MQTT protocol. It is possible that traffic is generated for the attack does not directly affect the MQTT protocol. Contains a total of 27 fields. For example frame.time\_delta, frame.cap\_len, frame.interface\_name.
- MQTT protocol fields: these are all the fields related to the MQTT protocol, with a total of 38 fields. These fields only have relevant information in the frames that use this protocol, keeping these fields empty in the rest of the frames, some examples of the fields used are, mqtt.clientid, mqtt.clientid\_len, mqtt.conack.flags
- Frame labeling: contains a special field 'type' to label the frames that are under attack with the name of the attack, in this case, MiTM. It is labelled considering the timestamp of when the attacks were executed. This type of frames are very important in order to apply supervised machine techniques.

The result is a dataset with MitM attacks that contains a total of 80.893 total frames, 1.898 of which are under attack and 78.995 are normal traffic frames (the dataset is available at <https://joseaveleira.es/dataset>. © (reg#LE-229-18)).

### 3 | SOFT COMPUTING METHODS

#### 3.1 | Feature extraction methods

In order to reduce training time and improve classifier performance, a dimensional reduction technique is implemented. For this purpose, the Principal Component Analysis technique is applied.

##### 3.1.1 | Principal component analysis

Principal component analysis (PCA) is a statistical technique to determine the correlation between multiple variables. PCA is commonly used for dimensional reduction; however, this method also has other applications, such as data compression or classification. The main goal of PCA is to reduce the dimensionality of the dataset with a minimal loss of information Bishop and Nasrabadi (2006). Therefore, PCA obtains a set of

orthogonal axes that maximize data variance. These axes are known as *principal components* that correspond to linear combinations of the original features. Figure 2 shows an example of PCA operation for a dimensional reduction in  $\mathbb{R}^2$ . The principal components obtained are shown in blue and orange.

PCA technique computes the data covariance matrix and obtains the eigenvectors using eigenvalue decomposition. The eigenvectors with the highest eigenvalues represent the *principal components* since they explain most of the variance in the data.

### 3.2 | Supervised machine learning techniques

Machine learning is characterized by using algorithms based on mathematical and statistical techniques to develop predictive systems. Although there are significant differences in how machine learning algorithms are classified, they may be categorized into four major groups based on their aims and how the underlying machine is taught: supervised, unsupervised, semi-supervised and reinforcement learning.

Supervised machine learning techniques are techniques that require a labelled dataset for training. They can be used for classification and regression problems Hastie et al. (2009); Mohamed (2017). Currently, supervised machine learning techniques have been implemented with success in multiple fields such as medicine Baumgartner et al. (2004), security Gharibian and Ghorbani (2007), and food-processing industry Saha and Manickavasagan (2021), among others. The methods implemented in this research are described below.

#### 3.2.1 | K-nearest-neighbours

K-nearest-neighbours (kNN) is a simple and easy-to-implement nonlinear supervised machine learning technique widely used in classification problems Moldagulova and Sulaiman (2017). This method is based on the idea that data of the same class are close in space. Therefore, kNN assumes that the data class depends only on the majority class of its k neighbours. So a fine adjustment of k is necessary to achieve good

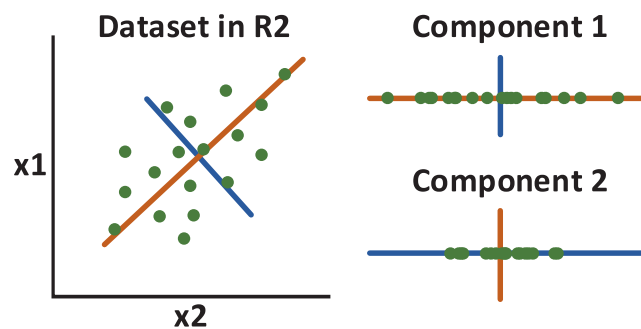


FIGURE 2 Example of component extraction with PCA.

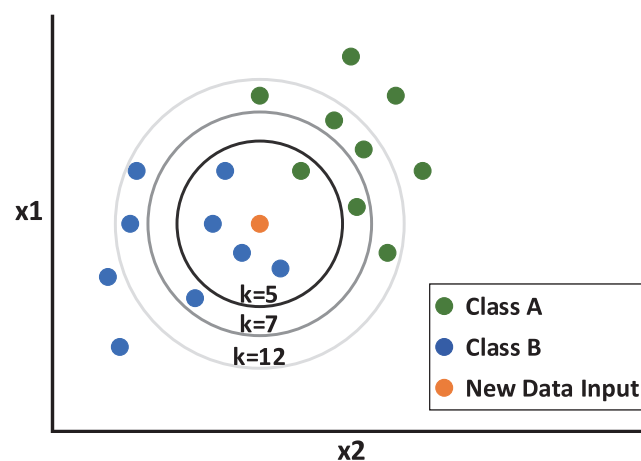


FIGURE 3 Example of k-NN classifier.

performance. There are different ways of choosing the  $k$  value, but one of the most frequent is comparing the results obtained with different  $k$  values and selecting the value that best fits the problem. Figure 3 shows the K-NN method for different  $k$  values schematically.

The kNN algorithm is lazy since no model is generated during training, but only the training data is stored. Thus the training of this method is very fast; however, the classification process of new input data takes considerable time because it is necessary to calculate the distance of this point to the rest of the stored data. Different distance measures such as Euclidean, Manhattan, Minkowski, and others can be used. Therefore, the kNN method is computationally costly, especially when there is a large amount of training data.

### 3.2.2 | Decision trees

Decision trees (DT) are a nonparametric supervised machine learning technique widely used for classification and regression problems Kotsiantis (2013). This method uses a tree-structured classification model, which consists of two different types of nodes:

- Decision nodes: these nodes are associated with one of the variables in the data set and have two or more output branches representing the values that variable can take.
- Response nodes (leaves): these nodes are associated with a specific class and return the classification of the tree.

Figure 4 shows an example structure of a decision tree classifier. Decision nodes are shown in red, whereas response nodes are shown in orange.

The decision nodes are joined together to form stages of the tree. Therefore, for a good performance of the model, selecting the variables in each of the stages of the tree is essential. To select the variables at each node, different criteria have been proposed. Most of them are based on measuring the variable uncertainty degree by calculating its entropy (or Gini index). The information gain value can be obtained from the entropy by comparing the dataset impurity before and after the node splitting. Finally, the tree is constructed from top to bottom by placing the variables with the highest information gain in the first layers of the tree.

Therefore, this method stands out for its ease of interpretation in the model classifications since the tree can be visualized. On the other hand, it is a method that cannot generalize well in complex models (overfitting).

### 3.2.3 | Random forest

Random forest (RF) is a supervised machine learning approach that is commonly used to solve classification problems but can also be used to solve regression problems. This method is based on using a certain number of decision trees, with the classification result of the RF being the class obtained in most of the implemented decision trees Parmar et al. (2018). Figure 5 shows the structure of an RF classifier.

Each decision tree is trained with a different sample of the training data. The Bootstrap Aggregation method is used to obtain these subsets of training data. This technique, commonly known as Bagging, is an ensemble strategy for fitting decision trees that generates random subsets of

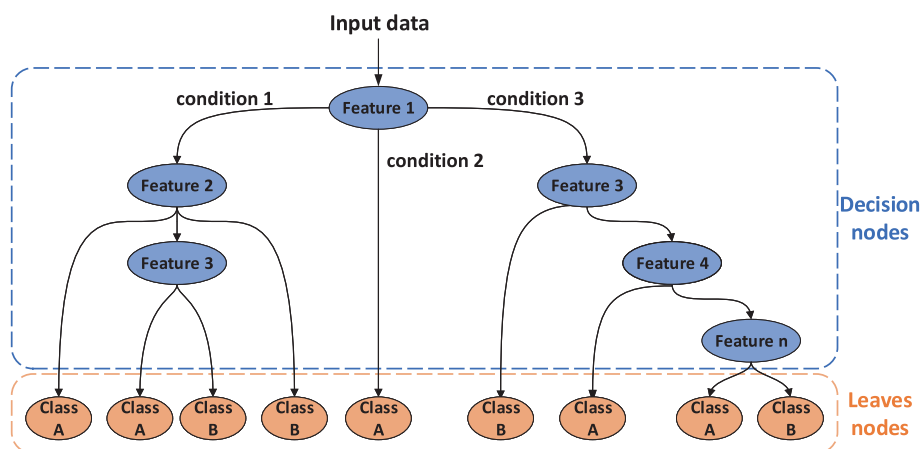
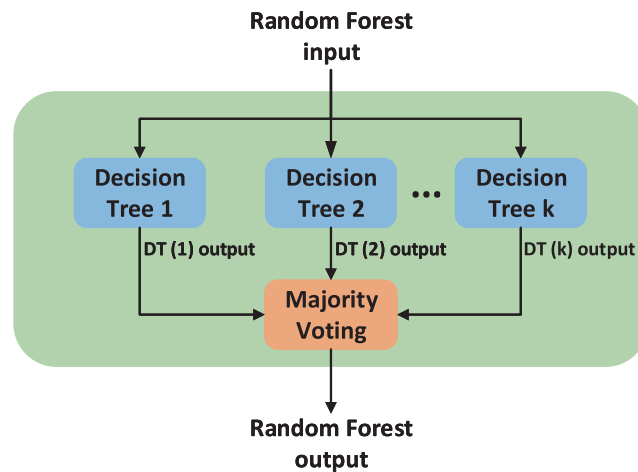
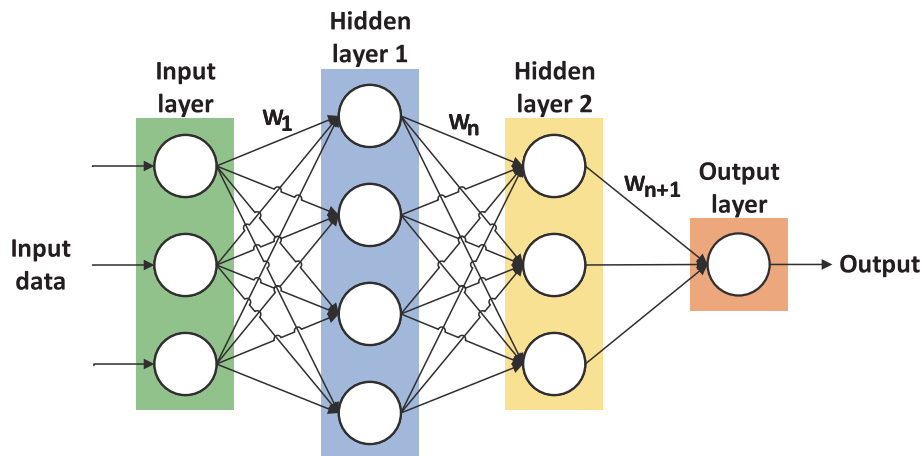


FIGURE 4 Example of decision tree classifier.



**FIGURE 5** Example of random forest classifier.



**FIGURE 6** Example of an MLP network structure.

the entire dataset. Since the bagging selection approach is equivalent to a substitution selection procedure, the same data sample can appear in many subsets.

### 3.2.4 | Artificial neural networks

Artificial neural networks (ANN) is a method that corresponds to a mathematical model inspired by the human brain and its neural networks. Thus, ANNs use artificial neurons, which are interconnected, forming a network or matrix Da Silva et al. (2017).

The artificial neuron is a mathematical model that uses three simple operations to obtain an output value from the input data. First, each neuron input is weighted or multiplied by a specific value (weight). Then, all the inputs weighted and an independent value, known as bias, are added to obtain a single value. Finally, the sum value is transformed by an activation function to obtain the artificial neuron's output value. Different activation functions can be used, such as, linear, sigmoid and so on.

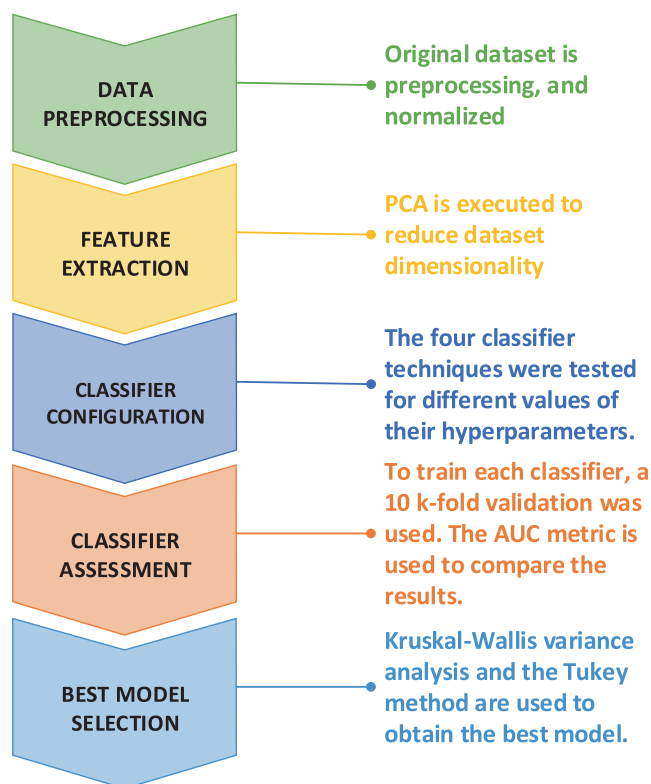
These neurons are organized in layers that define the structure of the neural network. Once the network is created, it must be trained to find the optimal weights for each layer. The training process is based on the back-propagation method, which is efficient for calculating the gradients necessary to optimize the network weights. Also, dropout layers are often used to modify the network regularization to avoid overfitting problems. This type of layer randomly ignores some network neurons. A probability value can tune the weight or influence of this layer on the network.

There are a wide variety of ANN configurations in function of the layers and neurons connections, the number of layers and neurons and so on. However, the Multilayer Perceptron (MLP) is the most popular ANN, with one input layer, one output layer, and one or more hidden layers. Figure 6 shows an example of the structure of an MLP network with two hidden layers. This kind of ANN is the one implemented in this research.

## 4 | EXPERIMENTS SETUP

This section outlines the experimental setting and the measures used to assess and compare each classifier. The Python programming language was used, as well as the machine learning packages Scikit learn and Keras. The experimental setup followed the steps shown in Figure 7. These steps will be described in detail below.

- **Preprocessing:** First, the original dataset was preprocessed. Initially, variables that remained constant for all recorded data were eliminated. In addition, variables that contained more than 10% of missing data were discarded. On the other hand, non-numeric variables were recoded using integer coding. Finally, in order to obtain the best results, the data were normalized using the standardized mean 0 and standard deviation 1.
- **Feature extraction:** Once the dataset was preprocessed and normalized, the PCA method was executed to check if the dataset could be dimensionally reduced. To this end, the method was run to determine the principal components obtained and the variance percentage explained by each. Taking into account this measure, the components used were selected.
- **Classifier configuration:** With the main goal of achieving the best intelligent model, the four supervised machine learning techniques explained above have been tested for different configurations of their hyperparameters. These configurations are listed below:
  - a. **k-NN:** The model was tested for different k-neighbour values. The values tested were: 5, 10, 15, 20, 20, 30, 40, 40, 50, 80 and 100. The distance used was the Euclidean distance.
  - b. **DT:** The decision tree method was tested for different maximum values of its depth, indicating the maximum number of layers the generated decision tree can have. The values tested were: 3, 5, 8, 8, 10, 10, 15 and “None” (the tree depth is not limited). Entropy was used as an uncertainty measure.



**FIGURE 7** Experiments setup.



- c. RF: The performance of the RF method was tested for the different number of decision trees used. The values tested were: 5, 10, 15, 20, 25, 30 and 40. As in the DT method, entropy was used as an uncertainty measure.
  - d. ANN: Different neural network structures were generated. In this case, the number of neurons per layer, the number of layers and the dropout value were varied. Networks of 1, 2, 3 and 4 layers with 2, 5, 7 and 10 neurons were tested. For each network created, their performance was tested for a dropout value of 20% and without applying it. On the other hand, ReLu activation functions were used for the hidden layer neurons and Softmax for the output layer (commonly used in classification problems).
- Classifier assessment and best model selection: Once the different classifiers were created, they were trained. A cross-validation process with a value of  $k = 10$  was used for this purpose. Then, with the trained models, classifiers of the same type were compared to check which configuration of their hyperparameters fits better. Finally, the best models of each technique were compared to obtain the one best adapted to the proposed problem. The metric used to compare the classifiers was the Area Under the receiving operating characteristic Curve (AUC) metric Bradley (1997), since this metric is insensitive to data distribution, and its use with unbalanced datasets is highly recommended. The Kruskal-Wallis analysis of variance was used together with Tukey's comparison method to evaluate the results of the created models. These two statistical methods are usually used together. The Kruskal-Wallis analysis of variance is used to determine whether or not, there are significant differences between the models compared. If there are differences in the results, Tukey's method is used to detect the different ones. Finally, the highest AUC model is chosen among the others detected as different by Tuckey's method. If the AUC value is the same for several ones, or if the analysis of variance has not detected any difference between them, the configuration with the lowest training time, will be selected.

For a better understanding of the experiment configuration, the pseudo code used is shown in [Algorithm 1](#) and [Algorithm 2](#). [Algorithm 1](#) corresponds to the main execution algorithm that uses several functions of well-known Python libraries such as Pandas or Scikit-learn, among others. On the other hand, [Algorithm 2](#) reflects the comparison process.

## 5 | RESULTS AND DISCUSSION

Once the configuration of the experiments performed has been described, the results obtained with each supervised technique are presented. Although the metric considered is AUC, this section also includes other metrics such as Recall, Accuracy, F1-Score and Specificity that provide complementary information on the classifier performance.

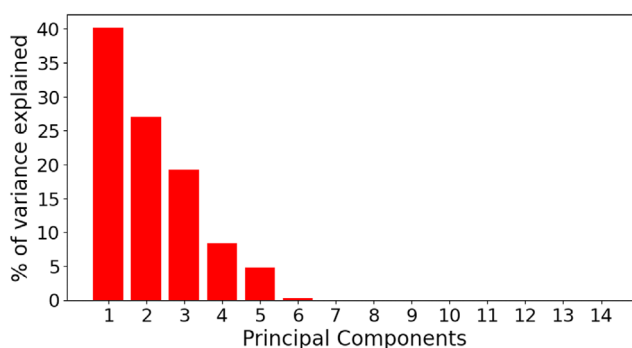
It is essential to highlight that for the execution of the experiments, a computer with Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz and a RAM of 8 GB has been used.

### Algorithm 1 Main algorithm

- 1: The configuration json file is loaded. This file contains different information of the experiment (*config*)
- 2: The data set is loaded (*data*)
- 3: Preprocessing and normalization ( $data_{nor} = preprocess_{func}(data)$ )
- 4: Feature extraction ( $PCA_{components} = pca_{func}(data_{nor}, config)$ )
- 5: Data split ( $data_{train}, data_{test} = split_{func}(PCA_{components}, data_{nor})$ )
- 6: **for** Each classification technique store in *config* file **do**
- 7:     **for** Each hyperparameter configuration store in *config* file **do**
- 8:         Train classifier ( $Trained_{model} = train_{func}(Model_i, data_{train})$ )
- 9:         Test classifier ( $Performance = test_{func}(Trained_{model}, data_{test})$ )
- 10:         Save performance results ( $Model_{results}$ )
- 11:     **end for**
- 12:     The best technique configuration is selected by means of [Algorithm 2](#) ( $Best_{setup} = comparison_{func}(Model_{results})$ )
- 13:     Save results of the best configuration model ( $Bestmodels_{results}$ )
- 14: **end for**
- 15: Finally, all techniques are compared and the best classifier is selected. [Algorithm 2](#) ( $Best_{model} = comparison_{func}(Bestmodels_{results})$ )
- 16: Final results are saved and displayed

**Algorithm 2** Comparison<sub>func</sub>

- 1: The results of the models to be compared are uploaded in (*Models*)
- 2: Kruskal-Wallis analysis ( $p_{val} = \text{Kruskal} - \text{Wallis}_{func}(\text{Models})$ )
- 3: **If** detect difference between models ( $p_{val} > 0.05$ ) then
- 4:   Tukey method ( $\text{different}_{model} = \text{Tukey}_{func}(\text{Models})$ )
- 5:   From the different classifiers select the one with the highest AUC
- 6: **else**
- 7:   Select the model with the shortest computational time
- 8: **end if**
- 9: Returns the best model of those compared



**FIGURE 8** Results of principal component analysis.

As mentioned in the previous section, a PCA analysis is performed once the original dataset has been preprocessed. The aim is to check the number of components that can be selected for dimensional reduction. Figure 8 shows the results obtained with PCA as a bar diagram. The plot shows the principal components on the horizontal axis and on the y-axis the percentage of explained variance. Analysing the results (Figure 8), it can be concluded that the PCA method can obtain six principal components to represent the dataset without loss of information. Therefore, in order to reduce training time and improve results, the six principal components have been selected as the characteristic variables, used to train and validate the models.

Once the principal components have been selected, the performance of the different techniques have been trained and tested. As mentioned in the previous section, the Kruskal-Wallis variance analysis method was combined with the Tukey comparison method, using AUC as the fundamental metric to compare the different classifiers and obtain the best one. These two statistical methods have been applied to select each technique's best configuration, and also to compare the best models of the different techniques.

First, the performance of the k-NN method has been tested for different values of k. The results obtained are shown in Table 2. As can be seen, the k-NN method obtains an AUC value higher than 90% for any k value. Once the Kruskal-Wallis and Tukey methods compared the k-NN classifiers, the  $k = 5$  was selected as the best classifier since, among the different classifiers, it was the one that obtained the highest AUC and one of those that required the least computational cost.

On the other hand, the decision tree results were also excellent (Table 3), achieving AUC values above 90% for the different hyperparameters selected. In this case, the variance analysis determined differences in the methods, so the one with the highest AUC was selected. The best Decision Tree classifier is obtained for a maximum depth of 15, reaching an AUC value of 95.1%.

The results obtained with RF classifiers also showed good performance in detecting man-in-the-middle attacks (Table 4). In this case, when performing the analysis of variance, no significant differences were obtained in the different classifiers, so the one with the lowest computational cost was selected. Therefore, the RF model with five decision trees was selected since its training time is significantly less than the other RF models.

Finally, the performance of the neural networks was evaluated. The results obtained are presented in Table 5. As in the other methods used, the neural networks obtained a high performance with AUC values above 91%. The best method was the neural network with four hidden layers,

**TABLE 2** K-Nearest-Neighbours results

k-NN						
k-neighbours	AUC	Recall	Precision	F1_score	Specificity	Train time (s)
5	<b>0.948</b>	<b>0.898</b>	<b>0.943</b>	<b>0.920</b>	<b>0.998</b>	<b>9.095</b>
10	0.934	0.870	0.963	0.914	0.999	8.478
15	0.931	0.864	0.949	0.904	0.998	11.132
20	0.925	0.851	0.956	0.900	0.999	16.598
30	0.927	0.856	0.945	0.898	0.998	9.031
40	0.923	0.848	0.948	0.895	0.998	10.264
50	0.919	0.840	0.954	0.893	0.999	11.990
80	0.917	0.837	0.934	0.883	0.998	13.868
100	0.917	0.836	0.926	0.878	0.998	12.125

Note: The best results are highlighted in bold.

**TABLE 3** Decision tree results

Decision tree						
Max. Depth	AUC	Recall	Precision	F1_score	Specificity	Train time (s)
3	0.918	0.841	0.875	0.858	0.996	5.062
5	0.931	0.864	0.924	0.893	0.997	7.955
8	0.949	0.902	0.905	0.904	0.997	16.308
10	0.947	0.897	0.919	0.908	0.997	14.617
<b>15</b>	<b>0.951</b>	<b>0.905</b>	<b>0.915</b>	<b>0.910</b>	<b>0.997</b>	<b>11.146</b>
None	0.949	0.902	0.909	0.905	0.997	15.172

Note: The best results are highlighted in bold.

**TABLE 4** Random Forest results.

Random forest						
N° of trees	AUC	Recall	Precision	F1_score	Specificity	Train time (s)
5	<b>0.948</b>	<b>0.898</b>	<b>0.938</b>	<b>0.917</b>	<b>0.998</b>	<b>12.798</b>
10	0.945	0.892	0.948	0.919	0.998	26.545
15	0.948	0.898	0.942	0.920	0.998	40.746
20	0.945	0.893	0.944	0.918	0.998	45.677
25	0.949	0.901	0.945	0.922	0.998	47.661
30	0.945	0.893	0.943	0.917	0.998	55.369
40	0.947	0.896	0.944	0.919	0.998	63.340

Note: The best results are highlighted in bold.

without Dropout and with 10 neurons per layer. Model selection is since the variance analysis determined differences among the models. Of the different models detected by the Tuckey comparison method, this was the one with the highest AUC (99.2%).

To conclude, the best model to detect man-in-the-middle attacks is selected. To this end, the Kruskal-Wallis analysis of variance was performed, which determined that the selected models showed differences. Therefore, the Tuckey method is used to obtain the results presented in Table 6. This table shows that the ANN model presents differences compared to the rest of the classifiers. Therefore, considering that the ANN model has the highest percentage of AUC (99.2%), it is selected as the best classifier.

**TABLE 5** Artificial neural network results

Artificial neural network (MLP)								
N° hidden layers	N° neurons per layer	Dropout (%)	AUC	Recall	Precision	F1_score	Specificity	Train time (s)
1	2	0	0.979	0.864	0.856	0.923	0.995	355.914
1	2	20	0.977	0.590	0.938	0.737	0.999	342.450
2	5	0	0.989	0.864	0.906	0.923	0.996	487.833
2	5	20	0.983	0.660	0.711	0.719	0.997	419.771
3	7	0	0.991	0.847	0.953	0.914	0.998	424.975
3	7	20	0.983	0.841	0.924	0.910	0.997	443.153
4	10	0	<b>0.992</b>	<b>0.844</b>	<b>0.957</b>	<b>0.912</b>	<b>0.999</b>	<b>437.460</b>
4	10	20	0.989	0.866	0.889	0.924	0.996	439.303

Note: The best results are highlighted in bold.

**TABLE 6** Results of Tuckey's multiple comparison method

Tuckey multicomparison method ( $p$ -value = 0.05)							
Model A	Model B	Mean diff	$p$ -adj	Lower	Upper	Reject	Different models
ANN	DT	-0.0416	0.0010	-0.0511	-0.0321	True	Yes
ANN	KNN	-0.0442	0.0010	-0.0536	-0.0347	True	Yes
ANN	RF	-0.0445	0.0010	-0.0540	-0.0350	True	Yes
DT	KNN	-0.0026	0.8744	-0.0121	0.0069	False	No
DT	RF	-0.0029	0.8207	-0.0124	0.0066	False	No
KNN	RF	-0.0003	0.9000	-0.0098	0.0091	False	No

## 6 | CONCLUSIONS AND FUTURES WORKS

The present research shows four supervised classification methods, complemented with a dimensional reduction technique based on the Principal Component Analysis for detecting MitM attacks over IoT environments. The obtained results from the experiments developed, have given highly satisfactory performance. In general, the four proposed methods have exceeded 90% AUC. Furthermore, the different models created have been evaluated using the Tuckey comparison method and the Kruskal-Wallis analysis of variance. Thanks to these statistical methods, it was able to determined that the best model was obtained with neural networks, reaching 99.2% AUC for a model without dropout, with 4 hidden layers and 10 neurons per layer.

This paper has proved how combining any proposed method with the PCA feature extraction process provides a robust and reliable solution for detecting these attacks. These results improve on previous approaches addressed by the authors when a multi-classifier was implemented for several kinds of MQTT attacks, using a dataset with samples of the different ones. Therefore, generating a specific dataset for the MitM attack for applying machine learning techniques is the best approach.

In future work, other types of dimensional reduction and feature extraction methods could be considered to compare their performances. In addition, the authors also consider the application of unsupervised (one-class) methods to avoid the need for a labelled dataset, which would facilitate the possibility of applying a larger dataset.

### ACKNOWLEDGEMENTS

Spanish National Cybersecurity Institute (INCIBE) and developed Research Institute of Applied Sciences in Cybersecurity (RIASC). CITIC, as a Research Center of the University System of Galicia, is funded by Consellería de Educación, Universidade e Formación Profesional of the Xunta de Galicia through the European Regional Development Fund (ERDF) and the Secretaría Xeral de Universidades (Ref. ED431G 2019/01). Álvaro Michelena's research was supported by the Spanish Ministry of Universities (<https://www.universidades.gob.es/>), under the "Formación de Profesorado Universitario" grant with reference FPU21/00932. Martín Bayón's research was supported by Universidad de León, under the "Programa

Propio de Investigación de la Universidad de León 2021” grant. University of Leon. Support for ULE local research projects program. Ref. 2021/00145/001. Internal Code U252.

## DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

## ORCID

Álvaro Michelena  <https://orcid.org/0000-0003-0134-5660>  
 José Aveleira-Mata  <https://orcid.org/0000-0001-5439-0997>  
 Esteban Jove  <https://orcid.org/0000-0002-0625-359X>  
 Martín Bayón-Gutiérrez  <https://orcid.org/0000-0003-1849-796X>  
 Paulo Novais  <https://orcid.org/0000-0002-3549-0754>  
 Oscar Fontenla Romero  <https://orcid.org/0000-0003-4203-8720>  
 José Luis Calvo-Rolle  <https://orcid.org/0000-0002-2333-8405>  
 Héctor Aláiz-Moretón  <https://orcid.org/0000-0001-6572-1261>

## REFERENCES

- Abdalgawad, N., Sajun, A., Kaddoura, Y., Zualkernan, I. A., & Aloul, F. (2022). Generative deep learning to detect cyberattacks for the IoT-23 dataset. *IEEE Access*, 10, 6430–6441. <https://doi.org/10.1109/access.2021.3140015>
- al Enany, M. O., Harb, H. M., & Attiya, G. (2021). A comparative analysis of MQTT and IoT application protocols. *ICEEM 2021 - 2nd IEEE International Conference on Electronic Engineering, Institute of Electrical and Electronics Engineers Inc.*, 1–6. <https://doi.org/10.1109/iceem52022.2021.9480384>
- Alaiz-Moretón, H., Aveleira-Mata, J., Ondicol-García, J., Muñoz-Castañeda, A. L., García, I., & Benavides, C. (2019). Multiclass classification procedure for detecting attacks on mqtt-iot protocol. *Complexity*, 2019, 1–11.
- Ashraf, S., Shawon, M. H., Khalid, H. M., & Muyeen, S. M. (2021). Denial-of-service attack on iec 61850-based substation automation system: A crucial cyber threat towards smart substation pathways. *Sensors*, 21(19) <https://www.mdpi.com/1424-8220/21/19/6415>. <https://doi.org/10.3390/s21196415>
- Aveleira-Mata, J., Muñoz-Castañeda, Á. L., García-Ordás, M. T., Benavides-Cuellar, C., Benítez-Andrades, J. A., & Alaiz-Moretón, H. (2021). IDS prototype for intrusion detection with machine learning models in IoT systems of the industry 4.0. *Dyna (Spain)*, 93(3), 270–275. <https://doi.org/10.6036/10011>
- Baumgartner, C., Böhm, C., Baumgartner, D., Marini, G., Weinberger, K., Olgemöller, B., ... Roscher, A. (2004). Supervised machine learning techniques for the classification of metabolic disorders in newborns. *Bioinformatics*, 20(17), 2985–2996. <https://doi.org/10.1093/bioinformatics/bth343>
- Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4). Springer.
- Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7), 1145–1159. [https://doi.org/10.1016/S0031-3203\(96\)00142-2](https://doi.org/10.1016/S0031-3203(96)00142-2)
- Čekerevac, Z., Dvorak, Z., Prigoda, L., & Ekerevac, P. (2017). Internet of things and the man-in-the-middle attacks—Security and economic risks. *MEST Journal*, 5(2), 15–15. <https://doi.org/10.12709/mest.05.05.02.03>
- Chatzoglou, E., Kambourakis, G., & Kolias, C. (2021). Empirical evaluation of attacks against IEEE 802.11 enterprise networks: The AWID3 dataset. *IEEE Access*, 9, 34188–34205. <https://doi.org/10.1109/access.2021.3061609>
- Chen, F., Huo, Y., Zhu, J., & Fan, D. (2020). A review on the study on MQTT security challenge. *Proceedings - 2020 IEEE international conference on smart cloud, SmartCloud*, 128–133. <https://doi.org/10.1109/SmartCloud49737.2020.00032>
- Da Silva, I. N., Spatti, D. H., Flauzino, R. A., Liboni, L. H. B., & dos Reis Alves, S. F. (2017). *Artificial neural networks* (p. 39). Springer International Publishing. <https://doi.org/10.1007/978-3-319-68612-7>
- Eigner, O., Kreimel, P., & Tavolato, P. (2016). Detection of man-in-the-middle attacks on industrial control networks. *International Conference on Software Security and Assurance (ICSSA)*, 64–69.
- Esfahani, A., Mantas, G., Ribeiro, J., Bastos, J., Mumtaz, S., Violas, M. A., ... Rodriguez, J. (2019). An efficient web authentication mechanism preventing man-in-the-middle attacks in industry 4.0 supply chain. *IEEE Access*, 7, 58981–58989.
- Fatani, A., Dahou, A., Al-Qaness, M. A., Lu, S., & Elaziz, M. A. (2022, dec). Advanced feature extraction and selection approach using deep learning and aquila optimizer for iot intrusion detection system. *Sensors*, 22(1), 140. <https://doi.org/10.3390/s22010140>
- García-Ordás, M. T., Aveleira-Mata, J., Casteleiro-Roca, J.-L., Calvo-Rolle, J. L., Benavides-Cuellar, C., & Alaiz-Moretón, H. (2020). Autoencoder latent space influence on iot mqtt attack classification. In C. Analide, P. Novais, D. Camacho, & H. Yin (Eds.), *Intelligent data engineering and automated learning - Ideal 2020* (pp. 279–286). Springer International Publishing. [https://doi.org/10.1007/978-3-030-62365-4\\_27](https://doi.org/10.1007/978-3-030-62365-4_27)
- Gerganov, R. (2018). Let's build from here. [nfqsd@github.com](https://github.com/nfqsd).
- Gharibian, F., & Ghorbani, A. A. (2007). Comparative study of supervised machine learning techniques for intrusion detection. In *Fifth Annual Conference On Communication Networks And Services Research (CNSR'07)*, 350–358. <https://doi.org/10.1109/CNSR.2007.22>
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). Overview of supervised learning. In *The elements of statistical learning* (pp. 9–41). Springer. [https://doi.org/10.1007/978-0-387-84858-7\\_2](https://doi.org/10.1007/978-0-387-84858-7_2)
- Hindy, H., Bayne, E., Bures, M., Atkinson, R., Tachtatzis, C., & Bellekens, X. (2021). Machine learning based iot intrusion detection system: An mqtt case study (mqtt-ids2020 dataset). In B. Ghita & S. Shiaeles (Eds.), *Selected papers from the 12th international networking conference* (pp. 73–84). Springer International Publishing. [https://doi.org/10.1007/978-3-030-64758-2\\_6](https://doi.org/10.1007/978-3-030-64758-2_6)
- Hintaw, A. J., Manickam, S., Aboalmaaly, M. F., & Karuppayah, S. (2021). *MQTT vulnerabilities, attack vectors and solutions in the internet of things (IoT)*. Taylor & Francis. <https://doi.org/10.1080/03772063.2021.1912651>

- Inayat, U., Zia, M. F., Mahmood, S., Khalid, H. M., & Benbouzid, M. (2022). Learning-based methods for cyber attacks detection in iot systems: A survey on methods, analysis, and future prospects. *Electronics*, 11(9), 1502.
- Khalid, H. M., Muyeen, S. M., & Peng, J. C.-H. (2020). Cyber-attacks in a looped energy-water nexus: An inoculated sub-observer-based approach. *IEEE Systems Journal*, 14(2), 2054–2065. <https://doi.org/10.1109/JSYST.2019.2941759>
- Khalid, H. M., & Peng, J. C.-H. (2016). A bayesian algorithm to enhance the resilience of wams applications against cyber attacks. *IEEE Transactions on Smart Grid*, 7(4), 2026–2037.
- Khalid, M. H., Murtaza, M., & Habbal, M. (2020, nov). Study of security and privacy issues in internet of things. In *Citisia 2020 - iee conference on innovative technologies in intelligent systems and industrial applications, proceedings*. Institute of Electrical and Electronics Engineers Inc. <https://doi.org/10.1109/CITISIA50690.2020.9371828>
- Khraisat, A., & Alazab, A. (2021, dec). A critical review of intrusion detection systems in the internet of things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity*, 4(1), 1–27. <https://doi.org/10.1186/s42400-021-00077-7>
- Kolias, C., Kambourakis, G., Stavrou, A., & Gritzalis, S. (2016, jan). Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Communications Surveys and Tutorials*, 18(1), 184–208. <https://doi.org/10.1109/COMST.2015.2402161>
- Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. *Future Generation Computer Systems*, 100, 779–796. <https://doi.org/10.1016/j.future.2019.05.041>
- Kotsiantis, S. B. (2013). Decision trees: A recent overview. *Artificial Intelligence Review*, 39(4), 261–283.
- Krejčí, R., Hujňák, O., & Švepeš, M. (2018, jan). Security survey of the IoT wireless protocols. *25th Telecommunications Forum, Telfor 2017-Proceedings*, 1–4. <https://doi.org/10.1109/TELFOR.2017.8249286>
- Liu, J., Kantarci, B., & Adams, C. (2020). Machine learning-driven intrusion detection for Contiki-NG-based IoT networks exposed to NSL-KDD dataset. In *Proceedings of the 2nd acm workshop on wireless security and machine learning*. ACM. <https://doi.org/10.1145/3395352.3402621>
- Mahbub, M. (2020). *Progressive researches on IoT security: An exhaustive analysis from the perspective of protocols, vulnerabilities, and preemptive architectonics* (Vol. 168, p. 102761). Academic Press. <https://doi.org/10.1016/j.jnca.2020.102761>
- Malik, P. K., Sharma, R., Singh, R., Gehlot, A., Satapathy, S. C., Alnumay, W. S., Pelusi, D., Ghosh, U., & Nayak, J. (2021). Industrial internet of things and its applications in industry 4.0: State of the art. *Computer Communications*, 166, 125–139. <https://doi.org/10.1016/j.comcom.2020.11.016>
- Mallik, A. (2019). Man-in-the-middle-attack: Understanding in simple words. *Cyberspace: Jurnal Pendidikan Teknologi Informasi*, 2(2), 109. <https://jurnal.ar-raniry.ac.id/index.php/cyberspace/article/view/3453> <https://doi.org/10.22373/cj.v2i2.3453>
- Meena, G., & Choudhary, R. R. (2017). A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA. In *2017 international conference on computer, communications and electronics, comptelix 2017* (pp. 553–558). Institute of Electrical and Electronics Engineers Inc. <https://doi.org/10.1109/comptelix.2017.8004032>
- Mohamed, A. E. (2017). Comparative study of four supervised machine learning techniques for classification. *International Journal of Applied*, 7(2), 1–15.
- Moldagulova, A., & Sulaiman, R. B. (2017). Using knn algorithm for classification of textual documents. *2017 8th International Conference on Information Technology (Icit)*, 665–671. <https://doi.org/10.1109/icitech.2017.8079924>
- Musleh, A. S., Khalid, H. M., Muyeen, S. M., & Al-Durra, A. (2019). A prediction algorithm to enhance grid resilience toward cyber attacks in wams applications. *IEEE Systems Journal*, 13(1), 710–719. <https://doi.org/10.1109/JSYST.2017.2741483>
- Nauman, A., Qadri, Y. A., Amjad, M., Zikria, Y. B., Afzal, M. K., & Kim, S. W. (2020). Multimedia internet of things: A comprehensive survey. *IEEE Access*, 8, 8202–8250. <https://doi.org/10.1109/access.2020.2964280>
- OASIS Open. (2015). MQTT Version 3.1.1. <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html> <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>
- Oconnor, T. J., Jessee, D., & Campos, D. (2021). Through the spyglass: Towards IoT companion app man-in-the-middle attacks. *ACM international conference proceeding series, Association for Computing Machinery*, 58–62. <https://doi.org/10.1145/3474718.3474729>
- OpenWRT. (2020). OpenWrt Project. <https://openwrt.org/https://openwrt.org/start>
- Pallavi, S., & Narayanan, V. A. (2019). An overview of practical attacks on BLE based IOT devices and their security. *5th International Conference On Advanced Computing And Communication Systems, ICACCS 2019, Institute of Electrical and Electronics Engineers Inc.*, 694–698. <https://doi.org/10.1109/icaccs.2019.8728448>
- Parmar, A., Katariya, R., & Patel, V. (2018). A review on random forest: An ensemble classifier. *International conference on intelligent data communication technologies and internet of things*, 758–763. [https://doi.org/10.1007/978-3-030-03146-6\\_86](https://doi.org/10.1007/978-3-030-03146-6_86)
- Peterson, J. M., Leevy, J. L., & Khoshgoftaar, T. M. (2021). A review and analysis of the bot-iot dataset. *IEEE International Conference On Service-Oriented System Engineering (Sose)*, 20–27.
- Rahman, M. A., Asyhari, A. T., Wen, O. W., Ajra, H., Ahmed, Y., & Anwar, F. (2021). Effective combining of feature selection techniques for machine learning-enabled IoT intrusion detection. *Multimedia Tools and Applications*, 80(20), 31381–31399. <https://doi.org/10.1007/s11042-021-10567-y>
- Raikar, M. M., & Meena, S. (2021). Vulnerability assessment of mqtt protocol in internet of things (iot). *2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC)*, 535–540.
- Saha, D., & Manickavasagan, A. (2021). Machine learning techniques for analysis of hyperspectral images to determine quality of food products: A review. *Current Research in Food Science*, 4, 28–44. <https://doi.org/10.1016/j.crfs.2021.01.002>
- Sharma, C., & Gondhi, N. K. (2018). Communication protocol stack for constrained IoT systems. *Proceedings - 2018 3rd International Conference On Internet Of Things: Smart Innovation And Usages, IOT-siu 2018. Institute of Electrical and Electronics Engineers Inc.*, 1–6. <https://doi.org/10.1109/iotsiu.2018.8519904>
- Sharma, N. V., Kavita Aggarwal, G., & Sharma, S. (2021). Performance study of snort and Suricata for intrusion detection system. *IOP Conference Series: Materials Science and Engineering*, 1099(1), 12009. <https://doi.org/10.1088/1757-899x/1099/1/012009>
- Statista. (2022). Number of IoT devices 2015–2025. <https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/>
- Teoh, Y. K., Gill, S. S., & Parlikad, A. K. (2021). Iot and fog computing based predictive maintenance model for effective asset management in industry 4.0 using machine learning. *IEEE Internet of Things Journal*, 10, 2087–2094.
- Ullah, I., & Mahmoud, Q. H. (2021). A framework for anomaly detection in IoT networks using conditional generative adversarial networks. *IEEE Access*, 9, 165907–165931. <https://doi.org/10.1109/access.2021.3132127>

Yang, Z., Liu, X., Li, T., Wu, D., Wang, J., Zhao, Y., & Han, H. (2022). A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Elsevier Advanced Technology*, 116, 102675. <https://doi.org/10.1016/j.cose.2022.102675>

## AUTHOR BIOGRAPHIES

**Álvaro Michelena**, University of A Coruña, CTC, CITIC, Department of Industrial Engineering, Rúa Mendizábal, s/n, 15403 Ferrol, A Coruña, Spain.

**José Aveleira-Mata**, University of León, RIASC: Research Institute of Applied Sciences in Cybersecurity, Vegazana Campus, León, Spain.

**Esteban Jove**, University of A Coruña, CTC, CITIC, Department of Industrial Engineering, Rúa Mendizábal, s/n, 15403 Ferrol, A Coruña, Spain.

**Martín Bayón-Gutiérrez**, University of León, Department of Electrical and Systems Engineering, Vegazana Campus, León, Spain.

**Paulo Novais**, University of Minho, Department of Informatics, ALGORITMI Centre, Gualtar Campus, Braga, Portugal.

**Oscar Fontenla Romero**, University of A Coruña, CITIC Research and Development - Laboratory in Artificial Intelligence (LIDIA), Elviña Campus, A Coruña, Spain.

**José Luis Calvo-Rolle**, University of A Coruña, CTC, CITIC, Department of Industrial Engineering, Rúa Mendizábal, s/n, 15403 Ferrol, A Coruña, Spain.

**Héctor Aláiz-Moretón**, University of León, RIASC: Research Institute of Applied Sciences in Cybersecurity, Vegazana Campus, León, Spain.

**How to cite this article:** Michelena, Á., Aveleira-Mata, J., Jove, E., Bayón-Gutiérrez, M., Novais, P., Romero, O. F., Calvo-Rolle, J. L., & Aláiz-Moretón, H. (2023). A novel intelligent approach for man-in-the-middle attacks detection over internet of things environments based on message queuing telemetry transport. *Expert Systems*, e13263. <https://doi.org/10.1111/exsy.13263>