electronics

MDPI

Article

Intelligent One-Class Classifiers for the Development of an Intrusion Detection System: The MQTT Case Study

Esteban Jove [1], Jose Aveleira-Mata [2], Héctor Alaiz-Moretón [2], José-Luis Casteleiro-Roca [1], David Yeregui Marcos del Blanco [3,*], Francisco Zayas-Gato [1], Héctor Quintián [1] and José Luis Calvo-Rolle [1]

[1] Department of Industrial Engineering, University of A Coruña, CTC, CITIC, Avda. 19 de Febrero s/n, 15405 Ferrol, Spain; esteban.jove@udc.es (E.J.); jose.luis.casteleiro@udc.es (J.-L.C.-R.); f.zayas.gato@udc.es (F.Z.-G.); hector.quintian@udc.es (H.Q.); jose.rolle@udc.es (J.L.C.-R.)
[2] Research Institute of Applied Sciences in Cybersecurity (RIASC) MIC, Universidad de León, 24071 León, Spain; jose.aveleira@unileon.es (J.A.-M.); hector.moreton@unileon.es (H.A.-M.)
[3] Department of Mechanical, Computer and Aerospace Engineering, University of León, 24071 León, Spain
* Correspondence: dmarcb01@estudiantes.unileon.es

Abstract: The ever-increasing number of smart devices connected to the internet poses an unprecedented security challenge. This article presents the implementation of an Intrusion Detection System (IDS) based on the deployment of different one-class classifiers to prevent attacks over the Internet of Things (IoT) protocol Message Queuing Telemetry Transport (MQTT). The utilization of real data sets has allowed us to train the one-class algorithms, showing a remarkable performance in detecting attacks.

Keywords: one-class; K-means; IoT; PCA; MQTT; APE; SVM; NCBoP

Citation: Jove, E.; Aveleira-Mata, J.; Alaiz-Moretón, H.; Casteleiro-Roca, J.-L.; Marcos del Blanco, D.Y.; Zayas-Gato, F.; Quintián, H.; Calvo-Rolle, J.L. Intelligent One-Class Classifiers for the Development of an Intrusion Detection System: The MQTT Case Study. *Electronics* **2022**, *11*, 422. https://doi.org/10.3390/electronics11030422

Academic Editor: Ping-Feng Pai

Received: 21 December 2021
Accepted: 26 January 2022
Published: 30 January 2022

1. Introduction

The "Internet of Things" (IoT) refers to any technology implementation, including a set of smart devices interconnected to the internet, interacting with external systems through information and data exchange. Currently, there are over 5 billion connected IoT devices [1], according to the previous definition. One of the most relevant applications of IoT is in, what is usually referred to as, Industry 4.0 [2]. It allows for real-time management of automated controlled systems through remote monitoring via a client device, such as a Smartphone, tablet, or PC. Additionally, by collecting and analysing data and information with cloud processing techniques, it is possible to achieve more complex interactions amongst the different elements of the IoT system [3,4].

IoT devices are usually quite affordable, but they usually present a rather limited computation capacity, not being feasible to implement cybersecurity primitives at a device level. Therefore, IoT devices tend to be fast and efficient but with limited resilience against network attacks.

IoT systems have traditionally been a target for attacks. They have been used in botnets, such as the Mirai attack in September 2016, in which 400,000 IoT devices were infected and eventually performed a massive DDoS attack. Another notable example took place in April 2020, when the Dark Nexus botnet, based on the Mirai code, compromised over 1350 IoT devices [5].

Some of the most widely used IoT protocols are: Bluetooth [6], zigbee [7], and LORA [8] in the physical/link layer and CoAP and MQTT [9–11] for the application layer.

To improve security in IoT systems, one of the most popular approaches is to use an intrusion detection system (IDS). This method is based on monitoring network traffic and, therefore, it does not require high computing capacity at a device level. In addition, the IDS does not require any change in the configuration in the existing IoT systems.

IDS can be signature-based or anomaly detection-based. Signature-based detection methods, using predefined rules, are effective in detecting attacks for previously known behaviours. On the other hand, anomaly-based identification is used in order to detect unknown attacks or attacks with patterns that are not clearly defined. In order to do so, the IDS is monitoring the entire system, comparing anomalous traffic with predefined behaviours assumed to be normal through previously trained artificial intelligence models. The most resource-consuming activities are performed offline, allowing for an overall smooth, efficient IDS activity.

One quite common approach to increase the security of an IoT network is the utilization of patterns of previous attacks within an IDS [12–15].

In particular, the two main methods for an IDS introduction are: attack recognition via real-time status monitoring and juxtaposition with the previous normal value [16] or signature-based pattern recognition in the network data flow [17]. In the latter, the implementation of a detection model is mandatory. This action is usually performed by introducing machine learning algorithms, such as Support Vector Machine and Random Forest [18,19] or clustering techniques [20].

A supplementary approach involves deep learning methods, such as auto-encoders and Deep Belief Networks (DBNs) due to their dimension-reduction capabilities in optimal classification models [21–23]. For thread detection modelling, more sophisticated techniques, such as Long Short Term Memory (LSTM), have been proved to be a valid solution [24,25].

More recent research lines have implemented the NSL-KDD dataset (enhanced version ofKDD99) to incorporate Remote to Local (R2L) and User to Root (U2R) attacks on IoT systems. Specifically, it introduces two-level classification algorithms using a Bayesian network and the K-Means method [26]. Another current study introduces the NSL-KDD dataset in an IDS, using Kontiki. In this environment, a network of IoT devices is simulated using the MQTT protocol [27]. Finally, in the AWID dataset, attack detection through Machine Learning is proposed in wireless IoT environments (WIDS) [3,28,29].

Regarding open source IDS systems, Snort and Suricata [30] are the most widely implemented to address IoT system security. In the present paper, the Snort 3.0 version, launched in 2021, is selected to be compared to the proposed IDS.

The purpose of the article is to improve security IoT environments, specifically in those presenting certain characteristics (in terms of computing capacity and protocols), making them a likely target for attacks (i.e., botnets). To that end, we propose the implementation of an IDS, introducing one-class classifiers using the MQTT protocol. This approach should constitute a viable solution since it analyses the network traffic without altering the system configuration or demanding additional computing capacity. The network analysis of the MQTT protocol can be used to prevent intrusion attacks through non-legitimate clients. This vulnerability is included in RFC5246 [31], "Communications could be intercepted, altered, re-routed or disclosed", which also prevents an attacker from performing code injections to alter the operation of the system.

The final factor to develop a truly applicable model is the utilisation of reliable datasets [32], such as e KDD99 [33], NSL-KDD [34], and AWID [35] for TCP/IP, containing network data of this protocol. Therefore, the present article also details how to create a dataset based on MQTT.

The main goal of this research is the development of an IDS with a new machine learning approach, complementing the traditional, signature-based method. The one-class machine Learning algorithm is a state-of-the-art technique previously presented in [36].

Regarding the composition of this article, Section 2 introduces the case study, while Section 3 refers to the one-class classifier outlook. Finally, Section 4 explains the experiments and results and Section 5 presents both the conclusions and some future lines of work.

2. Case Study

MQTT is a messaging protocol specifically designed for light machine-to-machine (M2M) communications, making it very suitable for connecting small devices to networks with limited bandwidth. The MQTT protocol is widely used in IoT [37] and industry [2] environments.

The architecture of a MQTT system follows a star topology [38], with a central node or "broker" acting as a server. The broker is responsible for managing the network and transmitting the messages in real time. The communication protocol is based on topics created by the client publishing the message and the receiving node(s) subscribing to the topic. Therefore, it allows for both one-to-one or one-to-many communications.

As previously mentioned, the rather limited computational capability of the devices in a MQTT system makes it vulnerable, which could lead to attackers monitoring and even affecting the normal activity.

One of the most common ways in which an attacker gains access to the system is by using the Shodan network scan on the default port 1883 [39]. Subsequently, and once inside the system, the invader can proceed to "sniff" the data packages and identify the plain text password [40]. Once obtained, it can gain access to the system, scan the broker messages (identified by the use of the '#' character), and ultimately manipulate the different topics.

For this article, the following IoT system implementing the MQTT protocol is defined:

- Actuators and sensors: Comprised of two integrated boards NodeMCU, including a low-power micro controller connected to a wireless network via a ESP8266 chip [41]. The NodeMCU chip is connected to a HC-SR04 ultrasonic sensor, which subscribes to the topic "distance/ultrasonic1", where it publishes the distance to any element in front of it, up to a range of 40 centimetres. The other NodeMCU is connected to an actuator consisting of a relay that turns a desk light on and off, subscribes to the topic "light/relay", and, depending on the value, changes the state to "0" off, "1" on.
- The server: Developed in node.js due to its efficiency in controlling multiple and simultaneous connections, with the npm package manager, which has installed the "Aedes' 'library [42], with which a MQTT broker server has been programmed. The server also hosts the client web application.
- Web application:Developed using angular.js, which connects to the broker as another client with the angular-MQTT library. The difference is that it implements web sockets instead of the MQTT protocol for the communication with the broker. The web application has an interface that shows the status of the different devices and allows for interaction with them in real time.
- System clients: A PC and a smartphone interacting with the sensors through the web app connected via WiFi while generating network traffic.

In the present article, the intrusion was conducted using the mosquito software from a client, apart from the IoT infrastructure [43]. By means of a distinctive symbol, the topics of the system were unveiled. Subsequently, the sensor was targeted to alter the associated temperature and the actions on the actuator. In order to produce additional random frames, a power shell was implemented. Finally, a router customized with OpenWRT [44] received the traffic from the regular traffic, the IoT system under attack, and the internet navigation in a PCAP file.

For file management purposes, the traffic included in the PCAP files was separated, taking into account the fields in the MQTT protocol, together with the common fields for every frame (ports, IP locations, and the time code according to the AWID data set). Additionally, a tag was assigned to each frame in order to mark whether it was under attack or not. The output was a data set in csv format with a total of 80.893 frames. A total of 78.995 frames (97.65%) were found to be normal, while 1.898 (2.35%) presented irregularities. Figure 1 introduces the steps for obtaining the data set, with the elements of the WLAN environment, as previously described. In this case, the system was attacked from

another computer with intrusion attacks against the server. All the traffic was collected, dissected, and tagged by the router to generate the data set in CSV format.

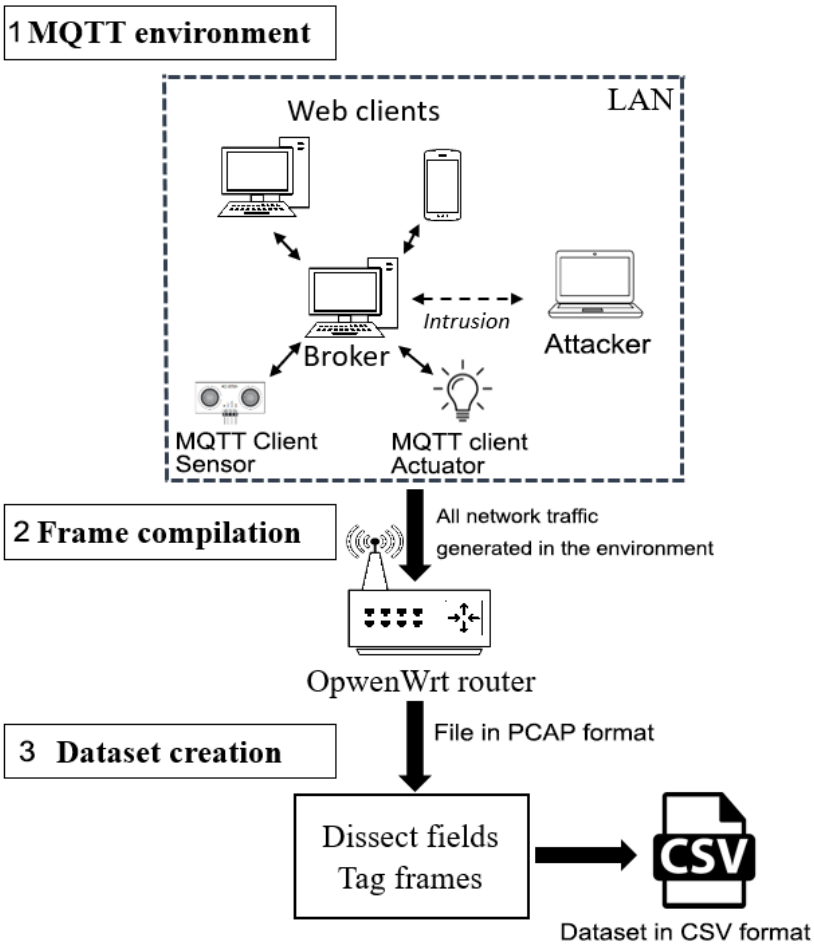

Figure 1. Intrusion dataset setup.

3. Intrusion Detection Classifier

In order to address the security problems associated with the MQTT protocol, a solution based on the one-class technique was introduced.

3.1. Classifier Approach

In order to implement a one-class classifier, it is necessary to have prior knowledge of which conditions under the MQTT environment are working correctly. The followed steps are:

- The target set comprised only of legitimate samples is divided into 10 random groups:
- for $i = 1{:}1{:}10$
  - All groups except the ith are used to train the classifier. An example of this process is shown in Figure 2.
  - Once the training stage has finished, the group i, together with all the non-target samples, are used to validate the classifier. An example of this process is shown in Figure 3.
- The mean value from the 10 iterations is used as a measure of the classifier performance.
- Finally, the classifier's performance is best selected and trained with all the target samples.

Figure 2. One-class classifier implementation.

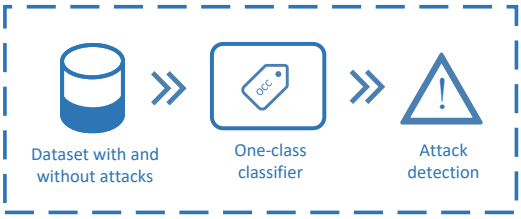

Figure 3. One-class classifier test stage.

3.2. Methods

In this section, different sets of anomaly detection techniques are introduced.

3.2.1. Approximate Convex Hull

An Approximate Convex Hull (ACH) refers to a one-class classification technique of the boundary subset, with a very positive track record of practical implementations [45]. The underlying core concept is to obtain a reliable approximation to the limits of a certain data set $S \in \mathbb{R}^n$; hence the "boundary" denomination. This is achieved by calculating the convex limits. Considering that the typical convex hulls of S with N samples and d variables implies a computational cost of $O(N^{(d/2)+1})$ [45], it is computationally advisable to opt for a reliable enough approximation. Therefore, p random projections of the hull are generated over $2D$ planes to subsequently calculate their respective convex boundaries [46], reducing the overall computational cost, as presented in the following Figure 4.

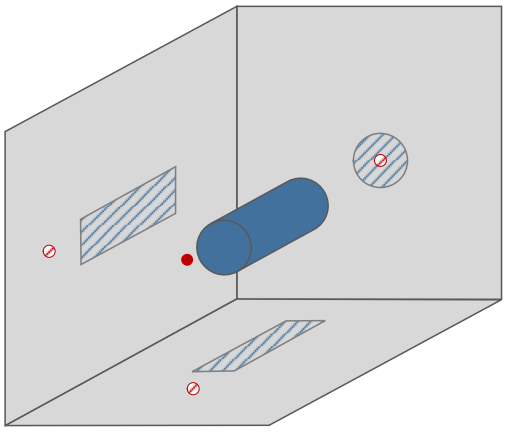

Figure 4. Representative example of Convex Hull Approximation.

With the approximation modelling completed with the p projections, a new data set was considered an anomaly when it surpassed the convex hull for any of the generated projections. Additionally, adding an expansion factor α, by which the convex limits were enlarged or contracted from the centroid of each projection, increased the adaptability to different typologies of data sets. An α over 1 means that the limits are expanded, while under 1 implies narrowed boundaries.

3.2.2. Non-Convex Boundary over Projections

The Non Convex Boundary over Projections (NCBoP) technique relies on concepts akin to those introduced in the previous section and Figure 4, but it produces remarkably improved results for non-convex data sets [36]. In the NCBoP, the limits are calculated via non-convex limits, therefore eliminating false positives occurring if an anomaly presents within the the convex hull. As for the previous method (ACH), it is possible to introduce an α factor to avoid over- or under-fitting [36].

Figure 5 shows the difference between the convex hull and non-convex hull calculation for a given dataset in $\mathbb{R}^2$.

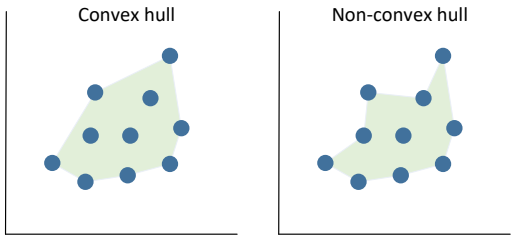

Figure 5. Difference between convex and non-convex hull.

One caveat worth mentioning for NCBoP is that the computational cost is higher than in ACH. Therefore, the decision regarding which one to go with should be based on the complexity and/or shape of each dataset to be processed.

3.2.3. K-Means

The non supervised family of learning algorithm, named K-Means, is well known for clustering purposes [47,48]. This procedure defines a set of groups for the initial data set based on the number of groups selected by the user. The algorithm uses the total sum of distances from each cluster centroid to every point.

The training set is utilized for calculating the centroid of each group. If the distance of a certain test data to its closest centroid is minimum, K-means can be considered a one-class technique. Therefore, the anomaly can be detected when the distance is higher than the distance of every cluster data to the centroid.

An example where the training set is divided in two clusters is presented in the following Figure 6. In this case, a test point, represented by a green dot, is labelled as a target because the distance to the nearest centroid (black star of Cluster #2) is lower than that of many others in the training samples.

Figure 6. Graphical representation of a sample labelling using K-means.

3.2.4. Principal Component Analysis

Principal Component Analysis (PCA) is a common technique oriented at reducing the dimension when this factor represents a potential threat [49,50]. In addition, PCA can provide a good solution for detecting anomalies and solving classification-related problems [51,52].

The PCA algorithm is based on the eigenvectors of the co-variance matrix for calculating the directions, where the set of data has higher variability. Upon definition of these directions, they are known as principal components. Subsequently, they are used for linear projections with fewer dimensions. If the criteria chosen is based on the distance between the data projected and the primitive data, a reconstruction error value is obtained as a one-class procedure. By doing so, if the reconstruction error of the test data is larger than the value obtained in the training process, an anomaly has been identified.

The following Figure 7 introduces an example of how the distance from a test point to its projection is greater than all the distances of the training points to their respective projections. In this case, only the first component is used. The limit between normal and anomalous behaviour is commonly related to the training distance percentile.

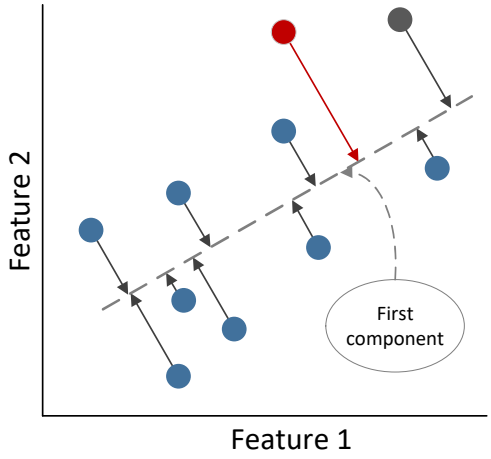

Figure 7. Representation of PCA for one-class.

3.2.5. One-Class Support Vector Machine

One of the most typical algorithms for anomaly detection is the Support Vector Machine (SVM) for one-class (OCSVM) tasks due to the high-quality outcome for various applications [53,54].

The OCSVM is a supervised-learning, procedure-mapping method supported by a kernel function. Subsequently, an hyper-plane able to maximize the distance between the origin and the mapped points is defined. Consequently, the instances near the hyper plane are considered the support vectors.

When the training procedure is completed and a new data set enters the classifier, it returns the distance from the high dimensional plane to the input data. Any negative result is flagged as an anomaly.

4. Experiments and Results

4.1. Experiments Setup

The following subsection introduces the set of experiments implemented for this article.

4.1.1. Techniques Configuration

In order to evaluate the performance of each of the five techniques presented so far, the following benchmarks have been set:

- ACH and NCBoP: Values for projections checked [10, 50, 100 , 500].
  Values for the α parameter in order to further evaluate restriction limits: [0.8, 0.9, 1, 1.1, 1].
- K-means: Number of clusters to divide the dataset into [1–15].
  Values defined for the outlier percentage in the training data: [0, 5, 10, 15, 20, 25].
- PCA: The amount of main components ranges from 1 to $n - 1$, with n being the number of variables in the training set. The K-means algorithm outlier fraction is taken into consideration as well.
- SVM: An outlier percentage is also considered, similarly to the K-means and PCA cases.

4.1.2. Data Pre-Processing

In order to improve the results, the dataset was normalized, ranging from 0 to 1 while introducing the z-score method [55]. Additionally, the categorical variables were converted into numerical values. On top of that, a data set without pre-processing was tested for complementary reasons.

4.1.3. Performance Measurement

In order to assess the output for each classifier, the "Area Under the Receiver Operating Characteristic Curve" (AUC) [56] was the selected measure. This measure represents the probability for a random positive sample of being labelled as positive. Furthermore, in contrast to other variables, such as sensitivity, precision, or recall, AUC is not sensitive to class distribution, which is a significant advantage, especially in one-class problems [57]. As an additional item, the required training time for each classifier was also introduced as an indicator for the associated computational cost. Finally, the benchmark tests were confirmed with a $k\text{-}fold$ cross-validation test, considering $k = 10$.

4.2. Results

The experiments presented so far have led to the results shown in Table 1 for Intrusion Detection and Table 2 for MiTM events, respectively. For each technique, the configuration presenting a stronger AUC is selected and represented.

Table 1. Best AUC results for intrusion detection attacks for each technique.

Technique	Features	Normalization	AUC (%)	Time (s)
ACH	500 projections $\alpha = 1$	Zscore	52.063	3.11
NCBoP	500 projections $\alpha = 1$	No	60.317	1211.73
K-means	Clusters = 10 Outlier Fraction = 15%	No	76.496	0.84
PCA	Components = 15 Outlier Fraction = 20%	0-1	89.296	0.21
SVM	Outlier Fraction = 0%	No	69.235	285.56

With the default set of rules, Snort can detect DoS attacks but not intrusions. Therefore, it is necessary to create a specific directive at the byte level to screen and discover a potential use for the special character "#" by an external client. Unfortunately, this approach is vulnerable if the attacker changes its network configuration. Conversely, the proposed IDS can detect intrusion attacks using the proposed model without any additional configuration required.

Table 2. Best AUC results for MiTM attack for each technique.

Technique	Features	Normalization	AUC (%)	Time (s)
ACH	100 projections $\alpha = 1$	No	89.564	0.33
NCBoP	500 projections $\alpha = 1$	No	90.317	2211.73
K-means	Clusters = 6 Outlier Fraction = 10%	No	91.696	0.48
PCA	Components = 8 Outlier Fraction = 10%	ZScore	92.628	0.27
SVM	Outlier Fraction = 10%	0–1	79.588	313.08

As per deployment easiness, it implements the XGBoost algorithm. Since it introduces GPU processes, the overall implementation becomes more complex and computationally costly compared to previous works [24].

5. Conclusions and Futures Works

The results have revealed that the best overall technique for MiTM detection is PCA, simultaneously presenting an AUC over 89% and the shortest training time. In particular, the highest performance level was obtained with eight components, Zscore normalization, and an outlier fraction of 10% in the training set. Similar conclusions were reached with the IDS, although the AUC values obtained were slightly lower in comparison. As stated above, the training time is a significantly important parameter in machine learning tasks. Therefore, K-means and PCA algorithms are suitable for these kinds of high-dimensional data sets. In other cases, NCBoP and SVM might be acceptable, since their associated AUC performance is still acceptable.

Under such an assumption, the implementation of the proposed approach would be presented as a very promising tool to monitor the potential appearance of two different types of attacks within an MQTT protocol environment. This could have a very positive, direct impact on the network performance, helping to increase the cybersecurity protection level.

For future works, additional one-class techniques could be applied in order to further improve the performance of the anomaly detection system. In particular, dimension reduction techniques, such as autoencoder, could be specially promising. Finally, the introduction of unsupervised techniques, as opposed to semi-supervised techniques, could prove extremely useful in the future. To that end, clustering techniques might play a significant role, which is worth being further studied. Finally, the application of the presented one-class techniques to new IoT datasets, such as Constrained Application Protocol (CoAP), could provide valuable sources of information against currently undetected attacks.

Author Contributions: Conceptualization, E.J.; Data curation, H.A.-M.; Formal analysis, J.A.-M.; Investigation, J.-L.C.-R.; Methodology, D.Y.M.d.B. and J.L.C.-R.; Project administration, D.Y.M.d.B.; Software, F.Z.-G.; Supervision, J.L.C.-R.; Validation, H.Q.; Writing—original draft, E.J., J.A.-M., H.A.-M., J.-L.C.-R., D.Y.M.d.B., F.Z.-G., H.Q. and J.L.C.-R. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by the Spanish National Cybersecurity Institute (INCIBE) and the Research Institute of Applied Sciences in Cybersecurity (RIASC).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Green, J. The Internet of Things Reference Model. In *Internet of Things World Forum*; CISCO: San Jose, CA, USA, 2014; pp. 1–12.
2. Ramamoorthy, K.; Karthikeyan, S.; Chelladurai, T. An investigation on Industrial Internet of Things for Mission Critical things in Industry 4.0.2. Literature Review. *Seybold Rep.* **2020**, *15*, 3294–3300.
3. M. Wollschlaeger, T.S.y.J.J. The Future of Industrial Communication. *IEEE Ind. Electron. Mag.* **2017**, *11*, 17–27. [CrossRef]
4. Jove, E.; Casteleiro-Roca, J.L.; Quintián, H.; Méndez-Pérez, J.A.; Calvo-Rolle, J.L. Virtual Sensor for Fault Detection, Isolation and Data Recovery for Bicomponent Mixing Machine Monitoring. *Informatica* **2019**, *30*, 671–687. [CrossRef]
5. Hamid, H.; Noor, R.M.; Omar, S.N.; Ahmedy, I.; Anjum, S.S.; Shah, S.A.A.; Kaur, S.; Othman, F.; Tamil, E.M. IoT-based botnet attacks systematic mapping study of literature. *Scientometrics* **2021**, *126*, 2759–2800. [CrossRef]
6. Al-sarawi, S.; Anbar, M.; Alieyan, K.; Alzubaidi, M. Internet of Things (IoT) Communication Protocols: Review. In Proceedings of the 2017 8th International Conference on Information Technology (ICIT), Amman, Jordan, 17–18 May 2017; pp. 685–690.
7. Alobaidy, H.A.H.; Mandeep, J.S.; Nordin, R.; Abdullah, N.F. A Review on ZigBee Based WSNs: Concepts, Infrastructure, Applications, and Challenges. *Int. J. Electr. Electron. Eng. Telecommun.* **2020**, *9*, 189–198. [CrossRef]
8. Zorbas, D.; Abdelfadeel, K.; Kotzanikolaou, P.; Pesch, D. TS-LoRa: Time-slotted LoRaWAN for the Industrial Internet of Things. *Comput. Commun.* **2020**, *153*, 1–10. [CrossRef]
9. Razzaq, M.A.; Gill, S.H.; Qureshi, M.A.; Ullah, S. Security Issues in the Internet of Things (IoT): A Comprehensive Study. *Int. J. Adv. Comput. Sci. Appl.* **2017**, *8*, 383. [CrossRef]
10. Ordóñez Galán, C.; Sánchez Lasheras, F.; de Cos Juez, F.J.; Bernardo Sánchez, A. Missing Data Imputation of Questionnaires by Means of Genetic Algorithms with Different Fitness Functions. *J. Comput. Appl. Math.* **2017**, *311*, 704–717. [CrossRef]
11. Kolias, C.; Kambourakis, G.; Stavrou, A.; Voas, J. DDoS in the IoT: Mirai and other botnets. *Computer* **2017**, *50*, 80–84. [CrossRef]
12. Hamed, T.; Ernst, J.B.; Kremer, S.C. A Survey and Taxonomy of Classifiers of Intrusion Detection Systems. In *Computer and Network Security Essentials*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 21–39.
13. Perdisci, R.; Ariu, D.; Fogla, P.; Giacinto, G.; Lee, W. McPAD: A multiple classifier system for accurate payload-based anomaly detection. *Comput. Netw.* **2009**, *53*, 864–881. [CrossRef]
14. Zhou, Q.; Pezaros, D. Evaluation of machine learning classifiers for Zero-Day intrusion detection—An analysis on CIC-AWS-2018 dataset. *arXiv* **2019**, arXiv:1905.03685.
15. Ben-Asher, N.; Gonzalez, C. Effects of cyber security knowledge on attack detection. *Comput. Hum. Behav.* **2015**, *48*, 51–61. [CrossRef]
16. Prabha, K.; Sudha, S. A Survey on IPS Methods and Techniques. *Int. J. Comput. Sci. Issues* **2016**, *13*, 38–43. [CrossRef]
17. Samrin, R.; Vasumathi, D. Review on anomaly based network intrusion detection system. In Proceedings of the International Conference on Electrical, Electronics, Communication Computer Technologies and Optimization Techniques, ICEECCOT 2017, Mysuru, India, 15–16 December 2018; pp. 141–147. [CrossRef]
18. Hasan, M.; Nasser, M.; Pal, B.; Ahmad, S. Support Vector Machine and Random Forest Modeling for Intrusion Detection System (IDS). *J. Intell. Learn. Syst. Appl.* **2014**, *2014*, 45–52. [CrossRef]
19. Nieto, P.G.; Fernández, J.A.; Lasheras, F.S.; de Cos Juez, F.; Muñiz, C.D. A new improved study of cyanotoxins presence from experimental cyanobacteria concentrations in the Trasona reservoir (Northern Spain) using the MARS technique. *Sci. Total Environ.* **2012**, *430*, 88–92. [CrossRef]
20. Chakrabarty, B.; Chanda, O.; Saiful, M. Anomaly based Intrusion Detection System using Genetic Algorithm and K-Centroid Clustering. *Int. J. Comput. Appl.* **2017**, *163*, 13–17. [CrossRef]
21. Tao, X.; Kong, D.; Wei, Y.; Wang, Y. A Big Network Traffic Data Fusion Approach Based on Fisher and Deep Auto-Encoder. *Information* **2016**, *7*, 20. [CrossRef]
22. Li, Y.; Ma, R.; Jiao, R. A hybrid malicious code detection method based on deep learning. *Int. J. Secur. Its Appl.* **2015**, *9*, 205–216. [CrossRef]
23. Nieto, P.G.; Torres, J.M.; de Cos Juez, F.J.; Lasheras, F.S. Using multivariate adaptive regression splines and multilayer perceptron networks to evaluate paper manufactured using Eucalyptus globulus. *Appl. Math. Comput.* **2012**, *219*, 755–763.
24. Alaiz-Moreton, H.; Aveleira-Mata, J.; Ondicol-Garcia, J.; Muñoz-Castañeda, A.L.; García, I.; Benavides, C. Multiclass Classification Procedure for Detecting Attacks on MQTT-IoT Protocol. *Complexity* **2019**, *2019*, 6516253. [CrossRef]
25. Kim, J.; Kim, J.; Thu, H.L.T.; Kim, H. Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection. In Proceedings of the 2016 International Conference on Platform Technology and Service (PlatCon), Jeju, Korea, 15–17 February 2016; pp. 1–5. [CrossRef]
26. Pajouh, H.H.; Javidan, R.; Khayami, R.; Dehghantanha, A.; Choo, K.K.R. A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks. *IEEE Trans. Emerg. Top. Comput.* **2019**, *7*, 314–323. [CrossRef]
27. Liu, J.; Kantarci, B.; Adams, C. Machine learning-driven intrusion detection for Contiki-NG-based IoT networks exposed to NSL-KDD dataset. In Proceedings of the WiseML 2020-Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning, Virtual, 13 July 2020; pp. 25–30. [CrossRef]
28. Thakkar, A.; Lohiya, R. *A Review on Machine Learning and Deep Learning Perspectives of IDS for IoT: Recent Updates, Security Issues, and Challenges*; Springer: Berlin/Heidelberg, Germany, 2020. [CrossRef]

29. Da Costa, K.A.; Papa, J.P.; Lisboa, C.O.; Munoz, R.; de Albuquerque, V.H.C. Internet of Things: A survey on machine learning-based intrusion detection approaches. *Comput. Netw.* **2019**, *151*, 147–157. [CrossRef]
30. Alsakran, F.; Bendiab, G.; Shiaeles, S.; Kolokotronis, N. Intrusion Detection Systems for Smart Home IoT Devices: Experimental Comparison Study. In *Communications in Computer and Information Science*; Springer: Singapore, 2020; Volume 1208 CCIS, pp. 87–98. [CrossRef]
31. *OASIS (Organization for the Advancement of Structured Information Standards)*. MQTT Version 3.1.1; 2014. Avaible online: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html (accessed on 1 December 2021).
32. Bhuyan, M.H.; Bhattacharyya, D.K.; Kalita, J.K. Towards generating real-life datasets for network intrusion detection. *Int. J. Netw. Secur.* **2015**, *17*, 683–701.
33. Stolfo, S.J. KDD Cup 1999 Dataset. 1999. Avaible online: https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed on 1 December 2021).
34. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
35. Kolias, C.; Kambourakis, G.; Stavrou, A.; Gritzalis, S. Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 184–208. [CrossRef]
36. Jove, E.; Casteleiro-Roca, J.L.; Quintián, H.; Méndez-Pérez, J.A.; Calvo-Rolle, J.L. A new method for anomaly detection based on non-convex boundaries with random two-dimensional projections. *Inf. Fusion* **2021**, *65*, 50–57. [CrossRef]
37. Sethi, P.; Sarangi, S.R. Internet of Things: Architectures , Protocols , and Applications. *J. Electr. Comput. Eng.* **2017**, *2017*, 9324035. [CrossRef]
38. Gupta, A.B.R. MQTT version 3.1.1. In *OASIS Standard*; 2014. Avaible online: https://www.oasis-open.org/ (accessed on 1 December 2021).
39. Andy, S.; Rahardjo, B.; Hanindhito, B. Attack scenarios and security analysis of MQTT communication protocol in IoT system. In Proceedings of the 2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Yogyakarta, Indonesia, 19–21 September 2017; pp. 1–6. [CrossRef]
40. Dinculeană, D.; Cheng, X. Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices. *Appl. Sci.* **2019**, *9*, 848. [CrossRef]
41. NodeMCU. NodeMCU—An Open-Source Firmware Based on ESP8266 Wifi-Soc., 2014. Avaible online: https://nodemcu.readthedocs.io/en/release/ (accessed on 1 December 2021).
42. aedes. GitHub-moscajs/aedes: Barebone MQTT Broker That Can Run on any Stream Server, the Node Way. Avaible online: https://github.com/moscajs/aedes (accessed on 1 December 2021).
43. Light, R.A. Mosquitto: Server and client implementation of the MQTT protocol. *J. Open Source Softw.* **2017**, *2*, 265. [CrossRef]
44. openwrt. openwrt.org. Avaible online: https://openwrt.org/ (accessed on 1 December 2021).
45. Casale, P.; Pujol, O.; Radeva, P. Approximate convex hulls family for one-class classification. In *International Workshop on Multiple Classifier Systems*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 106–115.
46. Jove, E.; Casteleiro-Roca, J.L.; Quintián, H.; Simić, D.; Méndez-Pérez, J.A.; Luis Calvo-Rolle, J. Anomaly detection based on one-class intelligent techniques over a control level plant. *Logic J. IGPL* **2020**, *28*, 502–518. [CrossRef]
47. Jove, E.; López, J.A.V.; Fernández-Ibáñez, I.; Casteleiro-Roca, J.L.; Calvo-Rolle, J.L. Hybrid intelligent system to predict the individual academic performance of engineering students. *Int. J. Eng. Educ.* **2018**, *34*, 895–904.
48. Jove, E.; Casteleiro-Roca, J.L.; Quintián, H.; Zayas-Gato, F.; Vercelli, G.; Calvo-Rolle, J.L. A One-class Classifier Based on a Hybrid Topology to Detect Faults in Power Cells. *Log. J. IGPL* **2021**, *13*, 801. [CrossRef]
49. Wu, J.; Zhang, X. A PCA classifier and its application in vehicle detection. In Proceedings of the IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222), Washington, DC, USA, 15–19 July 2001; Volume 1, pp. 600–604.
50. Jove, E.; Casteleiro-Roca, J.L.; Quintián, H.; Méndez-Pérez, J.A.; Calvo-Rolle, J.L. A fault detection system based on unsupervised techniques for industrial control loops. *Expert Syst.* **2019**, *36*, e12395. [CrossRef]
51. Jove, E.; Casteleiro-Roca, J.L.; Quintián, H.; Méndez-Pérez, J.A.; Calvo-Rolle, J.L. Anomaly detection based on intelligent techniques over a bicomponent production plant used on wind generator blades manufacturing. *Rev. Iberoam. Autom. Inform. Ind.* **2020**, *17*, 84–93. [CrossRef]
52. Jove, E.; Aláiz-Moretón, H.; Casteleiro-Roca, J.L.; Corchado, E.; Calvo-Rolle, J.L. Modeling of bicomponent mixing system used in the manufacture of wind generator blades. In *International Conference on Intelligent Data Engineering and Automated Learning*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8669, pp. 275–285.
53. Li, K.L.; Huang, H.K.; Tian, S.F.; Xu, W. Improving one-class SVM for anomaly detection. In Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693), Xi'an, China, 5 November 2003; Volume 5, pp. 3077–3081.
54. Quintián, H.; Corchado, E. Beta scale invariant map. *Eng. Appl. Artif. Intell.* **2017**, *59*, 218–235. [CrossRef]
55. Shalabi, L.A.; Shaaban, Z. Normalization as a Preprocessing Engine for Data Mining and the Approach of Preference Matrix. In Proceedings of the 2006 International Conference on Dependability of Computer Systems, Szklarska Poreba, Poland, 25–27 May 2006; pp. 207–214. [CrossRef]

56. Bradley, A.P. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognit.* **1997**, *30*, 1145–1159. [CrossRef]
57. Fawcett, T. An introduction to ROC analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [CrossRef]