electronics

MDPI

Article

Machine Learning for DDoS Attack Detection in Industry 4.0 CPPSs

Firooz B. Saghezchi [1,*], Georgios Mantas [1,2], Manuel A. Violas [3], A. Manuel de Oliveira Duarte [3] and Jonathan Rodriguez [1,4]

[1] Instituto de Telecomunicações, University of Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal; gimantas@av.it.pt (G.M.); jonathan@av.it.pt (J.R.)
[2] Faculty of Engineering and Science, University of Greenwich, Chatham Maritime ME4 4TB, UK
[3] Department of Electronics, Telecommunications and Informatics, University of Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal; manuelv@ua.pt (M.A.V.); duarte@ua.pt (A.M.d.O.D.)
[4] Faculty of Computing, Engineering and Science, University of South Wales, Pontypridd CF37 1DL, UK
* Correspondence: firooz@av.it.pt

Abstract: The Fourth Industrial Revolution (Industry 4.0) has transformed factories into smart Cyber-Physical Production Systems (CPPSs), where man, product, and machine are fully interconnected across the whole supply chain. Although this digitalization brings enormous advantages through customized, transparent, and agile manufacturing, it introduces a significant number of new attack vectors—e.g., through vulnerable Internet-of-Things (IoT) nodes—that can be leveraged by attackers to launch sophisticated Distributed Denial-of-Service (DDoS) attacks threatening the availability of the production line, business services, or even the human lives. In this article, we adopt a Machine Learning (ML) approach for network anomaly detection and construct different data-driven models to detect DDoS attacks on Industry 4.0 CPPSs. Existing techniques use data either artificially synthesized or collected from Information Technology (IT) networks or small-scale lab testbeds. To address this limitation, we use network traffic data captured from a real-world semiconductor production factory. We extract 45 bidirectional network flow features and construct several labeled datasets for training and testing ML models. We investigate 11 different supervised, unsupervised, and semi-supervised algorithms and assess their performance through extensive simulations. The results show that, in terms of the detection performance, supervised algorithms outperform both unsupervised and semi-supervised ones. In particular, the Decision Tree model attains an Accuracy of 0.999 while confining the False Positive Rate to 0.001.

Keywords: Industry 4.0; cybersecurity; intrusion detection system (IDS); DDoS attack detection; machine learning; SCADA; industrial control system (ICS); cyber-physical system (CPS)

Citation: Saghezchi, F.B.; Mantas, G.; Violas, M.A.; de Oliveira Duarte, A.M.; Rodriguez, J. Machine Learning for DDoS Attack Detection in Industry 4.0 CPPSs. *Electronics* **2022**, *11*, 602. https://doi.org/10.3390/electronics11040602

Academic Editors: Constantinos Kolias, Georgios Kambourakis and Weizhi Meng

Received: 21 January 2022
Accepted: 13 February 2022
Published: 16 February 2022

1. Introduction

Recent advancements in information and communications technologies, notably the emergence of Internet-of-Things (IoT), cloud-, fog-, and edge-computing networks, Machine-to-Machine (M2M) Communications, Artificial Intelligence (AI), Machine Learning (ML), and Big Data, along with the offer of Ultra-Reliable Low Latency Communication (URLLC) services by Fifth-Generation (5G) mobile operators to private industries, have transformed factories into intelligent, massively interconnected Cyber-Physical Production Systems (CPPSs), laying out the Fourth Industrial Revolution (Industry 4.0) [1], where man, product, and machine are fully interconnected across the whole value chain from the suppliers of the raw materials to the production plant and the front office.

This digitalization helps enhance the transparency of all production stages from the time when the order is dispatched until the end of life of the product. It further enables customized production and agile reconfiguration of manufacturing processes to respond to any change in the customer's requirements in a timely fashion [2]. It further allows

monitoring and tracking of the final goods even after they leave the production line. That is, the data can be collected and analyzed in real time to serve different purposes such as predictive maintenance, quality control, root-cause analysis, and so on, with no or minimal human intervention [3].

Along with this digitalization, the security landscape in the industrial production environment is evolving rapidly. First and foremost, connecting traditionally isolated factories to the Internet means that they are no longer protected by the "air gap" philosophy [4]. Wireless communications and cloud services are increasingly adopted in CPPSs to interconnect different stakeholders across the supply chain, which significantly expands the attack surface. Moreover, smart factories rely not only on resource constraint IoT nodes, but also historically installed legacy equipment that lacks security capabilities. These vulnerabilities can be leveraged by attackers to penetrate into the system and trigger larger consequences, e.g., shutting down the whole production line or manipulating the production processes, e.g., to compromise the quality of the delivered goods [5]. Hence, deploying an effective Intrusion Detection System (IDS) is crucial to safeguard the network perimeter while allowing different subsystems to interact openly without imposing unnecessary access control restrictions.

In computer networks, Distributed Denial-of-Service (DDoS) flooding attacks are deliberate attempts to deny the access of legitimate users to network services. In this type of attack, the offenders take control of a large number of computers (*zombies*) to build up an attack army (*Botnet*). Once the Botnet has been established, they start launching coordinated and large-scale attacks against one or more vulnerable targets in the network. The attackers also usually hide the real locations of the zombies by forging their IP addresses. The target of a DDoS attack can be either a host computer or a communication link. Consequently, the resulting attack is referred to as a *destination flooding* attack or a *link flooding* attack, respectively [6].

In this paper, we propose an ML-based IDS for DDoS attack detection in a real-world Industry 4.0 CPPS. For benign data, we collect network traffic data (PCAP files) from Infineon's semiconductor production facilities (this dataset was provided by Infineon Technologies of Austria in the context of H2020-ECSEL Semi40 project). For the malicious data, we use DDoS attack fingerprints and samples of actual attacks (e.g., PCAP and NFDUMP files) from the DDoSDB (https://www.csg.uzh.ch/ddosgrid/ddosdb/, accessed on 20 January 2022) database of the University of Twente (the Netherlands). We use the *NetMate* tool [7] to extract 45 bidirectional flow features from PCAP files (e.g., packet length, flow duration, and packet interarrival time) and construct labeled datasets for training and testing the proposed IDS. We apply feature selection techniques—e.g., Principal Component Analysis (PCA)—to reduce the data dimensionality and train eight supervised learning algorithms—namely *One Rule (OneR), Logistic Regression (LR), Naïve Bayes (NB), Bayesian Network (BN), K-Nearest Neighbors (K-NN), Decision Tree (DT), Random Forest (RF)*, and *Support Vector Machine (SVM)*. We further study two unsupervised learning algorithms, namely *simple K-Means* and *Expectation-Maximization (EM)* as well as one semi-supervised/statistical learning algorithm, namely *univariate Gaussian* algorithm. To the best of our knowledge, this is the first study analyzing data collected from a real-world factory and constructing ML models for detecting DDoS attacks in Industry 4.0 CPPSs. Previous research efforts have been focused on detecting intrusions in other critical infrastructures such as the power grid [8,9] or water and wastewater treatment [10,11], mostly adopting a model-based approach for the physical control system. Although there are some works applying ML for DDoS attack detection in Operational Technology (OT) networks [12,13], they rely on data either artificially synthesized or collected from an Information Technology (IT) network (e.g., *KDD CUP 99*) [14] or from small-scale lab testbeds [12,13].

The contributions of this paper can be summarized as follows.

- We propose an ML-based network anomaly detection system for detecting DDoS attacks in Industry 4.0 CPPSs.

- We analyze benign network traffic data collected from a real-world large-scale factory and actual DDoS attacks data from DDoSDB data base.
- We study the performance of 11 semi-supervised, unsupervised, and supervised ML algorithms for DDoS attack detection and discuss their merits for the proposed ML-based network anomaly detection system.

The rest of this paper is organized as follows. Section 2 reviews the current state of the art. Section 3 introduces the architecture of our proposed ML-based IDS, presenting the 45 bidirectional flow features that we have extracted from collected data and the incorporation of feature selection algorithms to reduce the dimensionality of data. In this section, we also describe ML algorithms that we have deployed in the detection engine of our proposed IDS, along with key metrics that we have used for their performance evaluation. Section 4 presents and discusses the simulation results. Finally, Section 5 concludes and draws guidelines for future work.

2. Related Work

2.1. Industrial Control Systems

An *Industrial Control System (ICS)* is a computer-based system that monitors and controls physical industrial processes [15]. ICSs are usually incorporated in critical infrastructures such as smart grid, water grid, oil processing and transportation, railway management, and discrete manufacturing systems [16]. For instance, in smart grid systems, they monitor demand and supply variations and help integrate distributed energy sources and orchestrate transactive peer-to-peer electricity trades among local consumers and presumes [17,18]. In contrast to ICS, a *Distributed Control System (DCS)* is a control system for a process or plant, with control components distributed throughout the system [19].

Figure 1 illustrates the reference model for automation integration in CPPSs introduced by ANSI/ISA-95 [20]. In this model, we can identify three main layers: (i) *Field-Area Network*, (ii) *Process-Area Network*, and (iii) *Business-Area Network*.

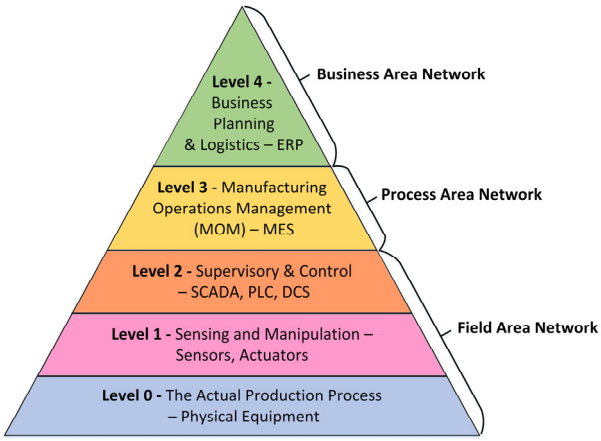

Figure 1. The ANSI/ISA-95 functional hierarchy model for automation integration in CPPSs.

The lower three levels (Levels 0–2) comprise the *Field-Area Network*, which exploits a Supervisory Control and Data Acquisition (SCADA) system for polling the data from Remote Terminal Units (RTUs) (e.g., sensors, relays, and Programmable Logic Controllers (PLCs)) and applying the control signals to the actuators in the field [21]. SCADA is an ICS widely used for telemetry and control in critical infrastructures, e.g., CPPSs. A typical SCADA system is composed of hardware and software components and the communication protocol. IEC 60870-5-104, Modbus, Probus, and Distributed Network Protocol (DNP3) are some of the most commonly used communication protocols for SCADA [22].

The *Process-Area Network* is the intermediate layer (Level 3 in Figure 1), which exploits a Manufacturing Execution System (MES) for smart process control to load recipes to

manufacturing equipment and track and record the transformation of raw material into the final product [23]. Finally, the third layer, which sits on the top of the Field- and Process-Area Networks in the functional hierarchy model of Figure 1 (Level 4) is referred to as the *Business-Area Network*, which exploits an Enterprise Resource Planning (ERP) system for the integration and optimization of business processes, including inventory and human resource management, accounting, order processing, and customer relationship management [16,24].

2.2. Intrusion Detection Systems

In a computing network carrying out predefined tasks, an intrusion is defined as any deviation from allowed operations. In most cases, it is generated with an intention to compromise or misuse the information system. An IDS aims to detect and trace such inappropriate, incorrect, unauthorized, and anomalous activities within the network [25].

An IDS essentially relies on analyzing data collected from a local host machine or from a network. The former is referred to as a *host-based IDS*, which analyzes the host's log files containing information such as invocation of *system calls* to request services from operating system's kernel, active processes running on the machine, outgoing/incoming network traffic profile, and so forth, to detect unauthorized or unusual behavior (e.g., malware running on the host machine) [26]. In contrast, a *network-based IDS* monitors the traffic flows/packets traversing the network to identify suspicious network activities, e.g., port scanning, replay/delay attacks, Denial-of-Service (DoS), and DDoS attacks [27,28].

The detection engine of an IDS can rely on either *anomaly* or *misuse* detection, or a combination of both—i.e., a *hybrid* approach. The *misuse detection* essentially builds a model for the malicious behavior (e.g., a database of attack signatures). Any traffic pattern that matches these attack signatures is considered to be an intrusion [29]. This approach is also referred to as *signature-based* or *knowledge-based* intrusion detection [30]. In contrast, the *anomaly detection* approach creates a model for the "normal" behavior and looks for patterns in the data that fail to conform to this notion of normality [31,32].

Although *misuse detection* is, in general, more accurate than *anomaly detection*, it is unable to detect *zero-day* attacks or any minor variant of a previously known attack. It also requires maintaining an up-to-date database of attack signatures, which can be an exhausting task in practice [26]. In contrast, anomaly detection—despite being prone to generating a large number of false positives—can detect both known and unknown attacks with high accuracy. Furthermore, training anomaly detection algorithms require a dataset containing mostly benign examples, with only few malicious examples for the calibration. This property makes them a viable option when malicious examples are unavailable or prohibitively expensive to generate, which is the case for critical infrastructures (e.g., factories). Although the mainstream IDS solutions are based on misuse detection, the research on IDS is mostly focused on anomaly detection [30].

2.3. Intrusion Detection in Critical Infrastructures

There are several works adopting a physics-based model for anomaly detection in ICSs [33]. For example, Guan and Ge [34] used state estimation models to detect False Data Injection (FDI) and jamming attacks in an industrial continuous-stirred tank reactor. The FDI attack was launched against the physical process to modify the system's state while the jamming attack targeted the cyber layer to render wireless links unavailable for carrying sensor signals to the remote controller. Dolk et al. [35] proposed control strategies that ensure the resilience of a *Networked Control System* under DoS attacks. Pasqualetti et al. [36] studied deception and DoS, stealth, FDI, replay, and covert attacks in power systems, adopting a model-based approach governing the physical dynamics of the system. Zamani et al. [37] proposed an islanding detection technique for Distributed Generation (DG) units in smart grid by analyzing the angle of the DG power factor. However, despite these efforts, although there are some studies addressing DDoS attacks detection in IT networks [38,39],

there is still a lack of knowledge of detecting DDoS attacks in OT networks (e.g., ICSs and SCADA).

2.4. Machine Learning for Intrusion Detection

The objective of an ML-based IDS is to analyze collected data to detect patterns that could reflect possible attacks on the target host machine and/or network [40]. The training dataset can be either *labeled* or *unlabeled*. Correspondingly, the exploited learning algorithm can be either *supervised* or *unsupervised*, respectively [41,42]. Although supervised learning can generally achieve better performance, it requires a labeled dataset containing a balanced set of representative examples from both benign and malicious data. This may require a substantial amount of human effort for collecting malicious examples and labeling the dataset. Consequently, researchers may resort to datasets consisting of a combination of normal examples from real data and malicious examples that are artificially generated by simulations [25].

There still exists a third category of ML algorithms, called *semi-supervised* learning. It relies mostly on benign examples for training while using a few malicious examples for calibration and fine tuning [42]. This makes them particularly attractive for *anomaly* detection in systems where generating anomalous examples is costly, and as such we end up having a very limited number of malicious examples in the training dataset.

There are several previous research efforts applying ML for network intrusion detection. Amouri et al. applied ML for intrusion detection in IoT networks [43]. Sarker et al. [44] ranked different features based on their importance and constructed a DT model for intrusion detection in IT networks. Once-Class Support Vector Machine (OCSVM) has been applied for anomaly detection in SCADA [45] and ICS [46] systems. Linda et al. [25] applied Artificial Neural Networks (ANNs) to detect anomalies in ICSs using a window-based feature extraction techniques from time series datasets. However, they collected the benign data from a small-scale lab testbed and generated the attack vectors randomly. Statistical learning algorithms have also been applied for anomaly detection in SCADA [47] and DCS [48,49] systems.

Despite these efforts, the research on applying ML techniques to DDoS attack detection in Industry 4.0 CPPSs is still very limited. The existing works use either datasets collected from conventional IT networks (e.g., *KDD CUP 99*) or small-scale lab testbeds, or generated by computer simulations.

3. Proposed DDoS Attack Detection System for Industry 4.0 CPPSs

3.1. Proposed IDS Architecture

Figure 2 illustrates the architecture of our proposed anomaly based network IDS for DDoS attack detection in Industry 4.0 CPPSs. The IDS starts by capturing network traffic traces (PCAP files) from the target CPPS (including, SCADA, MES, ERP, etc.). It then extracts bidirectional traffic flow features. The extracted features include attributes such as packet length, transport layer protocol, number of sent/received packets, packet interarrival time, and flow duration. Table 1 summarizes the complete list of 45 extracted features. The feature selection module applies PCA algorithm to map this data to a lower dimensional linear space to eliminate redundant and/or less informative features, without losing data variance. Afterward, the mapped data is fed directly into the detection engine, which generates an alert if a malicious network traffic pattern is observed. As the notion of benign behavior may change over time, the dataset generated by the feature selection module is saved in a database, and is used, from time to time, to retrain the ML models employed in the detection engine. Finally, the alert manager analyzes the output of the detection engine and triggers an alert if the probability of intrusion escalated to a user predefined security level, e.g., when multiple alerts are observed consecutively.

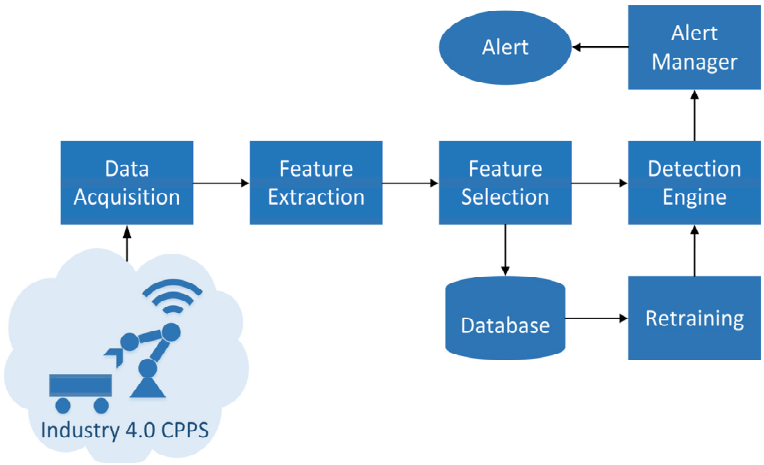

Figure 2. Architecture of the proposed ML-based DDoS attack detection system for Industry 4.0 CPPSs.

Table 1. List of extracted flow-based network traffic features using *NetMate*.

Feature Index	Feature Description	Feature Index	Feature Description
1	Source IP address	24	Max. packet interarrival time in backward path
2	Source Port number	25	Std. of packet interarrival time in backward path
3	Destination IP address	26	Flow (session) duration (µs)
4	Destination Port number	27	Min. active time of the flow (before going idle) (µs)
5	Transport protocol (TCP/UDP)	28	Average active time of the session (µs)
6	Total packets transmitted in forward direction	29	Max. active time of the session (µs)
7	Total Bytes transmitted in forward direction	30	Std. of active time of the session (µs)
8	Total packets transmitted in backward direction	31	Min. idle time of the flow (before going active) (µs)
9	Total Bytes transmitted in backward direction	32	Average idle time of the flow (µs)
10	Min. packet length in forward direction	33	Max. idle time of the flow (µs)
11	Average packet length in forward direction	34	Std. of idle time of the flow (µs)
12	Max. packet length in forward direction	35	Avg. number of packets in the forward sub flow
13	Std. of packet length in forward direction	36	Avg. number of bytes in the forward sub flow
14	Min. packet length in backward direction	37	Avg. number of packets in the backward sub flow
15	Average packet length in backward direction	38	Avg. number of bytes in the forward sub flow
16	Max. packet length in backward direction	39	Number of PSH flags sent forward (0 for UDP)
17	Std. of packet length in backward direction	40	Number of PSH flags sent backward (0 for UDP)
18	Min. packet interarrival time in forward direction	41	Number of URG flags sent forward (0 for UDP)
19	Avg. packet interarrival time in forward direction	42	Number of URG flag sent backward (0 for UDP)
20	Max. packet interarrival time in forward direction	43	The total bytes used for headers in forward path
21	Std. of packet interarrival time in forward path	44	The total bytes used for headers in backward path
22	Avg. packet interarrival time in backward path	45	Time Stamp
23	Mean packet interarrival time in backward path	-	-

3.2. Feature Extraction and Dataset Generation

To extract flow features from PCAP files, there exist several tools, e.g., *T-Shark* [50], *NetMate* [7], and *Libprotoident* [51]. We used *NetMate* since it extracts a rich set of statistical flow features that are more appropriate for our ML approach (cf. Table 1). We save the extracted flow features in a CSV (Comma Separated Values) file, where each row represents a bidirectional flow and each column represents a flow feature, except the last column, which holds the binary label of the flow ('0' stands for benign and '1' stands for malicious). We construct a dataset with 40 thousand examples, half of it generated from benign data and the other half from malicious data. Hence, the dataset is a CSV file with 40 k rows and 46 columns (45 features plus the label).

3.3. Pre-Analysis and Feature Selection

Before applying the PCA algorithm, we did some pre-analysis on the training dataset. We realized that two features, namely the TCP URG flag in forward and backward directions, are constant across the whole dataset. Their values were equal to zero for all benign and malicious examples. As such, they contained no information to help the classifier discriminate between the two data classes. Therefore, we eliminated them from train and test datasets.

Furthermore, to avoid overfitting, we removed the time stamp (feature number 45) and the source and destination IP addresses and port numbers (features 1 to 4) from the training data. The rationale behind this elimination was that our benign and malicious data have been collected in different and non-overlapping time intervals from two different networks (with different sets of IP addresses) communicating through considerably different sets of port numbers. Consequently, the benign and malicious data could be distinguished by these features, and therefore using them could lead to overfitting. Nevertheless, it is important to note that these features might be informative on other occasions, e.g., using timestamp for time series analysis. After eliminating these attributes, we eventually ended up with 38 features. Then, we normalized them by subtracting from each feature its mean value and dividing the result by the standard deviation of the same feature. At the end, each remaining feature stayed with zero mean and unit variance.

After this pre-analysis, we applied the PCA algorithm to the normalized dataset to reduce its dimensionality even further. Figure 3 illustrates the attained variance by PCA for different numbers of selected attributes. We observe that to maintain 95 percent of data variance, we would need to choose at least 13 features (out of the remaining 38 features), a considerable reduction in the data dimensionality by sacrificing only 5 percent of variations in the data. In fact, applying the PCA algorithm (with 95% variance retain) on the original 38-dimensional dataset (see it as a matrix) transforms it to a 13-dimensional space that is spanned by the first 13 principal component eigenvectors of the (original) dataset. Hereafter, we train and test the applied ML algorithms using this mapped dataset in the reduced space.

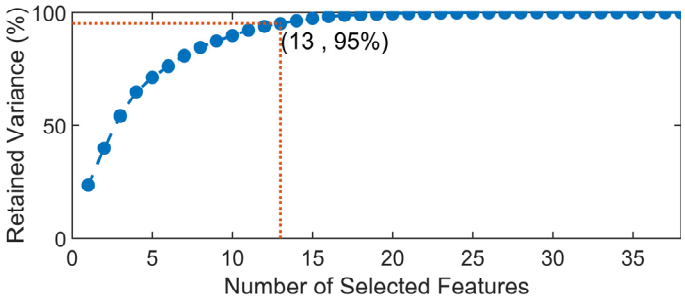

Figure 3. Retained variance by the PCA algorithm when the number of selected features varies between 1 and 38.

3.4. Applied Machine Learning Algorithms

As mentioned before, we applied 11 different ML algorithms for constructing data-driven models that can discriminate between anomalous network traffic flows (originated from DDoS attacks) and normal traffic flows. These algorithms are of three different categories, namely semi-supervised, unsupervised, and supervised. For semi-supervised learning, we used the *univariate Gaussian* algorithm, which is essentially a statistical learning algorithm [42]. For supervised learning, we applied eight different algorithms, namely *OneR*, *LR*, *NB*, *BN*, *K-NN*, *DT*, *RF*, and *SVM* [5]. Finally, for unsupervised learning, we used the *K-Means* and *EM* algorithms [52].

3.5. Performance Evaluation Metrics

Table 2 illustrates the confusion matrix, showing two possible predictive classes (benign or malicious) for each of the two actual classes. The diagonal elements, *True*

Negative (TN) and *True Positive (TP)*, imply the correct decisions made by the IDS. In contrast, the off-diagonal elements, *False Positive (FP)* and *False Negative (FN)*, imply the wrong decisions that it makes. Before introducing the Key Performance Indicators (KPIs) that we use for performance evaluation, let's first briefly define each of these four possible decision outcomes of the IDS, namely TP, TN, FP, and FN.

Table 2. Confusion matrix of network traffic flow classification.

True Label	Predicted Label	
	Benign	**Malicious**
Benign	TN	FP
Malicious	FN	TP

TP is a decision that correctly classifies a malicious activity as being malicious. TN is a decision that correctly classifies a benign activity as being benign. FP is a decision that wrongly classifies a benign activity as being malicious. Finally, FN is a decision that wrongly classifies a malicious activity as being benign.

Having defined the key terms, we now define the KPIs that we use for numerical validation as follows.

True Positive Rate (TPR), also known as *Recall*, is defined as the ratio of all malicious (positive) examples (TP + FN) that are successfully detected by the IDS [42].

$$TPR = Recall = \frac{\sum TP}{\sum (TP + FN)} \tag{1}$$

False Positive Rate (FPR) is the ratio of all benign (negative) examples (TN + FP) that are wrongly classified as positive.

$$FPR = \frac{\sum FP}{\sum (FP + TN)} \tag{2}$$

The *Receiver Operating Characteristic (ROC)* curve is the plot of TPR (y-axis) when FPR (x-axis) varies between 0 and 1. In general, the larger the area under the ROC curve, the better the detection performance of the IDS; the maximum area under ROC curve is one and it happens when the ROC curve is a unit step function.

Precision is the ratio of all generated alerts by the IDS (either true or false) that are true (i.e., emanated from security incidents).

$$Precision = \frac{\sum TP}{\sum (TP + FP)} \tag{3}$$

Accuracy is defined as the ratio between all correct decisions made by the IDS (TP + TN) and the total number of decisions that it makes (TP + FP + TN + FN).

$$Accuracy = \frac{\sum (TP + TN)}{\sum (TP + TN + FP + FN)} \tag{4}$$

F-Measure is defined as the harmonic mean of *Precision* and *Recall*.

$$F_1 = 2\frac{Precision \times Recall}{(Precision + Recall)} \tag{5}$$

4. Performance Evaluation Results

In this section, we present the numerical results for testing the three categories of constructed ML models (i.e., semi-supervised, unsupervised, and supervised) and evaluate their DDoS attack detection performance.

4.1. Semi-Supervised Learning Algorithms

Figure 4 illustrates the performance of the univariate Gaussian algorithm. The (diagonal) green dotted line shows the ROC curve of a random classifier (Accuracy = 0.5).

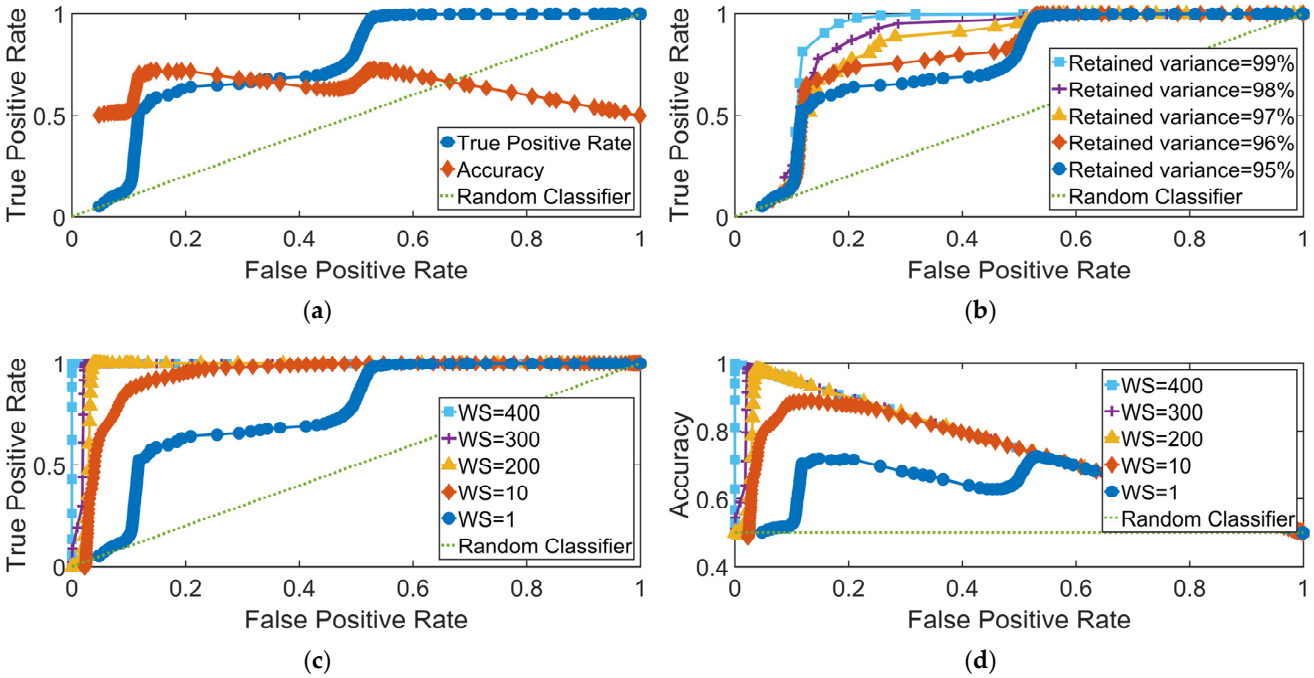

Figure 4. Receiver Operating Characteristic (ROC) and Accuracy curves of the univariate Gaussian algorithm for different variance retains of the PCA algorithm and for different Window Sizes (WSs) of the Moving Average function. (**a**) ROC and Accuracy for PCA with 95% variance retain. (**b**) ROC for different variance retains of the PCA. (**c**) ROC for different Window Sizes (WSs) of the Moving Average. (**d**) Accuracy for different Window Sizes (WSs) of the Moving Average.

Figure 4a shows the ROC curve and the Accuracy of the univariate Gaussian algorithm when PCA with 95 percent variance retain is applied. It is evident from this figure that the univariate Gaussian algorithm has a poor detection performance. In the worst case, for low values of FPR, the ROC curve of the IDS coincides with the ROC of a random classifier (illustrated by the diagonal green dotted line). However, as the FPR increases, the Accuracy improves and attains its maximum (0.7) at FPR = 0.15. Afterwards, the Accuracy starts to decline. Furthermore, when Accuracy peaks, the TPR is 0.6, which means that only 60% of security incidents are detected by the IDS and the rest 40% remain undetected—i.e., they are falsely classified as benign flows.

We tested the algorithm for different values of variance retain of the PCA algorithm, changing it from 95 to 99 percent. Figure 4b depicts the results. As seen from this figure, when the variance retain is increased, the TPR consistently improves, but the elbow point where the maximum Accuracy occurs remains unchanged (TPR always peaks at FPR = 0.15). This means that the IDS still generates a large number of false positives.

We observe from our further data analysis that the mean probability of benign data is noticeably higher than the mean probability of malicious data. This helps the learning algorithm discriminate between the two data types. However, the variances of benign and malicious data are quite high. This results in considerable overlap between the two probability densities, which makes it difficult for the detection engine to successfully discriminate between the two data classes. This manifests itself in poor detection performance. To help improve the performance of the learning algorithm, we applied a smoothing function in order to decrease the fluctuations of the variance of the Probability Density Function

(PDF), denoted by $p(x)$. To do so, we applied the following *Moving Average* function with a Window Size (WS) M, where $\overline{p}(x)$ denotes the smoothed PDF.

$$\overline{p}(x) = \frac{1}{M} \sum_{i=0}^{M-1} p(x)_{(M-i)} \tag{6}$$

Figure 4c,d show the ROC and the Accuracy after applying the moving average function when the variance retain of the PCA algorithm is set to 95 percent. We can see from these figures that applying the moving average function significantly improves the detection performance. In particular, when the WS is 400 samples, the detection engine perfectly discriminates the two data classes, with zero error (TPR = 1, FPR = 0).

Although increasing the WS improves the overall performance of the detection engine, we should note that this comes with a cost. When the IDS operates in real time, it receives the data examples in a stream, which in general can be all benign or all malicious, or a mix of both. When the acquired data within a window frame is all benign or all malicious, averaging them will, in general, result in a higher classification Accuracy. However, when the captured data within a window contain both benign and malicious examples, averaging them can smooth out malicious or benign examples in between, which can lead to a wrong decision by the classifier.

4.2. Unsupervised Learning Algorithms

For unsupervised learning algorithms, we examined K-Means and EM. Tables 3 and 4 summarize the results with and without applying the PCA algorithm, respectively. We observe from Table 3 that applying these clustering algorithms in combination with the PCA algorithm (with 95% variance retain) results in very poor detection performance. The Accuracy of both algorithms does not go beyond 0.52, which means that almost half of the decisions made by these algorithms are wrong. Moreover, their Precision is also around 0.5, which means that around half of the alerts generated by them falsely originate from benign examples. In contrast, as we can see from Table 4, when we deactivate the PCA algorithm, both clustering algorithms perform much better. In general, without applying the PCA, although the overall Accuracy of the two unsupervised algorithms is similar, they perform a bit differently. K-Means shows a very low false alarm rate, of course, with the expense of considerably missing the detection of malicious incidents. In contrast, EM generates a considerable number of false alarms, but it hardly fails to detect DDoS attack flows. Therefore, although these two unsupervised algorithms individually yield a poor performance, they seem to be a good complement for each other because K-Means generates a very low FPR (out of every ten thousand benign flows, only one is wrongly classified as a DDoS attack), and EM attains a very high TPR (out of every 100,000 DDoS flows, only five are misclassified as a benign flow).

Table 3. Performance of unsupervised learning algorithms after applying PCA algorithm with 95% variance retain.

Algorithm	TPR	FPR	Precision	Recall	F1	Accuracy
K-Means	0.822	0.78	0.51	0.822	0.63	0.52
EM	0.70	0.69	0.50	0.70	0.58	0.51

Table 4. Performance of unsupervised learning algorithms without applying the PCA algorithm.

Algorithm	TPR	FPR	Precision	Recall	F1	Accuracy
K-Means	0.90	0.0001	0.9999	0.90	0.95	0.95
EM	0.99995	0.09	0.91	0.99995	0.95	0.95

4.3. Supervised Learning Algorithms

As mentioned before, for supervised learning, we tested eight different algorithms: OneR, LR, NB, BN, K-NN (with K = 1), SVM, DT, and RF. Table 5 summarizes the results when PCA with 95% variance retain is applied. As we can see from this table, supervised algorithms outperform both unsupervised and semi-supervised ones. Even the simple OneR algorithm achieves Accuracy = TPR = 0.987 and FPR = 0.013. We further observe that K-NN, DT, and RF outperform their other supervised counterparts. They attain TPR = Precision = Accuracy = 0.999 and FPR = 0.001. The TPR performance (0.999) shows that out of every 1000 DDoS traffic flows, 999 are correctly detected and only one is misclassified as benign. On the other hand, the FPR performance (0.001) implies that out of every 1000 benign flows, only one is wrongly flagged as a DDoS attack flow.

Table 5. Performance of supervised learning algorithms after applying PCA algorithm with 95% variance retain. The three supervised algorithms that are highlighted in bold (i.e., K-NN, DT, and RF) are the ones that outperform the other ones.

Algorithm	TPR	FPR	Precision	Recall	F1	Accuracy
OneR	0.987	0.013	0.987	0.987	0.987	0.987
LR	0.970	0.029	0.972	0.970	0.970	0.970
NB	0.958	0.042	0.958	0.958	0.958	0.958
BN	0.995	0.005	0.995	0.995	0.995	0.994
K-NN	**0.999**	**0.001**	**0.999**	**0.999**	**0.999**	**0.999**
SVM	0.971	0.028	0.973	0.971	0.971	0.971
DT	**0.999**	**0.001**	**0.999**	**0.999**	**0.999**	**0.999**
RF	**0.999**	**0.001**	**0.999**	**0.999**	**0.999**	**0.999**

Overall, comparing the performance of the three types of our studied ML algorithms, the univariate Gaussian algorithm demonstrates a poor detection performance, mainly because it naively looks into every feature individually and tries to model it as a Gaussian distribution. At the other extreme, supervised algorithms (notably, DT, RF, and K-NN) demonstrate an outstanding performance due to their capacity to build a better predictive model characterizing not only individual features, but also their cross correlations. Even the OneR algorithm, which is essentially the simplest tree with only one node and two branches, results in a much higher predictive performance than what semi-supervised or unsupervised algorithms (e.g., K-Means or EM) can attain. In particular, DT, RF, and K-NN (K = 1) outperform all the other algorithms. Among them, DT is the most interesting one in terms of the computational cost and the explainability of the model, and K-NN is the most expensive one because it essentially lacks the training phase, and for any test observation, it needs to search for the nearest data point in the whole dataset.

5. Conclusions

In this paper, we applied ML for detecting DDoS attacks in Industry 4.0 CPPSs. We exported network traffic traces (PCAP files) from a real-world large-scale semiconductor production factory and employed 11 different semi-supervised, unsupervised, and supervised ML algorithms for anomaly detection in network traffic flows. The simulation results showed that supervised learning algorithms outperformed both unsupervised and semi-supervised ones. In particular, DT, RF, and K-NN detected DDoS attacks with Accuracy = Recall = 0.999, Precision = 0.999, and FPR = 0.001.

However, the two applied unsupervised algorithms (K-Means and EM) also showed a very good performance (Accuracy = 0.95, Recall > 0.9, Precision > 0.9, and FPR < 0.09), although their performance decreased significantly when the PCA algorithm was applied (even with 95% variance retain). This is an interesting finding, since unlike supervised learning, unsupervised learning does not require data labeling which is a tedious task in practice and needs a significant amount of human effort and intervention.

Moreover, semi-supervised algorithms are appealing for anomaly detection since they rely mostly on benign data for training and require only a few malicious examples for the calibration. Nevertheless, the applied univariate Gaussian algorithm showed a poor performance, since the network traffic flow features do not naturally possess a Gaussian distribution. Therefore, before applying this algorithm, proper mappings should be applied on the PDF of each selected feature so as their histogram looks like Gaussian. Our study also found that when a smoothing function (e.g., Moving Average) is applied on the generated decision outcomes at the output of the Detection Engine, the univariate Gaussian classifier yields a better performance. However, this may end up smoothing out malicious flows that are interleaved with benign flows and vice versa, leading to FN and FP results.

For future work, this study can be extended in two directions. The performance evaluation of a combination of applied ML algorithms, when they work in parallel (adopting an ensemble learning approach), needs to be investigated. Developing a collaborative IDS for DDoS attack detection in Industry 4.0 CPPSs based on a federated learning approach is another interesting topic that can be explored in the future.

Author Contributions: Conceptualization, F.B.S., G.M. and J.R.; methodology, F.B.S. and G.M.; software, F.B.S.; validation, F.B.S.; formal analysis, F.B.S.; investigation, F.B.S. and G.M.; resources, G.M. and J.R.; data curation, F.B.S.; writing—original draft preparation, F.B.S.; writing—review and editing, F.B.S., G.M., M.A.V. and A.M.d.O.D.; visualization, F.B.S.; supervision, G.M. and J.R.; project administration, F.B.S., G.M., J.R.; funding acquisition, G.M., J.R., F.B.S., M.A.V. and A.M.d.O.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was sponsored in part by the NATO Science for Peace and Security Programme under grant SPS G5797 (PHYSEC), which authors Jonathan Rodriguez and Georgios Mantas would like to acknowledge.

Acknowledgments: The first author wishes to acknowledge insightful discussions with Jair Santanna, Justyna Chromik, Anna Sperotto, and Aiko Pras, during his visit to the DACS research group at University of Twente, the Netherlands.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Lee, J.; Bagheri, B.; Kao, H.-A. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manuf. Lett.* **2015**, *3*, 18–23. [CrossRef]
2. Tjahjono, B.; Esplugues, C.; Ares, E.; Pelaez, G. What does Industry 4.0 mean to Supply Chain? *Procedia Manuf.* **2017**, *13*, 1175–1182. [CrossRef]
3. Esfahani, A.; Mantas, G.; Matischek, R.; Saghezchi, F.B.; Rodriguez, J.; Bicaku, A.; Maksuti, S.; Tauber, M.; Schmittner, C.; Bastos, J. A Lightweight Authentication Mechanism for M2M Communications in Industrial IoT Environment. *IEEE Internet Things J.* **2017**, *6*, 288–296. [CrossRef]
4. Perez, R.L.; Adamsky, F.; Soua, R.; Engel, T. Machine Learning for Reliable Network Attack Detection in SCADA Systems. In Proceedings of the 2018 17th IEEE International Conference on Trust, Security And Privacy in Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), New York, NY, USA, 1–3 August 2018; pp. 633–638.
5. Saghezchi, F.B.; Mantas, G.; Ribeiro, J.; Esfahani, A.; Alizadeh, H.; Bastos, J.; Rodriguez, J. Machine Learning to Automate Network Segregation for Enhanced Security in Industry 4.0. In *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*; Springer: Cham, Switzerland, 2019; Volume 263, pp. 149–158. ISBN 9783030051945.
6. Zargar, S.T.; Joshi, J.; Tipper, D. A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 2046–2069. [CrossRef]
7. NetMate Meter Download | SourceForge.net. Available online: https://sourceforge.net/projects/netmate-meter/ (accessed on 19 January 2022).
8. Ali, S.; Li, Y. Learning Multilevel Auto-Encoders for DDoS Attack Detection in Smart Grid Network. *IEEE Access* **2019**, *7*, 108647–108659. [CrossRef]
9. Saghezchi, F.B.; Mantas, G.; Ribeiro, J.; Al-Rawi, M.; Mumtaz, S.; Rodriguez, J. Towards a secure network architecture for smart grids in 5G era. In Proceedings of the 2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC, Valencia, Spain, 26–30 June 2017.

10. Adepu, S.; Mathur, A. Distributed Attack Detection in a Water Treatment Plant: Method and Case Study. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 86–99. [CrossRef]
11. Junejo, K.N.; Goh, J. Behaviour-Based Attack Detection and Classification in Cyber Physical Systems Using Machine Learning. In *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*; Association for Computing Machinery: New York, NY, USA, 2016; pp. 34–43.
12. Amin, S.; Litrico, X.; Sastry, S.; Bayen, A.M. Cyber Security of Water SCADA Systems—Part I: Analysis and Experimentation of Stealthy Deception Attacks. *IEEE Trans. Control. Syst. Technol.* **2013**, *21*, 1963–1970. [CrossRef]
13. Maglaras, L.A.; Jiang, J.; Cruz, T.J. Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems. *J. Inf. Secur. Appl.* **2016**, *30*, 15–26. [CrossRef]
14. Alhaidari, F.A.; AL-Dahasi, E.M. New Approach to Determine DDoS Attack Patterns on SCADA System Using Machine Learning. In Proceedings of the 2019 International Conference on Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, 3–4 April 2019; pp. 1–6.
15. IBM X-Force Research: Security Attacks on Industrial Control Systems—Security Intelligence. Available online: https://securityintelligence.com/media/security-attacks-on-industrial-control-systems/ (accessed on 15 January 2022).
16. Stouffer, K.; Lightman, S.; Pillitteri, V.; Abrams, M.; Hahn, A. *Guide to Industrial Control Systems (ICS) Security*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2015. [CrossRef]
17. Zamani, R.; Moghaddam, M.P.; Haghifam, M.-R. Dynamic Characteristics Preserving Data Compressing Algorithm for Transactive Energy Management Frameworks. *IEEE Trans. Ind. Inform.* **2022**, 1. [CrossRef]
18. Zamani, R.; Moghaddam, M.P.; Haghifam, M.-R. Evaluating the Impact of Connectivity on Transactive Energy in Smart Grid. *IEEE Trans. Smart Grid* **2021**, 1. [CrossRef]
19. Ding, D.; Han, Q.; Wang, Z.; Ge, X. A Survey on Model-Based Distributed Control and Filtering for Industrial Cyber-Physical Systems. *IEEE Trans. Ind. Inform.* **2019**, *15*, 2483–2499. [CrossRef]
20. ISO. IEC 62264-1:2013—Enterprise-control system integration—Part 1: Models and Terminology. Available online: https://www.iso.org/standard/57308.html (accessed on 17 January 2022).
21. IEEE. *IEEE Std C37.1-1994—IEEE Standard Definition, Specification and Analysis of Systems Used for Supervisory Control, Data Acquisition, and Automatic Control*; IEEE: Piscataway, NJ, USA, 1994.
22. Zhu, B.; Sastry, S. SCADA-Specific Intrusion Detection/Prevention Systems: A Survey and Taxonomy. In Proceedings of the 1st Workshop on SECURE Control Systems (SCS), Stockholm, Sweden, 12 April 2010; Volume 11, p. 7.
23. Almada-Lobo, F. The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES). *J. Innov. Manag.* **2015**, *3*, 16–21. [CrossRef]
24. Bartodziej, C.J. (Ed.) *The Concept Industry 4.0 BT—The Concept Industry 4.0: An Empirical Analysis of Technologies and Applications in Production Logistics*; Springer Fachmedien Wiesbaden: Wiesbaden, Germany, 2017; pp. 27–50. ISBN 978-3-658-16502-4.
25. Linda, O.; Vollmer, T.; Manic, M. Neural Network based Intrusion Detection System for critical infrastructures. In Proceedings of the 2009 International Joint Conference on Neural Networks, Atlanta, GA, USA, 14–19 June 2009; pp. 1827–1834.
26. Ribeiro, J.; Saghezchi, F.B.; Mantas, G.; Rodriguez, J.; Abd-Alhameed, R.A. HIDROID: Prototyping a Behavioral Host-Based Intrusion Detection and Prevention System for Android. *IEEE Access* **2020**, *8*, 23154–23168. [CrossRef]
27. Scarfone, K.A.; Mell, P.M. *SP 800-94. Guide to Intrusion Detection and Prevention Systems (IDPS)*; National Institute of Standards & Technology: Gaithersburg, MD, USA, 2007.
28. Liao, H.-J.; Richard Lin, C.-H.; Lin, Y.-C.; Tung, K.-Y. Intrusion detection system: A comprehensive review. *J. Netw. Comput. Appl.* **2013**, *36*, 16–24. [CrossRef]
29. Borges, P.; Sousa, B.; Ferreira, L.; Saghezchi, F.B.; Mantas, G.; Ribeiro, J.; Rodriguez, J.; Cordeiro, L.; Simoes, P. Towards a Hybrid Intrusion Detection System for Android-based PPDR terminals. In Proceedings of the IM 2017—2017 IFIP/IEEE International Symposium on Integrated Network and Service Management, Lisbon, Portugal, 8–12 May 2017.
30. García-Teodoro, P.; Díaz-Verdejo, J.; Maciá-Fernández, G.; Vázquez, E. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.* **2009**, *28*, 18–28. [CrossRef]
31. Barbosa, R.R.R.; Pras, A. *Intrusion Detection in SCADA Networks BT—Mechanisms for Autonomous Management of Networks and Services*; Stiller, B., De Turck, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 163–166.
32. Sperotto, A.; Schaffrath, G.; Sadre, R.; Morariu, C.; Pras, A.; Stiller, B. An Overview of IP Flow-Based Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2010**, *12*, 343–356. [CrossRef]
33. Giraldo, J.; Urbina, D.; Cardenas, A.; Valente, J.; Faisal, M.; Ruths, J.; Tippenhauer, N.O.; Sandberg, H.; Candell, R. A Survey of Physics-Based Attack Detection in Cyber-Physical Systems. *ACM Comput. Surv.* **2018**, *51*, 1–36. [CrossRef]
34. Guan, Y.; Ge, X. Distributed Attack Detection and Secure Estimation of Networked Cyber-Physical Systems Against False Data Injection Attacks and Jamming Attacks. *IEEE Trans. Signal Inf. Process. Netw.* **2018**, *4*, 48–59. [CrossRef]
35. Dolk, V.S.; Tesi, P.; De Persis, C.; Heemels, W.P.M.H. Event-Triggered Control Systems Under Denial-of-Service Attacks. *IEEE Trans. Control. Netw. Syst.* **2017**, *4*, 93–105. [CrossRef]
36. Pasqualetti, F.; Dörfler, F.; Bullo, F. Attack Detection and Identification in Cyber-Physical Systems. *IEEE Trans. Autom. Control.* **2013**, *58*, 2715–2729. [CrossRef]
37. Zamani, R.; Moghaddam, M.P.; Panahi, H.; Sanaye-Pasand, M. Fast Islanding Detection of Nested Grids Including Multiple Resources Based on Phase Criteria. *IEEE Trans. Smart Grid* **2021**, *12*, 4962–4970. [CrossRef]

38. Jonker, M.; Sperotto, A.; Pras, A. DDoS Mitigation: A Measurement-Based Approach. In Proceedings of the NOMS 2020—2020 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 20–24 April 2020; pp. 1–6.
39. Steinberger, J.; Sperotto, A.; Baier, H.; Pras, A. Distributed DDoS Defense: A collaborative Approach at Internet Scale. In Proceedings of the NOMS 2020—2020 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 20–24 April 2020; pp. 1–6.
40. Jiang, J.; Yasakethu, L. Anomaly Detection via One Class SVM for Protection of SCADA Systems. In Proceedings of the 2013 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Beijing, China, 10–12 October 2013; pp. 82–88.
41. Tsai, C.-F.; Hsu, Y.-F.; Lin, C.-Y.; Lin, W.-Y. Intrusion detection by machine learning: A review. *Expert Syst. Appl.* **2009**, *36*, 11994–12000. [CrossRef]
42. Ribeiro, J.; Saghezchi, F.B.; Mantas, G.; Rodriguez, J.; Shepherd, S.J.; Abd-Alhameed, R.A. An Autonomous Host-Based Intrusion Detection System for Android Mobile Devices. *Mob. Netw. Appl.* **2020**, *25*, 164–172. [CrossRef]
43. Amouri, A.; Alaparthy, V.T.; Morgera, S.D. A Machine Learning Based Intrusion Detection System for Mobile Internet of Things. *Sensors* **2020**, *20*, 461. [CrossRef]
44. Sarker, I.H.; Abushark, Y.B.; Alsolami, F.; Khan, A.I. IntruDTree: A Machine Learning Based Cyber Security Intrusion Detection Model. *Symmetry* **2020**, *12*, 754. [CrossRef]
45. Maglaras, L.A.; Jiang, J. Intrusion detection in SCADA systems using machine learning techniques. In Proceedings of the 2014 Science and Information Conference, London, UK, 27–29 August 2014; pp. 626–631.
46. Schuster, F.; Paul, A.; Rietz, R.; Koenig, H. Potentials of Using One-Class SVM for Detecting Protocol-Specific Anomalies in Industrial Networks. In Proceedings of the 2015 IEEE Symposium Series on Computational Intelligence, Cape Town, South Africa, 8–10 December 2015; pp. 83–90.
47. Do, V.L.; Fillatre, L.; Nikiforov, I. A statistical method for detecting cyber/physical attacks on SCADA systems. In Proceedings of the 2014 IEEE Conference on Control Applications (CCA), Antibes/Nice, France, 8–10 October 2014; pp. 364–369.
48. Rrushi, J.; Kang, K.-D. Detecting Anomalies in Process Control Networks. In *Critical Infrastructure Protection III*; Palmer, C., Shenoi, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 151–165. ISBN 978-3-642-04798-5.
49. Valdes, A.; Cheung, S. Communication pattern anomaly detection in process control systems. In Proceedings of the 2009 IEEE Conference on Technologies for Homeland Security, Waltham, MA, USA, 11–12 May 2009; 2009; pp. 22–29.
50. T-Shark: Terminal-Based Wireshark. Available online: https://www.wireshark.org/docs/wsug_html_chunked/AppToolstshark.html (accessed on 19 January 2022).
51. GitHub—Wanduow/Libprotoident: Network Traffic Classification Library that Requires Minimal Application Payload. Available online: https://github.com/wanduow/libprotoident (accessed on 19 January 2022).
52. Moon, T.K. The expectation-maximization algorithm. *IEEE Signal Process. Mag.* **1996**, *13*, 47–60. [CrossRef]