Received: 6 July 2020 | Revised: 18 March 2021 | Accepted: 26 April 2021 | IET Cyber-Physical Systems: Theory & Applications

DOI: 10.1049/cps2.12016

ORIGINAL RESEARCH PAPER

Flow-based intrusion detection algorithm for supervisory control and data acquisition systems: A real-time approach

Marcio Andrey Teixeira[1] | **Maede Zolanvari**[2] | **Khaled M. Khan**[3] | **Raj Jain**[2] | **Nader Meskin**[4]

[1]Department of Informatics, Federal Institute of Education, Science, and Technology of São Paulo, Catanduva, Brazil

[2]Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, USA

[3]Department of Computer Science and Engineering, Qatar University, Doha, Qatar

[4]Department of Electrical Engineering, Qatar University, Doha, Qatar

Correspondence

Marcio Andrey Teixeira, Department of Informatics, Federal Institute of Education, Science, and Technology of Sao Paulo, Sao Paulo, Brazil.
Email: marcio.andrey@ifsp.edu.br

Funding information

Qatar National Research Fund, Grant/Award Number: NPRP10-0206-170360; National Science Foundation, Grant/Award Number: CNS-1718929; São Paulo Research Foundation (FAPESP), Grant/Award Number: 2017/01055-4

Abstract

Intrusion detection in supervisory control and data acquisition (SCADA) systems is integral because of the critical roles of these systems in industries. However, available approaches in the literature lack representative flow-based datasets and reliable real-time adaption and evaluation. A publicly available labelled dataset to support flow-based intrusion detection research specific to SCADA systems is presented. Cyberattacks were carried out against our SCADA system test bed to generate this flow-based dataset. Moreover, a flow-based intrusion detection system (IDS) is developed for SCADA systems using a deep learning algorithm. We used the dataset to develop this IDS model for real-time operations of SCADA systems to detect attacks momentarily after they happen. The results show empirical proof of the model's adequacy when deployed online to detect cyberattacks in real time.

1 | INTRODUCTION

Industrial control systems (ICSs) use a diverse set of technologies to control and monitor industrial processes. ICSs include several systems such as supervisory control and data acquisition (SCADA) systems, remote terminal units, and distributed control systems [1]. ICSs are generally used in critical infrastructures such as power transmission and distribution, oil and gas, nuclear, and other industries where high availability is crucial.

Historically, ICSs were designed to operate on networks that were not connected to the Internet. However, Industry 4.0, also known as Industrial Internet of Things (IIoT), makes Internet-based ICS feasible [2]. These advances expose ICSs to cyberspace and introduce the possibility of cyberattacks through the Internet. An ICS-related security review reported more than 400 ICS vulnerabilities in 2019, in which more than 100 were zero-day vulnerabilities [3, 4]. Zero-day vulnerabilities are previously unseen attacks capable of exploiting the system for a long time until they are discovered.

Over the years, an extensive effort was made by the research community to identify and mitigate prevalent vulnerabilities that these systems face after integrating Internet technology (IT). The techniques and industrial standards for the cybersecurity of the ICS/SCADA system have been surveyed widely in different papers such as Zolanvari et al. [5] and Zhu and Sastry [6].

It is challenging to detect and prevent sophisticated attack incidents using traditional rule-based information technology approaches. Hence, there is a need for new IDS mechanisms for ICS capable of intelligently detecting out-of-ordinary events. Machine learning (ML) and deep learning (DL) algorithms are increasingly being used to constitute smart and efficient IDSs dedicated to ICSs.

The general approach to developing DL-based security systems in the literature consists of training the models using labelled network traces obtained from publicly available datasets [7]. These datasets include sets of intrusions and abnormal traffic, along with the standard traffic network behaviours [8]. Generally, the network traffic is captured in either packet-based or flow-based format. Packet-based data encompass complete payload information whereas flow-based data are more aggregated and usually contain only metadata from flows, that is, network connections [9]. Despite the richness of contributions in this field, providing a benchmark for publicly available flow-based datasets is still an open issue in IDS for ICS research. The performance of any intelligent IDS depends highly on the dataset used.

Moreover, the malware landscape is evolving rapidly, along with frequent changes in the attack strategies. Therefore, the datasets become outdated quickly. As a result, it is crucial to have up-to-date datasets.

A flow-based dataset developed for ICS cybersecurity research is presented, along with a unique flow-based IDS using DL. The ICS platform used is our SCADA system test bed [5, 10, 11]. Cyberattacks were conducted on the SCADA system test bed, and the network traffic was captured. The features of the network traffic were extracted. After extensive data preprocessing (including data cleaning and labelling), these features have been used to build a flow-based dataset ready for training. The DL algorithm studied is the deep artificial neural network (ANN). After training and testing, the DL model is embedded into our proposed IDS and deployed in the SCADA system network. Its efficiency to detect the attacks in real time is analysed. The performance evaluation of our IDS is studied in two phases: offline and online. The offline phase consists of training and testing the DL algorithm using the flow-based dataset. The online phase is composed of the trained model deployed in the SCADA test bed and testing it, using real-time network traffic. Afterwards, the results in the offline and online phases are compared.

The main contributions of this work are as follows: (1) development of a new flow-based dataset to be used in cybersecurity research that will be available for download and can be used in new cybersecurity research works; (2) development of a unique flow-based IDS using a DL algorithm; (3) integration of this model in an ICS to detect cyberattacks in real time; and (4) performance comparison of offline and online versions of a DL-based IDS.

We have organised the rest of this work as follows. Section 2 presents an overview of the current status of cybersecurity research in this field, our prior works, and other related works. Section 3 highlights the deep ANN model and performance measurement metrics. In Section 4, the attacks performed against the test bed are discussed, and the network features that compose the dataset are described. The proposed flow-based IDS for ICS is presented in Section 5, followed by Section 6, where the numerical results are provided. Finally, Section 7 presents a summary of our conclusions.

2 | BACKGROUND

In this section, we provide a brief background on cybersecurity systems and their specifications. We also highlight our prior work and the state of the art related to this topic.

2.1 | Cybersecurity system overview

Cybersecurity researchers aim to maintain the confidentiality, integrity, and availability triad of information through various cyberdefence systems to protect computers and networks from attacks [12]. Traditional cybersecurity systems address various cybersecurity threats, including spam, botnets, trojans, viruses, etc. Generally, cyberdefence systems combat cyberthreats at two levels, network and host, providing network- and host-based defence. Figure 1 shows these two levels of traditional cybersecurity systems.

In a network-based defence architecture, security is provided at the network traffic level by monitoring all traffic coming from the outside into the system. In the host-based defence, security is ensured on each workstation separately inside the system. In our work, the focus is on the network-based intrusion detection system (IDS).

The IDS monitors network traffic searching for malicious activity or threats and sends out alerts when it discovers such attacks. Intrusion detection methods used by IDSs can be classified as misuse-based (also known as signature-based), anomaly-based, or hybrid [12]. The misuse-based IDS is designed to detect known attacks by using signatures of the attacks; thus, it is more practical in detecting known types of attacks. The anomaly-based IDS is designed to learn normal traffic behaviours and identify anomalies as deviations from normal behaviours. This technique is more useful in detecting unknown attacks (i.e., zero-day attacks). The hybrid method combines misuse-based and anomaly-based methods to identify any sort of threats.

2.2 | Industrial control system security versus traditional Internet technology security

In the past, the ICSs were assumed to be protected against cyberattacks. This is because the ICS networks were isolated from the world, running proprietary control protocols, and used specialised hardware and software. In 2010, the Stuxnet virus [13] showed that this statement is not valid. Despite not being connected to the Internet, traditional ICSs are not protected. Since then, the environment of the ICSs has changed. IT solutions are being adopted to promote corporate connectivity and remote access to ICSs, such as transferring data production to the company's IT system as well as remote maintenance. Modern ICSs are connected to the Internet; therefore, risks for attacks against ICSs have substantially increased. Some models have been proposed to structure ICS architecture and organise it hierarchically [14]. One example is Purdue's model [15], shown in Figure 2.

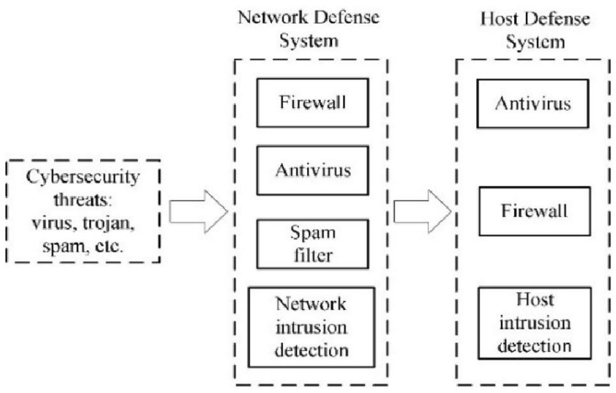


FIGURE 1 Traditional cybersecurity systems [12]

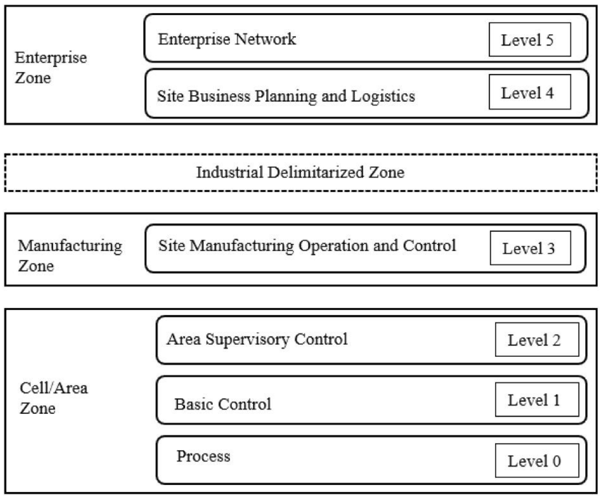


FIGURE 2 Purdue model [15]

As seen in Figure 2, Purdue's model divides the ICS architecture into zones and levels. Each level highlights functional components meant for different control and security levels expected from the layer [16]. The top levels (Levels 4 and 5) in the Enterprise zone are related to IT-like components, and the lower levels denote the edge network and ICS-like elements. Thus, the architecture offers a useful model for distinguishing between IT and ICS. The details of the modern ICS architecture shown in Figure 2 are described in Ackerman [15]. It provides network and security services to the devices, equipment, and applications found in a standard ICS.

ICSs have evolved from being traditional IT systems to controlling complex physical processes in critical infrastructures, whereas traditional IT systems are used to manage the data [1]. Both systems have different operational characteristics, risks, and priorities [1, 17, 18]. For example, the risks in ICSs include significant consequences to human health and safety, severe damage to the environment, and financial issues such as production loss. The security mechanisms used in a traditional IT system are not adequate to protect an ICS. Therefore, it is vital to develop new security mechanisms to safeguard ICSs, and in our case, the SCADA systems.

2.3 | Our prior work

In Teixeira et al. [10], we presented a SCADA system test bed used for cybersecurity research. Cyberattacks were conducted against the test bed and five network features were selected to compose the dataset. In this work, we selected 19 features to compose a new dataset, and a DL algorithm is trained to detect the attacks in real time.

In Zolanvari et al. [11], we studied the problem of an imbalanced dataset facing the training process of ML algorithms, which is a pragmatic challenge in the security of IIoT. It was shown how ML falls short when it comes to real-world security scenarios, and obstacles in applying these methods in real industrial settings. Different performance metrics and how they are affected by the imbalance were presented.

In Zolanvari et al. [5], we improved the test bed presented in Teixeira et al. [10] by integrating new kinds of attacks and data flows in the test bed. A comprehensive ML-based risk assessment when countering the most relevant cyberattacks in IIoT systems was presented.

A survey of the cybersecurity of ICSs is presented in Bhamare et al. [19], in which we studied the state of the art when it comes to upgrading the security of these systems from stand-alone local units to cloud-based approaches.

However, here, we use a flow-based dataset with relevant cyberattacks. The attacks are carried out against our test bed. The number of features used to compose the dataset has increased, and we use an ANN DL algorithm to train our model. Another difference from the previous works is that we analyse the performance of the ANN to detect attacks in real time. The flow-based dataset developed will be available for download to the research community. Our previous dataset from Teixeira et al. [10] is also available for download [20].

2.4 | Related work

Almost all existing research work in this domain used few publicly available datasets to train and test ML and DL algorithms. However, datasets from ICS networks are scarce compared with the numerous publicly available datasets for traditional IT networks. Whereas ML and DL algorithms have been applied in designing IDSs, to the best of our knowledge, real-time application of these algorithms on a well-represented SCADA network has not yet been studied. Some available research work on anomaly detection in ICS is described subsequently.

A survey of existing datasets used in cybersecurity research is provided by Ring et al. [21]. The datasets are classified into packet-based or flow-based categories. It is easy to verify that none of the existing datasets are specific to ICS security, and they mostly use a packet-based format. Our dataset is built using a flow-based format specific to ICSs; it is a better option for cybersecurity research.

Ferrag et al. [22] present a survey of DL approaches for IDSs, in which 35 well-known datasets used in DL cybersecurity research are described. The authors highlight that the community lacks datasets for cybersecurity research and emphasize the importance of having new and up-to-date datasets.

The lack of datasets to develop efficient IDSs is discussed as a significant problem by Malowidzki et al. [23]. Their discussion includes a set of requirements to obtain proper datasets. They state that available research works do not meet these requirements with the datasets they have used. We fulfil these requirements described in Malowidzki et al. [23] to obtain our flow-based dataset.

Youbiao He et al. [24] use DL techniques to discover the behaviour of false data injection attacks in smart grids. The attacks are detected in real time by analysing historical measurement data and the captured features. We explore DL techniques to detect three different types of attacks: command injection, denial of service (DoS), distributed denial of service (DDoS), and reconnaissance.

Yan and Yu [25] developed a detection system using DL to obtain features from sensor measurements of exhaust gas temperatures. The features are used as the input to a neural network classifier to perform combustor anomaly detection. Using real-world data collected from a gas turbine combustion system, the researchers demonstrated that DL-based anomaly detection significantly improves the combustors' anomaly detection performance. Here, the DL algorithms learn the linear and non-linear relationships between features from normal and abnormal network traffic. They then use it to perform attack detection in real time.

In [26], Niyaz et al. propose a DL-based approach to implement an effective and flexible network intrusion detection system. They use self-taught learning, a DL-based technique, on the NSL-KDD dataset, a benchmark dataset for network intrusion. We built a new dataset for use in cybersecurity research. Our target networks are those that have been used in the ICS/SCADA systems. Some attacks analysed are specific to the ICS/SCADA systems and are not implemented on benchmark datasets such as NSL-KDD.

Yin et al. [27] propose a DL approach to intrusion detection using recurrent neural networks (RNNs). They study the performance of the RNN model in binary classification and multiclass classification and compare its performance with some traditional ML algorithms also used in classification. They studied the performance of these algorithms using the benchmark NSL-KDD dataset. As mentioned, the NSL-KDD dataset does not have specific ICS/SCADA attacks.

Fernandes et al. [28] present labelled datasets for SCADA networks for use in cybersecurity research. Their datasets include packet captures from both malicious and non-malicious network traffic. However, their datasets are generated using a simulated environment, which is not a good representative of a real-world plant.

3 | DEEP LEARNING ALGORITHMS AND PERFORMANCE METRICS

This section describes the DL algorithm used in this work, along with the performance metrics used to evaluate the performance.

3.1 | Deep learning model

DL solutions have proven their effectiveness in different applications, including speech recognition [29–31], computer vision [32, 33], and natural language processing [34, 35]. Examples of common DL models include ANN, convolutional neural network, and RNN. We use an ANN model, which is a supervised DL model requiring a dataset with labelled samples. Figure 3 shows an ANN example.

As seen in Figure 3, an ANN consists of an input layer, multiples hidden layers, activation functions, and an output layer. Every node in a layer is connected to a subset of nodes in the next layer. The ANN becomes deeper by increasing the number of hidden layers. Deep neural networks can solve problems with high complexity and address more challenging learning tasks than shallow neural networks [27]. In this work, the IDS analyzes network traffic behaviour to identify whether an attack is happening, which is a classification problem using an ANN algorithm. Figure 4 shows in a simple way how the ANN works in the IDS context. For more technical details about the ANN algorithm, we refer the readers to other work [27, 32–35].

The left circles shown in Figure 4 represent features of the network flow, which are the input features used in the input layer. The circle in the middle represents an artificial neuron, and the right circle represents the output, which can be zero (no attack) or 1 (attack). Every input has a weight (w), and the weighted sum of the inputs triggers the activation function (shown inside the artificial neuron) to obtain one output from the neuron. This example consists of only one layer. However, in the case of an ANN with more layers, the outputs of each layer are used as inputs for the next layer. During training, the weights are trained and adjusted so that the model can distinguish the normal traffic from anomalous traffic. This is done using a back-propagation technique [36, 37]. In this case, as shown in Figure 4, the predicted output ($\hat{y}$) is compared with the expected output (y), and cost function C is used to minimise the difference.

The activation functions have an essential role in neural networks and are a crucial component of DL models. Activation functions decide whether an artificial neuron should be activated. This means determining whether the information that the neuron is receiving is relevant or should be ignored. Hence, the activation function affects the DL model's output, accuracy and the computational efficiency of the ANN model. There are several activation functions, each designed for a different application. In our work, we use two different activation functions: rectified linear unit (ReLU) [36] and sigmoid [37].

3.2 | Performance metrics

The performance of the ANN is analysed by metrics derived from the confusion matrix [38] (Table 1).

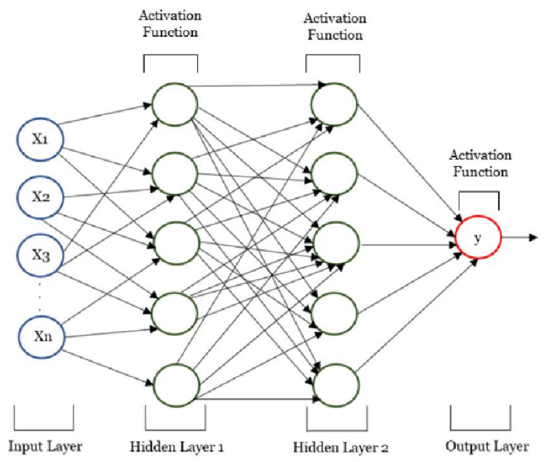


FIGURE 3 Artificial neural network example

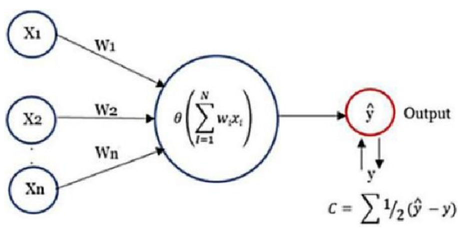

X1
X2
Xn
W1
W2
Wn
$\theta\left(\sum_{i=1}^{N} w_i x_i\right)$
$\hat{y}$
Output
y
$c = \sum \frac{1}{2}(\hat{y} - y)$

FIGURE 4 Artificial neuron example

The following performance metrics, derived from the confusion matrix, are used to evaluate the proposed ANN-based IDS:

- Accuracy (AC): It indicates the percentage of the total traffic that is classified correctly:

$$AC = \frac{TP + TN}{TP + TN + FP + FN} \times 100 \tag{1}$$

- False alarm rate (FAR): It is the percentage of normal traffic that is incorrectly classified as abnormal traffic (attack) by the model:

$$FAR = \frac{FP}{FP + TN} \times 100 \tag{2}$$

- Undetected rate (UND): It represents the fraction of abnormal traffic (attack) that is incorrectly classified as normal traffic by the model:

$$UND = \frac{FN}{FN + TP} \times 100 \tag{3}$$

TABLE 1 Confusion matrix

		Predicted Class	
		Normal	**Abnormal**
Actual class	**Normal**	True negative (TN)	False positive (FP)
	Abnormal	False negative (FN)	True positive (TP)

- TP rate (TPR): It is the fraction of attack traffic that is correctly predicted as an attack by the model:

$$TPR = \frac{TP}{TP + FN} \times 100 \tag{4}$$

- Receiver operating characteristic (ROC) curve: It is a graphical analysis of the TP rate (Equation 4) plotted versus the UND (Equation 3). The ROC curve is used to visualise the performance of the binary classifier.

4 | DEVELOPMENT OF FLOW-BASED DATASET

This section describes attacks conducted against the SCADA system, the network feature selection, and the development of our flow-based dataset. Different datasets are used to analyse network traffic, such as the KDD99 dataset, which is a well-known benchmark, as well as its derived versions (e.g., NSL-KDD). However, these datasets are outdated and are not representative of today's traffic for SCADA systems cybersecurity research. For this reason, there is a need to build a new flow-based dataset that represents network traffic containing various specific SCADA systems. The produced dataset a contribution of this work.

4.1 | Our supervisory control and data acquisition system test bed

The current experiments were conducted using our SCADA system test bed. Figure 5 illustrates a simplified schematic view of this SCADA system test bed. For more technical details about the SCADA test bed, we refer the reader to. [5, 10, 11]

As shown in Figure 5, the SCADA network interconnects field devices (physical process monitors), human–machine interfaces (HMIs), engineering workstations, Argus server, and proposed flow-based IDS. Argus is a real-time network flow monitor system [39] developed and adopted for cybersecurity at Carnegie Mellon's Software Engineering Institute. The Argus system works based on a client–server paradigm. The Argus server tracks and reports transactions detected through network interfaces. The Argus client is used to access data monitored and stored by the Argus

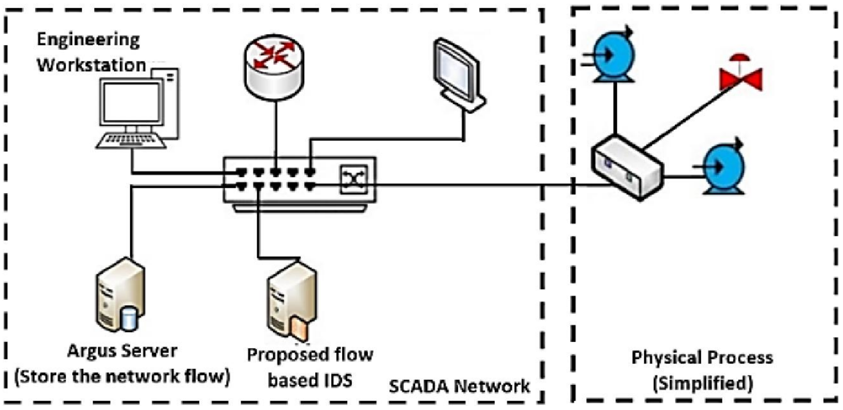
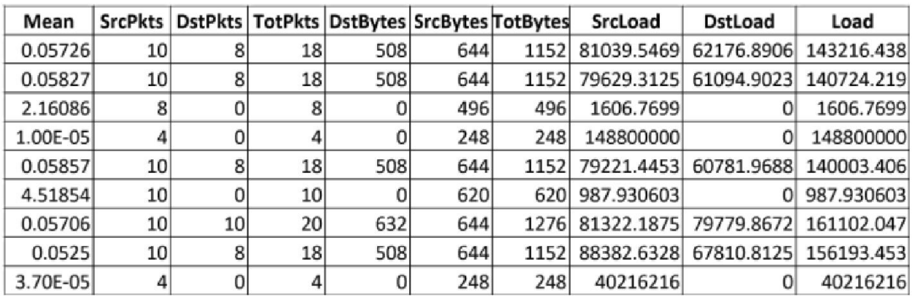

Engineering Workstation
Argus Server (Store the network flow)
Proposed flow based IDS
SCADA Network
Physical Process (Simplified)

FIGURE 5 Simplified supervisory control and data acquisition system test bed



Mean	SrcPkts	DstPkts	TotPkts	DstBytes	SrcBytes	TotBytes	SrcLoad	DstLoad	Load
0.05726	10	8	18	508	644	1152	81039.5469	62176.8906	143216.438
0.05827	10	8	18	508	644	1152	79629.3125	61094.9023	140724.219
2.16086	8	0	8	0	496	496	1606.7699	0	1606.7699
1.00E-05	4	0	4	0	248	248	148800000	0	148800000
0.05857	10	8	18	508	644	1152	79221.4453	60781.9688	140003.406
4.51854	10	0	10	0	620	620	987.930603	0	987.930603
0.05706	10	10	20	632	644	1276	81322.1875	79779.8672	161102.047
0.0525	10	8	18	508	644	1152	88382.6328	67810.8125	156193.453
3.70E-05	4	0	4	0	248	248	40216216	0	40216216

FIGURE 6 Network flows stored by the Argus server

server. As shown in Figure 5, the Argus server is connected to the Ethernet switch of our SCADA system test bed. The switch port, to which the Argus server is connected, is configurated with the mirroring function. Then, all SCADA network traffic (normal traffic and attack traffic) is mirrored to that specific port. Thus, the network flow is stored for further processing and analysis.

A network flow can be defined as a set of packets going from a specific source host to a particular destination host through the network. Packets belonging to the flow have a set of standard features, such as ports and protocols. Figure 6 shows an example of the network flows stored by the Argus server.

Each row of the table depicted in Figure 6 represents a network flow, and each column indicates a feature of the flow. The DL algorithm uses all of the network features described in Table 2 to identify malicious traffic. The feature selection process is described next.

4.2 | Feature selection

Selecting and extracting features from the dataset is an essential part of any ML/DL analytics. An ideal feature for network IDS would be a feature that changes during attacks. That means that the state of the feature should vary during the attack. If the selected features do not change during the attack and normal state, even the best algorithm will not detect an anomaly. We have picked features that are common in a network flow and also show variations during the attacks. Table 2 shows selected features used in this work.

Variations in selected features depend on the attack type. Figure 7 shows an example of traffic behaviour during a DoS attack against our programmable logic controller (PLC).

As shown in Figure 7, the attack phase is represented by red and the normal scenario by blue. Under normal conditions (without attack), the SrcPkts and DstPkts features have periodic behaviour. However, during the attack, traffic behaviour has changed. The number of packets sent (SrcPkts) varies during the attack phase, making the PLC out of service and not transmitting any packet during the attack (DstPkts is zero). We picked SrcPKts and DstPkts as an example here; the same concept applies to other selected features in Table 2.

4.3 | Attack scenarios

The attacks described in this subsection were conducted against our SCADA system test bed shown in Figure 4. Our SCADA system monitors and controls the water level in the water storage tank. Different tools were used to attack the SCADA system test bed. The attacks performed against the test bed and the tools used are described in Table 3.

A brief description of each attack is provided next. More details can be found elsewhere [41–44].

- Reconnaissance: Reconnaissance is the first stage of an attack. In this stage, using network scan tools, hackers try to

Features	Type	Descriptions
Mean flow (mean)	Float	Average duration of active flows
Source port (Sport)	Integer	Source port number
Destination port (Dport)	Integer	Destination port number
Source packets (SrcPkts)	Integer	Source/destination packet count
Destination packets (DstPkts)	Integer	Destination/source packet count
Total packets (Tpkts)	Integer	Total transaction packet count
Source bytes (Sbytes)	Integer	Source/destination bytes count
Destination bytes (Dbytes)	Integer	Destination/source bytes count
Total bytes (TBytes)	Integer	Total transaction bytes count
Source load (Sload)	Float	Source bits per second
Destination load (Dload)	Float	Destination bits per second
Total load (Toad)	Float	Total bits per second
Source rate (Srate)	Float	Source packets per second
Destination rate (Drate)	Float	Destination packets per second
Total rate (Trate)	Float	Total packets per second
Source loss (Sloss)	Float	Source packets retransmitted or dropped
Destination loss (Dloss)	Float	Destination packets retransmitted or dropped
Total loss (Tloss)	Float	Total packets retransmitted or dropped
Total percent loss (Ploss)	Float	Percent packets retransmitted or dropped

TABLE 2 Features in the dataset captured using Argus

find the topology of the target network. The goal of this attack is to get a list of the devices deployed on the target network as well as their vulnerabilities. Each reconnaissance attack performed against the test bed is described in Calderon [42].

- Command injection: In this attack, hackers try to send malicious Modbus commands to affect the normal production process. Some examples, in our case the tank system, include turning on the pump and turning off the sensors remotely, and making the water level exceed the threshold level without sending an alarm to the system's operator.
- DoS and DDoS: These attacks target the availability of the PLC and make all services unavailable. This can result in a severe impact on the system's production.

The attacks shown in Table 3 were carried out using the Linux BlackArch Penetration Testing Distribution [45]. All data generated during the attacks, as well as regular traffic (without attacks), were gathered and recorded by the Argus server, as shown earlier in Figure 5. The observations consist of more than 40 features of the network traffic. However, we selected 19 features for use by the proposed IDS. Statistical information of the captured traffic is listed in Tables 4 and 5.

4.4 | Network flow labelling process

The DL algorithm used in our flow-based IDS is an ANN model. The ANN is a supervised algorithm. This means that it is necessary to have a labelled dataset to train and test the model. Thus, all network flow stored by the Argus server is classified and labelled in accordance with the type of attacks described in Table 3. Figure 8 shows a sample of the network flow labelled in our dataset.

Our dataset is composed of normal flows and eight different types of attacks, as described in Table 3. As shown in Figure 8, column traffic of the dataset specifies whether the network flow is normal (without attack) or an attack. Each attacked flow is classified by its type.

5 | PROPOSED ONLINE INTRUSION DETECTION SYSTEM FOR INDUSTRIAL CONTROL SYSTEM/SUPERVISORY CONTROL AND DATA ACQUISITION SYSTEMS

The flow-based IDS analyzes network flow to detect anomalies in the network instead of conducting deep-packet inspection. Figure 9 presents the proposed flow-based IDS framework.

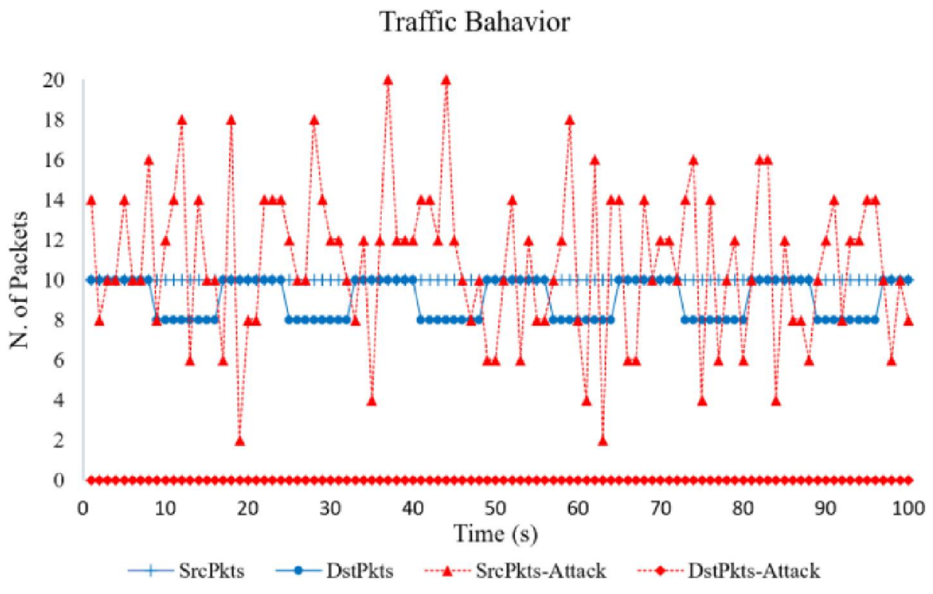


FIGURE 7 Example of traffic behaviour

As shown in Figure 9, the IDS framework consists of two main modules: offline and online. The offline module is described next.

5.1 | Offline module

The offline module is used to train and test the DL algorithm. This is made in two phases (training and test), as shown in Figure 9. The flow-based dataset, defined in Section 4, is used in the offline module. The dataset is split into two sets in which 80% of the dataset is used to train the model (training data in the training phase) and 20% is used to test and evaluate the model (testing data in test phase). The training phase consists of the following steps:

(1) Training data: part of the dataset to be used in the training process
(2) Feature extraction: extracts features defined for each flow stored in the training data
(3) Feature normalisation: features extracted in the previous step are normalized for use as input to the DL algorithm
(4) DL algorithm: the ANN algorithm is effectively trained using the labelled dataset. Any other DL algorithm could be used in this step.

The features normalisation is made using the formula:

$$x_{normalized} = \frac{(x - u)}{s} \tag{5}$$

where x is the raw data that will be normalized; u is the mean of the data and s is the standard deviation of the data.

As mentioned in Section 4.4, each row of the training dataset is labelled as an attack or not an attack. Afterwards, the DL will learn the parameters of a predictive model. When the DL model is trained, the evaluation of the trained model starts. This is made in the test phase, as shown in Figure 9. The test phase consists of the following steps:

(1) Testing data: part of the dataset used in the test process
(2) Feature extraction: extracts features defined for each flow stored in the test dataset
(3) Features normalisation: the features extracted in the previous step are normalized for use as input to test the DL model
(4) Prediction: using the trained model, predictions are made to verify whether a network flow is normal or an attack. In this case, the testing data are fed to DL algorithms without labels, and the model predicts their labels. All predicted results are logged in a database.
(5) Then, the confusion matrix showed in Table 1 is calculated, and the performance of the DL model is evaluated based on the metrics described in Section 3.2. The confusion matrix is produced based on whether a sample from either class is labelled correctly.

5.2 | Online module

The online module deploys the DL model previously trained and evaluated using the offline module in real time. The online module is connected to the Argus server. Therefore, all of the network flow monitored by the Argus server is used as input to the online DL model.

This module consists of the following steps:

(1) SCADA network traffic: the network flow is read from the Argus server in real time. These flows are unseen and

TABLE 3 Attacks carried out against our test bed

Attack Category	Used Tool	Attack Name	Attack Description
Reconnaissance	Nmap	Port Scanner [32, 33]	This attack is used to identify common SCADA protocols on the network. Using the Nmap tool, packets are sent to the target at intervals of 1–3 s. The TCP connection is not fully established, so the attack is difficult to detect by intrusion detection systems based on rules.
Reconnaissance	Nmap	Address scan [38]	This attack is used to scan network addresses and identify the Modbus server address. Each system has only one Modbus server, and disabling this device would collapse the whole SCADA system. Thus, this attack tries to find the unique address of the Modbus server so that it can be used for further attacks.
Reconnaissance	Nmap	Device identification [39]	This attack is used to enumerate the SCADA Modbus slave IDs on the network and to collect additional information such as vendor and firmware from the first slave ID found.
Reconnaissance	Nmap	Device identification [39]	This attack is similar to the previous attack. However, the scanning uses an aggressive mode, which means that the additional information about all slave IDs found in the system is collected.
Command injection	Metasploit Modbus client	Exploit [40]	Exploit is used to read the coils values of the SCADA devices. The coils represent the ON/OFF status of the devices controlled by the programmable logic controller, such as motors, valves, and sensors [34].
DoS	Synflood	DoS SYN Flooding [37]	In this attack, initial connection request (SYN) packets are sent to the PLC and HMI ports, causing the targeted device to respond to legitimate traffic sluggishly or not at all.
DDoS	Bonesi	DDoS botnet [37]	In this attack, many packets coming from several Internet protocols are sent to PLC and HMI computer.
DDoS	Hyde	DDoS with spoofing [37]	This attack is similar to the previous one with a spoofing attack added.

Abbreviations: HMI, human–machine interface; SCADA, supervisory control and data acquisition; SYN, synchronize; TCP, Transmission Control Protocol.

TABLE 4 Statistical information on the traffic during the reconnaissance and command injection attacks

Measurement	Value
Duration of capture	25 h
Dataset size	1.27 GB
Number of observations	7,049,989
Average data rate	419 kbit/s
Average packet size	76.75 bytes
Port scanner attack packets	0.0003%
Address scan attacks packets	0.0075%
Device identification attack packets	0.0001%
Device identification attack packets (aggressive mode)	4.9309%
Exploiting attack packets	1.1312%
Total attack packets	6.0700%
Total normal packets	93.9300%

TABLE 5 Statistical information on traffic during the DoS attacks

Measurement	Value
Duration of capture	25 Hours
Dataset size	1.25 GB
Number of observations	7,882,253
Average data rate	419 kbit/s
Average packet size	76.75 bytes
DoS SYN flooding attack packets	0.044%
Distributed DoS botnet attack packets	0.009%
Distributed DoS with spoofing attack packets	0.03 %
Total attack packets	8.590%
Total normal packets	

Abbreviation: DoS, denial of service; SYN, synchronize.

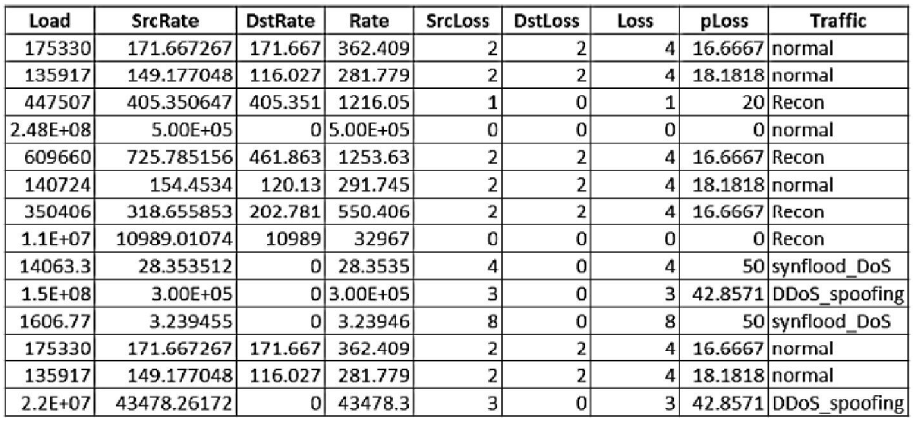
different from the flow-based dataset used in the offline module

(2) Feature extraction: extracts the same features as those in the offline phase from the network flow

(3) Features normalisation: extracted features are normalized for use as input to the deployed DL algorithm

(4) Online predictions: the model predicts, in real time, whether the network flow is an attack. All prediction results are logged in the dataset

(5) The confusion matrix is built again based on whether the model has classified each sample correctly or incorrectly. Attacks described in Table 3 were conducted against our SCADA system test bed at specified times. Thus, it is possible to verify the detection (attack or not attack) in real time and compare the online results with the offline results.



Load	SrcRate	DstRate	Rate	SrcLoss	DstLoss	Loss	pLoss	Traffic
175330	171.667267	171.667	362.409	2	2	4	16.6667	normal
135917	149.177048	116.027	281.779	2	2	4	18.1818	normal
447507	405.350647	405.351	1216.05	1	0	1	20	Recon
2.48E+08	5.00E+05	0	5.00E+05	0	0	0	0	normal
609660	725.785156	461.863	1253.63	2	2	4	16.6667	Recon
140724	154.4534	120.13	291.745	2	2	4	18.1818	normal
350406	318.655853	202.781	550.406	2	2	4	16.6667	Recon
1.1E+07	10989.01074	10989	32967	0	0	0	0	Recon
14063.3	28.353512	0	28.3535	4	0	4	50	synflood_DoS
1.5E+08	3.00E+05	0	3.00E+05	3	0	3	42.8571	DDoS_spoofing
1606.77	3.239455	0	3.23946	8	0	8	50	synflood_DoS
175330	171.667267	171.667	362.409	2	2	4	16.6667	normal
135917	149.177048	116.027	281.779	2	2	4	18.1818	normal
2.2E+07	43478.26172	0	43478.3	3	0	3	42.8571	DDoS_spoofing

FIGURE 8 Example of network flow labelled in our dataset

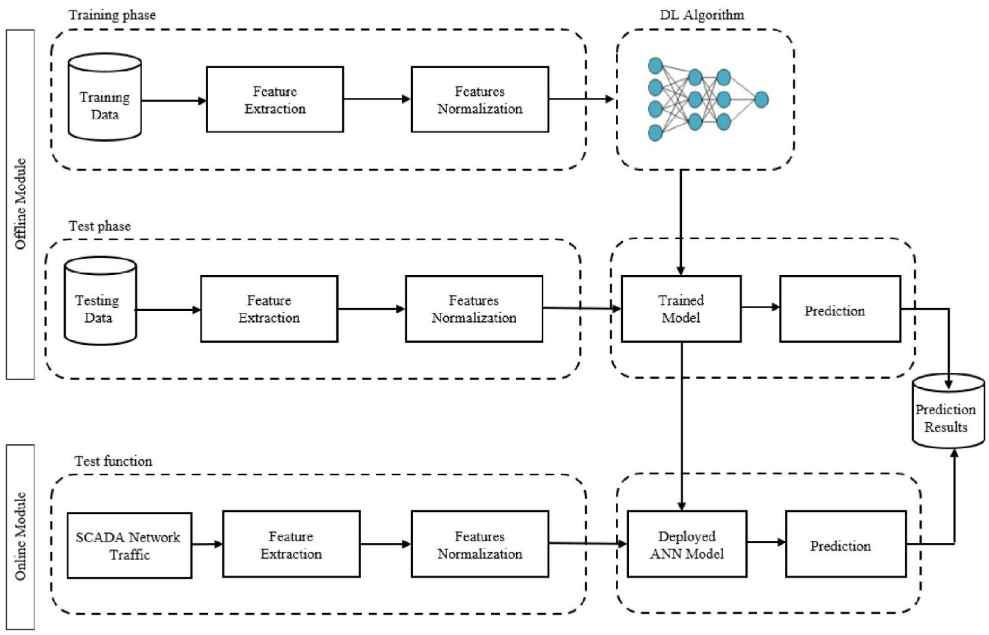


FIGURE 9 Flow-based intrusion detection system framework

Figure 10 presents a flowchart of the online module.

The classification results are listed in a confusion matrix and used to calculate the performance of the DL model.

5.3 | Artificial neural network model

As described in Section 3.1, the DL algorithm used in this work is the ANN. Figure 11 illustrates the architecture used in which each selected feature (Table 2) is composed of an input to the ANN input layer.

As presented in Figure 11, the ANN architecture is defined with one input layer, two hidden layers, and one output layer. The input layer neurons are connected to the first hidden layer, which in this example consists of nine neurons. The first hidden layer is connected to the second hidden layer, also composed of nine neurons. The second hidden layer is connected to the output layer consisting of one neuron (the answer neuron). The activation function used in the first and second layer is the ReLU function, and the output layer uses the sigmoid activation functions.

Current research states that the hidden layers of a neural network should use ReLU activation [36]. One reason for this is the computational performance and backpropagation training. The ReLU activation function has better performance than other activation functions for hidden layers; this is because the ReLU activation function is a linear, non-saturating function, as shown in Figure 12. It is defined as:

$$R(x) = max(x, 0),$$

where, $x = \sum_i w_i x_i$. Here x_i is the ith feature value and w_i is the normalising weight for that feature.

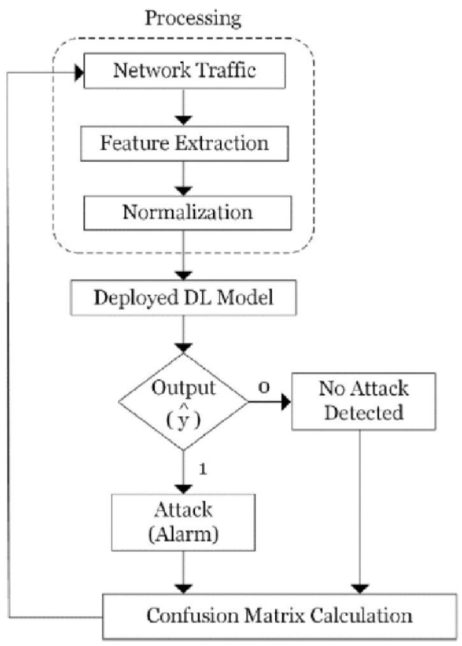


FIGURE 10 Online intrusion detection system framework evaluation

The sigmoid function (or logistic function) is a non-linear activation function. This makes the result of a neuron always between 0 and 1. We use the sigmoid function in our output layer. As mentioned, our IDS analyzes network traffic behaviour to identify whether an attack is happening, which is a classification problem. Figure 13 shows a graph of a sigmoid function and the equation used in the output layer:

6 | NUMERICAL RESULTS AND DISCUSSION

The empirical results of the proposed IDS are provided in this section. The performance of the online module is compared with the performance of the offline module.

Figure 14 shows the results of the accuracy metric, in which the performance of detecting the DOS attacks and reconnaissance/command injection (RCI) attacks are compared. To calculate these results, we used Equation (1). As mentioned, accuracy shows the percentage of total traffic that is classified correctly.

As shown in Figure 14, accuracy in detecting attacks decreases during the online phase. For DoS attacks, in the online phase, this decrease is minimal. However, for RCI attacks, the difference is more substantial. Despite this, the online detection results can be considered satisfactory. This is because RCI attacks are tough to detect, and also, in the real world, the amount of traffic generated by RCI attacks is much less compared with the amount of traffic generated by the DoS attack. Therefore, they are more challenging to detect. We need more training data to increase online accuracy when it comes to detecting RCI attacks.

As in any real-world scenario, the datasets used to train and test the security are imbalanced. It is well-known that accuracy is not a useful metric to evaluate the performance of imbalanced datasets [11]. In this case, other metrics are required to analyse the performance of the learning models further. The FAR results are shown in Figure 15. The FAR metric represents normal traffic that has been erroneously classified as abnormal traffic by the model (Equation 2).

Because this metric represents the percentage of false alarms, the lower FAR value is considered better. Again, the offline results show better performance compared with the online results. The performance in detecting the DoS attack online is better than that in detecting RCI attacks. This is again the result of the lower number of training packets for the RCI attacks compared with the standard packets.

Figure 16 presents the UND metric. This metric (Equation 4) indicates the percentage of abnormal traffic that is erroneously classified as normal traffic. For cybersecurity applications, this metric reveals attacks that happened without being detected by the system, making this metric more critical than the FAR metric. Furthermore, in our imbalanced dataset, if a model is biased towards classifying almost all traffic as normal, this metric would show how biased the model is.

As shown in Figure 16, the offline results are small for both types of attacks. The online performance is lower but acceptable. This shows that the selected features can detect DoS attacks even in an imbalanced dataset.

Figure 17 shows the ROC curve. An ROC curve defines how well the model can separate samples from both classes. This curve plots the trade-off between sensitivity (or TP rate) and specificity (1 – FP rate). Models that give curves closer to the top left corner indicate better performance. Because the ROC curve does not depend on the number of samples in each class, the imbalanced issue does not affect its results. The ANN algorithm shows excellent performance in detecting DoS attacks in both scenarios (offline and online). For the RCI attacks, our developed IDS shows excellent performance in the offline evaluation and satisfactory performance in the online evaluation.

For detection latency, we calculated the average time that the proposed IDS detects the attacks in real time. As mentioned, the IDS is connected to a port on the switch and receives a copy of the network flow. Thus, we define latency as the time between the occurrence of an attack and its detection by our IDS. The average time to detect an attack is approximately 55 ms. This time is composed of feature extraction, feature normalisation, and DL model analyses, as shown in Figure 10.

7 | CONCLUSION

We have presented a flow-based dataset to support IDS research for SCADA systems.[43, 44] Furthermore, a flow-based IDS system using a DL algorithm is proposed. Two different cyberattacks related to ICS/SCADA systems were carried out in our SCADA system test bed. Then, the dataset

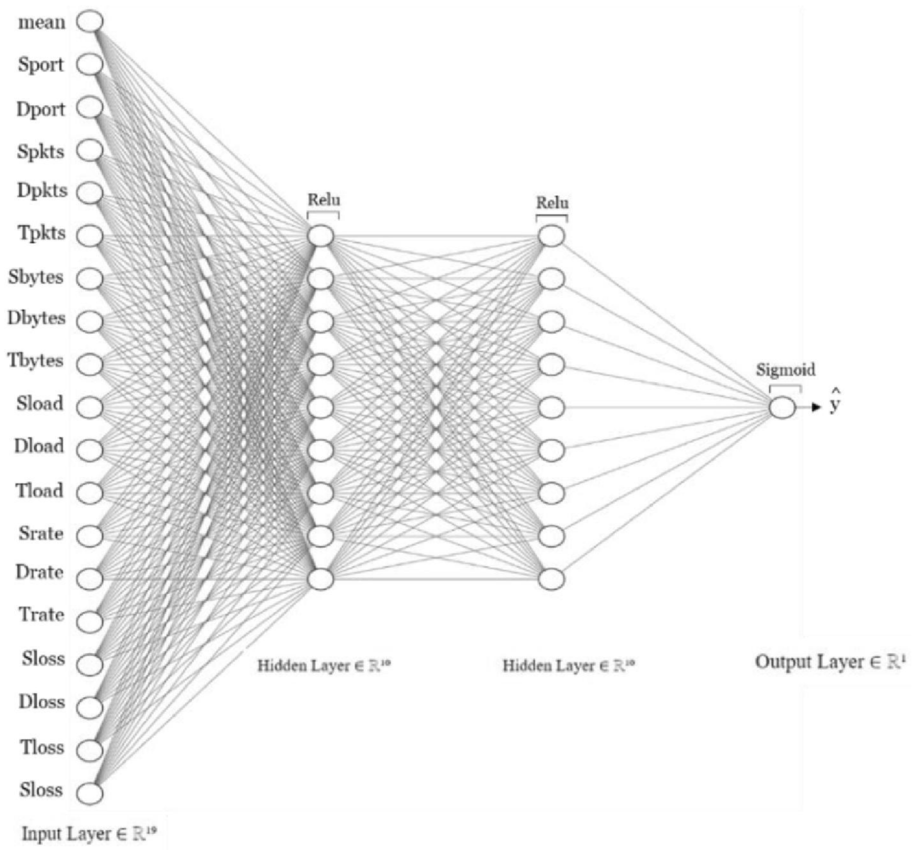


FIGURE 11 Architecture of the used artificial neural network model

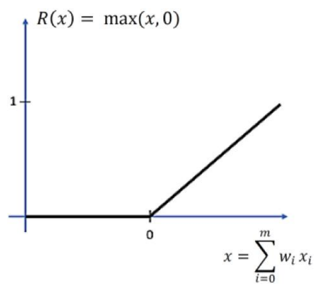


FIGURE 12 Rectified linear unit activation function

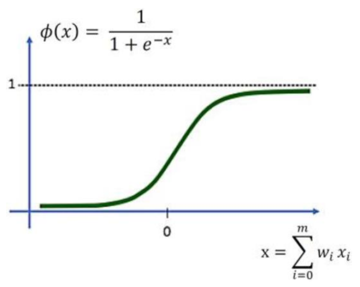


FIGURE 13 Sigmoid activation function

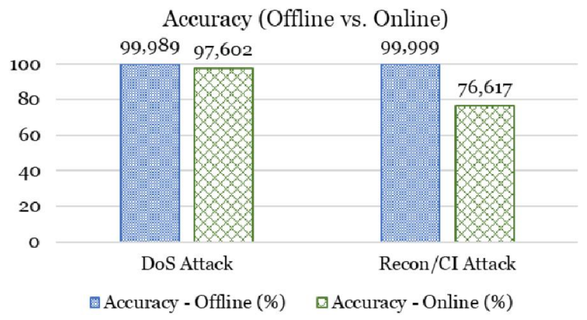


FIGURE 14 Accuracy results

was used to train and test the DL algorithms. The performance of the IDS was evaluated on the SCADA network in both offline and online modes.

The results show that the DL algorithms used in our IDS and deployed on our SCADA test bed have excellent performance during both offline and online phases. In the online evaluation, the performance obtained during the DoS attacks also showed reliable performance, close to the offline evaluation. Meanwhile, for RCI attacks, the online performance was lower than the offline performance. We attribute this difference to the unique characteristics of these attacks. More specifically, the number of attack packets generated during the RCI attacks is tiny compared with the DoS attacks. This was done to mimic the real-life scenarios in which DoS attacks are more frequent than RCI attacks.

As future work, we plan to conduct more experiments with RCI attacks and add new kinds of attacks (e.g., man-in-the-

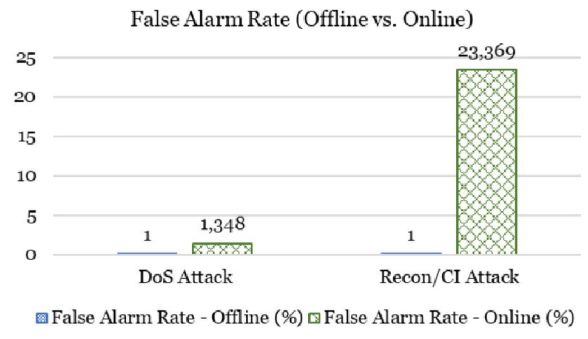


FIGURE 15 False alarm rate results: lower is better

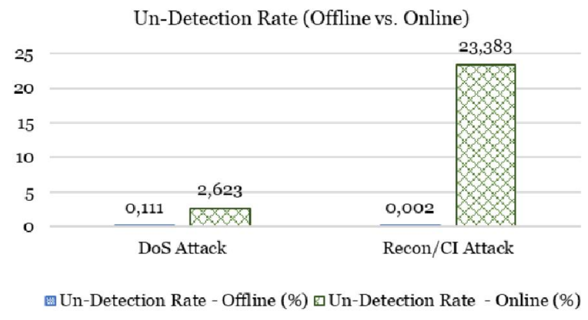


FIGURE 16 Undetected rate results: lower is better

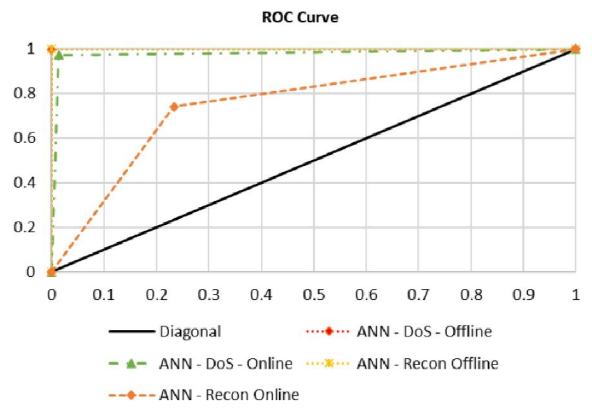


FIGURE 17 Receiver operating characteristic curve

middle attack) to the dataset. Moreover, experiments using unsupervised DL algorithms will be considered.

ACKNOWLEDGEMENTS

This work has been supported under Grant ID NPRP10-0206-170360 funded by the Qatar National Research Fund (QNRF), NSF Grants CNS-1718929 and 2017/01055-4 from the São Paulo Research Foundation (FAPESP). The statements made here are solely the responsibility of the authors. The authors would like to thank the Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP), São Paulo Research Foundation (FAPESP), Washington University in Saint Louis, and Qatar University.

ORCID

Marcio Andrey Teixeira https://orcid.org/0000-0003-1466-3596
Maede Zolanvari https://orcid.org/0000-0003-1428-7770
Nader Meskin https://orcid.org/0000-0003-3098-9369

REFERENCES

1. Stouffer, K., Falco, J., Scarfone, K.: Guide to industrial control systems (ICS) security. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r1.pdf. Accessed 3 December 2020
2. Flatt, H., et al.: Analysis of the cyber-security of industry 4.0 technologies based on RAMI 4.0 and identification of requirements. In IEEE 21st International Conference on Emerging Technologies and Factory Automation, Berlin. 1–4 (2016)
3. Kovacs, E.: Over 400 ICS Vulnerabilities Disclosed in 2019: Report, Security Week. https://www.securityweek.com/over-400-ics-vulnerabilities-disclosed-2019-report/. Accessed 3 December 2020
4. Paganini, P.: Analysis of ICS flaws disclosed in 2019. Security Affairs. https://securityaffairs.co/wordpress/98211/breaking-news/dragos-report-ics-flaws-2019.html. Accessed 3 December 2020
5. Zolanvari, M., et al.: Machine learning-based network vulnerability analysis of industrial Internet of Things. IEEE Internet Things J. 6(4), 6822–6834 (2019)
6. Zhu, B., Sastry, S.: SCADA-specific intrusion detection/prevention systems: a survey and taxonomyn. In: Proceedings of the 1st Workshop on Secure Control Systems. I (2010). SCS
7. Anna, L.B., Erhan, G.: A survey of data mining and machine learning methods for cybersecurity intrusion detection. IEEE Communications Surveys & Tutorials. 18(2), 1153–1176 (2016). Second Quarter
8. Mukkavilli, S.K., Shetty, S., Hong, L.: Generation of labelled datasets to quantify the impact of security threats to cloud data centres. J Inf Secur. 7(April), 172–184 (2016)
9. Ring, M., et al.: A survey of network-based intrusion detection data sets. Comput Secur. 86, 147–167 (2019)
10. Teixeira, M.A., et al.: SCADA system testbed for cybersecurity research using machine learning approach. Future Internet. 10(76) (2018)
11. Zolanvari, M., Teixeira, M.A., Jain, R.: Effect of imbalanced datasets on the security of industrial IoT using machine learning, in IEEE Intelligence and Security Informatics (ISI), pp. 112–117. Miami (2018)
12. Sumeet, D., Xian, D.: Data mining and machine learning in cybersecurity. CRC Press (2011)
13. Tian, J., et al.: Moving target defence approach to detecting stuxnet-like attacks. IEEE Trans. Smart Grid. 11(1), 291–300 (2020)
14. Flaus, J.M.: Cybersecurity of industrial systems, Ed. Wiley (2019)
15. Ackerman, P.: Industrial Cybersecurity: Efficiently secure critical infrastructure systems, 1 ed. Packt Publishing (2017)
16. Odewale, A.: Implementing secure architecture for industrial control systems. Proceedings of the 27th COREN Engineering Assembly, Abuja. 6–8 (2018)
17. Security standards, ISA/IEC 62443 standards. International Society of Automation (ISA). https://www.isa.org/intech-home/2018/september-october/departments/new-standard-specifies-security-capabilities-for-c/, Accessed 20 October 2021
18. ENISA recommendations to IT Industry. https://www.enisa.europa.eu/publications/enisa-position-papers-and-opinions/enisa-recommendations-to-it-industry/view
19. Bhamare, D., et al.: Khaled Khan, and Nader Meskin, cybersecurity for industrial control systems: a survey In Computers & Security, 89 (2020)
20. WUSTL-IIOT-2018 Dataset for ICS (SCADA) Cybersecurity Research, http://www.cse.wustl.edu/~jain/iiot/index.html, Accessed 3 December 2020
21. Ring, M., et al.: A survey of network-based intrusion detection data sets (2019). arXiv:1903.02460. https://arxiv.org/abs/1903.02460. Accessed 3 December 2020

22. Ferrag, M.A., et al.: Deep learning for cybersecurity intrusion detection: approaches, datasets, and comparative study. J. Infor. Security App. 50, 2214–2126 (2020)
23. Malowidzki, M., Berezinski, P., Mazur, M.: Network intrusion detection: Half a kingdom for a good dataset, In NATO STO SAS-139 Workshop. Portugal (2015)
24. He, Y., Mendis, G.J., Wei, J.: Real-time detection of false data injection attacks in smart grid: a deep learning-based intelligent mechanism. IEEE Trans. Smart Grid. 8(5), 2505–2516 (2017)
25. Yan, W., Yu, L.: On accurate and reliable anomaly detection for gas turbine combustors: a deep learning approach. In: Annual Conference of the Prognostics and Health Management Society, San Diego, 1–8 (2015)
26. Niyaz, Q., et al.: A deep learning approach for network intrusion detection. In The 9th EAI International Conference on Bio-inspired Information and Communications systems, New York. pp. 21–26. (2015)
27. Yin, C., et al.: A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access. 5, 21954–21961 (2017)
28. Lemay, A., Fernandez, J.M.: Providing SCADA network data sets for intrusion detection research. Proceedings of the 25th USENIX Security Symposium (2016)
29. Najafabadi, M.M., et al.: Deep learning applications and challenges in big data analytics. J. Big Data, Spring Open J. 2–21 (2015)
30. Dahl, G., et al.: Phone recognition with the mean-covariance restricted Boltzmann machine. In: Proceedings of the 23rd International Conference on Neural Information Processing Systems. 1, 469–477 (2010)
31. Hinton, G., et al.: Deep neural networks for acoustic modelling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag. 29(6), 82–97 (2012)
32. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems. 25, 1106–1114 (2012)
33. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. 18(7), 1527–1554 (2006)
34. Socher, R., et al.: Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In: Proceedings of the 24th International Conference on Neural Information Processing Systems, 801–809 (2011)
35. Bordes, A., et al.: Joint learning of words and meaning representations for open-text semantic parsing. In: Proceedings of 15th International Conference on Artificial Intelligence and Statistics. 22, 127–135 (2012)
36. Ide, H., Kurita, T.: Improvement of learning for CNN with ReLU activation by sparse regularisation. In: International Joint Conference on Neural Networks (IJCNN), 14–19 (2017)
37. Prasad, N., Singh, R., Lal, S.P.: Comparison of backpropagation and resilient propagation algorithm for spam classification. In: 5th International Conference on Computational Intelligence, Modelling, and Simulation, Seoul. 29–34 (2013)
38. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Inf Process Manag. 45, 427–437 (2009)
39. The Argus sensor, https://openargus.org/, Accessed 3 December 2020
40. Pontes, C., et al.: A new method for flow-based network intrusion detection using inverse statistical physics (2019). arXiv:1910.07266
41. Morris, T.H., Gao, W.: Industrial control system cyber-attacks. In: Proceedings of 1st International Symposium for ICS & SCADA Cyber Security Research, Swindon. 22–29 (2013)
42. Calderon, P.: Nmap: network exploration and security auditing cookbook – second edition: network discovery and security scanning at your fingertips, pp. 542–586. Packet Publishing (2017)
43. Nmap security, https://nmap.org/, Accessed 3 December 2020
44. Vulnerability, Database, E.: Modbus Client Utility. https://www.rapid7.com/db/modules/auxiliary/scanner/scada/modbusclient. Accessed 3 December 2020
45. Black Arch Linux Penetration testing distribution, https://blackarch.org, Accessed 3 December 2020

How to cite this article: Teixeira, M.A., et al.: Flow-based intrusion detection algorithm for supervisory control and data acquisition systems: a real-time approach. IET Cyber-Phys. Syst., Theory Appl. 6(3), 178–191 (2021). https://doi.org/10.1049/cps2.12016