A novel bi-anomaly-based intrusion detection system approach for industry 4.0

Salwa Alem, David Espes, Laurent Nana, Eric Martin, Florent de Lamotte

Future Generation Computer Systems 145 (2023) 267–283

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Review article

A novel bi-anomaly-based intrusion detection system approach for industry 4.0

Salwa Alem [a,b,*], David Espes [b], Laurent Nana [b], Eric Martin [a], Florent De Lamotte [a]

[a] University Bretagne Sud, Lab-STICC (Laboratoire des Sciences et Techniques de l'Information de la Communication et de la Connaissance), Lorient, France

[b] University of Western Brittany, Lab-STICC (Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la Connaissance), Brest, France

ARTICLE INFO

Article history:
Received 22 August 2022
Received in revised form 17 February 2023
Accepted 16 March 2023
Available online 22 March 2023

Keywords:
Cyber physical system (CPS)
Anomaly-based intrusion detection system (IDS)
Manufacturing executive system (MES)
ISA-95 industrial standard
Neural networks (NN)

ABSTRACT

Today, industry 4.0 is becoming a major target for cybercriminals due to its hyper-connectivity. Fortunately, there are several advanced means of securing industrial systems such as Intrusion Detection Systems (IDS). However, one of the main limitations of industrial IDS is the high rate of false positives and how to distinguish a real attack from an industrial failure. This paper deals precisely with the two latter points and proposes a way to reduce the rate of false positives and to distinguish attacks from industrial failures. The proposed approach combines two kinds of IDS using Neural Network (NN) through a Decision Making System (DMS). It was tested on a real industrial environment. The performance results are promising with a high percentage of accuracy and a low false positive rate.

Contents

* Corresponding author at: University Bretagne Sud, Lab-STICC (Laboratoire des Sciences et Techniques de l'Information de la Communication et de la Connaissance), Lorient, France.
E-mail address: salwa.alem@univ-ubs.fr (S. Alem).

https://doi.org/10.1016/j.future.2023.03.024

1. Introduction

Nowadays, Industrial Control Systems (ICS) exist in many different industrial sectors such as meatpacking, chemistry, construction, automotive, electronics industry. But also in vital industrial sectors such as energy, health, military and food sectors. Therefore, the suspension or the stopping of these systems could be costly for industrialists and cause consequent damage. Today, securing such equipment becomes more than necessary. Over the past decade, industry has become the center of attackers' focus and has been the victim of several attacks starting with Stuxnet, Black Energy, WannaCry. This wave of attacks has been succeeded by several ransomware attacks in 2020 during the coronavirus pandemic, especially with the increase in the number of remote workers and a lack of security in this new working model [1]. Kaspersky's ICS CERT researchers forecast a list of attacks likely to target industries in 2023 [2]. Among these attacks are phishing pages and emails, Torjans, N-day vulnerabilities, attacks on cloud services, exploiting vulnerabilities in legitimate software, the spread of malware via removable media ...

This cyber-criminality phenomenon is favored with the emergence of the industry 4.0. This 4th industrial revolution is characterized by the convergence of the worlds of Information Technology (IT) and Operation Technology (OT), the huge amount of generated data, the use of Cloud as new storage means. All these reasons increase the risk of cyber attacks in industry.

Fortunately, there are several solutions to secure the industry and its equipment. Among these mechanisms, we mention firewalls, anti-virus, auditing processes and IDS. Each of these securing mechanisms has a specific role such as detecting and removing malware, preventing unauthorized access or detecting intrusions by IDS. These latter give visibility to the system's activities, which allows a timely detection and response to any suspicious events [3,4].

Two kinds of IDS approaches exist in the literature, which are signature-based and anomaly-based IDS. This last one is more efficient in detecting advanced and zero-day attacks and is composed of two other IDS types, which are specification-based IDS and behavioral-based IDS. Each of these IDS has advantages and limitations.

In the industrial intrusion detection field, the main issue is how to distinguish a malfunction of the industrial process from a real intrusion. Malfunction in this paper refers to the process disruption such as equipment failure, process stopping, recipe modification...

In this paper, the proposed approach deals with this specific point. Commonly, the researchers carry out hybridization between signature-based and anomaly-based IDS. In this paper, the proposed approach consists of carrying out this hybridization between two IDS belonging to the same category: anomaly-based IDS. Therefore, this paper proposes an efficient IDS composed of a specification-based IDS and a behavioral-based IDS. The specification-based IDS uses an industrial standard called the ISA-95 as the main basis. The purpose of this IDS is detecting process anomalies and measuring and quantifying their impacts on the system. The behavioral-based IDS uses the network traffic that is captured from the industrial architecture. This latter is learned by a supervised neural network algorithm. The main mission of this IDS is to distinguish malfunction from a real attack and reduce the rate of false positives.

Several machine learning algorithms can be used to train and test behavioral-based IDS. However, in this paper, a neural networks model has been chosen, based on this study context which consists of how to detect anomalies in an industrial process and determine their nature with a high accuracy. This industrial process deals with a huge volume of data and does not allow any delay in the execution of its steps. Thanks to its parallelism characteristic (neuron parallelism and neuron connections parallelism), results could be obtained rapidly. Therefore, the real time constraint is overcome [5]. Because of its strong discrimination and generalization capabilities, the classification could be performed with significant success rates.

In [6,7], the authors draw a global landscape of defense mechanisms in industrial systems and pay particular attention to behavioral-based IDS by citing several works using this means of defense. However, none of the cited works used hybridization between two anomaly-based IDS. In addition, the authors also mention the problem of IDS, which consists of determining the nature of the detected anomalies. One of the novelties of this work is to deal with this specific point.

Therefore, the novelty of this paper consists of proposing a hybridization inside the anomaly-based IDS category and not only inside the IDS category. Commonly, we propose hybrid IDS between misuse and anomaly-based IDS. In this work, two anomaly-based IDS have been hybridized. This hybridization remains the most efficient method in distinguishing a malfunction in a process from a cyber-attack and perform good results in terms of precision and False Positive Rate (FPR). In addition, thanks to this hybridization, the proposed approach allows us to distinguish a malfunction process anomalies from the real intrusions and classify these latter into temporal or sequential categories.

Moreover, the approach targets an industrial level: Manufacturing Executive System (MES) that no research work has targeted before. Furthermore, this paper fills a gap in the industrial datasets by proposing another one with more features, which is also publicly available. This dataset has been captured and built to implement this approach. A paper has been published to give more details regarding this industrial dataset.

This paper is organized as follows. Section 2 presents the related works. In Section 3, the first module of the global approach is presented with its principles, its basis, and its results. In Section 4, the second module that is the specification-based IDS is detailed by outlining its basis i.e, the MESA model from the ISA-95 standard with its principle and its results. The entire approach called BI-ANOmaly IDS (BIANO-IDS) is then detailed in Section 5 with a global view of its principle followed by an outline of its components, an exploration of the Decision Making System (DMS) with its principle, its rules of function and its graphical interface and results. Finally, the paper ends with a conclusion section.

2. Related works

Intrusion Detection Systems (IDS) are security mechanisms put in place to monitor network traffic, suspicious activity, policy violations and to alert the system administrator when such activities or violations are detected. In the literature, two kinds

of IDS exist: Host-based Intrusion Detection (HIDS) and Network Intrusion Detection System (NIDS). HIDS use the machine logs to analyze the system activity and the second ones use network traffic logs in their analysis. According to the approach used, two kinds of IDS are distinguished: signature-based IDS and anomaly-based IDS:

- Signature-based IDS: they are based on a set of attack descriptions or signatures [8]. It consists of searching sequences and patterns that match a particular known attack signature. Signature-based IDSs are the most commonly used in industry.
- Anomaly-based IDS are composed of two types:
  - Behavioral-based IDS: These IDS model the normal behavior of a computer system [9]. Their implementation always includes a learning phase during which they will discover and learn the normal model of the monitored system. Once this learning step has been carried out, these IDSs will signal the divergences from the reference model.
  - Specification-based IDS: For this kind of IDS, the behavior normality of a system is described through its properties, its specifications and the security policy. Any deviation from these systems' specifications is considered as a security violation [10,11]. For instance, a heating system is working inside a temperature range between 40 °C and 250 °C; anything outside of this range is considered abnormal.

In this part of this paper, a focus has been put on the industrial IDS including both signature-based IDS and anomaly-based IDS.

2.1. Signature-based IDS

The first signature-based IDS were developed and implemented a long time ago in [8,9,12]. Since their emergence, several researchers have enhanced and improved their detection techniques by proposing diverse and varied approaches: distributed, innovative or basic ones. A new distributed approach based on Mobile Agent (MA) is detailed in [13]. Their distributed IDS is composed of four agents: Sniffer, Filter, Analyzer and Decision Agents. The Sniffer Agent intercepts and logs the real-time network events; the Filter Agent, which collects, treats and categorizes the detected events; the Analyzer Agent, which processes the filtered events by using a pattern matching method to check if there are similarities between the filtered and malicious packets and; finally, the Decision Agent, which uses the administrator knowledge for a deep detection. The authors simulated four attack kinds (Probe, R2L, U2R and Denial of Service (DOS)) and captured real traffic. They evaluated their approach by measuring the detection delay, false positives, and detection rate. Their performances seem good.

Another innovative approach is detailed in [14]. In this research work, the authors propose an innovative signature-based IDS, which combines a conventional detection method using known attack signatures and the system state checking methods. After analyzing the packets in the known signatures database, the IDS keeps in its memory a system state (a numerical representation of the system). Therefore, an alert is raised when the system reaches a critical state. For their implementation, the authors put in place a set of signatures for the Distributed Network Protocol (DNP3) and Modbus protocols, and device states. According to the evaluation, the results showed a high accuracy in detecting the simulated attacks.

Some other researchers have chosen a simple and a basic signature-based IDS approach such as in [15]. The proposed approach is composed of two modules. The first one is a filter, and the second one is an IDS. To detect intrusions, the latter uses the distance notion which leads an Industrial Control System (ICS) to a critical state. Therefore, by controlling this distance, an industrial system avoids that a system is leading to this critical state.

2.2. Anomaly-based IDS

Another type of IDS is proposed in the literature: anomaly-based IDS. This kind of IDS contains two other sub-categories: specification-based and behavioral-based IDS.

(1) Specification-based IDS: In [16], the authors propose a critical state-based IDS, which manages and monitors the evolution of Supervisory Control and Data Acquisition (SCADA) states. This approach is composed of three main modules to detect abnormal activity and lead a SCADA system to a critical state. These modules are: "System Description and Critical State Representation", where the system is described using Industrial State Modeling Language (ISML). ISML formalizes the condition-action of an industrial system. "A State Evolution Monitor" tracks the system states and shows how these states evolve. Using ISML, a virtual image of the managed system is created as an in-memory model. The latter is fed by network traffic in order to behave like a real system. The third module is "A Critical State Detector". By checking the Critical State (CS) rules defined using ISML language, this module detects whether the system goes through a CS or not.

In order to predict the critical state, the authors have added multidimensional metrics to previous modules. These metrics are: state–state distance, state-critical states distance and distance evaluation metrics. Their performance seems good but their proposed IDS targets a particular class of attacks aiming to lead an industrial system to a critical state.

In [17], the authors propose a new bi-anomaly-based IDS approach. The first step of this approach is to define a set of rules describing the normal behavior. They define five rules between rules related to the network, such as source IP, and others related to the process, such as operation time. After defining these rules, the authors model the normal behavior and consider any behavior which deviates from the normal model as an intrusion. They apply their approach to the Modbus protocol and use Bro to implement their rules. They test their approach in an emulated environment with a real Programmable Logic Controller (PLC). The authors rate the detection attack of their IDS as accurate and that results in a good performance.

In [11], a new specification-based IDS approach is proposed using the documentation of the networking system. The documentation is automatically used to reduce the human effort required to define the specification rules. The approach is composed of three steps: system discovery, features lookup, and rule definition. This IDS is implemented in a real environment using Building Automation and Control protocol (BACnet) and identifies process control errors and the level of danger caused by misconfiguration.

In [18], the authors present a new resilient IDS implemented in the Ethernet layer. It analyzes the vertical flows exchanged between the field devices level and SCADA system. The latter receives an alert if an intrusion is detected. This IDS is implemented for an electrical station using GOOSE protocol. The authors simulate Ethernet storms and usurp Generic Object Oriented Substation Events (GOOSE) attacks. After receiving alerts, the Intelligent Electronic Device (IED) control is rewritten to consider the alerts.

In [19], the authors propose an approach to mine specifications from attack execution traces and detect violations for a sequential process. They use Sequential Function Charts (SFC) which are suitable for this kind of process. Their specifications

consist of events and states of the evolution of both sensors and actuators. The proposed approach is composed of two steps. The first one is the mining and the inference of the specifications from traces captured between supervisors and PLC. The second step is a monitoring phase which raises an alert when a specification violation is detected. The authors propose a novelty which eliminates the redundancy to increase the alert's pertinence for a quick reaction by the operator.

In [20], a model-based approach for an IDS is proposed to protect a CPS. Their approach contains three stages: model construction, model transformation, and online implementation of the IDS. In the first step, the researchers define the system model using a formal modeling language, then this model is transformed into rules by using a model converter and finally, the generated rules are embedded in an open source IDS. Their results are not fully conclusive.

The authors [21] have proposed an approach similar to that proposed in this paper. The researchers propose a specification-based IDS using network traffic and process specifications. However, the limitation of their approach consists of having a high false positive rate, thus came our idea to hybridize two anomaly-based approaches.

In [22], the authors propose a specification-based IDS using Bro. They modified the latter by adding a DNP3 protocol analyzer to generate specific SCADA operation events. The semantics for each event are stored in the corresponding event handler. To analyze these semantics, they set up the protocol validation policies by defining the event handler in terms of Bro scripting. The policies interpreter script runs code to produce scan results, such as abnormal network activity alerts. The approach is tested in an experimental simulation. The authors believe that the proposed DNP3 analyzer holds potential to work in real SCADA systems.

(2) Behavioral-based IDS: In [23], another behavioral-based IDS is proposed by using a model-based approach. The main basis of this approach is building patterns that characterize the expected behavior of a system and detecting attacks on a system that runs off these models. In this approach, the authors use three techniques which consist of protocol-level models that define normal behavior using features characterizing the Modbus/TCP protocol. Then they use the Snort tool to define the rules, allowing the model violation detection. The authors then specify the expected communication patterns between network components and their access policies. Finally, they use a last detector based on a heuristic approach to detect any first time observed changes in servers or services. This last detector keeps a log of the whole components state to avoid duplicated alerts. The authors mention that the proposed approach is effective to monitor SCADA networks and detect attacks.

In [24], the authors present a PeriodAnalyser tool which uses an automated NIDS approach. This tool is based on message repetition and temporal information to define periodic cycles in the network traffic. Their approach is composed of three modules: multiplexer module, which analyzes the application headers to filter the Modbus/TCP, MMS packets and multiplexes them into different flows. Two kinds of flow are identified at this stage which are long-lived and short-lived connections. The second module is tokenizer, which transforms the filtered flows into a standard format independent of the protocol. And finally, a learner module is in charge of identifying all the cycles in the previously pre-processed flows. The authors use three traces of a real environment to evaluate the approach, which shows a good modeling accuracy.

In [25], the authors propose a new IDS approach to protect a SCADA system. They use a One-Class Support Vector Machine (OCSVM) classifier, which uses offline network traffic for its training step. The architecture is hierarchical since all IDS refer to a Security Information and Event Management (SIEM) database. Alerts are sent to the latter using the Intrusion Detection Message Exchange Format (IDMEF) standard. Their IDS is composed of four steps: training, classification, anomaly detection and sending of the IDMEF messages containing the incident source, their time and the alert classification. The accuracy of this IDS is good but still needs to be improved.

In [26], the authors propose an industrial IDS using an improved OCSVM model. They use particle swarm optimization to train their model with only one class. The approach is tested in a simulated environment. The efficiency of the IDS is high with 96% detection accuracy and a short training time.

In [27], the authors propose an IDS for a SCADA system aiming to detect both attacks against the process and those coming from the network. To detect attacks targeting the industrial process disruption such as Man In The Middle (MITM), replay and zero-day, they use a process-state validation. For that, they use a white-list and correlation scores to validate numerical and non-numerical parameters respectively. For other attacks such as Denial of Service (DoS) and buffer overflow, a non-parallel hyperplane-based fuzzy classifier is implemented. Their evaluation shows good accuracy compared to both Support Vector Machine (SVM) and fuzzy-based SVM classifiers. However, their approach does not provide the type of the attack or its location.

In [28], the authors outline a new hierarchically distributed intrusion detection scheme for industrial CPS, which is composed of three layers: the Perceptual Executive Layer (PEL), the Data Transmission Layer (DTL) and the Application Control Layer (ACL). The authors propose a detection technique for each layer. For PEL, they use Process Noise and Measurement Noise-Adaptive Kalman Filter (PNMN-AKF) with spare Bayesian and Relevant Vector Machine (RVM) classifiers to detect attacks. PNMN-AKF uses an estimation of the distributed system states. In the DTL layer, a network communication anomaly based detection system is proposed. It consists of the measuring of the distance between the actual network distribution characteristics and the normal ones. Kolmogorov–Smirnov (KS) divergence is used for this measurement. Finally, for the Application Control Layer (ACL), the authors use Cyclic Redundancy Check (CRC)-based identity authentication for an accurate detection, as compromising the application layer could be critical. This technique is coupled with a Regularized Sparse Deep Belief Network (RSDBN) model using the Restricted Boltzmann Machine (RBM) for its training step. Using an OPNET simulated environment, the efficiency of the IDS is demonstrated by a low false positives rate.

In [29], the authors propose a specific IDS for electrical substations. Their IDS contains four techniques: Access Control Detection (ACD), Protocol White-listing Detection (PWD), Model-Based Detection (MBD), and Multi-Parameter based Detection (MPD). ACD is responsible for a white-list strategy containing some network features such as Media Access Control (MAC) and Internet Protocol (IP) addresses. PWD only allows GOOSE/SV/IEEE 1588 packets. MBD is in charge of the definition of the normal behavior from Substation Configuration Description (SCD) and GOOSE protocol and considers all that deviates from this reference model as abnormal. MPD monitors sensitive electrical station parameters from which it carries out a deep detection for both external attacks and unintentional misuse. The authors evaluate their approach against several attacks such as DoS attack, Address Resolution Protocol (ARP) spoofing (Usurpation) and Man In The Middle (MITM). Their accuracy and process time seem good.

2.3. Industrial IDS for IoT equipment

Other researchers have focused on intrusion detection in Internet Of Things (IoTs), which are an integral part of Industry 4.0. In [30], the authors propose an approach called Intrusion Detection and Prevention Mechanism (IDPM). The IDMP consists of two layers, the first of which is based on learning the normal behavior of the system using a Random Neural Network (RNN), which takes diverse datasets and covers both valid and invalid cases as input parameters. The trained RNN model is then embedded in the base station of the IoT system to detect any anomalous behavior and to prevent its propagation. The second one is designed to detect a wide range of Illegal Memory Accesses (IMA) bugs and data integrity attacks. The proposed solution's also acts as a health monitoring system for the IoT sensor nodes by analyzing data transmitted to the base station. As in the case of any malfunction, the valid sensor node may either stop its operation and/or transmit invalid data to the base station. The RNN model has been trained to detect such cases as an intrusion and report them to the main server. The proposed solution effectiveness and performance overhead are measured for an existing IoT system consisting of sensor nodes transmitting data to a base station. Through an experimental setup, it is shown that without a proper security mechanism, it is possible to hack into the application running on the base station. Furthermore, it is also demonstrated that the base station successfully detected the presence of the malicious sensor node when the given IoT device is enabled with the proposed IDPM.

In [31], the IDS principle approach is designed to find any anomaly in the network by monitoring the characteristics of the neighboring nodes at one hop. The system learns and derives the normal behaviors of the monitored information. They use a distributed approach in which all of the observed information is locally managed in the nodes and the exchanges with the parent node appear only if an event is reported. The system has a configurable profile in which the criteria for critical tasks such as detection process interval, memory, threshold and grading parameters are defined. Their approach consists of four modules: the first one is in charge of the data collection and analysis. The second one processes the release of the report to the parent via the DPO (distress propagation object), the third one rejects the packets based on the status of the neighboring node, isolating the compromised node once the intrusion is detected and the fourth one is the BR module that, in the event of a positive anomaly decision, alerts the user. The authors did not disclose any details about their implementation or their results.

The authors propose a hybrid T-IDS for Sybil attacks in [32], it is a distributed cooperative and hierarchical IDS, involving Border Routers (BR), 6LoWPAN Border Router (6BR) and Monitoring Nodes (MN). Each actor has a specific goal. The BR holds the list of all of the Network's Nodes (NN) and their respective states and maintains the list of nodes allowed to access the network. In NN, each node is associated with a Trusted Platform Module (TPM) identifier, a node identifier associated with the TPM-ID, the status flag of the node (Mobile, Static), and the 6LoWPAN Border Router (6BR) prefix associated with the node after deployment. When a node wants to join the network, it must first be registered to the NNs list. BR also has a list of malicious nodes for all 6BR sub-nets. 6LoWPAN Border Router (6BR) which maintains three dynamic lists: the first list contains 6BR (6BRAN) area nodes. 6BRAN is developed and updated by the BR and transferred to 6BR in a secure channel. The second contains MObile Nodes (MON) and the third list contains MAlicious Nodes (MAN). The 6BR is responsible for setting the maximum delay response field in the Destination-Oriented Directed Acyclic Graph (DODAG) Information Object (DIO) message and the last actor is Monitoring Nodes (MN) which maintains a suspicious node list. T-IDS consists of three modules: IdentityMod, MobilityMod and IDSMod. For the IdentityMod module, after deploying the nodes, and before starting the construction of the Routing Protocol for Low-Power (RPL) topology, the BR uses IdentityMod to configure the 6BRAN list of each 6BR in the network. This list will be used to control access and authenticate nodes. The MobilityMod module manages hierarchically the mobility with the collaboration of BR, 6BR and nodes in the network. MobilityMod is used by different actors to maintain the network state regarding mobile nodes. In fact, 6BRAN contains the mobility status of each node. Intrusion Detection Module (IDSMod) detects attacks whenever the IDSMod asks the IdentityMod and the MobilityMod to check if the node belongs to the network and if it is a mobile node. Therefore, with minimal knowledge and collaborative observation, the nodes can detect the nodes behaving in a suspicious way based on certain RPL rules. Even if the authors show how T-IDS can deal with a Sybil attack, the resources appear to be costly.

In [33], the authors have chosen the anomaly-based detection as a method of intrusion detection in wireless sensor networks. To test and compare detection rates, a Multi-Layer Perceptron Back propagation Neural Network (BPN) was chosen from the architectures and was compared to a Support Vector Machine (SVM) classifier. In their approach, the authors use NSL- KDDTrain+_20Percent as a dataset. Before starting the training step, the authors have processed and normalized the data in order to convert the raw input data into an appropriate format. The machine learning the algorithms can then use this information for subsequent analysis. The authors then trained the model with the chosen dataset. Finally, they employed the SVM for the classification and the detection phase. In this work, the authors have focused on DOS (Smurf, Neptune, Back, Teardrop, Pod, Land) attacks. Their results show a very good level of performance using the neural network method for the majority of attack types, but the results are not significant with attacks with few samples such as the "Pod" and "Land" attacks. The SVM results are better than the BPN which was also able to detect attacks for very low samples as was the case with "Pod" and "land".

In [34], the author proposes an approach where one of its components can be described as an outer shell that acts as a wrapper for the entire application. Another component is a program simulating the normal activities of a sensor node in terms of collecting sensor values and sending them to the network. This method reads the energy consumption of the node and stores the values of the variation of energy consumption over time in an appropriate data structure. An algorithm analyzes the collected values to detect attacks. If the energy consumption of the node is uniform and somewhat linear, the detection method can take advantage of the linear regression and the history of energy consumption to predict the value of the upcoming energy consumption value. Due to the uncertainty regarding the power consumption profile of a sensor node, if the power consumption is not linear, another algorithm is used to collect and compare the variation of the average energy consumption during a period of energy consumption (node history). This method uses a past history to calculate an average value. If the computed average values increase for a consecutive period for a given node, the method concludes that the node is under attack. A second verification is proposed using RPL routing protocol involving three roles: the child, the parent and the grandparent.

2.4. Non-industrial IDS

Other research works have been published recently on intrusion detection fields, which are not targeting industrial systems. Among them, [35] where the authors propose a feature optimization and selection method to improve intrusion detection

for Fog Computing (FC) and Edge Computing (EC) called Machine Learning-Enabled Intrusion Detection System (ESOML-IDS). For their experimentation, they use the UNSW-NB15 dataset which contains only 42 features (including 3 categorical and 39 numeric features). The authors compare the results of the ESOML-IDS method with other machine learning algorithms. This comparison shows that the method is much better. However, by analyzing the performance metrics (Accuracy, Precision, Recall, F1-Score), these results do not exceed 84%. In order for this method to be more efficient, it would be better to have a dataset with more features (more than 42 features), so that the selection is efficient and makes sense.

In [36], authors present a study to train the neural network using meta-heuristic approaches and to enhance the perceptron neural network precision. They have used Invasive Weed Optimization (IWO) and Neural Network (NN). Their method aims to reduce the error and improve the prediction. For their experimentation, they have used three datasets: UCI Heart disease data (with only 13 features), Iris dataset (with only four features) and the Wisconsin Diagnostic Breast Cancer (WDBC with 31 features). Their results show a reduction of error while increasing the NN layers number which is a good improvement for neural network models. However, they do not show any results regarding the prediction. Furthermore, it would have been wiser, if they had calculated precision, recall, F1-score and the confusion matrix to see the rate of false positives and false negatives.

In [37], the authors propose an approach to detect Android malware using Support Vector Machine (SVM) classifier and Harris Hawks Optimization (HHO) algorithm. This latter is responsible for the optimization of SVM hyperparameters and features weighting while SVM performs and evaluates the classification. The authors have also proposed an approach to recognize the most relevant feature for the best malware detection. Their approach was tested and evaluated on five datasets from CIMalAnal2017 sampled datasets. The results are promising regarding the precision and accuracy rates. However, the approach suffers from time consumption and computational complexity. Furthermore, this work only focuses on one kind of attack–malware–and gives no indication of distinguishes a real malware attack from a simple bug in Android.

Another work [38], where the authors propose a new method of detection for intelligent systems called Fusion of Deep Learning based CyberAttack Detection and classification model for Intelligent Systems (FDL-CADIS) combines deep learning techniques and classification models. The proposed FDL-CADIS technique transforms the Malware binary files into two-dimensional images and employs them into a new model called MobileNetv2. The authors have used for their hyper parameter tuning process, the Black Widow Optimization (BWO) technique. The proposed method has been tested on 3 datasets and their results seem promising. Nevertheless, criticisms of this method include the fact that it only detects malware and that there is no information regarding the used features to check their relevance.

3. Behavioral-based IDS based on neural networks

3.1. Main principle

An Artificial Neural Network (ANN) is inspired by biological neuron behavior and the structure of the human brain [39,40]. The biological neuron is a cell composed of a cellular body and a kernel. The cellular body branches out to form what are called dendrites. Due to this latter, information is routed from outside to the neuron. Then it is processed by the neuron and it goes through the axon to be transmitted to the other neurons [41]. ANN imitates the biological neuron functioning with a better accomplished performance.

An artificial neural network is a set of elementary objects that we call formal neurons. ANNs are distinguished from each other by their complexity levels, their neuron types and their objectives.

In the literature, several kinds of ANNs can be found. However, in this research work, MultiLayer Perceptron (MLP) is used. ANN is organized into three kinds of layers:

- The input layer, which is a layer from which data is transmitted to the next layers.
- The hidden layers, which interface with the input layer and output layer. The whole computation is carried out inside them.
- The output layer, which is in charge of producing the result for given inputs.

In ANN, two learning procedures exist: Supervised Learning and Unsupervised Neural Network Learning [42]. The first one is used when input data are already labeled and categorized. This learning method is often used in feedforward or MultiLayer Perceptron (MLP) models [43]. The second one is much more complex since the system has to receive unlabeled data, detect similarities in the received data during the training process and organize them into categories. Self-Organizing Maps (SOM) are another type of ANN which uses the unsupervised learning technique for its training process [42].

3.2. Neural network: Motivations

In this paper, a neural network was chosen among other algorithms. This choice was made for its simple architecture and its easy implementation period here. Furthermore, this algorithm also complies with the context of our study, which consists of dealing with huge amounts of industrial data. Finally, a neural network was chosen because of its parallelism, which allows for data processing in real time. Therefore, the real-time constraint which is required for the industrial applications has been respected.

3.3. Intelligent behavioral based IDS: Assumptions

The approach of this IDS is valid whilst respecting several assumptions presented below:

- This IDS uses network traffic. Therefore, the industrial platform has to be connected to the network.
- This IDS uses the neural network algorithm which requires a huge amount of data for its training step to obtain good classification results. Therefore a large dataset is used in this work.
- Most attacks go through the network layer of the Open Systems Interconnection (OSI) model.

3.4. Industrial dataset

(1) Generation process:

To measure the IDS efficiency of the industry 4.0, a real industrial dataset is required. This 4th generation is characterized by the diversity of attacks: Information Technology (IT) attacks, Operation Technology (OT) attacks, and the side attacks that range from IT to OT equipment. To perform this study, a real industrial dataset was constructed and is available with this link: Industrial dataset

To construct the proposed dataset, several steps have been performed and detailed in Fig. 1.

To construct the dataset, seven attacks are simulated to obtain the malicious traffic:

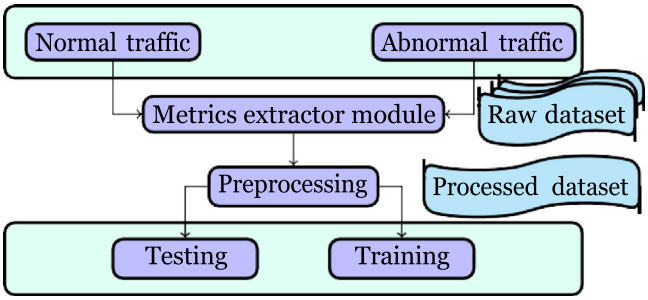

Fig. 1. Dataset extraction process.

(1) Implemented IT attacks:

- DoS/DDoS: A DDoS attack aims to make a server, service or infrastructure unavailable. The attack can take different forms: saturation of the server's bandwidth to make it unreachable or exhaustion of the machine's system resources, thus preventing it from responding to legitimate traffic.
  To perform these attacks, the following were sent; a TCP SYN flood, TCP ACK (Acknowledgment) flood, TCP RST (Reset) flood or Xmax flood by setting all TCP flags (CWR, ECN, URG, ACK, PSH, RST, SYN, FIN). UDP flood packets were also simulated. The script implementing these attacks uses hping3. If hping3 is not found, it attempts to use the nmap-nping tool instead.
- File Transfer Protocol (FTP) brute force: Brute force attack is a method used in crypt-analysis to find a password or key. As soon as the attacker obtains the FTP server login and the password, he can retrieve MES database information (configuration file, users, password...) and use this information to alter the database containing all the information regarding the production. This attack consists of testing, one by one, all the possible combinations. For it to be performed, the dictionary method is used to crack the FTP server connection.
  First step is to obtain the login and the password of the FTP server by using either a hydra or Metasploit tool then a connection to the server to retrieve some files and delete others related to the MES database.
- Web Hypertext Transfer Protocol (HTTP) DoS: The principle of this attack is to paralyze the Apache server which uses HTTP protocol to make it unavailable by sending a large number of requests.
  In this attack, a script written in Python called SLOWHTTPtest is used to perform an application layer Denial of Service attacks (DoS). Its principle relies on the fact that requests are not processed before being completely received by the server. If the data is not complete or the speed of the sending packets is too low, the resource is kept busy until it receives all of the data. In this attack, Apache is targeted by causing very significant memory and CPU usage on it.
- Botnet: A botnet is a network of computers infected with malware so that they can be controlled remotely, forcing them to send spam, spread viruses or launch DDoS attacks without the knowledge or approval of the real owner of the computer. In addition, an attacker can paralyze the entire production system by launching a DDoS with a botnet which can be financially very consequential for a factory.
  The Ares tool, which is a written remote administration tool RAT in Python, is used to perform this attack.

(2) Implemented OT attacks:

- SQL attack: This attack is a lateral attack which goes from IT to OT equipment. It is composed of two steps. The first one is a SQL brute force to retrieve the MES database's login and password. The principle is the same as for the FTP brute force but uses the mssql_login module of Metasploit. Then, the second step consists of dropping or altering data by taking control of the machine hosting the MES database.
- PLC disturbing process: Two attacks are launched to disturb the PLC process. Their principle consists of disrupting the production line functioning by modifying either the recipe scheduled (production order) by the operator, or by provoking a remote emergency stop. After studying and analyzing the registers' configuration of the platform PLC, we decide to modify the registers responsible for the worst and the most widespread scenarios in industrial attacks, namely the complete shutdown of the production line and the modification of the production line recipe. In our case, register 2 of M580 PLC is responsible for remotely stopping the conveyor, and register 120 of M340 PLC is in charge of seamlessly modifying the number of balls. Therefore, the vials are filled, capped and removed to destocking without the operator noticing the modification of his planned PO. To perform this, we correctly connected to the PLC line and modified the Modbus registers. For this attack, the Metasploit attack tool and its Modbus client module was used.
- MITM: This is an attack whose principle consists of intercepting communications between two machines, hosts, servers..., without either of them suspecting that the communication channel between them has been compromised. For this attack, we entered into the middle, between the PLC and the server using the Ettercap tool, then the content of the registers was modified to disturb the production line's nominal functioning. This attack triggers either a remote emergency stop or a modification in the operation recipe.

After simulating these attacks, both normal and malicious traffic are captured to get network files (PCAP files). From these files, raw data is extracted and pre-processed to obtain the final dataset used in the proposed behavioral-based IDS.

This dataset is divided into training and testing datasets. It is a labeled dataset with nine labels built in a real industrial platform at Brest university and available for the researchers community. More details regarding this dataset are given in Table 1.

(2) Extraction process:

To build the dataset, an extractor is developed in Python to extract features from the pcap files. The extractor gives a csv file as the output containing 133 transport and application layer features. The application layer features are related to the Modbus/TCP protocol. Some of these features are directly extracted from the TCP or Modbus headers such as IP addresses, ports numbers, protocols..., others are statistically computed such as average, minimum, maximum...in this raw data, digital and non-digital features are included requiring a pre-processing step to be used by the behavioral-based IDS algorithm. More details are given in [45].

(3) Pre-processing process:

After extracting normal and malicious traffic from the network, the obtained raw data was pre-processed to be used. This processing is semi-manually made because raw data contains both digital and non-digital features. The non-numerical features are converted according to several rules. For IP addresses, only the last number of this latter is kept since the three others are the same for all the architectural elements. For each protocol, a number is assigned and a features list is split into a single value by row.

Table 1

Year	Approach type	IDS type	Attacks type	IDS placement	Protocols	Techniques	Response type	Metrics of evaluation	Used tools	Ref.
2007	Behavioral	NIDS	Zero day and DOS	Centralized (SCADA)	Modbus/ TCP	Matching patterns	Passive	Not specified	EMERALD Bayes sensor, eXpert-Net SNL and Snort	[23]
2013	Specification-based	NIDS	DNP3 malformed packets	Centralized (SCADA)	DNP3	deterministic	Passive	Detection accuracy, latency, and analysis of the flow	VMware virtual machine with a single logical p...	[22]
2014	Specification-based	NIDS	Injection of packets, controller behavior imitation	SCADA and automates	Modbus/ TCP	deterministic	Passive	Attack detection capability	Bro	[17]
2014	Behavioral	NIDS	MITM, SYN Flooding and honeypot	Centralized (SCADA)	DNS, FTP, Modbus, TCP, UDP	Probabilistic (one-class SVM)	Passive	Accuracy of the classification	Not specified	[25]
2014	Behavioral	NIDS	MITM	Centralized (SCADA)	Modbus/ TCP	Probabilistic (un-supervised machine learning)	Passive	FPR (False Positive Rate), F-score, detection accuracy and classification time	Matlab	[44]
2015	Behavioral	NIDS	suspicious code injection	Centralized (PLC)	Modbus/ TCP	Probability (one-class SVM)	Passive	Detection Time	Wireshark, Unity Pro, M340 PLC of Schneider	[26]
2016	Behavioral	NIDS	Injection data, DOS, network attacks	Centralized (SCADA)	Modbus/ TCP and MMS	Probabilistic (machine learning, patterns)	passive	Traffic periodicity	Not specified	[24]
2016	Specification-based	HIDS	Injection suspicious execution traces	Decentralized (SCADA, PLC)	Modbus	deterministic (Patterns)	Passive	Detection accuracy, TPR, FPR	OpenModelica2, Schneider PLC M580, Spot library	[19]
2017	Misuse	NIDS	Packets injection	Decentralized (sensors, actuators, controllers)	Not specified	Statistics (rules measurements)	Passive	Detection accuracy	Petri Net (SED Modeling)	[15]
2017	Misuse	NIDS	malformed packet , DoS , ARP spoofing (Usurpation) and MITM	Centralized	IEC 61850 GOOSE	deterministic	Passive	Detection accuracy and process time	Wireshark, ITACA(C/C++)	[29]
2017	Specification-based	NIDS	GOOSE usurped and Ethernet storm	Centralized (Land)	GOOSE, Modbus/ TCP	deterministic	Passive	TPR, the number of destroyed packet by Bro, analysis time GOOSE77	Bro, Suricata, Snort, Metasploit, Scapy, Simulink, Matlab	[18]
2019	Specification-based	NIDS	Disturbing industrial process and MITM	Decentralized (sensors, actuators, controllers)	Modbus/TCP	deterministic	Passive	Detection accuracy	Bro	[20]
2020	Hybrid	NIDS	DoS and Replay attack	Sensors	Not mentioned	Probabilistic	Passive	Detection rate, accuracy rate, FP rate	Opnet, Simulink, Matlab	[28]

(*continued on next page*)

Table 1 (*continued*).

2020	Behavioral	NIDS	DoS (SYN Flood), MITM	SCADA	Modbus/TCP	Probabilistic (SVM, nonparallel hyperplane)	Passive	Accuracy rate	Matlab	[27]
2021	Behavioral	NIDS	Not specified	Not specified	Not specified	Probabilistic (ANN)	Passive	Error rate	Not specified	[36]
2022	Behavioral	NIDS	DoS, Backdoor, Exploits	FC and EC	Not specified	Probabilistic	Passive	Accuracy, Precision, Recall, F1-Score	Not specified	[35]
2022	Behavioral	NIDS	Malware	Not specified	Not specified	Probabilistic (SVM,HHO, BWO)	Passive	Accuracy, precision, recall	Matlab and Weka	[37],[38]

Table 2
Dataset digital description.

Lines number	Pcap files size	Capture duration	Attacks	Labels	Features
8735711	16 GB	5 days	7 attacks	9 labels	133

(4) Labeling process:

For classification purposes, the dataset is labeled with nine labels, one label for the normal traffic, seven labels for malicious traffic (one label for each attack) and one label for the equipment reaction against an attack. To understand the reaction traffic, take the example of a SYN flood attack. After several attempts and without receiving any ACK flag, communication is cut, the RST flag is set at 1 and State_cnx_S0 and State_cnx_S1 are set at 1. Finally, all packets meeting these conditions are labeled as a reaction to the SYN flood attack and those containing a huge amount of SYN flags are labeled as an attack. The purpose of reaction traffic is to test if the proposed approach is able to detect both the payload related to the attack itself and the behavioral change of the equipment.

This labeling operation is made on the different extracted features to distinguish one attack from another. The labeling process is manually made. This process is made easier by the extracted features and the configuration setting which separates the normal traffic from the malicious one. For instance, among the extracted features, SYN flag count and MAC address features help us to label the SYN flooding and the MITM traffic respectively.

(5) Extracted dataset: The used dataset is large which gives a good training of the neural network. It contains around nine millions lines in its csv file and has a size of 16 GB for its network traffic files. It was captured over five days, during which seven attacks were simulated. It contains 133 features.

To train and test the neural network algorithm used in the behavioral-based IDS, the dataset is divided into training and testing datasets (70%/30%). More details are given in Table 1:

3.5. Neural network: Architecture and experimentation parameters

To obtain the optimal neural network architecture, several tests were performed by changing either the layer's number, the layer's neuron number, the activation function, the epoch's number or the batch size. In the beginning of neural network development, either over-fitting or under-fitting phenomena were encountered. The first one occurs when the neural network is very good at learning its training set, but cannot generalize beyond the training set. The second one occurs when the network is not able to generate accurate predictions by using the training set.

To overcome both of these problems, firstly the correct splitting percentage between training and test datasets had to be found. After using several splitting configurations, 70% of the whole data was kept for the training dataset and 30% for the testing one. More than 70% led to the over-fitting phenomenon. Secondly, dropout function has been used but the neural network became too complicated and the results was added were not conclusive. Thirdly, we manipulated the layers and the layer's neuron numbers by increasing them. However, the results were noisy and some class predictions were lost. Finally, a model was kept with 102 neurons in the input layer corresponding to the retained features and 9 neurons in the output layers corresponding to the labels (seven labels for the simulated attacks in addition to one for normal traffic and another for the traffic related to the reaction of the equipment against an attack).

For the training stage, Keras was used with the TensorFlow backend. The Adam optimizer was used because it is the most frequent and it gives better results than the Batch Gradient Descent (BGD). All of the models were trained for 50 epochs. Batches of size 1024 were used because the dataset is large and smaller batches led to slower training with worse results. Table 2 gives the chosen model:

3.6. Performance results

In the evaluation phase, one of the main concepts of the machine learning field which is a confusion matrix (Fig. 2), has been used. A confusion matrix is a tool for measuring the performance of a machine learning model. It checks in particular how often its predictions are accurate, compared to reality. On the *x*-axis, true labels are presented and the coordinate axis shows the predicted labels. This confusion matrix is computed after classification by using the testing dataset. According to these results, all traffic types are correctly predicted as shown in the diagonal line of this confusion matrix. All attacks have been classified with high detection accuracy.

To evaluate the IDS performance in more depth, three metrics have been used: precision, recall and F1-score whose formulas are given below:

(1) Precision: a metric that gives the number of the correct computed positive predictions. It is calculated as the ratio of correctly predicted positive examples, divided by the total number of positive examples that have been predicted.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

- True Positive: the truth is positive, and the test prediction predicts a positive. There is an intrusion, and the test accurately reports this.
- False Positive: the truth is negative, but the test prediction predicts a positive. There is no intrusion, but the test inaccurately reports that there has been one.

(2) Recall: a metric that calculates the number of the correctly computed positive predictions from all positive predictions

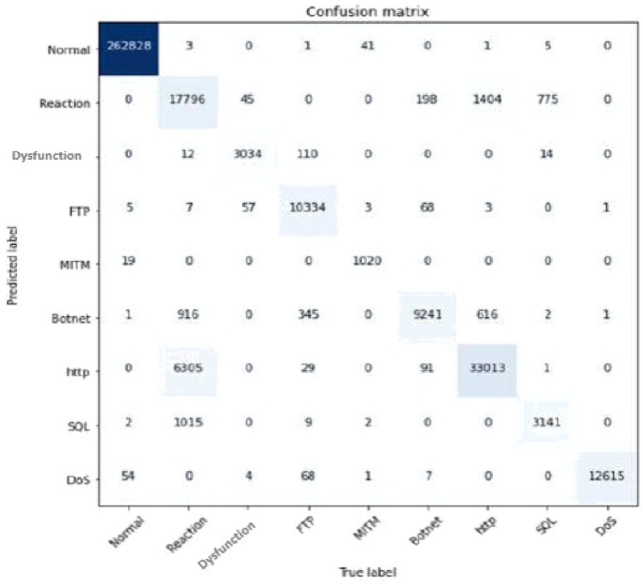

Fig. 2. Confusion matrix.

Table 3
Neural network parameters.

Parameters	Details
Learning	Supervised
Input layer	One input layer with 102 neurons (Used features)
Hidden layer	One hidden layer with 12 neurons
Output layer	One output layer with 9 neurons (9 class)
Number of epochs	50
Activation function	Sigmoid

that could have been made. It gives an indication of missed positive predictions.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

- False Negative: the truth is positive, but the test prediction are negative. There is an intrusion, but the test inaccurately reports that there has not been one.

(3) F1-score: a metric that combines the previous two scores. It is primarily used on imbalanced data, such as precision and recall metrics. The score must be near to 1.0 for the best classification.

$$F1\text{ - }score = 2_{*}\ \frac{Precision * Recall}{Precision + Recall}$$

The neural network model has first been trained using the training dataset and tested later with the testing dataset. The results show a good rate of precision, recall and F1-score as shown in Table 3:

According to the results shown in the Table 3, we notice that the normal traffic, malfunction, FTP, MITM and DoS are classified with high precision, recall and F1-Score values (more than 94%). However, the "Reaction" traffic is classified with only 88% precision. This value is due to the nature of this traffic. The "reaction" is not an attack but a reaction of the equipment against an attack. It is a specific traffic. SQL, Botnet and HTTP traffic also have lower precision values because a part of their traffic is falsely classified as "Reaction" as shown in the confusion matrix (see Fig. 2). This wrongly classified traffic is called "false negatives".

Table 4
Neural network performance before hybridization.

Traffic type	FPR	FNR	Precision	Recall	F1-Score
Normal	0.05%	0.04%	99.9%	99.9%	99.9%
Reaction	0.7%	31%	88.02%	68.3%	76.9%
Malfunction	0.03%	3.3%	95.7%	96.6%	96.1%
FTP	0.04%	5.1%	98.6%	94.8%	96.6%
MITM	0.005%	4.4%	98.1%	95.5%	96.7%
Botnet	0.5%	3.8%	83.08%	96.2%	89.15%
HTTP	1.9%	5.7%	83.7%	94.2%	88.6%
SQL	0.2%	19%	76%	79.7%	77.45%
DoS	0.03%	0.15%	98.94%	99.9%	99.4%

Furthermore, False Positive Rate (FPR), and False Negative Rate (FNR) for "reaction", "SQL", "Botnet", and "HTTP" traffic are high.

By having knowledge of the production environment, it is possible to determine the impact of an attack and therefore specify its type. Certain traffic can be reclassified and their FPR and FNR can be accordingly reduced. In order to reach this goal, another kind of anomaly-based IDS is added to our study: specification-based IDS, which is detailed in the next section. This quantification classifies the detected anomaly into three categories: temporal, sequential or content.

To test and compare the neural network model to other machine learning algorithms, a quick comparison was performed regarding all the performance metrics for the following models: K-Nearest Neighbors (KNN), Linear Regression (LR), Naive Bayes (NB), Random Forest (RF), Support Vector Machine (SVM), and Decision Tree (DT). Table 4 shows the results of this comparison.

According to this comparison, some models are good in terms of accuracy but poor in terms of the other metrics like DT, LR, NB except KNN, which is good in terms of accuracy, Recall and F1-score. ANN is the best one in terms of all performance metrics.

4. Specification-based IDS: The industrial ISA-95 standard

In this section, a second sub-category of anomaly-based IDS is proposed: the specification-based IDS. The main purpose of this one is to measure and quantify the impact of an intrusion according to three categories: temporal, sequential and content. This IDS is based on two bases: the MESA model from the ISA-95 standard (an industrial standard), which allows the checking of the sequential aspects of the production order (PO) execution and the Key Performance Indicators (KPI), which verify the temporal aspects. More details are given in the following subsections.

4.1. The ISA-95 standard: The MESA model

The ISA-95 is an industrial standard based upon the hierarchical structure. The development of this standard started in 1995 when the OPC Foundation, the ISA95 committee, and MESA merged their efforts to make a new model which adds ISA-95 object model representations of equipment, personnel, material, and physical assets to an OPC UA 95 specification. The aim of this model is to show how the OPC Foundation, the ISA95 committee, and MESA standards can be used together in a federated system architecture [46].

Its hierarchy is composed of five different levels: level four for business planning & logistics, level three for Manufacturing Operations & Control, level two responsible for the responsibility of the PLC, level one for the sensors and actuators, and finally level zero for the production process [47].

This research work focuses on the third level of this architecture, which is the Manufacturing Executive System (MES). The main basis of this work is driven by the generic activity model from the third part of the ISA-95 standard, also called

Table 5
Comparative of machine learning algorithms performance.

ML model	Accuracy	Precision	Recall	F1-score
KNN	0.93	0.80	0.82	0.93
LR	0.93	0.73	0.70	0.67
NB	0.90	0.68	0.64	0.62
RF	0.88	0.70	0.80	0.71
SVM	0.88	0.71	0.65	0.61
DT	0.87	0.70	0.67	0.67
ANN	0.91	0.92	0.92	0.92

the MESA model. This model is shown in Fig. 3. It is a finite state machine, which explains how a succession of transactions through the various entities of the MES occurs and orchestrates operation planning, execution and management activities. This model contains three exchanged data kinds: abstracted data of the different information systems, data exchanged within the MES and data exchanged with the automation level to execute an order [48].

The MESA model allows us to have a deterministic approach to the transactions that occur in a factory 4.0. Therefore, it generates a succession of events which feed the Specification-based IDS as inputs. If the succession of these transactions is not respected, IDS raises an alert. Hence, this model allows us to eliminate many false positives due to the prior knowledge of the succession of events.

4.2. Specification-based IDS: Assumptions

The approach of the IDS is valid while respecting several assumptions presented below:

- Industrial platform behavior is cyclic and deterministic.
- The proposed specification IDS targets the ISA95 compliant MES.
- The MES database is supposed to be accessible.
- Communications from the industrial control system are generally regular over time.
- PLCs are connected to the network.

4.3. Specification-based IDS: Principle

The main purpose of this IDS is to detect malfunction in the process. Based on the previously presented model, 13 common anomalies are identified in the next subsection. To detect these anomalies, a set of rules is established allowing their detection. In addition to this malfunction detection role, Specification-based IDS has another role which consists of classifying the intrusions detected by the behavioral-based IDS into temporal and/or sequential categories(see Tables 6 and 9).

These anomalies are divided into temporal, sequential and content anomalies. As previously mentioned, the MESA model allows the planning and the execution of task sequence. Therefore, thanks to this model property, the sequential anomalies are identified. To make this model more complete, we added some KPI (Key Performance Indicators) to monitor the temporal aspect. These KPI are provided by ISO 22400 standard and they were chosen according to these research requirements. The used KPI are: Actual Production Time (APT), Actual Transportation Time (ATT) and Actual Order Execution Time (AOET) [47].

- Actual Production Time (APT): The actual time spent on a machine. For instance, if the production line is composed of four machines, four APT are computed.
- Actual Transportation Time (ATT): The actual time spent between machines, such as loading and unloading time.
- Actual Order Execution Time (AOET): The total time spent to execute an order, from the start to its

All identified anomalies are detailed below in Table 5:

4.4. Specification-based IDS: Results

Specification-based IDS is developed to check the 13 identified anomalies. This checking is performed on the MES database which is compliant with the ISA-95 standard according to the set of rules detailed in Table 5. Depending on whether the rules are followed or not, the IDS detects if the system is working normally or if it is faulty.

Specification-based IDS gives the possibility to check some production orders for a quick checking or an indepth analysis for the entire database. The anomalies are recorded in a logs report to be used by Decision Making System (DMS) for a final decision making.

After testing all the anomalies and checking all the controls, specification-based IDS has detected all the played scenarios with a high accuracy in a short amount of time.

To test this IDS, 32 attack scenarios (Deny-Of-Service, Man-In-The-Middle, Brute force...) have been launched and have been all detected as faults. However, no discrimination between faults and real attacks has been made. Thanks to the behavioral-based IDS, industrial faults can be distinguished from a real cyberattack. However, this distinction is made with a high False Positive Rate (FPR).

On the other hand, the specification-based IDS allows the quantification of the detected anomaly and its classification into three categories previously detailed but without discrimination between faults and attacks. Therefore, to take advantage of both IDS and overcome their limitations mutually, the idea of hybridization of the two previous IDS has arisen. This hybridization is made through the last component: a Decision Support System (DMS) as explained in Section 5.

5. Proposed BI-ANOmaly-based IDS: BIANO-IDS

5.1. BIANO-IDS principle and components

The main purpose of the BI-ANOmaly-based IDS (BIANO-IDS) is to distinguish a real intrusion from a production line malfunction with a low FPR and measure the impact of the detected anomaly on the industrial platform.

DMS analyzes the detection results of the two previous IDS to determine the nature of the detected anomaly and measure its impact by quantifying and classifying it into the three previous anomaly categories (See Table 5). DMS works according to a set of rules as explained in Fig. 5.

Fig. 4 gives an overview of the proposed bi-anomaly-based IDS named BIANO-IDS and its components: Specification-based IDS, Behavioral-based IDS and DMS.

- Specification-based IDS : Assuming that the behavior of an industrial line is deterministic and cyclical, the specification-based IDS analyzes the production line operation with a focus on the planned and actual executed Production Order (PO). Its role is to check
  that there is no difference between the planned and executed PO according to the 13 identified rules from the MESA model.
- Behavioral-based IDS: Assuming that most attacks pass through the network layer of the OSI model, this IDS analyzes network traffic captured during the operation of the production line. It is based on the neural network which learns the nominal functioning of the industrial line, detects any suspicious activity and allocates it into an attack classification that it learned during its training phase.

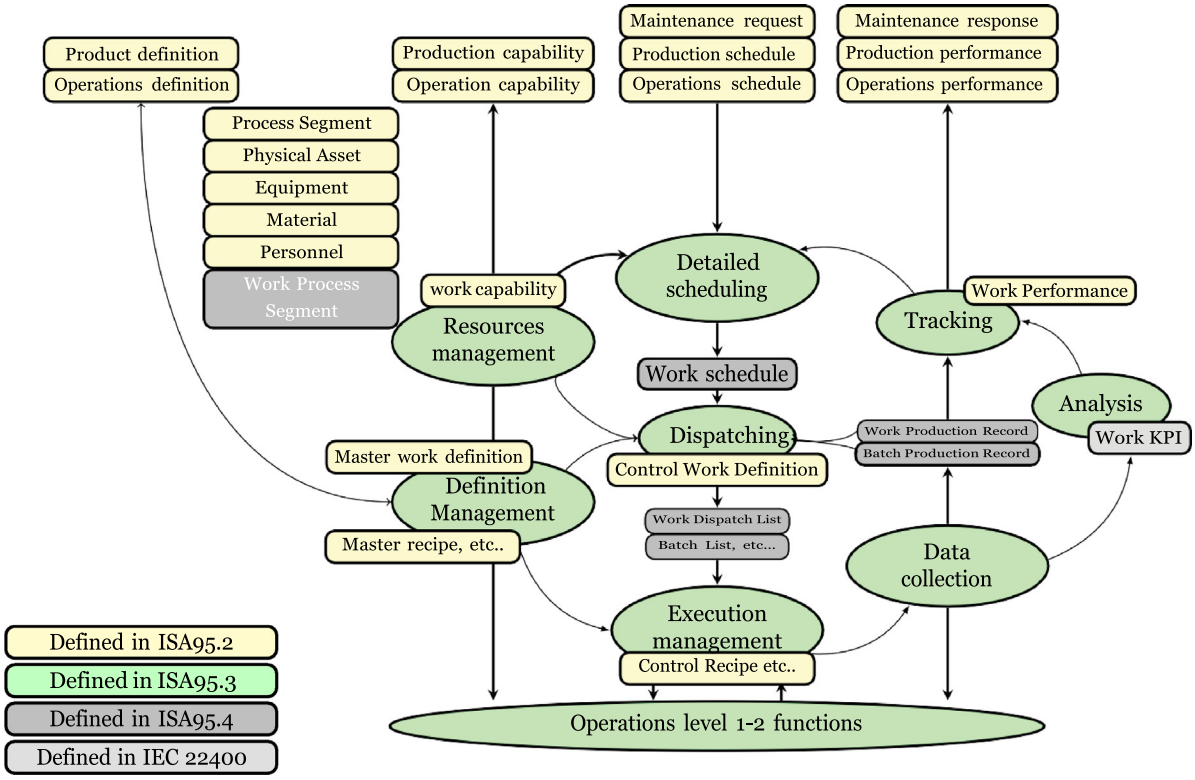

Fig. 3. The MESA model.

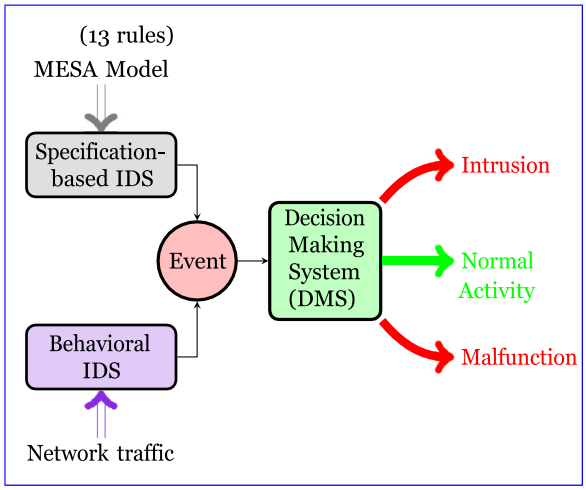

Fig. 4. BIANO-IDS components.

- Decision Making System (DMS): This is a decision-making system that aims to take advantage of the previous two components and overcome their limits. It analyzes the resulting logs of the two previous components to determine the nature of the anomaly. This module follows the logic explained in Section 5.2 with more details regarding this third component.

5.2. DMS principle

The third module of the proposed approach is the DMS. Two types of DMS in the DMS field exist: Decisions are either made collaboratively or independently. Since BIANO-IDS is composed of two kinds of IDS which work in parallel, it is logical that the DMS should work in a cooperative manner.

This module works according to many rules (explained in Fig. 5), which allow us to determine the nature of the detected anomaly (fault or real intrusion).

As mentioned in the behavioral-based IDS, we made the assumption that the majority of the attacks are launched from the network. Therefore, according to the behavioral-based IDS (which analyzes network traces) detection results, DMS checks the resulting logs, analyzes them and determines if the system is facing a simple fault or a real intrusion.

According to Fig. 5, several decisions could be made. Before detailing them, it is important to understand the notations used in this diagram. Letter "B" and "S" correspond respectively to the behavioral-based and the specification-based IDS log files. In addition, numbers "0" and "1" show respectively the absence or the presence of the alerts in the log files. For instance, S1 means that the behavioral-based IDS log file contains alerts and B0 shows that the specification-based IDS log file contains no alerts. The potential decisions that a DMS could make are detailed below:

- If there are no alerts in the behavioral-based IDS (B0) or the specification-based IDS (S0) log files ==> we are facing a normal activity.
- If there is an alert in behavioral-based IDS log file (B1) despite the fact that we have no alert in the specification-based IDS (S0) log file ==> we are facing a real intrusion since most played attacks in the industry have inevitably gone through a network.
- If there is no alert in the behavioral-based IDS log file (B0) but an alert exists in the specification-based IDS (S1) ==> we are facing industrial faults for the same reason as before.
- If there are alerts in both the behavioral-based IDS log file (B1) and in the specification-based IDS (S1) log files ==> we are facing a real intrusion.

Table 6
Anomalies identified from the MESA model and KPIs.

Anomalies	Category	Rules
1: Check that the response matches with the request segment	Content	OP_ID (Planned) = OP_ID (Executed)
2: Check that the order of the segments is respected	Sequential	StartTime (WO1) StartTime (WO2) StartTime (WO3) (planned) and StartTime (WO1) StartTime (WO2) StartTime (WO3) (executed)
3: Check the personnel skills planned/used	Content	ID_PersonnelClass(planned)= ID_PersonnelClass(executed)
4: Check that the PO is executed with the right equipment	Content	ID_EquipmentClass(planned) = ID_EquipmentClass(executed)
5: Check that the PO is executed with the correct material	Content	ID_MaterialClass(planned) = ID_MaterialClass (executed)
6: Check if the total expected time (AOET) is correct	Temporal	AOET(planned) = AOET(executed) +/-90S
7: Check that the duration per segment (APT) is correct	Temporal	APT1(planned) = APT1(executed) +/- 10S, APT2(planned) = APT2(executed) +/-10S, APT3 (planned) = APT3 (executed) +/-10S, APT4(planned) = APT4(executed) +/-10S
8: Check if times between segments (ATT) are correct	Temporal	ATT1(planned) = ATT1 (executed) +/-10S, ATT2 (planned) = ATT2 (executed)+/- 10S, ATT3(planned) = ATT3 (executed) +/-10S, ATT4 (planned) = ATT4 (executed) +/-10S
9: Check for overlapping between PO	Sequential	PODateEnd (executed) (Previous PO)< PODatStart (executed) (next PO)
10: Check that the quantity requested is the one manufactured	Content	PQ(planned) = PQ(Executed)
11: Check if resources are available before launching the PO	Content	QuantitePersonnelReady > QuantitePersonnel (planned) and QuantiteEquipmentReady > QuantiteEquipment (planned) and QuantiteMaterialReady > QuantiteMaterial (planned)
12: Check the launching order of the PO	Sequential	EndTime(Planned)(PO1) StartTime(Planned) (PO2),EndTime(Executed) (PO1) StartTime(Executed) (PO2)
13: Check if the equipment is down while continuing to send data	Content	PO_state (planned) = release and PO_state (executed)release, PODateStart (executed) = 0 PODateEnd (executed) = 0

5.3. DMS programming

The following code details the rules that the DMS must adhere in order to make a decision regarding the nature of the detected anomaly. It firstly defines two files: log1 and log2, which represent the behavioral-based IDS and the specification-based IDS log files respectively. Then, it reads their contents, which the code relies on in order to check the previously outlined rules. The term of ''controls'' corresponds to the anomalies that the specification-based IDS checks (the 13 anomalies presented in Section 4). Consequently, controls two and 12 are sequential. Controls six, seven and eight are temporal and the others controls denote the PO content.

5.4. DMS: Alerts classification

According to the logs content, thirteen categories of results are possible as mentioned in Fig. 5.

The previously presented decisions could be classified furthermore into either temporal, sequential or content alerts. By adopting the assumption that most attacks go through the network, the DMS could make one decision among 13 possibilities, as summarized below:

- Normal activity: if both IDS logs are empty and contain no alert.
- Industrial malfunction: if the behavioral-based IDS did not detect any intrusion (log file is empty) and the specification-based IDS has detected some alerts, the DMS decides that the detected anomaly is a malfunction since no anomaly is detected through the network.
  - Temporal malfunction: Faults due to a delay during the execution of the PO. For instance, the PO execution time is abnormally long
  - Sequential malfunction: Faults due to the disrespect of the PO order. For instance, detecting the risk of overlapping
  - Content malfunction: anomaly due to the content of the PO. For instance, checking the resource used to perform a PO
  - Temporal and sequential malfunctions: Faults due to both temporal and sequential reasons.
  - Temporal and content malfunctions: Faults due to both temporal and content PO reasons
  - Sequential and content malfunctions: Faults due to both content PO and sequential reasons

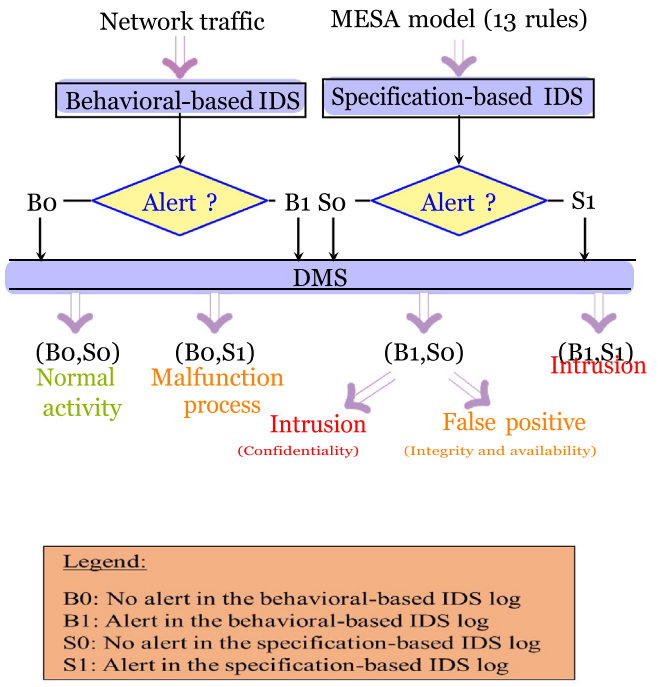

Fig. 5. DMS Rules.

- Real intrusion: if the behavioral-based IDS detects some intrusions (log file contains alerts)
  - Temporal intrusion: Attack due to a delay during the execution of the PO. For instance, checking the long time spent in a PO due to a DDoS or MITM attacks.
  - Sequential intrusion: Attack due to the disrespect of the PO order. For instance, the risk of over-lapping by checking the malfunction of a PO due to a MITM attack
  - Content intrusion: Attack due to the content of the PO. For instance, the use of the wrong resources to perform a PO due to SQL injection attack
  - Temporal and sequential intrusions: Attacks due to both temporal and sequential reasons. For instance, checking of the PO time or modified order due to MITM and DDoS attacks
  - Temporal and content intrusions: Attacks due to both temporal and content PO reasons. For instance, checking of the PO time or modified recipe due to SQL injections and DDoS attacks
  - Sequential and content intrusions: Attacks due to both content PO and sequential reasons

5.5. Proposed approach extension to other attack kinds

In this paper, the proposed approach has been tested on a set of attacks (MITM, DoS, FTP brute force ...). These attacks have varied from IT to OT (industrial) attacks. However, this approach is also valid and extensible to any other kind of attack. Taking the example of stealthy attacks such as buffer overflow, if the latter occurs in the server machine, the server will be blocked. Therefore, behavioral-based IDS will raise an alert. In addition, communication with the production line will be cut, and this impact will be detected on the industrial platform. Consequently, a second alert will be raised by the specification-based IDS. According to the DMS algorithm, BIANO-IDS will also detect this intrusion.

Algorithm 1: DMS programming code

```
Result:
log1 ← Behavioral - based IDS log;
log2 ← specification - based IDS log;
Read log1;
Read log2;
if "Normal" in log1 and log2 ←None then
  | print("Normal activity");
end
if "Normal" in log1 and controls [6, 7, 8] in log2
  then
  | print("Temporal anomalies");
end
if "Normal" in log1 and controls [2,12] in log2 then
  | print("Sequential anomalies");
end
if "Normal" in log1 and controls [1, 3, 4, 5, 9, 10,
  11, 13] in log2 then
  | print("Content anomalies");
end
if "Normal" in log1 and controls [1, 2, 3, 4, 5, 9,
  10, 11, 12, 13] in log2 then
  | print("Sequential and content anomalies");
end
if "Normal" in log1 and controls [2, 6, 7, 8, 12] in
  log2 then
  | print("Temporal and sequential anomalies");
end
if "Normal" in log1 and controls [6, 7, 8, 2, 12] in
  log2 then
  | print("Sequential and temporal anomalies");
end
if "Attack" in log1 and controls [6, 7, 8] in log2
  then
  | print("Temporal attacks");
end
if "Attack" in log1 and controls [2, 12] in log2 then
  | print("Sequential attacks");
end
if "Attack" in log1 and controls [1, 3, 4, 5, 9, 10,
  11, 13] in log2 then
  | print("Content attacks");
end
if "Attack" in log1 and controls [2, 12, 1, 3, 4, 5, 9,
  10, 11, 13] in log2 then
  | print("Sequential and content attacks");
end
if "Attack" in log1 and controls [6, 7, 8, 2, 12] in
  log2 then
  | print("Temporal and sequential attacks");
end
if "Attack" in log1 and controls [1, 3, 4, 5, 6, 7, 8,
  9, 10, 11, 13] in log2 then
  | print("Temporal and content attacks");
end
```

Table 7
DMS results.

	Anomalies	Sequential attacks	Temporal attacks	Content attacks
Without hybridization	100%	0%	0%	0%
Withhybridization	41%	6%	25%	28%

Table 8
Classification error probabilities.

Anomaly category	Attack	Classification error rate
Sequential	SQL	0.42%
	BOTNET	3.11%
Temporal	DoS	0.0085%
	Malfunction	0.90%
Content	MITM	0.35%
No category	HTTP	12.57%
	FTP	0.0085%

5.6. DMS results and discussion

Before combining both detection results of behavioral-based IDS and specification-based IDS, all the launched attacks have been detected as anomalies. 100% of anomalies have been detected without distinction whether these anomalies are industrial defects or real intrusions. After using DMS, which pairs both detection results, these anomalies have been categorized into industrial defects, temporal, sequential or content intrusions. The percentage has been split into 41% for faults, 6% for sequential attacks, 25% for temporal attacks and 28% for content attacks. Therefore, the detection has been improved with a low FPR and the discrimination has been made.

In addition, the confusion matrix is also improved due to this hybridization. This improvement is possible thanks to the specification-based IDS which categorizes the attacks into 3 categories: sequential, temporal, and content. According to these categories, an impact can be observed on the industrial platform.

To improve the results, it was necessary to compute the classification error probability for each class. This probability is calculated from the confusion matrix (obtained after the training time) and corresponds to the ratio of the misclassified traffic for a specified class in relation to all misclassified traffic. All these probabilities are given in Table 7. Thanks to this rate of error (probability), it is possible to reclassify the misclassified traffic into the right kind of attack.

According to Table 7, when a source of traffic is wrongly classified, these probabilities are used to reclassify it. For instance, the temporal category contains two attacks: DoS and malfunction. Based on these probabilities, when an attack has a temporal impact, it is more likely to be a DoS attack than a malfunction since the classification error is smaller for DoS compared to that of a malfunction attack.

The same goes for the normal traffic i.e., when no impact on the platform is detected. Therefore, all normal misclassified traffic is reclassified into the right class. Based on this principle, all traffic can be reclassified correctly except for FTP and HTTP attacks, where hybridization IDS have not contributed any added value, as these two attacks act at the IT level and not on the platform, especially in order to reclassify their traffic. However, since traffic of other attacks has been correctly reclassified (improved), FTP and HTTP have also been improved. Thanks to the hybridization, performance metrics and the confusion matrix are improved as shown in Table 8 and Fig. 6.

Table 9
Neural network performance after hybridization.

Traffic type	FPR	FNR	Precision	Recall	F1-Score
Normal	0%	0%	100%	100%	100%
Reaction	0.4%	31%	92.4%	68.3%	78.5%
Malfunction	0.003%	1.5%	95.7%	98.4%	98.9%
FTP	0.002%	0.26%	99.9%	99.7%	99.7%
MITM	0%	0%	100%	100%	100%
Botnet	0.27%	2.06%	91.3%	97.9%	94.4%
HTTP	1.9%	5.7%	84.1%	95.9%	89.6%
SQL	0.15%	5.5%	79%	99%	87.8%
DoS	0%	0%	100%	100%	100%

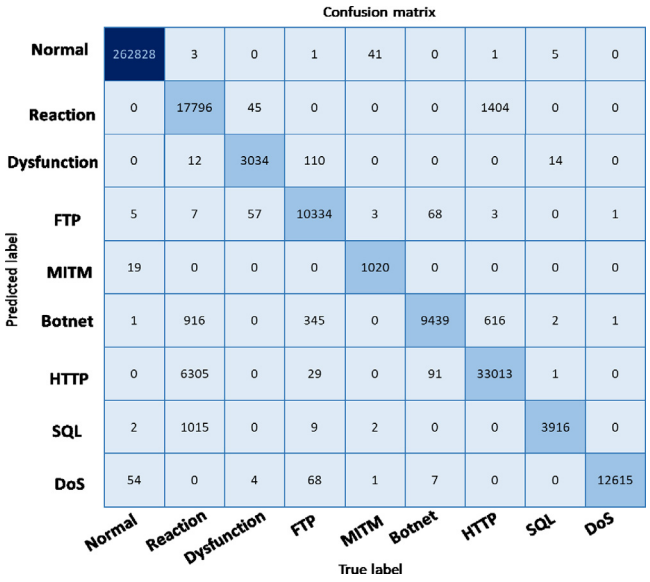

Fig. 6. Confusion matrix after hybridization.

According to Table 8, normal, DoS and MITM traffic have been well classified with 100% for precision, recall and F1-Score. The FPR and FNR have also been improved for FTP, Botnet, and SQL traffic: FTP: (0.04%, 5.1%), Botnet: (0.5%, 3.8%), SQL: (0.2%, 19%) without hybridization against: FTP: (0.002%, 0.26%), Botnet: (0.27%, 2.06%), SQL: (0.15%, 5.5%) respectively with hybridization. As for precision, it has also been improved for FTP, Botnet, HTTP, SQP traffic: (98.6%, 83.08%, 83.7%, 76%) without hybridization against (99.9%, 91.3%, 84.1%, 79%) respectively with hybridization. Except for reaction traffic, the performance metrics of all other traffic have been improved.

Regarding the confusion matrix, we note that all the traffic is classified in the diagonal of the matrix, which means that the traffic classification is correct (which means that the prediction of the label of a traffic corresponds perfectly to the label of the real traffic).

6. Conclusions

BIANO-IDS is a new intrusion detection approach combining two kinds of IDS: anomaly-based IDS and specification-based IDS. The approach has been tested on a real environment and the detection accuracy rate is high in addition to the different performance metrics. However, to further improve performance metrics, we intend to explore other traces such as system logs and apply reduction or selection features methods in the future to improve the computation and training time of neural networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Martyn Williams, The benefits and challenges of IT/OT convergence: Rewriting the rules, 2016, https://www.automation.com/en-us/articles/2016-2/the-benefits-and-challenges-of-itot-convergence-re. (Visited on 2016).

[2] Kaspersky, Kaspersky predicts shift in threat land- scape to ICS in 2023, 2022, https://www.kaspersky.com/about/press-releases/2022_kaspersky-predicts-shifts-in-threat-landscape-to-industrial-control-systems-in-2023. (Visited on 2022).

[3] Junaid Arshad, et al., A review of performance, energy and privacy of intrusion detection systems for IoT, Electronics 9 (4) (2020) 629.

[4] Thomas Menze, The state of industrial cybersecurity in the era of digitalization, 2020, https://ics-cert.kaspersky.com/media/Kaspersky_ARC_ICS-2020-Trend-Report.pdf, Kaspersky.

[5] Mouloud Bourouh, Zakaria Kanoun, Détection d'intrusions à base des réseaux de neu- rones et algorithmes génétiques (Ph.D. thesis), 2017, 14-01- 2018.

[6] Juan Enrique Rubio, et al., Analysis of intrusion detection systems in industrial ecosystems, in: SECRYPT, 2017, pp. 116–128.

[7] Juan Enrique Rubio, et al., Current cyber-defense trends in industrial control systems, Comput. Secur. 87 (2019) 101561.

[8] Debra Anderson, Thane Frivold, Alfonso Valdes, Next-generation intrusion detection expert system (NIDES): A summary, 1995.

[9] Vern Paxson, Bro: A system for detecting network intruders in real-time, in: 7th USENIX Security Symposium, USENIX Security 98, USENIX Association, San Antonio, TX, 1998, (Visited on 1998).

[10] Robin Berthier, William H. Sanders, Specification-based intrusion detection for advanced metering infrastructures, in: 2011 IEEE 17th Pacific Rim International Symposium on Dependable Computing, IEEE, 2011, pp. 184–193.

[11] Marco Caselli, et al., Specification mining for intrusion detection in networked control systems, in: 25th USENIX Security Symposium, USENIX Security 16, 2016, pp. 791–806.

[12] Judith Hochberg, et al., NADIR: An automated system for detecting network intrusion and misuse, Comput. Secur. 12 (3) (1993) 235–248.

[13] F. Barika, K. Hadjar, N. El-Kadhi, Artificial neural network for mobile IDS solution, Secur. Manage. (2009) 271–277.

[14] Igor Nai Fovino, et al., Modbus/DNP3 state-based intrusion detection system, in: 2010 24th IEEE International Conference on Advanced Information Networking and Applications, IEEE, 2010, pp. 729–736.

[15] Franck Sicard, Eric Zamai, Jean-Marie Flaus, Filters based approach with temporal and combinational constraints for cybersecurity of industrial control systems, IFAC-PapersOnLine 51 (24) (2018) 96–103.

[16] Andrea Carcano, et al., A multidimensional critical state analysis for detecting intrusions in SCADA systems, IEEE Trans. Ind. inform. 7 (2) (2011) 179–186.

[17] Masood Parvania, et al., Hybrid control network intrusion detection systems for automated power distribution systems, in: 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, IEEE, 2014, pp. 774–779.

[18] Maëlle Kabir-Querrec, et al., Architecture des systémes d'automatisation des postes résiliente aux attaques des trames GOOSE, 2015.

[19] Oualid Koucham, et al., Efficient mining of temporal safety properties for intrusion detection in industrial control systems, IFAC-PapersOnLine 51 (24) (2018) 1043–1050.

[20] Mohamad Houssein Monzer, Kamal Beydoun, Jean-Marie Flaus, Model based rules generation for intrusion detection system for industrial systems, in: 2019 International Conference on Control, Automation and Diagnosis, ICCAD, IEEE, 2019, pp. 1–6.

[21] Mingtao Wu, Young B. Moon, Alert correlation for detecting cyber-manufacturing attacks and intrusions, J. Comput. Inf. Sci. Eng. 20 (1) (2020) 011004.

[22] Hui Lin, et al., Adapting bro into scada: building a specification-based intrusion detection system for the dnp3 protocol, in: Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop, 2013, pp. 1–4.

[23] Steven Cheung, et al., Using model-based intrusion detection for SCADA networks, in: Proceedings of the SCADA Security Scientific Symposium, Vol. 46, Citeseer, 2007, pp. 1–12.

[24] Rafael Ramos Regis Barbosa, Ramin Sadre, Aiko Pras, Exploiting traffic periodicity in industrial control networks, Int. J. Crit. Infrastruct. Prot. 13 (2016) 52–62.

[25] Leandros Maglaras, et al., Teaching the process of building an intrusion detection system using datafrom a small-scale SCADA testbed, Internet Technol. Lett. 3 (1) (2020) e132.

[26] Wenli Shang, et al., Industrial communication intrusion detection algorithm based on improved oneclass SVM, in: 2015 World Congress on Industrial Control Systems Security, WCICSS, IEEE, 2015, pp. 21–25.

[27] Junlei Qian, et al., Cyber-physical integrated intru- sion detection scheme in SCADA system of process manufacturing industry, IEEE Access 8 (2020) 147471–147481.

[28] Jinping Liu, et al., Toward security monitoring of industrial cyber–physical systems via hierarchically distributed intrusion detection, in: Expert Systems with Applications, 2020, 113578.

[29] Yi Yang, et al., Multidimensional intrusion detection system for IEC 61850-based SCADA networks, IEEE Trans. Power Deliv. 32 (2) (2016) 1068–1078.

[30] Ahmed Saeed, et al., Intelligent intrusion detection in low-power IoTs, ACM Trans. Internet Technol. (TOIT) 16 (4) (2016) 1–25.

[31] Nanda Kumar Thanigaivelan, et al., Distributed internal anomaly detection system for internet of things, in: 2016 13th IEEE Annual Consumer Communications & Networking Conference, CCNC, IEEE, 2016, pp. 319–320.

[32] Faiza Medjek, et al., A trust-based intrusion detection system for mobile rpl based networks, in: 2017 IEEE International Conference on Internet of Things (IThings) and IEEE Green Computing and Commu- Nications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), IEEE, 2017, pp. 735–742.

[33] Christopher D. McDermott, Andrei Petrovski, Investigation of computational intelligence techniques for intrusion detection in wireless sensor networks, Int. J. Comput. Netw. Commun. 9 (4) (2017).

[34] Johan Becker, My Vester, Intrusion Detection System Framework for Internet of Things (Ph.D. thesis, MS thesis), Dept. Comput. Sci. Eng. Chalmers Univ. Technol. Gothenburg, Sweden, 2017.

[35] Omar A. Alzubi, et al., Optimized machine learning- based intrusion detection system for fog and edge computing environment, Electronics 11 (19) (2022) 3007.

[36] Ali Akbar Movassagh, et al., Artificial neural net- works training algorithm integrating invasive weed optimization with differential evolutionary model, J. Ambient Intell. Humaniz. Comput. (2021) 1–9.

[37] Omar A. Alzubi, et al., An efficient malware detection approach with feature weighting based on harris hawks optimization, Cluster Comput. (2022) 1–19.

[38] Omar A. Alzubi, Issa Qiqieh, Jafar A. Alzubi, Fusion of deep learning based cyberattack detection and classification model for intelligent systems, Cluster Comput. (2022) 1–12.

[39] Catherine.D. Schuman, et al., A survey of neuromor- phic computing and neural networks in hardware, 2017, arXiv preprint arXiv:1705.06963.

[40] S. Agatonovic-Kustrin, R. Beresford, Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research, J. Pharm. Biomed. Anal. 22 (5) (2000) 717–727.

[41] Matjaž Kukar Igor Kononenko, Artificial neural networks, 2007, https://www.sciencedirect.com/topics/computer-science/artificial-neural-network. (Visited on 2007).

[42] Elike Hodo, et al., Threat analysis of IoT networks using artificial neural network intrusion detection system, in: 2016 International Symposium on Networks, Computers and Communications, ISNCC, IEEE, 2016, pp. 1–6.

[43] Kushal Rashmikant Dalal, Analysing the role of supervised and unsupervised machine learning in IoT, in: 2020 International Conference on Electronics and Sustainable Communication Systems, ICESC, IEEE, 2020, pp. 75–79.

[44] Abdulmohsen Almalawi, et al., An unsupervised anomaly-based detection approach for integrity attacks on SCADA systems, Comput. Secur. 46 (2014) 94–110.

[45] Salwa Alem, et al., New dataset for industry 4.0 to address the change in threat landscape, in: Risks and Security of Internet and Systems: 15th International Conference, CRiSIS 2020, Paris, France, November (2020) 4–6, Springer Nature, p. 273, Revised Selected Papers.

[46] D. Brandl, Factory automation: New integration architectures for federated systems - ISA, 2007, https://www.isa.org/intech/20160403/ (Visited on 2016).

[47] Ningxuan Kang, et al., A hierarchical structure of key performance indicators for operation management and continuous improvement in production systems, Int. J. Prod. Res. 54 (21) 6333–6350.

[48] i-scoop, Industry 4.0: the fourth industrial revolution– guide to industrie 4.0, 2017, https://www.i-scoop.eu/industry-4-0/. (Visited on 2017).

Salwa Alem, research professor at ENSIBS (National School of Engineers of South Brittany) in Vannes, in cyberdefense specialty since September 2021.

Holder of an engineering degree in electronics and telecommunications from ENSEA (National School of Electronics and its Applications) in 2008, she worked for more than 5 years as a telecommunications engineer for telephone operators in France (SFR, Bouygues Telecom, NSN, etc.) and two masters in 2009 and 2017 in information systems and decision making system.

The master's degree obtained in 2017 was a research master's degree which allowed her to join the University of South Brittany (UBS) to prepare a Ph.D. obtained in May 2021, whose title was the proposal for an intrusion detection system. for industrial 4.0 equipment

After obtaining her Ph.D., she held a post-doc until September 2021 on the proposal of a fungal contamination prediction system for a brioche factory. This position was in collaboration between UBS and a Breton biscuit and brioche factory.

Today, beyond her responsibilities as a teacher, her research focuses on the detection of intrusions for industrial equipment, the analysis of the threat as well as the analysis and study of the human factor in the industrial cybersecurity.