An Ensemble Learning-Based Architecture for Security Detection in IoT Infrastructures

Adrien Hemmer, Mohamed Abderrahim, Remi Badonnel, Isabelle Chrisment

An Ensemble Learning-Based Architecture for Security Detection in IoT Infrastructures

Adrien Hemmer, Mohamed Abderrahim, Remi Badonnel and Isabelle Chrisment
Inria Nancy Grand Est - Loria, University of Lorraine
Campus Scientifique, 54600 Villers-les-Nancy, France
adrien.hemmer, mohamed.abderrahim, remi.badonnel, isabelle.chrisment@inria.fr

Abstract—**The Internet of Things has known an important development. However, security management is still a key challenge in particular for deploying complex IoT systems that provide sophisticated services. In this paper, we design an ensemble learning-based architecture to support early security detection in the context of multi-step attacks, by leveraging the performance of different detection techniques. The architecture relies on a total of five major methods, including process mining, elliptic envelope, one class support vector machine, local outlier factor and isolation forest. We describe the main components of this architecture and their interactions, from the data pre-processing to the generation of alerts, through the calculation of scores. The different detection methods are executed in parallel, and their results are combined by an ensemble learning strategy in order to improve the overall detection performance. We develop a proof-of-concept prototype and perform a large set of experiments to quantify the benefits and limits of this approach based on industrial datasets.**

Index Terms—**Security Management, Internet of Things, Management Architecture, Ensemble Learning, Process Mining, Machine Learning.**

I. Introduction

The Internet of Things (IoT) is an emerging paradigm that has rapidly gained ground. It consists in the pervasive presence around us of a large variety of connected objects (e.g. actuators, sensors, mobile phones) that are able to collect and exchange information and to cooperate with other devices to fulfill common goals [1]. This paradigm has different application domains, such as industry, smart cities, energy, health care, smart homes, transport, biometrics or agriculture [2]. However, the complexity of these systems built from connected objects, together with their often constrained resources, makes them particularly vulnerable to security attacks, such as distributed denial of service (DDoS) attacks, eavesdropping attacks, spoofing attacks, and malware attacks [3].

Different methods have been proposed to detect these attacks. In this context, machine learning solutions have been investigated and shown interesting performance [3]. The concept beyond these methods is typically to consider traces and logs in order to build common patterns corresponding to such attacks. An alternative strategy, known as anomaly detection, consists in using machine learning to determine the normal behavior of the system, and considering deviations as the symptoms of attacks and misbehaviors. Instead of using a single detection method, some works have proposed combining several of these methods to improve the detection performance. In particular, the approaches based on ensemble learning have been exploited in different fields [4]. Moreover, the sophistication of attacks against these IoT systems has also increased with the development of advanced persistent threats (APT) attacks, taking the form of multi-phase scenarios, that may use to their advantages the complexity of IoT systems, in order to remain as invisible as possible [5]. The detection of these scenarios requires to take into account the attack strategy, as well as the causal relationships amongst its different phases. It is important to detect the misbehavior of the IoT systems from the early phases, in order to properly activate counter-measures and mitigate these security attacks.

In this paper, we design an ensemble learning-based architecture aiming at supporting an early detection of multi-phase attacks in IoT infrastructures. It consists in leveraging the performance of different anomaly detection methods, namely process mining, elliptic envelope, one class support vector machine (OCSVM), local outlier factor and isolation forest, that are executed in parallel for the different phases of the attacks. The status of the observed system is determined by combining efficiently the results of each of these detection methods and applying an overall scoring function. The solution relies on the building of dependency graphs, typically using a cross-correlation of data sources, that permits to identify the phases and the relationships amongst them based on the structure and behavior of the IoT systems. It also supports an adaptive feedback mechanism in order to increase the detection performance. A proof-of-concept prototype of the solution has been developed, using the ProM [6] and Scikit-learn [7] libraries, and experimental results have shown the relevance of our proposal for supporting early security detection in these systems.

The main contributions of this paper include: (1) the design of an ensemble learning-based architecture for security detection in IoT infrastructures, that leverages the performance of five different major detection methods, (2) an approach for building dependency graphs amongst data sources, and facilitating the characterization of multi-phase attacks against these systems, (3) the formalization of our solution considering four ensemble learning-based scoring methods, combined with an adaptive feedback mechanism to achieve the best trade-off between accuracy and time, (4) the development of a proof-of-concept prototype complemented by large series of experiments based on industrial datasets, in order to quantify the benefits and limits of the proposed approach.

The remainder of the paper is organized as follows. Section II provides related work in this area. Section III presents our ensemble learning-based approach, with the description of the considered architecture. In particular, it formalizes the building of dependency graphs, the detection of attacks based on scoring methods, and the adaptive mechanism to leverage performance. Section IV describes the experimental setup with the proof-of-concept prototype, and evaluates the performance of the proposed approach based on large series of experiments. Section V concludes the paper and points out our future research perspectives.

II. Related Work

The large-scale deployment of connected objects has contributed to the development of complex IoT environments that may rely on different types of devices and protocols. Their complexity makes security management more difficult to be addressed. In addition to common elementary security attacks, such as denial of service (DoS) attacks, distributed denial of service (DDoS) attacks, eavesdropping, spoofing attacks [3], they are also facing more elaborated security attacks, that are composed of multiple phases (or steps). Several works have been proposed to predict these multi-phase attacks. In [5], these methods are classified into five categories. First, the similarity-based approaches construct the attack scenario based on the degree of similarity amongst a set of traces [8]. Second, the causal correlation approaches focus on the causal relationship between the different phases [9]. Third, structural-based approaches that project the traces to the network architecture [10]. Fourth, the detection methods are driven by the specification of well-known attacks [11]. Finally, a mix of the previous approaches is followed in [12]. However, none of these methods are considering using ensemble learning-based techniques for multi-phase attacks in IoT infrastructures, as we are doing in our proposed approach.

While each detection method has its own advantages and drawbacks and none of them may outperform the others in all cases, ensemble learning consists in using several detection methods in parallel and combining their results instead of relying on only one of them. It tends to perform better than specific detection methods [4]. For this reason, many studies have proposed to use it in different fields. However, only few of them dealt with security detection based on anomalies. In [13], the authors design an ensemble learning solution based on auto-encoder, random forest and support vector machine techniques. They determine the abnormality of data points by using majority voting. In [14], the authors address anomaly detection in 5G radio access networks. They specify an ensemble learning composed of long and short term memory models and show that their approach is well suited to environments where anomalies are difficult to identify. In [15], the authors deal with anomaly detection in industrial systems. They compare the performance of six machine learning methods (e.g., W-KNN, random forest, boosted tree, support vector machine, rotation forest) and investigate an ensemble learning architecture relying on them to improve the performance of these systems. In [16], the authors design an ensemble learning approach for real-time anomaly detection based on a limited number of methods, including perceptron and support vector machine techniques. These methods are managed by an algorithm called ADWIN, that automatically detects and adapts data change rates in order to minimize anomalies. In [17], a solution is designed for IoT infrastructures, but not considering the case of multi-phase attacks. In [18], the authors describe an ensemble learning solution, exploiting different supervised models to ensure security, but focusing on operational cellular networks.

There exists a large variety of detection methods for identifying anomalies. Anomaly detection consists in determining a normal class that represents the majority of data and an abnormal class representing the outliers, and is applied in different domains, including fraud detection, monitoring and cyber-security. In particular, anomaly detection based on elliptic envelope [19] establishes a normal class based on a boundary ellipse to the central data points, while techniques such as support vector machines [20] classify the data points in one or several subspaces. Methods such as local outlier factor [21] consider the sparsity of data, and evaluate the density of a point based on its nearest data points. Finally, solutions such as isolation forest [22] rely on the building of specific trees to classify data points, and techniques based on process mining consist in establishing the normal class based on an extracted process pattern [23], [24]. These individual detection methods will be further detailed in the paper, as they will be considered to establish our ensemble learning-based approach for IoT infrastructures.

There are several works that aim to improve the performance of such anomaly detection methods, by considering feedback mechanisms, or to adapt existing algorithms in order to facilitate their application on streaming data, by using temporal sliding windows. For example, in [25], the authors have proposed an approach based on the isolation forest algorithm using no-overlapping sliding windows to process data streams. In [26], the authors have adapted the variational autoencoder (VAE) algorithm considering and comparing different sizes of temporal sliding windows. Finally, in [27], the authors have adapted an anomaly detection method based on dynamic markov models, and established a balance between the order of the markov model and the effective size of sliding windows. However, these different works are not taking into account ensemble learning considerations.

III. Proposed Architecture

We propose an ensemble learning-based architecture for supporting security detection in IoT infrastructures. We will first overview this architecture, its different components and their interactions to support the detection of multi-phase attacks against IoT infrastructures. We will then formalize three important mechanisms related to this solution: the building of dependency graphs that allows us to establish relationships amongst data sources with respect to the different phases of security attacks, the security detection based on ensemble learning for leveraging the performance of five major detection methods using different scoring methods, and finally an adaptive mechanism enabling a feedback loop on the solution in order to improve detection time performance.

A. Architectural Overview

This ensemble learning-based architecture is described on Figure 1, with the different components and their interactions to support the security detection for IoT infrastructures. It can be seen as a pipeline starting with the raw data coming from the IoT infrastructure that is monitored. These raw data are transformed by a data pre-processing block into refined data, which are then interpretable by the considered detection

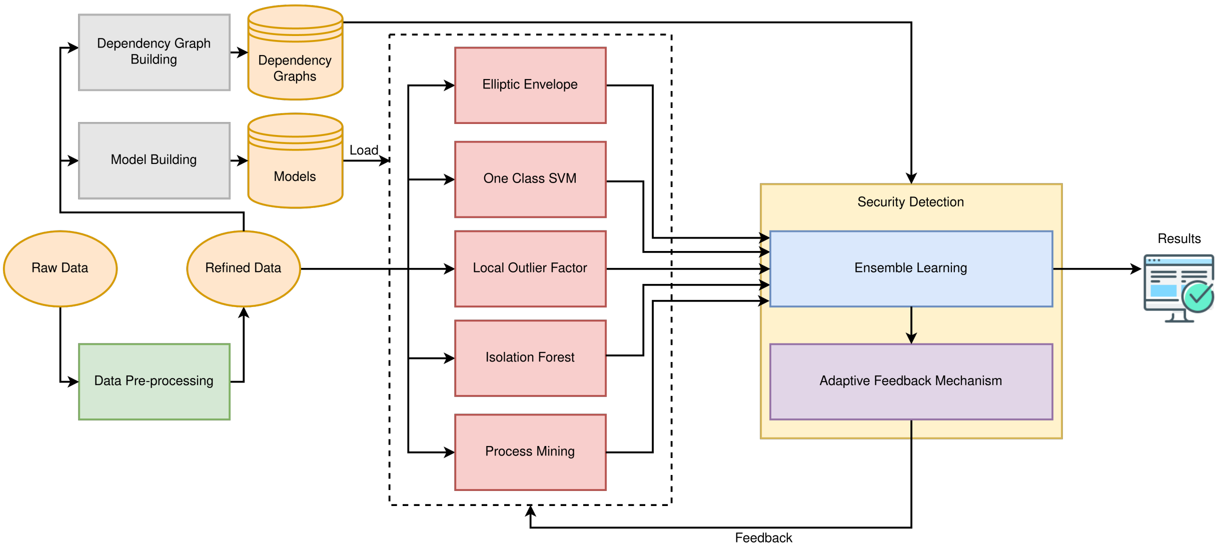

Fig. 1. Overview of our ensemble learning-based architecture for security detection in IoT infrastructures

methods. The refined data are exploited at different stages. First, they are used during the training stage to build the models that will be then used by the detection methods. Second, they are used to build the dependency graphs that permit to identify the dependencies amongst the different data sources of the system. Third, they are used during the detection stage by the detection methods in order to point out deviations from the learnt models and detect potential security attacks and misbehaviors.

Let us look further at the detection stage. The refined data are taken as inputs by the different detection methods (shown in red color on the figure), including process mining, elliptic envelope, one class SVM, local outlier factor and isolation forest, that are executed in parallel and provide their detection results, characterizing the deviation from their behavioral models, to the main security detection block (shown in yellow on the figure). This one implements the ensemble learning strategy, and permits to infer from the individual scores given by detection methods, an overall score that is then compared to a given threshold in order to detect security attacks against the IoT infrastructure. Different ensemble-learning scoring methods have been considered to support the combination of individual scores given by detection methods. The main block also includes an adaptive feedback mechanism (in purple color on the figure) in order to improve the performance of detection methods.

In the following of the paper, we will further detail the different methods that are considered for respectively building the dependency graphs, for supporting the security detection based on the ensemble learning strategy, and finally, for enabling the adaptive feedback mechanism.

B. *Building of Dependency Graphs*

The building of the dependency graphs aims at identifying the dependencies that may exist amongst the different data sources that are used to detect security attacks. These dependency graphs may be deduced from the architecture of the IoT infrastructure, and provides important information about it. Actually, IoT systems typically rely on distributed infrastructures composed of IoT nodes and networks connecting them. Consequently, the communication flows may represent dependencies between the IoT resources. In other words, if a resource is attacked, the ones connected to it will be attacked before the others. Thus, it permits to start a mitigation before critical elements of the system are reached. Figure 2 depicts a simple example of an IoT infrastructure: an IoT node is connected to a Wide Area Network (WAN) and a Local Area Network (LAN), while two IoT nodes are connected to the LAN only. Figure 3 depicts the corresponding dependency graph: the attack begins from the WAN, then it moves to the IoT node 1 before the LAN, and finally affects IoT node 2 and IoT node 3.

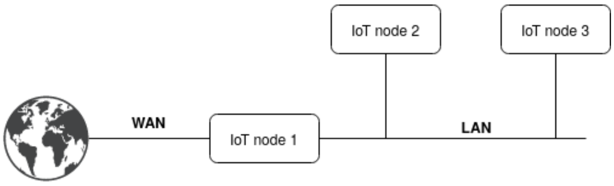

Fig. 2. Example of a simple IoT infrastructure

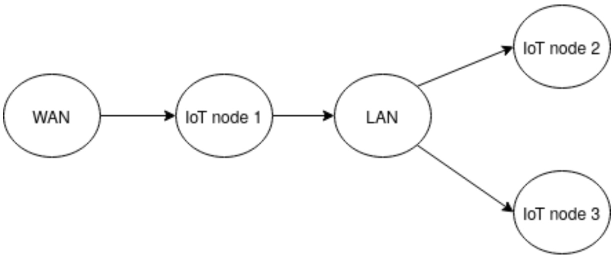

Fig. 3. Resulting dependency graph

The dependency graph may also be checked or built when a real or simulated security attack occurs by calculating correlation between the detection scores of each phase. Since the attack phases do not occur in the same time, it is better to calculate the cross correlation as follows. Let T denote the set of timestamps : $T = \{value_{i1} \in \mathbb{R}; i \in [\![1;n]\!]\}$, The scores returned by the detection methods are gathered in the set S. A score can be modeled as a function which returns a real for each timestamp : $S = \{s_i : T \to \mathbb{R}; i \in [\![1; I]\!]\}$. Let $s_i(t)$ the score of the detection method i at the instant time t, with $\sigma_{s_i(t)}$ corresponding to its standard deviation and $\mu_{s_i(t)}$ corresponding to its mean. The Pearson's correlation, noted $\rho_{s_i(t)s_j(t')}$, for two detection scores $s_i(t)$ and $s_j(t')$ in S stands for their covariance divided by the product of their standard deviation, as given by Equation 1.

$$\rho_{s_i(t)s_j(t')} = \frac{\mathbb{E}[\,(s_i(t) - \mu_{s_i(t)})(s_j(t') - \mu_{s_j(t')})]}{\sigma_{s_i(t)}\sigma_{s_j(t')}} \quad (1)$$

Let τ denote the maximum delay between the two phases of a security attack. The maximum cross correlation, noted C, is the maximum Pearson's correlation of a first detection score and a second shifted one. The second score is shifted between $-\tau$ and τ representing the maximum delay considered between two attack phases, as given by Equation 2.

$$C = \max_{-\tau \le d \le \tau} \frac{\mathbb{E}[\,(s_i(t) - \mu_{s_i(t)})(s_j(t+d) - \mu_{s_j(t+d)})]}{\sigma_{s_i(t)}\sigma_{s_j(t+d)}} \quad (2)$$

It is worth mentioning that these dependency graphs can also be provided by experts, using both the knowledge on the system architecture and correlation techniques in a complementary manner.

C. Security Detection Based on Ensemble Learning

We will now describe the security detection approach based on an ensemble learning-based strategy. This strategy relies on the detection scores provided by different detection methods applied on the refined data and executed in parallel, and this simultaneously on the different data sources. The scores for a given data source are then combined in order to establish an overall score, and detect potential security attacks and misbehaviors. We will first detail the different detection methods considered for the architecture, and then present the different scoring methods used to combine their results. We considered a total of five major detection methods, corresponding to the main categories of detection methods analyzed in [28], and corresponding to probabilistic, statistical, proximity-based, and isolation-based methods, that we complemented by an additional detection method based on process mining. The idea behind the use of these methods, that rely on different detection mechanisms, is to maintain high detection performance. Indeed, as they function differently, we can expect that, for a particular type of attack, at least one of them detect an anomaly. Thus, by combining them, it should detect more diverse attack types. These different detection methods are detailed below:

- The probabilistic detection method, called elliptic envelope, supposes that the refined data follow a Gaussian distribution, and determines the parameters of the distribution that best match the available data points in the training stage. After this, the detection method defines a boundary ellipse to the central data points and then considers the outsiders to be anomalous in the detection stage.
- The linear detection method, called One Class SVM (OCSVM), embeds the maximum of data points from the refined data in a subspace having a dimension lower than the features space. Hence, the training stage consists to find this subspace, whereas the points that do not fit the embedding are considered anomalous in the detection stage.
- The proximity-based detection method, called Local Outlier Factor, considers the sparsity of refined data from the trace with the distances between data points and the local densities. Thus, the method considers a data point as anomalous if its proximity is sparsely populated.
- The isolation-based detection method, called isolation forest, randomly splits refined data into subsets having reduced dimensions and builds a binary tree for each subset. The tree is obtained by recursively splitting the data subset based on a randomly selected attribute until a maximum tree height is reached, only one record remains in the subset or all the remaining records have the same values. Concerning the detection stage, the new data are passed through the trees built in the previous stage and their progress in the different trees permits to identify misbehaviors.
- The process mining detection method relies on the combination of two machine learning techniques. First, a clustering technique is used on the refined data in order to extract the states of the observed system, then a process mining algorithm is applied to build the behavior models of this system. This approach has been described in details in [24].

These different detection methods provide an individual score, noted s_i, as previously mentioned. The scores are then combined in order to obtain an overall score, using an ensemble learning-based strategy. This latter consists in a linear weighted algorithm, as defined in Equation 3, where I stands for the total number of detection methods. In our case, this variable is set to 5, while W_i indicates the weight associated to each detection method.

$$score = \sum_{i=1}^{I} W_i s_i \quad (3)$$

We considered different scoring methods based on this linear weighted scoring, and differing from the calculation of weights. In particular, we focused on the four scoring methods, noted EL_{uni}, EL_{acc}, EL_{exp}, EL_{max}, defined in [18], and standing respectively for the overall scores noted $score_{uni}$, $score_{acc}$, $score_{exp}$, and $score_{max}$. The first scoring method considers a uniform weight for combining the detection methods, as given by Equation 4.

$$score_{uni} = \sum_{i=1}^{I} \frac{1}{I} s_i \quad (4)$$

The second scoring method gives more weight to the detection method characterized by the highest accuracy, with the a_i variable standing for the accuracy of the i^{th} considered detection method, as given by Equation 5.

$$score_{acc} = \sum_{i=1}^{I} \frac{a_i}{\sum_{j=1}^{I} a_j} s_i \quad (5)$$

The third scoring method amplifies the importance of the accuracy, by adding an exponential parameter λ and reduces the influence of low accuracy predictors, as given by Equation 6. It is to be noticed that we have chosen to use the same value of lambda (i.e. value set to 10), corresponding to the value recommended in [18].

$$score_{exp} = \sum_{i=1}^{I} \frac{e^{\lambda ai}}{\sum_{j=1}^{I} e^{\lambda aj}} s_i \tag{6}$$

Finally, the fourth detection method aims at decreasing the detection time by only considering the higher detection score provided by the detection methods, as defined by Equation 7.

$$score_{max} = \textstyle\sum_{i=1}^{I} W_{max_i} s_i \quad with: \tag{7}$$

$$W_{max_i} = \begin{cases} 1 & \text{if } s_i = \max\limits_{\forall j \in I}(s_j) \ and \ \sum\limits_{j=0}^{i-1} \mathrm{W}_{max_j} = 0 \\ 0 & \text{else} \end{cases}$$

Once this overall score is calculated by the scoring method, it is compared to the detection threshold in order to identify potential security attacks and misbehaviors. This ensemble learning-based strategy permits to leverage the performance of the different detection methods, and is complemented by an adaptive feedback mechanism.

D. Adaptive Feedback Mechanism

The proposed architecture integrates an adaptive feedback mechanism capable to impact on the configuration of detection methods depending on the results obtained by the ensemble learning-based strategy, as shown by the feedback loop on Figure 1 going from the security detection block (in yellow color in the figure) to the detection methods (in red color in the figure). This mechanism relies on a temporal sliding window, and takes into account the dependency graph that is precedently built. This window corresponds to the time interval used to calculate the score, and is sliding every second. The objective is to increase the reactivity of detection methods on the other data sources, once a security attack (or one phase of this attack) has been detected on a first data source, by reducing the size of the window. Concretely speaking, the initial size of the sliding windows considered for the detection methods is set to an initial value, noted α, before any detection of security attack phases, while it is decreases of a value β each time a phase of the attack is detected in accordance with the dependency graph. The temporal sliding window sizes are kept higher than a given minimal value α_{min}, in order to prevent false positives.

Let us consider the simple example described by Figure 2, together with the resulting dependency graph given on Figure 3 that contains 5 nodes, corresponding to different data sources. This graph permits to drive the adaptive feedback mechanism. Let's consider that a security attack is observed on the data source corresponding to the WAN node, on which the different detection methods are applied in parallel, then the adaptive feedback mechanism will reduce the initial temporal sliding windows α of a given decrement β, as long as α is higher or equal to α_{min}. This applies for all the other data sources on which the detection methods are also applied in parallel, or may be limited to the successor nodes specified on the dependency graph, depending on scenarios. This mechanism based on a sliding window addresses the case of multi-phase attacks, by increasing the reactivity of detection methods, once the first phase of the given multi-phase attack has been detected.

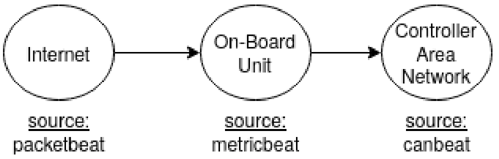

Fig. 4. Dependency graph of the analyzed datasets

IV. Performance Evaluation

We will now detail the extensive series of experiments that have been done to evaluate the performance of the proposed ensemble learning-based architecture. We will start by describing the experimental setup, and then analyze the results related to the dependency graph building and the performance of the ensemble-learning based strategy, in the case of a multi-phase attack.

A. Experimental Setup

The experimental setup relies on a proof-of-concept prototype developed in Python 3.6, which implements the ensemble learning strategy together with the four scoring methods, and exploits the ProM library [6] and the Scikit-learn library [7] to support the five detection methods previously mentioned. The experiments have been performed over a computer with an Intel 4th-generation Core i5 3.3 Ghz processor and 16 GB of RAM memory. During these experiments, we considered three datasets provided by the industrial partners of the H2020 SecureIoT european project [29]. These datasets correspond to the different phases of a security attack performed over a simulated connected car. They therefore provide the access to data attributes such as the CPU usage, the number of bytes that have been written on the disk between two data points, as well as the bus load of the controller are network (CAN). These data are collected by dedicated probes at runtime, and are then send to an Elasticsearch instance, where we can easily retrieve and analyze them. The considered connected car has an on-board unit that connects it to the Internet, while the CAN bus permits to interconnect the car devices amongst them. The multi-phase attack starts with a port scan that allows the attacker to identify an open port. This port is used to send a malicious file that performs a denial-of-service attack against the CAN bus. The first dataset is obtained from a packetbeat probe, which collects information about the network status. The second dataset comes from a metricbeat probe, which collects information about the writing of files on car storage disks. The last dataset corresponds to the canbeat probe, which provides statistics regarding the CAN bus.

B. Dependency Graph Building

We have first established the dependency graph deduced from the physical architecture of the connected car, and presented on Figure 4. It shows the dependencies that exist amongst the three data sources considered for the IoT environment. In particular, the Internet (i.e., WAN) is connected to the on-board unit (i.e., IoT node), which is in turn

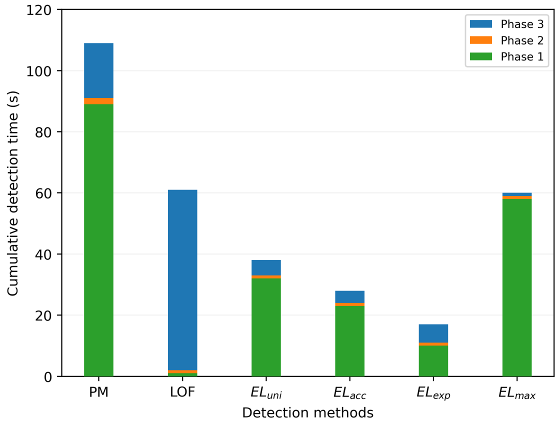

Fig. 5. Comparison of the cumulative detection times for the three phases of the considered attack, with different detection methods (including process mining method, local outlier factor method, and the four ensemble learning-based methods)

connected to the controller area network (i.e., LAN). The data sources used to monitor each data source correspond to the packetbeat, the metricbeat and the canbeat, respectively. We were also interested in complementing this analysis by measuring the maximum cross-correlation of the detection method scores in the case of the analyzed datasets. For each source of measures (i.e., packetbeat, metricbeat, canbeat), the individual detection scores have been calculated by the individual detection methods, as well as a perfect classifier, which only serves a a baseline and classifies the data points without errors. The results have shown that the highest values of correlation were obtained between the first and second phase of the attack (data sources corresponding to the packetbeat and metricbeat probes, respectively), while the lowest values are obtained between the second and the third phase of the attack (data sources corresponding ot the metricbeat and canbeat probes, respectively). They can contribute to the elaboration of dependency graphs, by complementing the knowledge that we have on the system architecture.

C. Performance of the Ensemble Learning Approach

In a next series of experiments, we were interested in evaluating the performance of our ensemble learning approach. The objective was to quantify to what extent the solution can leverage the performance of the set of individual detection methods previously mentioned, namely elliptic envelope, one class support vector machine, local outlier factor, isolation forest and process mining methods, applied on the IoT infrastructure. We executed our proof-of-concept prototype using the different data sources (provided by the packetbeat, metricbeat and canbeat probes), in order to detect the phases of the considered security attack. In particular, we measured the cumulative time and the accuracy for the three attack phases using the different detection methods, including both individual and ensemble learning-based ones. Note that the algorithms implementing the detection methods are not directly running on IoT devices, that are typically characterized by scarce resources, but are launched on the testbed computer resources, that may be outsourced in a cloud infrastructure or at the network edge.

TABLE I
ACCURACY OBTAINED FOR THE THREE PHASES OF THE CONSIDERED ATTACK USING DIFFERENT DETECTION METHODS

Methods	Phase 1	Phase 2	Phase 3
PM	82.61%	84.78%	97.78%
LOF	86.20%	13.51%	87.74%
EL$_{uni}$	56.05%	85.93%	99.15%
EL$_{acc}$	70.91%	85.93%	99.37%
EL$_{exp}$	97.24%	85.92%	98.94%
EL$_{max}$	60.08%	84.01%	99.79%

Figure 5 represents the cumulative time required to detect the three phases of the security attack, using a sliding window with a fixed size of 60 seconds. The security detection is performed using the four ensemble learning-based methods, noted $\mathrm{EL}_{\mathrm{uni}}$, $\mathrm{EL}_{\mathrm{acc}}$, $\mathrm{EL}_{\mathrm{exp}}$ and $\mathrm{EL}_{\mathrm{max}}$, specified in the previous section. We also plotted the results obtained for two individual detection methods, corresponding to two observed extreme cases serving as a baseline, the process mining detection method which produces the worst results in terms of detection time, and the local outlier factor detection method which generates the best results in terms of detection time amongst the individual detection methods. Moreover, these two methods have been able to detect all phases of the attack contrary to, for example, the isolation forest which misses the second phase. Considering now all the methods, the ensemble learning-based method EL_{exp} provides the best results in terms of detection time with a total time of around 18 seconds to identify the three phases of the security attacks. The other ensemble learning-based detection methods using the uniform, the accuracy and the maximum weights showed respectively a cumulative detection time of around 38 seconds, 28 seconds and 60 seconds. Therefore, their cumulative times are still better than the results obtained by using only the local outlier factor method, characterized by a detection time of 61 seconds, or even than the results observed with the process mining method, corresponding to 109 seconds. The relatively low performance of some ensemble learning-based detection methods during the first phase of the security attack can be explained by the particularly noisy nature of the collected data for which several anomaly detection methods have not performed well, except the one class support vector machine method and the local outlier factor detection method.

In the meantime, it is important to look at the accuracy obtained for the three phases of the security attacks using these different detection methods, as shown on Table I. The accuracy performance may vary depending on the data sources for a given detection method. For instance, the local outlier factor method provides interesting performance on the phase 1 of the attack, while they are significantly degraded on the second phase, with an accuracy of 13.51%. In the meantime, the process mining method provides relatively good accuracy performance for the three phases of the security attack, in particular for detecting the sending of the malicious file and the CAN denial of service attack. However, this detection method is characterized by high detection times. It is therefore required to take into account both the detection time and accuracy observed for the different detection methods. When looking further at the ensemble learning-based methods, we can observe that the method providing the best overall accuracy results is the method EL_{exp}, which consists in reducing the influence of low accuracy predictors using an exponential parameter. It gives respectively an accuracy of around 97% for the phase 1, an accuracy of around 86% for the phase 2, and an accuracy of around 99% for the phase 3, and provided the best cumulative detection time.

V. Conclusions

Securing IoT infrastructures is a major issue challenged by the heterogeneity, distribution and scale of these systems, but also by the sophistication of security attacks. We have proposed in this paper an ensemble learning-based architecture for supporting an early detection of multi-phase attacks in IoT infrastructures. The architecture leverages the performance of five major detection methods, namely process mining, elliptic envelope, one class support vector machine, local outlier factor and isolation forest. We have described the main components of this architecture, their operations and the interactions amongst them. In particular, we have specified the building of dependency graphs using a cross-correlation of data sources, to automate the identification of different phases and their relationships based on the structure and behavior of the IoT systems. We have formalized our detection solution by considering four ensemble learning-based scoring methods, that serve as a support to combine the results of the five considered detection methods to increase the detection performance. We have developed a proof-of-concept prototype based on the ProM and Scikit-learn libraries, and evaluated the performance of our proposed approach through a large set of experiments.

As future work, we are interested in performing complementary experiments with additional datasets coming from alternative application domains. We are also interested in investigating to what extent the proposed approach could take benefits and contribute to cyber-threat intelligence solutions for IoT infrastructures.

VI. Acknowledgment

This work has been partially supported by the SecureIoT project, funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 779899; the exploited datasets have been provided by the industrial partners of this project.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] Y. Perwej, K. Haq, F. Parwej, M. Mumdouh, and M. Hassan, "The internet of things (iot) and its application domains," *International Journal of Computer Applications*, vol. 975, no. 8887, p. 182, 2019.

[3] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "Iot security techniques based on machine learning: How do iot devices use ai to enhance security?" *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 41–49, 2018.

[4] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, no. 2, pp. 241–258, 2020.

[5] J. Navarro, A. Deruyver, and P. Parrend, "A systematic survey on multi-step attack detection," *Computers & Security*, vol. 76, pp. 214–249, 2018.

[6] W. Aalst, van der, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Germany: Springer, 2011.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.

[8] B. Chen, J. Lee, and A. S. Wu, "Active event correlation in bro ids to detect multi-stage attacks," in *Fourth IEEE International Workshop on Information Assurance (IWIA'06)*, vol. 1. IEEE Computer Society, 2006, pp. 32–50.

[9] P. Ning and Y. Cui, "An intrusion alert correlator based on prerequisites of intrusions," USA, Tech. Rep., 2002.

[10] S. Noel, E. Robertson, and S. Jajodia, "Correlating intrusion events and building attack scenarios through attack graph distances," in *20th Annual Computer Security Applications Conference*. IEEE, 2004, pp. 350–359.

[11] E. Vasilomanolakis, S. Srinivasa, C. G. Cordero, and M. Mühlhäuser, "Multi-stage attack detection and signature generation with ics honeypots," in *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2016, pp. 1227–1232.

[12] A. A. Ramaki, M. Amini, and R. E. Atani, "Rteca: Real time episode correlation algorithm for multi-step attack scenarios detection," *computers & security*, vol. 49, pp. 206–219, 2015.

[13] D. B. Araya, K. Grolinger, H. F. ElYamany, M. A. Capretz, and G. Bitsuamlak, "An ensemble learning framework for anomaly detection in building energy consumption," *Energy and Buildings*, vol. 144, pp. 191–206, 2017.

[14] T. Sundqvist, M. H. Bhuyan, J. Forsman, and E. Elmroth, "Boosted ensemble learning for anomaly detection in 5g ran," in *Artificial Intelligence Applications and Innovations*, I. Maglogiannis, L. Iliadis, and E. Pimenidis, Eds. Cham: Springer International Publishing, 2020, pp. 15–30.

[15] S. R. Moreno, L. d. S. Coelho, H. V. Ayala, and V. C. Mariani, "Wind turbines anomaly detection based on power curves and ensemble learning," *IET Renewable Power Generation*, vol. 14, no. 19, 2020.

[16] R. R. Reddy, Y. Ramadevi, and K. Sunitha, "Real time anomaly detection using ensembles," in *2014 International Conference on Information Science & Applications (ICISA)*. IEEE, 2014, pp. 1–4.

[17] S. Khare and M. Totaro, "Ensemble learning for detecting attacks and anomalies in iot smart home," in *Proceedings of the 3rd International Conference on Data Intelligence and Security (ICDIS)*, 2020, pp. 56–63.

[18] J. Vanerio and P. Casas, "Ensemble-learning approaches for network security and anomaly detection," in *Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, 2017, pp. 1–6.

[19] M. Hubert and M. Debruyne, "Minimum Covariance Determinant," *WIREs Computational Statistics*, vol. 2, no. 1, pp. 36–43, 2010.

[20] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, 2000, p. 93–104.

[22] F. T. Liu, K. M. Ting, and Z. Zhou, "Isolation Forest," in *Proceedings of the 8th IEEE International Conference on Data Mining*, Dec 2008, pp. 413–422.

[23] A. Hemmer, M. Abderrahim, R. Badonnel, J. François, and I. Chrisment, "Comparative assessment of process mining for supporting iot predictive security," *IEEE Transactions on Network and Service Management*, 2020.

[24] A. Hemmer, R. Badonnel, and I. Chrisment, "A process mining approach for supporting iot predictive security," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–9.

[25] Z. Ding and M. Fei, "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window," *IFAC Proceedings Volumes*, vol. 46, no. 20, pp. 12–17, 2013.

[26] T. Chen, X. Liu, B. Xia, W. Wang, and Y. Lai, "Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder," *IEEE Access*, vol. 8, pp. 47 072–47 081, 2020.

[27] H. Ren, Z. Ye, and Z. Li, "Anomaly detection based on a dynamic markov model," *Information Sciences*, vol. 411, pp. 52–65, 2017.

[28] C. C. Aggarwal, "Outlier analysis," in *Data mining*. Springer, 2015, pp. 237–263.

[29] "Datasets related to the SecureIoT European Project," https://marketplace.secureiot.eu/marketplace/dataset/, Last visited on July 2021.