IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 28, 2018, accepted October 20, 2018, date of publication October 29, 2018,
date of current version December 7, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2878576

Fast Defense System Against Attacks in Software Defined Networks

MARCOS V. O. DE ASSIS[1], MATHEUS P. NOVAES[2], CINARA B. ZERBINI[2], LUIZ F. CARVALHO[2], TAUFIK ABRÃO[3], AND MARIO L. PROENÇA Jr.[2]

[1]Engineering and Exact Department, Federal University of Paraná, Palotina 85950-000, Brazil
[2]Computer Science Department, State University of Londrina, Londrina 86057-970, Brazil
[3]Electrical Engineering Department, State University of Londrina, Londrina 86057-970, Brazil

Corresponding author: Mario L. Proença Jr. (proenca@uel.br)

This work was supported in part by CAPES, due to the concession of scholarships, in part by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grants 308348/2016-8 and 304066/2015-0, and in part the Federal University of Paraná (UFPR) under Project Banpesq/2014016797.

ABSTRACT With the ever-growing data traffic in computer networks nowadays, the management of large-scale networks is a challenge for guaranteeing the quality of the provided services. This is due to the increasingly usage of connected applications, such as Internet of Things and cloud computing environments. Software-defined networking (SDN) is a new paradigm that aims to make this management process easier by centralizing the configuration of all network devices into a single programmable central controller. However, as any centralized service, this architecture is susceptible to security vulnerabilities, such as distributed denial of service (DDoS) and port scan attacks. Thus, security methods are necessary to guarantee the normal operation of SDN's central controller. Furthermore, networks are transporting an increasingly amount of information day by day, which could mean data loss in case of long network unavailability. For this reason, security mechanisms must operate online, with fast-responding countermeasures to mitigate the impact of the detected attacks over the SDN. In this paper, we present a fast SDN defense system against DDoS and port scan attacks, which runs directly into the central controller and uses a game theoretical approach for attack mitigation. For the detection, we compare three different approaches, particle swarm optimization, multi-layer perceptron neural network, and discrete wavelet transform. We test our approach over IP flow data generated over Mininet network emulator, along with floodlight controller, and the presented defense system achieved good outcomes for both detection and mitigation processes.

INDEX TERMS DDoS, DWT, MLP, port scan, PSO, SDN.

I. INTRODUCTION

The amount of network applications and connected devices using the Internet as data transmission environment is rapidly increasing. Web applications, such as online banking, social networks and e-commerces, as well as mobile usage and the emergence of the Internet of Things (IoT) paradigm, are increasing in popularity every day. However, the network performance and the demands required by the referred applications are becoming a complex task for the network administrators to handle due to the heterogenous and static infrastructure of the traditional networks.

Software Defined Networking (SDN) is an emerging network architecture aiming to supply the demands of existing and future connected applications. This new paradigm has as main characteristic the division between the network's planes. In other words, the control and the data planes are decoupled from the network devices through an abstraction plane [1]. This division allows to control, modify and manage the network behavior through a dynamic software interface, unlike the traditional networks where network devices are proprietary locked boxes, which limits its flexibility relating to its internal control [2].

New monitoring and management resources that are able to improve the performance and reduce networks bottlenecks are present in SDN. Despite the discussed characteristics, such as control centralization and network programming, these networks are also subject to threats and security vulnerabilities. Due to the centralized nature of the network intelligence through an SDN controller, as any centralized service, this controller can be targeted by Denial of Service (DoS) attacks [3], [4]. The DoS attack attempts to exhaust the network resources and it is more powerful when performed in a distributed way (Distributed DoS, or DDoS). When attacking servers, the attacker aims to make a service unavailable by sending several requests, while in infrastructure attacks, the attacker overwhelms a network link [5]. Furthermore, DDoS attacks are frequently followed by port scan attacks, where an attacker scans the server's ports in order to find an opening for an intrusion process.

However, the management of the network's information security is a task of high complexity, since it is necessary to guarantee the availability, reliability and integrity of the network services provided to end users. Thus, it is necessary the usage of efficient techniques to help on the autonomous management and security processes, such as anomaly detection and mitigation, on SDN environments. Anomaly detection systems can be classified in two main groups: signature-based and based on the networks normal operation. The first one uses a database which contains the patterns of known anomalies. The second one generates a network's traffic profile, which represents its behavior in normal conditions and does not require knowledge of the anomalies to detect them [6], [7]. The main disadvantage of the profile based approach is the occurrence of false-positive alerts, when the traffic of legitimate users are detected and classified as abnormal [8], [9].

In this paper, we present a system for fast detection and mitigation of DDoS and port scan attacks on SDN environments. The presented system analyzes IP flow data in five-second intervals, providing a faster detection mechanism than traditional anomaly detection approaches, such as [10] and [11], which operate with five-minute time intervals. For this, the presented system is divided into three main modules: Detection, Identification and Mitigation modules.

On the Detection module, we compare the usage of three different models using a multidimensional IP flow analysis. This approach is based on the management of six different IP flow features: bits/s, packets/s, source/destination IP addresses and source/destination ports. The first method uses the Particle Swarm Optimization (PSO) on a non-supervised learning approach based on the data clustering for traffic characterization and an approach based on the Chebyshev Inequality for anomaly detection. The second one is an artificial neural network which uses Multi-Layer Perceptron (MLP), a supervised machine-learning process for anomaly detection. Finally, the third one uses the Discrete Wavelet Transform (DWT), a technique based on signal processing which decomposes the input traffic data into its constituent parts based on its frequency in order to characterize the traffic and detect the presence of anomalies.

On the Mitigation module, we use the game theoretical (GT) approach presented in [12] for DDoS mitigation, which proved itself to be efficient on preventing DDoS attacks over SDN border gateways. In this paper, we test the GT approach efficiency on mitigation DDoS attacks directly into the SDN central controller, which also prevents internal attacks. Furthermore, we extend the operation of the model to provide defense against port scan attacks.

To test the efficiency of the presented system, as well as the performance outcomes of the different methods tested for anomaly detection, we use simulated SDN data generated by Mininet network emulator, together with Floodlight SDN controller and OpenFlow IP flow data.

The main contributions of this paper are:

- A system for SDN defense against DDoS and port scan attacks;
- The performance comparison of three different fast anomaly detection methods on an SDN environment;
- Efficiency analysis on the usage of the mitigation approach presented in [12] directly into the SDN central controller instead of on the border gateway;
- Usage of reliable and replicable data through Mininet network emulator, since it is one of the most used mechanisms in SDN researches nowadays.
- Comparison between the presented anomaly detection methods and classic literature methods.

The remainder of this paper is organized as follows: Section II shows the related works; Section III describes the presented defense system for SDN environments; Section IV describe the anomaly detection methods used on the Detection Module of the presented SDN defense system; Section V discuss the performance results achieved; finally, Section VI presents the conclusions and future works.

II. RELATED WORKS

Software-Defined Networking (SDNs) is a new network paradigm that improves network control. SDN helps solve several problems faced nowadays with our traditional large-scale networks, such as resource allocation and online configuration. Thus, several researches are being performed within this area. Cox *et al.* [13] presented a survey on the state of the art of SDN. They highlighted the efficiency of this architecture, also pointing out implementation cases outside the academia, on companies like Google, AT&T and Microsoft. Furthermore, they describe the advantages and the challenges faced by this technology. Paliwal *et al.* [14] addressed SDN paradigm by presenting an extensive review report on various available central controllers. For each analyzed controller, the authors discussed their design aspects and architecture overview, besides evaluating their efficiency over performance metrics. Zhang *et al.* [15] introduced the concept of SD-ICN networking, which is the junction of SDN paradigm with Information Centric Networking (ICN). ICN is also an emerging network paradigm which uses features like in-network caching and name-based routing to support the ever increasing growth of Internet traffic. According to the authors, the junction of these two promising paradigms is able to improve management and security processes.

However, the centralized architecture in which SDNs operate brings possible security threats, such as Denial of Service (DoS) attacks. As security is a major concern for most network environments, several papers address this issue.

Xu *et al.* [16] proposed a Smart Security Mechanism (SSM) to defend SDN-based Internet of Things (IoT) environment against the new-flow attack. The authors performed simulations and testbed, and the achieved results pointed out the feasibility of the proposed system. Zhang and Sun [17] presented an SDN-based integrated IP source address validation architecture (ISAVA) which can cover both intra and inter-domain areas and effectively lower SDN devices deployment cost. This approach helps protect SDN networks against IP spoofing attacks. Conducted experiments proves that the proposed method was successful in solving the stated problem. Yu *et al.* [18] addressed the security of SDN vehicular networks against DDoS attacks. The authors highlight the vulnerability of the SDN environment against this attack and propose a detection mechanism based on OpenFlow messages, flow feature extraction and Support Vector Machine (SVM) classification. Through a simulation environment, the performed tests achieved effective results. Peng *et al.* [19] presented an anomaly detection method for SDN environments based on double P-value of transductive confidence machines for K-nearest neighbors (K-NN) algorithm. The proposed method is able to detect anomalies and to perform a classification of the detection over IP flows, and the test's outcomes points out a better performance than similar detection approaches. Carvalho *et al.* [20] presented an SDN-based ecosystem able to monitor the traffic of the network and proactively detect anomalies. After an anomalous behavior is detected, a deeper analysis is performed through the usage of multiple OpenFlow features, which are used to optimize mitigation policies to reduce the impact of the attack over the SDN operation.

As network's traffic are increasing day-by-day, the amount of information traveling on them is massive and any problem that causes the network services to become unavailable signify a huge amount of lost data. For instance, on a 10Gb link, up to 3Tb of data may be lost on a 5 minute interval (a common analysis time interval on traditional management and security systems) on a network stoppage. Thus, fast-response or online management systems are required to guarantee the quality of the network provided services. Zhao *et al.* [21] presented a novel framework for real time network traffic anomaly detection using machine learning algorithms. They collected and analyzed in real-time data from the University of Missouri-Kansas City, using big data processing frameworks along with machine learning tools to detect anomalies within the analyzed data. Wang *et al.* [22] proposed a network anomaly traffic detection method based on the IP flow template, capturing and analyzing network traffic in real-time. The authors performed tests in a controlled network environment, and highlight that the proposed approach accurately detect the anomaly network traffic.

One of the most important steps on mitigating the effect of network attacks is the anomaly detection. There is a vast amount of researches in this area due to its high importance and difficulty on providing efficient and fast-responsive solutions. In this paper, we test three different approaches on the anomaly detection step of our proposed SDN defense system: Particle Swarm Optimization (PSO), Multi-Layer Perceptron (MLP) and Discrete Wavelet Transform (DWT). These approaches are widely used in several computer networks study areas.

Several approaches apply clustering algorithms on the development of anomaly detection systems. K-means algorithm is a widely used approach on classification and detection of data anomalies in different areas. However, one of its limitations is the local convergence and sensitivity relating to the centroids of each cluster. Karami and Guerrero-Zapata [23] presented a new anomaly detection system that operates in two phases. In the first one, they applied a hybrid approach of PSO and K-means with two cost functions, one to find the distance between the clusters and another to set the local optimization, which determines the ideal number of clusters. On the second phase, they applied fuzzy logic for the classification on the anomaly detection. Experimental results demonstrated that the proposed algorithm is able to achieve the ideal amount of well-separated clusters, as well as to elevate the detection rate and lower false-positive rates. Lima *et al.* [24] applied PSO together with K-means for a baseline generation for backbone management applied to network's SNMP data. The objective of the PSO usage was to improve the clustering solutions and the cluster's centroids calculation. Numerical results show that detection and false-positive alarms was promising.

Some anomaly detection approaches apply a combination of machine learning techniques that are called ensembles methods. These approaches aims to achieve better predictive performance outcomes. Aburomman and Reaz [25] proposed an intrusion detection system (IDS) applying three different machine learning techniques: Support Vector Machine (SVM), PSO and K-NN. On the training phase, six K-NN classifiers and six SVM classifiers were used on the same data set. Then, the PSO method was applied by combining the output of the twelve classifiers on the generation of a final classifier. To validate the proposed system the authors uses five random subgroups of the KDD99 dataset. The evaluation metric used was the accuracy, which achieves results of 92% on the average.

A Multi-layer Perceptron (MLP) network alongside a Genetic Algorithm (GA) optimization method was proposed by Singh and De [26] for detecting DDoS attacks on the application layer. The algorithm was developed based on the analysis of the fields of received packets, such as HyperText Transfer Protocol (HTTP), the number of IP addresses during a time interval, port number mapping and size of the incoming packets. These four features were used as input for the construction of a classifier. Using MLP and GA, the dataset is classified as attacks or normal users. Experimental results show that MLP-GA provides a 98.04% efficiency rate on detecting DDoS attacks. Siaterlis and Maglaris [27] proposed a MLP classification network for DDoS detection using IP flow data. According to the authors, the number of neurons composing the MLP hidden layer influences the classification results up to $2N + 1$ neurons, where N is the number of neurons on the input layer. Adding more neurons to the hidden layer implies no further improvement on the classification results. The authors highlight the efficiency of the proposed method on detecting DDoS behaviors. Jadidi *et al.* [28] used a MLP network on detecting anomalies alongside with the Gravitational Search Algorithm (GSA) to optimize the neural weights of the MLP, highlighting the efficiency of the method on the stated problem. Furthermore, Nikravesh *et al.* [29] investigated the accuracy of the characterization process of mobile network traffic using the methods MLP, MLP with Weight Decay (MLPWD) and Support Vector Machine (SVM).

Tian and Ding [30] developed a method for network anomalies detection using two tools, a Traffic Matrix (TM) and wavelets. A TM corresponds to five minutes of network traffic, which may contain normal or anomalous traffic. A wavelet transform was performed in this matrix, producing coefficients that provide historical traffic information. Through this historical data, some parameters are collected. A comparison between these coefficients and abnormal coefficients was performed, aiming to identify DDoS attacks. This technique showed high detection rates, close to one hundred percent, and a false alarm rate close to six percent. The disadvantage in using this approach lies in the fact that the matrices have samples of five minute intervals, which in a current network with links of 10Gb, 100Gb, or even 400Gb, means the exchange of large amounts of information and data, causing the detection occurrence late and therefore ineffective. Still using wavelets, it is important to highlight the work of Kanarachos *et al.* [31] and Gao *et al.* [32], both developed in the traditional network environment. Kanarachos *et al.* [31] proposed a system that uses a combination of three techniques: wavelets, neural networks and Hilbert transform. The model is divided into three stages. In the first stage a wavelet transform of Daubechies with eight levels of decomposition was performed, and then a noise removal technique was applied. In the second stage, a subset of the noise-free data was chosen for neural network training, generating a traffic forecast. The third and last stage used the Hilbert transform in the signal error, which is the signal resulting from the difference between the filtered signal and the training output of the neural network. Gao *et al.* [32] presented an anomaly detection model using *wavelet packet*, which is a generalization of the pyramidal algorithm of the traditional wavelet transform. The authors have defined that the proposed method is capable of detecting ‘‘long-term’’ anomalies and medium frequencies. The system also guarantees an improvement in the reliability of the detection using an adaptive reconstruction of the detail coefficients from the wavelet transform, including anomaly. The presented methods scored satisfactory detection rates, but as they were developed for traditional networks, mitigation routines became more complex to implement.

In this paper, we present a defense system for SDN controllers able to detect and mitigate both DDoS and port scan attacks. The system operates online, collecting and analyzing IP flow data directly into the SDN controller every 5 seconds to detect these kinds of anomaly. Unlike other traditional anomaly detection systems, our proposal quickly responds to the detected threat by triggering a mitigation process, performed by a game theoretical approach, five seconds after the detection.

III. SDN DEFENSE SYSTEM

In this section, we describe the presented SDN defense system, its organization and operation. It aims to help defending the SDN central controller against DDoS and port scan attacks. To achieve this objective, an hexa-dimensional IP flow analysis is used through the collection of different IP flow features. These dimensions are used to characterize the network's normal behavior and, later, to detect the occurrence of a network anomaly or attack.

The presented SDN defense system is mainly composed of an IP flow exporter and three modules. The flow exporter protocol used on the development of this paper was OpenFlow. Each one of the three modules are composed of two other sub-modules, as described by Fig. 1. They are the Detection, Identification and Mitigation modules.

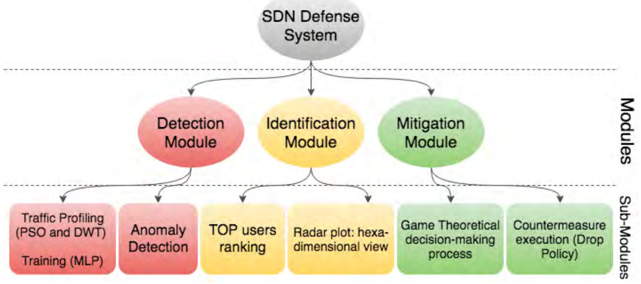
SDN Defense System
Detection Module
Identification Module
Mitigation Module
Modules
Traffic Profiling (PSO and DWT)
Training (MLP)
Anomaly Detection
TOP users ranking
Radar plot: hexa-dimensional view
Game Theoretical decision-making process
Countermeasure execution (Drop Policy)
Sub-Modules

FIGURE 1. General view of the presented SDN Defense System.

The Detection Module performs the detection of the anomaly/attack on the SDN. To perform this task, different methods were applied and compared in this paper to find which one is the most efficient approach. They are the Particle Swarm Optimization (PSO), the Multi-Layer Perceptron (MLP) neural network and the Discrete Wavelet Transform (DWT). Each one of them will be further described on Section IV. As shown by Fig. 1, this module is composed by two sub-modules: ‘‘Traffic profiling / training’’ and ‘‘anomaly detection.’’ The first one is divided into two parts due to the fact that the mentioned methods operates differently in this step. The PSO and DWT are non-supervised methods, which generate a normal online profile (traffic profiling every 5 seconds) of the analyzed SDN, *i.e.*, the network profile is generated on each IP flow collection. On the other hand, the MLP method is a supervised learning approach, requiring previous information (training) about the network's normal operation, as well as the anomalies and attacks (here defined as DDoS and port scan). The second sub-module, the anomaly detection, is responsible for detecting the abnormal behavior of the six analyzed dimensions that occurs when an attack is being performed.

The Identification Module deals with the identification of the attack on the SDN. It is an essential step towards a good mitigation process since different kinds of attacks require specific mitigation policies to achieve a satisfactory outcome. As observed in Fig. 1, it is composed of the sub-modules "Top users ranking" and "Radar plot." The first one stores the three most frequent source and destination ports and IP addresses, as well as the three most frequent protocols. The second one provides a general view of the network behavior on a single time interval. Together, both sub-modules help the identification of the attack, as well as the likely attackers and victims. PSO and DWT methods rely on this module for correct mitigation guidance, since they only detect the occurrence of anomalies. MLP method operates as a classifier, enabling the detection step in the identification of the detected anomaly if it is known by the system (present on the training step). Finally, when no attack is detected by the Detection Module, the Identification Module uses the collected data to feed a list of all source IP addresses analyzed on the past 5 minutes, here called "safe list." This list is used to prevent the packet drop of legitimate users.

Finally, the Mitigation Module is responsible for taking the optimal countermeasures against the detected and identified attack. As described in Fig. 1, it is composed of the sub-modules "Game Theoretical decision-making process" and "countermeasure execution." For the decision-making process, we used a game theoretical approach, presented in [12], that aims to mitigate DDoS attacks at SDN border gateways in order to protect the central controller. However, in this paper, we applied this approach directly into the SDN controller, which analyzes the traffic data with the three presented detection methods and triggers an alarm in case of a DDoS or a port scan detection. This alarm will activate the mitigation module, which will provide the SDN controller an optimal packet drop rate and a list of legitimate users or "secure hosts."

If the detected attack is a port scan, the countermeasure approach is to simply drop all packets of the attacker's source IP address. This identification is possible due to the operation of the Identification Module, as shown by de Assis *et al.* [12], where, using this approach, it was possible to identify significant information about DoS, DDoS, port scans and flash crowd anomalies. However, if the detected attack is a DDoS, then the game theory is invoked.

The game theoretical approach we use is a two-player game. As the attacker (malicious user and first player) tries to maximize the damage caused to the network while reducing its chance of being detected, the defense mechanism (second player) tries to reduce the impact posed by the attacker and preserve the SDN normal operation. Furthermore, it is a zero-sum game, *i.e.*, the gain of one player is the loss of another.

Each player has a set of possible actions that must be performed to increase its gain or payoff. For the attacker, it is possible to:

- Change the number of packets per second directed to the network by each attacking node;
- Modify the number of attacking nodes;

On the other hand, the defense mechanism is able to:

- Allow packets to traffic through the SDN controller;
- Drop packets to prevent them from being further processed by the SDN controller;

In order to measure the impact of each one of these actions, several metrics are used. They are i) the normalized error between the expected SDN behavior and the analyzed time interval, ii) the average bandwidth consumption of legitimate users in comparison to malicious ones, iii) the attack cost for the attacker, iv) and the estimated packet loss of legitimate users through the dropping process. For more implementation details, please refer to [12].

The second sub-module (countermeasure execution) generates as outcome a set of packet dropping policies that is provided to the SDN's central controller for instant implementation. It is important to highlight that the data provided by the Identification Module prevents the SDN controller from dropping some known legitimate users, which greatly improves the results of DDoS attacks, as shown by de Assis *et al.* [12].

An important characteristic of the presented defense system is its speed on detecting and taking the adequate countermeasure to mitigate the attack. It was designed to operate online and, thus, the entire process occurs in an autonomic way, *i.e.*, no human intervention occur besides receiving the alarms and reports about the detected attacks.

To enable this online characteristic, the network controller collects and exports IP flows every 5 seconds, submitting this data to a detection analysis. Thus, the mitigation process may quickly start and prevent further damage by the attacks.

The overall operation of the presented SDN defense system is shown by Fig. 2.

As shown, every 5 seconds the SDN controller exports through OpenFlow six flow dimensions: bits/s, packets/s, source IP address, destination IP address, source port and destination port. The first two dimensions are quantitative values, while the remaining are qualitative ones. To enable their usage on the anomaly detection process, they need to be converted into quantitative data. Thus, we apply the Shannon Entropy [33], which enables the information extraction relating to concentration and dispersion of data in these flow dimensions. For this purpose, given an feature $X = \{x_1, x_2, \ldots, x_n\}$ in which x_i represents the number of occurrences of the sample i at the time interval, the entropy H for X is given by:

$$H(X) = -\sum_{i=1}^{N} \left(\frac{x_i}{S}\right) log_2 \left(\frac{x_i}{S}\right) \tag{1}$$

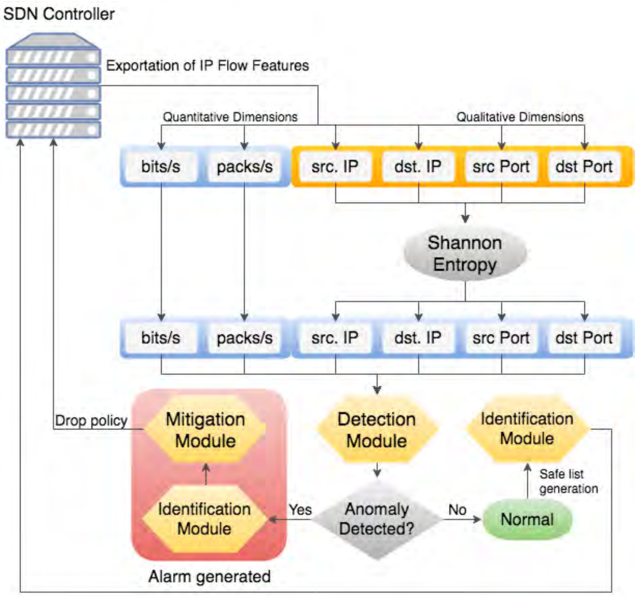

FIGURE 2. **SDN defense system operation.**

where $S = \sum_{i=1}^{N} x_i$ is the sum of all the values present on the histogram.

Then, the six dimensions are submitted to the Detection module. If an anomaly/attack is detected, then an alarm is generated and the data is submitted to the Identification Module for the attack identification and information collection. After that, these data are submitted to the mitigation module for countermeasure definition, generating a set of packet dropping policies. These policies are, then, sent to the SDN's central controller for mitigation implementation.

On the other hand, if no anomaly/attack is detected on the Detection Module, the network is considered to be operating normally. Then, the analyzed data is submitted to the Identification Module for the generation of the previously described "safe list," which is sent to the SDN's central controller to avoid future packet dropping of legitimate users.

IV. ANOMALY DETECTION METHODS

In this section, we detail the three anomaly detection methods used on the Detection Module of the presented system. The performance analysis of them is shown on Section V.

A. PARTICLE SWARM OPTIMIZATION FOR DIGITAL SIGNATURE

Several pieces of research were developed through the application of Swarm Intelligence (SI) to propose methods aiming to solve complex optimization problems, which are of difficult solution through classic optimization algorithms. Swarm Intelligence are nature-inspired metaheuristics based on the collective behavior of natural agents relating to their iteration mechanisms and environmental organization. The PSO metaheuristic was first introduced by Eberhart and Kennedy [34]. It was inspired by the social behavior of a flock of birds or shoal of fishes, introducing a new approach on functions optimization. On PSO method a flock of birds is randomly initialized into a search space in which each bird is referred as a particle and the set of particles is called "swarm."

PSO aims to optimize a specific function, known as fitness. At each iteration this function is used to measure the efficiency of the generated solutions, *i.e.*, the fitness is used to guide the particles movement towards the problem's solution. Thus, the fitness function measure how close the particles are from the solution (solution's performance), where each particle has an update speed that guides its movement along the search space. Consider a particle population P, where v_p and p_p represents the speed and the position of the particle p. The movement of each particle is performed by updating its movement speed and position through the Eq. (2) and (3), respectively.

$$v_{p+1} = wv_p + c_1 r_1 \left(p_{best_p} - x_p\right) + c_2 r_2 \left(g_{best} - x_p\right) \quad (2)$$

$$x_{p+1} = x_p + v_p \quad (3)$$

where w is the inertia coefficient, c_1 and c_2 are the acceleration constants, r_1 and r_2 are random numbers defined by the interval [0, 1], p_{best_p} is the best position occupied by the particle p until the given iteration and g_{best} represents the global solution at the given iteration by the swarm. According to [34] c_1 and c_2 can receive the value 2.05 and w equals to 0.5.

The particles p_{best_p} and g_{best} are evaluated each iteration through the fitness function. They are updated only in case the current solution presents a better outcome than the values already found until that iteration [35]. The update of the particles p_{best_p} and g_{best} are performed using the Eq. (4) and Eq. (5), respectively.

$$p_{best_p} = p'_{best_p} \quad if\ f(p'_{best_p}) < f(p_{best_p}) \quad (4)$$

$$g_{best} = g'_{best} \quad if\ f(g'_{best}) < f(g_{best}) \quad (5)$$

In several cases the system's convergence can be quickly achieved. The fast convergence makes this method an efficient optimization mechanism. As the approach used in this paper is based on an online multidimensional traffic characterization, this fast convergence is an essential factor. PSO was developed to be a simple method, implemented with few lines of code. It only requires primitive mathematical operations, also representing a low computational cost algorithm able to find optimal regions in multidimensional search spaces.

1) CHARACTERIZATION AND DETECTION MODULE

The anomaly detection method using PSO presented in this paper is divided into three steps:

1) The first step is the online traffic characterization, using the PSO for flow data optimization;
2) The second step is the anomaly detection using Chebyshev's inequality;

3) Finally, the mitigation module is activated when the an anomaly is detected on the previous step.

The traffic characterization is generated using IP flow data collected from the SDN controller, using both quantitative (bits/s and packets/s) and qualitative dimensions (source and destination IP addresses and ports).

The SDN traffic characterization was performed through the organization of the data into clusters. To optimize the clustering, the PSO method was used to find the centroid that best represents this flow set. The traffic characterization is performed each 5 seconds and uses a time window of n past minute to obtain the signature of the next second. This signature is here called as Digital Signature of Network Segment using flow analysis (DSNSF), and represents the networks normal operation behavior. The time window used in this paper is $n = 5$ (minutes). Each DSNSF point is achieved through the mean of the centroids of the C clusters obtained after the PSO optimization process. Algorithm 1 shows the process of DSNSF generation using PSO.

Algorithm 1 - PSO Used to Generate DSNSF

```
Require: Set of network information extracted from network flows
Ensure: Arrays representing the DSNSF with 17280 samples

for $i = 1 : 17280$ do
    Calculate inferior limit
    Calculate superior limit
    Generate population for time interval
    while a termination criterion is not met do
        Update $pBest$ (4)
        Update $gBest$ (5)
        Update the particle's velocity (2)
        Update the particle's position (3)
        Evaluate population fitness
    $DSNSF_i \leftarrow$ average among the centroids
return $DSNSF$
```

The fitness function applied on the optimization process was the Euclidean distance between IP flow data and the centroids, represented by the equation:

$$J = \sum_{i=1}^{E} \sum_{j=1}^{C} \sqrt{\sum_{a=1}^{A} (c_{ja} - x_{ia})^2} \tag{6}$$

in which E is the amount of flows to be clustered, C represents the number of clusters (for this method, we used the value $C = 2$) and A represents the amount of flow attributes or dimensions. As previously discussed, in this paper we use a six-dimensional analysis. The variable c_{ja} indicates the value of the cluster j belonging to the $a-th$ dimension and x_{ia} is de value of the feature a relating to the element i. The anomaly detection approach of this method is based on the Bienaymé-Chebyshev's inequality. This inequality is used to find behaviors that differs from the generated signature for the quantitative and qualitative dimensions. The Bienaymé-Chebyshev inequality determines a limiar of the data percentage that exists inside the $\pm k\times$ standard deviations interval around the mean. The inequality can be applied for outliers detection [36] when the data distribution is unknown.

The equation that describes Bienaymé-Chebyshev's inequality is:

$$P(\mid X - \mu \mid \geq k\sigma) \leq \frac{1}{k^2} \tag{7}$$

where X is a random variable, μ is the mean, $k > 0$ is the deviation parameter and σ is the standard deviation. If we set the parameter $k = 4.47$ on Eq. (7), the resultant probability will be equal to 0.05, which is the usual cut-off point for statistical significance [37]. In case that a sample is higher than k standard deviations relating to the average, this point is considered anomalous.

On the construction of the anomaly detection module for this method, an adaptation was performed on the Bienaymé-Chebyshev's inequality to create an upper and a lower threshold to determine what is considered a normal traffic behavior based on the generated DSNSF. Eq. (8) and Eq. (9) are used to determine the upper and lower limits, respectively. A dimension is detected as being anomalous when the actual traffic is higher than the $UPPER$ threshold or lower than $LOWER$ limit.

$$UPPER = DSNSF + k\sigma \tag{8}$$

$$LOWER = DSNSF - k\sigma \tag{9}$$

After the individual detection of each one of the flow features, it is necessary to define when in fact a general anomaly occurred. According to [10], [11], and [38], each type of anomaly affects the traffic flows in different ways. For instance, in a Denial of Service (DoS) attack a high concentration (low entropy values) on the features "source IP addresses" and "destination ports," while on a Flesh Crowd event occurs a higher dispersion (high entropy values) of the features "source IP addresses" and "source ports." We summarize the types of anomalies and flow attributes affected by them on Tab. 1.

TABLE 1. Type of anomalies and affected attributes.

Type of anomaly	Affected attributes
DoS	Source IP, destination IP, destination port and quantitative attributes
DDoS	Source IP, destination IP, source port, destination port, and quantitative attributes
Port Scan	Source IP, destination IP, destination port and quantitative attributes

Based on the behavior of the flow features when an anomaly occurs, we consider that at least three dimensions are detected as anomalous to activate the mitigation module, *i.e.*, if in a given time interval any three traffic features was detected as anomalous, then an alarm is triggered, and it is considered that a global anomaly occurred on this time interval.

B. MULTI-LAYER PERCEPTRON FOR DIGITAL SIGNATURE (MLP-DS)

MLP for Digital Signature (MLP-DS) is the term we designate the entire process of traffic characterization and anomaly detection using MLP network. The MLP is a neural network based on Perceptron networks operation which has at least one hidden neuron layer on its topology. According to Haykin [39], they are characterized by its wide application range within different areas, such as universal function approximator, pattern recognition, control and process identification, time series forecasting and systems' optimization.

One of the main characteristics that distinguish MLP from traditional Perceptron networks is the presence of hidden neuron layers. According to Haykin [39] the hidden neurons act as characteristics detectors, performing a key role in the network operation.

In brief, MLP are neural networks of supervised learning operating without feedback. The data are input separately through the "input layer." These data tend to be normalized to optimize the learning process of the network. Furthermore, the MLP topology is composed of at least one neuron's hidden layer and by an output layer, which will present the results of the network's classification. The topology is represented as a fully connected graph, *i.e.*, each input signal is connected to each one of the intermediary layer neurons. In turn, these neurons may be connected to each one of the neurons of a second hidden layer (if applicable) or each one of the output signals. A topology example is depicted in Fig 3.

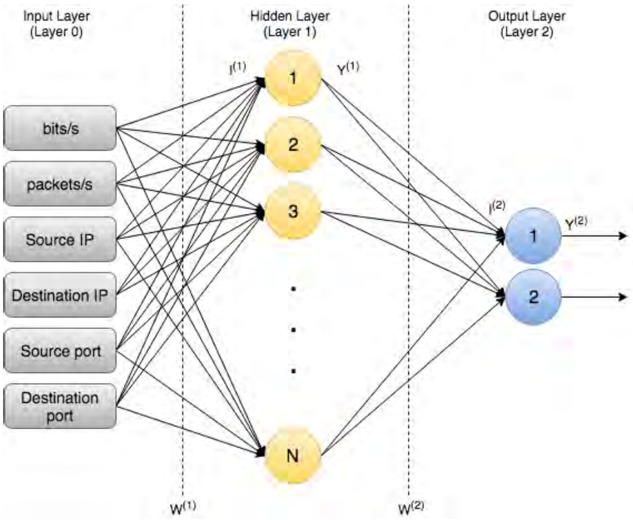

FIGURE 3. MLP topology with one hidden layer with N neurons and binary output.

Furthermore, each one of the connections is initialized with a random value from 0 to 1, representing the synaptic weights of each connection. These weights are adjusted during the learning process in order to allow the group classification, as previously described.

In this paper, MLP is used for SDN traffic characterization and anomaly detection processes on the analyzed network segment. It is important to highlight that this process is entirely performed in an autonomic way, without any network administrator's interference in the processes described herein.

As previously discussed on Section III, after the exportation of the collected flows into files, data relating to the analyzed IP flow dimensions are extracted in separate files so that they can be subjected to a traffic characterization process. This process is performed by using the Multi-Layer Perceptron (MLP) method, representing the training step.

Six flow dimensions are applied to the MLP method, responsible for learning the pattern behavior (DSNSF) of the SDN's normal and abnormal operation. The MLP-DS approach is able to detect not only DDoS attacks but also DoS and port scans. Thus, to enable the classification of four different states, the MLP was designed with two neurons. The outputs of the neurons are coded following Tab. 2. Fig. 3 shows the MPL topology used in this paper.

TABLE 2. Output Encoding for MLP-DS.

Detected behavior	Neuron $Y_1^{(2)}$	Neuron $Y_2^{(2)}$
Normal	0	0
DoS	0	1
DDoS	1	0
Port scan	1	1

The learning process of the MLP consists in submitting the neural network to a set of labeled data, *i.e.*, data of the six analyzed IP flow dimensions in addition to a label that describes whether this combination represents a normal traffic, a DDoS or a port scan behavior. However, even normal traffic data present different behaviors along a day of analysis. This occurs due to the fact that, for instance, the traffic early in the morning is different from the traffic on working hours, which does not make them anomalous. Similarly, different intensities of the same attack may generate different signatures for the same type of attack. In this manner, to improve the classification results of the MLP, the learning process is submitted through a clustering approach.

The first step is to separate the anomalous from the normal traffic training (labeled) data. Then, both groups are submitted to a clustering process using the K-means algorithm [40] in order to identify similar groups within the analyzed data. After this step, a sampling is performed within the different clusters in order to generate a reduced group that is able to represent the entire analyzed data. This sampling is important to the MLP learning to avoid a problem known as over-fitting, which impairs the classification results when the training dataset is too large. Finally, the sampled data from normal and anomalous traffic are united into a single training group, which is submitted as input for the MLP training method. The basic operation of the training process can be described in Algorithm 2.

Algorithm 2 - MLP Training Phase

```
1: Receive the input training (sample) data set
2: Associate the desired output (label) to all training data
3: Initialize the synaptic weights with random small values
4: Specify the learning rate $\eta$
5: Specify the required precision $\epsilon$

6: while $|MSE_{current} - MSE_{previous}| > \epsilon$ do
7:     $MSE_{previous} \leftarrow MSE$
8:     for each sample data do
9:         Calculate $I_j^{(1)}$ and $Y_j^{(1)}$; (Eq. (10) and (12))
10:        Calculate $I_j^{(2)}$ and $Y_j^{(2)}$; (Eq. (11) and (13))
11:        Calculate $\delta_j^{(2)}$; (Eq. (16))
12:        Adjust synaptic weights $W^{(2)}$; (Eq. (18))
13:        Calculate $\delta_j^{(1)}$; (Eq. (15))
14:        Adjust synaptic weights $W^{(1)}$; (Eq. (17))
15:    Calculate $Y_j^{(2)}$ through steps 9 and 10;
16:    Calculate MSE; (Eq. (19))
17:    $MSE_{current} \leftarrow MSE$;

   return Trained synaptic weights
```

In this algorithm, I_j^k stands for the weighted sum performed by the neuron j at the layer k (Eq. (10) and (11)):

$$I_j^{(1)} = \sum_{i=0}^{D} W_{ji}^{(1)} \cdot x_i \tag{10}$$

$$I_j^{(2)} = \sum_{i=0}^{N} W_{ji}^{(2)} \cdot Y_i^{(1)} \tag{11}$$

where D is the number of analyzed flow dimensions, $W_{ji}^{(1)}$ are the synaptic weights that connects neuron j to neuron i of the following neural layer, and x_i is the i-est neuron at the input layer. Y_j^k stands for the calculated output of the Perceptron j at the layer k (Eq. (12) and (13)):

$$Y_j^{(1)} = g(I_j^{(1)}) \tag{12}$$

$$Y_j^{(2)} = g(I_j^{(2)}) \tag{13}$$

where $g(\cdot)$ is an activation function. In this paper, we adopt the logistic activation function with inclination parameter β of 1, defined by Eq. (14).

$$g(u) = \frac{1}{1 + e^{\beta \cdot u}} \tag{14}$$

The variable δ_j^k is the local gradient of the neuron j at the layer k (Eq. (15) and (16)):

$$\delta_j^{(2)} = (d_j - Y_j^{(2)}) \cdot g'(I_j^{(2)}) \tag{15}$$

$$\delta_j^{(1)} = \left(\sum_{i=0}^{N} \delta_i^{(2)} \cdot W_{ij}^{(2)} \right) \cdot g'(I_j^{(1)}) \tag{16}$$

where the variable d_j represents the expected output for neuron j, $Y_j^{(2)}$ represents the output calculated for neuron j by the MLP and $g'(.)$ is the derivative of Eq. (14) regarding I_j.

The synaptic weights W_{ji} connecting neurons j to i of each layer are updated through the Eq. (17) and (18):

$$W_{ji}^{(1)} \leftarrow W_{ji}^{(1)} + \eta \cdot \delta_j^{(1)} \cdot x_i \tag{17}$$

$$W_{ji}^{(2)} \leftarrow W_{ji}^{(2)} + \eta \cdot \delta_j^{(2)} \cdot Y_i^{(1)} \tag{18}$$

where η is the learning rate, herein defined as 0.2 after exhaustive performance testing, and x_i is the i-th neuron at the input layer. The value η directly influences the convergence outcomes. When this value is high, the convergence can be quickly achieved. However, in this case, it is possible for the method to be unable to find the convergence due to the learning rate step. Lower η values mean a slower and more secure convergence process.

The Mean Square Error (MSE) between the desired (step 2 on Algorithm 2) and the achieved output is performed computing:

$$MSE = \frac{1}{p} \sum_{k=1}^{p} \varepsilon(k) \tag{19}$$

where p is the number of data samples analyzed and $\varepsilon(\cdot)$ is the square error, achieved through Eq. (20).

$$\varepsilon(k) = \frac{1}{2} \sum_{j=1}^{N2} (d_j(k) - Y_j^{(2)}(k))^2 \tag{20}$$

In this Equation, $N2$ is the number of neurons at the output layer. Finally, the variable ϵ (line 6 of Algorithm 2) represents the required precision of the results, herein defined as 10^{-7}.

After the training process, the calculated synaptic weights can be imported into the SDN controller, which will execute the classification process using the steps described by Algorithm 3. The computational cost of the MLP is high only in the training process, which only needs to be performed once. After this, a lightweight process of classification is performed every 5 seconds by the SDN controller and, if a DDoS or a port scan attack is detected, an alarm is triggered to invoke the Identification and the Mitigation modules, as described in Section III.

Algorithm 3 - MLP Operation Phase

```
Receive the input sample to classify;
Import the synaptic weights calculated with Algorithm 2;
Calculate $I_j^{(1)}$ and $Y_j^{(1)}$; (Eq. (10) and 12)
Calculate $I_j^{(2)}$ and $Y_j^{(2)}$; (Eq. (11) and 13)
return Classification provided by $Y_j^{(2)}$;
```

C. DISCRETE WAVELET TRANSFORM (WAVEDETECT)

The proposed anomaly detection called WaveDetect uses Discrete Wavelet Transform (DWT) [41] for anomaly detection. DWT consists of a mechanism to decompose or break signals (data) into their constituent spectral parts, *i.e.*, into different frequencies components [42].

These constituent parts are called coefficients, and the different frequencies are obtained throughout DWT decomposition levels. These coefficients can be of two types: approximation (scaling), or detail (wavelet). The approximation coefficients ($c_{j,k}$) are responsible for the coarsest information of the input signal, which consists of the lower frequencies. The detail coefficients, represented by $d_{j,k}$, carry the high frequencies of the previous level data.

These coefficients are obtained by a filter bank application, which consists of a matrix multiplication of high-pass (h) and low-pass (g) filters by the input data. By high-pass filters we obtain detail coefficients ($d_{j,k}$), and by low-pass filters, the approximation coefficients ($c_{j,k}$) at DWT decomposition level j with k elements, where $k = N/2$ and N being the input data size. The filter is defined by the wavelet function used [43]. In the proposed solution, the wavelet function chosen was Haar, because this kind of wavelet is computationally tractable and provides a low computational cost [44], [45]. DWT decomposition is depicted in Fig. 4.

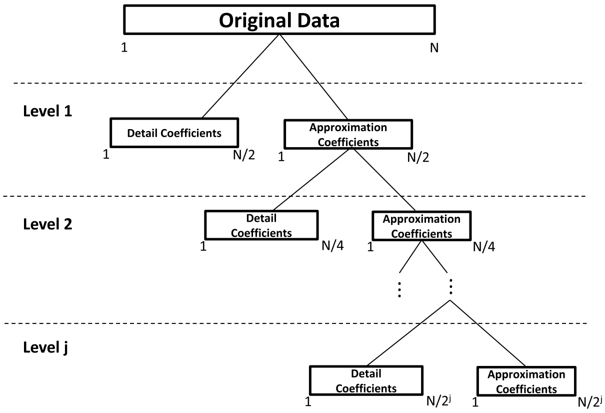

FIGURE 4. DWT decomposition process.

DWT can also be represented by Eq. (21), where $\varphi_{j_0,k}(t)$ is the scale function, also known as father wavelet, generating approximation coefficients $c_{j_0,k}$ from level j_0 and $\psi_{j,k}(t)$, which is the wavelet function, or mother wavelet, defining the filters from the DWT, and generating detail coefficients $d_{j,k}$ for all DWT decomposition levels; both wavelet coefficients are composed by k elements, where k depends of input data size.

$$f(t) = \sum_{k=1}^{\frac{N}{2}} c_{j_0,k}(k)\,\varphi_{j_0,k}(t) + \sum_{j=j_0}^{\infty}\sum_{k=1}^{\frac{N}{2}} d_{j,k}\psi_{j,k}(t) \qquad (21)$$

The proposed WaveDetect method is divided in two stages, the first is the Traffic Characterization level and the second is the Anomaly Detection level.

The WaveDetect method is basically a comparison between two sliding windows, the first (W_f) representing the traffic forecast, and the second (W_d) carrying the traffic with the sample that will be analyzed. Both windows and levels are going to be explained following.

1) FIRST LEVEL: TRAFFIC CHARACTERIZATION

It is responsible for sliding W_f and W_d. The traffic forecast window W_f will only slide to incorporate the forwarding traffic sample if this sample was previously classified as a normal point; if so, the oldest W_f sample is replaced by the new sample, *i.e.*, W_f will always carry the last M samples with normal traffic.

The second window, W_d has the same size of W_f and contains the last $M - 1$ points from W_f and its last point is the sample of interest, that is, the sample that is going to be analyzed. This window slides excluding the oldest point and including the following point. Fig. 5 depicts a visual explanation. Also, a mathematical explanation of both windows is depicted in Eq. (22)–(24), for a sample t.

$$|W_f| = |W_d| = M \qquad (22)$$

$$W_f = \{w_f(t-1)-M, w_f(t-M), \ldots, w_f(t-1)\} \qquad (23)$$

$$W_d = \{w_d\,(\text{t-M}), w_d(t-M)+1, \ldots, t\} \qquad (24)$$

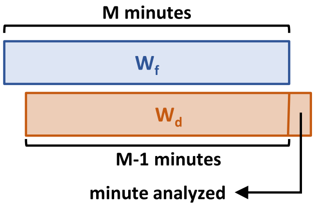

FIGURE 5. Sliding windows W_f and W_d.

As the solution uses previous data to generate W_f and W_d, when starting the Traffic Characterization level, it is required a database with the last M minutes considered within the pattern, for a bias-free forecast. As explained previously, W_f and W_d have size M. The value of M ranges between 16 and 8192. The proposal of using different sizes for M was to find the amount of historical traffic which best describes the network traffic. Also, all values of M range in values multiples of a power of 2, to facilitate the DWT decomposition process, as this process is similar to binary tree division. After tests that will be further explained in Results and Analysis section, the value chosen for M was 1024.

2) SECOND LEVEL: ANOMALY DETECTION

This level aims for the detection of DDoS and port scan attacks. This level is divided into two main stages: DWT and then DDoS couplet with port scan detection. The first stage performs a one-dimensional DWT in W_f and W_d, with one decomposition level. The decision of use one decomposition level was made based on tests explained in Results and Analysis section.

The second stage performs anomaly detection for each dimension. This process is divided in three steps. The first step calculates an interquartile range (IQR), obtained through the approximation coefficients calculated using W_f. *IQR* is detailed in (25) and (26) and, according to Hoaglin [46], this approach was proposed by John Tukey based on normal patterns of diastolic blood pressures.

$$IQR = Q3 - Q1 \tag{25}$$

$$(Q1 - (IQR \cdot 1.5)) \leq X \leq (Q3 + (IQR \cdot 1.5)) \tag{26}$$

where X is the data analyzed, $Q1$ is the first quartile and $Q3$ is the third quartile.

After this, the second step of DDoS and port scan detection compares the last value of approximation coefficients ($c_{1,N/2}$) from W_d with the interval obtained in Eq. (25). If it is within the interval, the traffic is classified as normal traffic, otherwise, the traffic is classified as anomalous for an specific dimension. Also, if the analyzed traffic is classified as anomalous, WaveDetect evaluate if the traffic is higher or lower than the normal behavior. It will help on a later anomaly classification.

The third and last step of detection accomplishes classification of anomalous traffic in DDoS or port scan. When a DDoS or a port scan attack is being performed, some changes occur in specific IP flow traffic dimensions (features). A DDoS attack increases source port entropy, decreases destination IP and port entropies and discreetly decreases the source IP entropy. A port scan attack modifies basically three dimensions, increasing destination port entropy, and decreasing source and destination IP entropies. By comparing this information with the anomaly detection output, it is possible to identify which anomaly is on the traffic sample. Algorithm 4 details the stages from Anomaly Detection level.

Algorithm 4 - WaveDetect Detection Phase

```
Calculate $f_{W_f}(t)$ and $f_{W_d}(t)$

Calculate $Q1$ and $Q3$ of $c_{(j,k)}$ from $W_f$
if $((c_1, \frac{N}{2})$ from $W_d$ is greater than $(Q1 - \text{IQR} \cdot 1.5)$ and smaller than $(Q3 + \text{IQR} \cdot 1.5))$ then
    Classifies traffic as Normal
else
    Classifies traffic as Anomalous
Identifies DDoS or port scan pattern
```

V. RESULTS AND ANALISYS

In this section we discuss the achieved results of performance tests of the presented SDN defense system. This analysis is performed in two sub-sections. In the first one, we discuss the parameters used by the methods described in Section IV. In the second one, we show the performance outcomes achieved by the presented system on anomaly detection through the comparison between the three presented detection methods and classic anomaly detection methods in literature. Furthermore, the mitigation performance over the port scan and DDoS attacks is discussed.

A. PARAMETERS ESTIMATION

The first step performed by any swarm based optimization algorithm, like PSO-DS, is the estimation of the individuals population, *i.e.*, the possible problems' solution. Knowing that the population of individuals is randomly generated, the population size must be chosen with caution. Through empirical tests applied by Bratton and Kennedy [47], the authors define the number of individuals comprises between 20 and 100 particles. To evaluate the amount of particles, we applied the Normalized Mean Square Error (NMSE) [48] for a day of PSO-DS traffic characterization. The NMSE calculates the absolute difference between the generated DSNSF and the actual SDN traffic. This metric generates outcomes between zero to infinite, where values close to zero indicate a good traffic characterization, while higher values indicate that the forecasting performed by the characterization process diverges from the actual traffic behavior. According to Fig. 6, the number of particles with lower NMSE outcome, *i.e.*, that generates the best traffic characterization for PSO-DS, is 50 particles.

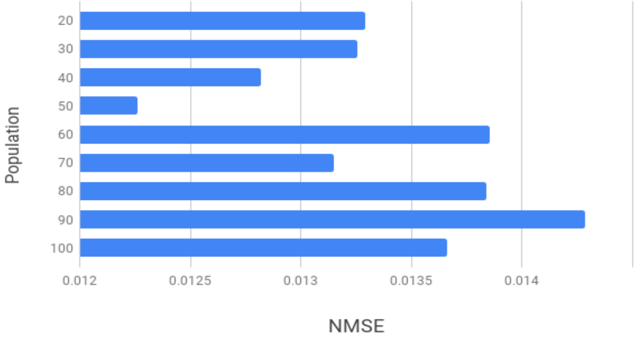

FIGURE 6. Estimation of population size for PSO-DS.

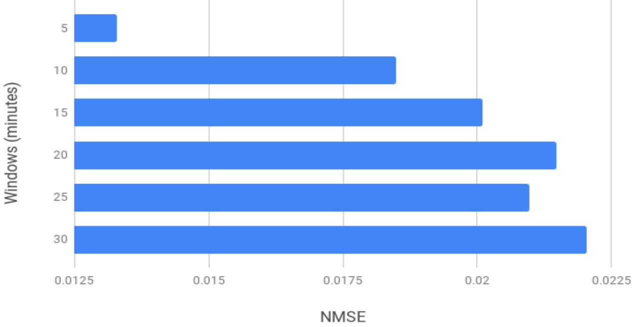

FIGURE 7. Estimation of time window size for PSO-DS.

For the generation of network traffic signatures, the PSO-DS uses data from the last n past minutes of the real traffic. The convergence evaluation of the used time window was performed through the NMSE. The values tested for n comprise between 5 and 30 minutes. As observed in Fig. 7, the time window which achieved best results was $n = 5$ minutes.

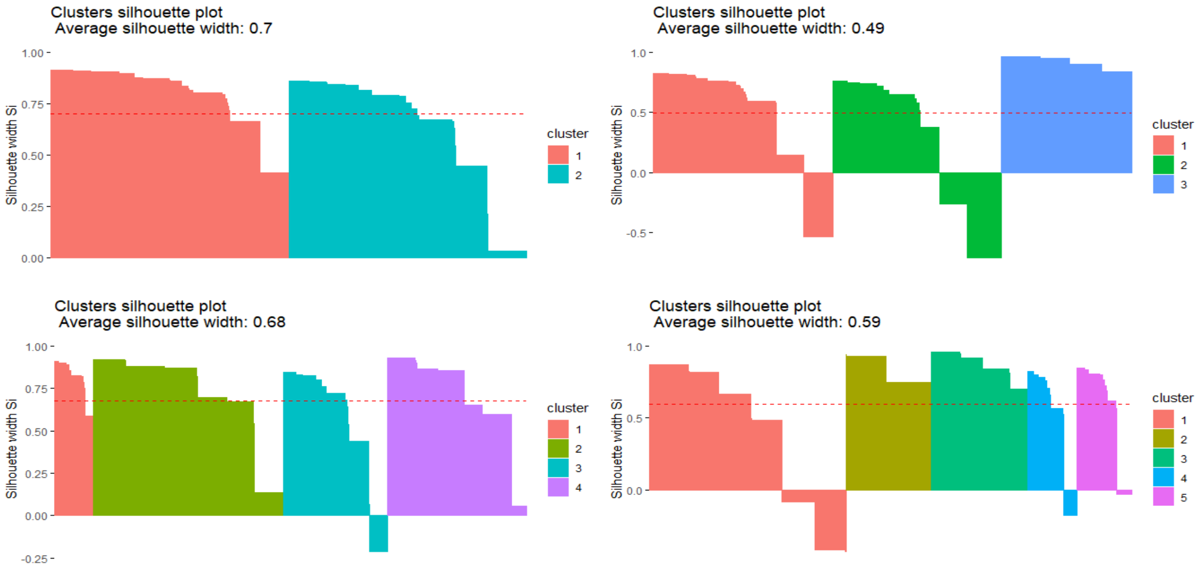

FIGURE 8. **Silhouette technique used for estimation of the amount of clusters.**

PSO-DS is based on the traffic characterization through the clusterization of flows extracted form the SDN controller. Thus, the Silhouette technique [49] was used to estimate the number of clusters used by PSO-DS. The application of this technique provides a graphical representation of the elements' arrangement inside each cluster. This graphical representation is useful when the proximity metric is in scale (like in the case of Euclidean distance) and when compact and clearly separated clusters are required. The outcome of this function comprise between $-1 \leq f(s) \leq 1$, and have three interpretations for the results. When $f(s)$ is close to -1, it means that the sample i was misclassified, *i.e.*, the element should be assigned to another cluster. When the value of $f(s)$ tends to zero, it is an intermediate case, which means that the element i could be assigned to more than one cluster. The best case is when $f(s)$ is close to 1, which indicates the existence of a high similarity between the element i and the other elements belonging to the cluster. Fig. 8 presents the values of C (number of clusters) varying from 2 to 5. From the analysis of C, it is possible to note that the best achieved outcome was achieved using 2 clusters. According to Fig. 8, the function did not obtain zero or negative outcomes, *i.e.*, no element was assigned to the cluster erroneously.

For the MLP-DS method, some of the variables are given by the stated problem. The number of input neurons is defined as 6 since there is six different analyzed flow dimensions used on traffic characterization and anomaly detection problem. The number of neurons on the output layer is defined as 2 due to the codification used on the classification process, as described by Tab. 2. As discussed in [27], the number of neurons on the hidden layer influences the MLP outcomes up to $2N + 1$ neurons, where N is the number of neurons on the input layer. This can be observed in Fig. 10.

As observed, using $2N$, $2N + 1$ or $2N + 2$ neurons on the hidden layer does not further improve the MLP classification outcomes. Thus, the number of neurons on the hidden layer was defined as 12. Another parameter used on the MLP-DS method is the number of clusters used on the training process. Fig. 11 shows the results relating to the estimation of this parameter.

As shown, there is an improvement on the classification efficiency of the MLP when using the described clustering approach on the analyzed data in comparison with the case where no clustering is used (when the number of clusters is 1). From 2 to 5 clusters, there was no considerable efficiency difference on the classification outcomes. Thus, the number of clusters was defined as 2. Fig. 12 shows as example a radar-plot of the network's normal behavior before and after the clustering process.

To evaluate the window size and the DWT decomposition level that provides the best detection, some tests were performed. For this, ten different window sizes (M) were defined, which are: $M = 16, 32, 64, 128, 256, 512, 1024, 2048, 4096$ or 8192 samples. For each M, tests with different DWT decomposition levels, from one to four levels were performed as well. To assess all tests, six metrics (Precision, Accuracy, False Positive rate[FP-rate], Recall, F-measure and Area Under the ROC Curve [AUC]) were applied. Fig. 9 and 13 present the best result for each value of M, *i.e.*, the level which provided the best result for each M. By analyzing both figures it is possible to conclude that using DWT with small windows (from 16 to 64 historical traffic's samples) presented bad detection results and high false-positive rates. Values of M greater than 64 and smaller than 4096 provided better results than using smaller windows, but its best result was achieved using $M = 1024$, which provided a detection rate of $99,32\%$, false-positive rate of $0,03\%$ and AUC equal to $99,45\%$. Windows bigger than 4096 showed a decreasing on Detection rates, with an average detection of $93,58\%$. An analysis of decomposition

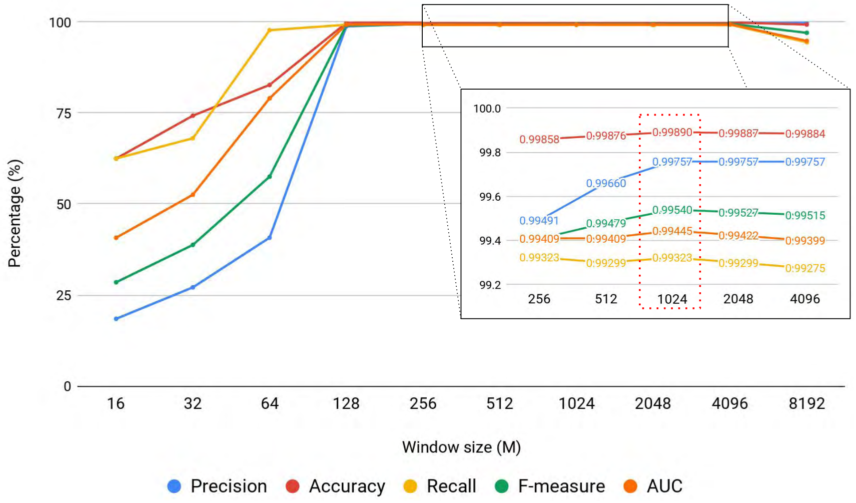

FIGURE 9. Results from five metrics to different window size and decomposition level of WaveDetect.

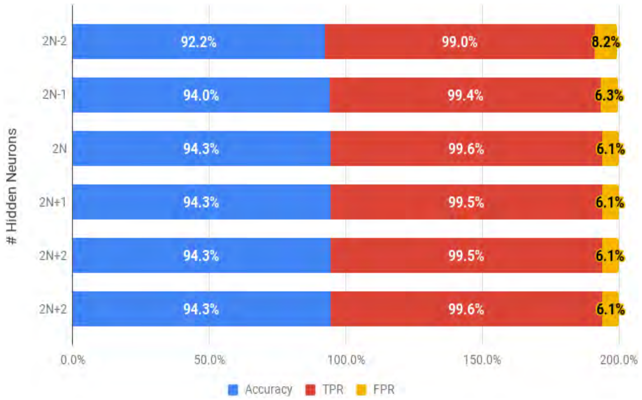

FIGURE 10. Estimation of required number of neurons on the hidden layer for MLP-DS.

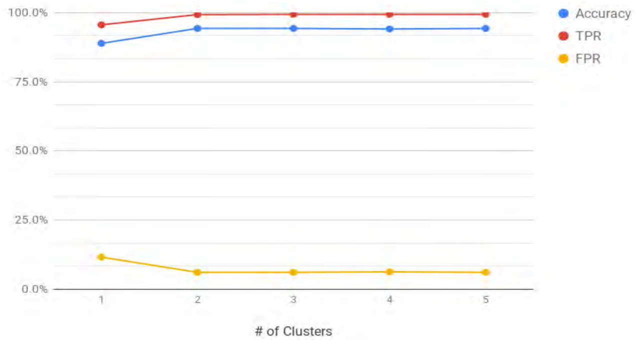

FIGURE 11. Estimation of required number of clusters for MLP-DS training process.

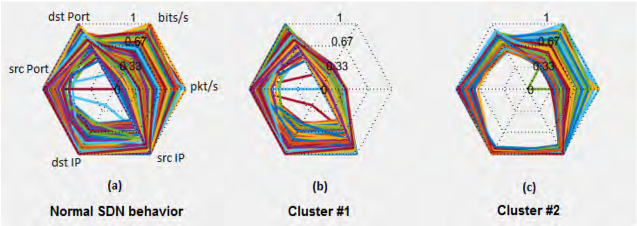

FIGURE 12. Radarplot describing a normal SDN behavior, showing the analyzed day without clustering (a) and the two clusters generated using k-means method (b and c). The lines represent the 6-dimensional view of the SDN behavior in each analyzed time interval.

levels were also made. Using one level regardless the window size, detection rates were better than using other levels, and false-positive rates were better using four decomposition levels. Considering all metrics and analyzing the use of each level, regardless the window size, one level of decomposition presented the best results. It can be explained by the fact that approximation coefficients carry the coarsest part of input data, so the deeper the level, the coarser will be the representation from original data. So by all these analysis, the value of M and the level chosen were 1024 and one, respectively.

B. PERFORMANCE OUTCOMES

Here we discuss the achieved outcomes of the performance tests. To perform these tests, we used simulated data generated by Mininet network emulator [50], a tool that allows the creation of realistic virtual networks composed by controllers, hosts, links, and switches in a single virtual machine. Mininet uses a lightweight virtualization on the creation of custom topologies through simple command lines. The experiments conducted in this paper used the Open vSwitch to control the network's switches, as Mininet offers support for it.

To implement the anomaly detection and mitigation mechanism, we used the SDN controller Floodlight, a Java-based controller widely used in literature. Finally, the data collected and managed are from the OpenFlow protocol. The SDN topology emulated is composed of four switches in a tree-like

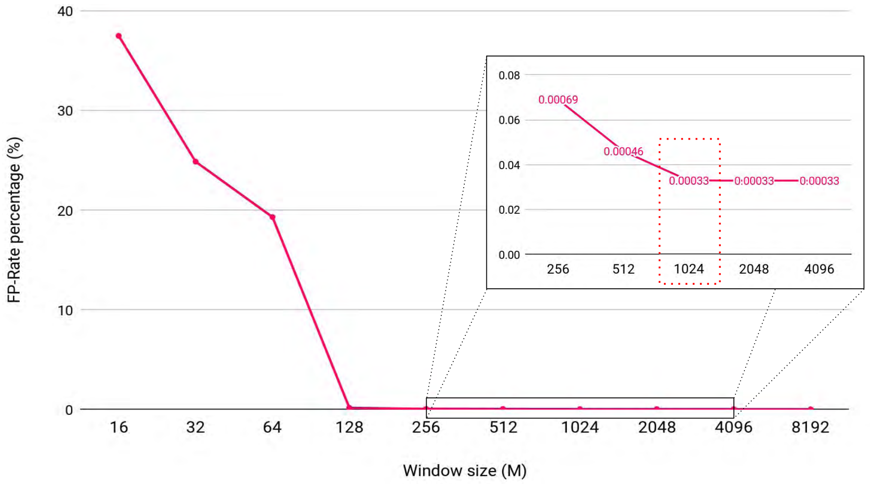

FIGURE 13. Results from FP-rate to different window size and decomposition level of WaveDetect.

TABLE 3. Description of test data.

	Attack 1	Attack 2	Attack 3
Day 2	Type: DDoS Attackers: 5 Attacking IPs: 10.0.0.20 - 10.0.0.24 Destination IP: 10.0.0.58 Duration: 07:00:00 - 07:59:40	Type: DDoS Attackers: 10 Attacking IPs: 10.0.0.21 - 10.0.0.30 Destination IP: 10.0.0.10 Duration: 12:00:00 - 13:02:20	Type: DDoS Attackers: 15 Attacking IPs: 10.0.0.21 - 10.0.0.35 Destination IP: 10.0.0.1 Duration: 15:00:00 - 15:59:05
Day 3	Type: Port Scan Attacking IP: 10.0.0.60 Destination IP: 10.0.0.30 Ports: 1 - 14961 Seconds to wait between 2 packets: 0.2 Duration: 06:00:00 - 06:59:40	Type: Port Scan Attacking IP: 10.0.0.41 Destination IP: 10.0.0.35 Ports: 1 - 19999 Seconds to wait between 2 packets: 0.1 Duration: 10:00:00 - 10:46:30	Type: Port Scan Attacking IP: 10.0.0.35 Destination IP: 10.0.0.1 Ports: 1 - 19126 Seconds to wait between 2 packets: 0.15 Duration: 16:00:00 - 16:56:55

topology, where one root switch connects the other three, each one connecting twenty different hosts. One of these switches represents a border gateway, and between its hosts are normal and malicious users. To guarantee that the emulated scenario is as close as possible to a real SDN environment, with high traffic rates passing through the network, in our experiments we used a tool named Scapy [51] to inject the emulated network with traffic. The data collection was performed using a REST API provided by Floodlight, which sends requests to a flow controller of a switch every five seconds.

Three days (72 hours) of SDN traffic were generated and used on our performance tests. Each one of the generated days emulates the normal behavior observed at the State University of Londrina (UEL), Brazil, where there is an increase in network usage in the morning (from 8:00 to 11:30) and in the afternoon (from 14:00 to 17:00). In the evening, the network usage is less intense, but similar to mornings.

The first generated day was injected with two occurrences of port scan and DDoS attacks of different intensities. This day was used for MLP-DS training process, since it is a supervised machine learning method. As PSO-DS and WaveDetect need no previous training, this data was not used by them.

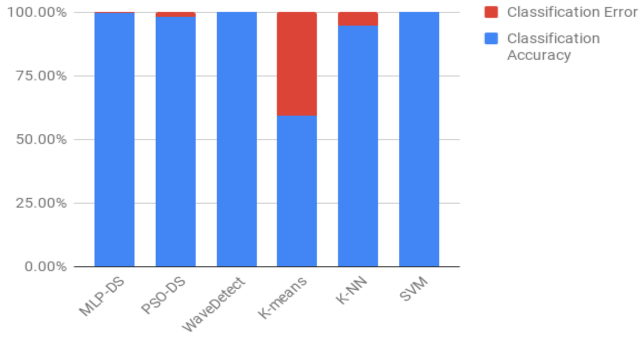

FIGURE 14. Classification Accuracy (CA) and Classification Error (CE) on anomaly detection for the analyzed methods.

The next two generated days were used on testing the efficiency of both anomaly detection and attack mitigation of the presented system. Three DDoS and port scan attacks of different intensities and in different time intervals were injected into SDN traffic, separating the DDoS attacks in one SDN traffic day and the port scan attacks on the another one. Tab. 3 describes the parameters of the performed attacks.

TABLE 4. Achieved results on DDoS and Port Scan detection of the tested methods.

	MLP-DS	PSO-DS	WaveDetect	K-means	K-NN	SVM
True Positive rate (TPR)	99.71%	99.66%	99.32%	64.80%	76.84%	99.37%
True Negative rate (TNR)	99.74%	97.85%	99.97%	58.62%	96.90%	99.98%
False Positive rate (FPR)	0.26%	2.15%	0.03%	41.38%	3.10%	0.02%
False Negative rate (FNR)	0.29%	0.34%	0.68%	35.20%	23.16%	0.63%
Positive Prediction Value (PPV)	98.12%	86.29%	99.76%	17.55%	77.14%	99.83%
Negative Prediction Value (NPV)	99.96%	99.95%	99.91%	92.45%	96.85%	99.91%
Classification Accuracy (CA)	99.74%	98.06%	99.89%	59.36%	94.50%	99.90%
Classification Error (CE)	0.26%	1.94%	0.11%	40.64%	5.50%	0.10%

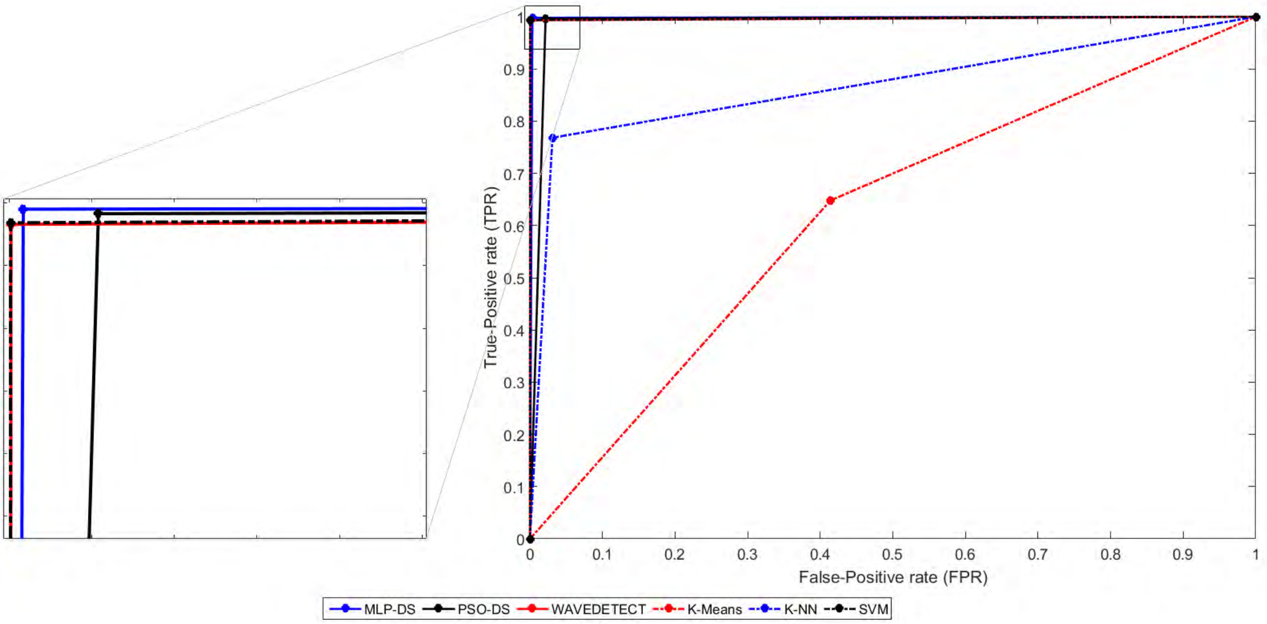

FIGURE 15. Roc Curve of the tested methods.

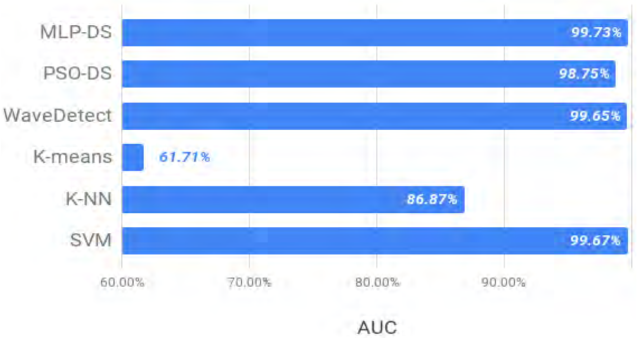

FIGURE 16. Area Under the Curve (AUC) of the generated ROC curves.

1) ANOMALY DETECTION

To test the efficiency of the presented methods on detecting port scans and DDoS attacks, we compare them with well stated anomaly detection algorithms, such as K-Means [40], K-Nearest Neighbors (K-NN) [52] and Support Vector Machine (SVM) [53]. The metrics used are classical anomaly detection statistic techniques [54], such as True and False Positive Rates (TPR and FPR), True and False Negative Rates (TNR and FNR), Positive and Negative Prediction Value (PPV and NPV), Classification Accuracy (CA) and Classification Error (CE). The results achieved by them are described on Tab. 4.

As observed, the presented methods fared better than K-means and K-NN classic approaches on most analysis scenarios. The SVM achieved performance is very similar to the one generated by WaveDetect method. The overall outcomes were good for MLP-DS, PSO-DS and SVM methods, with high TPR and TNR rates, and low FPR and FNR rates.

Fig. 14 shows the classification accuracy and classification error of the tested methods. As shown, the most inaccurate method on detecting anomalies was the K-means approach, with CA rate of nearly 60%, followed by K-NN method that, even though achieved an accuracy rate of 94.5%, misclassified around 23% of normal traffic intervals (FNR). MLP-DS, PSO-DS, WaveDetect and SVM methods achieved similar results, although PSO-DS faring slightly worse than the others due to its FPR rate of 2.15%.

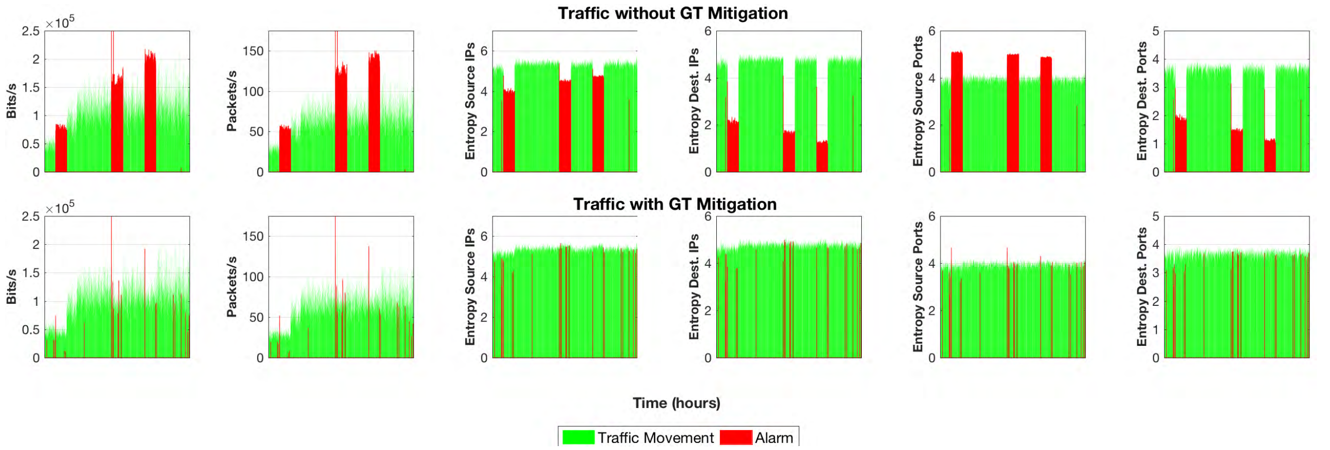

FIGURE 17. SDN six dimensional traffic movement, with three DDoS attacks, before and after the detection and mitigation processes using PSO-DS and GT-approach.

Furthermore, a Receiver Operating Characteristic (ROC) curve was constructed using the TPR and FPR rates for all tested methods to measure the classification efficiency of them. The ROC curve is shown in Fig. 15. As seen, the classification outcomes of MLP-DS, PSO-DS, WaveDetect and SVM are visually better than the ones achieved by K-Means and K-NN approaches. Their outcome was so similar that a zoom was needed in order to analyze their differences. As observed, PSO-DS method have the higher FPR rate, followed by MLP-DS, while WaveDetect and SVM achieved similar results for this metric. However, PSO-DS and MLP-DS achieved higher TPR rates than WaveDetect and SVM approaches.

To better quantify the efficiency of the tested methods, we analyze the area under the curve (AUC) of the ROC curve. The outcomes of this analysis is shown by Fig. 16. This figure shows that, relating to TPR and FPR rates, MLP-DS have the better trade-off, followed by SVM, WaveDetect, PSO-DS, K-NN and K-means.

It is noteworthy the differences between the analyzed methods. MLP-DS, K-NN and SVM methods needed to be trained before applied on the presented SDN defense system. In turn, PSO-DS, WaveDetect and K-Means need no previous data training for the anomaly detection. Furthermore, MLP-DS and K-NN methods are able to directly detect if the detected anomaly is a DDoS or a port scan attack, while the other methods only detects the anomaly occurrence. For those methods, the Identification Module of the presented SDN defense system is responsible for identifying the type of the anomaly in order to trigger the correct countermeasure on the Mitigation Module, as described in Section III.

Finally, to compare the computational efficiency and detection speed of the presented methods we compare their computational complexity. Although the computational complexity of MLP-DS method is $O(W^3)$ for the training step, where W is the number of synaptic weights of the neural network, for the operation step its complexity is asymptotically given by $O(W)$. As the number of synaptic weights is constant, the overall complexity of the operation step is $O(1)$, the lowest complexity among the presented methods. The WaveDetect is a Wavelet-based approach that carries out one DWT decomposition level. Thus, its computational complexity is $O\,(N)$, which is a linear complexity, where N is the size of the sparse Wavelet filters matrix. Finally, evolutionary algorithms, such as PSO-DS, have the complexity of $O(N{*}P{*}F)$ for each iteration, where N is the dimension of the problem, P is the population size, and F is the objective function size. Thus, we conclude that MLP-DS is the fastest of the presented anomaly detection approach, followed by WaveDetect (linear complexity) and PSO-DS faring worse, due to the need of a clustering process on each iteration.

2) MITIGATION PROCESS

In this section we discuss the outcomes achieved by the Mitigation Module of the presented SDN defense system against DDoS and port scan attacks. As previously discussed in Section III, the Mitigation Module receives data from the Identification Module, which is responsible of providing the system with relevant information about the detected anomaly, such as the most frequent source and destination IP addresses and ports.

When a DDoS attack is detected, either by the Identification module or by the anomaly detection method itself, the presented SDN defense system triggers a game theoretical (GT) approach to automatically defines the optimal drop rate to mitigate the attack while minimizing impact on legitimate users. Figures 17, 18 and 19 show the traffic on the six analyzed SDN dimensions from 6am to 7pm, before and after the DDoS GT mitigation approach.

As observed, the three anomaly detection methods tested presented similar results on detecting the DDoS attack, correctly triggering the alarms. However, after the mitigation process, PSO-DS generated a higher amount of false-positive alarms than MLP-DS and WaveDetect methods. As the

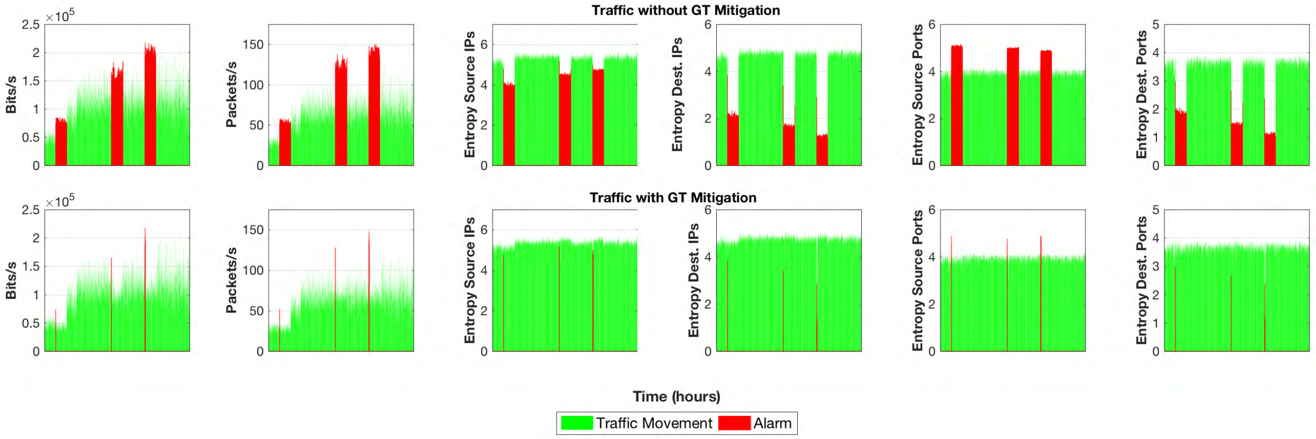

FIGURE 18. SDN six dimensional traffic movement, with three DDoS attacks, before and after the detection and mitigation processes using MLP-DS and GT-approach.

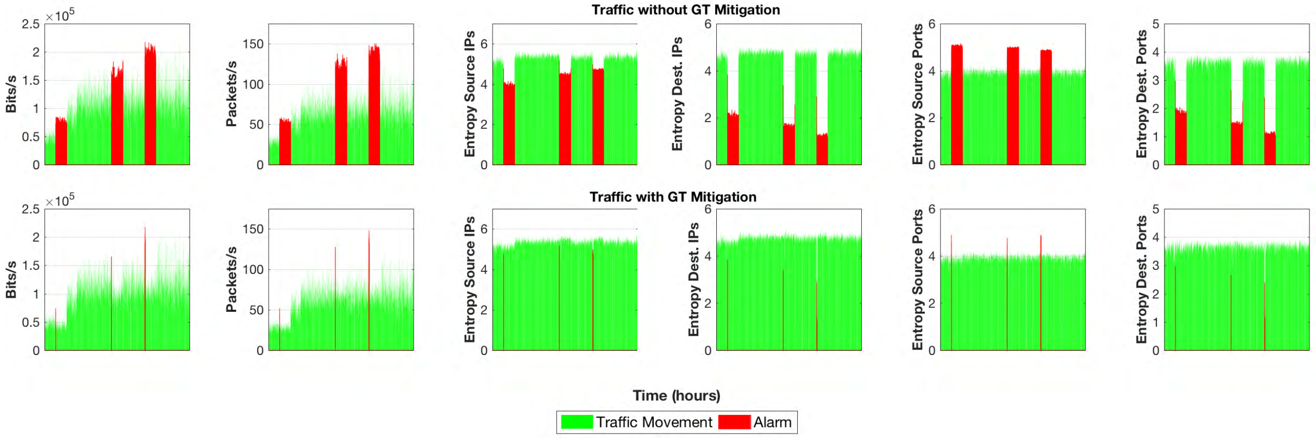

FIGURE 19. SDN six dimensional traffic movement, with three DDoS attacks, before and after the detection and mitigation processes using WaveDetect and GT-approach.

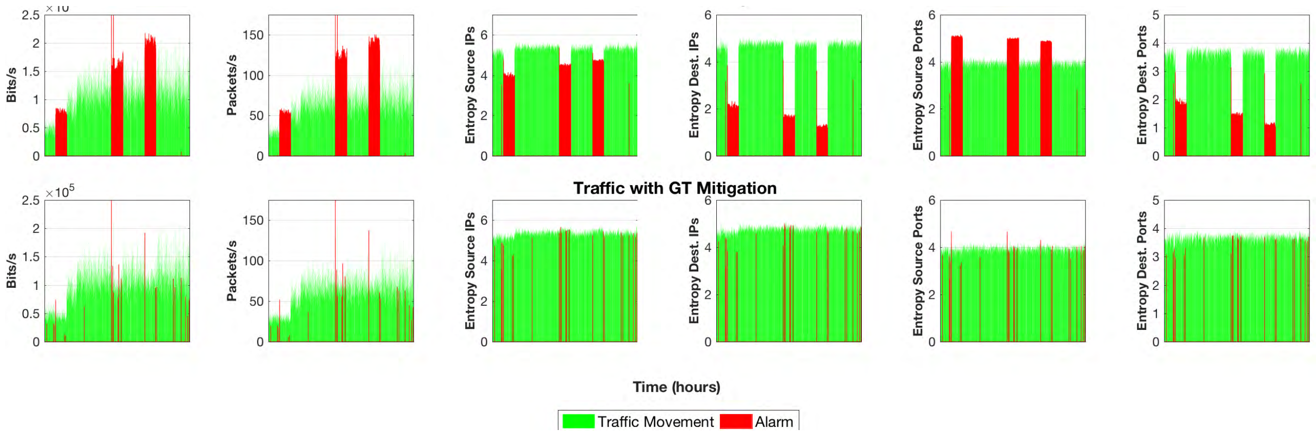

FIGURE 20. SDN six dimensional traffic movement, with three Port scan attacks, before and after the detection and mitigation processes using PSO-DS and directed drop policy.

mitigation succeeded on bringing the SDN back to a regular state, PSO-DS identified small traffic deviations as anomalous and, thus, generated the false-positive alarms. On the other hand, MLP-DS and WaveDetect achieved similar results, and the red lines present on the SDN traffic after the mitigation process are the time intervals where the attack was detected (if an attack is detected at 07:00:05, the mitigation starts only at the following time interval, *i.e.*, 07:00:10).

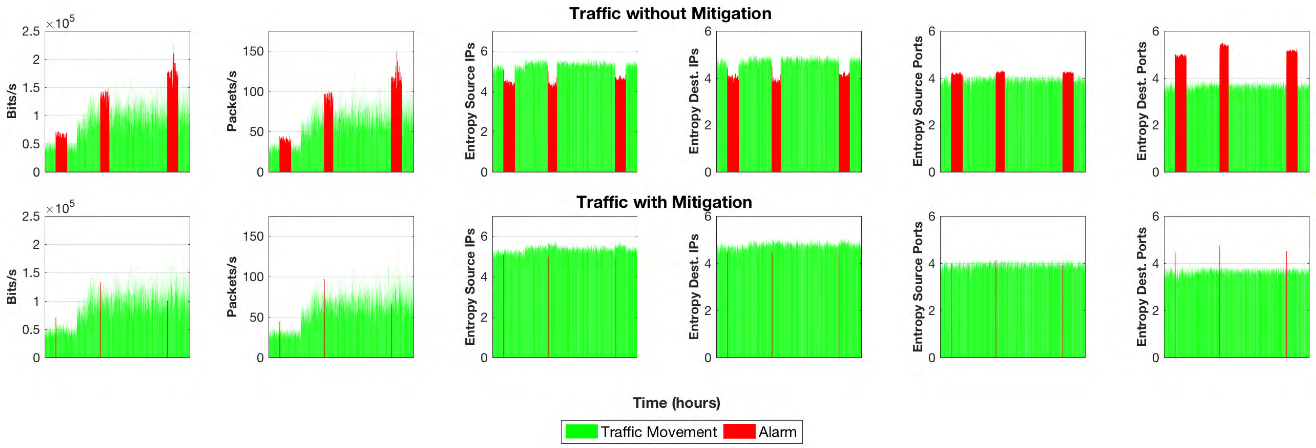

FIGURE 21. SDN six dimensional traffic movement, with three Port scan attacks, before and after the detection and mitigation processes using MLP-DS and directed drop policy.

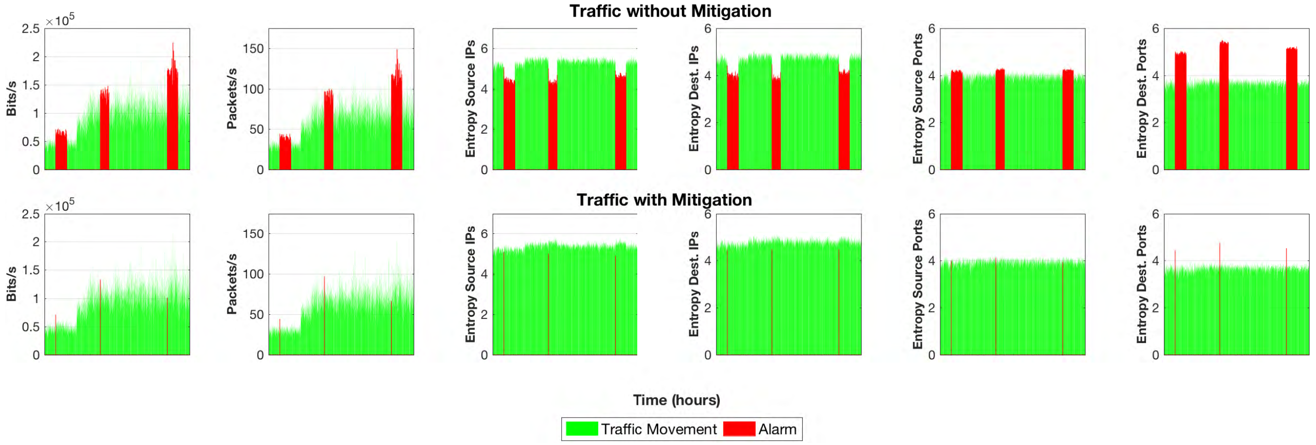

FIGURE 22. SDN six dimensional traffic movement, with three Port scan attacks, before and after the detection and mitigation processes using WaveDetect and directed drop policy.

Relating to port scan attacks, there is no need for the GT-approach on the mitigation process. This is due to the characteristics of this attack, since it is a centralized active scanning where a single host scans a range of ports of another. This generates a singular behavior, which is collected by the Identification module, enabling the isolation of the attacks' source IP address. With this feature, the Mitigation module sends a directed drop policy (drop of a single source IP address) to the SDN central controller. Figures 20, 21 and 22 show the traffic on the six analyzed SDN dimensions from 5:00:00 to 18:00:00, before and after the port scan's mitigation.

As observed, the directed drop policy was able to drop specifically the packets from the attacker, bringing the network to a normal state. As observed on the DDoS attacks, MLP-DS and WaveDetect methods achieved similar results, with just a few anomalous intervals observed in red lines (time interval when the port scan was detected). PSO-DS also achieved good mitigation outcomes for port scan attacks, although the false positive alarms generated may trigger unnecessary mitigation over legitimate users. The tests performed in this paper highlight the importance of the accuracy of anomaly detection methods. The more accurate the detection, the better the mitigation process.

VI. CONCLUSION

In this paper, we present an online defense system for SDN network environments against DDoS and Port Scans attacks. This system is able to analyze the SDN behavior in time intervals of five seconds, and is divided into three main modules, the detection, the identification and the mitigation modules. The first one is responsible for detecting abnormal SDN traffic behaviors, the second provides the system with relevant information about the anomaly and the third one mitigates its impact over legitimate users.

We present three methods for operating on the Detection module: MLP-DS, PSO-DS and WaveDetect. The first one is a supervised machine learning method which requires previous training data to operate, while the two others are unsupervised approaches of online detection. On the other hand, MLP-DS is able to directly identify the detected anomaly, while PSO-DS and WaveDetect need the Identification module to trigger the correct mitigation approach. We tested these methods against each other, as well as with classic anomaly detection approaches, such as K-means, K-NN and SVM. As the DDoS and port scan detection outcomes fared worse for K-means and K-NN, the PSO-DS, ML-DS, WaveDetect and SVM methods achieved similar results. However, the SVM approach have the burden of a supervised learning without the benefit of directly identifying the detected anomaly, which makes the MLP-DS a more efficient method to be applied when past data is available for training. When there is no training dataset, PSO-DS and WaveDetect can be applied. Between this two methods, WaveDetect achieved better detection outcomes.

Furthermore, we analyze the Mitigation module outcomes for both DDoS and port scan attacks. For the DDoS attacks, a Game Theoretical approach were used directly into the controller to optimize the packet drop rate to minimize the impact of the attack over legitimate users. For the port scans, a directed drop policy (single source IP drop) was applied since the Identification module of the presented SDN defense system was able to detect the source IP of the attacker. The results show that the mitigation approaches were efficient on bringing back the SDN to its normal operation.

For future works, we intend to analyze the behavior of different anomalies into SDN environments, such as flash crowds and worms. Furthermore, we intend to improve the mitigation approach aiming to reduce even more the impact suffered by legitimate users on the process.

REFERENCES

[1] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[2] A. Hakiri, A. Gokhale, P. Berthou, D. C. Schmidt, and T. Gayraud, "Software-defined networking: Challenges and research opportunities for future Internet," *Comput. Netw.*, vol. 75, pp. 453–471, Dec. 2014.

[3] K. Kalkan, G. Gur, and F. Alagoz, "Defense mechanisms against DDoS attacks in SDN environment," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 175–179, Sep. 2017.

[4] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 27–51, 1st Quart., 2014.

[5] M. Jonker, A. King, J. Krupp, C. Rossow, A. Sperotto, and A. Dainotti, "Millions of targets under attack: A macroscopic characterization of the dos ecosystem," in *Proc. ACM Internet Meas. Conf. (IMC)*, New York, NY, USA, 2017, pp. 100–113.

[6] L. F. Carvalho, S. Barbon, Jr., L. de Souza Mendes, and M. L. Proença, Jr., "Unsupervised learning clustering and self-organized agents applied to help network management," *Expert Syst. Appl.*, vol. 54, pp. 29–47, Jul. 2016.

[7] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, and M. L. Proença, Jr., "Network anomaly detection system using genetic algorithm and fuzzy logic," *Expert Syst. Appl.*, vol. 92, pp. 390–402, Feb. 2018.

[8] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.

[9] M. Ahmed, A. N. Mahmood, and M. Islam, "A survey of anomaly detection techniques in financial domain," *Future Gener. Comput. Syst.*, vol. 55, pp. 278–288, Feb. 2016.

[10] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 4, pp. 217–228, 2005.

[11] S. Chang, X. Qiu, Z. Gao, F. Qi, and K. Liu, "A flow-based anomaly detection method using entropy and multiple traffic features," in *Proc. 3rd IEEE Int. Conf. Broadband Netw. Multimedia Technol. (IC-BNMT)*, Oct. 2010, pp. 223–227.

[12] M. V. O. De Assis, A. H. Hamamoto, T. Abrão, and M. L. Proença, Jr., "A game theoretical based system using holt-winters and genetic algorithm with fuzzy logic for DoS/DDoS mitigation on SDN networks," *IEEE Access*, vol. 5, pp. 9485–9496, 2017.

[13] J. H. Cox *et al.*, "Advancing software-defined networks: A survey," *IEEE Access*, vol. 5, pp. 25487–25526, 2017.

[14] M. Paliwal, D. Shrimankar, and O. Tembhurne, "Controllers in SDN: A review report," *IEEE Access*, vol. 6, pp. 36256–36270, 2018.

[15] Q.-Y. Zhang, X.-W. Wang, M. Huang, K.-Q. Li, and S. K. Das, "Software defined networking meets information centric networking: A survey," *IEEE Access*, vol. 6, pp. 39547–39563, 2018.

[16] T. Xu, D. Gao, P. Dong, H. Zhang, C. H. Foh, and H.-C. Chao, "Defending against new-flow attack in SDN-based Internet of Things," *IEEE Access*, vol. 5, pp. 3431–3443, 2017.

[17] P. Zhang and S. Sun, "Decentralized network anomaly detection via a riemannian cluster approach," in *Proc. IEEE Global Commun. Conf. GLOBECOM*, Dec. 2017, pp. 1–6.

[18] Y. Yu, L. Guo, Y. Liu, J. Zheng, and Y. Zong, "An efficient SDN-based DDoS attack detection and rapid response platform in vehicular networks," *IEEE Access*, vol. 6, pp. 44570–44579, 2018.

[19] H. Peng, Z. Sun, X. Zhao, S. Tan, and Z. Sun, "A detection method for anomaly flow in software defined network," *IEEE Access*, vol. 6, pp. 27809–27817, 2018.

[20] L. F. Carvalho, T. Abrão, L. de Souza Mendes, and M. L. Proença, Jr., "An ecosystem for anomaly detection and mitigation in software-defined networking," *Expert Syst. Appl.*, vol. 104, pp. 121–133, Aug. 2018.

[21] S. Zhao, M. Chandrashekar, Y. Lee, and D. Medhi, "Real-time network anomaly detection system using machine learning," in *Proc. 11th Int. Conf. Design Reliable Commun. Netw. (DRCN)*, Mar. 2015, pp. 267–270.

[22] Y. Wang, R.-J. Jin, and W.-J. Han, "An anomaly traffic detection method based on the flow template for the controlled network," in *Proc. 15th Int. Conf. Opt. Commun. Netw. (ICOCN)*, Sep. 2016, pp. 1–3.

[23] A. Karami and M. Guerrero-Zapata, "A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks," *Neurocomputing*, vol. 149, pp. 1253–1269, Feb. 2015.

[24] M. F. Lima, L. D. H. Sampaio, B. B. Zarpelao, J. J. P. C. Rodrigues, T. Abrao, and M. L. Proença, Jr., "Networking anomaly detection using DSNs and particle swarm optimization with re-clustering," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–6.

[25] A. A. Aburomman and M. B. I. Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," *Appl. Soft Comput.*, vol. 38, pp. 360–372, Jan. 2016.

[26] K. J. Singh and T. De, "MLP-GA based algorithm to detect application layer DDoS attack," *J. Inf. Secur. Appl.*, vol. 36, pp. 145–153, Oct. 2017.

[27] C. Siaterlis and B. Maglaris, "Detecting DDoS attacks using a multilayer Perceptron classifier," in *Proc. 20th Int. Conf. Artif. Neural Netw., III*, Mar. 2004, pp. 118–123.

[28] Z. Jadidi, V. Muthukkumarasamy, E. Sithirasenan, and M. Sheikhan, "Flow-based anomaly detection using neural network optimized with GSA algorithm," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst. Workshops*, Jul. 2013, pp. 76–81.

[29] A. Y. Nikravesh, S. A. Ajila, C.-H. Lung, and W. Ding, "Mobile network traffic prediction using MLP, MLPWD, and SVM," in *Proc. IEEE Int. Congr. Big Data (BigData Congr.)*, Jun./Jul. 2016, pp. 402–409.

[30] H. Tian and M. Ding, "Diffusion wavelet-based anomaly detection in networks," in *Proc. 17th Int. Conf. Parallel Distrib. Comput., Appl. Technol. (PDCAT)*, Dec. 2016, pp. 382–386.

[31] S. Kanarachos, J. Mathew, A. Chroneos, and M. Fitzpatrick, "Anomaly detection in time series data using a combination of wavelets, neural networks and Hilbert transform," in *Proc. 6th Int. Conf. Inf., Intell., Syst. Appl. (IISA)*, Jul. 2015, pp. 1–6.

[32] J. Gao, G. Hu, X. Yao, and R. K. C. Chang, "Anomaly detection of network traffic based on wavelet packet," in *Proc. Asia–Pacific Conf. Commun.*, Aug./Sep. 2006, pp. 1–5.

[33] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, 2001.

[34] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, WA, Australia, vol. 4, Nov./Dec. 1995, pp. 1942–1948.

[35] R. Abdel-Kader, M. El-Tarabily, M. Marie, and G. Abdel-Azeem, "A PSO-based subtractive data clustering algorithm," *Int. J. Res. Comput. Sci.*, vol. 3, no. 2, pp. 1–9, 2013.

[36] B. G. Amidan, T. A. Ferryman, and S. K. Cooley, "Data outlier detection using the Chebyshev theorem," in *Proc. IEEE Aerosp. Conf.*, Mar. 2005, pp. 3–8, 2005.

[37] C. Taylor and J. Alves-Foss, "An empirical analysis of NATE: Network analysis of anomalous traffic events," in *Proc. Workshop New Secur. Paradigms (NSPW)*, New York, NY, USA, 2002, pp. 18–26.

[38] S. Chang, X. Qiu, Z. Gao, K. Liu, and F. Qi, "A flow-based anomaly detection method using sketch and combinations of traffic features," in *Proc. Int. Conf. Netw. Service Manage.*, Oct. 2010, pp. 302–305.

[39] S. Haykin, *Neural Networks & Learning Machines*. London, U.K.: Pearson, 2011.

[40] G. Münz, S. Li, and G. Carle, "Traffic anomaly detection using k-means clustering," in *Proc. GI/ITG Workshop MMBnet*, 2007, pp. 13–14.

[41] I. Daubechies, "Wavelets: An overview, with recent applications," in *Proc. IEEE Int. Symp. Inf. Theory*, Sep. 1995, p. 5.

[42] S. Jafarpour, G. Polatkan, E. Brevdo, S. Hughes, A. Brasoveanu, and I. Daubechies, "Stylistic analysis of paintings usingwavelets and machine learning," in *Proc. 17th Eur. Signal Process. Conf.*, Aug. 2009, pp. 1220–1224.

[43] W. Lu, M. Tavallaee, and A. A. Ghorbani, "Detecting network anomalies using different wavelet basis functions," in *Proc. 6th Annu. Commun. Netw. Services Res. Conf. (CNSR)*, May 2008, pp. 149–156.

[44] K. Limthong, P. Watanapongse, and F. Kensuke, "A wavelet-based anomaly detection for outbound network traffic," in *Proc. 8th Asia–Pacific Symp. Inf. Telecommun. Technol.*, Jun. 2010, pp. 1–6.

[45] T. Lotze, G. Shmueli, S. Murphy, S. Murphy, and H. Burkom, "A wavelet-based anomaly detector for early detection of disease outbreaks," in *Proc. 23rd Int. Conf. Mach. Learn. Workshop Mach. Learn. Algorithms Surveill. Event Detection*, 2006, pp. 1–6.

[46] D. C. Hoaglin, "John W. Tukey and data analysis," *Statist. Sci.*, vol. 18, no. 3, pp. 311–318, 2003.

[47] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp. (SIS)*, Apr. 2007, pp. 120–127.

[48] A. A. Poli and M. C. Cirillo, "On the use of the normalized mean square error in evaluating dispersion model performance," *Atmos. Environ. A, Gen. Topics*, vol. 27, no. 15, pp. 2427–2434, Oct. 1993.

[49] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.

[50] Mininet Team. (2018). *MiniNet Overview*. [Online]. Available: http://mininet.org/overview/

[51] Philippe Biondi and The Scapy Community. (2018). *Scapy*. [Online]. Available: http://www.secdev.org/projects/scapy/

[52] V. Hautamaki, I. Karkkainen, and P. Franti, "Outlier detection using k-nearest neighbour graph," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, vol. 3, Aug. 2004, pp. 430–433.

[53] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, New York, NY, USA, 2000, pp. 427–438.

[54] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

MARCOS V. O. DE ASSIS received the master's degree in computer science from the State University of Londrina, Brazil, where he is currently pursuing the Ph.D. degree with the Electrical Engineering Department. He is part of the Computer Networks and Data Communication Research Group. He is currently a Professor with the Engineering and Exact Department, Federal University of Paraná, Brazil. His research interests are in the management and security of large-scale computer networks and software defined networks.

MATHEUS P. NOVAES received the B.S. degree in computer science from the State University of Londrina (UEL), Brazil, in 2017, where he is currently pursuing the M.Sc. degree in computer science. Since 2015, he has been a member of the Computer Networks and Data Communication Research Group, Computer Science Department, UEL. His research focus is on management and security of computer networks.

CINARA B. ZERBINI received the B.S. degree in computer science from the State University of Londrina, Brazil, in 2016, where she is currently pursuing the master's degree with the Computer Department. She is currently a member of the Computer Networks and Data Communication Research Group. Her main interests are signal processing and statistical tools, security of computer networks, and software-defined networking.

LUIZ F. CARVALHO received the master's degree in computer science from the State University of Londrina in 2014 and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas in 2018. He is currently a Lecturer and a member of the Computer Networks and Data Communication Research Group, State University of Londrina. His main research interests are management and security of computer networks and software-defined networking.

TAUFIK ABRÃO (M'97–SM'12) received the B.S., M.Sc., and Ph.D. degrees in electrical engineering from the Polytechnic School, University of São Paulo, São Paulo, Brazil, in 1992, 1996, and 2001, respectively. From 2007 to 2008, he was a Post-Doctoral Researcher with the Department of Signal Theory and Communications, Polytechnic University of Catalonia, Barcelona, Spain. In 2012, he joined the Southampton Wireless Research Group, The University of Southampton, Southampton, U.K., as an Academic Visitor. Since 1997, he has been with the Communications Group, Department of Electrical Engineering, State University of Londrina, Brazil, where he is currently an Associate Professor of telecommunications and the Head of the Telecomm and Signal Processing Laboratory. He has participated in several projects funded by government agencies and industrial companies. He has supervised 21 M.Sc., six Ph.D., and three post-doctoral students. He has co-authored 10 book chapters on mobile radio communications and over 180 research papers published in specialized/international journals and conferences. His current research interests include communications and signal processing, especially massive multi-in multi-out, and OFDM/OFDMA systems, detection and estimation methods, cooperative communication and relaying, resource allocation, and heuristic and convex optimization aspects of 4G and 5G wireless communication systems. He served as a TPC Member for several symposiums and conferences. He is a Senior Member of the Brazilian Telecommunication Society. He is involved in editorial board activities of six journals in the telecommunications area. He has been serving as an Editor for the IEEE Communications Surveys and Tutorials since 2013, the IEEE Access since 2016, and the *IET Journal of Engineering* since 2014.

MARIO L. PROENÇA Jr. received the M.Sc. degree in computer science from the Informatics Institute, Federal University of Rio Grande do Sul, in 1998, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas in 2005. He is a Master's Supervisor of computer science with the State University of Londrina and a Ph.D. Supervisor with the Department of Electrical Engineering, State University of Londrina (UEL), Brazil. He is currently an Associate Professor and a Leader of the research group that studies computer networks with the Computer Science Department, UEL. He has authored or co-authored over 100 papers in refereed international journals and conferences and books chapters, and holds one software register patent. He has supervised 12 M.Sc. and two Ph.D. students. His research interests include computer network, network operations, management and security, and IT governance.

• • •