IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 27, 2020, accepted May 17, 2020, date of publication May 27, 2020, date of current version June 8, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2997939

Artificial Immune Systems and Fuzzy Logic to Detect Flooding Attacks in Software-Defined Networks

GUSTAVO F. SCARANTI[1], LUIZ F. CARVALHO[2], SYLVIO BARBON, Jr.[1], AND MARIO LEMES PROENÇA, Jr.[1]

[1]Computer Science Department, State University of Londrina, Londrina 86057-970, Brazil
[2]Computer Engineering Department, Federal University of Technology–Paraná (UTFPR), Apucarana 86812-460, Brazil

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

This work supported in part by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grant 310668/2019-0.

ABSTRACT Software-defined Networking (SDN) has been discovered as an architecture that uses applications to make networks flexible and centrally controlled. Although SDN provides innovative management, it still susceptible to attacks daily. Traditional detection approaches may not be sufficient to contain these threats. In this paper, we present an Artificial Immune System based IDS named AIS-IDS, which is inspired by the human body's defense cells. AIS-IDS can detect variations in network behavior and identify attacks without prior knowledge about them. Along with AIS, the fuzzy logic is applied on detection to minimize the uncertainty when there is no clear boundary between anomalous and normal traffic behavior. We have simulated portscan and flooding attacks as well as used a public dataset with several types of DDoS attacks to assess our proposal. We compared the AIS-IDS performance with Naive Bayes, k-nearest neighbors, and the Local Outlier Factor. The AIS-IDS outperformed the compared algorithms, achieving f-measure rates 99.97% and 92.28% when submitted to a simulated and a public dataset, respectively.

INDEX TERMS Negative selection, software-defined networking, anomaly detection, intrusion detection system.

I. INTRODUCTION

The networking landscape has continuously evolved to meet the demands of users and emerging services. New technologies, such as Internet of Things (IoT), smart cities, autonomous vehicle networks, and health monitoring systems, are some examples that are already part of everyday life. However, they have vulnerabilities that can be exploited [1], [2].

The vulnerabilities exploited by intruders are often operational problems or internal faults caused by equipment misconfigured. Taking advantage of these security breaches, the intruder may present deviant behavior in comparison to legitimate users when attacks the network services [3]. Anomaly is the term used to refer to a deviant behavior that can represent a threat able to impact service quality and even compromise the entire network [4], [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Yu.

Considering all these threats, it is critical to maintaining systems and networks capable of dealing with anomalies that affect their proper functioning. Network administrators use security mechanisms, like Intrusion Detection System (IDS), to analyze the network traffic [6], [7]. When an unusual traffic change is detected, the IDS raises the alarm to warn about the potential threat found. Alarms assist the network administrator in intervening as soon as possible, addressing the anomalies [8].

With the advance of networking technologies and the increasing amount of equipment connected, traditional managing of all network equipment is no longer viable, being necessary effective management, which provides flexibility to the administrators and ensures network resilience [9]–[11]. Software-defined Networking (SDN) is an emerging networking paradigm capable of managing different network types and sizes through a software perspective [2]. Since this type of network has a centralized control plan, called controller [12], it is possible to monitor the entire

network and to obtain an extensive analysis of its operations [13] [14], [15]. Contrary, in traditional networks, each device has its own software, security rules, link failure strategy, and forwarding mechanisms. If any of these mechanisms need to be updated, each network device should be managed individually [16], [17].

The possibility of network management via software allows numerous customization of network services [10], [18], including security. Further, SDN provides several advantages grounded by machine learning classification, pattern recognition, and meta-heuristic optimization to improve the administrator task [19]. In this regard, we proposed AIS-IDS, a biological inspired IDS with fuzzy logic, to automate the detection and mitigation of network anomalies. The proposal is placed on the control plane and has three modules with specific functions: one for collecting flows, another for detection, and the lastly for mitigation.

The flow collect module collects the IP flows every second and extracts traffic features that describe the network behavior. The AIS detection module uses the Artificial Immune Systems (AIS), a class of algorithms inspired by the functioning of body defense system. As the human immune system can identify and reacts to foreign organisms in our body, the AIS-IDS can also detect and respond to patterns that differ from those presented in its training phase. AIS-IDS enables an anomalous network traffic pattern recognition, only using normal traffic behavior. The Fuzzy Inference System is used to resolve the degree of uncertainty in the anomaly detection process, as in the network traffic analysis. The mitigation module will recognize the anomaly type and identifies suspicious IP addresses and ports to define the mitigation strategies to block the attack.

To evaluate the efficiency of the AIS-IDS, we compare our approach with Naive Bayes, k-nearest neighbors (kNN), and Local Outlier Factor (LOF) in simulated scenarios with DDoS and portscan attacks in the Mininet network emulator using the Floodlight SDN controller. We also use a public dataset, which contains several types of flooding attacks.

The rest of the paper is organized as follows. Section II presents the recent works of the area. Section III describes the structure and functioning of the proposed AIS-IDS. The experimental tests and results are discussed in Section IV. Finally, Section V concludes the paper.

II. RELATED WORK

Anomaly detection has become a significant data mining task for analysis of inconsistent or suspicious data. It has been attracting attention from many researchers in various application fields, such as financial analysis, fraud detection, network intrusion detection and in new environments, including IoT [20], [21]. Recently, researchers proposed various anomaly detection models to protect and keep the network safe from malicious users [1], [2], [22].

Many types of algorithms and techniques have been used to detect network traffic anomalies. Statistical algorithms [23], clustering [6], classification [24], evolutionary [25] are some examples of techniques used in IDS. The IDS may be inside or outside of the network to protect it from various attackers trying to gain access to the network [26].

Intrusion detection can be divided into signature-based and anomaly-based detection. The former uses signatures, which can be considered a sequence of occurrences that define an attack [27]. It can achieve high detection rates with known attacks but has a lower performance for new or unknown threats. The latter creates a profile representing the normal network behavior, and deviation on this pattern can be an anomaly [8], [28].

Arivudainambi *et al.* [29] presented an IDS using the Convolutional Neural networks (CNN) and Lion optimization algorithm (LOA). CNN is a model for visual recognition and LOA is a bio-inspired algorithm that mimics the lifestyle of lions. CNN is used to transforms the collected traffic features into high-level features, and LOA performs a feature selection in these features. The NSL-KDD dataset[1] is used to evaluate the approach. It compares with other bio-inspired methods in feature selection, such as Ant Colony Optimization and Bee colony optimization algorithm. The CNN with LOA approach outperforms the compared algorithms in 1.2%, reaching an accuracy value of 98.2%. Although the approach is lightweight, it is focused on DDoS attacks and did not use mitigation strategies for the detected threats.

Regarding signature-based in SDN networks, Lai *et al.* [24] proposed a flow-based anomaly detection using Multilayer Perceptron (MLP). Six features were extracted from the IP flows and used in the input layer of the MLP. In addition, it was developed a packet-based detector, and the approaches were compared using the NSL-KDD dataset. Both methods have been able to detect threats. However, the flow-based approach performed better because it caused less overhead and presented a satisfactory performance compared to the packet-based model. As the approach uses anomalous data for training, it can suffer from misclassification when new types of attacks occur. Also, the scheme presented for the authors has no attack mitigation policy.

Still exploring the techniques applied in SDN networks, Mansour *et al.* [30] proposed an approach using Genetic Algorithm (GA). Their method was designed to select the best attributes of the KDD Cup dataset[2] and also to calculate a fitness function for anomaly detection. A fitness function was defined for each protocol (TCP, UDP, and ICMP) to achieve better results. When an anomaly is detected, the approach divides its analysis into four attacks: DoS, probe, R2L, and U2R. After classifying the attack, the value of the fitness function is recalculated to classify other attack instances. Overall, the approach was able to detect most of the attacks, and the True Positive Rate (TPR) was close to 80%. However, the proposal used abnormal traffic for training that may restrict the recognition of unknown anomalous events. Also, the countermeasure strategy blocks the communication to the

[1] Available at https://www.unb.ca/cic/datasets/nsl.html

[2] Available at http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

attacked device, which can lead to the blocking of legitimate requests.

Duy *et al.* [31] presented an anomaly-based method that uses the concept of entropy to detect DDoS attacks. Their method has four modules: flow collector, entropy-based sensor, attack confirmation sensor, and mitigator. With the collected flows, it is computed entropy of destination IP addresses to find abnormalities. Once an unusual network behavior is detected, the attack confirmation module analyzes all relevant flows to find the origin of the threat. With the source of the attack located, the mitigator modifies the flows to block the attack. The approach was evaluated using the mininet network emulator. The authors reported that the approach detects attacks in early stages, but only DDoS was explored.

Vidal *et al.* [25] used NFV to elaborate a decentralized anomaly-based method through various agents spread through the network. These detectors can use innate or adaptive from immune system theory to detect potential threats and mitigate them. KDD Cup and CAIDA 2007 datasets were used to evaluate the system. Also, the tool DDoSIM was applied to generate flooding attacks in the subnet traffic of the Faculty of Computer Science of the Complutense University of Madrid. In the results, the authors pointed out the adaptive responses were more effective than innate reactions. Moreover, innate responses behave like conventional IPS (Intrusion Prevention System), and adaptive reactions can be used as an alternative over traditional mitigation schemes. The approach has been effective in blocking attacks, but it has to be spread over the entire network for its proper operation. The proposed detection is specialized in DDoS attacks, so attacks of other types may not be detected.

Rathore *et al.* [32] presented a bio-inspired anomaly-detection approach to detect DoS attack in SDN networks. The innate system is used to mitigate DoS attacks from malicious users, and the adaptative system is applied to mitigate DoS attacks caused by switches. When an attack is detected, a mitigation module blocks the malicious user cutting off the communication with the network. When an attack comes from switches, the mitigation module reduces the bandwidth from the switch to protect the SDN controller. The mininet network emulator was used to evaluate the proposed systems. The presented approach outperforms other techniques regarding the time necessary to react to threats and overhead of the detection scheme. However, the approach is specialized in DoS attacks, and the network used for the comparisons is tiny, containing only a few hosts.

Among the works that address AIS in traditional networks, Aziz *et al.* [27] proposed a two-tier IDS. The first layer uses negative selection to create the detectors with feature values representing the normal network behavior. When there are variations in the normal behavior of the network, the second layer applies classifiers to identify specific attacks. The classifiers GA, AIS, Decision Trees, MLP, and Naive Bayes were tested to ranking the best classifiers for the second layer. The IDS was evaluated using the NSL-KDD dataset.[3] In the tests, none of the tested classifiers were superior in detecting all types of attacks. The authors report that the average f-measure among all classifiers was 78%. The approach used the AIS to detect anomaly moments. However, it used classifiers that need training with the attack samples to identify anomalies with precision, hampering the approach in detecting attacks that are not present within the training dataset.

Hooks *et al.* [33] presented a comparative between negative selection approach and clonal selection in anomaly detection. Both techniques use the NSL-KDD dataset, and various scenarios were created, changing the number of attributes, instances, and detectors. The authors concluded that both approaches suffered from large amounts of samples and features, and also the negative selection delivered results faster than clonal selection. The approach requires at least 22 features for accuracy greater than 80%, which generates a high computational cost for large networks. Both algorithms use a training dataset with attacks so that the unknown attacks may not be detected.

Shen and Wang [34] presented an IDS based on negative selection. The paper also compared Rough Set, Linear Genetic Programming (LGP) and Multivariate Adaptive Regression Splines (MARS) to find the best feature selection algorithm. Each algorithm chose six features out of 41 available in the KDD Cup 99 dataset. According to the authors, the MARS algorithm obtained the best features, reaching detection rates similar to other IDS in the literature. The proposed IDS did not take into account the mitigation process and was deployed in a traditional network architecture.

Another approach was presented by Tabatabaefar *et al.* [35], which used two AIS techniques: negative and positive selection. The method has two categories of detectors. The first category was designed for recognizing legitimate traffic and the second one identifies anomalies. Particle Swarm Optimization algorithm was used to perform detector training. The tests were performed using the KDD Cup 99 dataset. Results showed that the proposed approach presented a detection rate over 99%. Using the same dataset, Suliman *et al.* [7] developed an IDS using the clonal selection technique. The authors reported that results obtained were comparable to other approaches found in the literature. The proposal was based on the detection of DoS and probe attacks. The paper is focused on traditional networks and does not discuss strategies for attack mitigation.

Among the works found in the literature, only one resembled our proposal. Zhou and Pezaros [36] presented an IDS using negative selection on SDN networks. The approach builds detectors and conducts training on data collected from network switches. Subsequently, the detectors are sent to the controller, which selects the best detectors to obtain the best collection possible. This collection is sent to all switches to use these detectors to locate anomalies. The proposed

[3] Available at https://www.unb.ca/cic/datasets/nsl.html

TABLE 1. Related works comparison.

Works	Year	Technique	Mitigation	Detect Varied Attacks	Network Type	Dataset
Arivudainambi et al. [29]	2019	CNN + LOA	Yes	Yes	SDN	NSL-KDD
Lai et al. [24]	2019	MLP	No	Yes	SDN	NSL-KDD
Mansour et al. [30]	2018	GA	Yes	Yes	SDN	KDD
Vidal et al. [25]	2018	AIS	Yes	No	SDN	KDD, CAIDA and Generated (DDoSIM)
Rathore et al. [32]	2019	AIS	Yes	No	SDN	Simulated (Mininet)
Aziz et al. [27]	2017	Varied	No	Yes	Traditional	NSL-KDD
Duy et al. [31]	2018	Entropy	Yes	No	SDN	Simulated (Mininet)
Hooks et al. [33]	2018	AIS	No	Yes	Traditional	NSL-KDD
Shen et al. [34]	2011	AIS	No	Yes	Traditional	KDD
Tabatabaefar et al. [35]	2017	AIS	No	Yes	Traditional	KDD
Suliman et al. [7]	2017	AIS	No	Yes	Traditional	KDD
Zhou et al. [36]	2019	AIS	No	Yes	SDN	KDD and NSL-KDD
	2020	**AIS (Proposed)**	**Yes**	**Yes**	**SDN**	

IDS was evaluated with two datasets, the KDD Cup and NSL-KDD. According to the authors, the approach proved to be more efficient in detection compared to previous IDS approaches using AIS and also lighter to other methods assessed in the paper. The solution did not propose a mitigation strategy and used the network switches to carry out the anomaly detection, which can lead to a data plane overload.

Table 1 summarizes all related works presented. Works [27] and [31] use techniques to profile regular network operation. However, if the network behavior changes, classification errors can occur. Similarly, [24] and [30] can suffer from misclassification of anomalous events because they use historical data with anomalies for training, and their approach may not generalize the detection from other unknown attacks. Besides, the former does not provide policies for mitigation of attacks, and the latter approach blocks all communication to the victim device, impairing legitimate packets.

Some works focused on detecting only one type of attack [25], operate in traditional network architecture [34], or does not act against the attack spreading [36]. In this paper, we present a comprehensive AIS-based IDS, which can detect anomalies in near-real time. Also, we focus specifically on the SDN environment to take advantage of the programming capabilities of network resources, providing countermeasures for detected threats, leveraging network resilience.

III. SYSTEM OVERVIEW

The AIS-IDS has three modules coupled in the SDN controller: Flow Collector, AIS Detection, and Mitigation. The first module periodically acquires IP flows from the data plane using the OpenFlow Protocol and preprocesses them to represent the network behavior. The processed data is sent to the second module, and the AIS is used to classify network behavior as abnormal or normal. If the classified behavior is normal, forwarding rules were created to forward the packets of benign flows to the destination. In contrast, when an anomalous event is detected, the malicious traffic is blocked by the Mitigation module by creating forwarding rules to block the packets of anomalous flows. All these forwarding rules are sent through the OpenFlow Protocol to switches in the data plane. Figure 1 presents an overview of the entire functioning of AIS-IDS.

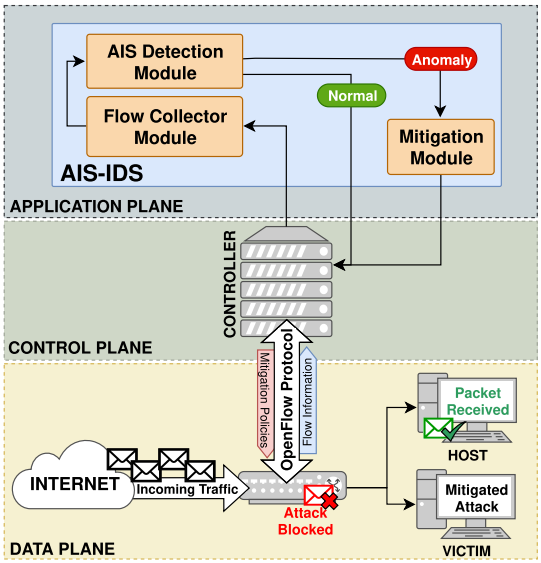

FIGURE 1. AIS-IDS Overview.

A. FLOW COLLECTOR MODULE

To collect the data used by the AIS detection module, we used the OpenFlow protocol. This protocol is the common interface used to instruct switches about new forwarding rules for incoming packets as well as provides access to statistical data and control plan configuration [37].

An IP flow is a collection of IP packets with similar features, like source and destination IP addresses, ports, and other features [22]. Each flow has rules assigned to it for enabling packet forwarding. Packets that share some characteristics use the same flow forwarding rule. For each new packet that arrives at a switch and does not have a flow that matches it, a new flow is created to allow the packet to be forwarded to its destination.

Every instant, many flows are created and expired in an SDN network. The more frequent the collection of flows

information is, the more accurate the anomaly detection and, consequently, the faster actions to stop an attack from spreading are taken. To detect attacks in near real-time, traffic statistics must be collected in short time intervals. However, a disadvantage of periodic data collection in a short time interval is the volume of flows to be analyzed. Thus, creating an IDS that reacts accurately in real-time when detecting threats is a challenging issue. In this regard, each module of our proposal was designed to seek the best trade-off between computational efficiency and anomaly detection effectiveness.

The first module collects statistics from flows in the switches and extracts the traffic features every second. The AIS-IDS uses four traffic features: source and destination IP addresses and source and destination ports. The benefit of using these traffic features is twofold. Firstly, IP addresses and ports become sensitive to changes in traffic behavior when converted to quantitative, leading to identification of moments in which the traffic behaves anomalously. Secondly, it is possible to locate hosts and services involved in the anomalous event and mitigate them [38].

The entropy is efficient in measuring the distribution of qualitative features [31] [39]. We use Shannon entropy [40] to quantify the level of concentration of information according to the distribution of a set of samples [41]. Shannon entropy is computed based on a histogram for each qualitative traffic feature and use this information in equation (1), in which X stands for one of the qualitative features, n_i is the number of times the i-th IP address or port was observed in the interval of analysis and $s = \sum_{i=1} n_i$ is the total of all occurrences in the histogram.

$$H(X) = -\sum_{i=1} \left(\frac{n_i}{s}\right) \log_2 \left(\frac{n_i}{s}\right) \tag{1}$$

Thus, the Flow Collector is responsible for collect the incoming traffic, process, summarize, and forward it to the AIS module for pattern characterization. The internal procedure of this module is presented in Figure 2.

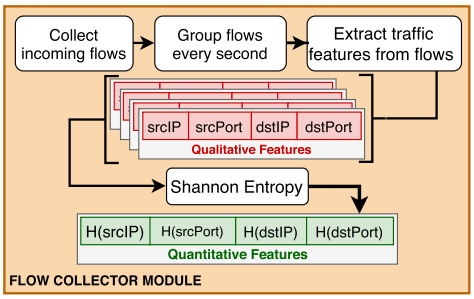

FIGURE 2. Flow Collector Module scheme: from incoming flows to quantitative features.

B. AIS DETECTION MODULE

Artificial Immune System (AIS) is a technique of computation intelligence inspired in biological immune systems [34]. There are several implementations of AIS theory, each one suitable to the objective and problem addressed [42].

One of the best-known AIS algorithms is the Negative Selection Algorithm (NSA), proposed by Forrest *et al.* [43]. NSA is a computational model to generate immune detectors. It simulates the recognition of biological antibody epitopes to accomplish the self and non-self classification [44]. NSA classifies the data into self when it is legit and non-self when it is anomalous. An advantage of this approach is the ability to detect threats without requiring abnormal examples in the training dataset [45].

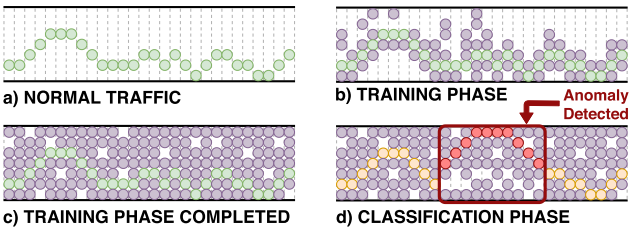

FIGURE 3. NSA algorithm overview: training and classification phases.

Figure 3 illustrates the operation of the Negative Selection. Figure 3(a) represents the free anomaly network behavior. Each column represents a one-second interval, and each green circle is the data collected and preprocessed from the Flow Collector Module. Figure 3(b) presents the Training Phase, in which the detectors, represented by the purple circles, are created. At the end of the training phase, the objective is that the maximum possible behaviors are covered by the detectors, as in Figure 3(c). The classification phase is shown in Figure 3(d), where the current traffic is captured by the Flow Collector and is represented by the yellow circles. If the current traffic is different than expected, the detectors present at the anomalous moment are activated and inform about the detected anomaly. In the figure, red circles represent the activated detectors.

The AIS Detection Module divides NSA into two phases. The training phase creates detectors that are used in the classification for recognizing unusual behavior. In both stages, the similarity calculation is used. In the training phase, the similarity is used to accepts or rejects the created detectors. In the application phase, similarity detects intervals that contain anomalies.

The Absolute Distance (AD) technique was chosen to calculate the similarity. It has been successfully used in other works [46], [47]. The AD is expressed by equation (2), in which p and q are the values being compared to calculate the similarity.

$$\delta = \sqrt{(p-q)^2} = |p-q| \tag{2}$$

A detector is represented by a set $\Phi_i^t = \{H(srcIP), H(dstIP), H(srcPort), H(dstPort)\}$, where $1 \leq i \leq n$ represents its index, t represents the second to which group belongs. Each detector has values of entropy of source and destination IP, and source and destination ports, respectively.

A collection of detector $\Omega_t = \{\Phi_1^t, \Phi_2^t, \ldots, \Phi_n^t\}$ contains all the detectors that were created for the instant t.

The training phase needs anomaly-free training dataset to generate the detectors. Thus, it is commonly generated in a controlled environment to provide a dataset without any type of attack. The training phase starts by generating a random value for each feature f present in the training dataset. This random value is generated from the uniform distribution on the interval zero and the highest value of this feature in the training data. After that, a similarity score δ_f is calculated between the random value and all samples of feature f, and the score must be less than the minimum similarity hyperparameter k; otherwise, a new random value is generated for that feature, and the process restarts.

A detector is complete only when its features have similarity values greater than the minimum similarity with the training dataset. The new detector is added to the detectors' collection and the generation process terminates when this collection reaches n detectors, i.e., $|\Omega_t| = n$.

As the detectors are generated, the similarity calculation between the detectors is also carried out, to avoid creating similar detectors and thus not covering the entire search space in an optimized way. The smaller the similarity score, the more similar the detector is to the instances of training. The objective is grounded on creating dissimilar detectors from the observed training and previously detector generated. Thus, a detector needs to represent a possible abnormal traffic event. Accordingly, the collection of detectors is intended to represent possible unusual behaviors.

Calculating feature-by-feature similarity, rather than a complete detector, speed up the generation process because the generated detector is not entirely discarded, but only feature values that were not accepted by the minimal similarity. Also, in the classification phase, it is possible to evaluate each feature separately and thus detect attacks with specific behaviors. At the end of the first phase, the generated detectors' collection is sent to the second phase of the AIS module. Algorithm 1 describes the first phase of the detectors' generation.

Algorithm 1 Pseudocode for Detectors Generation

Input : Training Data (tData), number of detectors (n) and minimum similarity (k)
Output: Detectors Collection (Ω_t)

```
Ω_t ← ∅
numDetectors ← 0
while numDetectors ≤ n do
    foreach feature ∈ tData do
        featColumn ← tData[feature]
        detFeatColumn ← Ω_t[feature]
        isused ← no
        while isused = no do
            rFeature ← genRandomFeature()
            if getSimilarity(rFeature, featColumn) ≥ k
            and
            getSimilarity(rFeature, detFeatColumn) ≥ k
            then
                newDetector[feature] ← rFeature
                isused ← yes
            end
        end
    end
    Ω_t ← Ω_t ∪ newDetector
    numDetectors ← numDetectors + 1
end
return Ω_t
```

The second phase of the AIS Module classifies the ongoing traffic into normal or abnormal. Samples collected and preprocessed by the Flow Collector Module are receipts to calculate the similarity with the detectors' collection previously generated.

Using collected or extracted data by statistical means to detect anomalies can lead to significant errors. In addition, there is no clear boundary between what is abnormal or usual behavior [48], [49]. In this sense, instead of using a hard threshold like classical classification, we used fuzzy logic to provide the recognition of network threats.

A Fuzzy Inference System aims to generate an output value supported by fuzzy logic on a given input. It assigns values ranging from 0 to 1 [50] to provide a rational analysis in an environment that has no precise information or incomplete ones [48], as the network traffic analysis. The first step is to fuzzify the input through a membership function, creating a fuzzy set. Several rules are applied in the fuzzy set, preparing it for the defuzzification process, which outputs a crisp set that takes into account all the rules used previously.

To use a Fuzzy Inference System in the similarity score δ_f, calculated with a given feature f from the incoming data and the detectors' collection (Ω_t), one can use the fuzzy membership function. We applied the Gaussian membership, expressed by the equation (3),

$$\zeta_f = \begin{cases} e^{\left(\frac{-(\delta_f - k)^2}{2\sigma_f^2}\right)}, & \text{if } \delta_f > k \\ 1, & \text{otherwise} \end{cases} \tag{3}$$

where δ_f is the similarity score, k is the minimum similarity and σ_f is the standard deviation of the similarity score of feature f. If δ_f is less than k, the result of this equation is 1. This rule was created to avoid the similarity value decreases when the δ_f is smaller than k.

After using the fuzzy membership function, each sample has four ζ_f fuzzy values, i.e., one value for each traffic feature. One way to calculate an overall score of the sample is to sum the similarity scores. However, each feature is affected differently by different types of attacks. Therefore, calculating an importance coefficient for each feature provides more accurate detection, making it possible to detect attacks that do not significantly affect the normal behavior of the network or have specific behaviors, such as portscan attacks.

In this manner, to obtain these four coefficients importance, it is used the multinomial logistic regression (MLR). The MLR is a method that uses a logistic regression in data that dependent variables are unordered, and independent values are continuous or categorical [51], [52].

The MLR method requires a labeled dataset containing normal and abnormal traffic samples to generate the importance coefficients. The coefficients were incorporated into the calculation of the overall score of each sample. The MLR was used only once to obtain these coefficients, and it is no longer necessary to have a labeled database for the rest of the detection.

Algorithm 2 Pseudocode for Data Classification

Input : Data to be classified (newData), detectors collection (Ω_t), importance values (impValues) and minimum similarity (k)
Output: Classification Result

```
score ← 0
foreach attribute ∈ newData do
    attrColumn ← Ω_t[attribute]
    simScore ← getSimilarity(attribute, attrColumn, k)
    fuzzyValue ← calcGaussianFuzzy(simScore, k, sigma)
    score ← score + (fuzzyValue × impValues[attribute])
end
if score > Γ then
    return anomalous
end
else
    return normal
end
```

Thus, the overall sample score is calculated, multiplying each fuzzy value with its respective importance coefficient, and sum these results. When this score is higher than the cutoff threshold of Γ, this data is considered abnormal; otherwise, it is normal. Unlike rate-limiting approaches that use a fixed number to define maximum connections, the AIS-IDS uses the normal network behavior to define when an attack is occurring. Algorithm 2 presents the pseudo-code for the second phase and Figure 4 presents an overview of both phases of the AIS detection module.

Traditional IDS typically uses several days as a training dataset. Thus, a new behavior can be delayed to be recognized. As a result, new patterns in the network data may not be detected rapidly [53]. To address this issue, we used a sliding window. This technique ensures that the AIS algorithm rapidly generates the detectors, and the analysis of new flows can be performed in near real-time. Another advantage is that only recent data is used for the detectors creation, thus improving the detection capacity, since, during the day, the behavior of the network can lead to erroneous detections.

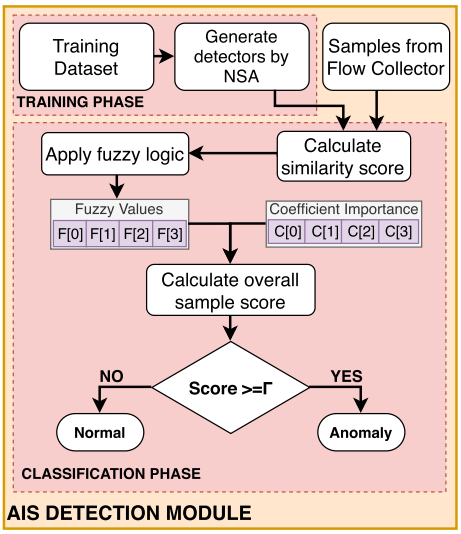

FIGURE 4. AIS detection module scheme.

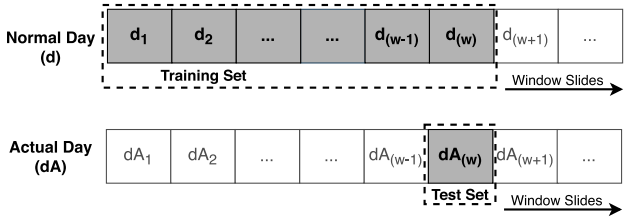

FIGURE 5. Sliding window scheme.

Figure 5 shows how the sliding window works using the window size w. As can be seen, two datasets are used, one of them containing one-day traffic without anomalies (d) and the other one with the measurements performed periodically during the current day (dA). The former is used to detector's generation process, and the latter is used in the classification process. For instance, the last w samples of the dataset d is used to detectors generation and classify the sample $dA_{(w)}$.

After the sample classification into normal or abnormal, both sliding windows advance one position to classify the next incoming sample. During the beginning of the process, if the last w samples are not available for training, all available samples are used until the number is reached. Thus, the w hyperparameter defines the number of samples used in the generation phase, and the variation directly affects the quality of the classification and the time required to perform this task.

C. MITIGATION MODULE

When the AIS detection module detects the anomaly, it forwards all flows from the anomalous interval to the mitigation module to apply packet dropping policies to contain the detected attack. DDoS and portscan alter the behavior of traffic attributes, making it possible to discover IP addresses and ports in malicious communication.

Two strategies are defined and used according to the type of anomaly, DDoS, and PortScan attacks. DDoS floods

a server to make an online service unavailable through multiple requests sent from various sources. The first step to cease the attack is to discover the IP address of the host that is under attack. Then it is identified all the hosts that are flooding a specific port associated with a service offered by the attacked host.

After the malicious IP address identification, the AIS-IDS mitigation module creates a flow with discard action enabled to drop packets from the attacker. This flow is sent from the controller to switches in the data plane via OpenFlow. Incoming packets that match with these mitigation flows are dropped to avoiding overwhelming the target. Portscan attacks attempt to discover active services by sending messages to different ports of the victim. The scanning of ports can be performed from a single-source or coordinated by multiple adversaries. This second approach is not easy to be detected by an IDS because the scanning traces are scattered on different hosts [54], [55]. After recognizing the target, the policy for interrupting a portscan attack finds the IP address of the attacker based on an IP address associated with several flows destined to different ports of the target.

When the attacker's IP address is identified, the policy enables the strategy of discard all packets from the malicious source intended to the destination. The blocker flow created has the highest possible priority (65535) among all flows, so it is executed before all regular forwarding flows. Algorithm 3 presents the pseudo-code for the mitigation module and Figure 6 shows the module scheme.

Algorithm 3 Pseudocode for Mitigation Module

Input : Samples of anomalous instant (anomalySamples)
Output: Mitigation Flow

1 Identify the target and ports under attacks from anomalySamples
2 Create a blocker flow entry
3 Define the destination IP and port of flow as Victim's IP address and Port
4 Define the drop rule to the flow actions
5 Define high priority (65,535) to the flow
6 Send the mitigation flow to the data plane via OpenFlow protocol

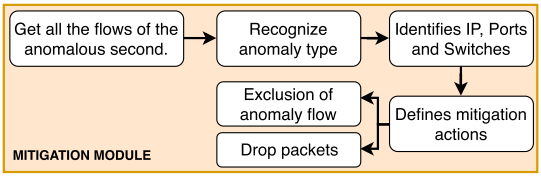

FIGURE 6. Mitigation Module Scheme.

IV. TEST SCENARIO AND RESULTS

A. TEST SCENARIO

To evaluate the AIS-IDS, we used Mininet [56], which enables the emulation of an SDN structure with switches, controllers, hosts, and links in a single Linux kernel. Applications developed in the emulated environment can be deployed in real scenarios with minimal changes. Also, Mininet supports Open vSwitch, a virtual multilayer switch that can handle both software and hardware-based switching. All steps required for traffic monitoring, feature extraction, detection, and mitigation of anomalies were performed by applications incorporated into the Floodlight, a Java-based open-source controller. The entire topology was tested on a six-core 2.6 GHz with 32 GB of memory, running Mininet 2.2.2, Floodlight 1.2, and Ubuntu 18.04.

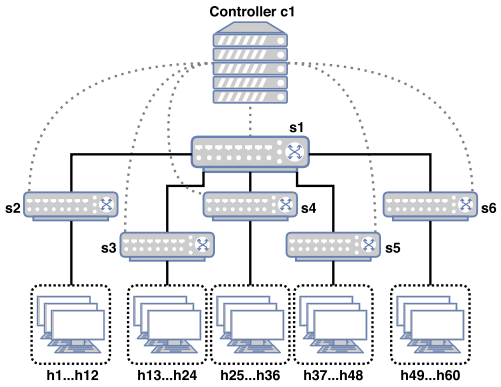

FIGURE 7. Emulated topology on Mininet.

The emulated topology comprised six Open vSwitch switches, sixty hosts and an SDN controller, as can be seen in Figure 7. The chosen topology was tree-based. s1 is the root switch, and all the other switches were connected to it. There were twelve hosts connected in each switch of the second level of the tree. The Floodlight controller communicated directly to each switch via OpenFlow protocol.

We used Scapy 2.4.0[4] to generate the usual traffic in the emulated network. Scapy is a powerful tool to support a testbed environment, to forge packets and send them through a network interface. Flows that composed the network traffic were generated randomly, and the volume of data has changed throughout the day to simulate variations in network usage.

Distributed Denial of Service (DDoS) attacks were generated using hping3[5] software. We have configured multiple sources to create several requests to a single destination in the emulated network. A host in the network was the victim of portscan attacks conducted by a host with Scapy. In this case, the attacker has sent packets with the SYN flag enabled to different ports of the destination host, trying to receive confirmation of ports that were in operation.

Three days of traffic was generated in the controlled environment for assessing the efficiency of our proposal. The first one comprises a day with non-anomalous traffic. These data were used as the training dataset of the AIS-IDS. It was also generated two more days, each of these days with a DDoS and a portscan attack. We make available the dataset of both

[4]https://scapy.net/
[5]https://github.com/antirez/hping

TABLE 2. Attacks Description.

Day	First Attack	Second Attack
1	None	None
2	Type of Attack: DDoS Attacking IPs: 10.0.0.3-10.0.0.11 Destination IP: 10.0.0.42:443 Duration: 10:15:00 - 11:30:00	Type of Attack: Portscan Attacking IPs: 10.0.0.48 Destination IP: 10.0.0.8:1-30000 Packet Interval: 0.25 Duration: 13:25:00 - 14:35:00
3	Type of Attack: Portscan Attacking IPs: 10.0.0.38 Destination IP: 10.0.0.19:1-30000 Packet Interval: 0.30 Duration: 09:45:00 - 11:10:00	Type of Attack: DDoS Attacking IPs: 10.0.0.23-10.0.0.26 Destination IP: 10.0.0.55:2222 Duration: 17:37:00 - 18:55:00

abnormal and normal traffic resulting from the traffic generation.[6] The complete information of the attacks is presented in Table 2.

B. AIS-IDS EVALUATION

The proposed AIS-based IDS has some hyperparameters that need to be defined for its use. The size of the sliding window is w, k is the minimum similarity score, n stands for the number of detectors and Γ is the cutoff threshold value. The tests were performed using the first day as the training dataset and the second day as the test dataset.

Five metrics were used for evaluation: accuracy, precision, recall, false positive rate and f-measure. Accuracy (ACC) evaluates the ratio of intervals correctly classified. The second metric, precision (PREC), emphasizes the detection of abnormal intervals and penalizes the normal intervals classified erroneously. Therefore, this metric complements the information provided by the accuracy, showing suitable results when the classes are not represented equally. Recall (REC) indicates the proportion of correctly classified anomalous samples from all abnormal samples. The False Positive Rate (FPR) indicates the proportion of wrong classified normal samples from all normal samples. Finally, f-measure consists of a general score given to the classifier. This score is obtained by the harmonic mean between precision and recall. The outcome of the metrics used to assess the detection scheme ranges from 0 to 1, in which the former is the worst-case scenario, and the latter represents optimal value.

In order to find the best value for w, tests were performed varying this hyperparameter and evaluated the f-measure score. The results are shown in Figure 8. The best results were obtained when the window size is 60. Therefore, hyperparameter s will be considered 60 for the experiments.

Grid search was used to define the optimal n and k hyperparameters. In the tests, the value n has varied from 30 to 140, and k from 0.1 to 0.4. The f-measure metric was used for comparing the overall performance.

All the results are depicted in Figure 9. The heatmap presents the results of each test. With this chart, it was possible to verify that from 90 detectors, the results obtained by the

[6]http://www.uel.br/grupos/orion/datasets.html

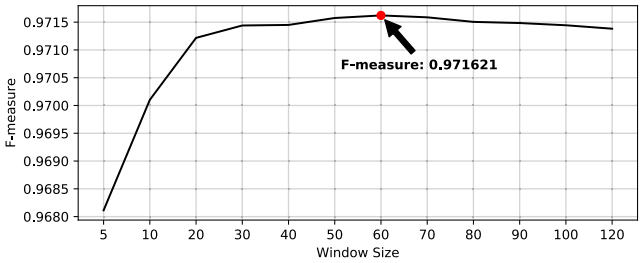

FIGURE 8. Sliding window size tests.

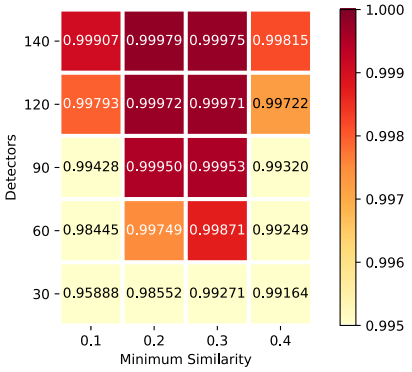

FIGURE 9. AIS hyperparameters tested.

minimum similarity 0.2 and 0.3 are very close. The best result was achieved using 140 detectors and the minimum similarity at 0.2. Thus, the k and n hyperparameters values for the next evaluations were defined as 0.2 and 140, respectively.

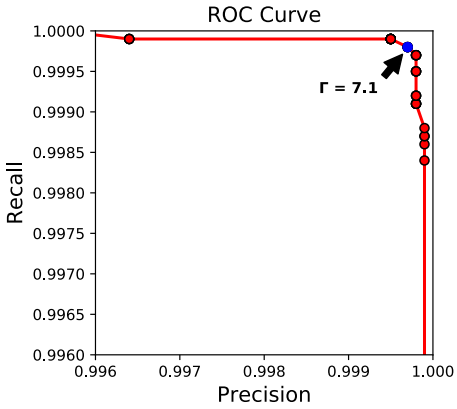

FIGURE 10. Precision-Recall curve for the estimation of Γ.

The last hyperparameter, Γ, representing the cutoff threshold, was defined using a precision-recall curve as depicted in Figure 10. This curve represents the precision and recall values of each cutoff value from 0 to 10.5, ranging from 0.1. The best cutoff value was chosen when the highest recall and accuracy values were reached. In the test performed, the best value achieved was 7.1, so this was the value assigned to Γ in the evaluation experiments.

After all the hyperparameters defined, AIS-IDS was executed and Figure 11 displays the behavior of each network

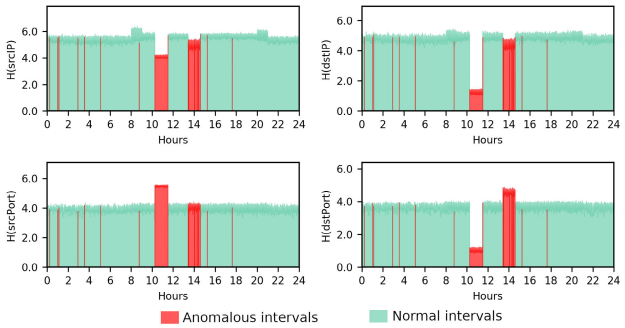

FIGURE 11. Alarms triggered on test day.

features and the intervals with attacks on the analyzed day. Intervals the AIS-IDS detected as anomalous are highlighted in red. As represented in the figure, it was possible to observe the most of the intervals in which the traffic was affected by the attacks was correctly detected. However, some periods without attacks were considered abnormal (false-positive) and also intervals with attacks were considered normal (false-negative). In general, the AIS-based IDS yielded reliable detection rates, the f-measure was superior to 99.97%.

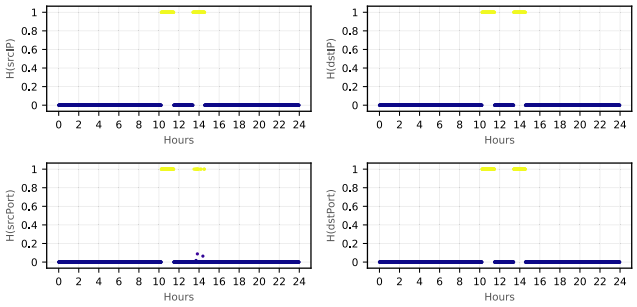

FIGURE 12. Scatterplot for fuzzy mean values on test day.

Another analysis is presented in the scatter plot, which demonstrates how each feature behaves during the day. Figure 12 represents the fuzzy values of each feature used in the test dataset. For a better view, every 60 seconds were grouped into a single point. It is possible to note at the moment where the attacks occur, the fuzzy values in all features are high, showing an apparent variation, demonstrating the attacks present in these intervals.

To evaluate the result of AIS-IDS attack detection and mitigation, the AIS detection module was configured for triggering the alarm and informing the mitigation module. Figure 13 shows in green the traffic after the mitigation and the red lines are the traffic before the mitigation process. It was possible to notice that variations in the features were eliminated after the mitigation process. This result confirms that attack flows have been correctly blocked and that normal flows have been maintained.

C. COMPARISON WITH OTHERS METHODS

In this section, the proposed AIS-based IDS is compared with some machine learning algorithms. We choose Naive Bayes (NB) [57], k-Nearest Neighbors (kNN) [58] and Random

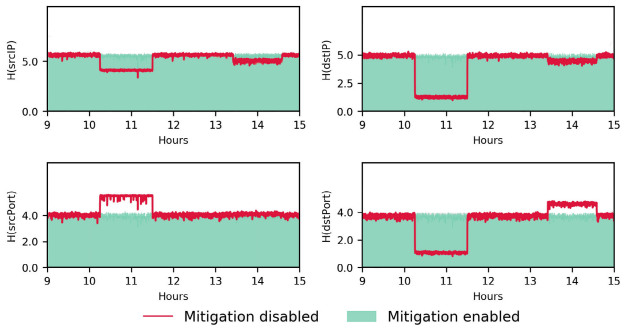

FIGURE 13. Result of mitigation module.

Forest (RF) [59] algorithms because they follow different learning paradigms with varying biases of learning. Besides, the Local Outlier Factor (LOF) [60] algorithm was selected because it is similar to the proposed solution, i.e., it does not use anomaly samples for training.

All the classifiers used in the comparison were implemented using the scikit-learn library version 0.21.3. We evaluated the result of kNN by varying the number of nearest neighbors k from 1 to 40, and the best result is when $k = 3$. Regarding the Random Forest (RF), we evaluate the number of estimators from 1 to 120, and the best results were reached with 60. For Local Outlier Factor (LOF), we evaluated the number of the neighbors from 1 to 20 and contamination values from 0.05 to 0.4. In the experiments, the neighbors value 23 and contamination 0.4 yielded the best values regarding the evaluation metrics.

NB, kNN and RF used in the comparison require previous knowledge about the attacks, hence we used day 3 (Table 2) as the training dataset. All the results presented in this section were obtained from the evaluation of the compared methods, including our IDS, using the traffic of day 2. All the results are detailed in Table 3.

TABLE 3. Comparing AIS to others approaches.

	AIS	NB	kNN	RF	LOF
FP rate	0.0014	0.0114	0	0	0.0363
Accuracy	0.9996	0.9988	0.9996	1.0000	0.9446
Precision	0.9998	0.9987	1.0000	1.0000	0.7049
Recall	0.9997	0.9999	0.9968	1.0000	0.7736
F-measure	0.9997	0.9993	0.9984	1.0000	0.7376

RF outperformed all the compared algorithms, achieving the best possible results in the simulated scenarios. AIS obtained the second-best result. Comparing the NB and kNN approaches, the results obtained have already been slightly lower when compared to AIS. However, when analyzing the LOF algorithm, which has a similar operation, the advantage is more significant.

D. COMPARISON USING PUBLIC DATASET

Our proposal was also evaluated on a public dataset, the CiCDDoS2019,[7] which contains generated flows

[7] Available at https://www.unb.ca/cic/datasets/ddos-2019.html

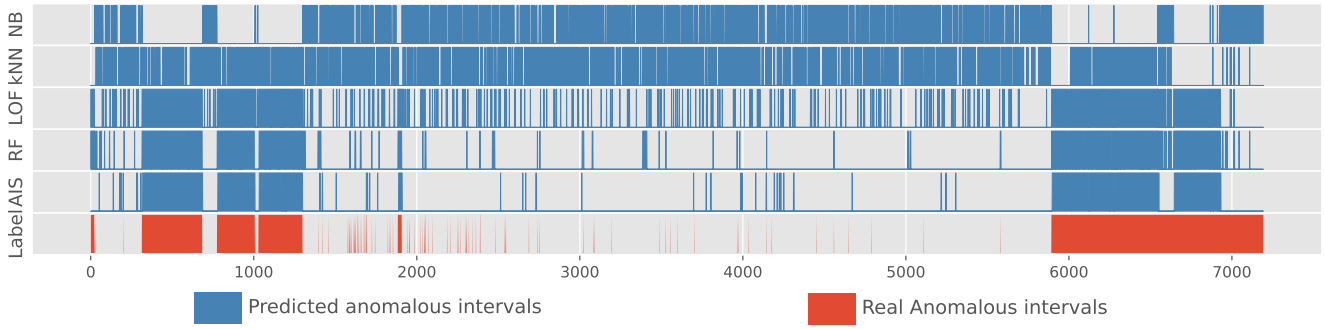

FIGURE 14. Comparative among approaches in the CiCDDoS2019 public dataset.

simulating the behavior of 25 users on the network, using various protocols, such as HTTP, FTP, and SSH. This dataset contains two days; the first is a training day, contains variations of DDoS attacks, such as DNS, TFTP, Syn, NTP, LDAP, SNMP, NetBIOS, UDP-Lag, SSDP, Web-DDoS, MSSQL, and UDP, totaling 12 types. The other one, the testing day, contains 6 DDoS attacks, which are MSSQL, LDAP, UDP-Lag, UDP, NetBIOS, and Syn.

Analyzing the dataset, it was found that anomalous and normal flows were imbalanced due to the flows of DDoS attacks, which usually are numerous to flood the target. Thus, a downsampling process was performed to balance the database relating to each type of attack. For each type of attack, five available normal flows were randomly collected. This ratio was adopted observing the frequency of attacks for maintaining the pattern network even with a fewer imbalanced distribution of common traffic.

In the approaches that require attacks in training such as NB, kNN and RF, an analysis was made, and it was possible to verify all one-second intervals have attacking flows, compromising the evaluation since all intervals would be considered attacks. In this manner, the anomalous and normal flows were regrouped to avoid this problem. After that, the same strategy of sampling of the training dataset was employed on the test dataset. For approaches that do not use sample attacks in the training dataset, such as AIS-IDS and LOF, all attacks have been removed.

TABLE 4. Comparing AIS to others approaches in the CICDDoS2019 dataset.

	AIS	NB	kNN	RF	LOF
FP rate	0.2646	0.4172	0.3043	0.0993	0.0567
Accuracy	0.8903	0.4502	0.5547	0.9098	0.8977
Precision	0.8865	0.1606	0.2841	0.7723	0.8696
Recall	0.2646	0.1691	0.2559	0.9349	0.8010
F-measure	0.9228	0.1647	0.2692	0.8459	0.8339

In the comparison of the CiCDDoS2019 dataset, the same features and algorithms were used. Comparing the metric results presented in Table 4, the RF maintains good performance, reaching an f-measure of 84.59%. Nonetheless, the AIS approach reached the best result in this scenario, reaching an f-measure of 92.28%. LOF algorithm achieved an f-measure of 83.39%. KNN and NB were the algorithms that had the worst performances, respectively.

Figure 14 shows the anomaly detections for each tested approach. Blue intervals are assigned as anomalous moments by the compared algorithm, and the red bars indicate intervals when anomalous events occurred. As can be seen, the AIS and RF approaches were the best approaches, detecting long anomalous moments and obtained less false positives when compared to other approaches. LOF was also able to detect long-term attacks; however, with many false positives compared to AIS and RF. On the other hand, NB and kNN had high amounts of false positives, impairing the quality of detection significantly.

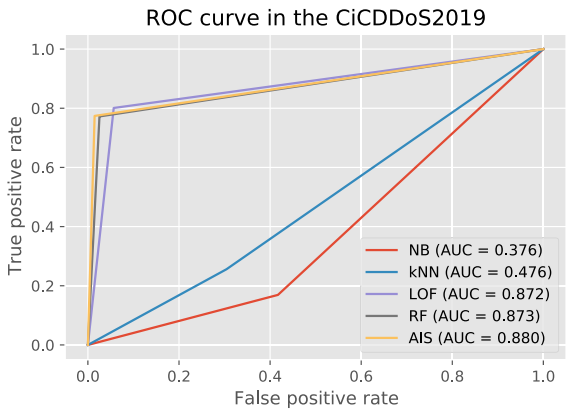

FIGURE 15. ROC curve for the CiCDDoS2019 public dataset.

Figure 15 shows the Operating Characteristic Curve (ROC) of each approach used in the comparison and their respective Area Under the Curve (AUC) value. AIS was the approach that obtained the best result, RF achieved a result very close to our approach, while LOF had a slightly lower result. The NB and kNN approaches had the worst results.

V. CONCLUSION

In this paper, we proposed an AIS-IDS to detect and mitigate anomalies in software-defined networks. Our proposal uses

four features of IP flows collected every second to create the detectors. Also, it was created a sliding window to perform detection in near real-time.

The experiments were performed in an emulated environment, show an f-measure value higher than 99.9%, showing the proposed IDS is reliable to protect a network. Also, using a public dataset of attacks, the value was higher than 92%, demonstrating the ability to detect the most different attacks without requiring prior information about them. Also, the mitigation module was able to drop anomalous packets and block the attacks. The AIS-IDS has proved to be an effective approach to detecting several types of flooding attacks and portscan. The approach recognizes anomalies only with network behavior and thus becoming an alternative to today's networks, which are suffering from new attacks every day and which traditional techniques do not end up keeping pace with these changes. In future work, we will focus on improving the creation of detectors, using other techniques in the generation stage. Also, we aim to distinguish between flash crowd and DDoS attacks, and pursue mitigation strategies to flash crowd as load balancing between servers. Finally, we intend to evaluate our proposal in real network environments.

REFERENCES

[1] G. Fernandes, J. J. P. C. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi, and M. L. Proena, "A comprehensive survey on network anomaly detection," *Telecommun. Syst.*, vol. 70, no. 3, pp. 447–489, Mar. 2019.

[2] R. Sahay, W. Meng, and C. D. Jensen, "The application of software defined networking on securing computer networks: A survey," *J. Netw. Comput. Appl.*, vol. 131, pp. 89–108, Apr. 2019.

[3] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Comput. Netw.*, vol. 174, Jun. 2020, Art. no. 107247.

[4] Y. Zhong, W. Chen, Z. Wang, Y. Chen, K. Wang, Y. Li, X. Yin, X. Shi, J. Yang, and K. Li, "HELAD: A novel network anomaly detection model based on heterogeneous ensemble learning," *Comput. Netw.*, vol. 169, Mar. 2020, Art. no. 107049.

[5] E. H. M. Pena, S. Barbon, J. J. P. C. Rodrigues, and M. L. Proenca, "Anomaly detection using digital signature of network segment with adaptive ARIMA model and paraconsistent logic," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2014, pp. 1–6.

[6] F. Hachmi, K. Boujenfa, and M. Limam, "Enhancing the accuracy of intrusion detection systems by reducing the rates of false positives and false negatives through multi-objective optimization," *J. Netw. Syst. Manage.*, vol. 27, no. 1, pp. 93–120, Jan. 2019.

[7] S. I. Suliman, M. S. Abd Shukor, M. Kassim, R. Mohamad, and S. Shahbudin, "Network intrusion detection system using artificial immune system (AIS)," in *Proc. 3rd Int. Conf. Comput. Commun. Syst. (ICCCS)*, Apr. 2018, pp. 178–182.

[8] S. Hosseinzadeh, M. Amirmazlaghani, and M. Shajari, "An aggregated statistical approach for network flood detection using gamma-normal mixture modeling," *Comput. Commun.*, vol. 152, pp. 137–148, Feb. 2020.

[9] O. Salman, I. Elhajj, A. Chehab, and A. Kayssi, "IoT survey: An SDN and fog computing perspective," *Comput. Netw.*, vol. 143, pp. 221–246, Oct. 2018.

[10] Y. Xu, H. Sun, F. Xiang, and Z. Sun, "Efficient ddos detection based on k-fknn in software defined networks," *IEEE Access*, vol. 7, pp. 160536–160545, 2019.

[11] L. Barki, A. Shidling, N. Meti, D. G. Narayan, and M. M. Mulla, "Detection of distributed denial of service attacks in software defined networks," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2016, pp. 2576–2581.

[12] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. P. C. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics," *Future Gener. Comput. Syst.*, vol. 89, pp. 685–697, Dec. 2018.

[13] M. Alsaeedi, M. M. Mohamad, and A. A. Al-Roubaiey, "Toward adaptive and scalable openflow-sdn flow control: A survey," *IEEE Access*, vol. 7, pp. 107346–107379, 2019.

[14] Y. Yu, X. Li, X. Leng, L. Song, K. Bu, Y. Chen, J. Yang, L. Zhang, K. Cheng, and X. Xiao, "Fault management in software-defined networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 349–392, 1st Quart., 2018.

[15] N. Meti, D. G. Narayan, and V. P. Baligar, "Detection of distributed denial of service attacks using machine learning algorithms in software defined networks," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2017, pp. 1366–1371.

[16] S. M. Mousavi and M. St-Hilaire, "Early detection of DDoS attacks against software defined network controllers," *J. Netw. Syst. Manage.*, vol. 26, no. 3, pp. 573–591, Jul. 2018.

[17] W. L. da Costa Cordeiro, J. A. Marques, and L. P. Gaspary, "Data plane programmability beyond OpenFlow: Opportunities and challenges for network and service operations and management," *J. Netw. Syst. Manage.*, vol. 25, no. 4, pp. 784–818, Oct. 2017.

[18] 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.

[19] M. V. De Assis, M. P. Novaes, C. B. Zerbini, L. F. Carvalho, T. Abráao, and M. L. Proença, "Fast defense system against attacks in software defined networks," *IEEE Access*, vol. 6, pp. 69620–69639, 2018.

[20] L. Jin, J. Chen, and X. Zhang, "An outlier fuzzy detection method using fuzzy set theory," *IEEE Access*, vol. 7, pp. 59321–59332, 2019.

[21] M. Fahim and A. Sillitti, "Anomaly detection, analysis and prediction techniques in iot environment: A systematic literature review," *IEEE Access*, vol. 7, pp. 81664–81681, 2019.

[22] S. Garg, K. Kaur, N. Kumar, and J. J. P. C. Rodrigues, "Hybrid deep-learning-based anomaly detection scheme for suspicious flow detection in SDN: A social multimedia perspective," *IEEE Trans. Multimedia*, vol. 21, no. 3, pp. 566–578, Mar. 2019.

[23] M. L. Proença, C. Coppelmans, M. Bottoli, A. Alberti, and L. S. Mendes, "The hurst parameter for digital signature of network segment," in *Telecommunications and Networking—ICT* (Lecture Notes in Computer Science), vol. 3124, J. N. de Souza, P. Dini, and P. Lorenz, Eds. Berlin, Germany: Springer, 2004.

[24] Y.-C. Lai, K.-Z. Zhou, S.-R. Lin, and N.-W. Lo, "F1ow-based anomaly detection using multilayer perceptron in software defined networks," in *Proc. 42nd Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, Oct. 2019, pp. 1154–1158.

[25] J. M. Vidal, A. L. S. Orozco, and L. J. G. Villalba, "Adaptive artificial immune networks for mitigating DoS flooding attacks," *Swarm Evol. Comput.*, vol. 38, pp. 94–108, Feb. 2018.

[26] S. Haider, A. Akhunzada, G. Ahmed, and M. Raza, "Deep learning based ensemble convolutional neural network solution for distributed denial of service detection in SDNs," in *Proc. UK/ China Emerg. Technol. (UCET)*, Aug. 2019, pp. 1–4.

[27] A. S. A. Aziz, S. E.-O. Hanafi, and A. E. Hassanien, "Comparison of classification techniques applied for network intrusion detection and classification," *J. Appl. Log.*, vol. 24, pp. 109–118, Nov. 2017.

[28] Y. Qin, J. Wei, and W. Yang, "Deep learning based anomaly detection scheme in software-defined networking," in *Proc. 20th Asia–Pacific Netw. Operations Manage. Symp. (APNOMS)*, Sep. 2019, pp. 1–4.

[29] D. Arivudainambi, V. K. K. A, and S. Sibi Chakkaravarthy, "LION IDS: A meta-heuristics approach to detect DDoS attacks against software-defined networks," *Neural Comput. Appl.*, vol. 31, no. 5, pp. 1491–1501, May 2019.

[30] A. Mansour, M. Azab, M. R. M. Rizk, and M. Abdelazim, "Biologically-inspired SDN-based intrusion detection and prevention mechanism for heterogeneous IoT networks," in *Proc. IEEE 9th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Nov. 2018, pp. 1120–1125.

[31] P. T. Duy, D. T. T. Hien, and V.-H. Pham, "A role-based statistical mechanism for DDoS attack detection in SDN," in *Proc. 5th NAFOSTED Conf. Inf. Comput. Sci. (NICS)*, Nov. 2018, pp. 177–182.

[32] H. Rathore, A. Samant, and M. Guizani, "A bio-inspired framework to mitigate DoS attacks in software defined networking," in *Proc. 10th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS)*, Jun. 2019, pp. 1–5.

[33] D. Hooks, X. Yuan, K. Roy, A. Esterline, and J. Hernandez, "Applying artificial immune system for intrusion detection," in *Proc. IEEE 4th Int. Conf. Big Data Comput. Service Appl. (BigDataService)*, Mar. 2018, pp. 287–292.

[34] J. Shen and J. Wang, "Network intrusion detection by artificial immune system," in *Proc. IECON - 37th Annu. Conf. IEEE Ind. Electron. Soc.*, Nov. 2011, pp. 4716–4720.
[35] M. Tabatabaefar, M. Miriestahbanati, and J.-C. Gregoire, "Network intrusion detection through artificial immune system," in *Proc. Annu. IEEE Int. Syst. Conf. (SysCon)*, Apr. 2017, pp. 1–6.
[36] Q. Zhou and D. P. Pezaros, "BIDS: Bio-inspired, collaborative intrusion detection for software defined networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[37] 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.
[38] L. F. Carvalho, T. Abráo, L. D. S. Mendes, and M. L. Proenáa, "An ecosystem for anomaly detection and mitigation in software-defined networking," *Expert Syst. Appl.*, vol. 104, pp. 121–133, Aug. 2018.
[39] M. V. O. de Assis, J. J. P. C. Rodrigues, and M. L. P. Junior, "A novel anomaly detection system based on seven-dimensional flow analysis," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 735–740.
[40] C. E. Shannon, "Ce Shannon, acm sigmobile mobile comput. commun. rev. 5, 3 (2001)," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, p. 3, Oct. 2001.
[41] S. Krishnaveni, P. Vigneshwar, S. Kishore, B. Jothi, and S. Sivamohan, "Anomaly-based intrusion detection system using support vector machine," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems* (Advances in Intelligent Systems and Computing), vol. 1056, S. Dash, C. Lakshmi, S. Das, and B. Panigrahi, Eds. Singapore: Springer, 2020.
[42] C. C. B. Fioravanti, T. M. Centeno, and M. R. De Biase Da Silva Delgado, "A deep artificial immune system to detect weld defects in dwdi radiographic images of petroleum pipes," *IEEE Access*, vol. 7, pp. 180947–180964, 2019.
[43] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, "Self-nonself discrimination in a computer," in *Proc. IEEE Comput. Soc. Symp. Res. Secur. Privacy*, 1994, pp. 202–212.
[44] Z. Fan, C. Wen, L. Tao, C. Xiaochun, and P. Haipeng, "An antigen space triangulation coverage based real-value negative selection algorithm," *IEEE Access*, vol. 7, pp. 51886–51898, 2019.
[45] K. Song, P. Kim, V. Tyagi, and S. Rajasekaran, "Artificial immune system (AIS) based intrusion detection system (IDS) for smart grid advanced metering infrastructure (AMI) networks," Virginia Polytech. Inst. State Univ., Blacksburg, VA, USA, 2018.
[46] C. Aggarwal, "Outlier analysis," in *Data Mining*. Cham, Switzerland: Springer, 2015.
[47] A. Brahma and S. Panigrahi, "Role of soft outlier analysis in database intrusion detection," in *Advanced Computing and Intelligent Engineering*. Springer, 2020, pp. 479–489.
[48] M. Masdari and H. Khezri, "A survey and taxonomy of the fuzzy signature-based intrusion detection systems," *Appl. Soft Comput.*, vol. 92, Jul. 2020, Art. no. 106301.
[49] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abráo, and M. L. Proenéa, "Network anomaly detection system using genetic algorithm and fuzzy logic," *Expert Syst. Appl.*, vol. 92, pp. 390–402, Feb. 2018.
[50] L. H. Tsoukalas and R. E. Uhrig, *Fuzzy and Neural Approaches in Engineering*. Hoboken, NJ, USA: Wiley, 1996.
[51] Y. Wang, "A multinomial logistic regression modeling approach for anomaly intrusion detection," *Comput. Secur.*, vol. 24, no. 8, pp. 662–674, Nov. 2005.
[52] C. Khammassi and S. Krichen, "A NSGA2-LR wrapper approach for feature selection in network intrusion detection," *Comput. Netw.*, vol. 172, May 2020, Art. no. 107183.
[53] M. L. Proenáa, G. Fernandes, L. F. Carvalho, M. V. O. de Assis, and J. J. P. C. Rodrigues, "Digital signature to help network management using flow analysis," *Int. J. Netw. Manage.*, vol. 26, no. 2, pp. 76–94, Mar. 2016.
[54] Z. Li, X. Yu, D. Wang, Y. Liu, H. Yin, and S. He, "SuperEye: A distributed port scanning system," in *Artificial Intelligence and Security* (Lecture Notes in Computer Science), vol. 11635, X. Sun, Z. Pan, and E. Bertino, Eds. Cham, Switzerland: Springer, 2019.
[55] M. A. Khan, "A survey of security issues for cloud computing," *J. Netw. Comput. Appl.*, vol. 71, pp. 11–29, Aug. 2016.
[56] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proc. 9th ACM SIGCOMM Workshop Hot Topics Netw.*, 2010, p. 19.
[57] D. D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval," in *Machine Learning: ECML* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence), vol. 1398, C. Nédellec and C. Rouveirol, Eds. Berlin, Germany: Springer, 1998.
[58] V. Hautamaki, I. Karkkainen, and P. Franti, "Outlier detection using k-nearest neighbour graph," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, 2004, pp. 430–433.
[59] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[60] A. L. M. Chiu and A. Wai-chee Fu, "Enhancements on local outlier detection," in *Proc. 7th Int. Database Eng. Appl. Symp.*, 2003, pp. 298–307.

GUSTAVO F. SCARANTI received the B.S. degree in computer engineering from the University of Northern Paraná (UNOPAR), in 2016. He is currently pursuing the master's degree with the Computer Department, State University of Londrina, Brazil. He is a member of the Research Group Computer Networks and Data Communication. His main research interests include artificial immune systems, 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 has experience in computer science with an emphasis in computer networks and is part of the Research Group Computer Networks and Data Communication. His main research interests include management and security of computer networks.

SYLVIO BARBON, Jr. received the degree in computer science and the master's degree in computational physics from the University of São Paulo, in 2005 and 2007, respectively, and the degree in computer engineering and the Ph.D. degree in applied computational physics from IFSC, in 2008, and 2011, respectively. In 2017, he was a Visiting Professor with the Universitá Degli Studi di Milano, Italy, and developed a postdoctoral project at the Universitá di Modena e Reggio Emilia, Italy. He is currently an Adjunct Professor with the Computer Science Course, Londrina State University, and a Research Productivity Fellow with the Araucária Foundation. His research interests include intelligent signal processing, machine learning, pattern recognition, and text mining.

MARIO LEMES PROENÇA, Jr. received the M.Sc. degree in computer science from the Informatics Institute, Federal University of Rio Grande do Sul (UFRGS), in 1998, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas (UNICAMP), in 2005. He is currently an Associate Professor and the Leader of the Research Group that studies computer networks with the Computer Science Department, State University of Londrina (UEL), Brazil. He has authored or coauthored over 100 papers in refereed international journals and conferences, books chapters, and 1 software register patent. His research interests include computer networks, network operations, management and security, and IT governance.