Smart Detection-IoT: A DDoS Sensor System for Internet of Things

Frederico Augusto Fernandes Silveira[1], Francisco Lima-Filho[1], Felipe Sampaio Dantas Silva[1]
Agostinho de Medeiros Brito Junior[1], Luiz Felipe Silveira[1]
[1]Federal University of Rio Grande do Norte, Natal, Rio Grande do Norte, Brazil
silveira.frederico@gmail.com

***Abstract*—The number of Distributed Denial of Service (DDoS) attacks using IoT devices has increased in recent years. Reasons for this growth include the security limitations of IoT devices, the number of devices, and their geographic distribution. Developing mechanisms to detect and mitigate DDoS attacks in this scenario is a current challenge in the area of network security. In the literature review, it is seen that recent academic works still tries to find the best way to combat this type of threat, with proposals that need to be tested against modern datasets that contain a diversity of modern denial of service attacks. This work proposes a detection module for an IoT controller that uses Machine Learning (ML) techniques to classify network traffic. The system was designed in the Software-Defined Networks (SDN) context and evaluated on an emulated platform using three actual and well-know datasets present in the literature. The results, at a sampling rate (S_R) of 20% of network traffic, show a high precision (P_R), above 93%, a low false alarm rate (FA_R), and detection rate (D_R) of attacks above 96%, using a low profile emulated device.**

***Keywords*—IoT, DoS Detection, Machine Learning, Network Security**

I. Introduction

The Internet has established itself as a platform that drives economic and technological development in every country in the world. In the information era, the network is seen as a natural destination for offering products and services due to its worldwide reach.

The number of new devices connected to the world wide web has increased substantially in recent years. High-speed mobile communication helped this phenomenon through smartphones and tablets, joined with the popularization of IoT devices. At this juncture, it is estimated that 50 billion of IoT devices will be connected to the Internet by 2020 [1], with an upward trend in the number of Mobile-to-Mobile (M2M) [2] because of the growing demand for services that operate on IoT networks, especially in the context of smart cities. Besides, the emergence of wearable devices will impact this growth, with the prospect of about 1.1 billion such devices connected to the network by 2022 [2].

While this new environment has undoubtedly introduced advances to online services, it has also brought concerns and new challenges related to the safe operation of the Internet. One of the main issues is related to the weak security of IoT devices, which has facilitated the use of these devices as vectors of cyberattacks, mainly for Denial of Service (DoS) attacks [1]. Distributed Denial of Service (DDoS) attacks using IoT devices has been catching the attention of cybersecurity experts since 2016 [1]. That year, four hundred thousand devices, including cameras and wireless routers, were infected with the Mirai malware and formed a massive botnet that paralyzed the Internet with an orchestrated DDoS attack, breaking the traffic record associated with a single event. More recently, new and more dangerous malware, such as Hajime and IoT_Reaper [1], have been identified on the network, indicating an even greater tendency to use IoT technology as a vector for DDoS attacks.

978-1-7281-7539-3/20/$31.00 ©2020 IEEE

Although the academic community and researchers from specialist companies have been working on the topic, there is still no consensus on the solution to the problem of DDoS attacks, especially when the source of the attacks is IoT equipment. In general, although the solutions proposed by the academy are scientifically secure, they do not present practical requirements for deployment on the Internet [3]. In contrast, commercial solutions are not effective in the context of IoT [3], as the attack will eventually lead to resource depletion of corporate network input equipment. Thus, the investigation of DDoS attack detection and mitigation techniques in the context of IoT networks is currently of great interest in the area of network security, whose solutions should significantly impact the availability of Internet services.

Machine learning (ML) is quickly expanding in many fields, such as science, technology, marketing, education, healthcare, and many other fields. Recent anomaly detection research has shown the promise of machine learning for recognizing malicious Internet traffic [4]. Machine learning technique in cybersecurity is helpful by recommending the proper decision for analysis and even doing the proper action automatically. However, little effort has been made to valuate ML models with features geared explicitly towards IoT device networks or IoT attack traffic.

A. Proposal

This work proposes an integrated detection mechanism against DDoS attacks for the IoT network environment, called Smart Detection-IoT (SD-IoT) system. The system architecture is designed to detect early DDoS attacks on the source network. The proposed detection approach works using a sensor installed on the IoT network access point, which classifies online traffic using a Machine Learning (ML) - based strategy. The proposed approach is compatible with today's Internet infrastructure and does not require software or hardware upgrades, which makes the deployment feasible. User data privacy is guaranteed at all stages of system operation since data fields are not accessed.

B. Contribution

In essence, this study presents the modeling, coding, and validation of an online detection system of DDoS attacks for an IoT network scenario. This system, called Smart Detection-IoT, classifies online random samples of IoT network traffic, as DoS attacks or regular traffic, conserving information privacy. Also, the properties of network traffic in this scenario are explored. As a result, a new signature database is created. Furthermore, it is also presented a practical online processing and validation strategy for raw network traffic data.

C. Paper Organization

The remainder of this paper is structured as follows: Section II conducts a study of related works to underline the our contribution. Section III introduces the Smart Detection-IoT system, exposing its architecture and main features. Section IV evaluates the proposed system and provides a discussion about the results. Finally, Section V examines the outcomes.

II. Literature Review

Recent works with the focus on securing against distributed denial of service attacks in the context of IoT networks is presented in this section, detailing different detection and mitigation techniques to place this work proposal in the state-of-the-art.

The work proposed by Yin et al. [5] provides a framework called SD-IoT (Software-Defined - Internet of Things) and an attack detection algorithm that analyzes the similarity of input packet rate vectors in SD-IoT switch ports border. The proposed system calculates the similarity ($\rho_{x,y}$) of the analyzed vector and compares it to a threshold value (η_U). If $\eta_U \leq \rho_{x,y} \leq 1$, SD-IoT switch will be under DDoS attack. The experiments were performed using the Mininet emulation tool [6]. However, the authors place devices only as targets for DDoS attacks, not as potential infected agents and originators of the attacks.

On the other hand, the work by Doshi et al. [7] presented a behavioral analysis of IoT network variables such as bandwidth, packet interval, protocols, packet size, and destination address. Also, the authors performed an analysis of several machine learning algorithms and validated the dataset generated by the authors. They ran tests with five algorithms: KNN (K-Nearest Neighbors), LSVM (Support Vector Machine with the Linear kernel), DT (Decision Tree), RF (Random Forest), NN (Neural Network). In this study, there is no evaluation with well-known datasets from literature or online detection of network traffic.

In [8], it was used fog computing in the SDN environment to detect DDoS attacks from IoT devices. The authors proposed the Edge-Centric Software-Defined IoT Defense (ECESID), using SDN switches to defend DDoS attacks and enforcing traffic control rules near the source of the attack. The authors use a combination of two attack detection algorithms: Threshold Random Walk with Credit-Based Rate Limiting (TRW-CB) and Rate Limiting (RL). The authors used Mininet to emulate IoT networks and connecting devices, using the Mirai malware to infect IoT devices, analyzing the behavior of infected devices, and mitigating attacks. At that work, the detection of infected devices occurs at the stage where these devices attempt to infect other healthy devices. This strategy can present a deficiency in the detection process, where infected devices are hibernating while waiting for the attack to take place or while performing DDoS attacks.

In work presented by Hasan [9], a set of attack data and regular traffic in the literature was used. They also engineered the existing data, separating and clearing the variables. After this step, tests were performed with different machine learning algorithms, such as Logistic Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF) and Artificial Neural Network (ANN). The analysis was performed offline, that is, no detection or mitigation system was implemented, and only statistical metrics were obtained regarding the classification of the existing dataset.

Another approach [10] presented a DDoS attack detection system using a network entropy analysis. The entropy variation of the destination IP address is measured using the Shannon equation. In the case of DDoS attacks, the entropy values fall compared to the values obtained in a network with regular traffic. Tests were performed by the authors using an SDN environment in Raspberry PI, with a POX controller to mitigate connections detected as an attack. In their work, false alarm rates, detection, and accuracy were not shown, and there are no comparisons with datasets present in the literature.

In [11], the authors proposed a DDoS attack detection and mitigation system that integrates an Intrusion Detection System (IDS) into the client-side SDN architecture for home or organizational network scenarios. The system operates through loop control between three essential architectural components: the network, the IDS, and the controller. IDS analyzes all exchanged traffic on the network, detecting ongoing DDoS attacks. The controller, when notified by IDS, transfers to the network devices some new flow rules to restore regular operation as quickly as possible. The authors use Snort as a detection solution, mirroring all port traffic, and deeply inspecting incoming packets. This type of approach proved to be efficient in the tests performed; however, by performing deep inspection, the solution becomes vulnerable in volumetric attacks.

The studies presented above were grouped considering their main characteristics, such as detection methods using (ML), Diversity of Attacks (DoA) addressed, Traffic Sample Analysis (TSA), Online Validation (OV) process, and Embedded Performed (EP) experiments. Table I summarizes these approaches in the literature, putting into perspective the proposal presented in this paper.

Given the literature review, we can see that recent works still tries to find the best way to combat this type of threat. Some works did not use a variety of attacks in their experiments, and the efficiency of the proposed work in the face of the specific

TABLE I
COMPARATIVE SUMMARY OF WORKS IN LITERATURE.

Proposal	**DoA**	**ML**	**OV**	**EP**	**TSA**
[5]	✗	✗	✓	✗	✗
[7]	✗	✓	✗	✗	✗
[8]	✓	✗	✓	✗	✗
[9]	✓	✓	✗	✗	✗
[10]	-	✗	✓	✗	✗
[11]	✗	✗	✓	✗	✗
SD-IoT	✓	✓	✓	✓	✓

threat cannot be proven. Another point to note, regarding the use of machine learning techniques, only three (3) of the seven (7) proposals use it. Regarding the online validation of the proposal, it is noted that the vast majority adopted this procedure, it is essential to verify the proposed solution in a real traffic scenario. However, only Smart Detection-IoT proposes to ship the solution in a wireless controller and position it as close as possible to the IoT network, serving as a means of accessing these devices. Another essential issue to note is that the SD-IoT works by analyzing only traffic samples to classify patterns, differently from the other works presented. Thus, it assures data privacy and efficiency in the use of computational resources. This work also shows that the use of ML in classifying these DDoS attack patterns has proven to be efficient and accurate.

III. PROPOSAL ARCHITECTURE OVERVIEW

The proposed system was designed to run embedded in a wireless access point. Internally, Smart Detection-IoT is incorporated into an SDN controller and uses a signature-based machine learning algorithm to identify DDoS attacks. Inferences are made based on samples of network traffic that are collected and forwarded to the detection system by an OpenFlow [12] switch. Fig. 1 presents an overview of the proposed architecture.

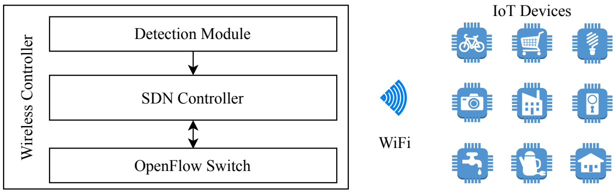

Fig. 1. Smart Detection-IoT Architecture

Fig. 2 illustrates the operation of Smart Detection-IoT. Traffic from IoT devices is sampled by the Open vSwitch (OVS) using the industry-standard sFlow protocol [13] that captures traffic samples down to the transport layer level. In possession of the samples, the system uses machine learning to detect denial of service attack patterns and then generates an attack notification. In this paper, it was evaluated three classification algorithms of machine learning, Logistic Regression (LR), Random Forest (RF), and Extreme Gradient Boost (XGB) [14] in the detection system core.

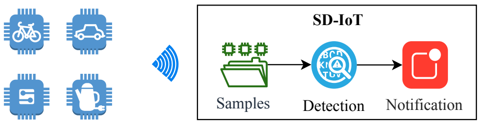

Fig. 2. Smart Detection-IoT Operation

A. Detection

The detection module processes network traffic samples obtained from network devices by using standard traffic sampling protocols, such as sFlow and Netflow.

Unidentified samples are obtained and arranged into flow queues in the input buffer. Whenever the queue length is greater than or equal to a set value (T_{max}), they are classified and assigned a label, as illustrated in Fig. 3.

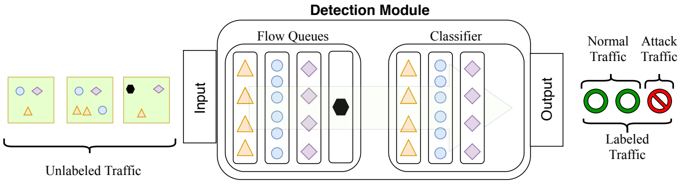

Fig. 3. Traffic Classification Scheme Overview.

Traffic samples are received and stored in a flow queue through each round of the detection process. For each new flow, a single identifier is calculated based on source IP address, destination IP address, source port, destination port, and IP protocol. If a new flow arrives, and no other flow queue is already stored with the same ID, a new flow queue is created in a shared memory buffer. Otherwise, if there is already a flow queue registered with the equivalent ID as the one calculated, the new flow data will be united with the existing flow queue data. After the data unification, if the queue size is greater than or identical to a reference value pre-settled by the user, the flow queue is classified. A notification is created if the flow queue is labeled as an attack.

B. Data and Model

The process of extracting, transforming, labeling signature database instances, and selecting variables are essential steps in building optimized supervised machine learning models. Fig. 4 illustrates how these steps were performed in this paper.

In the first step, the characteristics of regular network traffic were extracted directly from the network capture file available in the ISCXIDS2012 dataset [15]. Similarly, the DoS traffic behavior was estimated from malicious traffic capture, likewise to the process done in our previous work [16]. However, it was made some adjustments to take into account the relation between the size of the flow queue and the sampling of network traffic, generating a new signature database with 121,551 instances labeled as regular traffic and 121,551 instances labeled as malicious traffic. The header variables of the packets from the network and transport layers of the data flow set were analyzed. It was used IP packet size, source IP addresses, destination IP addresses, source port, destination port, TCP flags, and transport layer protocol to calculate 33 descriptors variables, listed in table II, by using the following statistical measures: variance (var), standard deviation (std), median, mean, entropy, rate of change (rte), coefficient of variation (cv), and quantile coefficient (cvq).

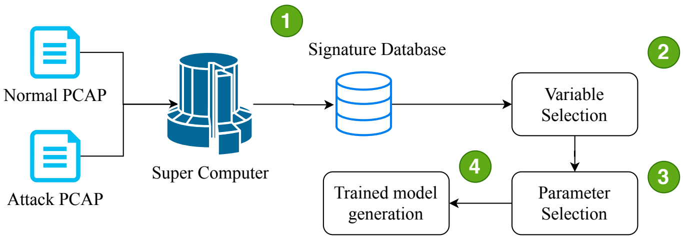

Fig. 4. Smart Detection-IoT data, variables, and model analysis process steps.

In step 2, a variable selection was performed using the Recursive Feature Elimination and Cross-Validated (RFECV) [14]. This method used three well-established machine learning techniques in the literature: i) Random Forest (RF), ii) Logistic Regression (LR), and iii) Extreme Gradient Boost (XGB) algorithms. Table II shows the variables selected for each algorithm. A more detailed description of each variable can be obtained in [16].

TABLE II
VARIABLES OPTIMIZED FOR RF, LR AND XGB MODELS.

#	Variables	RF	LR	XGB
01	ip_proto	-	-	✓
02	ip_len_mean	✓	-	✓
03	ip_len_median	✓	-	✓
04	ip_len_var	✓	-	✓
05	ip_len_std	✓	✓	-
06	ip_len_entropy	✓	✓	-
07	ip_len_cv	✓	✓	✓
08	ip_len_cvq	✓	✓	✓
09	ip_len_rte	✓	✓	-
10	sport_mean	✓	-	✓
11	sport_median	✓	-	✓
12	sport_var	✓	-	✓
13	sport_std	✓	-	
14	sport_entropy	✓	-	✓
15	sport_cv	✓	-	✓
16	sport_cvq	✓	-	✓
17	sport_rte	✓	✓	✓
18	dport_mean	✓	-	✓
19	dport_median	✓	-	-
20	dport_var	-	-	✓
21	dport_std	-	-	-
22	dport_entropy	-	-	-
23	dport_cv	-	✓	-
24	dport_cvq	-	✓	-
25	dport_rte	-	✓	-
26	tcp_flags_mean	✓	✓	✓
27	tcp_flags_median	✓	✓	-
28	tcp_flags_var	✓	✓	✓
29	tcp_flags_std	✓	✓	-
30	tcp_flags_entropy	✓	-	-
31	tcp_flags_cv	✓	✓	-
32	tcp_flags_cvq	✓	✓	✓
33	tcp_flags_rte	✓	-	✓

Next, in step 3, a fine-tune of model's parameters was performed using a grid search strategy, as proposed in [17].

In the last step, the binary file with a trained model was built and exported to the embedded system. Thus, whenever the system starts, it only loads the compiled trained model, saving considerable computational effort.

IV. Experiments and Results

The experiments were performed in a controlled laboratory environment using recent literature datasets. The proposed system was executed in a Raspberry PI 3 model B environment using the QEMU [18] platform, an emulator that simulates various types of CPUs such as x86, PowerPC, ARM, and Sparc.

The online evaluation method was used to assess the performance of the proposed system. The raw network traffic capture files were processed under similar conditions to reproduce their original conditions. The performance of the Smart Detection-IoT system was compared with other works in the literature with a similar methodology approach, as the use of ML, datasets tested, and evaluation metrics.

A. Datasets

In this work, the datasets CIC-DoS [19], CICIDS2017 [20] and customized [16] were used since they include modern threats and DoS techniques. The CIC-DoS dataset has 24 hours of network traffic, with a total size of 4.6 GB of data. It is composed of 26 different attacks, such as Goldeneye, hulk, ddossim, and others modern attacks [16], being 13 of high volume, and 13 of low volume. CICIDS2017 has five attacks, two low volume, two high volume, and one exploiting the Heartbleed vulnerability over the 8 hours. The customized dataset has 48 denials of service attacks (19 low volume, and 29 high volume) performed over 24 hours. This dataset was published in [16]. It is important to note that the attack traffic present in the dataset was not used in the signature base utilized in the model training.

B. Evaluation metrics

System performance was evaluated using the Precision (P_R), and F-Measure (F1) metrics present in the literature [21].P_R measures the ability to avoid false positive. F1 is a harmonic average between P_R and Recall (R_e), which measures system sensitivity. In this context, the number of times that the classifier hits the target class is called True Positive (TP). On the other hand, True Negative (TN) occurs when the classifier correctly recognizes examples that do not belong to the target class. Those examples incorrectly assigned to the target class is called False Positive (FP), while examples incorrectly assigned as not belonging to the target class constitute False Negative (FN). These metrics have computed by the following expressions:

$$P_R = \frac{TP}{TP + FP} \quad (1) \qquad R_e = \frac{TP}{TP + FN} \quad (2)$$

$$F1 = 2 \times \frac{P_R \times R_e}{P_R + R_e} \quad (3)$$

Besides, the Detection Rate (D_R) and False Alarm Rate (FA_R) metrics were also used. The D_R is the ratio between the number of Attacks Detected (A_D) by the system and the Total of Attacks (T_A) performed. FA_R is the ratio between FP and the sum of FP and TN. These metrics have computed by the following expressions:

$$D_R = \frac{A_D}{T_A} \quad (4) \qquad FA_R = \frac{FP}{FP+TN} \quad (5)$$

The D_R and FA_R calculations assume that only malicious traffic was sent from the attacker to the victim at the time of the attack.

C. Experimental setup

The experimental setup consists of a network traffic processor, a virtual switch containing a packet sampler, and the Smart Detection-IoT system. TcpReplay [22] software was used to process PCAP files whose traffic was forwarded to the Open vSwitch (OVS) [23], where packets were sampled at a rate of 20% using a built-in sFlow agent OVS. These samples were received and analyzed by the Smart Detection-IoT system, configured with $T_{max} = 50$. Several previous tests were performed to choose the most balanced parameters to be used. In scenarios where the sample rate is too low, and the T_{max} is too large, for example, traffic samples are discarded before processing by the classifier. On the other hand, if T_{max} is too small, the FA_R increases because the classifier has little data to analyze. Fig. 5 summarizes the operation of the experimental setup.

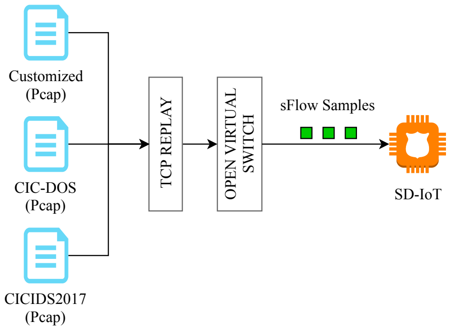

Fig. 5. Smart Detection-IoT online validation scheme.

D. Results

The proposed system was evaluated using three different classification algorithms, and three recent datasets. It is important to observe that the data (Datasets) used for the classification tests are not the same as those used in training the model. The table III presents the result of each algorithm grouped by dataset. Overall, the tested MLAs achieved relevant results in terms of attack Detection Rate (D_R), Precision (P_R), F1-Score (F1), False Alarm Rate (FA_R).

Results show the Logistic Regression algorithm performed promisingly, detecting 100% of attacks for the customized dataset, 96.1% of attacks for the CID-DOS dataset, and 80% of them in CICIDS2017. In terms of precision, considering the three datasets, it was observed a rate from 92.8% to 98.6%. For F1, we obtained a rate between 92.8% and 99.2%, besides a low false alarm rate, with values between 2.3% and 8.3%.When compared to the previous work [16], we can see an improvement of 3% to 4% in the detection rate, and a worsening in precision and F1 when we look at the customized and CID-DOS datasets.

TABLE III
Detection system performance.

Dataset	MLA	$\mathbf{D_R}$	$\mathbf{FA_R}$	$\mathbf{P_R}$	$\mathbf{F_1}$
CIC-DoS	LR	0.961	0.052	0.928	0.928
	RF	0.807	0.000	1.000	1.000
	XGB	0.961	0.002	0.990	0.990
CICIDS2017	LR	0.800	0.083	0.967	0.967
	RF	0.800	0.000	1.000	1.000
	XGB	0.800	0.000	1.000	1.000
Customized	LR	1.000	0.023	0.986	0.992
	RF	0.800	0.000	1.000	1.000
	XGB	0.854	0.000	1.000	0.873

Particular attention should be taken to the CICIDS2017 dataset because it contained only five attacks, and one of them being an exploit of the vulnerability Heartbleed. For this dataset, the various system configurations achieved similar results, including no detection of the Heartbleed attack, since its characteristics were not in the system training base.

The performance of the approach presented in this paper was also compared with other solutions in the literature, as shown in table IV. The proposed system with Logistic regression was competitive in terms of D_R and P_R, as can be seen, even with a lower sampling rate (S_R).

TABLE IV
Comparison with other approaches in the literature.

Work	DATASET	$\mathbf{S_R}$	$\mathbf{D_R}$	$\mathbf{FA_R}$	$\mathbf{P_R}$
[19]	CIC-DoS	30%	0.769	-	-
[24]	CICIDS2017	-	-	-	0.821
[16]	CIC-DOS	20%	0.936	0.000	0.999
[16]	CICIDS2017	20%	0.800	0.002	0.992
SD-IoT	CIC-DoS	20%	0.961	0.052	0.928
SD-IoT	CICIDS2017	20%	0.800	0.000	1.000

It was also observed the consumption of computational resources in the environment during the experiments. The system was stable, as shown in Fig. 6, with CPU spikes below 60% even during attacks. The computational effort is an essential point because the system runs in a hardware-constrained environment.

V. Conclusion

This paper presented the Smart Detection-IoT system, a solution that uses machine learning to classify IoT network traffic and detect denial of service attacks by only analyzing the IP/TCP header of network traffic samples, thus not compromising data privacy. The intention is to detect the attack as

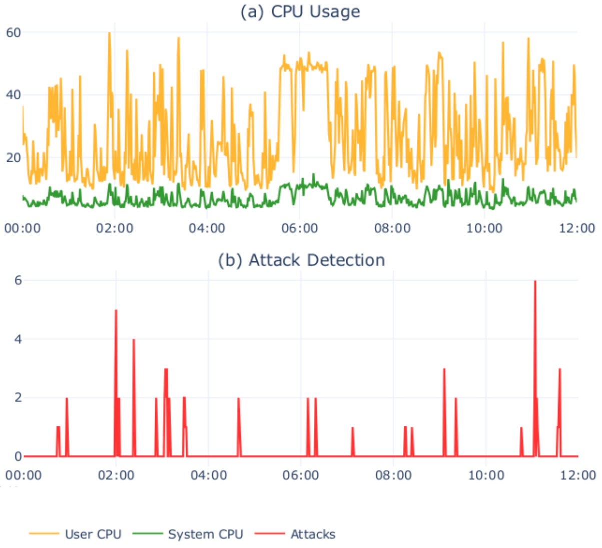

Fig. 6. CPU usage (a) and Attack alerts (b) during the experiment.

close as possible to the threat, allowing actions to be taken as quickly as possible to mitigate them.

The Smart Detection-IoT system performance was assessed under three classification algorithms, and competitive results were observed when the proposed system is compared to other approaches from the recent literature. The system was tested with three datasets: CIC-DOS, CICIDS2017, and a customized containing several DoS/DDoS attacks, such as UDP flood, TCP flood, HTTP flood, and HTTP slow. Based on the experimental results, the Smart Detection-IoT approach delivers enhanced $PREC$, FA_R, and D_R. For instance, in the CIC-DoS and customized datasets, the proposed system acquired D_R and PREC higher than 96% with FA_R less than 6%.

Forthcoming work must incorporate protection functionality to the wireless controller and analyze new DDoS attacks based on the service's vulnerabilities, to improve the signatures base. Another issue that must be treated in future work is the improvement of the false alarm rates related to some scenarios.

VI. Acknowledgments

The authors would like to acknowledge the High-Performance Computing Center of UFRN (NPAD/UFRN) for support this work. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] N. Vlajic and D. Zhou, "IoT as a Land of Opportunity for DDoS Hackers," *Computer*, vol. 51, no. 7, pp. 26–34, 2018.

[2] E. Summary, "Cisco public Cisco Visual Networking Index: Global Mobile Data Traffic The Cisco® Visual Networking Index (VNI) Global Mobile Data," pp. 2017–2022, 2019.

[4] V. Chandola, A. BANERJEE, and V. KUMAR, "Survey of Anomaly Detection," *ACM Computing Survey (CSUR)*, vol. 41, no. 3, pp. 1–72, 2009.

[3] Y. Cao, Y. Gao, R. Tan, Q. Han, and Z. Liu, "Understanding Internet DDoS Mitigation from Academic and Industrial Perspectives," *IEEE Access*, vol. 6, pp. 66 641–66 648, 2018.

[5] D. Yin, L. Zhang, and K. Yang, "A DDoS Attack Detection and Mitigation with Software-Defined Internet of Things Framework," *IEEE Access*, vol. 6, no. Mcc, pp. 24 694–24 705, 2018.

[6] R. L. S. De Oliveira, C. M. Schweitzer, A. A. Shinoda, and L. R. Prete, "Using Mininet for emulation and prototyping Software-Defined Networks," in *2014 IEEE Colombian Conference on Communications and Computing, COLCOM 2014 - Conference Proceedings*, 2014.

[7] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in *Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018*, 2018.

[8] M. Ozcelik, N. Chalabianloo, and G. Gur, "Software-Defined Edge Defense Against IoT-Based DDoS," *IEEE CIT 2017 - 17th IEEE International Conference on Computer and Information Technology*, pp. 308–313, 2017.

[9] M. Hasan, M. M. Islam, M. I. I. Zarif, and M. Hashem, "Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches," *Internet of Things*, vol. 7, p. 100059, 2019.

[10] N. Sambandam, M. Hussein, N. Siddiqi, and C. H. Lung, "Network Security for IoT Using SDN: Timely DDoS Detection," *DSC 2018 - 2018 IEEE Conference on Dependable and Secure Computing*, no. January, pp. 1–2, 2019.

[11] P. Manso, J. Moura, and C. Serrão, "SDN-based intrusion detection system for early detection and mitigation of DDoS attacks," *Information (Switzerland)*, vol. 10, no. 3, pp. 1–17, 2019.

[12] ONF, "OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06) M," pp. 1–283, 2015.

[13] P. Phaal and M. Lavine, "sFlow Version 5," *Request for Comments*, 2004.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2012.

[15] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers and Security*, vol. 31, no. 3, pp. 357–374, 2012.

[16] F. Sales, D. L. Filho, F. A. F. Silveira, A. D. Medeiros, B. Junior, G. Vargas-solar, and L. F. Silveira, "Smart Detection : An Online Approach for DoS / DDoS Attack Detection Using Machine Learning," *Security and Communication Networks*, vol. 2019, p. 15, 2019.

[17] A. Geron, *Hands-On Machine Learing With Scikit-Learn & Tensor Flow*. O'Railly, 2017.

[18] F. Bellard, "QEMU , a Fast and Portable Dynamic Translator," *USENIX Annual Technical Conference. Proceedings of the 2005 Conference on*, 2005.

[19] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting HTTP-Based Application Layer DoS Attacks on Web Servers in the Presence of Sampling," *Computer Networks*, vol. 121, pp. 25–36, 2017.

[20] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *ICISSP 2018 - 4th International Conference on Information Systems Security and Privacy*, no. Cic, 2018, pp. 108–116.

[21] M. Powers, "Evaluation: from Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[22] A. Turner, "Tcpreplay," 2013.

[23] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon, A. Networks, and M. Casado, "The Design and Implementation of Open vSwitch," *Nsdi*, pp. 117–130, 2015.

[24] M. Aamir and S. M. A. Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification," *Journal of King Saud University - Computer and Information Sciences*, vol. In press, no. xxxx, 2019.