Hindawi
Security and Communication Networks
Volume 2019, Article ID 1574749, 15 pages
https://doi.org/10.1155/2019/1574749

Research Article

Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning

Francisco Sales de Lima Filho,[1] **Frederico A. F. Silveira**,[1] **Agostinho de Medeiros Brito Junior**,[1] **Genoveva Vargas-Solar**,[2] **and Luiz F. Silveira**[1]

[1]*Computer Engineering and Automation Department, Federal University of Rio Grande do Norte, P.O. Box 1524, Natal 59078-970, Brazil*
[2]*Centre National de la Recherche Scientifique LIG, Grenoble, France*

Correspondence should be addressed to Francisco Sales de Lima Filho; salesfilho@gmail.com

Received 11 June 2019; Accepted 31 August 2019; Published 13 October 2019

Academic Editor: Leandros Maglaras

Users and Internet service providers (ISPs) are constantly affected by denial-of-service (DoS) attacks. This cyber threat continues to grow even with the development of new protection technologies. Developing mechanisms to detect this threat is a current challenge in network security. This article presents a machine learning- (ML-) based DoS detection system. The proposed approach makes inferences based on signatures previously extracted from samples of network traffic. The experiments were performed using four modern benchmark datasets. The results show an online detection rate (DR) of attacks above 96%, with high precision (PREC) and low false alarm rate (FAR) using a sampling rate (SR) of 20% of network traffic.

1. Introduction

In recent years, distributed denial-of-service (DDoS) attacks have caused significant financial losses to industry and governments worldwide, as shown in information security reports [1]. These records are in line with the growing number of devices connected to the Internet, especially driven by the popularization of ubiquitous computing, materialized through the Internet of Things (IoT) [2] paradigm and characterized by the concept of connecting anything, anywhere, anytime. In most Internet scenarios, devices interact with applications that run remotely on the network, which enables malicious agents to take control of devices. In this way, it is possible to have the interruption of services or the use of devices as a launching point of attacks for diverse domains, as is the case of the DDoS attack [3], which has been consolidated for several reasons, such as (i) simplicity and facility of execution, not requiring vast technological knowledge on the attacker side, and (ii) variety of platforms and applications for facilitated attack orchestration. Many of these attacks succeeded in disrupting essential Internet services such as DNS, affecting millions of users around the world [4], and commercial platforms such as the GitHub [5], prompting severe financial losses to the organizations that depend on those services.

One of the most dangerous malicious traffic on the Internet is the DDoS volumetric attack, which is responsible for more than 65% of all such attacks [6]. In a volumetric DDoS attack, several attackers coordinate the sending of a high rate of useless data in an attempt to overload the victim's computing resources or the near network links. On the one hand, the high success rates for this type of attack occur because the main Internet routers typically use the FIFO (First-In-First-Out) and DROP-TAIL queuing disciplines, which do not differentiate between types of traffic, imposing equal loss rates for attacks and legitimate traffic. Although legitimate traffic tends to retreat to prevent further congestion, attack traffic does not have this commitment and causes the links to be exceeded. As a consequence, legitimate traffic is also obstructed [6]. On the other hand, the attackers are using more advanced techniques to potentiate attacks and flood the victim such as DDoS-for-hire, IoT-based

DDoS attacks, and reflection DDoS attacks [7–9], profiting from the computational capability and geographical distribution promoted by the wide variety of devices and its diverse mobility patterns, typically founded in IoT and mobile IoT scenarios.

In addition to the volumetric DDoS attack, low-volume attacks are on the radar of security experts. It is a more sneaky attack that uses few invading hosts; events are fast, sometimes lasting only a few minutes and usually less than an hour. For these reasons, security teams do not know that their sites are under attack because common tools do not detect this type of threat [10]. Typically, low-volume DDoS exploits application layer protocols, respects other protocols, does not overload links, but causes exhaustion of victim resources.

1.1. Problem Statements. DDoS detection and mitigation have been under study in both the scientific community and industry for several years. The related literature reveals that several studies have undertaken to propose solutions to deal with this problem in a general way [6, 11–15]. Another group of works dedicated themselves to presenting specific solutions for high-volume and low-volume DDoS attacks [8, 13, 16]. Furthermore, despite the diverse recommendations for mitigating DDoS attacks proposed by the Computer Emergency Response Team (CERT) and guidelines documented through Request for Comments (RFC), these attacks still occur with high frequency.

A study carried out years ago [17] revealed that the ineffectiveness of detecting and mitigating DDoS attacks is directly related to constant configuration errors and wasted time due to the lack of tools that follow the dynamics of the network without constant human interference. This has led researchers to use autonomous solutions that can operate (detect and mitigate) based on the behavior and characteristics of the traffic. In this sense, the adoption of solutions with techniques based on artificial intelligence, mainly machine learning (ML), has been distinguished by offering high flexibility in the classification process, consequently improving the detection of malicious traffic [18, 19].

The industrial sector offers DDoS protection as a service through large structures, usually operated by specialized providers [6] such as Akamai, Cloudflare, and Arbor Networks, which have large processing capacity and proprietary filtering mechanisms. But the industry also has problems, such as fragility in routing traffic, usually via Domain Name System (DNS) or Border Gateway Protocol (BGP), difficulty detecting slow attacks, and privacy issues, which drive away some customer segments as governments.

Finding the balance between academic propositions and the industrial practice of combating DDoS is a big challenge. The academy invests in techniques such as machine learning (ML) and proposes to apply them in areas such as DDoS detection in Internet of Things (IoT) [20, 21] sensors, wireless sensors [22], cloud computing [23] and software-defined networking (SDN) [18] and work on producing more realistic datasets [24, 25] and more effective means of result validation [26, 27]. On the other hand, industry segments gradually invested in new paradigms in their solutions such as network function virtualization (NFV) and SDN [28, 29] to apply scientific discoveries and modernize network structures. Even so, the incidents of DDoS still happen daily, reinforcing that the problem is not solved.

1.2. Proposal. Realizing these issues, this article proposes Smart Detection, a novel defense mechanism against DDoS attacks. The system architecture was designed to detect both high- and low-volume DDoS attacks. The proposed system acts as a sensor that can be installed anywhere on the network and classifies online traffic using an MLA-based strategy that makes inferences utilizing random traffic samples collected on network devices via stream protocol. The proposed approach is compatible with the Internet infrastructure and does not require software or hardware upgrades. Besides, user data privacy is guaranteed at all stages of system operation.

1.3. Contributions. In summary, the significant contributions of Smart Detection are as follows:

(i) The modeling, development, and validation of the detection system are done using a customized dataset and other three well-known ones called CIC-DoS, CICIDS2017, and CSE-CIC-IDS2018, where the system receives online random samples of network traffic and classifies them as DoS attacks or normal.

(ii) The proposed detection system differs from other approaches by early identification of a variety of volumetric attacks, such as TCP flood, UDP flood, and HTTP flood, as well as stealth attacks such as HTTP slow headers, HTTP slow body, and HTTP slow read, even with a low traffic sampling rate. In addition, Smart Detection is compatible with the current Internet infrastructure and does not require any software or hardware upgrades on ISPs. At the same time, the proposed system uses advanced technologies such as ML and NFV.

(iii) Unlike existing security service providers, the proposed system does not require traffic redirection or connection intermediation. Data privacy is guaranteed at all stages. First, the system randomly processes only a small part of the network traffic. Second, it does not do packet deep inspection. Instead, Smart Detection parses only network layer header data.

(iv) The pattern recognition of normal network traffic and several DoS attack types are addressed. As a result, a novel signature database is created, which is used by Smart Detection and can be applied to other systems.

(v) An approach to automatic feature selection has been developed using the cross-validation technique for model searches that meet specific classification quality criteria. This approach was used to define the signatures adopted by Smart Detection.

2. Related Works and Background

The research on intrusion detection in computer networks is widely discussed in the literature. Several detection techniques and protection strategies have been proposed in recent years. Studies in the literature classify the IDSs as signature-based, anomaly-based, and hybrid systems. The first type identifies potential attacks by comparing current observed events with its stored signatures. The second one detects anomalies by identifying significant deviations between the preestablished normal profile and the current events. In all cases, an alert will be generated if any signature is matched or if a deviation occurs above a set threshold. The main advantage of the signature-based approach is the low false alarm rate. However, the challenge is to write signatures that cover all possible attack variations. In contrast, the anomaly-based approach has the ability to detect unknown attacks, but it requires more computational resources and often produces more false alarms. Hybrid solutions try to exploit the benefits of both techniques [11, 30]. DoS attacks are a specific type of network intrusion that has drawn the attention of academia, as highlights recent surveys on network applications, wireless networks, cloud computing, and big data [8, 13, 14, 31].

Several classification strategies of DDoS attacks have been proposed in the literature in the last decade. However, DDoS flooding attacks have been further studied, being classified into two categories based on the protocol level that is targeted [3]:

(i) Network/transport-level DDoS flooding attacks: such attacks are mostly launched using Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), and Domain Name System (DNS) protocol packets.

(ii) Application-level DDoS flooding attacks: such attacks are focused on disrupting legitimate user services by exhausting the server resources, e.g., sockets, central processing unit (CPU), memory, disk/database bandwidth, and input/output (I/O) bandwidth. Application-level DDoS attacks generally consume less bandwidth and are stealthier in nature than volumetric attacks since they are very similar to benign traffic.

The biggest challenge in combating DDoS attacks lies in the early detection and mitigation of attacks as close as possible to their origin; however, the implementation of a comprehensive solution that addresses these features has not yet been achieved [3, 32].

Some recent work has inspired the development of the Smart Detection system. These approaches are listed in Table 1 for comparative purposes.

Table 1: Recent related works.

References	Dataset	Online	L/H DoS	Sampling
[16]	CIC-DoS	✗	✓	✓
[33]	None	✓	✗	✗
[34]	MIT Lincoln, FIFA98, DDoSTB, CAIDA	✓	✓	✗
[35]	Customized (developed by the authors)	✓	✓	✗
[36]	Customized (developed by the authors)	✓	✓	✗
[37]	CICIDS2017	✗	✓	✗
The proposed approach	CIC-DoS, CICIDS2017, CSE-CIC-IDS2018, Customized	✓	✓	✓

A Hypertext Transfer Protocol- (HTTP-) based technique [16] was proposed to detect flood attacks in web servers using data sampling. The authors used the CUMSUM algorithm to determine whether the analyzed traffic is normal or a DoS attack by focusing on two features: the number of application layer requests and the number of packets with payload size equal to zero. The results showed a detection rate between 80 and 88% using a sampling rate of 20%. Although it has made important advances, the proposed method does not seem applicable in automatic mitigation systems, especially in production environments that do not support high sampling rates.

D-FACE is a collaborative defense system [34] that uses a generalized entropy (GE) and generalized information distance (GID) metrics in detecting different types of DDoS attacks and flash events (FEs). In this context, an FE is similar to a volumetric DDoS wherein thousands of legitimate users try to access a particular computing resource such as a website simultaneously. The results show that D-FACE can detect DDoS attacks and FEs. Although the work presents relevant contributions, the validation used obsolete datasets. In addition, the proposed collaboration approach requires a high degree of ISP engagement, so it restricts the industrial use of the solution.

Antidose system [33] presents a means of interaction between a vulnerable peripheral service and an indirectly related Autonomous System (AS), which allows the AS to confidently deploy local filtering rules under the control of the remote service. The system was evaluated using Mininet, but no benchmark dataset was used. The approach proposed by the authors faces strong resistance from ISPs for two reasons: the first is software and hardware update requirement and the second is having no control over local traffic control policies.

The SkyShield system [35] has been proposed to detect and mitigate DDoS attacks on the application layer. In the detection phase, SkyShield exploits the divergence between two hash tables (Sketches) to detect anomalies caused by attacker hosts. The mitigation phase uses filtering, whitelisting, blacklisting, and CAPTCHA as protection mechanisms. The system was evaluated using customized datasets. SkyShield has focused on the application layer, more specifically on the HTTP protocol, so the proposed system is vulnerable to flooding at the network layer and transport layer.

Umbrella [36] develops a multilayered defense architecture to defend against a wide spectrum of DDoS attacks. The authors proposed an approach based on detection and protection exclusively on the victim's side. The system was

evaluated using the customized testbed in terms of traffic control. The authors claim that the system is capable of dealing with mass attacks. However, this approach is widely used in industry and proved to be inefficient against truly massive DDoS attacks.

Recently, a semi-supervised machine learning system addressed the classification of DDoS attacks. In this approach, CICIDS2017 dataset was used to evaluate system performance metrics [37]. Although the work addresses the recent DoS vectors, the online performance of the method has not been evaluated. Finally, Table 1 summarizes these recent works whose approach is related to the proposal of this article.

In Table 1, Online indicates that the proposed system has been tested in online experiments and dataset informs the dataset used for validation, while L/H DoS indicates whether it detects slow and high DDoS attacks. Sampling indicates whether some network traffic sampling method is used.

Based on the open questions in the literature and recent specialized reports, DoS attacks may remain on the Internet for some time. The solution to this problem includes the adoption of detection and mitigation strategies that are practical and economically viable. Besides, these approaches should leverage existing provider infrastructure and be implemented in light of new scientific and technological trends.

3. Smart Detection

Smart Detection is designed to combat DDoS attacks on the Internet in a modern collaborative way. In this approach, the system collects network traffic samples and classifies them. Attack notification messages are shared using a cloud platform for convenient use by traffic control protection systems. The whole process is illustrated in Figure 1.

The core of the detection system consists of a Signature Dataset (SDS) and a machine learning algorithm (MLA). Figure 2 shows the crucial steps from model build to system operation.

First, normal traffic and DDoS signatures were extracted, labeled, and stored in a database. SDS was then created using feature selection techniques. Finally, the most accurate MLA was selected, trained, and loaded into the traffic classification system.

The architecture of the detection system was designed to work with samples of network traffic provided by industrial standard traffic sampling protocols, collected from network devices. The unlabeled samples are received and grouped in flow tables in the receiver buffer. Thus, when the table length is greater than or equal to the reference value, they are presented to the classifier responsible for labeling them, as shown in Figure 3. If the flow table expires, it may be processed one more time. The occurrence of small flow tables is higher at lower sampling rates or under some types of DoS attacks, e.g., SYN flood attacks. Table 2 details the parameters for fine tuning of the system.

The complete algorithm of the detection system is summarized in Figure 4. During each cycle of the detection process, traffic samples are received and stored in a flow table. For each new flow, a unique identifier (FlowID) is calculated based on the 5-tuple (src_IP, dst_IP, src_port, dst_port, and transport_protocol) in steps 1 and 2. If this is a new flow, i.e., there is not any other flow table stored with the same FlowID, the flow table is registered in a shared memory buffer. Otherwise, if there is a flow table registered with the same FlowID such as the previously calculated one, the data of the new flow will be merged with the data in the existing flow table in steps 3 and 4. After the merging operation, if the table length is greater than or equal to the reference value ($T_l \geq T_{max}$), the flow table is classified, and if it is found to be an attack, a notification is emitted. Otherwise, it is inserted back into the shared memory buffer. Meanwhile, in step 7, the cleanup task looks for expired flow tables in the shared buffer, i.e., flow tables that exceed the expiration time of the system ($E > E_T$). For each expired flow table, the system checks the table length. If the flow table length is less than or equal to the minimum reference value ($T_l \leq T_{min}$), this flow table will be processed by step 8. A new FlowID is calculated using the 3-tuple (src_IP, dst_IP, and transport_protocol), as the flow table is routed back to steps 3 and 4.

3.1. Traffic Sampling. Smart Detection uses a network traffic sampling technique because processing all the packets in the network can be a computationally expensive task, even if only the packet headers are parsed. In many cases, performing a deep inspection and analyzing the data area of the application layer is unfeasible for detection systems. Among the protocols adopted by the industry for sampling network traffic, the sFlow protocol is widely used in current devices. The technique used by sFlow is called n-out-of-N sampling. In this technique, n samples are selected out of N packets. One way to achieve a simple random sample is to randomly generate n different numbers in the range of 1 to N and then choose all packets with a packet position equal to one of the n values. This procedure is repeated for every N packets. Besides, the sample size is fixed in this approach [38].

The sFlow monitoring system consists of an agent (embedded in a switch, a router, or an independent probe) and a collector. The architecture used in the monitoring system is designed to provide continuous network monitoring of high-speed switched and routed devices. The agent uses the sampling technology to capture traffic statistics from the monitored device and forward them to a collector system [39].

3.2. Feature Extraction. In supervised classification strategies, a set of examples is required for training the classifier model. This set is commonly defined as the signature database. Each instance of the database has a set of characteristics or variables associated with a label or a class. In this work, the goal is to identify characteristics in network traffic that are able to distinguish the normal network behavior from DoS attacks. The study is focused on the analysis of the header variables of the network and transport layer packets of the TCP/IP architecture because it allows saving computational resources and simplifies the deployment in the ISP networks.

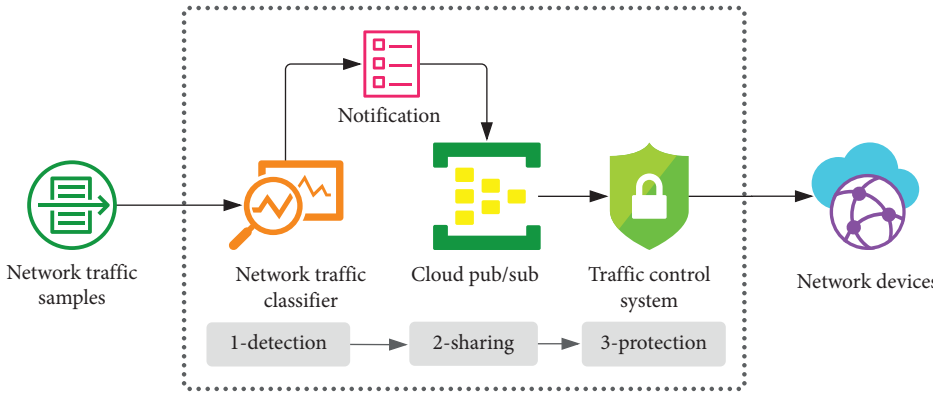

Figure 1: Operation scenario overview of the Smart Detection system.

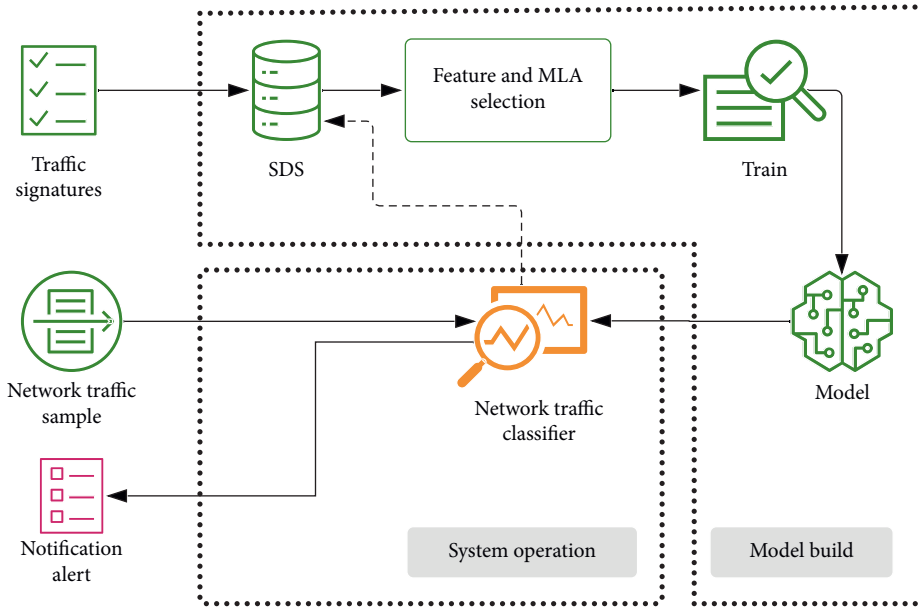

Figure 2: Detection system overview.

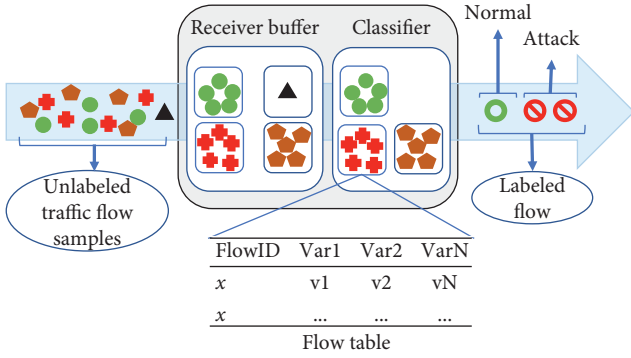

Figure 3: Overview of traffic classification scheme.

In IPv4-compliant networks, the network and transport layer protocols are IP, TCP, and UDP, which are specified in RFC 791 [40], RFC 793 [41], and RFC 768 [42], respectively. Together, such protocols have a total of 25 header variables. However, widely used network traffic sampling protocols such as NetFlow [43] and sFlow [39] use only a portion of these variables in the sampling process. Commonly, the seven used variables are the source and destination IP addresses, source and destination ports, transport layer protocol, IP packet size, and TCP flags.

Table 2: Detection system parameters.

Parameter	Description
T_{min}	Minimum flow table length
T_{max}	Maximum flow table length
E_T	Flow table expiration time

The source and destination IP addresses are not very useful for identifying the network traffic behavior in the Internet environment, which reduces the number of variables available for analysis to five in the most common cases. Based on the five variables mostly used by the flow monitoring protocols, 33 variables were derived, as described in Table 3, which use statistical measures that express data variability. In the calculation context of the database variables, the references to the mean, median, variance (var), and

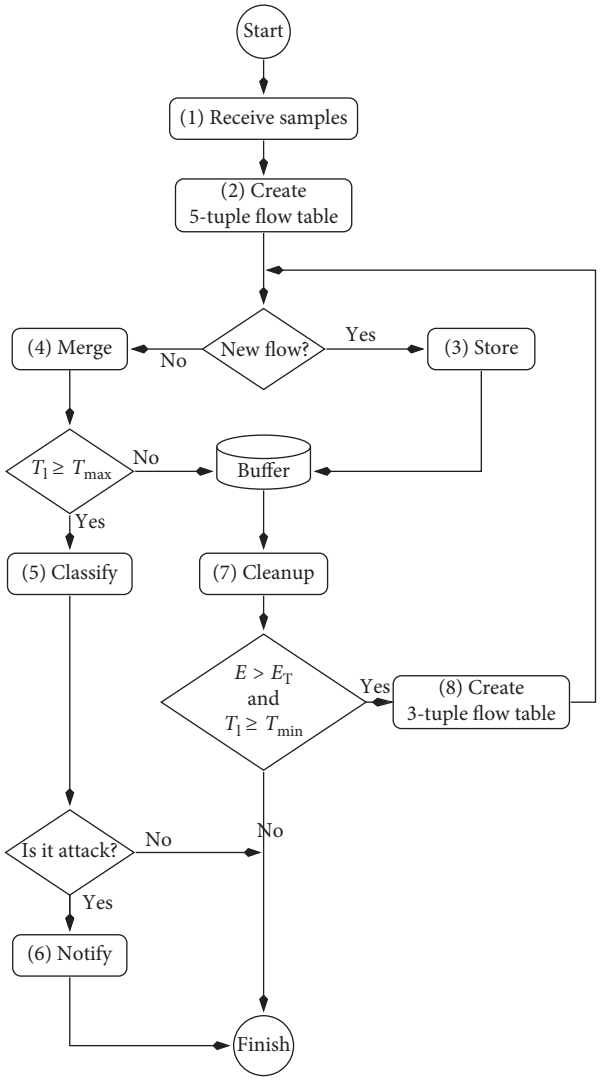

Figure 4: Detection system algorithm.

standard deviation (std) should be interpreted as sample measures.

The variable named protocol is a simple normalization of the protocol field extracted from the transport layer packet headers in the form:

$$\text{ip\_proto} = \frac{N_{\text{proto}}}{K}, \tag{1}$$

where N_{proto} is the code of the protocol and K is an normalization constant set to the value 1,000. For instance, $N_{\text{proto}} = 6$ and $N_{\text{proto}} = 17$ in TCP and UDP protocols, respectively.

With the main four variables mostly used in flow monitoring, it is possible to calculate the following associated statistic measurements:

(i) Entropy: the *entropy* of the variable is calculated by

$$\text{Entropy}(X) = -\sum_i p(X_i)\log_2 p(X_i), \tag{2}$$

Table 3: Extracted variables.

#	Variable	Detail
01	ip_proto	Normalized protocol number
02	ip_len_mean	Mean of IP length
03	ip_len_median	Median of IP length
04	ip_len_var	Variance of IP length
05	ip_len_std	Stand. deviation of IP length
06	ip_len_entropy	Entropy of IP length
07	ip_len_cv	Coeff. of variation of IP length
08	ip_len_cvq	Quantile coeff. of IP length
09	ip_len_rte	Rate change of IP length
10	sport_mean	Mean of src port
11	sport_median	Median of src port
12	sport_var	Variance of src port
13	sport_std	Stand. deviation of src port
14	sport_entropy	Entropy of src port
15	sport_cv	Coeff. of variation of src port
16	sport_cvq	Quantile coeff. of src port
17	sport_rte	Rate change of src port
18	dport_mean	Mean of dest. port
19	dport_median	Median of dest. port
20	dport_var	Variance of dest. port
21	dport_std	Stand. deviation of dest. port
22	dport_entropy	Entropy of dest. port
23	dport_cv	Coeff. of variation of dest. port
24	dport_cvq	Quantile coeff. of dest. port
25	dport_rte	Rate change of dest. port
26	tcp_flags_mean	Mean of TCP flags
27	tcp_flags_median	Median of TCP flags
28	tcp_flags_var	Variance of TCP flags
29	tcp_flags_std	Stand. deviation of TCP flags
30	tcp_flags_entropy	Entropy of TCP flags
31	tcp_flags_cv	Coeff. of variation of TCP flags
32	tcp_flags_cvq	Quantile coeff. of TCP flags
33	tcp_flags_rte	Rate change of TCP flags

where X is the variable of interest, e.g., the source port.

(ii) Coefficient of variation: the *coefficient of variation* is calculated by

$$\text{cv}(X) = \frac{\text{std}(X)}{\text{mean}(X)}, \tag{3}$$

where $\text{std}(X)$ is the estimated standard deviation and $\text{mean}(X)$ is the estimated average of the variable.

(iii) Quantile coefficient: this parameter is defined herein by

$$\text{cvq}(X) = \frac{\widehat{Q}_X(1-p) - \widehat{Q}_X(p)}{\widehat{Q}_X(1-p) + \widehat{Q}_X(p)}, \tag{4}$$

where $\widehat{Q}_X(p)$ is the sample p-quantile, $p \in (0, 0.5)$, expressed by [44]

$$\widehat{Q}_X(p) = X_{(k)} + \left(X_{(k+1)} - X_{(k)}\right) * f, \tag{5}$$

with being $\{X_{(1)}, \ldots, X_{(n)}\}$ the order statistics of independent observations, $\{X_1, \ldots, X_n\}$, $k = \lfloor p * n \rfloor$, and f is the fractional part of the index surrounded by $X_{(k)}$ and $X_{(k+1)}$.

(iv) Rate of change: this metric is given by

$$\text{rte}(X) = \frac{U_X}{S_X}, \tag{6}$$

(v) where U_X is the amount of unique values and S_X is the overall number of X values.

The data traffic with normal activity behavior was extracted from the ISCXIDS2012 dataset [45]. The data traffic with DoS behavior was obtained in a laboratory controlled environment using tools such as hping3 [46], hulk [47], Goldeneye [48], and slowhttptest [49].

Processes such as extracting, transforming, and labeling the database instances are summarized in Figure 5. The raw network traffic was extracted from the capture files, as the packets were then grouped into sessions. For each session, one instance of the descriptor database containing all variables listed in Table 3 was calculated. In this study, only the sessions with five hundred packets or higher were considered to better represent each network traffic type.

The final database contains examples of normal traffic (23,088 instances), TCP flood attacks (14,988 instances), UDP flood (6,894 instances), HTTP flood (347 instances), and HTTP slow (183 instances).

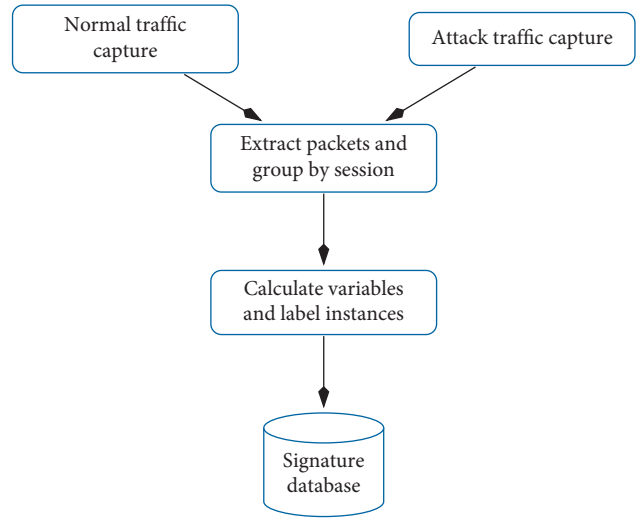

Figure 5: Process of network traffic extraction, transformation, and labeling.

Table 4: 10-fold RFECV results.

#	MLA	No. of features	Accuracy
1	RF	28	0.996010
2	DTree	25	0.994182
3	LR	26	0.972327
4	SGD	16	0.969474
5	Perceptron	28	0.937256
6	AdaBoost	7	0.931131

3.3. Feature and MLA Selection. Feature selection is an important step in the pattern recognition process and consists of defining the smallest possible set of variables capable of efficiently describing a set of classes [50]. Several techniques for variable selection are available in the literature and implemented in software libraries as scikit-learn [51]. In this work, the selection of variables was performed in two stages. First, Recursive Feature Elimination with Cross-Validation (RFECV) was used with some machine learning algorithms widely used in the scientific literature, i.e., random Forest (RF), logistic regression (LR), AdaBoost, stochastic gradient descent (SGD), decision tree (DTree), and perceptron. RF obtained higher precision using 28 variables, while AdaBoost selected seven variables, but obtained lower accuracy, as shown in Table 4. In the second stage, a new feature selection test was performed with RF using proposed Algorithm 1.

In the proposed feature selection approach using RF, the number of variables was reduced from 28 to 20 with a small increase in accuracy, as shown in Table 5. The proposed algorithm was executed using the following input parameters: 1,000 rounds, 99% of variable importance, 95% of global accuracy, and 85% of per-class accuracy. Figure 6 shows that most models tested used 20 variables. However, each model used specific sets of variables. In order to choose the most relevant variables from the selected models, the RF variable importance criterion was used, as described in line 25 of Algorithm 1. The final result of feature selection is shown in Figure 7.

The results show that RF obtained higher accuracy than the other algorithms. Although it uses more variables than SGD and AdaBoost, a low false alarm rate is a prime requirement in DDoS detection systems. In this case, RF proved to be the best algorithm option for the Smart Detection system. Random forest is a supervised learning algorithm that builds a large number of random decision trees and merges them together to make predictions. Each tree is trained with a random subset of the total set of labeled samples. In the classification process, the most voted class among all the trees in the model indicates the result of the classifier [52]. In the proposed detection system algorithm shown in Figure 4, RF is used to classify online network traffic, a task that requires computational efficiency and high hit rates.

4. Results

The network traffic was classified by the detection system in a controlled network environment using different sampling rates. In the experiments, raw network traffic of the CIC-DoS [16], CICIDS2017 [25], and CSE-CIC-IDS2018 [25] datasets and the raw network traffic captured in the customized testbed experiments were employed. The Smart Detection system has reached high accuracy and low false-positive rate. Experiments were conducted using two Virtual Linux boxes,

Input: database descriptors, variable importance threshold, accuracy threshold, and number of rounds
Output: selected variables

```
(1) begin
(2)   Create empty optimized model set;
(3)   for i ⟵ 1 to Number of rounds do
(4)     Define all the descriptor database variables as the current variables;
(5)     while True do
(6)       Split dataset in training and test partitions;
(7)       Create and train the model using training data partition;
(8)       Select the most important variables from the trained model;
(9)       Calculate the cumulative importance of variables from the trained model;
(10)      if max (cumulative importance of variables) < Variable importance threshold then
(11)        Exit loop;
(12)      end
(13)      Train the model using only the most important variables;
(14)      Test the trained model and calculate the accuracy;
(15)      if Calculated accuracy < Accuracy threshold then
(16)        Exit loop;
(17)      end
(18)      Add current model to optimized model set;
(19)      Define the most important variables from the trained model as the current variables;
(20)    end
(21)  end
(22)    Group the models by number of variables;
(23)    Remove outliers from the grouped model set;
(24)    Select the group of models with the highest frequency and their number of variables "N";
(25)    Rank the variables by the mean of the importance calculated in step 7;
(26)    Return the "N" most important variables;
(27) end
```

Algorithm 1: Feature selection.

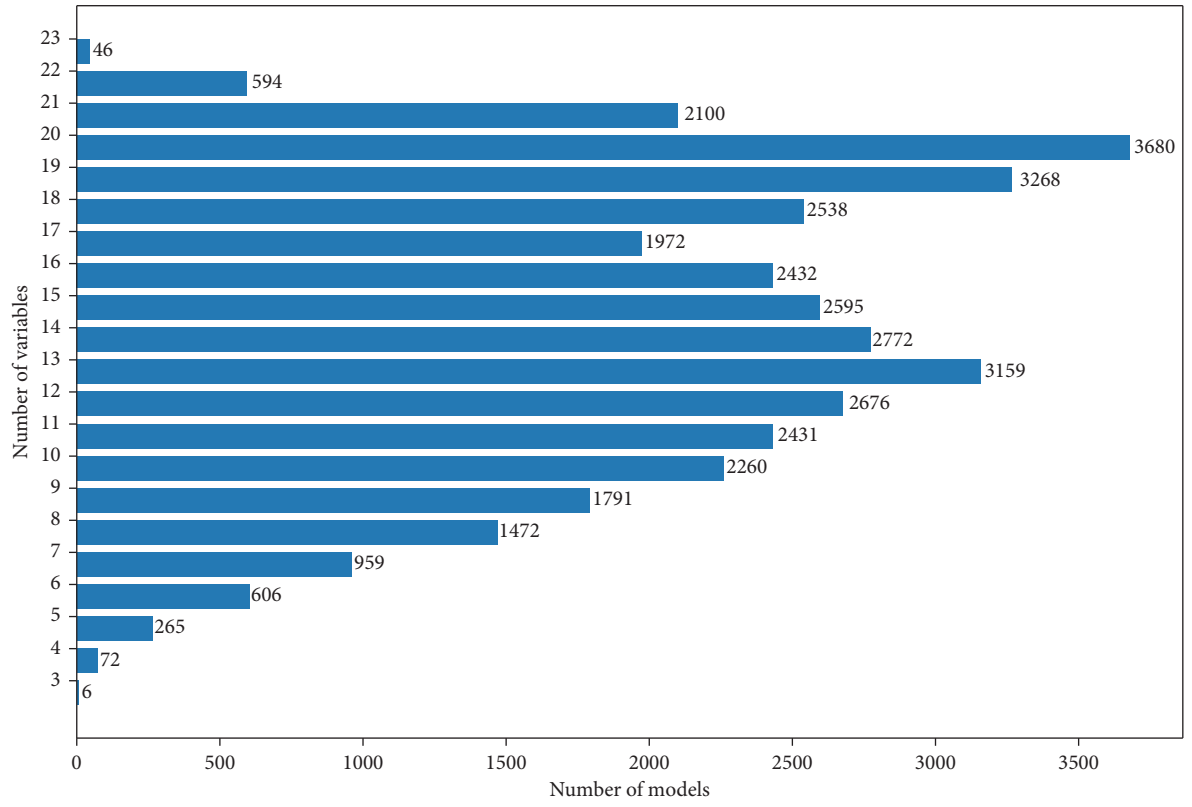

Figure 6: Number of variables versus number of models.

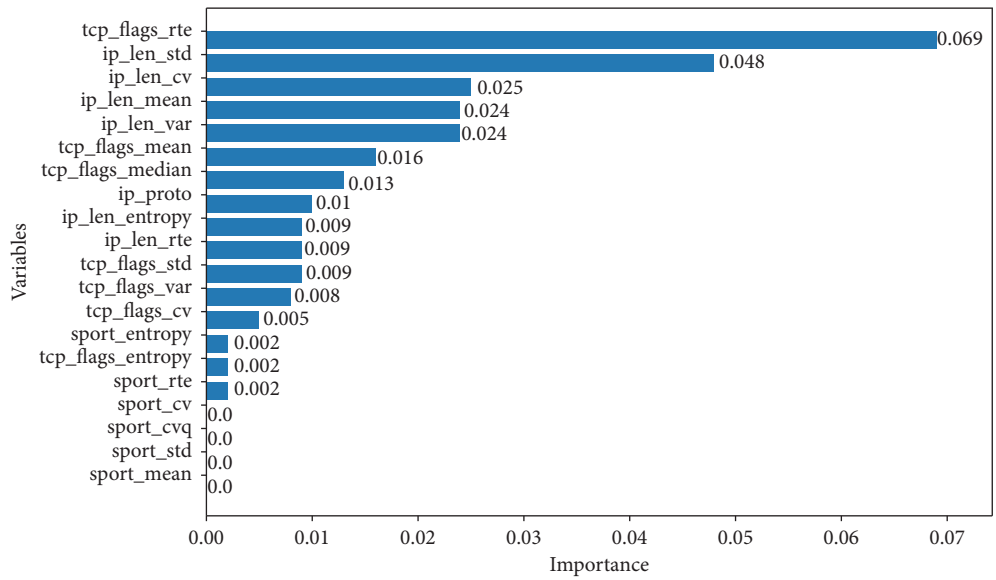

Figure 7: Selected variables by the proposed feature selection algorithm.

each one of them using 8 virtual CPUs (vCPUs) with 8GB RAM.

4.1. Description of the Benchmark Datasets. Many different datasets such as DARPA (Lincoln Laboratory 1998-99), KDD′99 (University of California, Irvine 1998-99), and LBNL (Lawrence Berkeley National Laboratory and ICSI 2004-2005) have been used by the researchers to evaluate the performance of their proposed intrusion detection and prevention approaches. However, many such datasets are out of date and unreliable to use [25]. In this study, the datasets CIC-DoS, CICIDS2017, and CSE-CIC-IDS2018 and customized dataset were used since they include modern threats and DoS techniques.

4.1.1. The ISCXIDS2012 Dataset. The Information Security Center of Excellence (ISCX) IDS 2012 dataset (ISCXIDS2012) was built in at the University of New Brunswick to provide a contemporary benchmark. The dataset traced real packets for seven days of network activity, including HTTP, SMTP, SSH, IMAP, POP3, and FTP protocols covering various scenarios of normal and malicious activities. ISCXIDS2012 consists of labeled network traces, including full packet payloads in pcap format and is publicly available (https://www.unb.ca/cic/datasets/ids.html) [45]. This work focuses on the normal activities of the ISCXIDS2012 pcap file for extraction of signatures, more specifically data file of Friday, 11/6/2010.

4.1.2. The CIC-DoS Dataset. The CIC-DoS dataset focuses on application layer DoS attacks mixed with the ISCXIDS2012 dataset attack-free traces. Four kinds of attacks were produced with different tools, yielding 8 different DoS attack strokes from the application layer [16]. The resulting set contains 24 hours of network traffic with a total size of 4.6 GB and is publicly available (https://www.unb.ca/cic/datasets/dos-dataset.html). A summary of the attack events and tools used in the CIC-DoS is presented in Table 6.

In the execution of the low-volume attacks using tool slowhttptest [49], the default value of 50 connections per attack was adopted, thus making the attacks more sneaky according to [16].

4.1.3. The CICIDS2017 Dataset. CICIDS2017 dataset was recently developed by ISCX and contains benign traffic and the most up-to-date common attacks. This new IDS dataset includes seven common updated family of attacks that met the real-world criteria and is publicly available (http://www.unb.ca/cic/datasets/IDS2017.html) [25].

This work focuses on the malicious DoS activities of the Wednesday, July 5, 2017, capture file, which consist of five DoS/DDoS attacks and a wide variety of normal network traffic. The resulting set contains 8h of network traffic with a total size of 13G. The attack tools used include slowloris, Slowhttptest, Hulk, GoldenEye, and Heartbleed.

4.1.4. The CSE-CIC-IDS2018 Dataset. This dataset is a collaborative project between the Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC). The final dataset includes seven different attack scenarios: brute force, Heartbleed, Botnet, DoS, DDoS, web attacks, and infiltration of the network from inside. The attacking infrastructure includes 50 machines, and the victim organization has 5 departments and includes 420 machines and 30 servers. A research document outlining the dataset analysis details and similar related principles has been published by [25]. All data are publicly available (https://www.unb.ca/cic/datasets/ids-2018.html).

This work focuses on the malicious DoS/DDoS activities of Friday, February 16, 2018, and Tuesday, February 20,

Table 5: 10-fold cross-validation results using the variables selected by the proposed algorithm.

#	MLA	No. of features	Accuracy
1	RF	20	0.999363
2	DTree	20	0.999011
3	Perceptron	20	0.996000
4	SGD	20	0.986989
5	LR	20	0.982704
6	AdaBoost	20	0.956512

2018, capture files. The attack tools used include Slow-HTTPTest, Slowhttptest, Hulk, LOIC, and HOIC.

4.1.5. The Customized Dataset. The customized dataset was developed in a controlled network environment, as shown in Figure 8. VLANs 10, 20, 30, and 40 are used as victim hosts. VLAN 165 is dedicated to the users of an academic unit. VLAN 60 is used as an attacking host, while monitoring occurs in VLAN 1. All networks have regular access to the Internet.

The attack plan was configured so that one attack is generated every 30 minutes, in a total of 48 attack events in 24 hours, starting at 00 h00 m00 s and ending at 23 h59 m00 s. All attacks were executed by attacker host 172.16.60.100, during which it did not transmit legitimate traffic to the victims. The attack tools were parameterized to produce sneaky low-volume, medium-volume or light mode, and massive high-volume attacks. Ten variations of protocol-based and application-based attacks were adopted using four attack tools, as shown in Table 7. The duration of protocol-based and high-volume application-based attacks was 30 seconds, while low-volume application-based attacks ranged from 30 to 240 seconds.

In the accomplishment of low-volume attacks using tool slowhttptest [49], the number of connection parameters was adopted as 1,500, instead of the default option corresponding to 50 connections.

4.2. Online Experiments. Online experiments were performed in a controlled laboratory environment according to the following validation methodology:

(1) Raw data of the network traffic are obtained for analysis in pcap file format.

(2) The attack plan indicating the origin, destination, attack type, and respective duration for the traffic indicated in step 1 is drawn.

(3) The environment for reprocessing and classifying the traffic is configured.

(4) The traffic is processed and classified.

(5) The system performance is properly evaluated by comparing the output from step 4 with the attack plan described in step 2.

Following this validation methodology, the sources of traffic capture were used: CIC-DoS, CICIDS2017, CSE-CIC-IDS2018, and customized dataset, thus fulfilling steps 1 and 2. The environment for traffic reprocessing and classification as described in step 3 was configured using two Virtual Linux boxes running Open Virtual Switch (OVS) [28], TcpReplay software [53], and the Smart Detection system, as shown in Figure 9.

In the reprocessing, classification, and evaluation of the traffic during steps 4 and 5, the raw data traffic was replayed by TcpReplay software in a specific OVS port and sampled by the sFlow agent for OVS. The sampled traffic was sent to the Smart Detection system and the classification result was compared with the attack plan. Figure 9 summarizes the procedures carried out by the proposed validation methodology. The raw network traffic file is reprocessed on VM-01, and the sFlow agent collects traffic samples and sends them to Smart Detection on VM-02.

4.2.1. System Setup. The smart Detection system has three main parameters that directly influence its performance. These parameters shown in Table 1 allow the user to calibrate the detection system according to the operating environment. In scenarios where the SR 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 FAR increases because the classifier has few data to analyze. In the case of slow DDoS, low SR and large T_{max} also reduce the attack detection rate due to in-memory flow table expiration time ($E_T = 2$).

So, several experiments to calibrate the system were performed using (i) SR of 1%, 5%, 10%, and 20%; (ii) T_{max} parameter of 25, 50, and 100; and (iii) E_T of 2, 5, and 10 seconds in the testing environment. The most balanced result was obtained with SR = 10%, $T_{max} = 50$, and $E_T = 2$.

4.2.2. Evaluation Metrics. System performance was evaluated using the Precision (PREC), Recall (REC), and F-Measure (F1) metrics present in the literature [54, 55]. PREC measures the ability to avoid false positive, while REC measures system sensitivity. F1 is a harmonic average between PREC and REC. In this context, (i) true positive (TP) is the attack traffic predicted correctly, (ii) true negative (TN) is normal traffic also predicted correctly, (iii) false positive (FP) is the normal traffic predicted incorrectly, and (iv) false negative (FN) is the attack traffic predicted incorrectly. These metrics were computed by the following expressions:

$$\begin{aligned} \text{PREC} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{REC} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ \text{F1} &= 2 \times \frac{\text{PREC} \times \text{REC}}{\text{PREC} + \text{REC}}. \end{aligned} \tag{7}$$

Besides, the detection rate (DR) and false alarm rate (FAR) metrics were used. The DR is the ratio between the number of attacks detected by the system and the actual number of attacks performed. FAR is the ratio between FP

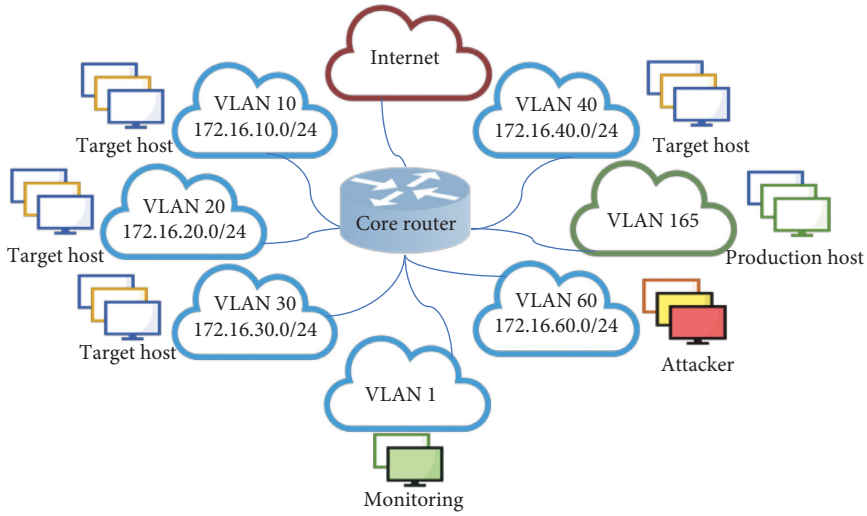

Figure 8: Customized testbed topology.

Table 6: CIC-DoS dataset events and tools.

Attack and tool	Events
ddossim (ddossim)	2
DoS improved GET (Goldeneye [48])	3
DoS GET (hulk [47])	4
Slow body (slowbody2)	4
Slow read (slowread)	2
Slow headers (slowheaders)	5
Rudy (rudy)	4
Slowloris (slowloris)	2

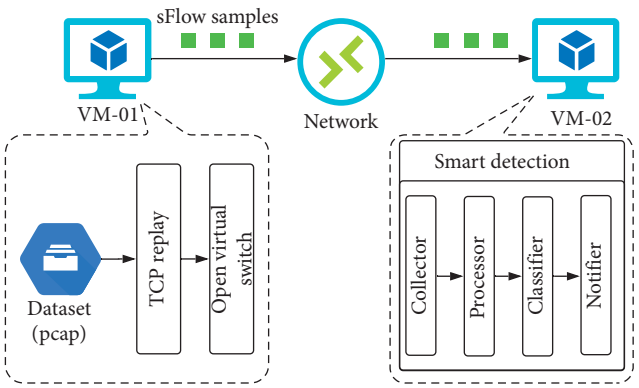

Figure 9: Smart Detection online validation scheme.

and the sum of FP and TN. These metrics were computed by the following expressions:

$$\mathrm{DR} = \frac{A_D}{T_A}, \tag{8}$$

where A_D is the number of detected attacks and T_A is the total number of performed attacks.

$$\mathrm{FAR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}, \tag{9}$$

where FP corresponds to the false-positive classifications and TN is the true-positive classifications.

Table 7: Customized dataset events and tools.

Attack and tool	Events
TCP SYN flood (hping3 [46])	4
TCP SYN flood-*light mode* (hping3 [46])	4
TCP ACK flood (hping3 [46])	4
UDP flood (hping3 [46])	4
UDP flood-*random dst port* (hping3 [46])	4
DoS improved GET (Goldeneye [48])	5
DoS GET (hulk [47])	4
Slow headers (slowhttptest [49])	5
Slow body (slowhttptest [49])	5
Range attack (slowhttptest [49])	4
Slow read (slowhttptest [49])	5

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

4.3. Results and Discussion. The proposed approach has been evaluated using the aforementioned datasets, system setup, and metrics. Table 8 summarizes the system performance for each dataset.

As can be observed, the best performance was obtained in the CSE-CIC-IDS2018 dataset, with a DR of 100%, an FAR of 0.000%, and a PREC of 100%. During the analysis, there was a low occurrence of normal network traffic and well-defined bursts of malicious traffic. This type of behavior facilitates detection by the system and justifies the high hit rates achieved. However, a slightly lower performance was obtained in the customized dataset and CIC-DoS dataset, with a DR of 96.5% and 93.6%, an FAR of 0.2% and 0.04%, and a PREC of 99.5% and 99.9%, respectively. In those datasets, there is a higher volume of normal traffic and various types of attacks, including stealth application layer attacks. In this more realistic scenario, the proposed system presented some detection failures, but still obtained a competitive performance. On the other hand, the worst result was obtained with the CICIDS2017 dataset with 80%

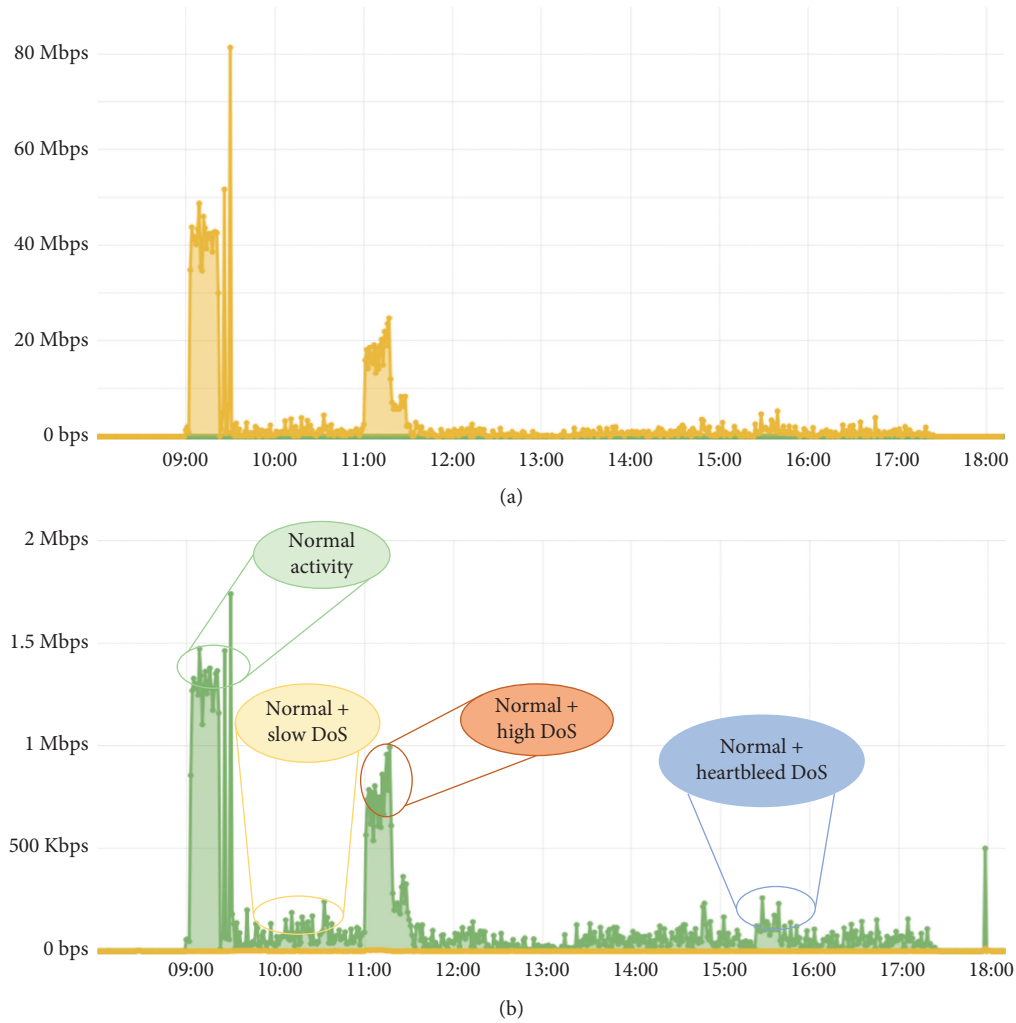

Figure 10: Network traffic in online experiment using the CICIDS2017 dataset. (a) Overall network traffic. (b) Sampled network traffic.

DR, 2% FAR, and 99.2% PREC. This dataset expresses a more realistic network scenario, which includes normal traffic mixed with high-volume and low-volume malicious traffic with sneaky behavior, such as slow application layer attacks. Even so, the proposed system detected 4 out of 5 attacks with PREC greater than 90% and FAR less than 1%, showing that the method is feasible.

To discuss online detection and consumption of computing resources during experimentation, the CICIDS2017 dataset was chosen because it is quite realistic, recent, and summarizes the major vectors of DoS attacks. Even in the most adverse scenario, the experiment was completed normally, as shown in Figures 10 and 11. Overall network traffic is demonstrated in Figure 10(a), while Figure 10(b) highlights the sampled traffic sent to the detection system. As can be seen, for network traffic of 81.3 Mbps, the detection system receives only 1.74 Mbps, making this approach scalable. Overall traffic rating is shown in Figure 11(a), while Figure 11(b) exclusively highlights malicious traffic rating. It can be said that the system was efficient in distinguishing the normal traffic from the DoS attacks because of all the attacks performed, only the Heartbleed attack was not detected, highlighted in Figure 10(b) between 15 h and 16 h. This kind of attack is primarily intended to collect data by exploiting OpenSSL software vulnerabilities as described in CVE-2014-0160, although it can also assume the behavior of a DDoS attack, as in any application. However, in this case, the system raised a false negative. The most obvious reasons for this FN are (i) the execution of the Heartbleed attack without DoS exploitation, or (ii) statistical coincidence in traffic sampling. In the first case, the attack is performed using legitimate and regular connections, while in the second case, the collected samples coincide with legitimate traffic signatures.

In terms of resource use, the system remained stable during the experiment, as shown in Figure 11(c), with small swings in CPU usage.

Finally, the Smart Detection system was tested using online network traffic in four distinct scenarios. The results presented in Table 8 show that the system can distinguish legitimate traffic from various types of DoS/DDoS attacks, such as TCP flood, UDP flood, HTTP flood, and HTTP slow,

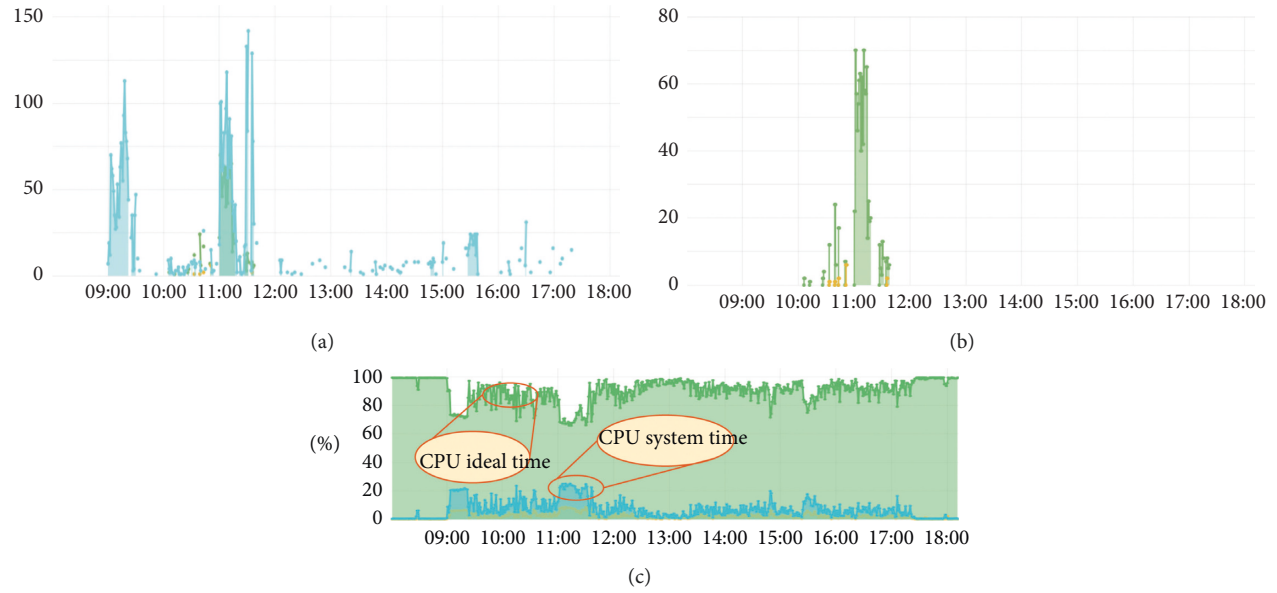

Figure 11: Traffic classification and CPU usage in online experiment using the CICIDS2017 dataset. (a) Overall traffic rating. (b) Malicious traffic rating. (c) Detection system CPU utilization.

with significant rates of accuracy. The experiments also highlighted the importance of adjusting the T_{max} and E_T parameters. These variables correlate with the network traffic sampling rate (SR) and directly influence the detection rate and system accuracy.

4.3.1. Additional Comparison. Compared with some recent similar works available in the literature, the approach introduced in this work is quite competitive in terms of the evaluated performance metrics, as shown in Table 9.

The comparison is not completely fair because the experimental scenarios and data were slightly different, but it is sufficient to allow an evaluation of the obtained results. For instance, in the offline experiments performed with the CIC-DoS dataset in [16], DR was 76.92% using an SR of 20%, while the proposed system obtained an online DR of 90% for the attacks with a FAR of 1.8% using the same sampling technique. In [37], a PREC of 82.1% was obtained using the CICIDS2017 dataset in the offline and unsampled analysis. In this work, the proposed method obtained a PREC of 99.9%, which can be considered competitive for an online detection system based on samples of network traffic. Besides that, in the CICIDS2017 dataset experiments, where the legitimate traffic rate is similar to that of attack traffic, according to Figures 10(b), 11(a), and 11(b), the system was also able to distinguish malicious traffic from normal traffic, such as studied in the lecture [34].

5. Conclusion

This article has presented the Smart Detection system, an online approach to DoS/DDoS attack detection. The software uses the Random Forest Tree algorithm to classify network traffic based on samples taken by the sFlow protocol directly from network devices. Several experiments were performed to calibrate and evaluate system performance. Results showed that the proposed method is feasible and presents improved performance when compared with some recent and relevant approaches available in the literature.

Table 8: Evaluation of system performance for all datasets.

Dataset	DR	FAR	PREC	F1
CIC-DoS	0.936	0.0004	0.999	0.999
CICIDS2017	0.800	0.002	0.992	0.992
CSE-CIC-IDS2018	1.000	0.000	1.000	1.000
Customized	0.965	0.002	0.995	0.995

Table 9: Comparison with research approaches of related works.

Work	Dataset	DR	FAR	PREC
[16]	CIC-DoS	0.7690	N/A	N/A
[37]	CICIDS2017	N/A	N/A	0.8210
The proposed approach	CIC-DoS	0.9360	0.0004	0.9990
The proposed approach	CICIDS2017	0.8000	0.0020	0.9920

The proposed system was evaluated based on three intrusion detection benchmark datasets, namely, CIC-DoS, CICIDS2017, and CSE-CIC-IDS2018, and was able to classify various types of DoS/DDoS attacks, such as TCP flood, UDP flood, HTTP flood, and HTTP slow. Furthermore, the performance of the proposed method was compared against recent and related approaches. Based on the experimental results, the Smart Detection approach delivers improved DR, FAR, and PREC. For example, in the CIC-DoS and CSE-CIC-IDS2018 datasets, the proposed system acquired DR and PREC higher than 93% with FAR less than 1%. Although the system has achieved significant results in its scope, it needs some improvements, such as a better hit rate among attack classes and an automatic parameter calibration mechanism that maximizes the detection rate of attacks.

Future works include analysis of DDoS attacks based on the vulnerabilities of services such as Heartbleed and web brute force attack, enhancement in the multiple-class classification, self-configuration of the system, developing methods for correlating triggered alarms, and formulating protective measures.

Data Availability

We produced a customized dataset and a variable selection algorithm and used four additional datasets to support the findings of this study. The customized dataset used to support the findings of this study has been deposited in the IEEE Data Port repository (https://doi.org/10.24433/CO.0280398.v2). The feature selection algorithm used to support the findings of this study has been deposited in the Code Ocean repository (https://doi.org/10.24433/CO.0280398.v2). The benchmark datasets used to support the findings of this study have been deposited in the Canadian Institute for Cybersecurity repository publicly available as follows: (1) the ISCXIDS2012 dataset (https://www.unb.ca/cic/datasets/ids.html), (2) the CIC-DoS dataset (https://www.unb.ca/cic/datasets/dos-dataset.html), (3) the CICIDS2017 dataset (http://www.unb.ca/cic/datasets/IDS2017.html), and (4) the CSE-CIC-IDS2018 dataset (https://www.unb.ca/cic/datasets/ids-2018.html).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Funding

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brasil (CAPES), finance Code 001.

Acknowledgments

The authors would like to acknowledge the Digital Metropolis Institute (IMD/UFRN) and High-Performance Computing Center of UFRN (NPAD/UFRN) for the overall support given to this work and the Canadian Institute of Cybersecurity (CIC/UNB) for publicly sharing the datasets.

References

[1] D. Anstee, C. F. Chui, P. Bowen, and G. Sockrider, *Worldwide Infrastructure Security Report*, Arbor Networks Inc., Westford, MA, USA, 2017.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: a survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDOS) flooding attacks," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[4] A. Marzano, D. Alexander, O. Fonseca et al., "The evolution of bashlite and mirai IoT botnets," in *Proceedings of the 2018 IEEE Symposium on Computers and Communications (ISCC)*, 2018.

[5] S. Kottler, "February 28th DDoS incident report," 2018, https://github.blog/2018-03-01-ddos-incident-report/.

[6] 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. 66641–66648, 2018.

[7] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[8] K. Singh, P. Singh, and K. Kumar, "Application layer HTTP-GET flood DDoS attacks: research landscape and challenges," *Computers & Security*, vol. 65, pp. 344–372, 2017.

[9] N. Vlajic and D. Zhou, "IoT as a land of opportunity for DDoS hackers," *Computer*, vol. 51, no. 7, pp. 26–34, 2018.

[10] S. Newman, "Under the radar: the danger of stealthy DDoS attacks," *Network Security*, vol. 2019, no. 2, pp. 18-19, 2019.

[11] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, 2014.

[12] Y. Wang, L. Liu, B. Sun, and Y. Li, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," in *Proceedings of the 2015 6th IEEE International Conference on Software Engineering and Service Science, ICSESS 2015*, pp. 1034–1037, Kochi, India, 2015.

[13] M. Masdari and M. Jalali, *A Survey and Taxonomy of DoS Attacks in Cloud Computing*, 2016, http://onlinelibrary.wiley.com/doi/10.1002/sec.1539/epdf.

[14] R. Singh, A. Prasad, R. Moven, and H. Samra, "Denial of service attack in wireless data network: a survey. devices for integrated circuit," in *Proceedings of the 2017 Devices for Integrated Circuit (DevIC)*, pp. 23-24, Kalyani, India, March 2017.

[15] A. Praseed and P. Santhi Thilagam, "DDoS attacks at the application layer: challenges and research perspectives for safeguarding web applications," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 661–685, 2019.

[16] 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.

[17] H. Kim, T. Benson, A. Akella, and N. Feamster, "The evolution of network configuration: a tale of two campuses,"in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference IMC '11*, ACM, p. 499, New York, NY, USA, 2011.

[18] T. V. Phan and M. Park, "Efficient distributed denial-of-service attack defense in sdn-based cloud," *IEEE Access*, vol. 7, pp. 18701–18714, 2019.

[19] Y. Gu, K. Li, Z. Guo, and Y. Wang, "Semi-supervised K-means DDoS detection method using hybrid feature selection algorithm," *IEEE Access*, vol. 7, pp. 64351–64365, 2019.

[20] M. Hasan, M. Milon Islam, I. Islam, and M. M. A. Hashem, "Attack and anomaly detection in IoT sensors in IoT sites using machine learning Approaches,"Internet of Things, vol. 7, Article ID 100059, 2016, https://linkinghub.elsevier.com/retrieve/pii/S2542660519300241.

[21] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019, https://ieeexplore.ieee.org/document/8629941/.

[22] X. Tan, S. Su, Z. Huang et al., "Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm," *Sensors (Switzerland)*, vol. 19, no. 1, 2019.

[23] J. Cheng, M. Li, X. Tang, V. S. Sheng, Y. Liu, and W. Guo, "Flow correlation degree optimization driven random forest for detecting DDoS attacks in cloud computing," *Security and Communication Networks*, vol. 2018, Article ID 6459326, 14 pages, 2018.

[24] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing intrusion detection Systems," *International Journal of Engineering & Technology*, vol. 7, pp. 479–482, December, 2018, https://www.researchgate.net/publication/329045441.

[25] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the ICISSP 2018—4th International Conference on Information Systems Security and Privacy No. Cic*, pp. 108–116, Madeira, Portugal, 2018.

[26] R. Kohavi, "A study of cross–validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (II)*, pp. 1137–1143, 1995, https://www.ijcai.org/Proceedings/95-2/Papers/016.pdf.

[27] S. Behal and K. Kumar, "Trends in validation of DDoS research," *Procedia Computer Science*, vol. 85, pp. 7–15, 2016.

[28] B. Pfaff, J. Pettit, T. Koponen et al., "The design and implementation of Open vSwitch," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pp. 117–130, USENIX Association, Oakland, CA, USA, 2015, https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/pfaff.

[29] J. Le, R. Giller, Y. Tatsumi, H. Huang, J. Ma, and N. Mori, *Case Study Developing DPDK-Accelerated Virtual Switch Solutions for Cloud Service Providers*, Intel Corporation, Santa Clara, CA, USA, 2019, https://builders.intel.com/docs/cloudbuilders/developing-dpdk-accelerated-virtual-switch-solutions-for-cloud-service-providers.pdf.

[30] W. Meng, W. Li, C. Su, J. Zhou, and R. Lu, "Enhancing trust management for wireless intrusion detection via traffic sampling in the era of big data," *IEEE Access*, vol. 6, pp. 7234–7243, 2017.

[31] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and big heterogeneous data: a survey," *Journal of Big Data*, vol. 2, no. 1, 2015.

[32] R. K. C. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 42–51, 2002.

[33] S. Simpson, S. N. Shirazi, A. Marnerides, S. Jouet, D. Pezaros, and D. Hutchison, "An inter-domain collaboration scheme to remedy DDoS attacks in computer networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 879–893, 2018.

[34] S. Behal, K. Kumar, and M. Sachdeva, "D-face: an anomaly based distributed approach for early detection of DDoS attacks and flash events," *Journal of Network and Computer Applications*, vol. 111, pp. 49–63, 2018.

[35] C. Wang, T. T. N. Miu, X. Luo, and J. Wang, "SkyShield: a sketch-based defense system against application layer DDoS attacks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 3, pp. 559–573, 2018.

[36] Z. Liu, Y. Cao, M. Zhu, and W. Ge, "Umbrella: enabling ISPs to offer readily deployable and privacy-preserving DDoS prevention services," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1098–1108, 2019.

[37] 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*, 2019.

[38] R. E. Jurga, R. E. Jurga, M. M. Hulb, M. M. Hulb, H. P. Procurve, and H. P. Procurve, *Technical Report Packet Sampling for Network Monitoring*, 2007.

[39] P. Phaal and M. Lavine, *sFlow version 5*, InMon Corp, San Francisco, CA, USA, 1981, https://sflow.org/sflow_version_5.txt.

[40] J. Postel, *Internet Protocol*, RFC Editor, 1981, https://www.rfc-editor.org/info/rfc0791.

[41] J. Postel, "Transmission Control Protocol," RFC Editor, 1981, https://www.rfc-editor.org/info/rfc0793.

[42] J. Postel, "User datagram protocol," RFC Editor, 1980, https://www.rfc-editor.org/info/rfc768.

[43] B. Claise, Ed., *Cisco systems netflow services export version 9*, RFC Editor, 2004, https://www.rfc-editor.org/info/rfc3954.

[44] R. J. Hyndman and Y. Fan, "Sample quantiles in statistical packages," *Statistical Computing*, vol. 50, no. 4, pp. 361–365, 1996.

[45] 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.

[46] S. Sanfilippo, N. Jombart, D. Ducamp, Y. Berthier, and S. Aubert, "Hping," september 2019, http://www.hping.org.

[47] A. Grafov, "Hulk DoS tool," 2015, https://github.com/grafov/hulk.

[48] J. Seidl, "GoldenEye HTTP DoS test tool," 2012, https://github.com/jseidl/GoldenEye.

[49] S. Shekyan, "SlowHTTPTest," 2011, https://github.com/shekyan/slowhttptest.

[50] S. Ganapathy, K. Kulothungan, S. Muthurajkumar, M. Vijayalakshmi, P. Yogesh, and A. Kannan, "Intelligent feature selection and classification techniques for intrusion detection in networks: a survey," *EURASIP Journal on Wireless Communications and Networking*, vol. 2013, no. 1, p. 271, 2013, http://jwcn.eurasipjournals.com/content/2013/1/271.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort et al., "Scikit-learn: machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2012, http://dl.acm.org/citation.cfm?id=2078195.

[52] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[53] A. Turner, "Tcpreplay–Pcap editing and replaying utilities," 2013, https://tcpreplay.appneta.com/.

[54] S. H. Park, J. M. Goo, and C. H. Jo, "Receiver operating characteristic (ROC) curve: practical review for radiologists," *Korean Journal of Radiology*, vol. 5, no. 1, p. 11, 2004.

[55] M. W. 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.