

An intelligent system to detect slow denial of service attacks in software-defined networks

Prathima Mabel John, Rama Mohan Babu Kasturi Nagappasetty

Department of Information Science and Engineering, Dayananda Sagar College of Engineering, Visvesvaraya Technological University, Bengaluru, India

Article Info

Article history:

Received May 18, 2022

Revised Oct 12, 2022

Accepted Dec 2, 2022

Keywords:

Machine learning

Multi-layer perceptron

Slow DoS attack

Slowloris

Software-defined network

ABSTRACT

Slow denial of service attack (DoS) is a tricky issue in software-defined network (SDN) as it uses less bandwidth to attack a server. In this paper, a slow-rate DoS attack called Slowloris is detected and mitigated on Apache2 and Nginx servers using a methodology called an intelligent system for slow DoS detection using machine learning (ISSDM) in SDN. Data generation module of ISSDM generates dataset with response time, the number of connections, timeout, and pattern match as features. Data are generated in a real environment using Apache2, Nginx server, Zodiac FX OpenFlow switch and Ryu controller. Monte Carlo simulation is used to estimate threshold values for attack classification. Further, ISSDM performs header inspection using regular expressions to mark flows as legitimate or attacked during data generation. The proposed feature selection module of ISSDM, called blended statistical and information gain (BSIG), selects those features that contribute best to classification. These features are used for classification by various machine learning and deep learning models. Results are compared with feature selection methods like Chi-square, T-test, and information gain.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Prathima Mabel John

Department of Information Science and Engineering, Dayananda Sagar College of Engineering

Visvesvaraya Technological University

Kumarswamy Layout, Bengaluru-560078, India

Email: prathimamabel-ise@dayanandasagar.edu

1. INTRODUCTION

Computer networks have enormously evolved over the years to enable high standards of data communication in science and technology. The amount of data that needs to be handled by the network is vast. In such a scenario, it is highly desirable to have an easily manageable and programmable network that is vendor independent. The ease of programmable networks has gained the attention of many researchers and data center professionals in recent times. One such architecture that provides a high degree of programmability in networking is software-defined network (SDN). It is a new paradigm of networking where the entire network is divided into two planes: the control plane and the data plane. The data plane is responsible for carrying data via network elements like hosts, switches, routers, and gateways. The behavior of the data plane is controlled by a device called a controller which is present in the control plane.

The controller is the brain of the network and runs algorithms to provide network services. The control plane interfaces with the data plane with the help of a protocol called OpenFlow. OpenFlow acts as the carrier of messages between the control and data planes. It coordinates the functions of the data plane by installing rules in the OpenFlow tables that are maintained in the OpenFlow switches of the data plane. The controller formulates these rules using algorithms. Even though the SDN brings many advantages concerning

programmability, its centralized architecture makes it more vulnerable to attacks. Therefore, it is of the utmost importance to secure the controller.

SDN can be attacked in several ways, as explained in [1]. A compromised control plane is a major risk SDN can face. A common type of attack that can bring SDN down is the distributed denial of service (DDoS) attack. It is a type of attack where the attacker establishes many connections to the server to exhaust its thread pool. This brings the server down and makes it unavailable to other genuine users who want to connect to the server. A DDoS attack can be large and voluminous, like flooding, or low and slow, like a slow DoS attack. This paper deals with the detection of slow DoS attacks on Apache2 and Nginx servers in an SDN network.

Slow DoS attacks are usually application-level attacks that bring the server down when the attacker generates many concurrent connection requests to the server. These connections are kept open indefinitely until the attack is active. The most common slow attacks are Slowloris, R U Dead Yet, and Slow Read. They exploit the hypertext transfer protocol (HTTP) request and response message headers to create attacks. This paper deals with the detection and mitigation of Slowloris attacks. A Slowloris attack is generated by a script that runs on the attacker's machine. It creates enormous open connections by default to a web server and keeps these connections active by sending incomplete HTTP GET messages every 15 seconds. The server waits for the complete request to arrive without knowing that it has been attacked. This makes the server inaccessible to other genuine users.

The proposed method in this paper, called an intelligent system for slow DoS detection using machine learning (ISSDM), detects Slowloris attacks on Apache2 and Nginx servers. ISSDM has a data generation module which monitors and records server parameters such as time to test, the number of concurrent connections, and timeout statistics. The Apache Bench tool (ab) [2] is used to test the server behavior. A Monte Carlo simulation is used for the estimation of the threshold value for the time to test and connection count parameters. Along with monitoring server statistics, it performs header inspection to identify the attack pattern in HTTP GET request headers using regular expression. The parameters generated as a part of data generation module are recorded in a csv file. Parameters of CSV file are used as features for attack classification using certain machine learning (ML) and deep learning models such as support vector machine (SVM), k-nearest neighbors (KNN), decision trees (DT), naïve Bayes (NB), random forest (RF) and multi-layer perceptron (MLP). ISSDM includes a proposed feature selection module called blended statistical and information gain (BSIG). It is used to select the most suitable features for classification and increase the accuracy of detection models. The performance of the detection models trained using BSIG is compared with other feature selection methods such as the Chi-square test, T-test, and information gain. This work is carried out in a laboratory set up with 20 clients, 3 Zodiac FX [3] OpenFlow switches, Apache2 server, Nginx server, and Ryu controller [4]. Data are generated both for legitimate and attack scenarios for training the detection models. Further, models are tested with attacks generated by the Slowloris script on the attacker nodes.

2. LITERATURE SURVEY

This section summarizes the work related to DoS attacks and methods to detect and mitigate them. Research in [5] discusses DDoS, its evolution, its types, and the ways in which a normal system on the internet becomes vulnerable to such vulnerabilities. DDoS depletes the victim's resources by exhausting disk, bandwidth, and other resources, thus making them unavailable to anyone who actually needs the resources. Researchers describe two types of attacks: undetectable and devastating low-rate DDoS, and detectable high-rate DDoS. John and Nagappasetty [6] proposes a slow DoS detection and mitigation method called Slowloris detection and mitigation mechanism (SDMM). Detection used expectation of burst size based on identification of bursts of data during slow DoS attack.

Pascoal *et al.* [7], [8] delve into two types of attacks called slow ternary content-addressable memory (TCAM) exhaustion attack and slow saturation attack. These attacks deplete the TCAM of OpenFlow switches by forcibly installing new forwarding rules. Another type of saturation attack called a table miss striking attack is identified in [9]. It exploits sensitive packet fields to trigger a table miss and initiate unnecessary communication between the control pane and the data plane. SDNGuardian is proposed as a countermeasure to detect such attacks. A new algorithm, MultiQueue, which is designed to protect the controller from DDoS attacks is proposed in [10]. In [11] to detect reduction of quality (RoQ) attacks, the authors used four machine learning algorithms; MLP, K-NN, SVM, and multinomial naïve Bayes (MNB), fuzzy logic (FL) and Euclidean distance (ED). Aamir and Zaidi [12] use unlabeled or partially labeled datasets. Then they cluster them using two different algorithms: agglomerative and K-means with feature extraction under principal component analysis (PCA) clustering.

An ensemble framework for feature selection methods (EnFS) is proposed in [13]. The methods fall into three major categories of feature selection methods: filter-based methods, wrapper-based methods, and

embedded methods which are built-in mechanisms for selecting certain features during model training time. Myint Oo *et al.* [14] have proposed an approach to detect DDOS attacks that adapts advanced SVM that is more efficient than the SVM algorithm. The accuracy of the proposed model is 97%. Ye *et al.* [15] proposed a model for detecting DDoS attacks by using a combination of SVM classification algorithms in SDN. The platform is set by mininet [16] and Floodlight. Osanaiye *et al.* [17] proposed a method that involves an ensemble-based multi-filter feature selection method that adds the output of different filters to reach optimum selection.

Studies [18], [19] used machine learning and feature selection methods to train and test the dataset. ML algorithms such as SVM, NB, artificial neural network (ANN), and KNN are used along with feature selection methods based on thresholds. To identify attacks, Studies [20], [21] used Chi-square test and information gain feature selection mechanisms, as well as NB, SVM, C4.5, K-NN, K-means, fuzzy C-means, and a number of other models. A system that extracts only important attributes from network traffic in a computer network is proposed in [22].

The work in [23] discusses the detection of three types of DDoS attacks, namely controller, bandwidth, and flow-table attacks in SDN networks using machine learning techniques such as SVM, MLP, DT, and RF. The research work in [24] focuses on detecting distributed reflection denial of service (DrDoS) attacks in the internet of things (IoT). It uses a hybrid intrusion detection system (IDS) to detect IoT-DoS attacks, which detects suspicious network traffic from network nodes based on long short-term memory (LSTM). In [25], [26] the application of deep learning in the detection and mitigation of DDoS attacks on SDN controllers using models such as LSTM and convolutional neural network (CNN) is discussed. The models were implemented to detect transmission control protocol (TCP), user datagram protocol (UDP), and internet control message protocol (ICMP) flood attacks. Studies [27], [28] study the performance of emulating SDN and the impact of firewalls on throughput of the network.

The study of existing research work that is presented in this section has unlocked avenues for research in DDoS. This has been a motivation for selecting slow DDoS attacks as a problem for study. The objective is to study the nature of the Slowloris attack and design a method to detect and mitigate it. A variety of machine learning and deep learning models are considered in the proposed work. Their accuracy is compared with that of the models used by researchers in this section.

3. PROPOSED METHOD

The methodology proposed in this paper is called “An intelligent system for slow DoS detection using machine learning (ISSDM)”. It is responsible for generating the data set using the data generation module, selecting features that contribute more to data classification using the feature selection module, and classifying traffic flows using some of the ML and deep learning techniques. The process begins when the data generation module continuously monitors and collects server parameters. These parameters are carefully monitored based on threshold values to observe any variations in the behavior of the server. Threshold values are generated using a Monte Carlo simulation. Further, if parameters exceed the threshold value, ISSDM performs header inspection of the HTTP GET header to look for a match to the attack pattern using regular expressions. If a match is found, the flow is marked as attack traffic in the data set. Using this process, a data set is generated for legitimate and attack traffic. Once the data set is generated, the proposed feature technique called BSIG is used to select the best possible features for traffic classification. Classification is then performed on this set of features using some of ML and deep learning techniques such as SVM, KNN, DT, NB, RF, and MLP. The detection accuracy of the machine learning algorithms using BSIG is compared to other feature selection methods such as Chi-square, T-test, and Information gain. The process of the proposed methodology is shown in Figure 1.

A DoS attack is generated by the Slowloris attack script, which attacks both the servers with 150 simultaneous connections by default. However, because the Nginx server is resistant to Slowloris attack with 150 sockets, the number of sockets is increased to 1250 to ensure a successful attack. In ISSDM, data generation and feature selection using BSIG are the crux of the process. The remaining part of this section explain the two modules and the entire process in detail.

3.1. Data generation

Data traffic is generated for this work using Zodiac FX OpenFlow switch, Ryu controller, Apache2 server, Nginx server, hosts, and the Apache Bench tool (ab). Legitimate traffic is generated using tools like *ping* and *iperf*. Attack traffic is generated using Slowloris. The topology used for the experiment is shown in Figure 2.

The Apache bench works on HTTP servers by sending numerous concurrent requests and measuring the server's response to the offered load. In this work, ab offers a load every 2 seconds to Apache2 and Nginx to study the server's response. The server's response statistics are sent to the Ryu controller, where ISSDM

writes the parameters into the dataset. Apart from server parameters, header inspection is performed at the controller to identify the attack pattern in the HTTP GET header. The following parameters are considered the features of the data set:

- *Time_to_test*: this parameter indicates the amount of time taken for the server to complete the requests of the ab tool. In this experiment, “ab-n 10-c 1 http://localhost:80” is the ab load used to test the server with n (10) http requests and concurrency of c (1). A server with normal load responds with a *time_to_test* value of 0.001 s. When under attack this variable increase by more than 0.01 s. A threshold is chosen for this variable to predict an attack. The threshold for *time_to_test* is chosen by using Monte Carlo simulation.

The Monte Carlo simulation is used when there is uncertainty in estimating a single or average value of a variable. It takes the variable that has uncertainty and makes the closest approximation by assigning random numbers. This process is repeated hundreds of times with different random numbers. On completion of the simulation, the results are averaged to obtain the closest estimate. In this paper, the Monte Carlo simulation was conducted for more than 400 values of *time_to_test* variable when the server was in a normal state as well as an attack state. The final estimation obtained from the simulation was 0.22 s. This was chosen as the threshold value to suspect an attack on the server. The graph of the Monte Carlo model is shown in Figure 3.

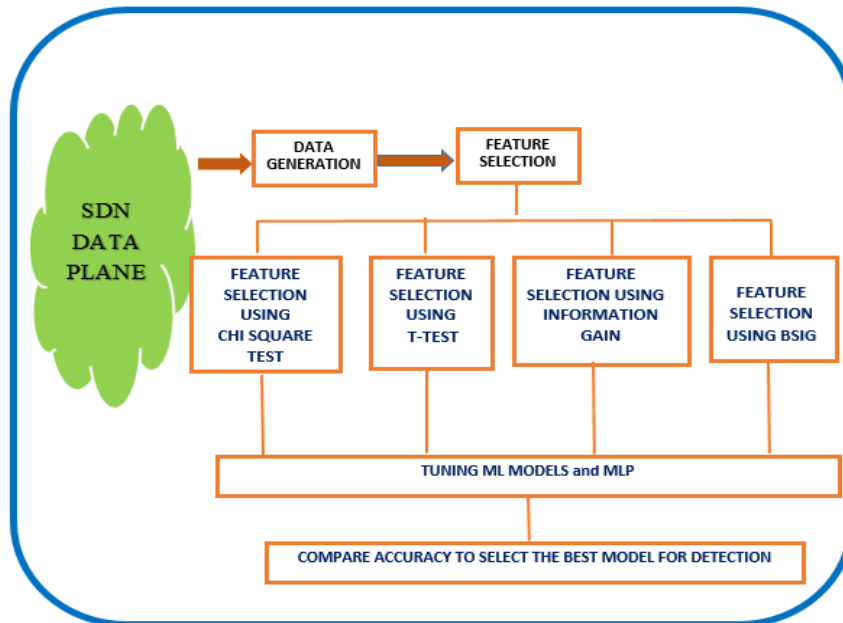


Figure 1. ISSDM model

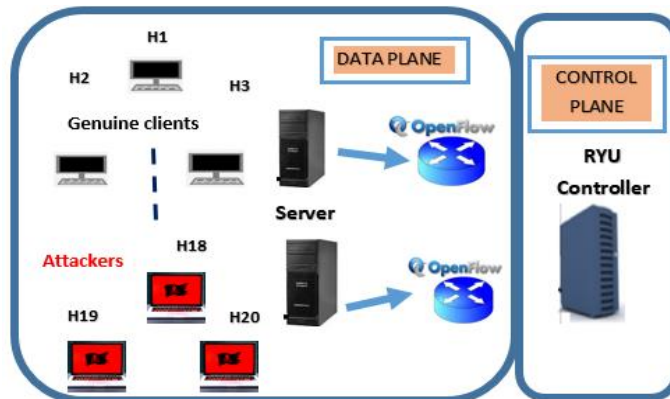


Figure 2. Network topology

- *Timeout*: the ab tests the server response by presenting n requests with concurrency c . If the server does not respond for 30 s, ab times out by default. When a server is under attack, ab times out as it cannot find any thread to get connected to. This property of ab is used as another server parameter to detect a DoS attack on the server. Timeout is set to 1 on ab time out, 0 otherwise.
- *Connection count*: Slowloris attacks open a large number of connections to the server at once to bring it down. The drastic increase in the number of connections from a specific client is considered a server parameter for attack detection. A Monte Carlo simulation was used to consider the data of server in a normal state and an attack state to make an approximation of the average connection count on the server. This value was calculated to be 135 and chosen as the threshold for attack detection. The graph for this simulation is shown in Figure 4.

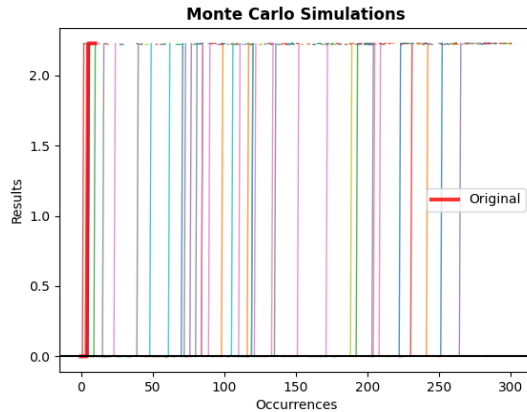


Figure 3. Monte Carlo simulation graph for estimation of *Time_to_test*

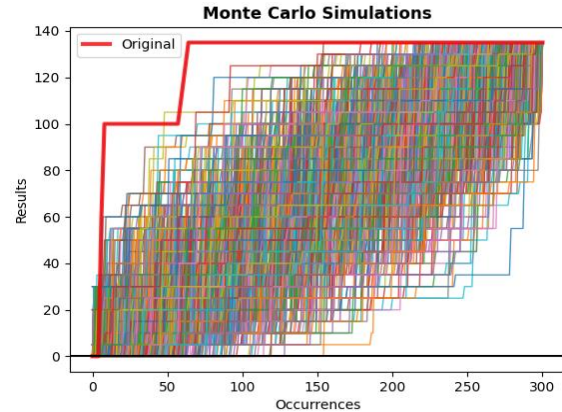


Figure 4. Monte Carlo simulation graph for estimation of *Connection_Count*

- *Pattern match*: This is a Boolean variable considered as a feature of the data set. This parameter is obtained by inspecting the header of the HTTP GET message. The pattern of the request header, which is generated by an attack every 15s becomes a key factor for identifying the attack. The general format of an HTTP GET message includes a header followed by the message body [29]. The message body is separated from the header by a blank line. It means that 2 consecutive carriage return and line feed characters ($\r\n$) must be present after the header. The attack sends incomplete requests every 15s by including only one $\r\n$ after the header instead of two. The server keeps waiting for the complete request to arrive, but indefinitely keeps receiving incomplete requests.

ISSDM uses regular expressions to identify incomplete GET requests. When the *time_to_test* or *connection_count* exceed the threshold, or a timeout occurs on ab, the controller installs flows in the switch to redirect all traffic from the suspected client to the controller. The ISSDM module running in the controller buffers the data and starts inspecting the packet headers which are coming in as byte streams, using the regular expression $r"\r\n\r\n"$. The absence of this pattern in the header is an indication that the blank line is missing at the end of the header. If there is no match for the regular expression, then the Pattern match variable is set to true. If the header does not match the attack pattern, the buffered data are forwarded to the destination after installing a new flow in the switch.

- *Server Type*: indicates the type of server (Apache2 or Nginx).
- *Traffic type*: this feature holds two categorical values, namely: legitimate and attack. It classifies the incoming traffic based on the server parameters and header inspection. The entire process of data generation is shown in Figure 5.

The size of the data set generated is around 400. It includes both the classes (legitimate and attack) of data. This data set is trained and tested using SVM, RF, KNN, NB, DT, and MLP algorithms. The Chi-square test is a feature selection method that tests the relationship between features. It is used when the predictor and response features are categorical. When two variables are given along with their values, the observed count O and the expected count E can be measured with Chi-square using (1).

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i} \quad (1)$$

where, c is the degree of freedom, O is the observed values, and E is the expected values.

In this paper, Chi-square and p values are calculated for the features in the data set generated by the proposed system. For independent variables, the Chi-square value and p value are small. Feature selection is aimed at finding features that are highly dependent on the response. Hence, after the Chi-square test is performed, those features with higher values for the Chi-square statistic and p value are considered for training the machine learning models for prediction.

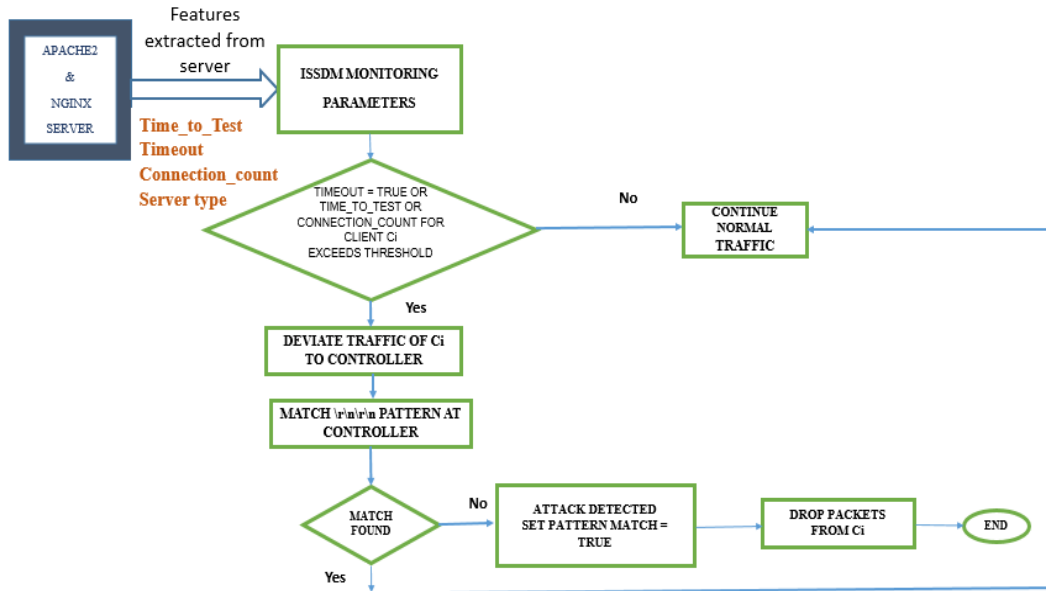


Figure 5. Flowchart of data generation process of ISSDM

A T-test is a statistical test used in hypothesis testing to determine if a process has an effect on the population or if the groups differ from each other. A T-test is used when the means of two groups are to be compared. There are 3 types of T-tests, namely: one sample, when one group is being compared with a standard value; two sample, when groups belong to two different populations; and paired T-test, when a group belongs to a single population. In the proposed method, a paired sample T-test is performed on the individual numeric population of the data set. The T value is calculated using the formula shown (2).

$$t = \frac{\overline{X_D} - \mu_0}{s_D / \sqrt{n}} \tag{2}$$

where, $\overline{X_D}$ is the sample mean of differences, s_D is the standard deviation of differences, n is the sample size, and μ_0 is the population mean.

After T-test, the p value is calculated and features having a p value less than 0.05 (level of significance) are chosen for modeling as they indicate a larger difference in the mean. This plays an important role in identifying an attack. Machine learning models are trained with the selected features and their accuracy is compared.

Information gain (IG): to train a model or to predict or classify between classes, DT are made. The features or attributes of a dataset that are most important in predicting the outcome are chosen. The importance of each attribute needs to be understood before actually computing or modeling, and that's where IG comes into the picture. IG tells us how important an attribute of a feature vector is. This information is then used to decide the ordering of attributes in the decision tree. Thus, an attribute that increases IG, minimizes the entropy or impurity from the dataset, thus giving better prediction. IG provides a way to use entropy to calculate how a change to the dataset impacts the purity of the dataset. IG is calculated by comparing the entropy of the dataset before and after a transformation. For a binary classification, problem entropy is calculated as in (3).

$$\text{Entropy} = ((p(1)*\log(P(1)) - (p(0)*\log(P(0)))) \tag{3}$$

IG is given by (4),

$$IG(S, a) = H(S) - H(S|a) \tag{4}$$

where, $IG(S, a)$ is the information for the dataset S , $H(S)$ is entropy of dataset before any change, and $H(S|a)$ is the conditional entropy for the dataset given the variable a . Information gain is calculated for each feature of the generated data set in this experiment. Features of high importance are considered for training the machine learning models and their accuracy is compared.

BSIG is a feature selection method proposed in this paper which combines the optimal features selected by Chi square, T-test, and IG. It is proposed with the aim of improving the accuracy of the detection models. Let S be the set of all the features of the generated data set.

$$S = (Time_to_test, Timeout, Connection_count, Pattern_match, Server_type, Traffic\ type)$$

A Chi-square test is performed on the categorical features, namely *Timeout*, *Pattern_match*, and *Server_type*. Let the set of features $C \subset S$ be chosen based on the p values. Similarly, the T-test is performed on numerical features such as *time_to_test* and *Connection_count*. Let $T \subset S$ be the chosen subset of features based on the p value obtained from the T-test. Let I be the set of features where $I \subset S$, chosen based on the p value of information gain applied to S . Let $Z = C \cup T \cup I$ be the set of features chosen from Chi square, T-test and information gain. The final set of features having the maximum impact on the prediction are used to train the models. The machine learning models are trained for features in C , T , and I separately to analyze the behavior of each model. Finally, models are trained using features from set Z to determine the best model for Slowloris attack prediction.

4. EXPERIMENT AND RESULT

The topology of the experimental setup is as shown in Figure 2. It consists of 20 clients, out of which 10 are attackers and 10 are legitimate clients; 1 Apache2 server, and 1 Nginx server connected to 2 Zodiac FX switches in the data plane. The control plane consists of a Ryu controller running the ISSDM module. Data generation is the first step toward detection of an attack in this work. As explained previously, both legitimate and attack traffic data are collected for the features *Time_to_test*, *Timeout*, *Connection_count*, *Pattern_match*, *Server_type*, and *Traffic_type*. A Monte Carlo simulation is performed to choose thresholds for *Time_to_test* and *Connection_count*. Detection models such as SVM, KNN, DT, RF, NB, and MLP are trained with features set S . The data generated by attackers using the Slowloris tool is used to test these models for attack detection. The accuracy and F-score for this detection using feature set S are shown in Table 1 along with the receiver operating curve (ROC) in Figure 6.

Table 1. Accuracy and F-score of models trained with feature set S

Algorithm	Accuracy	F-score
SVM	0.93	0.96
KNN	0.89	0.94
DT	0.85	0.91
NB	0.89	0.94
RF	1	1
MLP	0.85	0.91

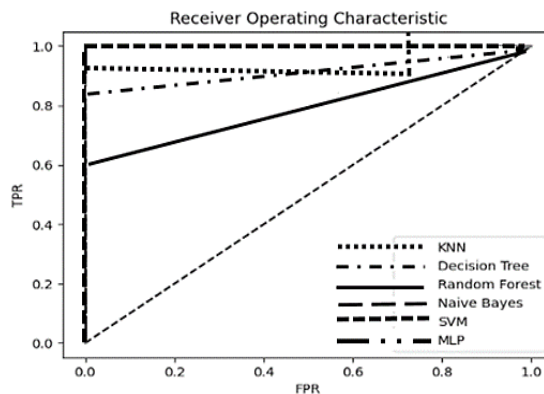


Figure 6. ROC curve of models trained with feature set S

The accuracy of a classification model is the ratio of the number of correct predictions to the total number of input samples as given by (5).

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Sample}} \quad (5)$$

F-score indicates how precise the classifier is i.e., how many instances it classifies correctly. It given by (6).

$$F - \text{score} = 2 * \frac{1}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} \quad (6)$$

where,

$$\text{Precision} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \quad (7)$$

$$\text{Recall} = \frac{\text{True positives}}{\text{True positives} + \text{False Negatives}} \quad (8)$$

Feature selection using the Chi-square test: the categorical features of S , namely *timeout*, *Pattern_match*, and *Server_type*, are subject to the Chi-square test to obtain the Chi-square statistics and p value. Table 2 shows the result of Chi-square test on the categorical features. According to the Chi-square test result, features with a higher p value are chosen because the class variable *Traffic_type* is considered to be more dependent on features with a higher p value. *Pattern_match* and *Timeout* are chosen as the elements of set C . Table 3 and Figure 7 show the accuracy, F-score, and ROC curve indicating true positive rate (TPR) and false positive rate (FPR) for models trained with feature C .

Table 2. Chi-Square test statistics

Feature	Chi-square statistic	p value
Timeout	2.72727273e-01	6.01508134e-01
Pattern_match	1.00000000e+00	3.17310508e-01
Server_Type	8.90909091e-01	3.45231072e-01

Table 3. Accuracy and F-score of models trained with feature set C

Algorithm	Accuracy	F-score
SVM	0.89	0.94
KNN	0.89	0.94
DT	1	1
NB	0.89	0.94
RF	1	1
MLP	1	1

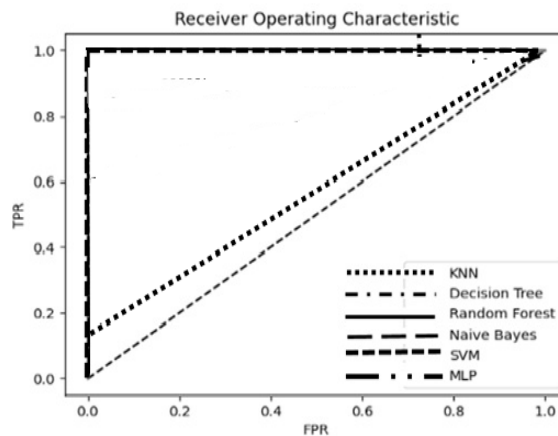


Figure 7. ROC curve of models trained with feature set C

Feature selection using T-test: The numerical features of S , namely $Time_to_test$ and $Connection_count$, are subject to T-test to obtain the statistics and p value. Table 4 shows the result of the T test on the chosen numerical features. Features with a higher p value are chosen for set T as a higher p value implies higher correlation between the dependent and independent variable. Here, set T contains $Connection_count$ as its element. Table 5 and Figure 8 show the accuracy, F-score, and ROC curve for models trained with feature T .

Table 4. T-test statistics

Feature	T-test statistic	p value
Time_to_test	-10.99497051	2.07390283e-10
Connection_count	5.04253005	4.75313476e-05

Table 5. Accuracy and F-score of models trained with feature set T

Algorithm	Accuracy	F-score
SVM	0.85	0.91
KNN	0.89	0.94
DT	0.85	0.91
NB	0.85	0.91
RF	0.85	0.91
MLP	0.85	0.91

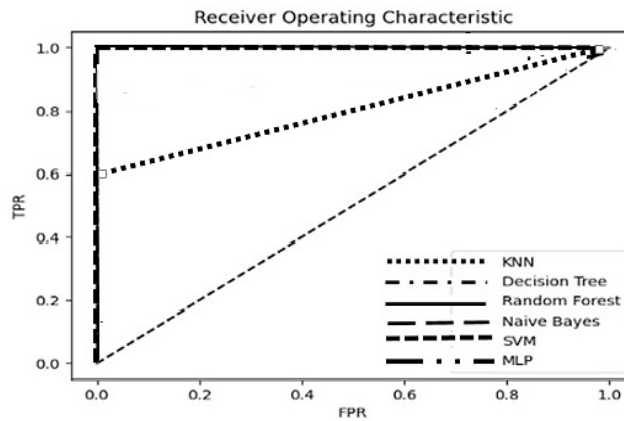


Figure 8. ROC curve of models trained with feature set T

Feature selection using IG: All the features of S are subject to IG to obtain the statistics. Table 6 shows the result of IG. IG calculates the importance of each independent feature with respect to the dependent feature. Therefore, features with higher values are considered to be more significant. $Pattern_match$ and $Connection_count$ are the features that have higher IG statistics. Hence, set $I=(Pattern_match, Connection_count)$. Table 7 and Figure 9 show the accuracy, F-score, and ROC curve for models trained with feature T .

Feature selection using BSIG: the proposed method BSIG uses the features selected by Chi-square, T-test and IG to train the models. From the previous observations, we have: i) C is $(Pattern_match, Timeout)$, ii) T is $(Connection_count)$, iii) I is $(Pattern_match, Connection_count)$, iv) Z is $C \cup T \cup I$ is $(Pattern_match, Timeout, Connection_count)$. Table 8 and Figure 10 show the accuracy, F-score, and ROC curve for models trained with feature T .

Table 6. IG statistics

Feature	IG statistics
Timeout	0.098217
Pattern_match	0.475789
Server_Type	0.016875
Time_to_test	0.016875
Connection_count	0.302149

Table 7. Accuracy and F-score of models trained with feature set *I*

Algorithm	Accuracy	F-score
SVM	0.85	0.91
KNN	0.89	0.94
DT	0.85	0.91
NB	0.85	0.91
RF	1	1
MLP	0.85	0.91

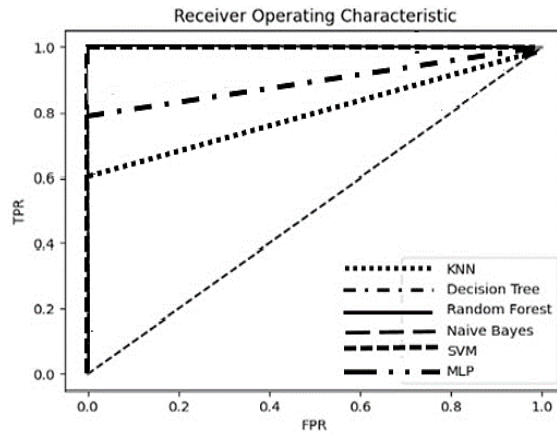


Figure 9. ROC curve of models trained with feature set *I*

Table 8. Accuracy and F-score of models trained with feature set *Z*

Algorithm	Accuracy	F-score
SVM	1	1
KNN	0.89	0.94
DT	0.85	0.91
NB	1	1
RF	1	1
MLP	1	1

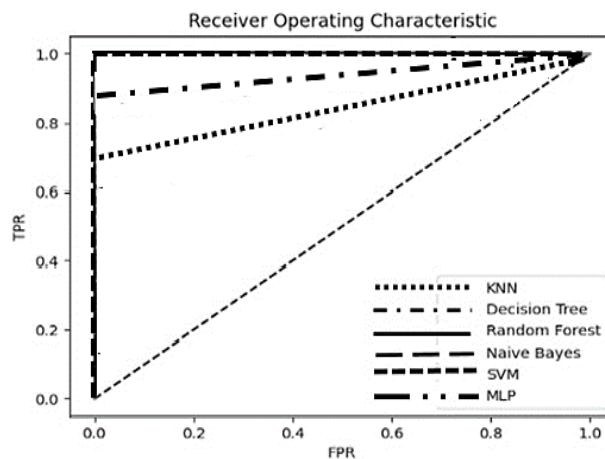


Figure 10. ROC curve of models trained with feature set *Z*

The results shown above indicate the performance of models with different features, obtained from various feature selection techniques. The results are summarized in Figure 11. The accuracy of SVM, NB, RF, and MLP increases to 100% by using BSIG when compared to the accuracy of models using the entire feature set *S*. Therefore, it can be concluded that Slowloris can be detected with greater accuracy by using BSIG as a feature selection technique for SVM, NB, RF and MLP. BSIG keeps KNN’s accuracy consistent

with other feature selection techniques. MLP performs well for the Chi-square selection method as well. DT offers the best results when the Chi-square test is used for feature selection, while being consistently less when other methods are considered for detection of Slowloris attack.

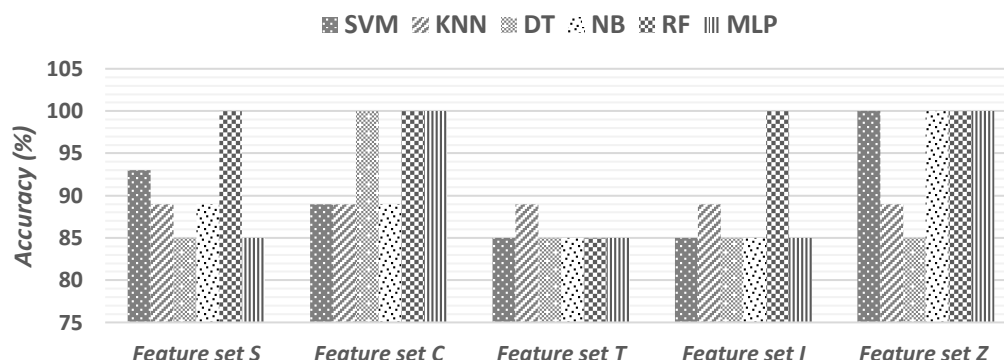


Figure 11. Comparison of detection accuracy obtained by using feature set of proposed feature selection method with other feature selection methods

5. CONCLUSION

This research focuses on detecting a common type of slow DDoS attack known as Slowloris. The proposed method, called ISSDM, is responsible for detecting and mitigating the Slowloris attack on Apache2 and Nginx servers. It gathers data from Apache2 and Nginx servers along with an HTTP GET request header to generate the data set for legitimate and attack traffic. Monte Carlo simulations are performed to find the threshold of certain server parameters during data generation to segregate attack and genuine flows. A feature selection method called BSIG is proposed to select the most appropriate features for traffic classification. The features extracted by BSIG are used to train ML models such as SVM, KNN, DT, NB, and RF, along with a deep learning technique called MLP. This is done to detect attacks. The accuracy of the models is compared by training them with different feature selection methods like Chi-square, T-test, and information gain and the proposed feature selection method named BSIG. It was observed that SVM, NB, RF, and MLP performed extremely well with 100% accuracy while detecting Slowloris attacks using BGIS. This work can be extended further to detect different types of attacks.




REFERENCES

- [1] J. P. Mabel, K. Vani, and K. R. M. Babu, "SDN security: challenges and solutions," in *Emerging Research in Electronics, Computer Science and Technology*, Springer, 2019, pp. 837–848.
- [2] Apache, "Apache HTTP server benchmarking tool," HTTP Server Project. <https://httpd.apache.org/docs/2.4/programs/ab.html> (accessed Oct. 20, 2022).
- [3] P. Zanna, "Zodiac FX-the world's smallest, most affordable openflow SDN switch is now on kickstarter," *Northbound Networks*. <https://northboundnetworks.com/pages/zodiac-fx-the-world-s-smallest-most-affordable-openflow-sdn-switch-is-now-on-kickstarter> (accessed Sep. 10, 2021).
- [4] Ryu, "Ryu SDN framework." Ryu SDN Framework (ryu-sdn.org). <https://ryu-sdn.org/> (accessed June 30, 2022).
- [5] M. P. Singh and A. Bhandari, "New-flow based DDoS attacks in SDN: taxonomy, rationales, and research challenges," *Computer Communications*, vol. 154, pp. 509–527, Mar. 2020, doi: 10.1016/j.comcom.2020.02.085.
- [6] P. M. John and R. M. B. K. Nagappasetty, "An approach for slow distributed denial of service attack detection and alleviation in software defined networks," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 25, no. 1, pp. 404–413, Jan. 2022, doi: 10.11591/ijeecs.v25.i1.pp404-413.
- [7] T. A. Pascoal, Y. G. Dantas, I. E. Fonseca, and V. Nigam, "Slow TCAM exhaustion DDoS attack," in *ICT Systems Security and Privacy Protection*, Springer International Publishing, 2017, pp. 17–31.
- [8] T. A. Pascoal, I. E. Fonseca, and V. Nigam, "Slow denial-of-service attacks on software defined networks," *Computer Networks*, vol. 173, May 2020, doi: 10.1016/j.comnet.2020.107223.
- [9] J. Xu, L. Wang, and Z. Xu, "An enhanced saturation attack and its mitigation mechanism in software-defined networking," *Computer Networks*, vol. 169, Mar. 2020, doi: 10.1016/j.comnet.2019.107092.
- [10] Q. Yan, Q. Gong, and F. R. Yu, "Effective software-defined networking controller scheduling method to mitigate DDoS attacks," *Electronics Letters*, vol. 53, no. 7, pp. 469–471, Mar. 2017, doi: 10.1049/el.2016.2234.
- [11] V. de M. Rios, P. R. M. Inácio, D. Magoni, and M. M. Freire, "Detection of reduction-of-quality DDoS attacks using Fuzzy Logic and machine learning algorithms," *Computer Networks*, vol. 186, Feb. 2021, doi: 10.1016/j.comnet.2020.107792.
- [12] M. Aamir and S. M. Ali Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 4, pp. 436–446, May 2021, doi: 10.1016/j.jksuci.2019.02.003.




- [13] S. Das, D. Venugopal, S. Shiva, and F. T. Sheldon, "Empirical evaluation of the ensemble framework for feature selection in DDoS attack," in *2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, Aug. 2020, pp. 56–61, doi: 10.1109/CSCloud-EdgeCom49738.2020.00019.
- [14] M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, and S. Vasupongayya, "Advanced support vector machine-(ASVM-) based detection for distributed denial of service (DDoS) attack on software defined networking (SDN)," *Journal of Computer Networks and Communications*, pp. 1–12, Mar. 2019, doi: 10.1155/2019/8012568.
- [15] J. Ye, X. Cheng, J. Zhu, L. Feng, and L. Song, "A DDoS attack detection method based on SVM in software defined network," *Security and Communication Networks*, pp. 1–8, 2018, doi: 10.1155/2018/9804061.
- [16] ONF, "Mininet," *The Open Networking Foundation (ONF)*. <https://opennetworking.org/mininet/> (accessed Mar. 23, 2022).
- [17] O. Osanaiye, H. Cai, K.-K. R. Choo, A. Dehghantaha, Z. Xu, and M. Dlodlo, "Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing," *EURASIP Journal on Wireless Communications and Networking*, no. 1, Dec. 2016, doi: 10.1186/s13638-016-0623-3.
- [18] H. Polat, O. Polat, and A. Cetin, "Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models," *Sustainability*, vol. 12, no. 3, Feb. 2020, doi: 10.3390/su12031035.
- [19] S. Hosseini and M. Azizi, "The hybrid technique for DDoS detection with supervised learning algorithms," *Computer Networks*, vol. 158, pp. 35–45, Jul. 2019, doi: 10.1016/j.comnet.2019.04.027.
- [20] O. S. Akanji, O. A. Abisoye, and M. A. Iliyasu, "Mitigating slow hypertext transfer protocol distributed denial of service attacks in software defined networks," *Journal of Information and Communication Technology*, vol. 20, no. 3, pp. 277–304, Jun. 2021, doi: 10.32890/jict2021.20.3.1.
- [21] M. Suresh and R. Anitha, "Evaluating machine learning algorithms for detecting DDoS attacks," in *International Conference on Network Security and Applications*, 2011, pp. 441–452.
- [22] W. Wang and S. Gombault, "Efficient detection of DDoS attacks with important attributes," in *2008 Third International Conference on Risks and Security of Internet and Systems*, Oct. 2008, pp. 61–67, doi: 10.1109/CRISIS.2008.4757464.
- [23] R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno, "Machine learning algorithms to detect DDoS attacks in SDN," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 16, Aug. 2020, doi: 10.1002/cpe.5402.
- [24] M. Shurman, R. Khrais, and A. Yateem, "DoS and DDoS attack detection using deep learning and IDS," *The International Arab Journal of Information Technology*, vol. 17, no. 4A, pp. 655–661, Jul. 2020, doi: 10.34028/iajit/17/4A/10.
- [25] J. D. Gadze, A. A. Bamfo-Asante, J. O. Agyemang, H. Nunoo-Mensah, and K. A.-B. Opare, "An Investigation into the application of deep learning in the detection and mitigation of DDOS attack on SDN controllers," *Technologies*, vol. 9, no. 1, Feb. 2021, doi: 10.3390/technologies9010014.
- [26] D. Tang, L. Tang, W. Shi, S. Zhan, and Q. Yang, "MF-CNN: a new approach for LDoS attack detection based on multi-feature fusion and CNN," *Mobile Networks and Applications*, vol. 26, no. 4, pp. 1705–1722, 2021, doi: 10.1007/s11036-019-01506-1.
- [27] T. A. Assegie, "Performance analysis of emulated software defined wireless network," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 16, no. 1, pp. 311–318, Oct. 2019, doi: 10.11591/ijeecs.v16.i1.pp311-318.
- [28] M. H. H. Khairi, S. H. S. Ariffin, N. M. Abdul Latiff, K. Mohamad Yusof, M. K. Hassan, and M. Rava, "The impact of firewall on TCP and UDP throughput in an openflow software defined network," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 20, no. 1, pp. 256–263, Oct. 2020, doi: 10.11591/ijeecs.v20.i1.pp256-263.
- [29] R. Fielding *et al.*, "RFC2616: hypertext transfer protocol-HTTP/1.1." RFC Editor, 1999.

BIOGRAPHIES OF AUTHORS



Prathima Mabel John    is Assistant Professor at Dayananda Sagar College of Engineering, Visvesvaraya Technological University (VTU), Bengaluru, Karnataka, India. She received her Bachelor of Engineering and Master of Technology degree in Computer Science and Engineering from VTU, Belagavi, Karnataka, India. She is currently pursuing Ph.D. from VTU, Belagavi, Karnataka, India. She has about 13 years of experience in teaching and industry together. Her areas of interest are computer networks, SDN, mobile networks, network security and machine learning. She can be contacted at email: prathimamabel-ise@dayanandasagar.edu.



Rama Mohan Babu Kasturi Nagappasetty    is currently working as Professor in the Department of Information Science and Engineering at Dayananda Sagar College of Engineering, Bengaluru, India. He obtained his B.Tech in Computer Engineering from Mangalore University, India, M.S from BITS-PILANI, India and Ph.D. from Dr. MGR University, India. His areas of interest are computer networks, wireless mobile networks, SDN and network security. He can be contacted at ramamohanbabu-ise@dayanandasagar.edu.