Volume 63, Number 1, 2023

ACTA TECHNICA NAPOCENSIS
Electronics and Telecommunications

DDoS ATTACK DETECTION USING SUPERVISED MACHINE LEARNING ALGORITHMS OVER THE CIDDOS2019 DATASET

Daniel ZINCA, Virgil DOBROTA
Communications Department, Technical University of Cluj-Napoca, Romania
Corresponding author: Daniel.Zinca@com.utcluj.ro

Abstract: **Distributed Denial-of-Service (DDoS) attacks are one of the most common types of cyber-attacks that can cause severe damage to networks and systems. Traditional methods to detect them rely on signature-based Intrusion Detection Systems (IDS), which are limited by the need of prior knowledge of specific patterns and by the usual ineffectiveness against zero-day attacks. However machine learning (ML) algorithms have the potential to support the detection of new and unknown attacks. This article compares the DDoS detection performance of three Machine Learning techniques: Gaussian Naïve Bayes, Logistic Regression and Random Forest, based on validation metrics such as precision, recall and F1 score. The system was trained using three datasets extracted from CICDDoS2019 database. The results proved the detection of attacks at Layer 4 (TCP SYN/ UDP flood), and at reflective Layer 7 (MSSQL, NetBIOS). The Random Forests and Logistic Regression methods achieved a precision between 93.7% and 99.4 % over these three datasets.**

Keywords: *CICDDoS2019, DDoS, Gaussian Naïve Bayes, Intrusion Detection Systems, Logistic Regression, Random Forest.*

I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks are a major threat for online services and organizations, causing downtime, financial loss, and also reputational damage. They involve the flooding of a targeted system with a large number of requests, overwhelming its resources and causing it to become unavailable to legitimate users. DDoS attacks are typically carried out using botnets, i.e. networks of compromised computers that are controlled by a central command-and-control (C&C) server. The botnet is used to send a large volume of requests to the targeted system, with the goal of overwhelming its resources and causing it to become unavailable. Traditional approaches to detect DDoS attacks rely on signature-based Intrusion Detection System (IDS) implementations. They are limited by the need for prior knowledge of specific attack patterns. In general, IDS monitor network traffic to search for signatures of malicious activity or for violations of rules previously created by security policies. Signature- and anomaly-based methods are the two main approaches used to build such systems [1]. A signature represents an indicator of compromise created based on known attacks. Snort is considered to be one of the most popular Open-Source IDS that implements such approach [2]. It has capabilities to detect DDoS attacks either by using the built-in set of rules, either by developing new ones [3]. The detection capabilities in the case of this approach greatly depend on the rules that are written and a slightly different attack may not be detected.

Recently machine learning (ML) algorithms proved to be a promising approach for DDoS detection, having the potential to detect new and unknown attack patterns by implementing anomaly-based IDS. ML algorithms can analyze large volumes of network traffic data and learn how to identify anomalous patterns. Papers [4]-[8] present various algorithms used to assess the performance against different datasets.

Depending on the layer the affected protocol belongs to, there are Layer 3, Layer 4 and Layer 7 attacks [7]-[10]. The following two categories represent a possible classification of DDoS:

1. *Flooding attacks* that attempt to exhaust the resources available at the targeted entity, by sending a large number of packets, flooding the channel or the server resources. One subcategory is represented by an exploitation attack where a “vulnerability” in the protocol design is used, like in the case of the TCP SYN-Flood.
2. *Reflection and amplification attacks* where the attacker is spoofing packets having as source address the target device. This determines other computers to send back packets to the victim’s IP address. This category of issues is called DrDoS (Distributed Reflection Denial of Service).

For the traditional approach of a rule-based IDS, the metrics that are used are the number of true positives, true negatives, false positives and false negatives. In [12] the performance of rule-based IDS (using snort) was assessed. Also, in [12] it was demonstrated that a collaborative detection mechanism using flow-based anomaly detection with Deep Neural Networks improves the performance over the usage of the rule-based IDS by achieving more than 90% true positive rate with less than 5% of false alarms.

A recent dataset CIC-DDoS2019 was created by the Canadian Institute of Cybersecurity. It contains records for

Manuscript received March 26, 2023; revised June 22, 2023

DDoS attacks on Layer 4 (TCP/ UDP) protocols and on Layer 7 protocols in addition to benign traffic [9]-[10]. The CIC-DDoS2019 dataset was captured in two different days and contains not only the entire traffic in *pcap* format but also csv files with 80 features that were extracted from them using the CICFlowMeter v3 tool [9]. Each record was labeled with the corresponding classification (benign or a certain type of DDoS attack). Each csv file was named according to the attack present in the majority of the records, along with some benign traffic and in some cases, with other attacks. Table I summarizes the files with corresponding attacks in each folder. The total number of records for each attack is not evenly distributed and it does not follow the percentages encountered in the case of real network traffic. The behavior for each category can be very different than the others, making the detection of new attacks more difficult.

TABLE I. CIC-DDOS2019 DATASET FILE CONTENTS WITH BENIGN, L4 AND L7 DDOS ATTACKS SPECIFICATIONS

Day 1	**Day 2**
Syn (TCP) UDP UDPLag	Syn (TCP) DrDoS UDP UDPLag
LDAP MSSQL NetBIOS Portmap	DrDoS LDAP DrDoS MSSQL DrDoS NeBIOS DrDos NTP DrDoS SNMP DrDoS SSDP DrDoS DNS TFTP

The performances obtained in paper [9] against the complete dataset are rather low, with the RF algorithm achieving the F1 value of 0.62. In [8] the approach was to split each csv file into the training and testing part and then separately measuring the performance of each algorithm for each attack. Even if the resulting performance in [8] is better than in [9], the F1 score being 0.99 in most cases, the disadvantage in [8] is that the algorithm is trained to classify one attack only versus normal traffic. Paper [10] used a mixture of records from different datasets. In this work we propose a novel combination of data from the same dataset to obtain three different ones in order to test the performance of each Machine Learning algorithm involved. The remainder of the paper is organized as follows: Section II describes the contents of the derived datasets and the Machine Learning algorithms, Section III discusses the experimental results and the last section concludes the paper. The advantages of partitioning the dataset into three datasets as described in Section II consist in a better detection in terms of Precision, Recall and F1 score compared to [9] and in the classification of multiple attacks for several protocols, compared to [8].

II. METHODS

The source code for data processing and for training, testing and measuring performance was done by using the Google Colab platform[13], the Python language and the `scikit` library. Each file in the CICDDoS2019 dataset contained records from one attack targeted to a certain protocol. However there are two aspects that can lead to a lower performance when using just the original dataset and caused the results in [8]:

1. In the case of some protocols, the file from one day contains records for a different attack (for example a flooding attack) than the records targeting the same protocol in the other file (where a DrDoS attack is recorded).
2. The behavior of a Layer 4 attack is different than the behavior of a L7-based one.

Due to the fact that the size of each file in the original dataset is very large and it contains in some situations millions of records, the load operation can be inefficient. In DoS/ DDoS attacks, there is a high number of packets with identical characteristics, differentiated only by their timestamp. For a signature-based method, this aspect is important and it can increase the detection capability. On the other hand, the timestamp value is not useful as a feature for a Machine Learning algorithm and all duplicate records should be removed to eliminate biased results in the training stage.

We addressed these aspects and we generated from the original dataset, a total number of three different datasets we worked with. The size problem was solved by using the compressed parquet feature available in the `read_parquet` from the `pandas` Python library [14], combined with the removal of duplicated records. For all datasets we selected protocols that were present in both days. For Layer 4 we selected TCP (Syn) and UDP, whilst for Layer 7 we chose LDAP, MSSQL and NetBIOS. Each dataset had 78 features.

The first dataset we assembled contains the records from Day 1 from the files with TCP(Syn), UDP, MSSQL, LDAP and NetBIOS. Then we split the file on 80% for training and 20% for testing. The number of benign and attack traffic is presented in Table II.

TABLE II. DATASET1 WITH BENIGN, L4 AND L7 DDOS ATTACKS

Attack Class	**Training Dataset**	**Testing Dataset**
Benign	30,678	7,581
UDP	1,030,024	258,171
TCP (Syn)	362,353	90,512
MSSQL	206,672	51,339
LDAP	13,717	3,324
NetBIOS	9,004	2,186
TOTAL	**1,652,448**	**413,113**

For the rest of the datasets we used record from distinct files for training compared to testing. In the second dataset we choose only UDP in L4, but the same protocols for L7 attacks as in the previous case. For this dataset, the training dataset was taken from the corresponding Day 1 files and the testing dataset from the corresponding Day 2 files. The numbers for benign and attack records for each part are presented in Table III.

TABLE III. DATASET2 WITH BENIGN, UDP AND L7 DDOS ATTACKS

Attack Class	Training Dataset	Testing Dataset
Benign	11,225	6931
UDP	1,288,195	1,074,277
MSSQL	258,011	193,346
LDAP	17,041	28,843
NetBIOS	11,190	17,886
TOTAL	**1,585,662**	**1,321,283**

Finally in the last dataset we were focused on L7 attacks only. Therefore it has fewer records than Dataset 2, because the UDP records were eliminated. See Table IV for the number of records in Dataset 3.

TABLE IV. DATASET3 WITH BENIGN AND L7 DDOS ATTACKS

Attack Class	Training Dataset	Testing Dataset
Benign	8,392	4,889
MSSQL	253,051	193,346
LDAP	17,041	28,843
NetBIOS	11,190	17,886
TOTAL	**289,674**	**244,964**

After the dataset is loaded from `parquet` files and then concatenated, the features for which the `corr()` function has a value higher than 0.97 were removed, the number of remaining features being 41.

The next step was to train the model using ML. For the purposes of this article we choose three algorithms: Gaussian Naïve Bayes, Decision Tree and Random Forests. For each dataset, we trained the model for each of the three algorithms and we compared their performance.

A. Gaussian Naïve Bayes

Gaussian Naïve Bayes represents a Supervised Machine Learning Algorithm based on the Bayes Theorem that is used to determine, based on the conditional probability, the posterior probability of a hypothesis that considers the occurrence of each feature is independent of the other features. The Gaussian variant of the Naïve Bayes algorithm uses a specific probability equation that is expressed in (1).

$$(xi|y) = \frac{1}{\sqrt{2\pi\sigma y^2}} exp(-\frac{(\mathrm{xi}-\mu\mathrm{y})^2}{2\sigma y^2}) \qquad (1)$$

where σy represents the dispersion for class y and μy is the median value for class y, both being estimated with a maximum probability. We used the `GaussianNB` from `sklearn.naive_bayes`, `scikit` library.

B. Decision Tree

This algorithm is a Supervised Learning technique that can be used in the case of classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node is the outcome. The decisions are made on the basis of features of the given dataset.

Decision Tree is a graphical representation for getting all the possible solutions to a problem/ decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. A tree is composed of nodes, and those nodes are chosen looking for the optimum split of the features. We implemented this algorithm by using the `DecisionTreeClassifier` class from the `sklearn` library.

C. Random Forest

This is a Supervised ML algorithm that is used widely in classification and regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression. Random Forest uses the Bagging *(*`Bootstrap Aggregation`*)* which is an ensemble method. Ensemble techniques consist in combining multiple models. Bagging chooses a random sample from the data set. Hence each model is generated from the samples provided by the original data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Then each model is trained independently and generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation. We implemented Random Forest by using the `RandomForestClassifier` class from the `sklearn` library.

III. EXPERIMENTAL RESULTS

In order to assess the performance of each of the three algorithms for the case of each of the three datasets, we computed the precision (or detection rate), recall, and F1 score (which combines both precision and recall) using (2) – (4). In addition, we also show the confusion matrix in each case.

$$Precision = \frac{TP}{TP+FP} \qquad (2)$$

$$Recall = \frac{TP}{TP+FN} \qquad (3)$$

$$F1 = 2 * \frac{Precision*Recall}{Precision+Recall} \qquad (4)$$

True Positive (TP) represents the number of records correctly matched as attack traffic; True Negative (TN) represents the number of records correctly matched as normal traffic; False Positive (FP) is the number of normal records incorrectly labeled as attack traffic; and finally False Negative (FN) is the number of DDoS attack records incorrectly labeled as normal traffic.

Tables V-VII present the performance in the case of each Dataset, with respect to each ML algorithm. We analyzed the confusion matrices, as in Tables VIII-X, XII-XIV and XVI-XVIII, to determine the best algorithm for each Dataset. The algorithm performance was compared for each Dataset in Tables XI, XV and XIX.

TABLE V. PRECISION, RECALL AND F1 SCORE FOR DATASET1

Attack Class	**Machine Learning Algorithm**								
	Gaussian Naïve Bayes			**Decision Tree**			**Random Forest**		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
Benign	0.85	0.16	0.26	0.98	1.00	0.99	1.00	1.00	1.00
LDAP	0.00	0.00	0.00	0.85	0.87	0.86	0.92	0.90	0.91
MSSQL	0.90	0.57	0.70	0.97	0.99	0.98	0.98	0.98	0.98
NetBIOS	0.11	0.04	0.06	0.98	0.95	0.96	0.97	0.97	0.97
TCP	0.98	0.89	0.93	1.00	1.00	1.00	1.00	1.00	1.00
UDP	0.87	0.99	0.93	1.00	1.00	1.00	1.00	1.00	1.00

TABLE VI. PRECISION, RECALL AND F1 SCORE FOR DATASET2

Attack Class	**Machine Learning Algorithm**								
	Gaussian Naïve Bayes			**Decision Tree**			**Random Forest**		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
Benign	0.73	0.17	0.28	0.74	0.98	0.84	0.78	1.00	0.87
LDAP	0.00	0.00	0.00	0.87	0.91	0.89	0.88	0.93	0.91
MSSQL	0.84	0.67	0.74	0.93	0.98	0.95	0.94	0.98	0.96
NetBIOS	0.01	0.01	0.01	0.96	0.48	0.64	0.97	0.50	0.66
UDP	0.73	0.17	0.28	0.74	0.98	0.84	0.78	1.00	0.87

TABLE VII. PRECISION, RECALL AND F1 SCORE FOR DATASET3

Attack Class	**Machine Learning Algorithm**								
	Gaussian Naïve Bayes			**Decision Tree**			**Random Forest**		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
Benign	0.75	0.15	0.24	0.85	0.99	0.92	0.83	1.00	0.91
LDAP	0.00	0.00	0.00	0.88	0.90	0.89	0.89	0.94	0.91
MSSQL	0.79	1.00	0.88	0.95	0.98	0.96	0.95	0.98	0.97
NetBIOS	0.27	0.01	0.02	0.99	0.49	0.66	0.99	0.50	0.67

A. Results obtained based on Dataset1

TABLE VIII. CONFUSION MATRIX FOR GAUSSIAN NAÏVE BAYES (DATASET1)

Benign	**1186**	52	22	56	784	5481
LDAP	1	**0**	1925	6	9	1383
MSSQL	1	0	**29158**	77	291	21812
NetBIOS	0	0	226	**88**	687	1185
TCP	202	4	33	140	**80295**	9838
UDP	1	0	929	420	2	**256819**

TABLE IX. CONFUSION MATRIX FOR DECISION TREE (DATASET1)

Benign	**7545**	4	6	5	21	0
LDAP	16	**2882**	410	3	12	1
MSSQL	35	513	**50648**	4	32	107
NetBIOS	10	1	34	**2075**	15	51
TCP	53	0	34	3	**90400**	22
UDP	4	0	926	27	6	**257208**

Figure 1 presents the ten most important features according to the coefficient's values in the case of Decision Tree for Dataset1.

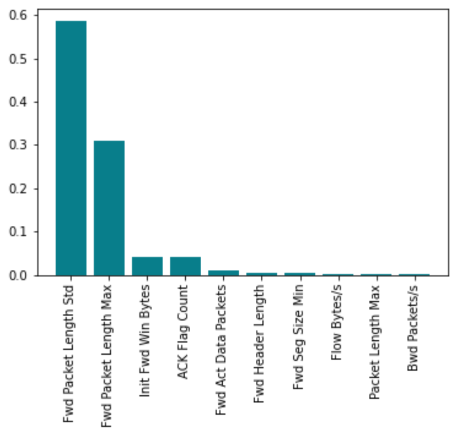

Figure 1. Ten most important features for Decision Tree (Dataset 1)

TABLE X. CONFUSION MATRIX FOR RANDOM FOREST (DATASET1)

Benign	**7578**	0	0	1	2	0
LDAP	4	**2991**	313	3	12	1
MSSQL	3	251	**50391**	24	38	632
NetBIOS	2	4	26	**2111**	12	31
TCP	11	6	28	12	**90439**	16
UDP	0	0	776	27	11	**257357**

Comparing the precision in the case of all three algorithms using Dataset1 we came to the conclusion, according to Table XI that Random Forest presents the best performance.

TABLE XI. ALGORITHM PERFORMANCE (DATASET1)

Algorithm used with Dataset1	**Precision (training dataset)**	**Precision (testing dataset)**
Gaussian Naïve Bayes	0.8884	0.8896
Decision Tree	0.9943	0.9943
Random Forest	0.9979	**0.9945**

B. Results based on Dataset2

TABLE XII. CONFUSION MATRIX FOR GAUSSIAN NAÏVE BAYES (DATASET2)

Benign	**1200**	40	161	114	5416
LDAP	58	**0**	11913	151	16721
MSSQL	98	5	**129588**	34	63621
NetBIOS	106	15	6547	**100**	11118
UDP	189	2	6567	10512	**1056997**

TABLE XIII. CONFUSION MATRIX FOR DECISION TREE (DATASET2)

Benign	**6827**	36	62	6	0
LDAP	332	**26138**	2357	16	0
MSSQL	397	3345	**189377**	35	195
Net BIOS	695	291	8112	**8631**	157
UDP	995	306	3956	297	**1068723**

TABLE XIV. CONFUSION MATRIX FOR RANDOM FORESTS, FOR THE CASE OF DATASET2

Benign	**6907**	0	6	17	1
LDAP	268	**26753**	1807	11	4
MSSQL	343	3103	**188913**	32	955
Net BIOS	441	147	7988	**9003**	307
UDP	927	235	2866	193	**1070056**

Comparing the precision in the case of all three algorithms using Dataset2 we came again to the conclusion, according to Table XV that Random Forest has the best performance.

TABLE XV. ALGORITHM PERFORMANCE (DATASET2)

Algorithm used with Dataset2	**Precision (training dataset)**	**Precision (testing dataset)**
Gaussian Naïve Bayes	0.9019	0.8990
Decision Trees	0.9938	0.9836
Random Forests	0.9969	**0.9851**

C. Results obtained based on Dataset3

Table XVI presents the confusion matrix. The results obtained in this case showed that Gaussian Naïve Bayes should not be used to detect L7 DDoS attacks.

TABLE XVI. CONFUSION MATRIX FOR GAUSSIAN NAÏVE BAYES (DATASET3)

Benign	**713**	24	4065	87
LDAP	51	**0**	28551	241
MSSQL	93	5	**193149**	99
NetBIOS	98	14	17616	**158**

TABLE XVII. CONFUSION MATRIX FOR DECISION TREE (DATASET3)

Benign	**4863**	3	4	19
LDAP	127	**25930**	2730	56
MSSQL	366	2961	**189996**	23
NetBIOS	357	543	8203	**8783**

TABLE XVIII. CONFUSION MATRIX FOR RANDOM FOREST (DATASET3)

Benign	**4866**	0	9	14
LDAP	164	**26978**	1684	17
MSSQL	325	3301	**189668**	52
NetBIOS	504	168	8184	**9030**

Figure 2 presents the most important 10 features according to the coefficient's values in the case of Random Forests for Dataset3.

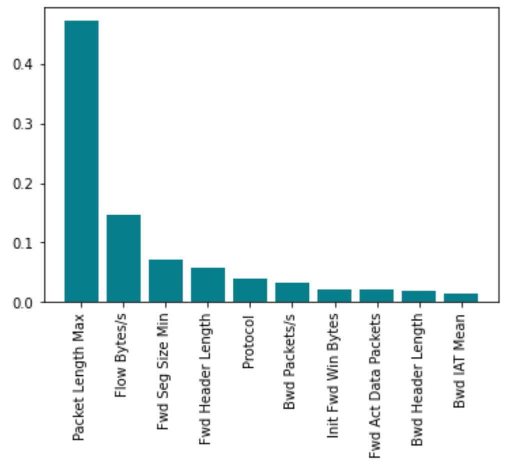

Figure 2. Ten most important features for Random Forest (Dataset 3)

Comparing the precision in the case of all three algorithms using Dataset3, again the conclusion, according to Table XIX, the Random Forest has the best performance.

TABLE XIX. ALGORITHM PERFORMANCE (DATASET3)

Algorithm used with Dataset3	**Precision (training dataset)**	**Precision (testing dataset)**
Gaussian Naïve Bayes	0.8766	0.7920
Decision Trees	0.9852	0.9371
Random Forests	0.9964	**0.9411**

IV. CONCLUSIONS

In this work we analyzed the performance of three Machine Learning algorithms: Gaussian Naïve Bayes, Decision Tree, Random Forest on the detection of DDoS attacks, both flooding and reflection/amplification, on L4 and L7. We derived three datasets from the CIC-DDoS-2019 dataset. Dataset 1 contains records for TCP(Syn) and UDP flood for L4 attacks and L7 attacks for the protocols MSSQL, NetBIOS, LDAP. Dataset 2 contains UDP flood attacks and L7 attacks, where the testing data was captured on a different date. Dataset 3 contains L7 attacks only. From the measurements we performed, the Random Forest algorithm performed very well with precision between 0.94 and 0.99 depending on the dataset. The Decision Tree algorithm offered also good performance, between 0.93 and 0.99 depending on the dataset. The third algorithm was Gaussian Naïve Bayes that offered much lower performance than the other algorithms, between 0.79 and 0.88.

Future work will consider the implementation of Deep Learning algorithms for the purpose of DDoS attack detection, based on the same datasets.

REFERENCES

[1] O.U. Olouhal, T.S. Yange, G.E. Okerekel and F.S. Bakpol, “Cutting Edge Trends in Deception Based Intrusion Detection Systems-A Survey”. J. Inf. Secur. 2021, 12, 250–269. https://doi.org/10.4236/jis.2021.124014.

[2] Snort3, [Online].Available: https://www.snort.org/snort3

[3] Z. Hassan, Shahzeb, R. Odarchenko, S. Gnatyuk, A. Zaman and M. Shah, "Detection of Distributed Denial of Service Attacks Using Snort Rules in Cloud Computing & Remote Control Systems," 2018 IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC), Kiev, Ukraine, 2018, pp. 283-288, doi: 10.1109/MSNMC.2018.8576287.

[4] V. Verma, and V. Kumar, “DOS/DDOS Attack Detection using Machine Learning: A Review”. Proceedings of the International Conference on Innovative Computing & Communication (ICICC) 2021, doi: 10.2139/ssrn.3833289.

[5] T.E. Ali, Y.-W. Chong, and S. Manickam, “Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review,” *Applied Sciences*, vol. 13, no. 5, p. 3183, Mar. 2023, doi: 10.3390/app13053183.

[6] M. Mittal, K. Kumar and S. Behal, “Deep learning approaches for detecting DDoS attacks: a systematic review”. *Soft Comput*, 2022, doi: 10.1007/s00500-021-06608-1

[7] O. Bamasag et al., “Real-time DDoS flood attack monitoring and detection (RT-AMD) model for cloud computing”, PeerJ Comput. Sci., 2022, doi: 10.7717/peerj-cs.814. Avaible: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9202629/

[8] K.B. Dasari, N. Devarakonda, “Detection of Different DDoS Attacks Using Machine Learning Classification Algorithms”, *Ingénierie des Systèmes d’Information*, 2021, pp. 461-468, doi: 10.18280/isi.260505

[9] I. Sharafaldin, A.H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy", 2019 International Carnahan Conference on Security Technology (ICCST), Chennai, India, 2019, pp. 1-8, doi: 10.1109/CCST.2019.8888419.

[10] I. Sharafaldin, A.H. Lashkari, S. Hakak, and A. A. Ghorbani. (2022), “CIC-DDoS2019 Data set”. Kaggle. Available: https://doi.org/10.34740/KAGGLE/DSV/4059918

[11] T. Booth and K. Andersson, “Mitigating DRDoS Network Attacks via Consolidated Deny Filter Rules”, *ReBICTE*, vol. 6, pp. 19–29, Oct. 2020.

[12] R. M. A. Ujjan, Z. Pervez and K. Dahal, "Suspicious Traffic Detection in SDN with Collaborative Techniques of Snort and Deep Neural Networks," 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Exeter, UK, 2018, pp. 915-920, doi: 10.1109/HPCC/SmartCity/DSS.2018.00152.

[13] Google Colab platform, [Online], Available: https://colab.research.google.com/

[14] Laurens D'Hooge, CIC-DDoS2019-00-Cleaning, StrGenIx 2019, [Online], Available: https://www.kaggle.com/code/dhoogla/cic-ddos2019-00-cleaning.