A Novel Multilevel Framework for DoS Detection in SDN

Rejo Rajan Mathew[1], Amarsinh Vidhate[2]
Department of Information Technology, Ramrao Adik Institute of Technology, D Y Patil University, India[1]
Department of Information Technology, MPSTME, SVKMs NMIMS University, India[1]
Department of Computer Engineering, Ramrao Adik Institute of Technology, D Y Patil University, India[2]

***Abstract*—DoS attacks have been the most popular type of attack on SDNs. The threat landscape has widened due to advanced persistent threats. Recent studies have focused on a single level of defence and conventional detection methods, which have become redundant. The study proposes and implements a novel multilevel DoS attack detection, which has a three-pronged approach to counter modern-day DoS attacks. The first level emphasizes the Zero Trust mechanism using Hash SHA-256 to validate the clients. The second level uses hybrid deep learning models to detect DoS attacks, which are trained and tested across three latest datasets, namely NSLKDD, CIC DOS 2019 and IOT2023, giving an accuracy of 95% consistently. The third level is a lightweight adaptive DoS detection, which can detect fast and low-rate DoS attacks, ensuring that the SDN is secure in a few milliseconds by ruling out any possibility of congestion. The results clearly indicate how a three-level approach can thwart most advanced persistent threats.**

***Keywords*—Software defined network; distributed denial of service; openflow**

I. INTRODUCTION

Traditional networks are decentralized, rigid to allow changes, and complex to configure and manage due to vendor-specific issues. SDN is a disruptive innovation as it allows remote management of large-scale deployments with ease and takes care of the rising demand for processing, storage, and networking. It has a centralized and programmable controller along with switches in the data plane. SDN takes advantage of virtualization to provide flexibility, adaptability, elasticity, scalability, robustness, security, and abstraction of the hardware [1]. Below mentioned are the vulnerabilities in SDN based on layers:

A. Control Layer (Control Plane)

As the central element of SDN, if the controller is not managed correctly, it could be a single point of vulnerability and attack. The controller establishes the direction of data flow within the data plane. When the controller is attacked and compromised, it will have a negative impact on the network.

1) Hijacked rogue controller: The attacker takes control of the original controller or changes its functionality such that they can play around with the network. Some attackers replace the actual controller with a rogue controller, which controls the entire network thereafter. [2]

2) Fake traffic and network manipulation: Intruder exhausts all the resources by introducing fake packets and simultaneously initiates other attacks on the entire network. [3, 4, 5]

3) Weak authentication, authorization and information disclosure: When the controller has weak authentication and incomplete encryption, the attacker breaches confidentiality, leading to information disclosure, making it susceptible [6].

B. Infrastructure Layer (Data Plane)

Flow Switches are non-intelligent and can be manipulated by attackers [7, 8].

1) Flow table modification: The malicious nodes alter the flow table rules by adding or deleting rules and denying updates to the table.

2) Flow table overloading/buffer saturation: Flow table rules are stored in TCAM, which is expensive, power hungry, with limited space, making it vulnerable to overloading and buffer saturation attacks. They can jam the bandwidth to cause overloading.

3) Topology spoofing/traffic diversion: Create a fake network view to redirect traffic flows to attackers and explore vulnerabilities such as eavesdropping. They can forge non-existent links between switches.

4) Side channel attack: It targets the timing information by collecting ping timings and adding delays. Also, it can inform an attacker if a flow rule exists or not.

C. Application Layer (Application Plane)

Some contemporary attackers have started to exploit the applications present in the application layer [9].

1) Malicious applications: Attacker can modify or run malicious applications to ruin the network or collect information of the network.

2) Resource exhaustion: Attacker can run resource intensive requests to controller bringing down the system.

3) Policy conflicts: Applications running on single controller give conflicting policy instructions leading to delay and confusion.

4) Privilege elevation: One application gains higher privilege access over other applications and later controls and manipulates them.

D. Interfaces

1) Northbound interface: It serves as the application layer's and the control layer's open, local, non-standard, and non-encrypted connection interface. It collects network details, state resource requirements from the upper layer and gives directions to the controller in the control plane. As there are various types of REST APIs used here, this becomes a huge point of vulnerability for attackers to tamper or eavesdrop [10].

2) Southbound interface: It serves as the control layer's and data layer's open, local, and non-encrypted connection interface. The controller uses it to create and modify rules in the Flow table. Rules can be intercepted and manipulated due to a lack of security [11]. TLS security is recommended but not mandatory, making it vulnerable to contemporary attacks [12].

3) East-West bound interface: In most architectures, there is a single controller, and there are distributed multi-controller architectures to manage the SDN. The interface between these controllers is called the East-West bound interface. These controllers synchronise and work with each other for high availability by exchanging notifications. They share updates about the network, like the state, devices, changes in the flow table, list of hosts, etc. Due to a lack of security, attackers compromise and exploit the controller [13] interface between the switches. In the data plane, switches are interconnected via links on which packets traverse till they reach the destination. Links are open, local and not encoded, making it vulnerable.

Section II discusses the various DDoS defense approaches in related work. Section III presents the proposed method. Section IV highlights the experimental setup and results. Section V presents the conclusion.

II. Related Work

DDoS defense approaches are classified into two categories based on where these defense models have been deployed. Most attacks are carried out on the Controller rather than the switch.

A. Controller-Based Solutions

Bringing down the controller helps destroy the entire SDN. The main target is the controller, and the only way to protect the entire SDN is by protecting the controller. Over the years, many solutions have been proposed by researchers to protect the controller. Four major categories are: one is entropy-based, Machine learning, Neural Networks, Statistical analysis-based.

1) Information Entropy: During normal days, the traffic flow is normal, and the entropy value is maximum. As soon as the attack starts, the traffic flow increases and entropy starts to decrease. The threshold level for entropy is set. If the traffic flow goes below this level, then it triggers the detection mechanism. In [14], the authors compute the threshold value five times, and then it triggers the detection mechanism. In [15], there is a two-step mechanism, in the first step, it checks the decrease in the threshold value, and in the second step, if the value decreases, it checks MAC addresses to differentiate between flash traffic and attack traffic. There are two modules in [16], the first one is for detection based on the threshold value, and the second module is used for mitigation by blocking unacknowledged SYN packets from the client. In [17], there are three phases, the first is normal phase of counting of packets, second is the preparatory phase, if value goes below the threshold, third is mitigate phase by dropping packets. Authors in [18] built a trust framework to identify malicious flow based on Kullback–Leibler divergence or relative entropy method. In [19], the authors use Jensen-Renyi divergence measure to estimate the variation from the threshold value for normal as well as malicious traffic. This can detect both high and low-rate attacks. Authors in [20] use three values- counter, entropy, and traffic rate. For detection as the packet flow increases the entropy decreases count gets incremented. For mitigation, it collects DPID, port and IP information to block the malicious packets.

2) Machine learning: Based on historical data or changes in traffic flow, the system learns and detects the presence of DDoS attacks. Authors [21] propose aggregating packets to create signatures which can be later used to detect attacks. In [22] authors use a sequence of feature values to predict and detect attacks. Authors in [23] classify attacks using machine learning and neural networks. In [24], an SVM classifier was used after the authors extracted key features. In [25], the authors solved the multi-class classification problem by advancing the SVM algorithms. In [26], the authors implemented and compared MLP, Decision Tree, Random Forest and SVM to conclude that Decision Tree gives the best results. Authors in [27] extracted the features from the traffic to draw the feature pattern graph. In [34], the authors used entropy on ports and applied the KNN algorithm to detect and drop malicious traffic. In [28], the authors worked on KNN to get better accuracy. In [29], the authors classify the traffic based on packet flow, size and speed. In [30], the authors figured out the attack's intensity and classified the traffic using a KNN classifier. In [31], the authors implemented the Factorization Machine algorithm to mitigate low-rate DDoS attacks. In [32], the authors developed a multi-plane security framework named VARMAN, where the switch is modified to detect anomalies and then apply ML-based classifiers to detect attacks. In [33], the authors worked on programmable switches to improve efficiency. In [34], the authors deployed a trigger mechanism on the switches which alerts the controller, who activates the ML classifier (KNN and K-means) to detect the malicious traffic. In [35], the authors implemented a low-volume defense solution with the following models (MLP, Random Forest, Decision Tree, SVM, REP Tree, J48). In [36], the authors implemented a hybrid ML algorithm (Support Vector Classifier and Random Forest) to detect malicious traffic. In [37], the authors implemented the Redis Simple Message Queue (RSMQ) mechanism with the following machine learning models (ExtraTrees, Decision Tree, KNN and Naive Bayes) to improve the accuracy of detection. In

[38], the authors classified traffic using following machine learning models (MLP, Random Forest, Logistic Regression, DT and AdaBoost).

3) Artificial neural networks: Authors in [39] implement a mechanism which traces the source of the attack as it maintains the details of the entire network and a mitigation module to delete the malicious table flow entries. Authors in [40] classify the traffic using fuzzy logic combined with L1-Extreme Learning Machines. By employing both unsupervised hashing and the discrete-time finite-state Markov chain (DTMC) model, it mitigates attacks. In [41], the controller acts selectively and focuses only on affected switches by applying the fuzzy c-means algorithm.

4) Statistical analysis: In [42], the source is verified before accepting packets. The controller acts as a proxy and verifies the source, and sends SYNACK. In [43], it maintains the state and connection list of all TCPs. This list keeps on increasing till it reaches the set limit. Once it crosses that limit, the controller blocks packets from those sources. The authors designed the RADAR framework in which the controller selects certain flows to be investigated. Authors too confirm the legitimacy of the source, and the algorithm verifies the information in the Packet-In message with this dictionary of Port, IP, and MAC addresses. The malicious traffic is taken care of by a backup controller, which temporarily saves these requests as the attack begins, then quickly sends them to the main controller. Authors compare the threshold value to the quantity of Packet-In messages with a certain header received during a specific time interval. If it crosses the predetermined value, all packets are deep scanned.

B. Switch-Based Solutions

1) TCP proxies: LineSwitch deploys proxies based on random distribution and ensures blocking malicious traffic from reaching the control plane. In another paper, both act as proxy and minimizes the timeout to improve the efficiency. Authors provide multi-layer protection by implementing filters at all entry and exit points on the switches.

2) Information entropy: Implements the algorithm on the switches instead of overloading the controller. Authors deep-scan the packets for spoofing/changes, and once it locates the attacker host, it changes the table rules to block this host. To identify an attack, it also computes entropy within a predetermined time interval. The programmable switches cannot decode the arithmetic logarithmic function, so to calculate entropy, the authors utilize the longest-prefix match (LPM) table. It determines the IP-based entropy at the source and destination. Authors detect an attack on programmable switches by computing the entropy for IP addresses. They use the longest-prefix match (LPM) table and bespoke count-sketches, respectively, to approximate frequencies and carry out computationally complex mathematical operations.

3) Artificial neural networks: Used switches as neurons in a distributed intrusion prevention system based on BPNN. Parallel operation is used by switches on the same layer. For easy computations, each swap uses minimal resources. Further, they deployed a Radial Basis Function (RBF) neural network to enhance this work's accuracy. Authors propose a simple DDoS mechanisms having machine learning based, traditional IDS on a heterogeneous system including FPGA boards, GPU, and host processors.

4) Flow statistics: The security tools such a DNS reflection defense, firewall and extensive filtering have been installed. In [14], the authors, to counteract HTTP GET flooding assaults, per-URL counting technique was created on NetFPGA-based Openflow switches. On the programmable switches, the DDoS detection logic was deployed. The switch computes a hash value using the source's MAC and IP addresses and compares it to one that was already computed.

5) Smart switches: Authors deployed smart switches which would perform source address matching. If there is a mismatch it will automatically alert and block the traffic. It helps in early detection but will lead to overloading the controller with invalid requests. In another paper the source packets are filtered, and genuine packet traffic is given specific score so as to be allowed. This method is useful to limit the attack, but if the attacker bypasses this filter, it will fail. Authors use a port-hopping defense mechanism, where the target is moved to the controller, reducing the chances of an insider attack. In this, we need to add more moving target defense mechanisms, such as address hopping, service hopping, path hopping, etc.

III. Proposed Method

The traffic flow consists of information from regular clients as well as the DoS Attacker. The proposed method shown in Fig. 1 comprises three levels to ensure robust DoS attack detection:

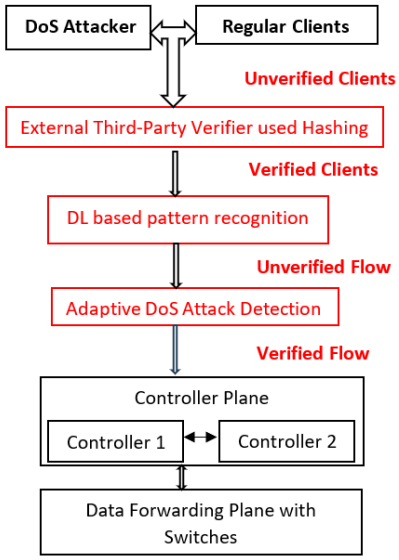

Fig. 1. Proposed method.

First Level – The first level allows only regular flow by matching the hash of the current flow with the hash of the

previously stored record. This ensures that only the verified clients go ahead to the next level.

Second Level – It consists of a deep learning-based hybrid model, namely RNN + LSTM, which has been trained and tested across three datasets to classify DoS attacks based on flow and performance-based parameters.

Third Level – Most APTs these days can mix Fast as well as Low-rate DoS Attacks to bypass the defense systems. So, we deploy the Adaptive DoS Attack detection, which will check all parameters and ensure only verified flows reach the controller.

A. Client Validation – Level 1

Fig. 2 shows an external third-party component that performs this client validation to ensure authorized users move on to the next level and reach the controller.

1) Collect - MAC Address and Source Port Number from each flow is collected by the third-party verifier.

2) Convert and Store – Third-party verifier converts the collected data into a hash value (SHA-256) and stores it in a hash table. The hash table consists of the hash value of known/regular clients.

3) Match - Repeat Step 1 and Step 2 to match the hash value of the current flow with the hash value of the flows stored in the table. If it matches, it will be classified as normal traffic and move to the next level or else block and do not allow that client to move ahead.

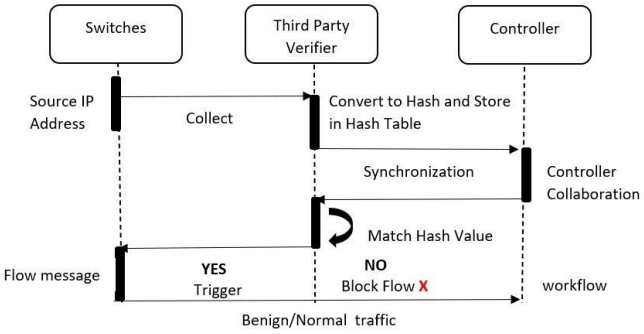

Fig. 2. Client validation.

B. Deep Learning Based DoS Detection – Level 2

The process of creating a Deep Learning based DoS attack detection system employs a methodical process beginning with the selection of three popular datasets. These datasets are NSL-KDD, CIC-DDoS 2019, and IoT-2023, which are intended to cover different types of network traffic patterns as well as possible attack types. To better illustrate these attack types, the datasets were focused on these specific attacks because they represent very prevalent attack types in real-life conditions. Certain attacks including SYN flood, UDP flood, and SQL Injection, were narrowed down for training of the model with intent of distinguishing between DoS/DDoS attacks and other attacks examples being ARP spoofing and command injection attacks. During model implementation, experimentation was carried on several deep learning architectures, namely the RNN, LSTM networks and GRU. These models were selected due to their suitability to time series data as well as their capabilities of recognizing sequential patterns in the network traffic, with LSTM networks being the superior of all in learning long-term dependencies of the data. Among the reasons for adopting such models in this work is the affordability of the economies delivered. That's the sheer speed in real-time attack detection that the models are known to allow. Some models that incorporated RNN, LSTM and GRU were also trained. Due to phased traffic across complex systems aimed at enhancing detection rates. These models were learnt in interludes in order to save time, along with hyperparameter tuning to manage factors such as learning rates, batch sizes, and number of epochs, among others, with the aim of improving model accuracy and reducing overfitting. The models achieved an assessment based on vital performance indicators, specifically accuracy, precision, recall, and the f1 score. Accuracy is the ratio of the correct predictions made to the total amount of predicted traffic given. Precision is the ratio of the correctly predicted attack traffic to the overall predicted attack traffic. Recall is the ratio of the predicted attacks captured relative to the total predicted attacks. The F1-score is the harmonic mean of precision and recall. High precision ensures accurate identification of attack traffic while minimizing false positives. A high recall reflects the system's ability to detect most attacks, reducing missed threats. Accuracy should be maximized by correctly classifying both attacks and normal traffic. The F1-score, which balances precision and recall, should approach 1, indicating a well-rounded, effective IDS. High values across all these metrics demonstrate a robust and reliable deep learning-based IDS.

Key performance metrics:

- Accuracy (AC): The number of correct forecasts relative to total traffic.

$$AC = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

TP: True Positive: The model correctly predicts the positive class.

TN: True Negative: The model correctly predicts the negative class.

FP: False Positive: The model incorrectly predicts the positive class.

FN: False Negative: The model incorrectly predicts the negative class.

- Precision (P): It is used to figure out how much of the attack traffic is accurately identified.

$$P = \frac{TP}{TP+FP} \quad (2)$$

- Recall (R): It measures the rate of anticipated attacks relative to overall attack traffic.

$$R = \frac{TP}{TP+FN} \quad (3)$$

- F1-score (F1): It provides a more accurate, balanced estimate of both precision and recall.

C. Adaptive DoS Attack Detection - Level 3

Earlier attackers used to flood the network or system with excessive amounts of traffic (ICMP, UDP), data or requests. Modern day attackers do not depend on only one type of DoS Attack. They use timeouts (LDoS), small queries (DNS/NTP Amplification) to bypass the DoS attack detection systems. Effective LDoS attack detection can be achieved using time-based network traffic properties.

Parameters used in the proposed mechanism are listed. Other than $Th_{Discard}$ and $TS_{Duration}$, all values are listed in Table I below:

TABLE I. LIST OF PARAMETERS

Symbol	Description
Con	No. of connections to the server in current Time slot
IA	Packet Inter-arrival time in current Time Slot
AIA	Average Packet Inter-arrival time
TO	No. of Time-outs in current Time Slot
DP	No. of discarded packets in current Time Slot
D	Time Slot Duration (in Seconds)
T	Traffic in current Time Slot
AT	Average Traffic per Time Slot
TH	Threshold number of packets discarded in a particular Time slot

$$AT = \alpha * AT + (1 - \alpha)T \quad (3)$$

$$AIA = \alpha * AIA + (1 - \alpha)IA \quad (4)$$

To analyse network traffic, there are two steps:

Step 1 - The following parameters are calculated using network traffic data from the network interface card: T, TO, DP

Step 2 - Sleeps for D before calculating the following parameters using information gathered from Step 1: IA, AIA, T. It employs the following algorithm using the previously computed parameters. Algorithm 1 details the Adaptive DoS Detection.

Algorithm 1: Adaptive DoS Detection

Step 1: Evaluate T and AT ;

If ($T \geqslant ((1+\beta) * AT)$)

Then Goto Step 2 Else Goto Step 4

Step 2: Evaluate TO with Con and DP with TH;

If ((TO >= 2*Con) and ($DP \geqslant TH$))

Then Goto step 3 Else Goto step 4

Step 3: Evaluate IA with AIA; If ($(AIA \leqslant \gamma * IA)$)

Then Assume Slow Rate DoS attack and not congestion

Else Goto step 4

Step 4: Halt processing till the end of current time slot interval

Traffic and Average traffic are compared. When the volume of traffic in the current time slot T exceeds $(1 + \beta)$ times the volume of that time slot's average traffic (AT), we proceed. We continue when there are more connections than timeouts, and more packets are refused, than the set threshold. The inter-arrival time and its average are then compared. We conclude that we are not experiencing congestion, but rather a Slow DoS assault when IA decreases. The processing is halted for that time period if any of the steps are unsuccessful.

IV. EXPERIMENTAL SETUP AND RESULTS

A. Flow Validation – Level 1

The experimental setup is shown in Fig. 3. The SDN consists of three hosts, namely Host 1 having an IP address of 192.168.0.2, Host 2 having an IP address of 192.168.0.3, and Host 3 having an IP address of 192.168.0.5. Here we use a single controller named Controller 1 with IP address 192.168.0.4. The Attacker having IP Address 192.168.0.1 is an external entity, which does fast, as well as Slow DoS-based attacks, as controlled by us.

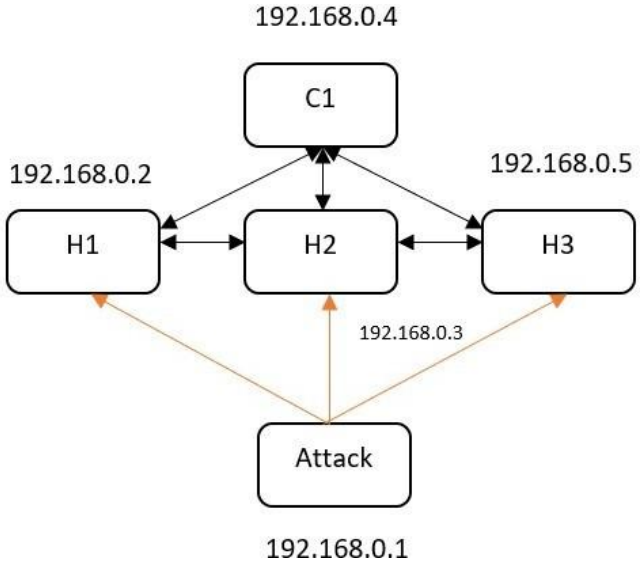

Fig. 3. Experimental setup (level 1).

We have noted their MAC addresses to verify. Data from the switches is extracted in CSV. Here we note the IP address, as well as the MAC Address, which is stored in the external storage. To store this data, remove the colon from the MAC Address and concatenate it with the Source Port Number (which is converted to a string). All the information is stored after applying SHA-256, as shown below. Each entry is stored in Table II to be matched. If the match is found, move to level 2 or wait for approval. During simulation, we found that the frequent IDs are stored and retrieved quickly from the database. New entries are not accepted and need time for approval from the controller. As soon as the entry from the attacker with source port number 6688 arrived, the hash matching failed. The attacker was blocked from moving to level 2, as expected.

TABLE II. CLIENT MATCH TABLE

SrcPort Nr	MAC Address	HashValue	Match
4233	00:0A:95:9D:68:16	b1b05f2632443eff30f3a71e5790298557b27d7d2f969e5859aee7f0081b547e	Yes
5311	00:1B:21:3F:55:A1	1057886baf2908dad4183baf45f653f2591c4db72182a56c6636858361a0b4fe	Yes
9612	00:23:A1:5B:44:65	35f956835f0c51c0aaaf3521f01120ad4a5386f6581031de524cd4f417bb9717	Yes
1239	00:17:CB:01:2A:B3	04674e1d4150b7c464c67277d0c928c6ce4fe52975b5fc32afdb56248a3a1bc6	Yes
6688	00:14:22:01:23:45	2519d99444f45cc88521846451c8b423ec345ecbd96cf069415c88f118c4a29b	No

B. Deep Learning based DoS Detection – Level 2

We have tested three datasets to find the best predictive model so that we can rule out flows based on the performance metrics discussed earlier [48].

The DoS attacks detection results are shown in Table III using the NSL-KDD dataset. The RNN + LSTM model records the highest accuracy of 95.14% in terms of finding DoS attacks. It is also the best in finding precision, recall and achieving F1-score. LSTM and its hybrid models also perform strongly with accuracies of more than 93%. In comparison, the accuracy rate for the SOM model is the lowest at 90.6%. This leads to the observation that, whereas models are useful in DoS detection, advanced deep learning models are on another level in DoS detection.

Table IV illustrates the performance of various machine learning models for detecting DoS attacks using the CIC-DDoS 2019 dataset. The LSTM model achieves the highest accuracy at 95.73%, followed closely by the RNN + LSTM model at 95.60%, showcasing their strong capability in identifying DoS attacks. Precision (P), recall (R), and F1 scores are consistent across all models, with values ranging between 0.85 and 0.88, indicating reliable detection performance across different approaches. However, while accuracy varies slightly, the models demonstrate comparable performance in precision, recall, and F1 score, with LSTM leading in accuracy, but all models performing similarly in overall detection quality. Thus, LSTM proves to be the most accurate, but other models like RNN + LSTM also offer strong results.

TABLE III. NSL-KDD DATASET [45]

Score	*SOM*	*RNN*	*GRU+LSTM+RNN*	*GRU*	*GRU+LSTM*	*LSTM*	*RNN+LSTM*
AC(%)	90.6	92.45	92.95	93.07	93.20	94.34	95.14
P	0.92	0.94	0.94	0.94	0.94	0.93	0.94
R	0.89	0.91	0.92	0.92	0.92	0.91	0.92
F1	0.90	0.92	0.93	0.93	0.93	0.92	0.93

TABLE IV. CIC-DDOS 2019 DATASET [46]

Score	*SOM*	*RNN*	*GRU+LSTM+RNN*	*GRU*	*GRU+LSTM*	*RNN+LSTM*	*LSTM*
AC(%)	90.6	92.53	92.61	93.52	93.67	95.60	95.73
P	0.88	0.88	0.88	0.88	0.87	0.87	0.87
R	0.87	0.87	0.87	0.87	0.87	0.87	0.87
F1	0.86	0.86	0.86	0.85	0.86	0.86	0.86

TABLE V. CIC-IOT 2023 DATASET [47]

Score	*SOM*	*GRU*	*GRU+LSTM+RNN*	*RNN*	*GRU+LSTM*	*LSTM*	*RNN+LSTM*
AC(%)	90.8	92.40	92.81	92.92	93.07	94.23	95.79
P	0.98	1	0.98	0.99	1	0.96	1
R	0.24	0.20	0.24	0.25	0.21	0.40	0.24
F1	0.38	0.33	0.38	0.40	0.35	0.56	0.39

Table V presents the performance of different models for detecting DoS attacks using the CIC-IoT 2023 dataset. The RNN + LSTM model achieves the highest accuracy at 95.79%, followed by LSTM at 94.23%, indicating their effectiveness in identifying DoS attacks in IoT environments. Interestingly, although GRU and GRU + LSTM exhibit perfect precision (P = 1), their recall values are significantly lower, leading to lower F1-scores. This indicates that while these models are highly precise, they miss a substantial portion of true positives. On the other hand, LSTM balances precision and recall, yielding a higher F1-score of 0.56, making it one of the more reliable models in terms of both accuracy and detection performance. While the RNN + LSTM model leads in accuracy, LSTM offers a better overall balance in terms of detection metrics.

The analysis in Fig. 4 highlights the effectiveness of models in accurately detecting DoS attacks on IoT networks, with RNN+LSTM being the most effective. Models like SOM underperform, indicating that not all models are equally suitable for this specific dataset.

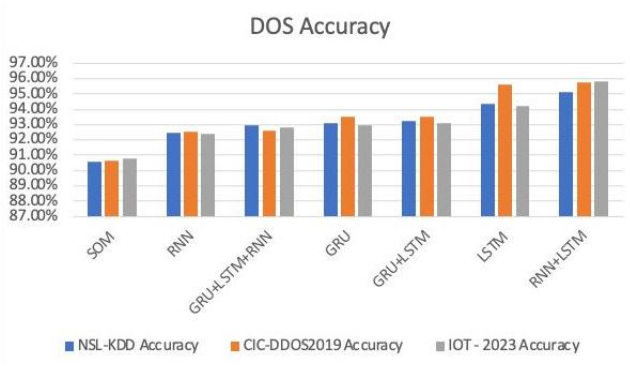

Fig. 4. Comparative Analysis of datasets based on their accuracy for DoS attack.

C. Adaptive DoS Attack Detection - Level 3

The experiment [49] is done using the Mininet Simulator, and we use a Python file to collect the flow, port, and timing statistics. Network simulation is run for 250 minutes, and 1,04,345 rows of data are collected. The simulation is run for a defined interval again, and more data can be collected. The test is divided into two phases. Initially, the simulation is started and kept ready to detect flows. Phase I – Normal simulation is done, allowing the flow of packets across all switches and controllers. Phase II – Attack is simulated, where 100 forged packets are sent. Following values of α, β, γ, and TH are taken as 0.75, 0.30, 0.70 and 1000, respectively. For Phase I: During this phase, data is gathered and data measuring the switch performance is measured, and the data is represented graph. From Fig. 5, it is clear that when there is no attack, the timeouts and discards are null. Also, the packets arrive in an orderly fashion.

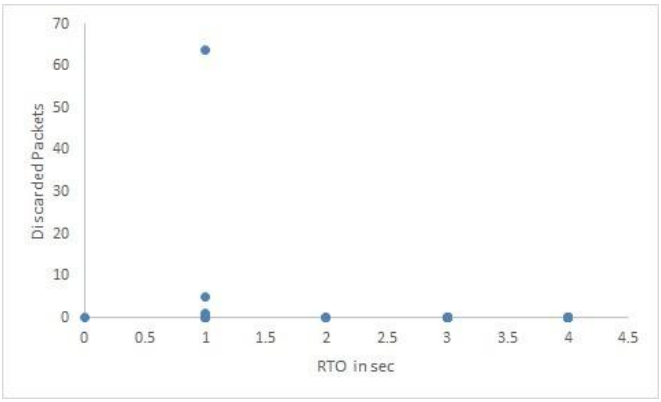

Fig. 5. Normal scenario (RTO vs discarded packets).

The typical traffic exceeds the current traffic in the time slot as soon as the attack begins, as shown in Fig. 6.

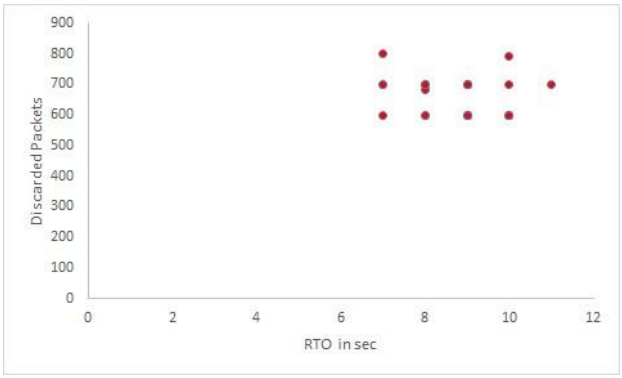

Fig. 6. Attack scenario (RTO vs discarded packets).

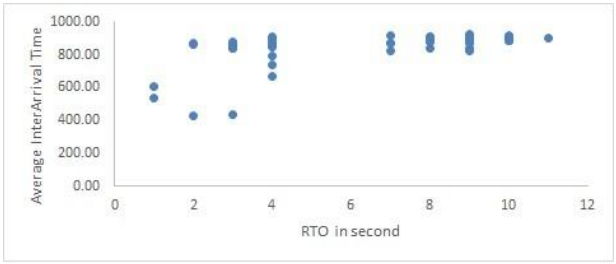

Fig. 7. Interarrival time.

Also, one can observe that the connections are 3 timeouts are 6. The number of packets discarded are way above 600. Looking at Fig. 7, the average interarrival time has gone below the packet interarrival time showing that it could be a low-rate DoS attack rather than congestion.

V. Conclusion

The results clearly indicate that the proposed method is effective against all modern DoS attacks in SDN. The level 1 zero trust approach ensures that only verified clients get access to the SDN. Level 2 based on deep learning, ensures that flows are classified and only verified flows are allowed to move forward. In case when the attacker bypasses levels 1 and 2, the level 3 adaptive DoS detection mechanism will ensure that both fast and low rates are thwarted. Researchers used blockchain for ensuring a zero-trust approach in SDN, which is time-consuming and complex. Hashing ensures that the initial check is done quickly and accurately. Researchers used various hybrid models, but the LSTM+GRU gives accurate predictions with better precision and recall. Researchers working on fast and low-rate attacks haven't worked on a simple, lightweight, combined approach relevant to real-time SDNs. This solution can be integrated with contemporary IDPS, as it is flexible and scalable.

References

[1] Jarraya Y, Madi T, Debbabi M. A survey and a layered taxonomy of software-defined networking. IEEE Commun. Surv. Tutor. 16(4):1955–80,2014.

[2] R. Nagarathna and S. M. Shalinie, SLAMHHA: A supervised learning approach to mitigate host location hijacking attack on SDN controllers, 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), 2017.

[3] Han T, Jan SRU, Tan Z, Usman M, Jan MA, Khan R, Xu Y. A comprehensive survey of security threats and their mitigation techniques for next-generation SDN controllers. Concurr. Comput.,3–5,2019.

[4] Wang Y, Hu T, Tang G ,Xie J ,Lu J . Sgs: safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking. IEEE Access , 2019.

[5] Shang G, Zhe P, Bin X, Aiqun H, Kui R. FloodDefender: protecting data and control plane resources under SDN-aimed DoS attacks. Proceedings of the IEEE INFOCOM, 2017

[6] Maham Iqbal, Farwa Iqbal, Fatima Mohsin, Dr.Muhammad Rizwan, Dr.Fahad Ah- mad, Security Issues in Software Defined Networking (SDN): Risks, Challenges and Potential Solutions, published in International Journal of Advanced Computer Science and Applications (IJACSA) Vol. 10, No. 10, 2019.

[7] Shang Gao, Zecheng Li, Bin Xiao, and Guiyi Wei,Security Threats in the Data Plane of Software-Defined Networks, published in IEEE Network, pg 108-113.August 2018.

[8] Durner R, Lorenz C, Wiedemann M, Kellerer W,Detecting and mitigating denial of service attacks against the data plane in software defined networks. Proceedings of the IEEE Conference on Network Softwarization: Softwarization Sustaining a Hyper- Connected World: en Route to 5G, NetSoft 2017.

[9] Danping A ,Pourzandi M ,Scott-Hayward S ,Song H ,Winandy M ,Dacheng Z , Threat analysis for the SDN architecture. Tech. Recomm. pp 1-21, July 2016.

[10] Jan J.Laan,Securing the SDN Northbound Interface, Master Research Project sub- mitted at University of Amsterdam. July 2015.

[11] B. Agborubere and E. Sanchez-Velazquez, OpenFlow Communications and TLS Se- curity in Software-Defined Networks, 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 560-566, 2017.

[12] Mohamed Mahdy, SDN Southbound Threats, Global Information Assurance Cer- tification by The SANS Institute, 2018.

[13] Thangasamy, Pandikumar, Early Detection of DDoS Attacks in a Multi-Controller Based SDN, published in International Journal of Engineering Science and Computing. 07. 13422-13429, 2017.

[14] Mousavi SM, St-Hilaire M. Early detection of DDoS attacks against SDN con- trollers. In: Proceedings of the International Conference on Computing, Networking and Communications, ICNC 2015; p. 77–81.,2015.

[15] Jiang Y, Zhang X ,Zhou Q ,Cheng Z . An entropy-based DDoS defense mechanism in software defined networks. In: Proceedings of the International Conference on Communications and Networking in China. p. 169–78, Springer; 2016.

[16] Kumar P, Tripathi M, Nehra A, Conti M, Lal C. SAFETY: early detection and mitigation of TCP SYN flood utilizing entropy in SDN. IEEE Trans. Netw. Serv.Manag,15(4):1545–59., 2018

[17] Kalkan K, Altay L, Gu¨r G, Alago¨z F. JESS: joint entropy-based DDoS defense scheme in SDN. IEEE J. Sel. Areas Commun. 36(10):2358–72.,2018

[18] AbdelAzim NM, Fahmy SF, Sobh MA, Eldin AMB. A hybrid entropy-based dos attacks detection system for software defined networks (SDN): a proposed trust mechanism. Egypt. Inform. J.,22(1):85–90, 2021.

[19] Singh J, Behal S. A novel approach for the detection of DDoS attacks in SDN using information theory metric. In: Proceedings of the 8th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE,p. 512–16, 2021.

[20] Mishra A, Gupta N, Gupta B. Defense mechanisms against DDoS attack based on entropy in SDN-cloud using pox controller. Telecommun. Syst.,77(1):47–62, 2021.

[21] AlEroud A, Alsmadi I. Identifying cyber-attacks on software defined networks: an inference-based intrusion detection approach. J. Netw. Comput. Appl.,80:152–64, 2017.

[22] Fan Z, Xiao Y, Nayak A, Tan C. An improved network security situation assessment approach in software defined networks. Peer-to-Peer Netw. Appl.,12(2):295–309, 2019.

[23] Meti N, Narayan D, Baligar V. Detection of distributed denial of service attacks using machine learning algorithms in software defined networks. In: Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE,p. 1366–71,2017.

[24] Makori DO, Ari S.Intelligent hybrid detection and mitigation of distributed denial of service attacks in SDN. Int. J. Eng. Comput. Sci.,6(12):22243–9, 2017.

[25] Myint Oo M, Kamolphiwong S, Kamolphiwong T, Vasupongayya S. Advanced support vector machine-(ASVM-) based detection for distributed denial of service (DDoS) attack on software defined networking (SDN). J. Comput. Netw. Commun. 2019,2019.

[26] Santos, R., Souza, D., Santo, W., Ribeiro, A., Moreno, E., Machine learning algorithms to detect DDoS attacks in SDN. Concurr. Comput. Pract. Exp.,2019.

[27] Xiao Y, Fan Z-j, Nayak A, Tan C-x. Discovery method for distributed denial-of- service attack behavior in SDNS using a feature-pattern graph model. Front. Inf. Technol. Electron. Eng.,20(9):1195–208, 2019.

[28] Tuan NN, Hung PH, Nghia ND, Van Tho N, Phan TV, Thanh NH, A robust TCP- SYN flood mitigation scheme using machine learning based on SDN. In: Proceedings of the International Conference on Information and Communication Technology Con- vergence (ICTC). IEEE,p. 363–8, 2019.

[29] Xu Y, Sun H, Xiang F, Sun Z. Efficient DDoS detection based on k-fknn in software defined networks. IEEE Access,7:160536–45,2019.

[30] Mehr SY, Ramamurthy B. An SVM based DDoS attack detection method for RYU SDN controller. In: Proceedings of the 15th International Conference on Emerging Networking Experiments and Technologies, p. 72–3, 2019.

[31] Dong S, Sarem M. DDoS attack detection method based on improved KNN with the degree of DDoS attack in software defined networks. IEEE Access 2019.

[32] Zhijun W, Qing X, Jingjie W, Meng Y, Liang L. Low-rate DDoS attack detection based on factorization machine in software defined network. IEEE Access 2020,8:17404–18,2020.

[33] Krishnan P, Duttagupta S, Achuthan K. Varman: Multi-plane security framework for software defined networks. Comput. Commun. 2019,148:215–39,2019.

[34] Musumeci F, Ionata V, Paolucci F, Cugini F, Tornatore M. Machine-learning- assisted DDoS attack detection with p4 language. In: Proceedings of the ICC 2020- 2020 IEEE International Conference on Communications (ICC). IEEE,p. 1–6,2020.

[35] Tan L, Pan Y, Wu J, Zhou J, Jiang H, Deng Y. A new framework for DDoS attack detection and defense in SDN environment. IEEE Access,8:161908–19,2020.

[36] Perez-Diaz JA, Valdovinos IA, Choo K-KR, Zhu D. A flexible SDN-based archi- tecture for identifying and mitigating low-rate DDoS attacks using machine learning. IEEE Access 2020,8:155859–72,2020.

[37] Ahuja N, Singal G, Mukhopadhyay D, Kumar N. Automated DDoS attack detection in software defined networking. J. Netw. Comput. Appl.,187:103108,2021.

[38] Tayfour OE, Marsono MN. Collaborative detection and mitigation of DDoS in software-defined networks. J. Supercomput.,1–25, 2021.

[39] Swami R, Dave M, Ranga V. Detection and analysis of TCP-SYN DDoS attack in software-defined networking. Wirel. Pers. Commun.,118(4):2295–317, 2021.

[40] Cui Y, Yan L, Li S, Xing H, Pan W, Zhu J, Zheng X. SD-Anti-DDoS: fast and efficient DDoS defense in software-defined networks. J. Netw. Comput. Appl.,68:65–79, 2016.

[41] Abdulqadder IH, Zou D, Aziz IT, Yuan B. Validating user flows to protect software defined network environments. Secur. Commun. Netw. 2018,2018.

[42] Gao D, Liu Z, Liu Y, Foh CH, Zhi T, Chao H-C . Defending against packet-in messages flooding attack under SDN context. Soft Comput.,22(20):6797–809,2018.

[43] Fichera S, Galluccio L, Grancagnolo SC ,Morabito G ,Palazzo S . Operetta: an openflow-based remedy to mitigate TCP synflood attacks against web servers. Com- put. Netw.,92:89–100,2015.

[44] Mohammadi R, Javidan R, Conti M. Slicots an SDN-based lightweight countermeasure for TCP SYN flooding attacks. IEEE Trans. Netw. Serv. Manag.,14(2):487–97, 2017.

[45] Kaggle, NSL KDD Dataset. 2023. [Online]. Available: https://www.kaggle.com/datasets/hassan06/nslkdd/. [Accessed: Jul. 20, 2024]

[46] UNB, CIC DDoS 2019 Dataset. Canadian Institute for Cybersecurity, University of New Brunswick. [Online]. Available: https://www.unb.ca/cic/datasets/ddos-2019.html. [Accessed: Jul. 20, 2024].

[47] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," Journal of Sensors, vol. 23, p. 5941, 2023.

[48] H. K. Lamba, H. Gala, R. Mathew, and S. Shinde, "AI based intrusion detection for DDoS and SQL injection attacks," in Proc. 2024 First Int. Conf. for Women in Computing (InCoWoCo), Pune, India, 2024, pp. 1–7.

[49] R. R. Mathew and A. Vidhate, "Adaptive DoS Attack Detection in SDN," 2024 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2024, pp. 1-5,