A Collaborative Approach to Detecting DDoS Attacks in SDN Using Entropy and Deep Learning

Narayan D.G., Heena W, and Amit K

KLE Technological University, Hubballi, Karnataka, India

https://doi.org/10.26636/jtit.2024.3.1609

Abstract — Software-defined networking (SDN) is an approach to network management allowing to enhance the performance of the network and making it more flexible. The centralized architecture of SDN makes it vulnerable to cyberattacks, especially distributed denial of service (DDoS) attacks. Existing research investigates the detection of DDoS attacks separately on the control plane and data plane. However, there is a need for efficient and accurate detection of these attacks using features obtained from both control and data planes. Therefore, we present a mechanism for identifying DDoS attacks using entropy, multiple feature selection mechanisms, and deep learning. Initially, we use entropy on the control plane to detect anomalous activity and identify suspicious switches. Next, we capture traffic on the suspicious switches to detect DDoS attacks. To detect these attacks, we utilize multi-layer perceptron (MLP) deep learning models, convolutional neural network (CNN), and the long short-term memory (LSTM) approach. An InSDN dataset is used to train the model and test data are generated using Mininet emulation and the Ryu controller. The results reveal that LSTM outperforms MLP and CNN, achieving an accuracy of 99.83%.

Keywords — DDoS, deep learning, LSTM, machine learning, random forest, SDN

1. Introduction

Software-defined networking (SDN) is a popular technology providing a centralized and efficient network infrastructure. In contrast to traditional distributed and composite network designs, SDN separates the network traffic forwarding process (infrastructure layer) from the control process (control plane). This separation offers numerous advantages, such as programmable configuration capabilities. By leveraging open APIs within software applications, SDN allows network behavior to be programmed centrally across the entire network and its constituent parts. However, SDN is also associated with some drawbacks stemming from its centralized architecture, i.e. a single point of failure. In the event of an intrusion, this can lead to disruptions throughout the entire network.

Distributed denial of service (DDoS) attacks pose a significant risk to SDN security. The primary objective of these attacks is to prevent authorized users from accessing resources. Additionally, these attacks consume the targeted source's capacity, leading to delays and disruptions in normal operations. While several studies have explored the use of classification algorithms to identify and prevent DDoS attacks, current research faces such challenges as achieving realistic performance rates for the detection system, addressing detection delays, and effectively handling high volumes of network traffic [1]. The absence of a comprehensive real-world solution to detect and prevent different types of attacks has motivated the development of a dedicated DDoS attack detection system.

Recently, AI-driven methodologies for identifying DDoS attacks have been utilized [2] and most of these research projects rely on machine/deep learning approaches to detect DDoS attacks [3]. While some studies in the literature have focused on addressing DDoS attacks by employing entropy or unsupervised learning, they have turned out to be less accurate [4]. Hence, there exists a necessity to integrate an entropy-based approach with machine/deep learning approaches to augment detection precision.

In this work, we have introduced a mechanism that employs an entropy-based approach to identify suspicious switches in the SDN control plane. We utilize a multi-feature selection mechanism with a deep learning-based method on the data plane to identify attacker machines. This deep learning module captures packets from potentially compromised switches over a predetermined time frame and employs a trained model to identify and classify potential attacks. Through the integration of both entropy and deep learning modules, the proposed approach is capable of effectively detecting unauthorized attacks and provides robust defense mechanisms against potential unknown threats targeting the application layer.

This study offers the following contributions:

- an entropy-based method is proposed that serves as a primary detection method on the control plane. This module identifies OpenFlow switches that are potentially under attack,
- machine and deep learning modules are developed that capture traffic on the susceptible switches to identify a DDoS attack. A multi-feature selection mechanism using SelectKBest, ANOVA F-value scores and feature importance scores from the random forest classifier is proposed,

- a comprehensive performance evaluation of the proposed method is performed, including an assessment of the accuracy and efficiency of the detection techniques.

The remaining part of the article is divided into four sections. Section 2 reviews related research, while Section 3 outlines the proposed model and algorithms. The results are presented in Section 4, and conclusions are drawn in Section 5.

2. Related Work

In [5], the authors developed a real-time DDoS attack detection mechanism using a deep neural network solution and the CICIDS 2017 dataset. The results show that the proposed model performs better than other models with different datasets, achieving an accuracy of 97.59% while consuming less time and resources. The authors of [6] initially set up a DDoS attack detection mechanism on the data plane. Then, to enhance the precision of detection, k-nearest neighbors and k-means machine learning algorithms were employed. These algorithms utilize a combination of five features to identify vulnerable flows in the network. The results indicate that the proposed method achieves a superior accuracy rate of 97.53%, outperforming both entropy and self-organizing map techniques.

Some of the work is based on statistical techniques. In [7], a joint entropy-based DDoS defense scheme is proposed. The authors use joint entropy measurements from multiple network features to detect anomalies. These features include, inter alia, packet rate, flow count, and byte count. By calculating the entropy of these features and analyzing their joint distributions, deviations indicative of DDoS attacks are identified. Paper [8] introduces an entropy-based detection method that analyzes various network traffic metrics to detect DDoS attacks. This approach aims to address challenges posed by the dynamic and potentially large-scale nature of DoS attacks in SDN networks. Through experimental validation, the paper demonstrates the effectiveness of such a framework in accurately identifying DDoS attacks.

In [9], the authors proposed a DDoS attack detection method using a combination of entropy and ensemble learning techniques. An inspection module deployed on the data plane collects real-time packets from the network through the edge switch. When vulnerable packets are detected, the controller is immediately notified. To achieve more accurate detection, a detection module is established on the control plane. This module incorporates five features and utilizes the random forest algorithm to classify network traffic. Once an attack packet is confirmed, the controller takes immediate action by dropping the packets from the edge switch, thereby blocking the attack. The simulation results, focusing on ICMP and SYN flood attacks, demonstrate a quick response and a more efficient detection rate.

In article [10], a cross-plane DDoS mitigation approach is introduced relying on two events to detect and react to DDoS attacks. Firstly, a data plane detection module is established, including a flow-monitoring algorithm. Secondly, a machine learning-based detection module is implemented on the control plane to ensure more precise attack detection. The results demonstrate that the system achieves a higher accuracy rate of 96%. In [11], a hybrid approach is proposed that combines features from both control and data planes. An entropy module is utilized as the primary detection feature, while a machine learning module is employed for more accurate attack detection. The results reveal that the integration of data and control plane features leads to better accuracy with reduced overhead.

Another framework, as proposed in [12], focuses on the control plane. The work emphasizes the security of both data and control planes. A deep neural network is used for packet classification. Preliminary detection occurs on the data plane, and attack traffic is mitigated. To detect attacks at an earlier stage, a detection module is set up on the control plane, providing better service to users using a priority queue. In article [13], the authors developed a safeguard mechanism to protect the control plane from DDoS attacks in software-defined networks. The safeguard framework is deployed across multiple control planes. The detection module works on the data plane.

In [14], the authors proposed a DDoS attack detection method using new features. On the control plane, flow messages are collected from the data plane, which is followed by extraction of important features. Next, the SVM model is trained using a dedicated dataset and is tested with real-time data. The results reveal that the system achieves a better accuracy rate compared to other existing solutions.

In paper [15], a method for detecting DDoS attacks by combining general entropy and the neural network particle swarm optimization with personal best (PSO-BP) is introduced. The entropy module is positioned on the OpenFlow switch to detect traffic, and the detection result is classified into normal and anomalous. To detect an attack, the controller uses the neural network PSO-BP. Results reveal that the DDoS attack detection rate is accurate and imposes a low CPU load on the controller. In [16], the authors proposed a mechanism for detecting and mitigating DDoS attacks using the data layer. The results show that the time to detection and mitigation is between 100 and 150 s.

The authors of [17] have developed a method to mitigate DDoS attacks originating from compromised OpenFlow switches on the controller. The research shows how a DDoS attack can be launched on an SDN controller with cooperating switches. It also directly mitigates such an attack within the second recurring request. In [18], the authors have proposed security mechanisms against attacks based on entropy in SDN-cloud using a POX controller. The developed mechanism has three advantages: high detection rate, low false-positive rate, and the ability to mitigate the attack. The proposed framework has the highest accuracy of 98.2%.

In [19], the authors introduced a framework to detect DDoS attacks with the cross-plane. The primary detection mechanism is set up at the data plane, and for precise DDoS attack detection the mechanism is set up at the control plane. The result reveals that the mentioned system is efficient and reduces

detection delay with less communication over the heading rate at the southbound.

Article [20] presents a mechanism on the control plane for a hybrid classification model, which utilizes SVM and SVM-SOM (self-organized map) machine learning algorithms for packet classification. The machine-learning module responsible for classification is implemented on the control plane. The results indicate that SVM-SOM achieves higher accuracy compared to SVM and SOM, with a lower false-positive rate than other algorithms. In paper [21], an efficient framework for an intrusion detection system on the data plane using data plane development kit (DPDK) is used to process packets at high speeds and monitor the data plane. The results demonstrate that the proposed DDoS attack detection framework effectively addresses high-speed networks in intrusion detection systems.

An automated system using a hybrid machine-learning algorithm for traffic packet classification is presented in [22]. The results show that a hybrid model combining SVM with random forest classification achieves the highest accuracy of 98.8%, with a low false-positive rate. The authors of [23] focus on the detection and mitigation of DDoS attacks using deep learning. CNN and recurrent neural network (RNN) long short-term memory models are employed for classification. RNN and LSTM perform well in detecting and mitigating DDoS attacks, achieving an accuracy of 89.63%. Performance of deep learning models is better when using a split ratio of 7:3 compared to 8:2 or 6:4. In [24], a big data framework for large-scale SDN that combines OpenStack with SDN using the ONOS controller, Apache Spark, and Apache Kafka is researched. The work overcomes the limitations of traditional data processing and validates the strength, scalability, and efficiency of the proposed framework.

The framework from [25] mitigates DDoS attacks on the data plane. The detection method analyzes header fields of the TCP connection and hashes them. Upon detecting an attack, the packets are dropped. In [26], the authors develop a mechanism using deep learning. They employ a standard dataset for traffic classification after preprocessing. The results show that the stacked auto-encoder multi-layer perceptron achieves an accuracy rate of 99.75%. The aim of the paper is to classify traffic using deep learning techniques.

The majority of the work mentioned above was carried using Internet datasets, such as CICIDS and NSL-KDD. Furthermore, most of the authors did not focus on feature selection techniques. We address these research gaps by employing a hybrid approach incorporating the inSDN dataset [27] and multi-feature selection methods.

3. Proposed Methodology

This section outlines the design of the proposed system, including the entropy module, the multi-feature selection mechanism, the InSDN dataset utilized for model training, and the deep learning algorithms employed for classifying network traffic in a test environment, using Mininet emulation.

3.1. System Model

As shown in Fig. 1, the process of detecting DDoS attacks in SDN environments involves a two-layered methodology combining entropy-based anomaly detection and the deep learning technique. The approach begins by leveraging entropy thresholds to identify anomalous network traffic behaviors. Entropy, in this context, measures the unpredictability or randomness of source IP addresses within a defined packet window (e.g. 50 packets). When entropy drops below a specified threshold during this window, it means that potentially suspicious activity has been detected. We monitor the drop in entropy for 10 consecutive packet windows (entropy drop counter), providing a cumulative measure of suspicious behavior.

Upon detecting sustained drops in entropy indicative of potential DDoS activity (e.g. when entropy drop counter reaches a predefined threshold, such as 10, i.e. an anomaly threshold), the next phase involves identifying the switches associated with the suspicious IP addresses using SDN flow rules. These switches are critical nodes within the network infrastructure where abnormal traffic patterns are observed.

To further validate and enhance the detection process, a deep learning module is employed. This module operates by capturing real-time network traffic data from identified switches for fixed time intervals (3 s) and then applies a deep learning algorithm to classify traffic patterns into those that are normal and malicious. The deep learning model is trained using the InSDN dataset to recognize the signature characteristics of DDoS attacks, enabling it to accurately distinguish between legitimate and malicious traffic.

The iterative, entropy-based anomaly detection process combined with deep learning ensures a proactive approach to detecting and mitigating DDoS attacks in SDN environments.

3.2. Entropy Computation

In an OpenFlow-enabled SDN network, when a TCP or UDP packet arrives at a switch, the switch checks its flow table for an existing rule. If no rule is found, the switch encapsulates the packet in a “packet_in” message and sends it to the SDN controller over a TCP connection. The controller analyzes the packet_in message, determines the optimal forwarding path, and sends back a “flow_mod” message with the new rule to the switch. The switch then forwards the packet according to the new rule and uses this rule to handle future packets.

We then use the packet_in to compute entropy and detect the suspicious host machines. In the SDN controller, entropy for packet_in messages using the source IP is calculated. This measures the randomness or unpredictability of the source IP addresses from a dataset of 50 packets. Entropy is a statistical measure that quantifies the level of uncertainty or diversity in a dataset. For packet_in messages, this can be done by first collecting the source IP addresses from 50 incoming packet_in messages.

To calculate entropy, the following steps are taken:

- determine the frequency of each unique source IP address in the 50 packets,

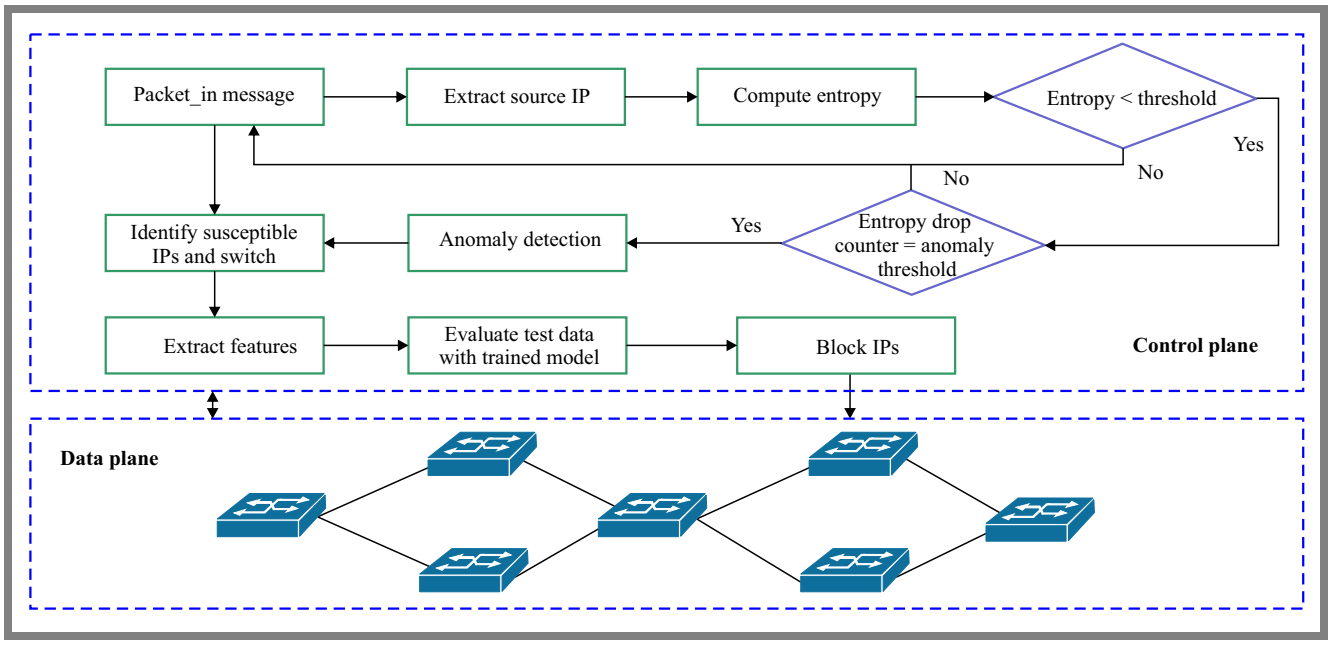

Fig. 1. Diagram of the proposed system.

- compute probability p_i of each source IP i occurring, which is the frequency of i divided by 50 (the total number of packet_in messages),
- use the entropy formula $H(X) = -\sum p_i \log_2(p_i)$, where X represents the set of source IPs.

The resulting entropy value $H(X)$ indicates the distribution of source IP addresses. A higher entropy value suggests a more diverse and unpredictable set of source IPs, while a lower entropy value indicates less diversity and more predictability. For instance, if each of the 50 packets has a unique source IP, the entropy will be higher compared to a scenario where all packets originate from a few IPs. The process of computing entropy is shown in Algorithm 1.

We utilize an entropy threshold of 0.5 to flag potentially suspicious IPs. The entropy drop counter serves as a metric to monitor such activity. Entropy is calculated over a 50–packet window. If entropy falls below the threshold during this window, indicating suspicious behavior, the counter is incremented. Detection of suspicious activity occurs when entropy remains below the threshold for 10 consecutive instances. Suspicious switches associated with these IPs are identified through flow rules. Table 1 presents the statistically derived entropy values from the different scenarios which were used to set the 0.5 threshold.

Once an anomaly is detected, a deep learning module is then employed to capture test traffic for 3 s on the identified switch to detect potential DDoS attacks. This iterative process continues until the end of the emulation process. The algorithm for malicious activity detection is presented as Algorithm 2.

3.3. Data Preprocessing

InSDN [27] addresses the lack of publicly available datasets for evaluating IDSs in SDN environments. The dataset is created using a virtualized network testbed with various VMs to simulate both legitimate and malicious traffic.

Algorithm 1 Calculate_Entropy subroutine

Input: IP_List, packet_count
Output: $Entropy$

```
Entropy ← 0.0
for each (IP, value) in IP_List.items() do
    p ← value / packet_count
    if p > 0 then
        Entropy ← Entropy − (p · log2(p))
    end if
end for
return Entropy
```

The dataset includes attacks such as DDoS, scanning, spoofing, and other common network attacks with 81 features. This dataset was used for training and the test data were collected from Mininet emulation with Ryu as the controller. Preprocessing of the InSDN dataset involves several key steps to prepare the data for effective use in training and evaluating machine learning models for DDoS attack detection in SDN environments. The preprocessing phase includes cleaning the dataset to remove noise and inconsistencies, handling missing values, and normalizing numerical features to ensure uniformity and enhance model performance. Feature engi-

Tab. 1. Entropy value with different attack rates.

Scenario	Attack packet rate [%]	Entropy value
1	80	0.39
2	60	0.54
3	40	0.66

Algorithm 2 Detect_Malicious_Activity procedure

Input: Network traffic
Output: Hosts with malicious activity

Initialize $packet\_count \leftarrow 0$
Initialize $Entropy\_Drop\_Counter \leftarrow 0$
Initialize $Threshold \leftarrow 0.3$
Initialize IP_List as empty dictionary
Initialize $Flag\_DDoS\_Detected \leftarrow 0$
while receiving packets **do**
    $src\_ip \leftarrow$ get_source_ip(packet)
    $packet\_count \leftarrow packet\_count + 1$
    if src_ip exists in IP_List **then**
        $IP\_List[src\_ip] \leftarrow IP\_List[src\_ip] + 1$
    else
        $IP\_List[src\_ip] \leftarrow 1$
    end if
    if $packet\_count = 50$ **then**
        $Entropy \leftarrow$ Calculate_Entropy
        if $Entropy \leqslant Threshold$ **then**
            Increment $Entropy\_Drop\_Counter$
        else
            $Entropy\_Drop\_Counter \leftarrow 0$
        end if
        if $Entropy\_Drop\_Counter = 10$ **then**
            $Flag\_DDoS\_Detected \leftarrow 1$
            Identify the suspicious IPs
            Identify suspicious switches
            Invoke DL model on identified switches
        end if
        $packet\_count \leftarrow 0$
        Clear IP_List
    end if
end while

neering techniques are applied to extract relevant features that are capable of distinguishing between normal and malicious network behaviors.

Additionally, data augmentation methods are utilized to increase diversity and robustness of the dataset, ensuring comprehensive coverage of potential attack scenarios. The test data is generated using a Mininet-emulated topology, as shown in Fig. 2. Mininet provides a realistic virtual network environment in which various network configurations and topologies can be simulated. This allows for thorough testing and evaluation of network performance and security measures, ensuring that the results apply to real-world scenarios.

3.4. Multiple Feature Selection Methods

Three feature extraction methods, namely SelectKBest, ANOVA (analysis of variance) F-value scores, and feature importance scores from a random forest (RF) classifier were employed to select the top ten features from a total of 81. Each method initially selected 20 features, and the ten most frequently occurring features across these selections were then chosen based on their rankings.

Tab. 2. Top 10 features selected from the InSDN dataset.

No.	Name of feature	Description
1	Src IP	IP address source from which the packet is sent
2	Dst IP	IP address at which the packet is intended to be sent
3	Protocol	The communication network protocol (e.g. TCP, UDP)
4	Pkt Len Std	The standard deviation of the flow's packet lengths
5	Flow ID	Distinct number for every flow
6	Bwd Pkt Len Mean	The mean packet length in the reverse direction
7	Pkt Len Var	Variation in the duration of packets within the flow
8	Src Port	Source port number used by the packet
9	Bwd Seg Avg Size	Segment size average moving backward
10	Bwd Pkt Std Len	Packet length standard deviation in the reverse direction

SelectKBest is a feature extraction method that selects the top k features based on univariate statistical tests. This method scores each feature individually using a chosen statistical test and retains the highest-scoring features, making it useful for quickly identifying the most relevant features for predictive modeling.

ANOVA F-value scores are used to assess the significance of the differences between group means in a dataset. In feature selection, ANOVA F-value scores evaluate the relationship between each feature and the target variable, selecting features that show the highest F-values, indicating a stronger discriminatory power between classes.

Feature importance scores from a RF classifier provide a measure of the significance of each feature in predicting the target variable. The random forest algorithm computes these scores by averaging the reduction in impurity across all trees in the forest for each feature. Features with higher importance scores contribute more to the model's predictive accuracy, making them valuable for feature selection.

By employing these three methods, researchers can identify the most informative features in their dataset, improving the performance and efficiency of machine learning models. The ten best-selected features are represented in Tab. 2. Algorithm 3 illustrates the steps for feature selection.

3.5. Deep Learning Module

We have used MLP, CNN and LSTM to evaluate the performance of the classification approach. The deep learning model is trained using the InSDN dataset, and testing is performed

Algorithm 3 Feature selection using multiple methods

Input: Dataset D with 81 features, target variable y
Output: Top 10 selected features

Initialize parameters:
$n \leftarrow 81$
$k \leftarrow 20$
$m \leftarrow 10$
Apply feature selection methods:
Initialize an empty list $selected\_features$
for each method in {SelectKBest, ANOVA F-value, RandomForest} **do**
    if method is SelectKBest **then**
        Perform univariate statistical tests
        Select the top k features based on their scores
    else if method is ANOVA F-value **then**
        Compute the F-value for each feature
        Select top k features based on F-value scores
    else if method is RandomForest **then**
        Train a RandomForest classifier on the dataset
        Calculate feature importance scores
        Select the top k features based on their importance scores
    end if
    Append the chosen features to $selected\_features$
end for
Combine selected features:
Initialize a dictionary $feature\_frequency$ to count the frequency of each feature
for feature in $selected\_features$ **do**
    if feature is already in $feature\_frequency$ **then**
        Increment the count of $feature$ by 1
    else
        Add $feature\_frequency$ with count 1
    end if
end for
Select top m features:
Sort the features in $feature\_frequency$ by their frequency in descending order
Select the top m features with the highest frequency
Output: Return the list of the top 10 features

using real-time traffic generated with the help of Mininet and the Ryu controller in SDN.

Algorithm 4 illustrates the steps involved in the design.

4. Results and Discussion

As shown in Fig. 2, a Mininet emulator is used to create the SDN environment. We employed a leaf and spine topology consisting of three OpenFlow virtual switches and 20 hosts. All the switches are managed by the Ryu controller, which provides centralized control and enables dynamic network management. Both normal and abnormal traffic are sent from various source IPs to multiple target machines. We also simulated spoofed source IP addresses for the attack. The `hping3` tool is used to generate the traffic.

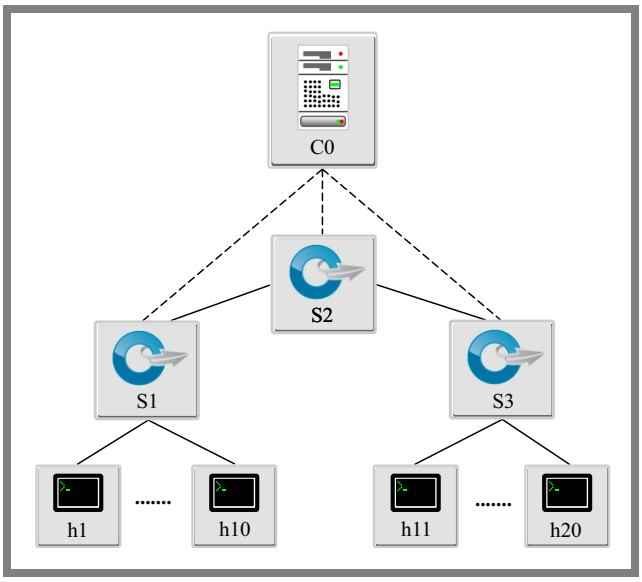

Fig. 2. Mininet network topology.

Algorithm 4 LSTM model for DDoS attacks detection

Input: Preprocessed dataset $selected\_features\_df$ with features and labels
Output: Trained LSTM model and evaluation metrics

Build LSTM model:
    Initialize a sequential model
    Add an LSTM layer with ReLU activation
    Add a dense layer with sigmoid activation function
    Compile the model using Adam optimizer and binary_cross_entropy loss function
Train the model:
for $epoch \leftarrow 1$ to $epochs$ **do**
    Train the model on X_{train} and y_{train} for one epoch with $batch\_size$
    Validate the model on X_{test} and y_{test} during training
    if $epoch$ is the final epoch **then**
        Print "Final epoch reached: Epoch" $epoch$
    else
        Print "Continuing to next epoch: Epoch" $epoch$
    end if
end for
Evaluate the model:
    Evaluate the model on the test set X_{test} and y_{test}
    Print the test accuracy

4.1. Machine Learning Results

We use machine learning (ML) algorithms to detect the attack on suspicious switches. Based on anomaly detection, traffic is captured, over a fixed time period, from suspicious switches in the SDN environment. Figure 3 illustrates the accuracy of various ML algorithms. The trained model utilized for this evaluation is based on the InSDN dataset.

We assessed the performance of machine learning algorithms using the InSDN dataset with the top 10 features selected, from an initial set of 81, as training and test data generated using the Mininet emulator and the Ryu controller. The results

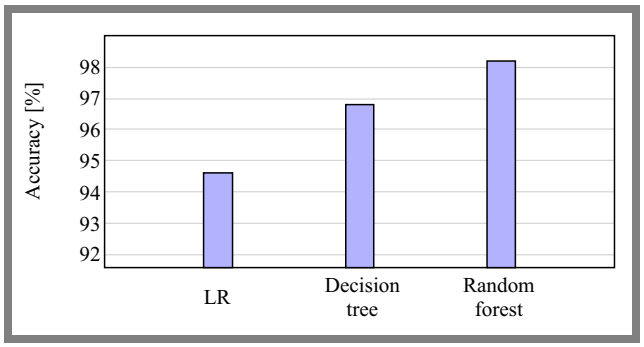

Fig. 3. Accuracy of machine learning algorithms.

showed that logistic regression (LR) achieved an accuracy of 94.6%, the decision tree classifier reached an accuracy of 96.8%, and the random forest (RF) classifier outperformed the other models with an accuracy of 98.2%.

Table 3 shows the precision, recall and F1-score levels of ML models. The results indicate that RF performs better than decision tree and LR in terms of accuracy due to its ensemble learning approach, which reduces overfitting and stabilizes predictions by averaging multiple trees. This method effectively handles complex relationships and interactions among features, making it more robust to noise and outliers. RF also evaluates feature importance across all trees, ensuring that key features are prioritized.

These factors collectively contribute to the superior accuracy of random forest, compared with other models. The receiver operating characteristic curve (ROC) and confusion matrix are represented in Figs. 4 and 5, respectively. The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR), with a red line representing the performance of the RF model and a blue dashed line representing a random classifier, for comparison. The coordinates used for the RF model indicate its performance at various threshold levels, demonstrating its effectiveness in distinguishing between classes with the result of 0.99. A ROC value of 0.99 indicates that the RF model has a high TPR and a low FPR, demonstrating its effectiveness and reliability in detecting the targeted instances. Figure 5 presents a confusion matrix for a random forest model, illustrating its performance in terms of classification accuracy. This graph illustrates the model's effectiveness and the distribution of prediction errors, emphasizing its performance in distinguishing between the two classes.

Tab. 3. Performance of the ML models.

ML algorithms	Precision (P)	Recall (R)	F1-score
Decision tree	0.966	0.97	0.968
Logistic regression (LR)	0.951	0.94	0.942
Random forest (RF)	0.984	0.98	0.982
Trained with the InSDN dataset and tested with data generated using Mininet emulation			

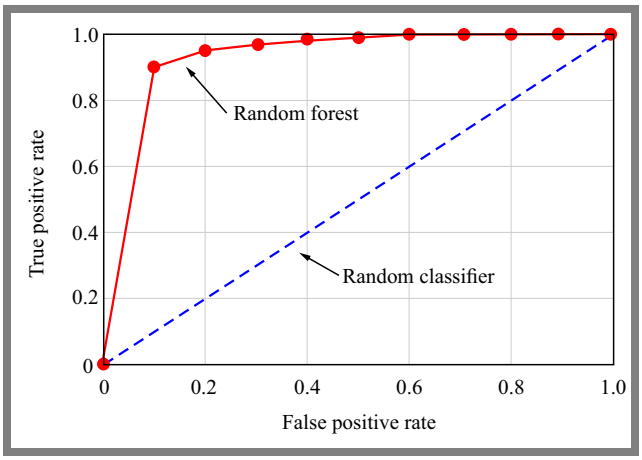

Fig. 4. Receiver operating characteristic.

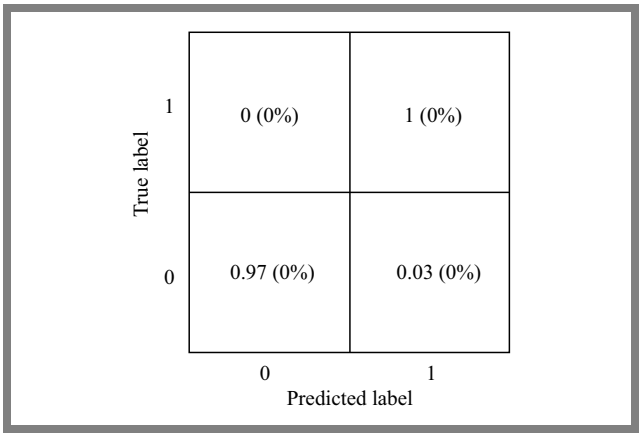

Fig. 5. Confusion matrix of random forest.

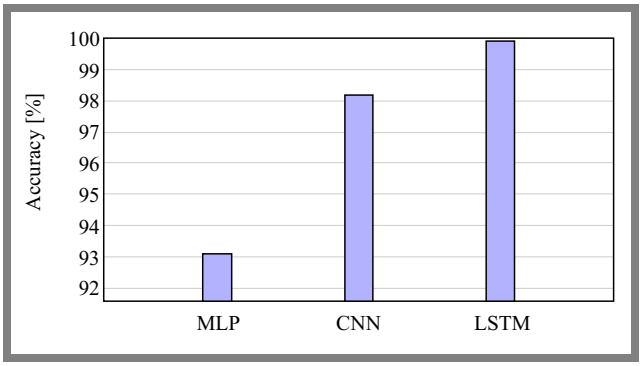

Fig. 6. Accuracy of deep learning algorithms.

4.2. Deep Learning Results

Deep learning models are essential due to their ability to capture complex patterns and relationships within data. These models handle high-dimensional data effectively, distinguishing between normal and malicious traffic, even when significant variability is at play. The scalability of deep learning allows it to be applied in large-scale network environments without performance losses.

We use three deep learning models: LSTM, CNN, and MLP, each with different hyperparameter tuning. The LSTM model consists of an LSTM layer followed by dense layers, capturing sequential patterns within the feature set, with a softmax out-

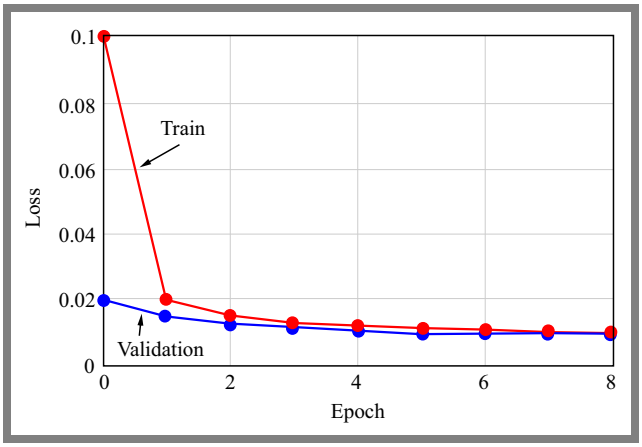

Fig. 7. Model loss of LSTM.

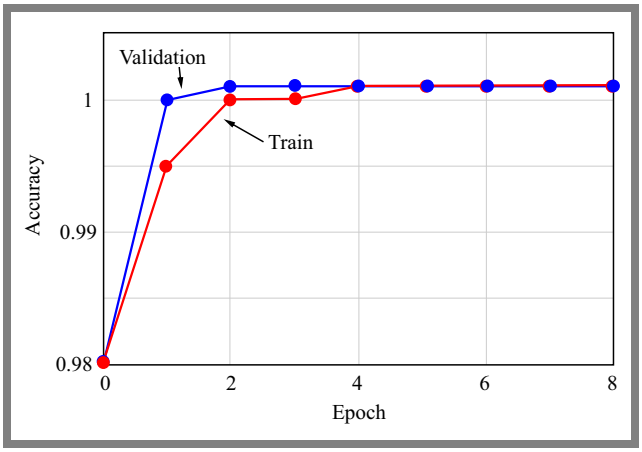

Fig. 8. Model accuracy of LSTM.

put layer. It is trained using categorical cross-entropy loss and the Adam optimizer. The MLP model is configured with an input layer matching the number of selected features, several hidden layers with ReLU activation functions, and a softmax output layer. It is trained using backpropagation with a categorical cross-entropy loss function. The CNN model includes an input layer reshaped for a 2D convolutional architecture, multiple convolutional layers with ReLU activation, max-pooling layers, and fully connected layers, culminating in a softmax output layer.

As shown in Fig. 6, LSTM achieved the highest accuracy among the three models (99.83%), outperforming both CNN and MLP models, thus highlighting its superior capability in capturing temporal relationships within the selected features.

LSTM is better at managing sequential data and capturing temporal dependencies, making them suitable for analyzing networktraffic. The built-in memory mechanism retains long-term dependencies to store anomalous traffic patterns. LSTM has allowed to enhance the process of analyzing temporal patterns and detecting anomalies within network traffic data.

Figure 7 represents the loss graph during the training phase of the dataset and the validation loss obtained after evaluating the validation dataset. The graph demonstrates that the loss experienced decreases gradually over the course of multiple epochs. On the other hand, the validation loss exhibits fluctuations at epoch 8 before stabilizing over the remaining epochs. Figure 8 depicts the accuracy graph during the training phase of the dataset and the validation accuracy obtained after evaluating the validation dataset. The results reveal that the accuracy improves gradually over the number of epochs during training. However, at epoch 8, there is a drop in validation accuracy, followed by a sudden increase, after which the accuracy remains stable over subsequent epochs.

In summary, the LSTM model achieves the highest accuracy level (99.83%) among the deep learning models. The random forest model achieves the highest accuracy (98.2%) among machine learning models. Therefore, we conclude that deep learning outperforms machine learning. Consequently, we use the LSTM model to detect DDoS attacks in data captured from suspicious switches identified by the entropy module. This collaborative approach helps reduce the DDoS attack detection lead time.

5. Conclusions

SDN is a centrally controlled and programmable network architecture that enables a flexible network environment. The primary concern in SDN networks is the issue of security. This work aims to address the security challenge by designing a method for detecting DDoS attacks in SDN. The proposed approach consists of two modules. Initially, the entropy module is deployed on the control plane to detect suspicious switches and hosts based on entropy. Then, a deep learning module is used to detect DDoS attacks using the test data captured from the suspicious switches. The deep learning module performs the important feature extraction and attack classification tasks using deep learning algorithms, such as MLP, CNN, and LSTM. Based on the experiments and evaluation results, the system achieves an average accuracy of 99.83% in categorizing various types of DDoS flooding attacks, e.g. UDP, TCP-SYN, and ICMP. The experiments indicate that the deep learning techniques yield better evaluation results, particularly in terms of model accuracy and model loss.

As future work, we plan to implement the proposed mechanism within a real-time SDN testbed based on the OpenStack cloud.

References

[1] B. Alhijawi *et al.*, "A Survey on DoS/DDoS Mitigation Techniques in SDNs: Classification, Comparison, Solutions, Testing Tools and Datasets", *Computers and Electrical Engineering*, vol. 99, art. no. 107706, 2022 (https://doi.org/10.1016/j.compeleceng.2022.107706).

[2] A.A. Bahashwan *et al.*, "A Systematic Literature Review on Machine Learning and Deep Learning Approaches for Detecting DDoS Attacks in Software-defined Networking", *Sensors*, vol. 23, no. 9, art. no. 4441, 2023 (https://doi.org/10.3390/s23094441).

[3] I.A. Valdovinos, J.A. Pérez-Díaz, K.-K.R. Choo, and J.F. Botero, "Emerging DDoS Attack Detection and Mitigation Strategies in Software-defined Networks: Taxonomy, Challenges and Future Directions", *Journal of Network and Computer Applications*, vol. 187, art. no. 103093, 2021 (https://doi.org/10.1016/j.jnca.2021.103093).

[4] H. Zhang, L. Zhou, and J. Lei, "Renyi Entropy-based DDoS Attack Detection in SDN-based Networks", *2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI)*, Changchun, China, 2023 (https://doi.org/10.1109/ICETCI57876.2023.10176631).

[5] A. Makuvaza, D.S. Jat, and A.M. Gamundani, "Deep Neural Network (DNN) Solution for Real-time Detection of Distributed Denial of Service (DDoS) Attacks in Software Defined Networks (SDNs)", *SN Computer Science*, vol. 2, 2021 (https://doi.org/10.1007/s42979-021-00467-1).

[6] L. Tan *et al.*, "A New Framework for DDoS Attack Detection and Defense in SDN Environment", *IEEE Access*, vol. 8, pp. 161908–161919, 2020 (https://doi.org/10.1109/ACCESS.2020.3021435).

[7] K. Kalkan, L. Altay, G. Gür, and F. Alagöz, "JESS: Joint Entropy-based DDoS Defense Scheme in SDN", *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2358–2372, 2018 (https://doi.org/10.1109/JSAC.2018.2869997).

[8] R.N. Carvalho, J.L. Bordim, and E.A.P. Alchieri, "Entropy-based DoS Attack Identification in SDN", *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, Rio de Janeiro, Brazil, 2019 (https://doi.org/10.1109/IPDPSW.2019.00108).

[9] S. Yu *et al.*, "A Cooperative DDoS Attack Detection Scheme Based on Entropy and Ensemble Learning in SDN", *EURASIP Journal on Wireless Communications and Networking*, 2021 (https://doi.org/10.21203/rs.3.rs-154522/v1).

[10] B. Han *et al.*, "OverWatch: A Cross-plane DDoS Attack Defense Framework with Collaborative Intelligence in SDN", *Security and Communication Networks*, 2018 (https://doi.org/10.1155/2018/9649643).

[11] A. Yadav *et al.*, "A Hybrid Approach for Detection of DDoS Attacks Using Entropy and Machine Learning in Software Defined Networks", *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Kharagpur, India, 2021 (https://doi.org/10.1109/ICCCNT51525.2021.9580057).

[12] B. Celesova, J. Val'ko, R. Grezo, and P. Helebrandt, "Enhancing Security of SDN Focusing on Control Plane and Data Plane", *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, Barcelos, Portugal, 2019 (https://doi.org/10.1109/ISDFS.2019.8757542).

[13] Y. Wang *et al.*, "SGS: Safe-guard Scheme for Protecting Control Plane Against DDoS Attacks in Software-defined Networking", *IEEE Access*, vol. 7, pp. 34699–34710, 2019 (https://doi.org/10.1109/ACCESS.2019.2895092).

[14] W.G. Gadallah, N.M. Omar, and H.M. Ibrahim, "Machine Learning-based Distributed Denial of Service Attacks Detection Technique using New Features in Software-defined Networks", *International Journal of Computer Network Information Security*, vol. 13, no. 3, pp. 15–27, 2021 (https://doi.org/10.5815/ijcnis.2021.03.02).

[15] Z. Liu, Y. He, W. Wang, and B. Zhang, "DDoS Attack Detection Scheme Based on Entropy and PSO-BP Neural Network in SDN", *China Communications*, vol. 16, no. 7, pp. 144–155, 2019 (https://doi.org/10.23919/JCC.2019.07.012).

[16] H.S. Abdulkarem and A. Dawod, "DDoS Attack Detection and Mitigation at SDN Data Plane Layer", *2020 2nd Global Power, Energy and Communication Conference (GPECOM)*, Izmir, Türkiye, 2020 (https://doi.org/10.1109/GPECOM49333.2020.9247850).

[17] R. Sanjeetha, A. Pattanaik, A. Gupta, and A. Kanavalli, "Early Detection and Diminution of DDoS Attack Instigated by Compromised Switches on the Controller in Software Defined Networks", *2019 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics*, Manipal, India, 2019 (https://doi.org/10.1109/DISCOVER47552.2019.9007925).

[18] A. Mishra, N. Gupta, and B.B. Gupta, "Defense Mechanisms Against DDoS Attack Based on Entropy in SDN-cloud Using POX Controller", *Telecommunication Systems*, vol. 77, pp. 47–62, 2021 (https://doi.org/10.1007/s11235-020-00747-w).

[19] X. Yang, B. Han, Z. Sun, and J. Huang, "SDN-based DDoS Attack Detection with Cross-plane Collaboration and Lightweight Flow Monitoring", *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Singapore, 2017 (https://doi.org/10.1109/GLOCOM.2017.8254079).

[20] V. Deepa, K.M. Sudar, and P. Deepalakshmi, "Detection of DDoS Attack on SDN Control Plane using Hybrid Machine Learning Techniques", *2018 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, 2018 (https://doi.org/10.1109/ICSSIT.2018.8748836).

[21] J.E. Varghese and B. Muniyal, "An Efficient IDS Framework for DDoS Attacks in SDN Environment", *IEEE Access*, vol. 9, pp. 69680–69699, 2021 (https://doi.org/10.1109/ACCESS.2021.3078065).

[22] N. Ahuja, G. Singal, D. Mukhopadhyay, and N. Kumar, "Automated DDOS Attack Detection in Software Defined Networking", *Journal of Network and Computer Applications*, vol. 187, art. no. 103108, 2021 (https://doi.org/10.1016/j.jnca.2021.103108).

[23] D. Gadze *et al.*, "An Investigation into the Application of Deep Learning in the Detection and Mitigation of DDOS Attack on SDN Controllers", *Technologies*, vol. 9, art. no. 14, 2021 (https://doi.org/10.3390/technologies9010014).

[24] P.T. Dinh and M. Park, "BDF-SDN: A Big Data Framework for DDoS Attack Detection in Large-scale SDN-based Cloud", *2021 IEEE Conference on Dependable and Secure Computing (DSC)*, Fukushima, Japan, 2021 (https://doi.org/10.1109/DSC49826.2021.9346269).

[25] R. Durner, C. Lorenz, M. Wiedemann, and W. Kellerer, "Detecting and Mitigating Denial of Service Attacks Against the Data Plane in Software Defined Networks", *2017 IEEE Conference on Network Softwarization (NetSoft)*, Bologna, Italy, 2017 (https://doi.org/10.1109/NETSOFT.2017.8004229).

[26] N. Ahuja, G. Singal, and D. Mukhopadhyay, "DLSDN: Deep Learning for DDOS Attack Detection in Software Defined Networking", *2021 11th International Conference on Cloud Computing, Data Science and Engineering (Confluence)*, Noida, India, 2021 (https://doi.org/10.1109/Confluence51648.2021.9376879).

[27] M.S. Elsayed, N.-A. Le-Khac, and A.D. Jurcut, "InSDN: A Novel SDN Intrusion Dataset", *IEEE Access*, vol. 8, pp. 165263–165284, 2020 (https://doi.org/10.1109/ACCESS.2020.3022633).

Narayan D.G., Ph.D.
School of Computer Science and Engineering
https://orcid.org/0000-0002-2843-8931
E-mail: narayan_dg@kletech.ac.in
KLE Technological University, Hubballi, Karnataka, India
https://www.kletech.ac.in

Heena W, Ph.D.
School of Computer Science and Engineering
https://orcid.org/0009-0009-7998-3049
E-mail: heenakwalikar@gmail.com
KLE Technological University, Hubballi, Karnataka, India
https://www.kletech.ac.in

Amit K, Ph.D.
Department of MCA
https://orcid.org/0000-0002-8917-8678
E-mail: amitk@kletech.ac.in
KLE Technological University, Hubballi, Karnataka, India
https://www.kletech.ac.in