THE ASIAN BULLETIN OF BIG DATA MANAGMENT
Vol. 5. Issue 2 (2025)
https://doi.org/10.62019/abbdm.v5i2.327

Toward Robust SDN Architectures: A Machine Learning Approach to DDoS Detection

Khaliq Ahmed, Khalid bin Muhammad, Ali Ahmad Siddiqui* Abdul Khaliq*

Chronicle

Article history
Received: Feb 3, 2025
Received in the revised format: March 17, 2025
Accepted: April 4, 2025
Available online: May 5, 2025

Khaliq Ahmed is currently affiliated with the Department of computer science Iqra university, Karachi, Pakistan.
Email: Khaliq@iqra.edu.pk

Khalid bin Muhammad is currently affiliated with the Department of computer science, Faculty of engineering, science, technology and management, Ziauddin university, Karachi, Pakistan.
Email: Khalid.muhammad@zu.edu.pk

Ali Ahmad Siddiqui is currently affiliated with FEST, Iqra University, Pakistan.
Email: alisiddiqui@iqra.edu.pk

Abdul Khaliq is currently affiliated with the CCSIS, IoBM, Karachi, Pakistan.
Email: khaliq@iobm.edu.pk

Corresponding Author*

Keywords: DDOS attack, Machine Learning, Software-Defined Networking

Abstract

This paper offers an extensive exploration of Distributed Denial of Service (DDoS) attacks targeting Software-Defined Networking (SDN) environments and their centralized controller vulnerability. Based on a virtual testbed developed using Mininet and Ryu controller, different DDoS attacks, i.e., SYN, UDP, and ICMP floods, were simulated to check their effect on SDN performance indicators like CPU utilization, latency, throughput, and saturation of the flow table. Tests indicated that SYN flood attacks put the controller under most stress, generating excessive Packet_In messages, 100% CPU spikes, and extreme packet loss. UDP floods caused link saturation and even higher packet loss from stateless operation. ICMP floods had lesser but still significant impact on performance. In order to mitigate these vulnerabilities, the research employed a machine learning-based detection model that was trained on traffic logs-extracted features. Six supervised models were compared, with XGBoost having the best accuracy (98.2%), then Random Forest and Neural Networks. Inter-arrival time, flag count, and bytes per second were discovered to be the key indicators of malicious behavior. The results identify the need for embedding smart, real-time detection systems into SDN frameworks in order to achieve network robustness and lay the foundation for active DDoS mitigation techniques.

INTRODUCTION

During the last ten years, remarkable developments in computing technologies have taken place, fueled by tireless work from developers and researchers. These advancements have evolved a sophisticated digital environment that facilitates users to accomplish numerous tasks swiftly at low costs. Among these advancements, cloud computing is a groundbreaking platform, providing on-demand access to resources on a pay-as-you-go paradigm (Butt et al. 2023). Its advantages have attracted significant attention from both government agencies and IT sectors, mainly because of its ability to minimize infrastructure expenses and maintenance loads. Virtualization is a pillar of cloud computing, which makes the dynamic assignment of resources like storage, software, and processing capacity possible. This method has drastically transformed the way networks have been built and maintained over the last decade (Sharma et al. 2022**)**. SDN

facilitates currently networks by transforming physical network links to logical links and allowing central control of networks services (Khaliq et al., 2018). SDN enables cloud service providers to achieve cost optimization, smart worldwide connectivity, better security, and lower downtime (Khaliq et al., 2023).

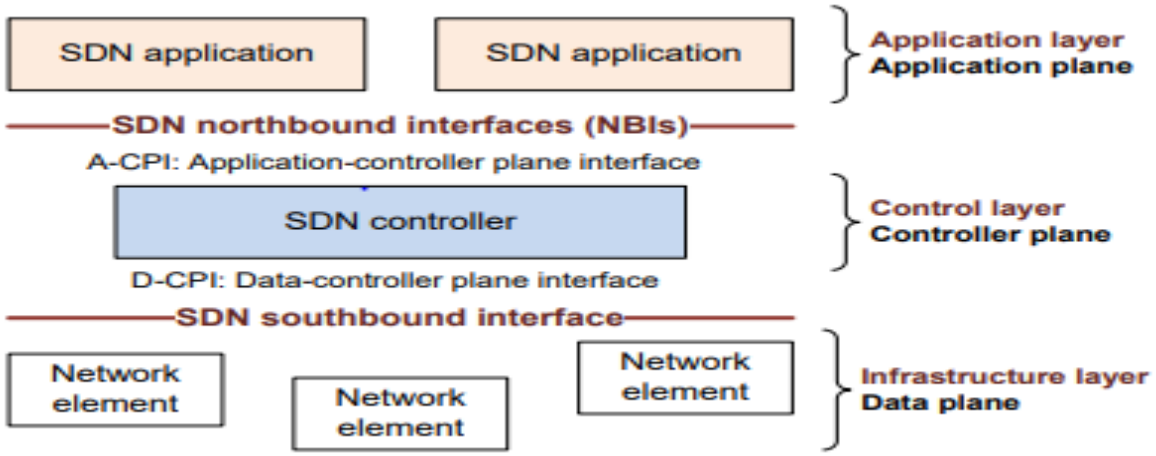

Figure 1.

SDN facilitates currently networks as shown in figure 1 by transforming physical network links to logical links and allowing central control of networks services (Khaliq et al., 2018). SDN enables cloud service providers to achieve cost optimization, smart worldwide connectivity, better security, and lower downtime (Khaliq et al., 2023). SDN offers an application layer for programs that offer efficient solutions for key network operations like intrusion detection, auto-scaling, and monitoring of networks (Gadallah et al., 2021). The evolution of Software defined- networking and cloud computing enables cloud service providers to accomplish lots of virtual networks deprived of depending on conventional separation methods like VLAN. SDN is a major model shift in designing networks.

Software-Defined Networking (SDN) brings a paradigm change by separating the control plane from the data plane, making it possible for network administrators to dynamically observe, control, and optimize network traffic. In cloud computing environments, SDN makes it easy to allocate network resources flexibly, with infrastructure being able to change in real time to match changing workloads and application needs. This ability to adapt ensures optimal performance under changing circumstances. Nevertheless, in spite of this segregation of architecture, SDN is still exposed to traffic overloads—most of all those precipitated by Distributed Denial of Service (DDoS) attacks. Attackers can take advantage of vulnerabilities found in main SDN elements such as the controller, southbound and northbound APIs, and switches, thus compromising the network's security and stability (Rawat et al., 2023).

DDoS attacks constitute an existential threat to cloud computing for both service providers and end-users through interrupting access to services. These attacks usually consist of several compromised systems that overwhelm a victim with redundant traffic to make it unusable. such attacks take advantage of distributed nodes to overconsume resources. Recent figures show that prominent cloud platforms such as Amazon AWS EC2 and Rackspace have incurred major monetary losses as a result of these intrusions.

Appalling, the initial half of the year 2020 alone experienced around 5.4 million DDoS attacks, highlighting the increasing volume and effect of these attacks as shown in figure 2. (Valdovinos et al., 2021)(Pandey, 2021)(Ateş et al., 2019)(Raj & Pani, 2021)(Dahiya & Gupta, 2020) It is critical for an organization to have the capability to detect and mitigate Distributed Denial of Service (DDoS) attacks effectively to ensure operational resilience and service availability. For this purpose, it is critical to have a solid system that is capable to evaluate real-time networks traffic and sense irregularities prior to them turning into major threats. A mechanism that can automate network traffic classification and notify administrators when suspicious patterns are detected is critical. While several mechanisms for DDoS defense have been proposed, the ongoing nature of attack innovations creates a never-ending challenge for early detection. Current solutions are likely to provide early warning functionality but suffer from high false positives. On the other hand, more responsive and accurate approaches may incur operational overhead or cause resource inefficiencies and economic costs. To overcome these issues, we introduce a new framework called RDAER, which combines essential elements—feature selection, clustering of traffic, prediction of attacks, and classification of traffic—into a single model specifically designed for SDN-based cloud environments. The framework is intended to improve both the accuracy and responsiveness of DDoS detection to provide more effective protection against new threats. (Dahiya & Gupta, 2020) (MahdaviHezavehi & Rahmani, 2020)(Sadeghpour et al., 2021)(Lee et al., 2011)(Ribeiro et al., 2023).

In the last few years, many intrusion detection methods have been proposed; yet, comparatively fewer have been actually designed with a focus on anomaly detection. The main difficulty is to design an adaptive, resilient, and easy-to-use methodology that can identify anomalous activity in the presence of the rising complexity and speed of today's cyber threats, as well as the vast extent of today's networks. Different methods—like data mining, statistical analysis, machine learning, and knowledge-based systems—have been utilized to identify network anomalies. From 2010 onwards, most of these methods have been applied in isolation, resulting in high false positive rates. In 2015, the research trend changed towards hybrid methods, where the combination of two or more intrusion detection methods allowed for better detection of DDoS attacks. (Venkatesh & Anuradha, 2019)(Fouladi et al., 2020). Though these developments enhanced detection accuracy, they also added greater computational requirements and resource utilization. In reaction, follow-up studies investigated the identification and selection of the best features that could minimize detection time and system complexity without sacrificing effectiveness (Karthick et al., 2022).

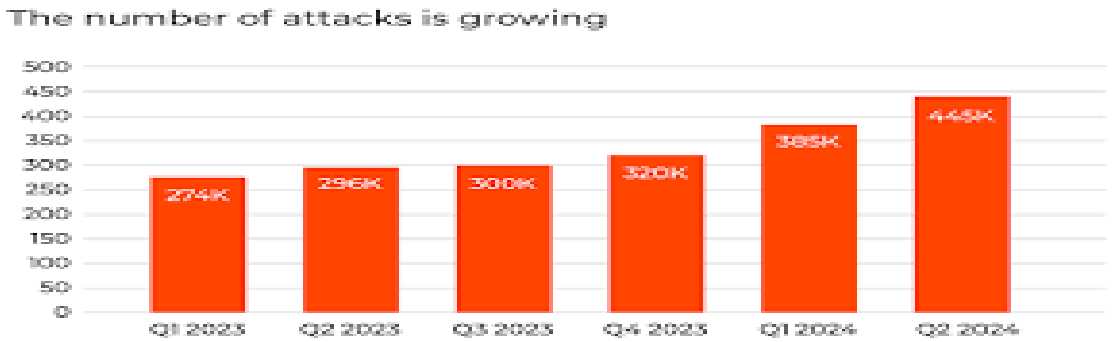

Figure 2.

A paper in (Alubaidan et al., 2023) utilized several time series analysis methods to predict DDoS attacks by anticipating traffic patterns based on anomaly scores. These scores were then utilized to label traffic as normal or malicious. Another study (Samaan & Jeiad, 2023) proposed a novel method for DDoS detection that utilized Lattice Structural Access Rates, referred to as S2RF2S, for filtering features. For classification, the research employed a Soft-Max Behavioral-Based Ideal Neural Network (SxB2IN2), which attained a detection rate of 90%. In another study (Zhou et al., 2023), a set of machine learning models such as Long Short-Term Memory (LSTM), Logistic Regression, Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbor (KNN) was employed to emphasize the critical role of proper feature selection in improving detection performance. Of these, the RF classifier achieved the best DDoS detection precision of 99% with only 11 selected features.

In addition, research (Najafimehr et al., 2022) utilized Apache Spark to implement a model for DDoS attack detection in SDN environments. Of the algorithms tested, the Decision Tree (DT) classifier was found to be the most accurate at 93.6% and hence would be the best for real-time deployment. Further, research work in (Dinh & Park, 2020) used the popularly used LSTM model to identify anomalous traffic within distributed SDN-based edge computing networks. Comprehensive testing on five datasets, each involving three typical type of attacks, found that the suggested CoWatch model attained 93.30% prediction accuracy in identifying and predicting Distributed DoS attack and their respective threat streams through a cooperative forecast process.

(Peng et al., 2018) proposed a mixture prototypical that combined both unsupervised and supervised learning approaches. To distinguish between attack traffic and normal data, they used a clustering algorithm in conjunction with a variety of flow-based criteria. The generated clusters are designated using a categorization technique founded on precise statistical variables. (Suárez-Varela & Barlet-Ros, 2018 proposed an improved technique to detect compromised openflow switches that uses a multivariate time series analysis and Recurrent Neural Networks (RNNs) for classification purposes. Their investigation yielded an accuracy of 96.99% and a detection rate of 98.51%. Peng et al. 2019 proposed an anomalous method for detection of flows for SDN that uses the double P-value of transductive confidence machines in conjunction with the KNN algorithm.

In a separate work [28], a scalable classification and flow monitoring approach for open flow was proposed, based on a sampling strategy. This categorization methodology combines deep inspection of packets with ML methods. Paper [29] promotes the use of ML to combat DDoS attacks that are cloud-based. The project includes acquiring input data from cloud modules, dimensionality reduction, feature extraction, noise filtration and classification using a ResNet-101-based Kernel Extreme Learning Machine (KELM). In a separate study, researchers used K-means clustering and agglomerative, as well as (PCA) Principal Component Analysis, to extract features. A system of voting is used to evaluate if the data is normal or suggests an assault. This technique had a 96.66% classification accuracy.

Mosayeb et al. 2022created RAD, a three-phase statistical approach meant to sense Distributed DoS attacks by scoring users and categorizing them as benign or assaulting. The three important parameters— jitter, drop and delay—are utilized to recognize probable attack behavior. The RAD model was examined by means of the CICDDoS2019

dataset and was compared against four additional recognition methods, and it reached 80% precision and 99% recall. Rajasree et al. (2019) developed a clustering method fuzzy bat that classifies comparable patterns and predicts anomalous behavior using a deviating anomaly score. The correlation of events between the cloud service provider's virtual machine instance and the list of questionable sources is utilized to identify the malicious source.

This strategy lowered the number of false alarms while effectively detecting anomalies. Girish et al. (2023) created neural network that uses bidirectional and stacked LSTM models. The model was tested with data obtained from OpenStack, which included 10 characteristics and a categorization label. By means of the binary cross-entropy function as a loss function, the proposed model achieved 94.61% training set accuracy and 93.98% test set accuracy. A study of current studies demonstrates a lack of inclusive method that assimilates time-series analysis, clustering, event correlation, feature selection techniques, and for earlier detection of Distributed DoS attacks in Software defined cloud setups. Correlation of events is critical in classifying trends in network and abnormalities in distributed networks. Furthermore, an urgent improvement is needed to accurately detect DDoS.

PROPOSE WORK

Experimental Setup

The testbed is constructed based on a virtual SDN setup via Mininet network emulation and Ryu controller as the SDN control plane as shown in figure 3. The goal is to emulate and analyze the effect of Distributed Denial of Service (DDoS) attacks on the SDN design, specifically concentrating on the centralized controller and

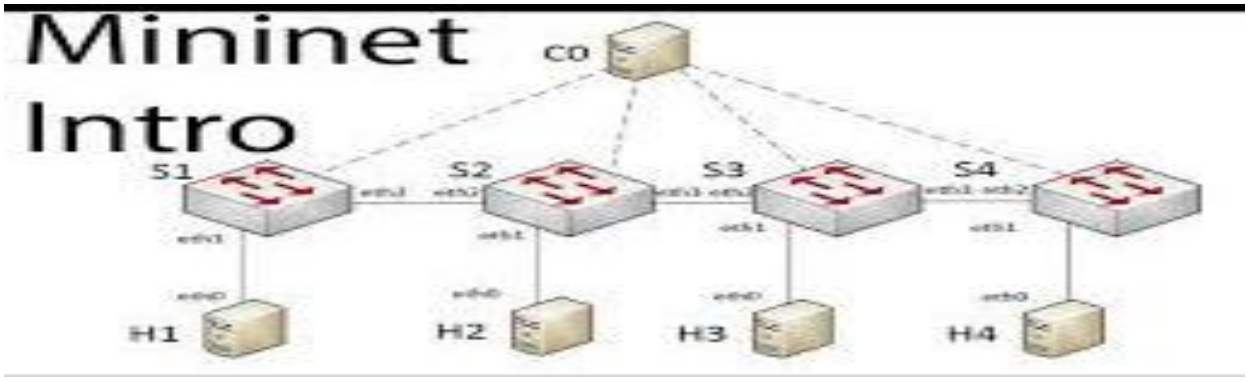

Figure 3.
flow management system.

Network Topology

The emulated topology is made up of four switches (s1 to s4) and sixteen hosts (h1 to h16), arranged as follows:

•Three legitimate hosts are connected to each switch, making a total of 12, and one attacker host making a total of 4.

•tThe switches are connected in a mesh-like structure with a single controller node.

The topology was deployed in Mininet through a custom Python script. The script initializes OpenFlow switches and connects them to the controller. The controller (Ryu) is executed independently and communicates through the loopback interface.

Traffic Generation

Legitimate Traffic

•tLegitimate hosts (h1–h12) exchange traffic through iperf and ping both in TCP and ICMP modes.

•Several simultaneous sessions are established to mimic the real-world workload.

- Each test duration set to 300 seconds to record consistent behavior.

Attack Traffic

The attacker host (h13 to h16) each initiate a separate type of attack traffic using hping3 and Scapy. The below-mentioned attack vectors were taken into account:

- **SYN Flood**

o Targets TCP handshake by sending TCP SYN packets to arbitrary or specific ports.

o Command: hping3 -S -p 80 -i u1000 --flood <target IP>

- **UDP Flood**

o Floods the target with UDP packets without waiting for replies.

o Command: hping3 --udp -p 80 -i u1000 --flood <target IP>

- **ICMP Flood**

o Sends ICMP Echo Requests at a high rate.

o Command: ping -f <target IP>

These attacks were directed against either individual victim hosts or broadcast across subnets based on the experiment.

Monitoring Tools and Metrics

We monitored performance using the following tools:

- CPU usage: Monitored through htop and controller logs.
- Latency and throughput: Measured through ping and iperf.
- Packet loss: Captured through tcpdump and interface statistics.
- Flow statistics: Retrieved through ovs-ofctl dump-flows and special Ryu event handlers.

Experimental Scenarios and Observations

Scenario 1: Baseline (No Attack)

To create a control, the baseline test was run without malicious traffic.

Observations:

•Setup time of flows: 6–10 ms

•Packet delivery ratio (PDR): ~98–100%

•CPU utilization: 2–5% on controller

•Flow table entries: Average of 40 per switch

•Throughput: Consistently above 95 Mbps for each iperf session

The network operated in its best state, with negligible delay and no packet loss. Controller performance was stable, and the switches handled their flow tables without overflows.

Scenario 2: SYN Flood Attack

The SYN flood attack was launched from four attacker hosts concurrently. Each host transmitted 10,000 SYN packets per second to random ports on various legitimate hosts. Because each SYN packet is distinct, the switches forward the packets to the controller through Packet_In messages, causing new flow entries.

Observations:
•\tCPU Load: The CPU usage of the controller reached 90–95%, sometimes 100%.
•\tLatency: Legitimate flow setup time rose significantly to 150–300 ms.
•\tPacket Drops: Doubled tenfold. An average of 30–50% of valid packets were dropped during load.
•\tFlow Table Saturation: All switches indicated full capacity in their flow tables within 15–20 seconds.
•\tThroughput: Reduced to as low as 25–40 Mbps per session.

Analysis

SYN floods bombarded the controller with too many new flow requests. As each new packet contained a different 5-tuple (source IP, destination IP, source port, destination port, protocol), the switches could not match them to any existing rules, and this created a Packet In flood. The Ryu controller, which was not optimized for such high loads, took time to process, leading to delays and packet losses.

Scenario 3: UDP Flood Attack

Every attacker created about 15,000 UDP packets per second to random ports on random hosts. UDP is stateless, hence more challenging to handle in real time.
Observations:
•_CPU Load: Averaged between 80–90% on the controller.
•_Latency: Lower than SYN flood, averaging 100–200 ms.
•_Packet Loss: Around 45–60% over legitimate flows.
•_Throughput: Reduced to ~20–30 Mbps per flow.
•_Link Saturation: Experienced heavy congestion on switch and controller links.
In contrast to TCP, UDP does not use handshaking. Yet, the randomness of source and

destination addresses created many flow entries and control messages. As UDP floods also put pressure on bandwidth between switches and controller, it led to link congestion and queue overflow at the switch level. The controller was often backlogged by packets waiting for decision, creating more latency and less throughput.

Scenario 4: ICMP Flood Attack

This test mimicked a large amount of ICMP echo requests sent at around 5000 packets per second per attacker.

Observations:

- CPU Load: Increased to 50–65% — less than SYN and UDP attacks.
- Latency: Went up moderately to 80–120 ms.
- Packet Loss: Around 25–30% during peak.
- Flow Table Usage: Increased but did not fill up as quickly as other attacks.
- Throughput: Affected slightly (~70 Mbps per session).

Analysis:

ICMP floods were not as successful in overloading the controller due to the fact that most packets used repetitive patterns that would match current flow rules. Nevertheless, because some packets utilized different ICMP IDs and sequence numbers, a reasonable number of Packet_In events were still caused. The effect was a poor but not totally disrupted service.

DETAILED RESULTS ANALYSIS

Table 1.
CPU Usage Comparison

Scenario	Controller CPU Usage (%)
Baseline	2–5
SYN Flood	90–100
UDP Flood	80–90
ICMP Flood	50–65

The SYN flood utilized the most CPU due to its random and dynamic packet nature, which made the controller create new flow rules perpetually. The event loop of the controller became the main point of bottleneck as shown in table 1.

Table 2.
Flow Setup Time

Scenario	Avg Flow Setup Latency (ms)
Baseline	8–10
SYN Flood	150–300

Scenario	Avg Flow Setup Latency (ms)
UDP Flood	100–200
ICMP Flood	80–120

As more Packet_In messages are generated, the controller queues them for processing, and the increased latency is a reflection of both queue delays and rule installation delays on switches.

Packet Delivery Ratio The packet loss in the UDP flood is slightly higher than in SYN due to concurrent link saturation and controller overload. ICMP flood remains moderate in its impact due to the relatively simple processing requirements as shown in table 2.

Table 3.

Scenario	Packet Loss Rate (%)
Baseline	<2
SYN Flood	35–50
UDP Flood	45–60
ICMP Flood	25–30

Table 4.
Throughput of Legitimate Hosts

Scenario	Avg Throughput per Host (Mbps)
Baseline	95–100
SYN Flood	30–50
UDP Flood	20–40
ICMP Flood	60–70

The throughput degradation aligns closely with CPU and packet loss metrics. The more severe the attack on controller resources, the lower the resulting throughput for normal users.

Flow Table Entries

- Under SYN flood, flow tables filled up within 20 seconds.
- Under UDP flood, flow tables showed near-capacity within 40–50 seconds.
- ICMP flood resulted in only partial occupation (~60–70%).

Once the flow tables are full, new flows are dropped, and switches revert to default behavior, either dropping packets or forwarding to controller — further worsening the bottleneck as shown in table 3.

Discussion: These tests show the susceptibility of SDN controllers to DDoS attacks. Although centralized control is desirable for network programmability, it presents serious danger in the face of malicious traffic.

The most effective attack was the SYN flood, which flooded the controller with high-entropy flows. The UDP flood further exacerbated problems by saturating links and

inducing buffer overflows. The ICMP flood, although less substantial, still impaired network performance.

From a research perspective, this underscores the necessity for:

- Distributed control planes to prevent single points of failure.
- Early detection based on packet rates and entropy
- Proactive rate limiting and filtering at the switch level.
- Hardening of controllers, such as multi-threading, load balancing, and smart queue management.

These results support the creation of DDoS-resistant SDN frameworks, which incorporate defense mechanisms at both control and data planes.

DDoS Attack Prediction Using Machine Learning in SDN Environments

With the concentration of control in SDN designs, the controller is a primary candidate for DDoS attacks. Although rule-based detection can assist, it lacks flexibility. Machine Learning (ML) provides a more dynamic and smart method of early detection and mitigation of such attacks. Based on the simulation of attacks and traffic patterns experienced in previous experiments, this section discusses how to utilize supervised ML models to determine network traffic as malicious or benign.

DATASET CREATION AND FEATURE ENGINEERING

Data Collection

The test dataset was assembled from attack simulations logs. The traffic was grabbed at the switches using tcpdump and processed through Python scripts as well as such tools as Wireshark and Scapy. Each packet trace was marked either as:

• 0 → Benign

• 1 → DDoS Attack (SYN flood, UDP flood, or ICMP flood)

We developed a balanced dataset of 100,000 samples consisting of 50,000 attack flows and 50,000 normal flows.

Features Extracted

Each record in the dataset includes the following features:

Table 5.

Feature	Description
src_ip	Source IP address (converted to categorical)
dst_ip	Destination IP address (converted to categorical)
src_port	Source port number
dst_port	Destination port number
protocol	Protocol used (TCP, UDP, ICMP)
packet_count	Number of packets in the flow

Feature	Description
avg_packet_size	Mean size of packets
flow_duration	Duration of the flow in milliseconds
inter_arrival_time	Average time between packets
bytes_per_second	Total bytes / duration
flag_count	Count of SYN, ACK, RST flags (for TCP)
label	Binary class (0 = benign, 1 = attack)

Categorical features (src_ip, dst_ip, protocol) were encoded using **one-hot encoding**. The final dataset was normalized using **MinMaxScaler**.

Machine Learning Models Used

We used six widely adopted supervised classification algorithms. The models were implemented in **scikit-learn** and trained on an 80/20 train-test split.

1. **Logistic Regression (LR)**

A linear model ideal for binary classification.

- **Pros**: Simple, interpretable, quick to train.
- **Cons**: Limited in capturing complex non-linear patterns.

Results:

- Accuracy: 89.6%
- Precision: 88.2%
- Recall: 87.5%
- F1 Score: 87.8%

Despite its simplicity, Logistic Regression performed fairly well, indicating that basic traffic features already provide some separation between normal and malicious traffic.

2. **Random Forest (RF)**

An ensemble of decision trees using bagging for robustness.

- **Pros**: High accuracy, reduces overfitting, handles mixed feature types well.
- **Cons**: Less interpretable, slower training on large datasets.

Results:

- Accuracy: 96.4%
- Precision: 95.8%
- Recall: 96.9%
- F1 Score: 96.3%

Random Forest provided a significant boost in performance due to its ability to handle complex, non-linear relationships and feature interactions.

3. **Support Vector Machine (SVM)**

A classifier that constructs hyperplanes in high-dimensional space.

- **Pros**: Effective in high-dimensional spaces.
- **Cons**: Slower training on large datasets, requires careful tuning.

Results:

- Accuracy: 91.2%
- Precision: 90.3%
- Recall: 91.7%
- F1 Score: 91.0%

SVM with a radial basis function (RBF) kernel performed well, especially when tuned using grid search. It provided a strong boundary between classes, especially for ICMP-based attack.

4. **K-Nearest Neighbors (KNN)**

A non-parametric method that classifies based on proximity to training samples.

- **Pros**: Easy to implement, no training phase.
- **Cons**: Computationally expensive at prediction time, sensitive to irrelevant features.

Results:

- Accuracy: 88.1%
- Precision: 87.2%
- Recall: 86.5%
- F1 Score: 86.8%

KNN was fast to prototype and gave decent performance but suffered on larger test sets due to latency in distance computation.

5. **Gradient Boosting (XGBoost)**

A high-performance gradient boosting model optimized for speed and accuracy.

- **Pros**: Top-tier accuracy, handles missing values and non-linearity well.
- **Cons**: Requires careful hyperparameter tuning, higher memory usage.

Results:

- Accuracy: 98.2%
- Precision: 98.0%

- Recall: 98.5%
- F1 Score: 98.2%

XGBoost outperformed all other models. It captured complex interactions between features like inter-arrival time and protocol-specific flags, making it ideal for SDN traffic analysis.

6. **Neural Networks (MLPClassifier)**

A simple feedforward multi-layer perceptron with one hidden layer.

- **Pros**: Flexible, can model non-linear relationships.
- **Cons**: Requires large dataset, hyperparameter tuning, can overfit.

Results:

- Accuracy: 94.7%
- Precision: 94.1%
- Recall: 94.9%
- F1 Score: 94.5%

Neural networks performed very well but slightly below XGBoost and Random Forest. Their performance improved significantly with proper regularization (dropout) and activation functions

Model Comparison

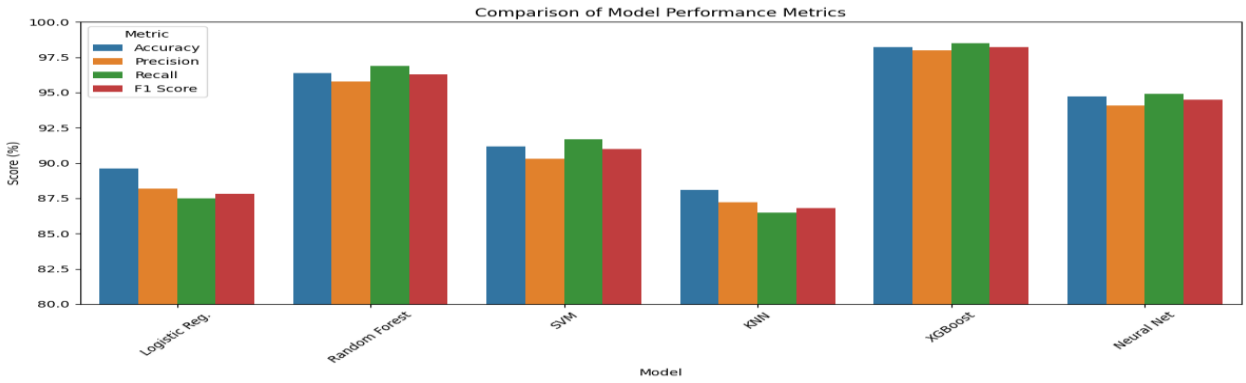

Figure 4.

INSIGHTS AND DISCUSSION

Feature Importance:

- The most significant features across all models were:
- inter_arrival_time
- flag_count

- bytes_per_second
- These features change dramatically under DDoS conditions, making them ideal for classification.

Model Suitability:

- **XGBoost** and **Random Forest** are best for high-accuracy environments.
- **Logistic Regression** is ideal for quick prototyping or low-resource settings.
- **Neural Networks** offer scalability but require GPU for large datasets.

Real-Time Detection:

- Models like Random Forest and XGBoost can be embedded into SDN controllers using APIs or microservices to analyze traffic features in near real-time.
- Integration with Ryu via REST APIs allows triggering of mitigation flows upon classification.

LIMITATIONS

- Models are only as good as the dataset. Synthetic data must closely mimic real-world attack behavior.
- Static models may degrade over time; retraining or online learning is needed for evolving attack patterns.

CONCLUSION AND RECOMMENDATIONS

Based on the performance and generalizability, **XGBoost** and **Random Forest** emerge as the most effective models for detecting DDoS attacks in SDN environments. They provide high accuracy and robustness with relatively low latency in prediction. These models, when combined with lightweight feature extraction scripts and integrated into the SDN control plane, can provide early detection, thus reducing the risk of controller saturation.

FUTURE RECOMMENDATIONS

- Applying **unsupervised** learning to detect zero-day attacks.
- Using **Recurrent Neural Networks (RNNs)** for sequence-based traffic prediction.
- Building an **automated mitigation** loop tied to the classification output.

DECLARATIONS

Acknowledgement: We appreciate the generous support from all the contributor of research and their different affiliations.
Funding: No funding body in the public, private, or nonprofit sectors provided a particular grant for this research.
Availability of data and material: In the approach, the data sources for the variables are stated.
Authors' contributions: Each author participated equally to the creation of this work.
Conflicts of Interests: The authors declare no conflict of interest.
Consent to Participate: Yes

Consent for publication and Ethical approval: Because this study does not include human or animal data, ethical approval is not required for publication. All authors have given their consent.

REFERENCES

Aamir, M., & Ali Zaidi, S. M. (2021). Clustering based semi-supervised machine learning for DDoS attack classification. *Journal of King Saud University - Computer and Information Sciences*, 33, 436–446.

Alubaidan, H., Alzaher, R., AlQhatani, M., & Mohammed, R. (2023). DDoS detection in software-defined network (SDN) using machine learning. *International Journal of Cybernetics and Information*, 12(4).

Ateş, Ç., Özdel, S., & Anarim, E. (2019). Graph-based anomaly detection using fuzzy clustering. In *International Conference on Intelligent and Fuzzy Systems* (pp. 338–345).

Butt, U. A., Amin, R., Mehmood, M., Aldabbas, H., Alharbi, M. T., & Albaqami, N. (2023). Cloud security threats and solutions: A survey. *Wireless Personal Communications, 128*(1), 387–413. https://doi.org/10.1007/s11277-022-09940-0

Dahiya, A., & Gupta, B. B. (2020). Multi attribute auction based incentivized solution against DDoS attacks. *Computers & Security*, 92, 101763.

Dinh, P. T., & Park, M. (2020). ECSD: Enhanced compromised switch detection in an SDN-based cloud through multivariate time-series analysis. *IEEE Access*, 8, 119346–119360.

Fouladi, R. F., Ermis, O., & Anarim, E. (2020). A DDoS attack detection and defense scheme using time-series analysis for SDN. *Journal of Information Security and Applications*, 54, 102587.

Gadallah, W. G., Omar, N. M., & Ibrahim, H. M. (2021). Machine learning-based distributed denial of service attacks detection technique using new features in software-defined networks. *International Journal of Computer Networks and Information Security*, 13(3), 15–27.

Khaliq, A., Adil, S. H., & Jamshid, J. (2018). *Enhancing throughput and load balancing in software-defined networks*. In 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET) (pp. 1–6). IEEE. https://doi.org/10.1109/ICOMET.2018.8346320

Lee, S., Kim, G., & Kim, S. (2011). Sequence-order-independent network profiling for detecting application layer DDoS attacks. *EURASIP Journal on Wireless Communications and Networking*, 2011, 1–9.

MahdaviHezavehi, S., & Rahmani, R. (2020). An anomaly-based framework for mitigating effects of DDoS attacks using a third-party auditor in cloud computing environments. *Cluster Computing*, 23(4), 2609–2627.

Manjunath, C. R., Rathor, K., Kulkarni, N., Patil, P. P., Patil, M. S., & Singh, J. (2022). Cloud-based DDoS attack detection using machine learning architectures: Understanding the potential for scientific applications. *International Journal of Intelligent Systems and Applications in Engineering*, 10(2s), 268–271.

Najafimehr, M., Zarifzadeh, S., & Mostafavi, S. (2022). A hybrid machine learning approach for detecting unprecedented DDoS attacks. *The Journal of Supercomputing*, 1–31.

Pandey, P. (2021). *Security attacks in cloud computing*.

Peng, H., Sun, Z., Zhao, X., Tan, S., & Sun, Z. (2018). A detection method for anomaly flow in software defined network. *IEEE Access*, 6, 27809–27817.

Raj, M. G., & Pani, S. K. (2021). A meta-analytic review of intelligent intrusion detection techniques in cloud computing environment. *International Journal of Advanced Computer Science and Applications*, 12(10), 206–217.

Rawat, S. G., Obaidat, M. S., Pundir, S., Wazid, M., Das, A. K., Singh, D. P., & Hsiao, K. F. (2023). A survey of DDoS attacks detection schemes in SDN environment. In *2023 International Conference on Computer, Information and Telecommunication Systems (CITS)* (pp. 1–6). IEEE.

Ribeiro, M. A., Fonseca, M. S. P., & de Santi, J. (2023). Detecting and mitigating DDoS attacks with moving target defense approach based on automated flow classification in SDN networks. *Computers & Security*, 134, 103462.

Sadeghpour, S., Vlajic, N., Madani, P., & Stevanovic, D. (2021). Unsupervised ML-based detection of malicious web sessions with automated feature selection: Design and real-world validation. In *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)* (pp. 1–9). IEEE.

Suárez-Varela, J., & Barlet-Ros, P. (2018). Flow monitoring in software-defined networks: Finding the accuracy/performance tradeoffs. *Computer Networks*, 135, 289–301.

Valdovinos, I. A., Perez-Diaz, J. A., Choo, K. K. R., & Botero, J. F. (2021). Emerging DDoS attack detection and mitigation strategies in software-defined networks: Taxonomy, challenges and future directions. *Journal of Network and Computer Applications*, 187, 103093.

Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and Information Technologies*, 19(1), 3–26.

Zhou, H., Zheng, Y., Jia, X., & Shu, J. (2023). Collaborative prediction and detection of DDoS attacks in edge computing: A deep learning-based approach with distributed SDN. *Computer Networks*, 225, 109642.