Hybrid Approach for Detection and Mitigation of DDoS Attacks Using Multi-feature Selection, Unsupervised Learning, and Game Theory

Amit Kachavimath and Narayan D.G.

KLE Technological University, Hubballi, Karnataka, India

https://doi.org/10.26636/jtit.2025.4.2261

Abstract — Software-defined networking (SDN) is now widely used in modern network infrastructures, but its centralized control design makes it vulnerable to distributed denial of service (DDoS) attacks targeting the SDN controller. These attacks are capable of disrupting the operation of the network and reducing its availability for genuine users. Existing detection and mitigation methods often suffer from numerous drawbacks, such as high computational costs and frequent false alarms, especially with standard machine learning or basic unsupervised approaches. To address these issues, a new framework is proposed that relies on multistep feature selection methods, including SelectKBest, ANOVA-F, and random forest to select the most important network features, to detect anomalies in an unsupervised manner using agglomerative clustering in order identify suspicious hosts, and to mitigate adverse impacts by relying on posterior probability and game theory. An evaluation conducted using benchmark datasets and validated through Mininet emulation demonstrates that the approach achieves better performance with silhouette scores of 0.86 for InSDN and 0.95 for Mininet. The framework efficiently computes reputation scores to distinguish malicious hosts, thus enabling to rely on adaptive defense against evolving attack patterns while maintaining minimal computational overhead.

Keywords — agglomerative clustering, DDoS attacks, game theory, SDN, unsupervised learning

1. Introduction

The software-defined networking (SDN) concept enhances the process of managing a network by separating the control plane from the data plane and thus enabling centralized data traffic oversight. Such an architecture enables network operators to configure and optimize traffic while using a single control point, simultaneously simplifying policy updates and facilitating the deployment of new services. However, the approach relying on central control comes along with new vulnerabilities. If attackers flood the SDN controller with malicious traffic, with such an attempt known as a distributed denial of service (DDoS) attack, the traffic in the entire network may be slowed down. As more organizations use SDN in cloud and enterprise settings, the detection and mitigation of these attacks is mandatory.

Conventional DDoS detection in SDN often uses fixed rules or simple thresholds to identify malicious traffic. Although these methods are easy to use, they are not flexible enough to handle constantly changing dynamic traffic.

Machine learning and deep learning techniques have shown promise in detecting attacks more accurately, but usually need large amounts of labeled data to train and require significant computing resources [1]. This makes them less practical for real-time implementations in SDNs, especially when attacks take new forms that the model has not experienced before.

Unsupervised learning approaches, such as clustering algorithms, are promising because they do not rely on pre-labeled datasets. These methods are capable of detecting anomalies or suspicious behavior within network traffic by grouping similar patterns, thereby identifying potential attacks. However, the effectiveness of unsupervised detection is critical to the selection of relevant features from network data.

Detection alone is not adequate. Effective mitigation strategies are equally important to ensure the security of SDN environments. Game theory, which analyzes strategic interactions between adversarial entities, offers a valuable framework for improving SDN security. By mathematically modeling the behaviors of both attackers and defenders, game-theoretic methods allow the network controller to dynamically adapt its mitigation policies [2]. This adaptive response improves the controller's ability to manage threats in real time, maintaining an optimal balance between security enforcement and network performance for legitimate users.

The contributions of this research work include the following:

- A combined approach that uses three distinct feature selection methods: SelectKBest, ANOVA F-value and random forest to identify the most relevant network traffic features for effective attack detection.
- Use of agglomerative clustering, i.e. an unsupervised algorithm, to detect anomalous traffic flows that may hint at attacks, even in the absence of labeled attack data.
- Development of a novel game theory-based mitigation process that analyzes traffic flows and applies penalties to suspicious users based on their behavior and puzzle solving speed.

The paper is organized as follows. Section 2 reviews previous work concerning the field in question. Section 3 presents the methodology and modeling details. Section 4 provides an analysis of the experimental findings and Section 5 contains concluding remarks.

2. Related Work

As SDNs become ever more popular, the level of security they offer, especially in defending against DDoS attacks, has become a major area of concern. Legacy techniques, such as statistical analysis and entropy-based models, have been commonly applied for detecting abnormal activities. However, these traditional solutions often face difficulties when it comes to accuracy and do not scale well with large high-speed networks.

In recent years, the use of machine learning and deep learning has become more popular, as these approaches may interpret complex network behaviors and improve detection rates. Research relying on artificial intelligence methods has shown better results in identifying malicious traffic and adapting to changing attack patterns. Despite these advances, some challenges – such as delays in detecting threats, frequent false alarms, and difficulties with managing different types of attacks – still remain. This highlights the need for more adaptable and robust security frameworks for SDN environments.

The authors of [3] propose a clustering framework based on the whale optimization algorithm (WOA-DD) for detecting DDoS attacks in SDNs, using metaheuristic clustering to dynamically group and identify malicious traffic. The methodology separates control and data planes, utilizing programmable SDN switches to analyze network flows and applying WOA-based clustering to detect attack patterns. In [4], the efficiency of an entropy-based technique is identified to detect DDoS attacks on SDN controllers, covering both low- and high-rate attack patterns. The method computes the entropy using the source IP distribution in network traffic, applying a statistical traffic analyzer, and comparing the results with a predefined threshold. The experiments, conducted in a Mininet environment with a POX controller and 64 hosts, evaluate detection rate (DR) and false positive rate (FPR) across eight simulation scenarios, covering both single and multiple attackers targeting one or multiple victims. The results show that the entropy-based approach achieves higher detection rates and lower false positives for high-rate attacks compared to low-rate attacks, with enhancements in DR by 6.25% (to 20.26%) and reductions in FPR by 64.81% (to 77.54%).

The authors of [5] introduced a semi-supervised DDoS detection approach using the K-means clustering algorithm to classify network traffic as normal or malicious. The methodology involves training and testing the model on the CICIDS 2017 dataset, employing hybrid feature selection techniques to optimize input attributes.

Article [6] introduced CAPoW – a context-aware AI-assisted proof-of-work (PoW) framework designed to mitigate DDoS attacks. Using context-aware AI models, such as dynamic attribute-based reputation (DAbR) and temporal activity models (TAM), the framework calculates context scores from deviations in attributes such as IP and temporal data. The CIC-IDS 2017 dataset is used for training and testing, with flow-level features analyzed for adaptive puzzle difficulty. CAPoW achieves a 96% precision level and evaluates performance using such metrics as latency and computational cost, demonstrating its effectiveness in reducing adversarial requests while maintaining network integrity.

In [7], the authors introduce an AI-assisted PoW framework to mitigate DDoS attacks. The methodology uses reputation scores calculated through an AI model trained on the Cisco Talos dataset, which includes malicious IP addresses and attributes. The framework assigns dynamic PoW puzzle difficulties based on reputation scores, integrating Java (Springboot) and Python (Flask) for implementation.

The authors of [8] propose a dynamic game-theoretic framework to mitigate DDoS attacks in SDN-enabled cloud environments. The framework employs the Nash folk theorem to enforce cooperative behavior through reward and punishment mechanisms. Using OpenDaylight controllers, OpenFlow rules, and Snort IDS to detect malicious activity, the solution reduced attack traffic by over 90% during evaluations in Mininet with ICMP, TCP SYN, and UDP flood attacks, while maintaining legitimate traffic throughput.

In [9], a cost-effective shuffling algorithm (CES) is proposed within a moving target defense (MTD) framework to counter DDoS attacks. The methodology utilizes multiobjective Markov decision processes (MOMDP) to model interactions between attackers and defenders, enabling dynamic shuffling decisions. The CES algorithm was implemented using OpenStack and Open vSwitch in an SDN testbed with 50 virtual machines.

The authors of [10] propose an autonomous DoS/DDoS defense system for SDN networks, called GT-HWDS. It combines the Holt-Winters for digital signature (HWDS) and the game-theoretical (GT) approaches. The methodology includes seven-dimensional IP flow analysis for anomaly detection and a game theory-based mitigation module for optimal defense strategies. Using real IP flow data from a large-scale network and the Scorpius simulator for attack scenarios, the system achieves an anomaly detection accuracy of over 98%.

In [11], the authors propose a game-theoretical approach to mitigate DDoS attacks in edge computing environments, focusing on edge DDoS mitigation (EDM). The methodology introduces two approaches: EDMOpti for small-scale optimization using integer programming and EDMGame for large-scale suboptimal solutions employing Nash equilibrium. Evaluations conducted using the EUA data set and simulated edge server scenarios demonstrate the ability to reduce service latency and effectively mitigate over 90% of malicious traffic.

A security enforcement framework aimed at strengthening SDN controllers through game-theoretic defense models against various attack vectors is proposed in [12]. The ap-

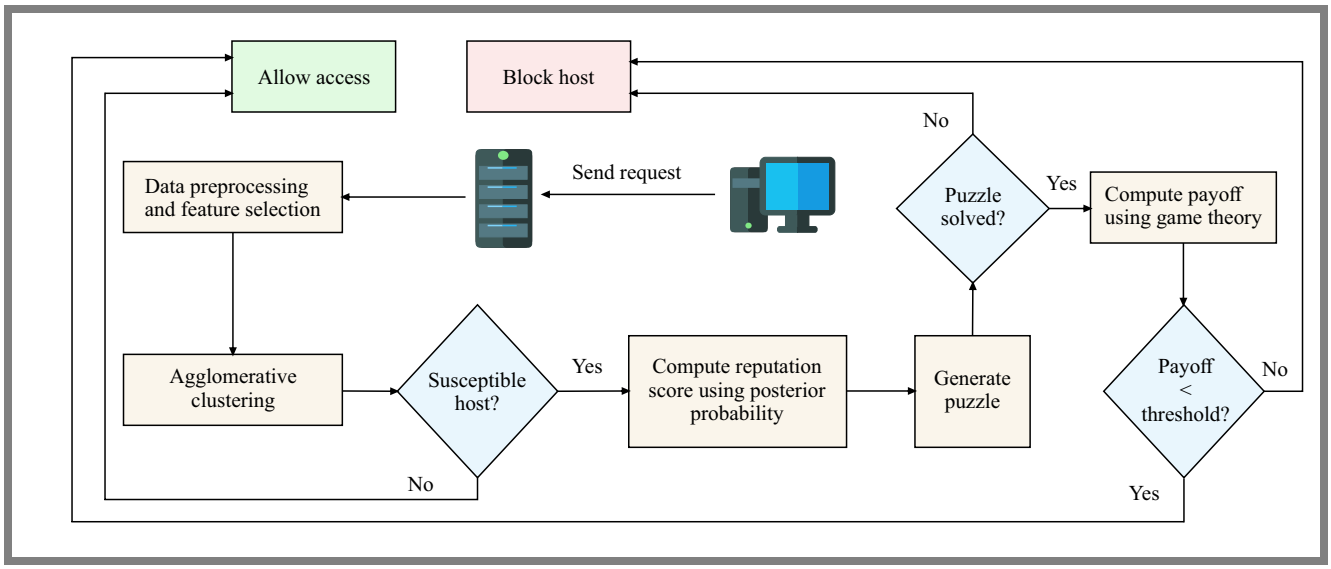

Fig. 1. Architecture of the proposed system.

proach incorporates a trust-based detection scheme that uses signaling game principles for early identification of threats, along with a risk-oriented prevention mechanism to assess and control threat levels in packet flows. To evaluate its effectiveness, the framework models potential threats using the STRIDE methodology and performs experiments in Mininet, achieving a detection accuracy of 98%.

In [13], the authors present a dynamic defense strategy to counter volumetric DDoS attacks by employing client puzzles as a PoW mechanism. The proposed framework adopts a multi-tier approach, incorporating a traffic analysis unit, a puzzle generation component, and a dynamic provisioning module to regulate and suppress attack traffic, with the evaluation conducted using the NS2 simulator.

A zero-sum game-based anti-DDoS firewall framework is proposed to mitigate DDoS attacks in [14]. To optimize mitigation strategies, the framework models interactions between attackers and defenders as a linear programming problem. Two mitigation algorithms are introduced, employing connection limits and sending rates to manage traffic on 80/443 web ports. It is validated using a pay-off matrix, demonstrating probabilistic decision making for effective defense. The study described in [15] proposes a hybrid defense mechanism designed to mitigate distributed reflection denial-of-service attacks within 5G network environments. The framework utilizes a Stackelberg game model to optimize packet sampling rates, combining SDN detection with target-side defenses.

The authors of [16] propose a PoW mechanism integrated with the game theory to mitigate DDoS attacks using computational puzzles with dynamic difficulty levels based on the request frequency of clients. Simulations using 2 000 clients, including 1 000 legitimate users and 1 000 scalpers, demonstrate the system's ability to limit malicious traffic while ensuring that legitimate users are capable of accessing more resources.

In [17], the authors present a game-theoretic framework for identifying and countering low-rate denial-of-service (LR-DoS) attacks. Their approach employs a sigmoid-based filter to distinguish between legitimate and malicious traffic using bandwidth thresholds, while directing potentially harmful flows towards honeypot systems for further analysis. The game theory model evaluates strategies of both attackers and defenders, identifying optimal actions, and achieving Nash equilibrium to enhance security. It uses the NS-3 simulator for traffic generation and Matlab for analysis.

A dynamic Bayesian game-based approach to safeguard software-defined space-air-ground integrated networks (SD-SAGIN) from DDoS attacks is introduced in [18]. This method utilizes support vector machines (SVM) for detecting attacks and leverages Nash equilibrium to dynamically optimize strategies for both attackers and defenders and is validated through simulations conducted in Mininet using a Ryu controller.

All the related works highlight significant gaps in DDoS mitigation strategies for SDN. Traditional methods, such as entropy-based and statistical detection approaches lack the accuracy and scalability needed for modern high-traffic networks. Machine learning and deep learning techniques have shown promise but face challenges, including high false positives, detection latency, and handling heterogeneous attack types. Existing game theory and PoW-based methods improve mitigation, but often involve excessive computational overhead or fail to adapt dynamically to evolving attack scenarios. These limitations emphasize the need for innovative, scalable, and adaptive solutions that balance detection efficiency with resource optimization.

3. Proposed Methodology

The proposed system model, illustrated in Fig. 1, intended for mitigating DDoS attacks in SDN environments, is initiated by

continuously monitoring real-time network traffic and capturing each incoming flow for analysis. During preprocessing, relevant traffic attributes, such as source IP and protocol information, are extracted and normalized. A comprehensive feature selection process is then performed using multiple statistical techniques. SelectKBest, ANOVA-F, and random forest are relied upon to identify the most significant features for downstream analysis. These selected attributes are input to an unsupervised anomaly detection module, where agglomerative clustering is applied to distinguish between normal and suspicious network activities.

If abnormal traffic is detected, the system proceeds to assess the reputation score and dynamically assigns a puzzle difficulty threshold based on posterior probabilities and statistical characteristics of the traffic. A game-theoretic mitigation strategy is used as the model constructs a pay-off matrix and calculates expected utilities to guide response decisions. The final decision module implements mitigation measures, such as blocking or isolating the malicious source, while allowing legitimate traffic to proceed. This adaptive and multilayered approach enhances detection accuracy and ensures efficient mitigation in real-time programmable network environments.

The proposed framework operates in a periodic monitoring cycle to enable continuous detection and mitigation of malicious activities. A one-time feature selection stage is executed at initialization to identify the most relevant flow attributes which are then reused across subsequent intervals. During each monitoring window, the system executes three tasks: unsupervised clustering of captured flows, reputation scoring of active sources, and assignment of proof-of-work puzzle difficulties. The mitigation rules derived from these steps are enforced by the controller in the subsequent cycle. This design avoids redundant recomputation, ensuring that only clustering, scoring, and mitigation are repeated. In our experiments, each cycle required approximately 3 to 5 s to complete, confirming the feasibility of real-time deployment in SDN environments.

3.1. Traffic Capture Using Wireshark

In the proposed framework, real-time network traffic is continuously monitored at the SDN controller using Wireshark. Each monitoring session is configured to capture packets at fixed intervals of 120 s, with the resulting traffic data stored in `pcap` format to maintain detailed records for analysis. After data collection, categorical attributes such as protocol types are converted into unique numerical representations, and all features are standardized to integer or floating-point types for consistency in subsequent analysis.

The set of packets captured during each interval is denoted as $P = \{p_1, p_2, \ldots, p_m\}$, where m represents the total number of packets observed in a single 120-second window. Feature extraction is applied to this set, and the ten most informative attributes are selected, resulting in a feature subset $F_{10} = \{f_1, f_2, \ldots, f_{10}\}$ for downstream anomaly detection. To improve computational efficiency and focus the analysis on potentially abnormal events, the system introduces a packet rate filter for all observed source IP addresses. For each capture interval, the mean packet rate is computed across all unique sources. This is mathematically defined as:

$$\frac{1}{N}\sum_{i=1}^{N} R_i < 100\ , \tag{1}$$

where N denotes the number of distinct source IP addresses present in the interval, and R_i is the packet rate corresponding to the i-th source IP.

If the average packet rate across all sources is less than 100 packets per second (pps), the interval is considered to represent normal traffic and is excluded from clustering and further analysis. This approach ensures that the anomaly detection module processes only intervals with significant activity, which may indicate the onset of a DDoS attack.

The specific selection of 100 pps as a threshold is based on empirical studies and baseline measurements of typical network usage patterns in SDN environments. Under normal conditions, genuine sources usually remain within this rate, whereas DDoS attacks often show sudden increases in the number of packets sent. By setting the threshold at 100 pps, the model effectively filters out benign fluctuations and minimizes false positives, thereby enhancing the detection accuracy and focusing computational resources on suspicious events.

Algorithm 1 Preprocessing network traffic

```
Input: List of pcap files
Output: Numerical feature matrix X
for all file p in pcap files do
    Extract flows: F ← CICFlowMeter(p)
    for all flow f in F do
        Extract features: v ← Features81(f)
        Add v to feature list V
    end for
end for
for all feature c in V do
    if c is categorical then
        c ← LabelEncode(c)
    end if
end for
X ← Convert V to numerical matrix
return X
```

3.2. Data Pre-processing

The data preprocessing phase begins after network traffic is captured using Wireshark in pcap format, initiating a structured workflow to prepare data for unsupervised machine learning detection – see Algorithm 1. The raw pcap files are first processed by CICFlowMeter, which extracts 81 comprehensive flow-based features from each network session, encompassing statistical, temporal, and protocol-specific characteristics of the observed traffic. These features provide a detailed view of typical and potentially abnormal behaviors in the network.

To ensure compatibility with machine learning algorithms that require numerical input, all categorical variables within the feature set are systematically transformed into unique numeric

values using label encoding. This standardization process produces a uniform numerical feature matrix, facilitating effective anomaly detection by unsupervised models.

3.3. Feature Selection

The feature selection process shown as Algorithm 2 is a combination of three well-known methods deloyed to find the most relevant features from data set D, consisting of 81 variables and a target variable y. The aim is to identify ten most important features of F_{top} that contribute the most to distinguishing different types of network traffic.

Algorithm 2 Multi-method feature selection

Input: Dataset D with 81 features, target variable y
Output: Top 10 selected features F_{top}
Initialize $k \leftarrow 20$, $m \leftarrow 10$
$S \leftarrow$ empty list ▷ Store selected features
for all method ∈ {SelectKBest, ANOVA F-value, random forest} **do**
  if method = SelectKBest **then**
    Compute statistical scores for each feature
    Select top k features; add to S
  else if method = ANOVA F-value **then**
    Compute F-value for each feature
    Select top k features; add to S
  else if method = random forest **then**
    Train random forest classifier
    Rank features by importance
    Select top k features; add to S
  end if
end for
Initialize $freq$ as an empty map ▷ Count feature ▷ occurrences
for all feature f in S **do**
  if f in $freq$ **then**
    $freq[f] \leftarrow freq[f] + 1$
  else
    $freq[f] \leftarrow 1$
  end if
end for
Sort features in $freq$ by frequency (descending)
$F_{top} \leftarrow$ Top m features with highest frequency
return F_{top}

First, the algorithm sets $k = 20$, meaning that each method will select its top 20 features, and $m = 10$, i.e. the number of final features to be chosen. An empty list S is used to collect all the features selected by each method. The algorithm evaluates three feature selection techniques in sequence: SelectKBest, ANOVA F-value, and random forest. For SelectKBest, the method performs a statistical test for each feature in D and selects 20 features with the highest test scores, adding them to S. The ANOVA F-value method calculates how well each feature separates the classes in y and also picks the top 20 features.

The random forest method builds a classifier using all features and calculates an importance score for each feature.

Tab. 1. Top 10 selected features from the InSDN dataset.

No.	Name	Description of feature
1	Src IP	Network address from which the data originates
2	Dst IP	Destination address to which the data is directed
3	Protocol	Specifies the type of network protocol, e.g. TCP or UDP
4	Pkt Len Std	Standard deviation of all packet sizes within a flow
5	Flow ID	Unique identifier allocated to each network flow
6	Bwd Pkt Len mean	Average size of packets sent in the reverse direction
7	Pkt Len Var	Measure of how packet sizes vary within a single flow
8	Src port	Port number used by the sender of the packet
9	Bwd Seg Avg size	Average segment size calculated for the backward path
10	Bwd Pkt Std Len	Standard deviation of packet lengths in the backward direction

Again, the 20 highest-ranked features are added to S. Once all three methods have selected their features, the algorithm counts how often each feature appears in the combined list S using a frequency map. Features that appear more frequently across the various methods are considered more consistently important.

The algorithm then sorts all features by their frequency, in descending order, and selects the top m features as F_{top}. This approach ensures that the final selection is not biased toward any single method and that only features considered important by multiple techniques are retained. Then, ten features in F_{top} are used in the subsequent steps of anomaly detection and model development process, helping to improve both detection accuracy and computational efficiency (Tab. 1).

3.4. Unsupervised Learning

The proposed Algorithm 3 describes an unsupervised anomaly detection procedure relying on hierarchical agglomerative clustering and subsequent evaluation of cluster quality using the silhouette score.

The input to the algorithm is a feature set $X = x_1, x_2, \ldots, x_m$, where x_i denotes a data point in the selected feature space. Before clustering, the algorithm enforces two prerequisites: network traffic must be aggregated over a 120-second interval, and the average packet rate per flow must exceed 100 pps. This ensures that only sufficiently active and potentially anomalous traffic segments are subjected to further analysis.

If these conditions are satisfied, agglomerative clustering is applied to X to partition the data into two clusters, corresponding to normal and suspicious traffic. The result of

the clustering assigns label $c_i \in 0, 1$ to each data point x_i, resulting in the cluster assignment set $C = c_1, c_2, \ldots, c_m$.

Equation (2) represents the mean intracluster distance a_i for a data point x_i. It is obtained by computing the average Euclidean distance between x_i and every other member of its own cluster C_{xi}, excluding the point itself:

$$a_i = \frac{1}{|C_{x_i}| - 1} \sum_{x_j \in C_{x_i},\, x_j \neq x_i} d(x_i, x_j) \,, \tag{2}$$

where $|C_{x_i}|$ denotes the total number of points in the cluster C_{x_i} and $d(x_i, x_j)$ represents the Euclidean distance between the points x_i and x_j.

Equation (3) defines the mean distance b_i for x_i. This is determined by locating the cluster C_k such that $C_k \neq C_{x_i}$ and is closest to x_i, then computing the average distances from x_i to every point within C_k:

$$b_i = \min_{C_k \neq C_{x_i}} \frac{1}{|C_k|} \sum_{x_j \in C_k} d(x_i, x_j) \,. \tag{3}$$

The silhouette score s_i for a data point x_i is computed as the difference between the mean distance from x_i to the closest neighboring cluster and the mean distance to points within its own cluster, scaled by the larger of these two quantities:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} \,. \tag{4}$$

This score indicates how similar x_i is to its own cluster relative to the nearest neighboring cluster. The overall silhouette score S for the entire data set as the mean of all individual silhouette scores is:

$$S = \frac{1}{m} \sum_{i=1}^{m} s_i \,. \tag{5}$$

A higher value of S (close to 1) suggests well-defined, separate clusters. Values near zero indicate overlapping or ambiguous clusters, and negative values may suggest misclassified points. The algorithm ultimately outputs both cluster assignments C and the computed silhouette score S, which form the basis for further traffic classification and model validation.

3.5. Computation of Reputation Score

The prior probability that the traffic from source IP IP_i is:

$$P(\text{Normal} \mid IP_i) = \frac{n_0(IP_i)}{n_{IP_i}} \,, \tag{6}$$

where $n_0(IP_i)$ is the count of normal traffic samples and n_{IP_i} is the total number of samples from IP_i within a monitoring window. This value serves as the initial baseline for further traffic classification.

Equation (7) defines the prior probability that the traffic from IP_i is malicious:

$$P(\text{Malicious} \mid IP_i) = \frac{n_1(IP_i)}{n_{IP_i}} \,, \tag{7}$$

where $n_1(IP_i)$ is the number of malicious samples from IP_i. This value helps to estimate the likelihood of attack activity from each source.

Algorithm 3 Agglomerative clustering and silhouette score evaluation

Input: Feature set $X = \{x_1, x_2, \ldots, x_m\}$
Output: Cluster assignments C, silhouette score S
Check clustering conditions:
if traffic interval = 120 s **and** avg. flow packets/s > 100 **then**
  Proceed to clustering
else
  Exit (do not cluster)
end if
Apply agglomerative clustering on X to form two clusters $(n = 2)$
Obtain cluster labels $C = \{c_1, c_2, \ldots, c_m\}$, $c_i \in \{0, 1\}$
for each data point x_i in X **do**
  Calculate intra-cluster distance:

$$a_i = \frac{1}{|C_{x_i}| - 1} \sum_{x_j \in C_{x_i},\, x_j \neq x_i} d(x_i, x_j)$$

  Calculate nearest-cluster distance:

$$b_i = \min_{C_k \neq C_{x_i}} \frac{1}{|C_k|} \sum_{x_j \in C_k} d(x_i, x_j)$$

  Compute silhouette score for x_i:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

end for
Compute overall silhouette score:

$$S = \frac{1}{m} \sum_{i=1}^{m} s_i$$

return cluster labels C, silhouette score S

The mean μ_{Normal} and standard deviation σ_{Normal} for a feature x among all samples labeled normal ($c_i = 0$) for a specific IP are defined as follows. These statistics summarize the typical behavior of normal flows.

$$\mu_{\text{Normal}} = \frac{1}{n_0^{(j)}} \sum_{i:c_i=0} x_i \,, \tag{8}$$

$$\sigma_{\text{Normal}} = \sqrt{\frac{1}{n_0^{(j)}} \sum_{i:c_i=0} (x_i - \mu_{\text{Normal}})^2} \,. \tag{9}$$

The mean value $\mu_{\text{Malicious}}$ for a particular feature x is calculated on all data samples labeled as malicious for a specific source IP.

$$\mu_{\text{Malicious}} = \frac{1}{n_1^{(j)}} \sum i : c_i = 1 x_i \,, \tag{10}$$

where $n_1^{(j)}$ is the total count of malicious samples for IP j.

The standard deviation $\sigma_{\text{Malicious}}$ for the same feature x provides the spread of feature values around the malicious mean $\mu_{\text{Malicious}}$ for samples with $c_i = 1$. This statistic helps capture the variability of the selected feature among flows classified

as attacks for the particular IP.

$$\sigma_{\text{Malicious}} = \sqrt{\frac{1}{n_1^{(j)}} \sum i : c_i = 1 (x_i - \mu_{\text{Malicious}})^2} \,. \tag{11}$$

The probability of observing feature value x given class-specific mean μ and standard deviation σ under a Gaussian distribution is modeled in following manner. This likelihood supports probabilistic classification for each flow.

$$P(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(- \frac{(x-\mu)^2}{2\sigma^2} \right) . \tag{12}$$

The posterior probability that a flow with features x belongs to a specific class, integrating both likelihood and prior probability using Bayes' theorem, is defined as:

$$P(\text{Class} \mid x) = \frac{P(x \mid \mu_{\text{Class}}, \sigma_{\text{Class}}) \cdot P(\text{Class})}{P(x)} \,. \tag{13}$$

The normalization constant $P(x)$ ensuring that the posterior probabilities across all classes sum up to one is:

$$\begin{aligned} P(x) = {} & P(x \mid \mu_{\text{Normal}}, \sigma_{\text{Normal}}) \cdot P(\text{Normal}) \\ & + P(x \mid \mu_{\text{Malicious}}, \sigma_{\text{Malicious}}) \cdot P(\text{Malicious}) \,. \end{aligned} \tag{14}$$

The reputation score R for a flow, scaling the posterior malicious probability to a score between 0 and 10, which influences subsequent mitigation actions, is defined by the following:

$$R = 10 \cdot P(\text{Malicious} \mid x) \,. \tag{15}$$

Equation (16) assigns the puzzle difficulty D according to the calculated malicious probability. Higher risk results in a more challenging proof-of-work puzzle for the source.

$$D = \begin{cases} 5 \text{ (Very high)}, & P(\text{Malicious} \mid x) > 0.8 \\ 4 \text{ (High)}, & 0.6 < P(\text{Malicious} \mid x) \leqslant 0.8 \\ 3 \text{ (Medium)}, & 0.4 < P(\text{Malicious} \mid x) \leqslant 0.6 \\ 2 \text{ (Low)}, & 0.2 < P(\text{Malicious} \mid x) \leqslant 0.4 \\ 1 \text{ (Very low)}, & P(\text{Malicious} \mid x) \leqslant 0.2 \end{cases} \tag{16}$$

3.6. Mitigation using PoW and Game Theory

The construction of the hash input for puzzle generation, concatenating flow attributes, and a nonce value before SHA-256 computation, is formulated as:

$$\begin{aligned} H = {} & \text{SHA-256 (Src IP} \mid \text{Protocol} \mid \text{Flow duration} \\ & \mid R \mid \text{Nonce)} \,. \end{aligned} \tag{17}$$

The condition for puzzle completion is stated in this way. The hash output H must begin with D consecutive zeros, signifying successful proof-of-work.

$$H[1:D] = \text{"0} \ldots \text{0"} \ (D \text{ leading zeros}) \,. \tag{18}$$

The adaptive time threshold for solving the puzzle is modeled by the following formula which integrates puzzle difficulty D, current network load N_{load}, average solving time $T_{\text{normal avg}}$, and reputation score R weighted by coefficients $\alpha, \beta, \gamma, \delta$.

$$T_{\text{threshold}} = \alpha D + \beta N_{\text{load}} + \gamma T_{\text{normal avg}} - \delta R \,. \tag{19}$$

Tab. 2. Description of the notation used.

Symbol	Description
p_i	Individual packet in the captured traffic
F	Set of all flows extracted from traffic
f_j	A single flow in the dataset
$X = \{x_1, x_2, \ldots, x_m\}$	Feature set of m data points (flows) used for clustering
m	Total number of packets observed in the interval
k	Number of features selected per method during feature selection
$F_{10} = \{f_1, f_2, \ldots, f_{10}\}$	Reduced set of ten most informative features selected from the packet stream
a_i	Mean intra-cluster distance for data point x_i
b_i	Mean nearest-cluster distance for data point x_i
s_i	Silhouette score for data point x_i
S	Overall silhouette score for clustering
$P(\text{Normal} \mid IP_i)$	Probability that traffic from source IP_i is normal
$P(\text{Malicious} \mid IP_i)$	Probability that traffic from source IP_i is malicious
R	Reputation score of a source (range 0 – 10)
D	Puzzle difficulty level assigned to a source
H	Hash value used in proof-of-work puzzle generation
N	Total number of unique source IP addresses identified in the interval
R_i	Packet rate for the i-th source IP address, measured in pps
N_{load}	Current network load during puzzle assignment
T_{solve}	Actual puzzle solving time for a source host
$T_{\text{threshold}}$	Maximum allowed puzzle solving time based on system parameters
U_{A1}, U_{A2}	Expected utility values for attacker strategies in the game-theoretic model

The host status is determined in this way. If the puzzle is solved within the threshold time, the host is normal. Otherwise, it is

flagged as malicious.

$$T_{\text{solve}} \leqslant T_{\text{threshold}} \implies \text{Normal Host}\,, \tag{20}$$

$$T_{\text{solve}} > T_{\text{threshold}} \implies \text{Malicious Host}\,. \tag{21}$$

The expected utility for the host, if it attempts to solve the puzzle, considering both correct S_1 and incorrect S_2 system classifications, is computed as:

$$U_{A_1} = P(S_1) \cdot 10 + P(S_2) \cdot (-5)\,. \tag{22}$$

The expected utility for the host, if it chooses not to solve the puzzle, is once again based on the system classification outcomes. These utility functions inform the optimal strategy and support the mitigation decision.

$$U_{A_2} = P(S_1) \cdot 0 + P(S_2) \cdot (-20)\,. \tag{23}$$

Table 2 provides a consolidated overview of the mathematical notation used throughout the study. It describes the symbols used to represent packets, flows, feature sets, clustering measures, and reputation scores, thus ensuring consistency in the formulation of equations.

4. Results and Discussion

Mininet is used to emulate the SDN environment, allowing for the creation of a programmable network topology for realistic testing. To simulate DDoS behavior, Hping3 is deployed to generate legitimate and attack traffic.

The POX controller, an open source SDN controller, manages network flows, applies predefined rules, and processes packets adaptively. Wireshark captures network traffic, offering a detailed view of packet behavior and transmission patterns. CICFlowMeter is then applied to extract flow-level statistics which are later used to select key features for the detection algorithm. The integration of these tools facilitates continuous monitoring, traffic classification, and dynamic response to network anomalies.

4.1. Analysis of Benchmark Datasets

The CICDDoS 2019 dataset, created by the Canadian Institute of Cybersecurity, provides a collection of real and attack traffic across various DDoS types such as TCP, UDP, SYN flood, and HTTP flood, as represented in Tab. 3. Captured over several days in a controlled environment, it includes labeled flow data generated using CICFlowMeter, with more than 80 features extracted per flow. This structure supports the development and evaluation of both supervised and unsupervised detection models. Due to its diversity and scale, the data set is widely used in SDN security research to benchmark anomaly detection techniques [19].

The InSDN dataset was generated within a realistic SDN architecture, featuring a layered topology that separates data, control, and application planes, as shown in Tab. 4. The testbed includes OpenFlow enabled switches, a centralized POX controller, and a range of hosts that act as both normal users and attackers. Traffic traces were captured from multiple points in the network to reflect operational conditions. The data set comprises labeled flow records that distinguish between normal, DDoS, and other intrusion events, offering more than 80 statistical features per flow using CICFlowMeter. Attack scenarios include volumetric floods and controller-targeted exploits, all systematically varied in terms of their intensity and duration [20].

Tab. 3. CICDDoS 2019 dataset summary.

Parameter	Value
Total duration	5 days
Total packets	50 million+
Attack types	12
Features extracted	84 flow-based attributes
Flow extraction tool	CICFlowMeter
Data format	PCAP and CSV
Label availability	Benign / attack
Traffic type	Normal and malicious

Tab. 4. InSDN dataset summary.

Parameter	Value
SDN topology	Layered (3 planes)
Controller	POX (centralized)
Switch type	OpenFlow-enabled
Attack types	DDoS (SYN, UDP, ICMP), others
Total DDoS flows	291 330
Total DDoS packets	27 million
Feature extraction	CICFlowMeter (80+ features)
Label classes	Normal, DDoS, others
Traffic capture	Multi-point, real/attack flows

The CICIoT 2023 data set was collected in a realistic smart IoT lab environment using 105 diverse IoT devices, including cameras, smart plugs, lights, and home automation controllers, as illustrated in Tab. 5.

Data gathering focused on both benign traffic and malicious activity, especially on DDoS attacks which were launched

Tab. 5. CICIoT 2023 dataset summary.

Parameter	Value
Environment	Smart IoT Lab (real devices)
Number of devices	105 IoT devices
Attack types	DDoS attacks
Total DDoS rows	33 million
Traffic format	PCAP (Wireshark), CSV
Feature extraction	Flow-based, 50+ features
Label classes	Normal, DDoS, other attacks
Attack tools	Raspberry Pi Bots, scripts
Traffic capture	Network tap

by grouping multiple Raspberry Pi devices used as attackers. Various types of DDoS attacks, such as UDP flood, TCP flood, SYN flood, ICMP flood, and more advanced threats such as ACK fragmentation and PSH-ACK flood, were executed to target IoT devices. Network traffic was captured using Wireshark in the pcap format via network taps that mirrored all communication flows without impacting normal operations. After collection, the data were processed into CSV files with their features extracted, resulting in a data set that covers 33 different attacks in seven categories and totaling over 33 million DDoS attack rows [21].

4.2. Mininet Setup Environment

The network is emulated using Mininet (as show in Fig. 2) – a platform that is widely adopted for creating and testing programmable network topologies in a controlled environment. The POX controller is used to manage flow rules and dynamically control network behavior throughout the experiment. The attached network represents a tree topology with the POX controller at the root, connected to core switches S1 and S2, which branch out to the edge switches S3, S4, S5 and S6, and finally connect to end hosts, with their designations ranging from H1 to H100.

This hierarchical structure is commonly used in SDN experiments to model scalable, realistic aggregation, and distribution of network traffic. The raw network traffic was initially captured in pcap format using Wireshark to ensure comprehensive recording of all packets.

For the experiments, the range of IP addresses assigned to the end hosts ranged from 10.0.0.1 to 10.0.0.100. Each host within the topology was uniquely identified by its respective IP address, ensuring clear mapping and traceability of network flows during traffic generation and analysis. This configuration enabled a precise evaluation of both normal and attack scenarios across the network. The HPing3 tool was used to generate both normal and attack traffic in the experimental setup.

Normal flows simulated routine host communications, while malicious traffic included various DDoS patterns such as TCP SYN, UDP, and ICMP floods. The intensity and timing were varied to test the detection system under various network conditions. The resulting traffic was used to rigorously evaluate the performance of the proposed detection framework.

Wireshark was used to capture the pcap files. These were then processed by CICFlowMeter to extract detailed flow-level features, resulting in CSV files suitable for machine learning analysis. During the data preprocessing stage, label encoding was applied to convert categorical variables to numerical values, and any missing or duplicate entries were systematically removed to maintain data quality.

To enhance the effectiveness of subsequent anomaly detection, a feature selection process was conducted, narrowing the data set to the ten most relevant attributes. The refined dataset was then used for unsupervised clustering to identify anomalous traffic patterns. The silhouette score is a metric that is widely used for evaluating the quality of clustering in unsupervised

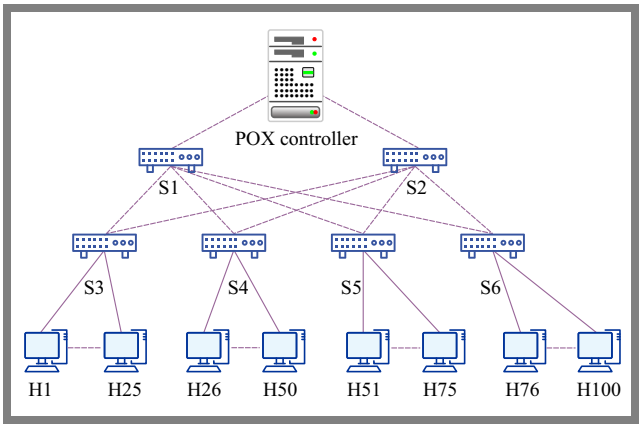

Fig. 2. Mininet topology with 100 hosts.

Tab. 6. Performance evaluation of the silhouette score.

Dataset	Agglomerative	K-means	IF
CICDDoS 2019	0.7084	0.6016	0.5393
InSDN	0.8658	0.3579	0.4203
CICIoT 2023	0.8253	0.2669	0.4114
Mininet	0.9558	0.8023	0.2473

learning. It is computed for each data point as the difference between the mean nearest-cluster distance and the mean intra-cluster distance, divided by the maximum of these two values:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} , \tag{24}$$

where a_i represents the average distance between a point and all other points in its assigned cluster and b_i denotes the lowest average distance to points in any other cluster.

The overall score is the average of s_i across all data points. A silhouette score close to 1 indicates that clusters are well separated and compact, while values near zero suggest overlapping or ambiguous clusters. Negative values indicate potential misclassification. Therefore, higher silhouette scores signify more effective clustering.

Table 6 and Fig. 3 present the silhouette scores obtained for different algorithms, including agglomerative clustering, K-means, and isolation forest with three benchmark datasets and real-time emulation using Mininet. Agglomerative clustering consistently achieves the highest silhouette scores in all datasets, with values of 0.7084 for CICDDoS 2019, 0.8658 for InSDN, 0.8253 for CICIoT 2023, and 0.9558 for the Mininet emulation.

In contrast, K-means and isolation forest yield significantly lower scores, particularly in the case of InSDN and CICIoT 2023 datasets. Consistently high silhouette scores achieved by the proposed approach further validate its effectiveness for real-time DDoS attack detection and traffic analysis.

The Davies-Bouldin (DB) index is a metric that is commonly used to assess clustering performance, as it measures the average similarity between each cluster and its most similar cluster. It is defined as the ratio of scatter to between-cluster separation between clusters for all clusters. Lower DB in-

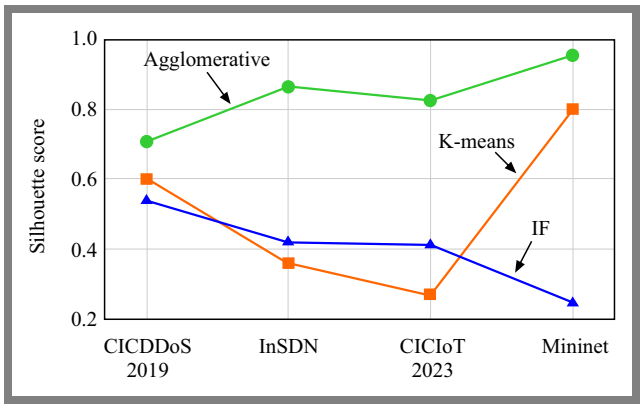

Fig. 3. Performance evaluation using silhouette score.

Tab. 7. Performance evaluation of Davies-Bouldin index scores.

Dataset	Agglomerative	K-means	IF
CICDDoS 2019	0.5139	0.7809	2.9820
InSDN	0.0951	1.4336	2.3158
CICIoT 2023	0.2932	1.8647	3.0906
Mininet	0.0392	0.1766	3.0786

dex values indicate better clustering as they reflect compact clusters with greater separation from each other.

The DB index for a clustering solution with k clusters is calculated as follows:

$$DB = \frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\left(\frac{S_i + S_j}{M_{ij}}\right), \tag{25}$$

where S_i and S_j are the average distances between all points in i and j and their respective centroids, and M_{ij} is the distance between the centroids of clusters i and j.

Table 7 and Fig. 4 present the Davies-Bouldin index scores for agglomerative clustering, K-means, and isolation forest approaches. Agglomerative clustering consistently achieves the lowest values of the DB index, with scores of 0.5139 for CICDDoS 2019, 0.0951 for InSDN, 0.2932 for CICIoT 2023, and 0.0392 for Mininet. In comparison, K-means and isolation forest yield higher DB index values, reflecting less compact and poorly separated clusters, particularly for the IoT and InSDN datasets. The consistently low DB index achieved by agglomerative clustering highlights its effectiveness for DDoS attack detection in SDN and IoT environments.

The Calinski-Harabasz (CH) index, also known as the variance ratio criterion, is a standard metric relied upon for evaluating clustering performance. It is defined as the ratio of between-cluster dispersion to within-cluster dispersion:

$$CH = \frac{\mathrm{Tr}(B_k)}{\mathrm{Tr}(W_k)} \cdot \frac{n-k}{k-1}, \tag{26}$$

where $\mathrm{Tr}(B_k)$ is the trace of the dispersion matrix, $\mathrm{Tr}(W_k)$ is the trace of the dispersion matrix, n is the number of samples and k is the number of clusters.

Higher CH index values indicate better clustering with well-separated and compact groups.

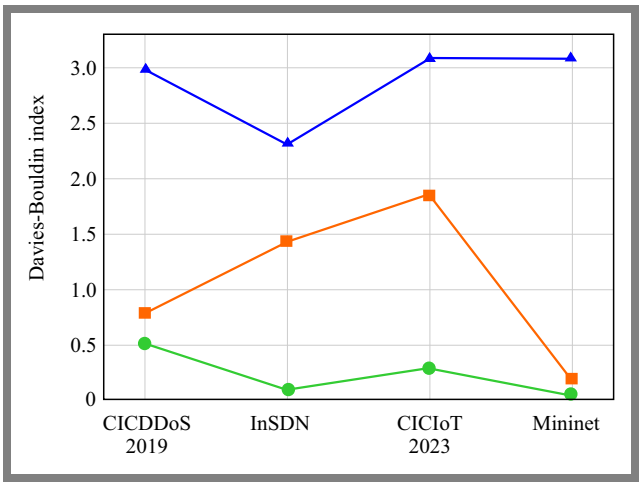

Fig. 4. Performance evaluation using Davies-Bouldin index.

Tab. 8. Performance evaluation of Calinski-Harabasz index scores.

Dataset	Agglomerative	K-means	IF
CICDDoS 2019	9832.68	7402.58	718.94
InSDN	56297.22	3443.00	1074.22
CICIoT 2023	264.94	1816.02	528.59
Mininet	13902.40	84479.42	129.07

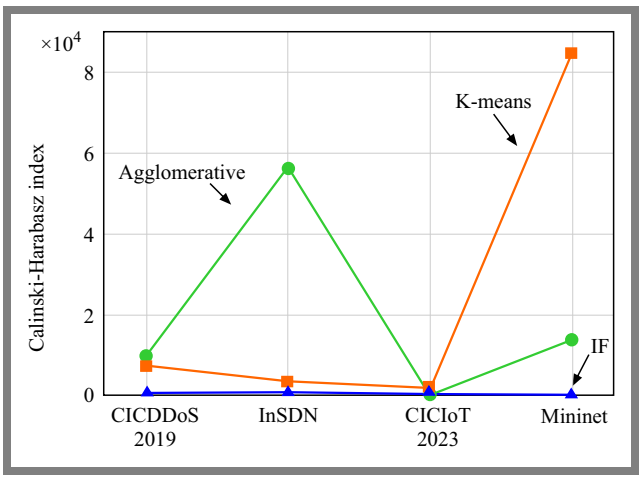

Fig. 5. Performance evaluation using the Calinski-Harabasz index.

Table 8 and Fig. 5 show that agglomerative clustering achieves high CH index scores across most datasets, especially in InSDN and Mininet, suggesting strong cluster separation and compactness.

Figure 6 presents the distribution of the reputation scores assigned during the detection process. The histogram shows that the majority of traffic flows are concentrated around reputation scores between 9 and 10, with the highest frequencies observed being close to 9.5. This pattern indicates that most observed flows represent malicious traffic, reflecting the prevalence of benign or ambiguous traffic in the dataset.

Only a small number of flows receive reputation scores below 9. The spread and shape of the distribution support the effectiveness of the reputation scoring mechanism, proving its

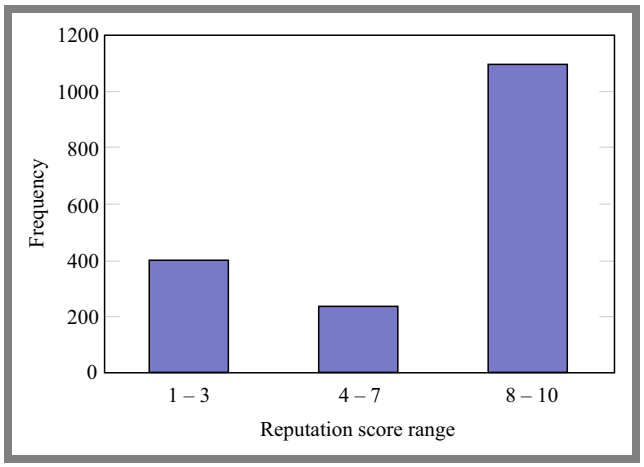

Fig. 6. Distribution of reputation scores (grouped ranges).

Tab. 9. Simulation parameters used to analyze the proposed system.

Parameter	Description	Value
Number of clusters k	Partitioning of dataset into groups	2
Linkage criterion	Rule for merging clusters	Ward
Affinity / Distance metric	Distance computation method	Euclidean
Compute full tree	Construction of complete hierarchy	True
Compute distances	Store distances between merged clusters	True
Silhouette score	Measure of clustering quality	0.95

effectiveness in distinguishing between typical background traffic and potential threats in the SDN environment.

Table 9 summarizes the hyperparameter settings for agglomerative clustering. The configuration with $k = 2$, Ward linkage and Euclidean distance achieved the highest silhouette score of 0.95. These parameter choices ensured effective cluster separation and classification of benign and malicious flows.

The distribution of reputation scores among the three source IP addresses identified during DDoS detection experiments is shown in Fig. 7. The chart provides a visual comparison of how individual IP addresses are assessed by the reputation-based detection framework, which is a central component of the proposed mitigation strategy.

In this analysis, each color-coded bar represents the reputation score assigned to a specific source IP, reflecting its observed behavior within the network during both normal and attack scenarios. The green bar for IP 10.0.0.2 indicates a reputation score of 2.61, which is relatively benign based on its traffic characteristics and the probability of malicious activity derived from model analysis. On the contrary, the red bars for IPs 10.0.0.3 and 10.0.0.5 correspond to substantially higher reputation scores of 9.75 and 9.99, respectively.

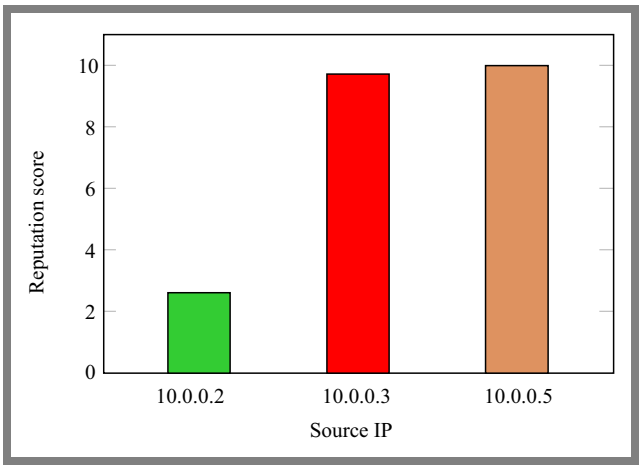

Fig. 7. Source IPs by reputation score.

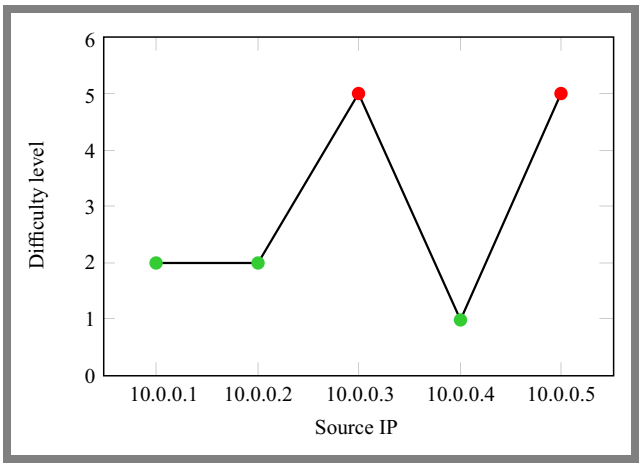

Fig. 8. Difficulty level of source IPs.

The evaluated difficulty levels assigned to the top five source IP addresses observed during the DDoS mitigation experiment in the emulated SDN environment are presented in Fig. 8. Each point on the graph represents a unique source IP, with its corresponding difficulty score, as determined by the game theory PoW mechanism integrated within the mitigation framework.

The plotted line connects the five IP addresses (10.0.0.1, 10.0.0.2, 10.0.0.3, 10.0.0.4, and 10.0.0.5), facilitating visual comparison of difficulty allocation between sources. The difficulty score reflects the level of challenge imposed on each source when attempting to transmit further traffic, with higher values denoting a greater degree of suspicion of malicious intent.

To distinguish between benign and potentially malicious traffic, the data points are color coded: green dots correspond to source IPs 10.0.0.1, 10.0.0.2, and 10.0.0.4, which are classified as benign, while red dots indicate sources that have been assigned high difficulty scores (10.0.0.3, 10.0.0.5) based on their behavior and are identified as malicious. This differentiation is derived from the Bayesian reputation score and the probability of malicious activity, as outlined in the methodology. When the calculated probability that an IP is malicious exceeds a specified threshold, the system responds by increas-

Tab. 10. Analysis of the execution time of proposed methodology.

Stage	Description	Average time [s]
Feature selection (one-time)	Identification of the top 10 features from 80+ attributes	12 – 15
Clustering	Agglomerative clustering of traffic flows	1.5 – 2.0
Reputation scoring	Bayesian probability and reputation computation	0.8 – 1.2
Puzzle assignment and mitigation	Difficulty allocation and enforcement at controller	0.7 – 1.3
Total per monitoring cycle	The periodic cycle includes clustering, reputation scoring, and mitigation, while the one-time feature selection phase is excluded	3 – 5

ing the difficulty level, thus slowing or restricting the actions of these suspicious hosts.

The results discuss patterns about how the detection and mitigation system works in practice. The high values seen in the silhouette and the CH index show that once the system processes and selects key features from the network data, it is able to group normal and attack traffic with clear boundaries. This means that the approach is successful not just with one kind of data, but across different types of network environments, including complex IoT setups and simulated SDN traffic, where network behavior is often unpredictable.

One key insight is that the agglomerative clustering method consistently forms well-separated groups, even in datasets where attacks might be less comparable to normal traffic. This is important because, in real-world networks, attack patterns change constantly, and normal traffic can take many forms. The system's performance suggests that it can keep up with these changes, making it a flexible option for real deployments. In situations where other models, such as K-means or isolation forest, do not distinguish normal and malicious activity, this approach continues to find reliable clusters.

4.3. Processing Time Analysis

The computational efficiency of the proposed methodology was evaluated alongside the clustering performance, measured using the silhouette score represented in Tab. 10.

The feature selection phase, executed once during initialization, required approximately 12 to 15 s when applied to data sets with more than 80 flow-level attributes. From this pool, the proposed methodology consistently retained only 10 highly discriminative features, reducing dimensionality, and improving efficiency. As feature selection is a one-time process, it is excluded from periodic execution, thus eliminating redundant overhead. Each monitoring cycle, comprising clustering, reputation scoring, and PoW puzzle assignment, was completed in 3 to 5 s. The reduced feature set further improves scalability, and future optimization through parallelization or GPU acceleration could enhance performance in larger topologies with heavy traffic.

The authors of [3] used a whale optimization algorithm-based clustering (WOA-DD) for the detection of DDoS in SDN, achieving moderate adaptability but reporting higher false positives, with cluster compactness metrics not exceeding a silhouette score of 0.70 in comparable scenarios. In [4], an entropy-based method is used that achieved detection rate improvements of 6.25 to 20.26% for high-rate attacks, but suffered a notable drop in accuracy for low-rate attacks, with false positive reductions limited to a level between 64.81 and 77.54%. The authors of [5] applied K-means clustering in a semi-supervised setting, which yielded faster classification but lower clustering cohesion, with silhouette scores around 0.60 and DB index values above 0.78. In [6] CAPoW, a context-sensitive AI-assisted PoW framework was developed, reaching a classification accuracy level of 96%, but without unsupervised anomaly detection, which resulted in no silhouette or Davies–Bouldin benchmarks for heterogeneous datasets. The authors of [8] proposed a dynamic game-theoretic defense in SDN that reduced attack traffic by more than 90%, but lacked integrated multi-feature selection and clustering for early detection.

On the contrary, the proposed hybrid approach achieved silhouette scores of 0.8658 for InSDN and 0.9558 for Mininet, with low DB index values of 0.0951 and 0.0392, and high CH index scores exceeding 56 000 for InSDN and 13 900 for Mininet, ensuring clearer separation between attack and legitimate traffic with minimal false positives.

5. Conclusions

The proposed framework integrates multimethod feature selection, unsupervised anomaly detection, and adaptive game-theoretic mitigation to protect against DDoS attacks in SDN environments. Evaluations performed using CICDDoS 2019, InSDN, CICIoT 2023, and Mininet emulation confirmed its effectiveness, with agglomerative clustering achieving low DB index values of 0.0951 for InSDN and 0.0392 for Mininet, together with high CH scores that indicate clear separation between legitimate and malicious traffic.

The adaptive PoW mechanism, guided by posterior probability using reputation scores, ensured that only malicious IP sources obtained a high reputation score of 9.99 with IP 10.0.0.5 that failed the puzzle threshold and triggered targeted defense, while benign hosts, like IP 10.0.0.2 with a score of 2.61, experienced minimal challenge and offered uninterrupted service. This selective, multilayered approach achieved minimal false positives, adapted effectively to mitigate malicious traffic, and demonstrated strong readiness for deployment in practical programmable networks, with the game-theoretic model providing one of the most effective strategies for allocating defense resources efficiently while keeping legitimate network activities unaffected.

References

[1] A.K. Jain, H. Shukla, and D. Goel, "A Comprehensive Survey on DDoS Detection, Mitigation, and Defense Strategies in Software-defined Networks", *Cluster Computing*, vol. 27, pp. 13129–13164, 2024 (https://doi.org/10.1007/s10586-024-04596-z).

[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, art. no. 4441, 2023 (https://doi.org/10.3390/s23094441).

[3] M. Shakil *et al.*, "A Novel Dynamic Framework to Detect DDoS in SDN Using Metaheuristic Clustering", *Transactions on Emerging Telecommunications Technologies*, vol. 33, art. no. e3622, 2022 (https://doi.org/10.1002/ett.3622).

[4] M.A. Aladaileh *et al.*, "Effectiveness of an Entropy-based Approach for Detecting Low- and High-rate DDoS Attacks Against the SDN Controller: Experimental Analysis", *Applied Sciences*, vol. 13, art. no. 775, 2023 (https://doi.org/10.3390/app13020775).

[5] M.N. Jasim and M.T. Gaata, "K-means Clustering-based Semi-supervised for DDoS Attacks Classification", *Bulletin of Electrical Engineering and Informatics*, vol. 11, pp. 3570–3576, 2022 (https://doi.org/10.11591/eei.v11i6.4353).

[6] T. Chakraborty, S. Mitra, and S. Mittal, "CAPoW: Context-aware AI-assisted Proof of Work Based DDoS Defense", *Proc. of the 20th International Conference on Security and Cryptography (SECRYPT)*, vol. 1, pp. 62–72, 2023 (https://doi.org/10.5220/0012069000003555).

[7] T. Chakraborty, S. Mitra, and S. Mittal, and M. Young, "AI_Adaptive_POW: An AI Assisted Proof of Work (POW) Framework for DDoS Defense", *Software Impacts*, vol. 13, art. no. 100335, 2022 (https://doi.org/10.1016/j.simpa.2022.100335).

[8] A. Chowdhary, S. Pisharody, A. Alshamrani, and D. Huang, "Dynamic Game-based Security Framework in SDN-enabled Cloud Networking Environments", *Proc. of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*, pp. 53–58, 2017 (https://doi.org/10.1145/3040992.3040998).

[9] Y. Zhou *et al.*, "Cost-effective Dynamic Shuffling for Mitigating DDoS Attacks Using Moving Target Defense", *Proc. of 6th ACM Workshop on Moving Target Defense (MTD'19)*, pp. 57–66, 2019 (https://doi.org/10.1145/3338468.3356824).

[10] M.V.O. De Assis, A.H. Hamamoto, T. Abrão, and M.L. Proença Jr., "A Game Theoretical Based System Using Holt-winters and Genetic Algorithm With Fuzzy Logic for DoS/DDoS Mitigation on SDN Networks", *IEEE Access*, vol. 5, pp. 9485–9496, 2017 (https://doi.org/10.1109/ACCESS.2017.2702341).

[11] Q. He *et al.*, "A Game-theoretical Approach for Mitigating Edge DDoS Attacks", *IEEE Transactions on Dependable and Secure Computing*, vol. 19, pp. 2333–2348, 2022 (https://doi.org/10.1109/TDSC.2021.3055559).

[12] M. Priyadarsini, P. Bera, S.K. Das, and M.A. Rahman, "A Security Enforcement Framework for SDN Controller Using Game Theoretic Approach", *IEEE Transactions on Dependable and Secure Computing*, vol. 20, pp. 1500–1515, 2023 (https://doi.org/10.1109/TDSC.2022.3158690).

[13] P. Gulihar and B.B. Gupta, "Anomaly-based Mitigation of Volumetric DDoS Attack Using Client Puzzle as Proof-of-Work", *3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, Bangalore, India, 2018 (https://doi.org/10.1109/RTEICT42901.2018.9012127).

[14] E. Okewu, S. Misra, U. Diala, and E.B. Fernandez, "Anti-DDoS Firewall: A Zero-sum Mitigation Game Model for Distributed Denial of Service Attack Using Linear Programming", *4th IEEE International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Tehran, Iran, 2017 (https://doi.org/10.1109/KBEI.2017.8324973).

[15] C. Guo, S. Wang, X. Rong, and X. Tao, "Game-theoretic Modeling of Hybrid Defense Strategies Against DRDoS Traffic in 5G Networks", *IEEE International Conference on Communications (ICC)*, Denver, USA, 2024 (https://doi.org/10.1109/ICC51166.2024.10622381).

[16] K.-Y. Sung and S.-W. Hsiao, "Mitigating DDoS with PoW and Game Theory", *2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, USA, 2019 (https://doi.org/10.1109/BigData47090.2019.9006081).

[17] P. Cotae and R. Rabie, "On a Game Theoretic Approach to Detect the Low-rate Denial of Service Attacks", *2018 International Conference on Communications (COMM)*, Bucharest, Romania, 2019 (https://doi.org/10.1109/ICComm.2018.8429980).

[18] Z. Li, B. Yang, X. Zhang, and C. Guo, "DDoS Defense Method in Software-defined Space-air-ground Network from Dynamic Bayesian Game Perspective", *Security and Communication Networks*, vol. 2022, art. no. 1886516, 2022 (https://doi.org/10.1155/2022/1886516).

[19] I. Sharafaldin, A.H. Lashkari, and A.A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", *Proc. of the International Conference on Information Systems Security and Privacy (ICISSP)*, pp. 108–116, 2019 (https://doi.org/10.5220/0006639801080116).

[20] 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).

[21] E.C.P. Neto *et al.*, "CICIoT2023: A Real-time Dataset and Benchmark for Large-scale Attacks in IoT Environment", *Sensors*, vol. 23, art. no. 5941, 2023 (https://doi.org/10.3390/s23135941).

Amit Kachavimath, M.Tech.
School of Computer Science and Engineering
https://orcid.org/0000-0002-8917-8678
E-mail: amitk@kletech.ac.in
KLE Technological University, Hubballi, Karnataka, India
https://www.kletech.ac.in

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