Cluster Computing (2024) 27:14295–14313
https://doi.org/10.1007/s10586-024-04660-8

A comprehensive detection and mitigation mechanism to protect SD-IoV systems against controller-targeted DDoS attacks

Behaylu Tadele Alemu[1] · Alemu Jorgi Muhammed[2] · Habtamu Molla Belachew[3] · Mulatu Yirga Beyene[3]

Received: 12 March 2024 / Revised: 11 June 2024 / Accepted: 1 July 2024 / Published online: 19 July 2024

Abstract
Software-defined networking (SDN) has emerged as a transformative technology that separates the control plane from the data plane, providing advantages such as flexibility, centralized control, and programmability. This innovation proves particularly beneficial for Internet of Vehicles (IoV) networks, which amalgamate the Internet of Things (IoT) and Vehicular Ad Hoc Network (VANET) to implement Intelligent Transportation Systems (ITS). IoV provides a safe and secured vehicular environment by supporting V2V, V2I, V2S, and V2P. By employing an SDN controller, IoV networks can leverage centralized control and enhanced manageability, leading to the emergence of Software-Defined Internet of Vehicles (SD-IoV) as a promising solution for future communications. However, the SD-IoV networks introduces a potential vulnerability in the form of a single point of failure, particularly susceptible to Distributed Denial of Service (DDoS) attacks. This is because of the centralized nature of SDN and the dynamic nature of IoV. In this context, the SDN controller becomes a prime target for attackers who flood it with massive packet-in messages. To address this security concern, we propose an efficient and lightweight attack detection and mitigation scheme within the SDN controller. The scheme includes a detection module that utilizes entropy and flow rate to identify patterns indicative of attack traffic behavior. Additionally, a mitigation module is designed to minimize the effect of attack traffic on the normal operation, this is performed through analysis of payload lengths.The mitigation flow rule is set for specific traffic type if its payload is less than the threshold value to decrease the false positive rate. An adaptive threshold computation for all parameter values enhances the scheme's effectiveness. We conducted simulations using SUMO, Mininet-WiFi, and Scapy. We evaluated the system performance by using Mininet-wifi SDN simulation tool and Ryu controller for control plane. The system detects DDoS attack traffic within a single window by checking both entropy and flow rate simultaneously. The simulation results demonstrate the efficacy of our proposed scheme in terms of detection time, accuracy, mitigation efficiency, controller load, and link bandwidth consumption, showcasing its superiority compared to existing works in the field.

Keywords Software-defined networking · SD-IoV · DDoS attack · Control plane · Entropy

1 Introduction

This paper is an extension of work originally presented in 2023 International Conference on Information and Communication Technology for Development for Africa (ICT4DA) [1]. The widespread adoption of the IoT and the rapid expansion of wireless communication have significantly improved wireless connectivity in our daily lives. The IoT has been applied in various sectors, such as healthcare for patient and doctor support, smart cities for better urban

✉ Behaylu Tadele Alemu
behaylutadele18@gmail.com

Alemu Jorgi Muhammed
alemu1163@gmail.com

Habtamu Molla Belachew
habtamumullu@gmail.com

Mulatu Yirga Beyene
mulatyir@gmail.com

[1] Department of Computer Science, Debark University, Debark, Ethiopia

[2] Department of Information Technology, Wollo University, KIoT, Kombolcha, Ethiopia

[3] Department of Information Technology, Debark University, Debark, Ethiopia

management, and transportation for ITS. To enhance interaction within the transportation environment, researchers have introduced VANETs and enables communication among vehicles and between vehicles and roadside infrastructures [2]. However, due to the growing demand for ITS and the potential diversity of technologies involved, relying solely on VANETs may not be sufficient. Consequently, the concept of the IoV has emerged, combining IoT and VANET principles to seamlessly integrate the transportation system with other advanced technologies [3].

The IoV, a crucial application within the IoT for ITS, seeks to automate security, safety, and efficiency functions in vehicles while promoting the commercialization of vehicular networks [4]. Nevertheless, it faces obstacles concerning security, privacy, and the autonomous management of vehicles. Fortunately, the rise of SDN has attracted considerable research attention, with its adaptable and programmable characteristics achieved through the separation of the control plane from the data plane [5].

SDN is one of the emerging technologies that decouples the control plane from the forwarding plane [6]. Its architecture typically involves three planes: the application plane, the control plane, and the data plane [7]. The application plane consists of network applications such as driving safety apps, infotainment, and mobility management that leverage the exposed northbound application interface to interact with the control plane [8]. The control plane is the most essential part of the SDN architecture. It controls the networking element at the data plane and provides many network services based on the policies defined on it [3, 9]. The data plane is used as a simpler forwarding element with no software capable of making an instant decision. It treats the packet based on the rules installed on the flow table of the OpenFlow-enabled device, where the flow rules are installed by the control plane [10, 11].

The concept of Software-Defined Internet of Vehicles (SD-IoV) has been introduced to manage vehicle networks using centralized software known as an SDN controller [12]. The SD-IoV combines two innovative technologies, SDN and IoV, to effectively operate in the vehicular environment. By leveraging the global network view provided by SDN controllers, SD-IoV can efficiently manage transmission control, hand-off techniques, and network resource allocation without compromising data integrity [13].

Even though SDN comes with a flexible, central, and programmable feature for IoV networks, the SD-IoV encounters various challenges that require in-depth investigation. Privacy and security are the key challenge due to the centralized feature of SDN and the dynamic nature of IoV. Currently the DDoS attack is rapidly increasing and have a capability to stop the entire network. In the dynamic network environment the elements have high probability to compromise by DDoS attacker and the central controller becomes a clear target for DDoS attackers [14]. The centralized and dynamic aspect of SD-IoV makes it vulnerable to DDoS attacks [10], emphasizing the need for robust detection and mitigation approaches within the logically centralized SD-IoV environment. Due to the dynamic nature of the vehicular environment, the attacker can easily compromise elements within the IoV [11] and flood an attack traffic. These issues motivate us to design a lightweight detection and mitigation system to safeguard the central controller of SD-IoV.

A DDoS attacker may target the controller in an SD-IoV environment by compromising the vehicles and initiating a substantial volume of new traffic towards the OpenFlow switch. As a result, the switch experiences a flood of packet-in messages because it cannot find a corresponding match in its flow table. These packet-in messages are then forwarded to the central controller, overwhelming it and causing congestion in the switch-to-controller link. This congestion has the potential to render the link and the controller unavailable for legitimate packet-in messages, effectively making the DDoS attack focused on flooding the controller with packet-in messages. If the central controller becomes overwhelmed by a flood of packet-in messages, it can cease to function. When the central controller stops working, the entire network becomes paralyzed.

This study's primary contribution lies in designing a DDoS attack detection and mitigation scheme specifically tailored for SD-IoV networks, emphasizing a lightweight and efficient approach. The deployment of the detection and mitigation system directly on the controller necessitates a lightweight solution to avoid adding undue burden to the controller. Therefore, an information theory-based system, as demonstrated in related research [6, 7], is deemed preferable for attacks targeting the controller. Previous studies, including [15–17], have explored DDoS attack detection in SDN using entropy theory with a fixed threshold value. Recognizing the dynamic nature of environments like IoV, introducing a dynamic threshold in attack detection could enhance performance [18]. Solely relying on entropy for detection often resulted in false positives due to entropy's high sensitivity to minor variations in packet distributions. To address these challenges, our approach incorporates flow rate and packet payload analysis alongside entropy values, implementing adaptive threshold computation. This integrated method aims to reduce both false positive and false negative rates, thereby minimizing the load on the controller. The system design involves analyzing network traffic's entropy, flow rate, and payload, with adaptive threshold computation implemented for these parameters to classify traffic as either an attack or non-attack. More specifically, the novelty and contribution of this paper is summarized as follows:

- We propose a novel DDoS attack detection mechanism specifically designed for Software-Defined Internet of

Vehicles (SD-IoV) systems. The mechanism leverages techniques to analyze network traffic patterns and identify anomalies indicative of a DDoS attack targeting the SD-IoV controller. The authors specifically targeted to minimize the latency to detect the attack in a single window by using entropy and flow rate simultaneously instead of using multiple windows.

- Another important contribution is the use of adaptive threshold computation. Unlike previous methods that rely on static thresholds, our system adjusts the thresholds dynamically. This is crucial because static thresholds aren't effective in the ever-changing environment of SD-IoV networks. By adapting the thresholds, our system can better identify attacks in these dynamic conditions.
- The authors also introduce a comprehensive mitigation approach to effectively respond to and defend against controller-targeted DDoS attacks in SD-IoV environments. This includes techniques for traffic filtering, resource provisioning, and load balancing. Previously, these methods relied on a default mitigation technique set by the central controller, often involving blocking entire ports. In contrast, the proposed system analyzes the payload of suspicious packets and creates specific mitigation rules for OpenFlow switches. Instead of blunt port blocking, we selectively drop packets based on their protocol number, focusing on those with small payloads. This targeted approach minimizes the accidental blocking of legitimate traffic sharing the same port as the attack packets.
- The authors evaluate the performance and effectiveness of the proposed detection and mitigation algorithms through extensive simulations to assess the accuracy, detection time, and mitigation capabilities under various DDoS attack scenarios. Furthermore, the authors compare the results with state-of-the-art research to demonstrate the superiority of the proposed approach.

The rest of the paper has the following structure. Section 2 presents the related works. Section 3 discusses the proposed detection and mitigation approach and explains each approaches in detail. Section 4 presents the performance evaluation of the proposed system. Here we discussed on simulation topology, parameters and simulation results with their discussions. Finally, Sect. 5 presents the conclusion of the paper.

2 Related works

The rapid proliferation of the Internet of Things (IoT) and the Internet of Vehicles (IoV) has led to significant advancements in smart environments, yet it has also introduced substantial security challenges, particularly in the realm of Distributed Denial of Service (DDoS) attacks. The integration of Software-Defined Networking (SDN) into IoV (SD-IoV) frameworks offers promising avenues for enhanced network management and security. Researchers have been exploring various sophisticated techniques, including Information Theory, Machine Learning [19, 20], Deep Learning [21], and Fog/Edge Computing [14, 22], to detect and mitigate DDoS attacks in these environments. After conducting a thorough examination of pertinent literature, we identified numerous studies that have employed information theory, specifically focusing on entropy theory. We will now provide an in-depth explanation of how this theory can be applied for the purpose of detecting attacks because we are proposed to detect an attack based on this theory. For instance [23] works on the identification of attacks in SDN-based VANET by using entropy theory. In the study, the authors used packet number as a window size and takes a static threshold value. Mostly the study assured attack existence when the entropy value of 5 consecutive window sizes (i.e., number of packets = 30) is less than the predetermined threshold value [23–26].

In similar way, [27] analyzes the effect of an attack on the SDN environment and proposes an entropy-based method to identify this attack. It uses the flexibility of OpenFlow protocols and OpenFlow controller (i.e., POX) to counteract the attack upon detection. In their study, they used a fixed window size of 50 packets and used a dynamic threshold entropy value. When the entropy value of the current window is greater than the threshold value, this indicates that there is no attack. Whereas the current entropy value lowers than the threshold indicates the possibility of attack.

It is vital to have fast and lightweight detection because attackers could severely disrupt controller services. The study in [15] works on detecting DDoS attack that targets the controller in SDN for preventing controller disruption and data loss. A DDoS attacks which targets to the controller could cripple the entire network. So, we need to always look for ways to detect DDoS attacks against the controller with better accuracy and minimal false positive rate. The study proposes an entropy-based method to identify low-rate and high-rate attacks against the SDN controllers.

Entropy based approach detection technique may vary their performance depending on network topology and number of controllers used. In [17], the authors implements the entropy-based approach in SDN using POX controller to detect DDoS attacks. The study uses various topology and multiple number of POX controllers to evaluate the performance of entropy-based approach. At the end, the paper concludes that entropy-based method performs great and lighter to implement and using multiple controllers makes the system more secure than single controller.

To measure the entropy variation, it is suggested to use destination address rather than source addresses. In

[16, 28], researchers use entropy variation of destination IP address to detect the attack and shows the capability of detecting User Datagram Protocol (UDP) flood attack detection. The entropy variation could improved by including checking the variation of Log Energy Entropy (LEE) in addition with Information Entropy (IE) as proposed in [29]. LEE significantly reduces entropy in an attack scenario compared to IE entropy and can identify an attack early on. Mostly the above related papers have used only entropy theory to detect the attack. Due to this there may have an increment of false positive rate.

The detection accuracy could also be increased by using other parameters in addition to entropy. Furthermore, in [24, 30], the authors proposes an improved DDoS attacks detection system in SDN. Here the study includes flow initiation rate calculation in addition to entropy value calculation. The study is implemented in SDN environment with distributed multiple controller to make the system more secure and reduces the effect of attacks. To ensure the detection of attacks, we typically look for an entropy value below a certain threshold and a flow initiation rate above another threshold over five consecutive windows. However, relying solely on this approach can result in more false negatives and impact the accuracy of detection. To address this limitation, we propose detecting attacks within a single window.

However, in order to reduce the impact of the attack on that network, we must develop an effective mitigation strategy that is used after detection. The most commonly used mitigation technique is blocking of attack traffic incoming port as proposed in [31, 32]. Permanently blocking the port means that even benign traffic attempting to use it will also be denied access, which consequently impacts the accuracy of mitigation efforts. To address this issue, we plan to create a tailored mitigation flow rule targeting specific types of attack traffic instead of implementing a blanket block on the entire port.

The vehicular network needs special concern in the detection and mitigation of attacks because of its real-time and highly dynamic properties. many works are conducted to address security issue of this environment. For instance [33] focus on promptly identifying DoS attacks within the context of connected vehicle systems. They employ adaptive estimation theory to assess the impact of such attacks on the affected region in real-time. The dynamic nature of vehicles increases the likelihood of them being compromised by attackers.

In the event that vehicles are compromised by attackers, their patterns of traffic flow could be altered. This misbehavior could also be detected in different methods such as in [34], introduces a trust-based approach for detecting misbehavior, aiming to identify DDoS attacks within the context of a Software Defined Vehicular Network (SDVN). DDoS attack uses a jamming technique by sending unnecessary signals to degrade the vehicular network's performance. During an attack, the detection of misbehavior can be achieved by assessing the trustworthiness of vehicles. This trustworthiness is determined through the calculation of a trust value, which can be derived from either direct trust evaluations or recommendations (indirect trust). Because the environment is dynamic, a node may trust its malicious neighbour node as a legitimate one. Relying solely on trust value (indirect trust) for attack detection may prove ineffective, as the previous node could potentially misplace trust in the traffic. To overcome this limitation, conducting a comprehensive packet analysis from all nodes is preferable, as it facilitates more accurate identification of attack traffic.

In [26], detection and mitigation against DDoS flooding attacks in SDVN have also been investigated. The method identifies flooding attacks through the analysis of packet flow over time. It then counters the attack by constructing a flow tree, which aids in tracing the origin of spoofed packets. It also shows the detection rate of victim vehicles and mitigation accuracy. The detection of the attack assures that the average flow rule generated by the controller is greater than the threshold consecutively. To mitigate the detected attack, the study builds a flow tree but this takes time and resources for tree building.

Additionally, there have been other studies conducted within the SD-IoV environment, shedding light on the performance of machine learning techniques for the detection of DDoS attacks,in [5], it provides a comprehensive overview of how machine learning methods can be harnessed to effectively identify and counter DDoS attacks in the context of SD-IoV. The study works on the creation of an intelligent method for securing SD-IoV from DDoS attacks by configuring the network periodically and evaluating the vehicle's trust value after shuffling RSU-vehicle associations dynamically. This shuffling makes the system more complex and consumes more resources. Leveraging prior research, we undertook a thorough examination of existing literature. Our goal was to develop a brand-new, lightweight, and highly efficient method for attack detection and mitigation. This innovative technique is specifically designed for deployment on controllers within the SD-IoV environment.

The key concern in SDN is ensuring the security of the central controller against potential malicious attackers. In particular, in SD-IoV, it is crucial for the controller to provide rapid responses to new packet requests in order to maintain the time-sensitive nature of the IoV environment. To this end, it is vital to have fast and lightweight detection as attackers could severely disrupt controller services and filter out a single point of failure. In this context, we put forward an efficient and lightweight detection approach to enhance the security of the controller within the highly dynamic and centralized SD-IoV networks.

3 Proposed approach

3.1 Proposed system architecture

The proposed system architecture have three planes: lower data plane, upper data plane and Control plane. The lower data plane encompasses the vehicles that are with embedded sensors. The legitimate and attack traffics are generated from this plane. The upper data plane consists an OpenFlow enabled roadside unit and base stations. They handle the incoming traffic based on the rules stored in their flow tables. The control plane controls the new traffic by setting rule on the OpenFlow enabled RSU or BS. We are proposed to perform the detection and mitigation on the control plane.

To ensure the security of SD-IoV, we have devised detection and mitigation modules integrated into the control plane of the SD-IoV infrastructure, as illustrated in Fig. 1. We have selected the Ryu controller to manage the control plane and implement these designed modules. Both modules are deployed in the same plane to make the communication between them efficient. The detection module is tasked with recognizing attacks by assessing entropy and flow rate values, derived from the packet-in messages received within a specified window. This module computes entropy, flow rate, average payload, and thresholds for each parameter within each window. Based on these calculated values, the module determines whether an attack is occurring. The mitigation module is activated in response to attack alarms generated by the detection module. Upon the generation of an attack alarm, the mitigation module takes action by installing a mitigation rule on the flow table of OpenFlow-enabled Base Stations (BS) and Roadside Units (RSU). This rule takes into consideration the payload and protocol number of the network traffic. To ensure effective communication between the detection and mitigation modules, they interact through an Application Programming Interface (API).

3.2 Proposed attack detection approach

The focus is on detecting DDoS attacks that target multiple valid or invalid victims, as these attacks aim to overload the controller through an influx of packet-in messages. To identify such attacks, we incorporate additional parameters such as the flow rate of packet-in message and packet payload with the entropy. We retrieve flow rate and payload information from the flow table of the switch, specifically when the destination IP address is valid. Otherwise, we obtain this data based on the established counter settled at the controller side.

When packets reach the controller, they are considered new packets that are forwarded by the switch to request a new flow entry. The controller then installs a flow rule in the switch to handle the remaining packets of the flow without requiring further processing. By recording the destination IP addresses of the new incoming packet-ins, we aim to assess the randomness of these destination IP addresses by calculating the entropy value. This calculation of entropy takes into account a specified window size and involves comparing it with a threshold value for classification purposes. Similarly, it will perform the flow rate and payload. In this proposed work, the calculation technique utilized the adaptive threshold values and involves the application of exponential moving average and exponential moving standard deviation. Let's consider a scenario where a window size is denoted by W, and W as 50 packet-ins, and also consider a hash table to store recorded destination IP addresses along with their respective number of repetitions. At the start of each new window, the hash table is cleared, effectively resetting the counts to zero. From this hash table, we extract X_i representing the destination IP address and Y_i representing the number of repetitions for each entry. In total, the controller's hash table records N packets within each window, which ideally align with the specified window size if it is defined in terms of the number of packets. Then the hash table records in our possession are denoted by H [24].

$$H = (X_1, Y_1), (X_2, Y_2), (X_3, Y_3), \ldots, (X_M, Y_M) \tag{1}$$

The total number of packets, denoted as N, received within the window is determined by summing up the distinct repetition numbers Y_i.

$$N = \sum_{i=1}^{M} Y_i \tag{2}$$

The probability (P_i) of each distinct destination IP address is [7, 24]

$$P_i = \frac{Y_i}{N} \tag{3}$$

Using the calculated probabilities of each distinct destination IP address, we can determine the entropy value for the window. The entropy value is influenced by the probability distribution. The entropy value, denoted as E, can be computed as follows [7, 24]

$$E = -\sum_{i=1}^{M} (P_i * \log_2 P_i) \tag{4}$$

During normal flow conditions, traffic has to be distributed across multiple destinations. However, during a DDoS attack, there is a sudden increase in packets sent to a limited number of destinations or even a single destination. This leads to a significant decrease in the entropy value for that particular window. When the entropy value drops, it

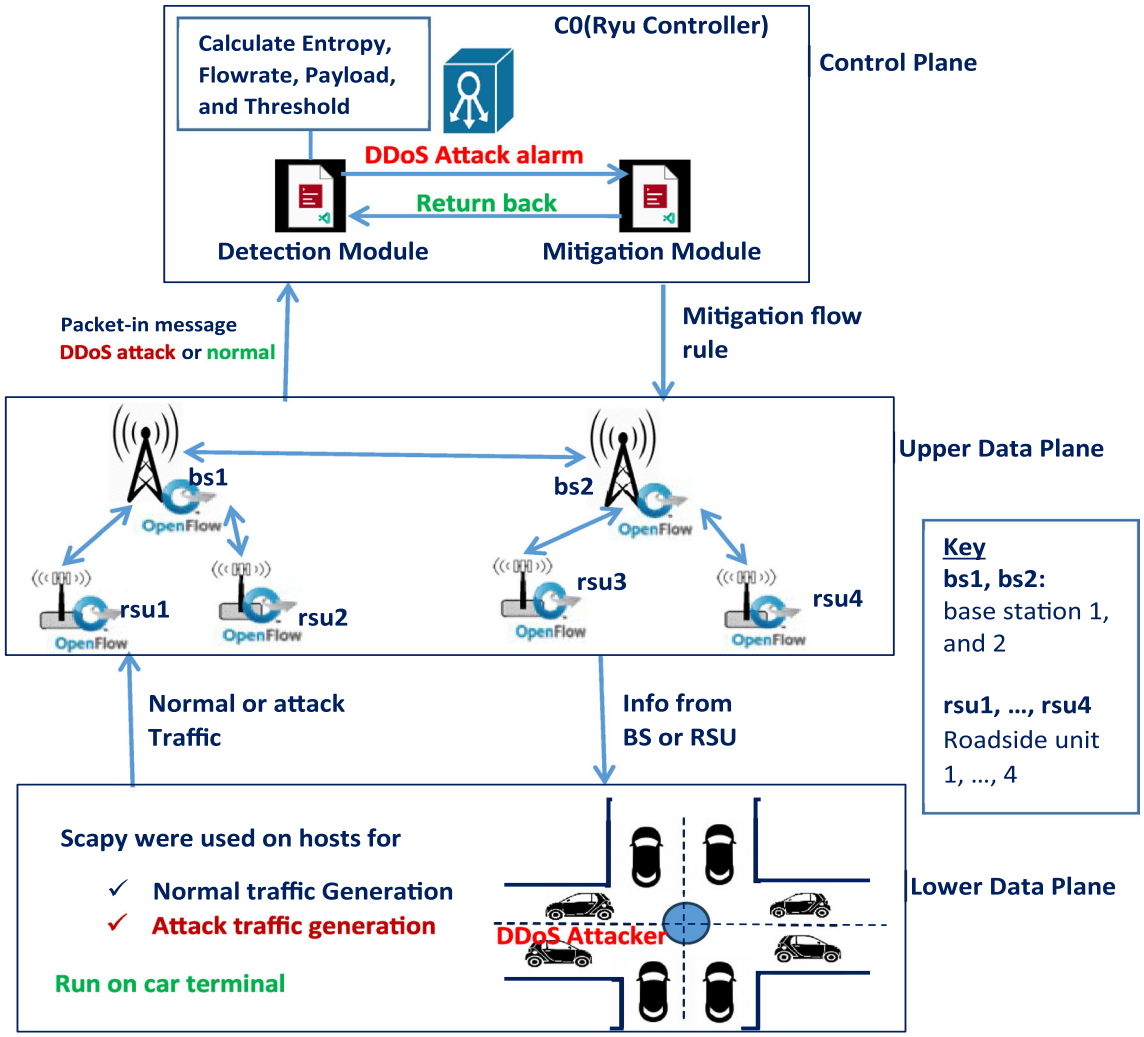

Fig. 1 Proposed system architecture

serves as a warning signal for network monitoring, indicating a potential attack.

The flow rate and average payload could also be calculated in each window. Those parameters are calculated based on the received packet-in messages per window, bytes length of a packet, and Window duration. The controller manages the aforementioned three things by extracting relevant statistical information either from the counterpart of the flow table or from the packet-in messages. Subsequently, we compute the flow rate and packet payload using consecutive mathematical equations based on this statistical data [24].

$$FR = \frac{\text{Total number of packets}}{\text{window duration}} \tag{5}$$

$$PP = \frac{\text{Total transferred Bytes}}{\text{Total number of packet-ins}} \tag{6}$$

3.2.1 Adaptive threshold computation

In the proposed approach, we determined the threshold value through an analysis of normal traffic patterns. However, unlike the static threshold value used in the study mentioned in [24], we employed an adaptive threshold value based on the current characteristics of the packets. This adaptive computation technique is necessary in the dynamic SD-IoV environment to mitigate the potential increase in false positives and false negatives.

Therefore, we used EMA and EMSD to calculate adaptive threshold values. Those terms makes the threshold computation with high focus on the current traffic behavior. By utilizing mean and standard deviation, we can obtain more precise measurements of traffic parameters, leading to more accurate thresholds. These threshold values are determined based on the analysis of normal traffic behavior. In the proposed approach, the detection system examines the network's normal behavior in the absence of any attacks. The

threshold values employed are dynamic and continuously updated to reflect the characteristics of the current traffic. At the end of each window, the adaptive threshold is calculated and compared to the current entropy value.

Consider X_n as the traffic parameter measured in the nth window. The exponential moving average, denoted as $\overline{X}_n$, and exponential moving standard deviation, represented as σ_n, are calculated using the following equations [35, 36]:

$$\overline{X}_n = \alpha * X_n + (1 - \alpha) * \overline{X}_{n-1} \tag{7}$$

$$\sigma_n = \sqrt{\alpha * (X_n - \overline{X}_{n-1}) + (1 - \alpha) * \sigma_{n-1}^2} \tag{8}$$

where $\overline{X}_{n-1}$ represents the previous EMA, σ_{n-1} represents the previous EMSD, and α is a constant whose value can be between 0 and 1. Choosing a smaller value of alpha, such as approaching zero, places greater emphasis on recent data points during calculations. This increased sensitivity to recent changes allows for a more responsive analysis. Conversely, selecting a larger alpha value, closer to one, prioritizes past data points, resulting in smoother and more stable calculations. Taking this into account, we have opted for an alpha value of 1/3. The Adaptive Threshold (AThr), in the nth window can be calculated by [36].

$$AThr_n = \overline{X}_n \pm k * \sigma_n \tag{9}$$

where k is a constant. This is the general formula of threshold calculation using mean and standard deviation. But it may depend on what we will evaluate in the scenario. The selection of k involves a trade-off between false positives and false negatives. Larger values of k lead to higher false negative rates, while smaller values of k result in higher false positive rates. In this paper, the value of k has been set to 1 for deriving the threshold values used in attack detection. In the calculation of Entropy Threshold (EThr), the equation becomes:

$$EThr = \overline{X}_n - k * \sigma_n \tag{10}$$

The sign change to minus is due to the fact that the threshold entropy should be the lowest possible value compared to the average. By comparing the current entropy value with the threshold value, we can determine the likelihood of an attack in that window. If the current entropy value is below the specified threshold value, it indicates the possibility of an attack.

To calculate flow rate threshold, we use the same step of finding exponential moving average (EMA) and exponential moving standard deviation (EMSD) which is used in entropy threshold computation. The difference is that, in flow rate, the possibility of attack is happen when the current flow rate is greater than current Flowrate Threshold (FThr). So, the threshold value should be greater than average values. In this case the equation will become [36]

$$FThr = \overline{X}_n + k * \sigma_n \tag{11}$$

Similarly, packet payload threshold also calculated using EMA and EMSD. The packet payload detects the probability of attack when the current payload is less than the payload threshold because DDoS attack traffics have smaller packet payloads than normal traffics. In this case, the equation of calculating Payload Threshold (PThr), becomes:

$$PThr = \overline{X}_n - k * \sigma_n \tag{12}$$

Algorithm 1 has been designed to calculate adaptive thresholds for parameters, including entropy threshold, flow rate threshold, and payload threshold. These algorithms enable the dynamic determination of optimal threshold values based on the specific characteristics of the data being analyzed. The input values that we have taken in this algorithm is the collected parameter(i.e., entropy, flowrate, or payload) values that is computed for past consecutive windows. So, in this research we are using 10 consecutive window parameter values for corrent adaptive threshold computation (that is based on EMA and EMSD). The EMA and EMSD takes previous data inputs to calculate current average values.

Algorithm 1 Adaptive threshold computation

Input: 10 consecutive parameter values
Output: Timely threshold value
BEGIN Algorithm:
  Calculate simple average (A);
  Calculate standard deviation (SD);
  $AThr = A \pm SD$;
  EMAprev = A; EMSDprev = SD;
  if $window \geq 11$ **then**
    Calculate EMA; Calculate EMSD;
    Calculate AThr = $EMA \pm K * EMSD$;
    EMAprev = EMA; EMSDprev = EMSD;
    Return AThr;
END Algorithm

Algorithm 2 Proposed detection algorithm

```
Input: Packet-in messages
Output: Attack alarm with High Probability DestIP
WindowSize ← w;
PacketCounter ← 0;
Timer ← 0;
Initialize TemporaryHashTable;
LOOP:
while received packet-in message do
    PacketCounter ← PacketCounter + 1;
    if DestIP is in hashTable then
        Increment repetitionNum;
    else
        Add DestIP to hashTable;
        Set repetitionNum to 1;
    end
    if PacketCounter ≥ WindowSize then
        filteredDestIP ← max(repetitionNum);
        calculateEntropy(hashTable);
        calculateFlowrate(packetCounter, Timer);
        calculateEntropyThreshold();
        calculateFlowRateThreshold();
        if entropy ≥ EThr & flowRate ≤ FTh then
            resetParameters(packetCounter, Timer, hashTable);
        else
            if entropy < EThr then
                Increment entropywarn;
                if entropywarn ≥ 3 then
                    Display Attack Alarm and Call mitigation module;
                end
            else
                if flowrate > FThr then
                    Increment flowratewarn;
                    if flowratewarn ≥ 3 then
                        Display Attack Alarm and Call mitigation module;
                    end
                else
                    if entropy < EThr & flowRate > FThr then
                        Display Attack Alarm and Call mitigation module;
                    end
                end
            end
        end
    else
        gotoLOOP
    end
end
```

Furthermore, we have designed an algorithm for the proposed detection system, which includes entropy, flow rate, payload and adaptive threshold computation. This algorithm provides a detailed step-by-step procedure for carrying out the detection process as designed in Algorithm 2.

3.3 Proposed mitigation approach

As the SDN has been a flow-based system, the mitigation module installs a flow entry as a mitigation rule. This rule is installed based on the matches of the incoming traffic and the matches are specified by the mitigation system. The mitigation rule is managed in a similar way to other entries

Algorithm 3 Proposed mitigation algorithm

```
Input: Attack Alarm with high prob DestIP
Output: Mitigation flow rules
BEGIN Algorithm:
 if DestIp is not Valid or then
 |  Block all packets with DestIP;
else
 |  Request OFswitch for Flow stat;
 |   Calculate Average PPL;
 |   if PPL > PThr then
 |   |  Hold DestIP for next check;
 |   else
 |   |  Select protocol number having high flow;
 |   |   Block packets with specified protocol number and DestIP;
Reset Parameter: PCounter=0; Time=0; Hashtable = None;
 Back to detection module;
 END Algorithm
```

in the flow table (eg. it could be managed by idle timeouts and hard timeouts).

The proposed mitigation system utilizes two different techniques to address the distinct characteristics of attack traffic based on the validity of the destination IP address and whether it is within or outside of the controller network:

1. *For valid destination address*: which refers to an address within the network controlled by the SDN controller, packets sent to this address may vary in their protocol or type, such as UDP, TCP, ICMP, and others. To mitigate potential issues, the system employs a specific approach for packets with valid destination addresses. This approach involves selectively dropping packets based on their protocol type while allowing others to continue their transmission.
2. *For invalid destination address*: When the detection system identifies an invalid destination IP address with a higher probability through the filtering process, the mitigation system installs a mitigation rule where all packets with a destination address matching the specified invalid IP address to drop (Fig. 2).

3.4 Joint detection and mitigation approach

This work utilizes the packet-in messages received by the controller as input for the detection module. To calculate the traffic parameters, we have set a static window size, packet counter. As the controller starts receiving packets, the packet counter automatically begins, and the initial time is recorded. Once the window size reaches the specified value, the packet counter stops, and the final time is recorded. Within each window, the controller captures the destination IP addresses of the packets and their repetition numbers, storing this information in a temporary hash table. The hash table consists of two columns: destination IP and repetition number. Based on the cached information in the hash table, the controller calculates the entropy value. Concurrently, the flow rate and packet payload are also calculated at the end of each window. To determine the adaptive threshold values for each parameter, an adaptive threshold calculation module is invoked. This module calculates the timely threshold value and compares it with the current values, ensuring that the thresholds remain updated and relevant. If the calculated entropy value and flow rate fall within the normal range as determined by the threshold evaluation, the system proceeds to set a rule in the switch to manage the packet as normal. Additionally, all relevant parameters (i.e., packet counter, timer, hash table, etc.) are reset to zero in preparation for the next window. However, if the entropy value and flow rate indicate a malicious condition, the system takes action. It filters out the destination IP address with a higher probability and generates an attack alarm. This alarm signifies the potential presence of an attack. At this point, the detection module completes its task by sending an attack alarm, including the destination IP, to the mitigation module. The joint flow diagram for the detection and mitigation system is presented in Fig. 3. The mitigation module receives the attack alarm, including the filtered high-probability destination IP, as input. In this module, the validity of the filtered IP address is checked. If the IP address is determined to be valid or within the controller network, the module proceeds to examine the packet payloads associated with the headers of packets having that IP address.

Upon payload analysis, if the payload size is found to be smaller than the payload threshold, the mitigation module

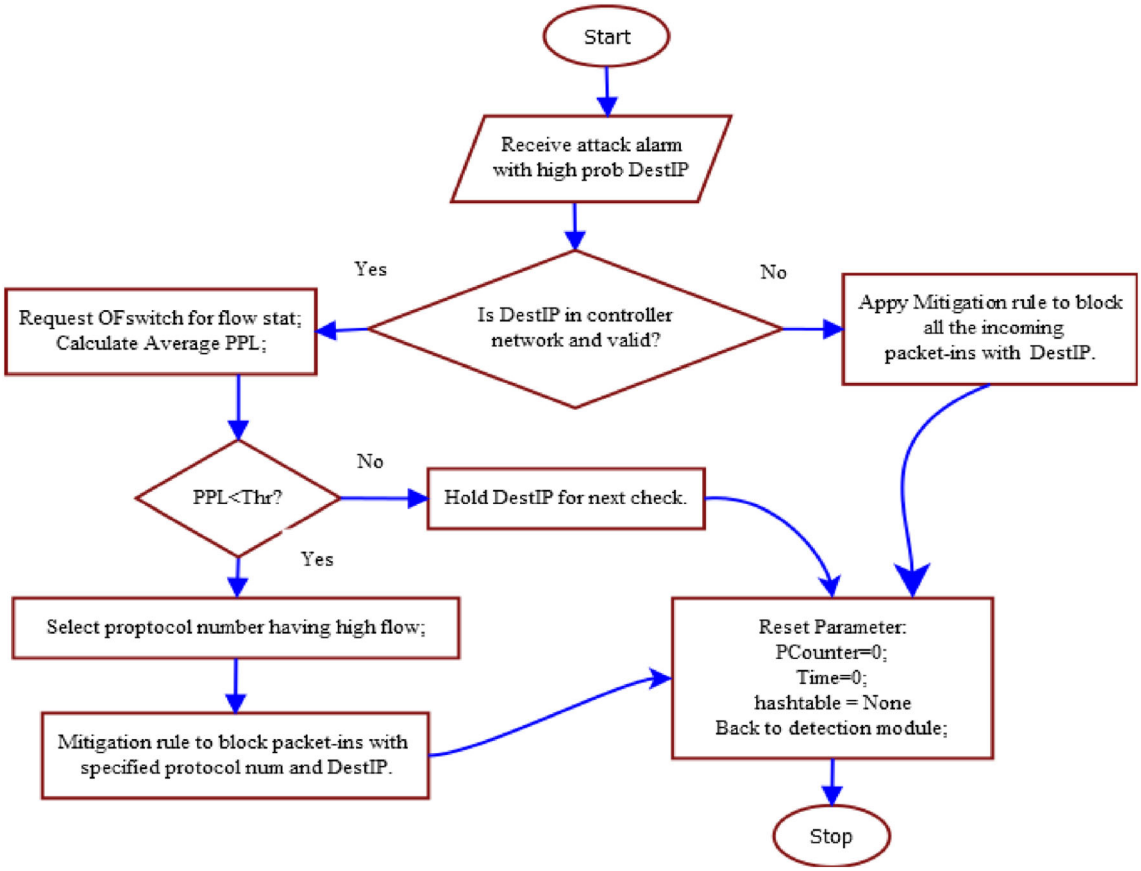

Fig. 2 Mitigation module technique flow diagram

takes action. It selects high-flow protocols and installs a rule to block packets with header information containing the filtered IP address and the selected protocol number. Conversely, if the filtered IP address is determined to be invalid or not within the controller network, the mitigation module installs a rule to block all incoming packets with header information matching that IP address. These mitigation rules are installed on OpenFlow switch by SDN controller. The SDN controller sends the mitigation rule to OpenFlow switch as a packet-out message and the OpenFlow switch stores the rule on its flow table and manages it as other flow entries based on flow management tools (such as timeouts).

4 Performance evaluation

4.1 Simulation topology and parameters

The experiment is performed by using the topology depicted in the Fig. 4 and simulation parameters listed in Table 1. In the topology, we have utilized an OVSSwitch, which is integrated with both the roadside units and base stations. The communication between the remote Ryu controller and the roadside units or base stations is facilitated through the OpenFlow protocol. The OpenFlow protocol offers multiple versions, including ofproto v1.0, ofproto v1.1, ofproto v1.2, ofproto v1.3, ofproto v1.4, and ofproto v1.5. It should be noted that the Ryu controller does not support ofproto v1.1. For the implementation of the proposed system, we used ofproto v1.3, which is fully supported by the Ryu controller. The topology itself has been created using a Python script.

Each vehicle in the simulation is equipped with two wireless interfaces. One interface is used to establish a connection with the roadside unit, while the other interface enables ad hoc communication with neighboring vehicles. The vehicles' movement in the simulation follows a random mobility model. The duration of a vehicle's lifetime in the simulation environment is determined by factors such as trip delays and distances, which are configured in the.-sumocfg file used by SUMO. The integration of SUMO and Mininet-Wifi allows for seamless coordination, with Mininet-Wifi triggering the vehicle mobility based on the information provided by SUMO. The simulation area focuses on approximately one-quarter of Kombolcha city's road network, which is replicated and simulated using SUMO.

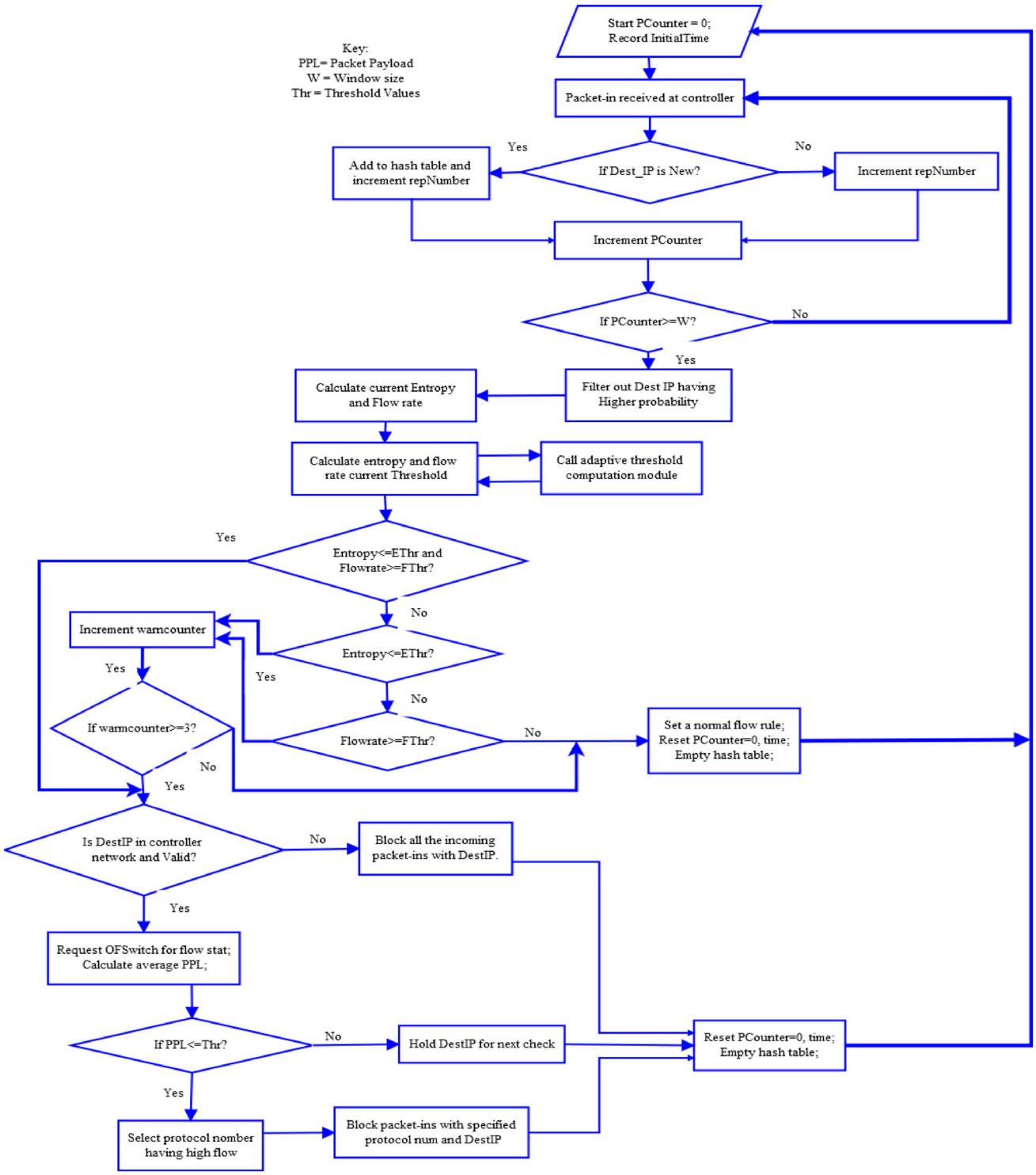

Fig. 3 Joint detection and mitigation techniques flow diagram

As shown in simulation parameters lists in Table 1, attack traffic payload is zero. The reason for why we make the payload of attack traffic to be zero is that to simulate the attack traffic behavior. Mostly, the attack traffics have smaller payload or null payload. So, we make it zero byte. A single targeted attack traffic has a noticeable impact on the entropy value of each traffic flow, whereas a multi-target traffic has a comparatively smaller effect on the entropy value.

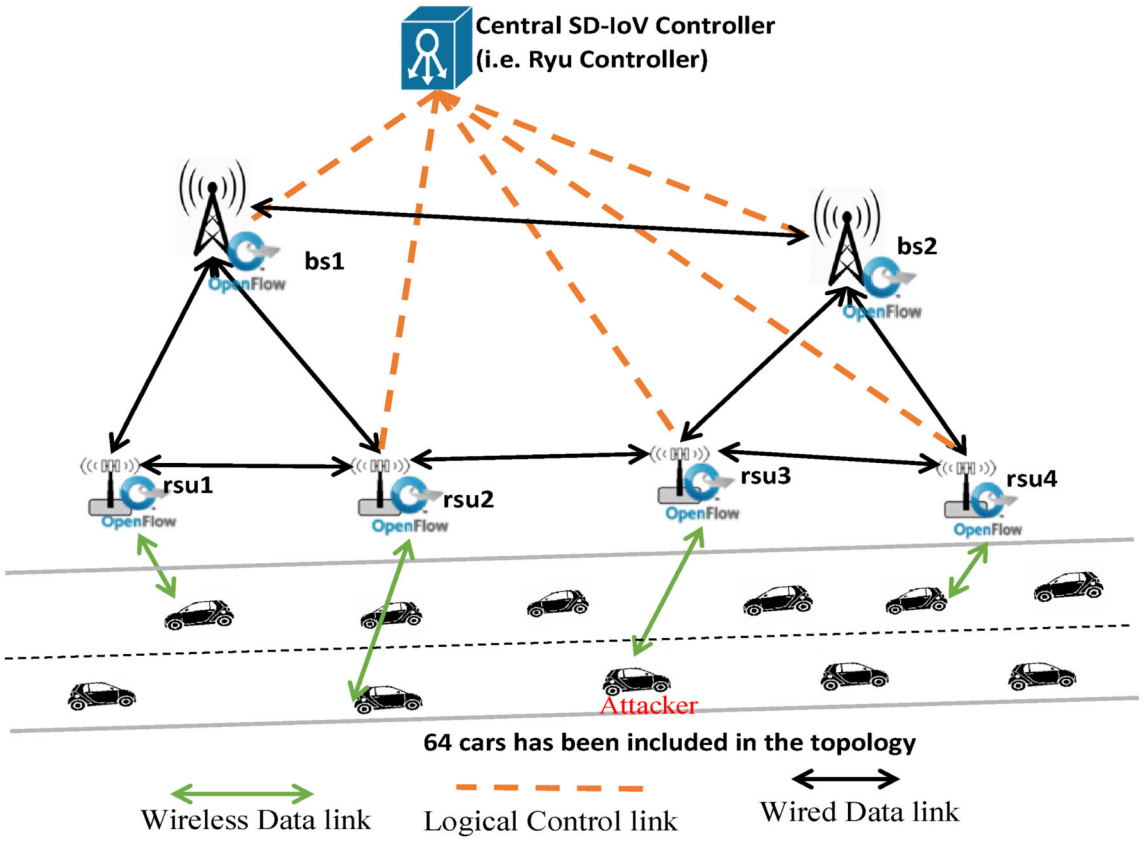

Fig. 4 Simulation topology

4.2 Simulation results and discussion

We conducted the simulation with a window size of 50 packets, which resulted in a maximum entropy value of 5.64385. This maximum value was obtained when the probability of each destination IP address was 1/50. Conversely, the minimum entropy value of 0 was achieved when all packets were directed to the same address, resulting in a probability of 1.

4.2.1 Entropy and flow rate variation

To observe variations in the entropy value and flow rate of the traffic during normal and attack periods, we generated normal packets at intervals of 0.2 s for a duration of 180 s. For the attack packets, we generated them at intervals of 0.08 s for a duration of 60 s. The attack traffic began approximately 60 s after the start of the normal traffic.

In Fig. 5, we observe that the values of entropy and flow rate exhibit variations starting from the sixth window. This indicates the initiation of the attack, which continues until approximately the twenty-eighth window. During the attack period, the flow rate increases significantly as shown in Fig. 5b, while the entropy decreases compared to the normal traffic as shown in Fig. 5a. In the single-target attack scenario, there is a substantial difference in the entropy value between the attack and normal traffic. However, in the multi-target attack scenario, the difference is relatively small. The flow rate difference remains consistent in both scenarios because the flow rate depends on the traffic interval not on the distribution of traffics. When we varied the traffic interval while generation, we get varied flow rate value. The singe-target and multi-target traffics were only varies the distribution of traffics and these are only varies the entropy value. Due to that reason the flow rate variation of single target and multi-target attack is almost the same as shown in Fig. 5. Based on these observations, we conclude that an entropy-based detection method is most suitable for single-target attacks. To efficiently detect both single-target and multi-target attacks, we utilize both entropy and flow rate together.

The effectiveness of the proposed detection and mitigation algorithm has been evaluated using a range of evaluation metrics. These metrics offer valuable insights into the system's overall performance. The evaluation metrics utilized to evaluate detection effectiveness encompass detection time and accuracy, while the evaluation of mitigation efficiency involves examining controller load and switch-to-controller link bandwidth consumption.

Table 1 Simulation parameters

Parameters	Values
Emulator tool	Mininet-WiFi
Traffic generation tool	Scapy
Vehicles mobility simulator	SUMO version cbfefb484
Simulation area	From OpenStreetMap
SUMO delay	200 millisecond
Controller	Ryu controller
OF Protocol version	ofproto v1.3
No of controller	1
No of base station	2
No of roadside units	4
No of vehicles	64 cars
Mobility model	RandomWayPoint
Propagation model	LogDistance with exponent 2.8
Normal traffic duration	180 s
Attack traffic duration	60 s
Normal traffic interval	0.2 s
Attack traffic interval	0.08 s
Packet type	TCP, UDP, and ICMP
Normal packet payload	18 bytes
Attack packet payload	0 byte
Window size	50 packet-ins

4.2.2 Detection time

The detection time can be evaluated by observing the number of windows in which the detection alert is triggered. A smaller number of windows indicating the detection alert suggests a shorter detection time, indicating a more efficient detection process. In Fig. 6, we compare the detection time of the proposed system with works presented in [24] by utilizing entropy, flow rate, and both entropy and flow rate at a time. When utilizing both entropy and flow rate simultaneously, the proposed system achieves a significantly improved detection time, triggering the alert at the first window of the attack.

Previous works relied on a static threshold, triggering an alert when the entropy value fell below a predetermined level or when the flow rate exceeded the threshold within five consecutive windows. These methods used entropy and flow rate separately, resulting in a longer detection time. In contrast, our proposed approach employs a dynamic threshold computation, utilizing both entropy and flow rate simultaneously to minimize detection time to a single window. We compare our proposed method with previous work by evaluating scenarios using entropy only, flow rate only, and both metrics simultaneously.We also detect attacks using entropy and flow rate separately. Here, an alert is triggered when the entropy is below the current threshold or the flow rate exceeds the current threshold for three consecutive windows. However, this approach is effective only when both entropy and flow rate conditions are not met in the same window, aiming to reduce the false negative ratio. In previous works, the alert is triggered when conditions are met in five consecutive windows across all scenarios (entropy only, flow rate only, and both).

4.2.3 Detection accuracy

Detection accuracy is a measure of how much of an attack has been detected as a true positive from all of the attacks generated. In other words, it is the exactness with which a proposed detection system is able to detect malicious traffic. We evaluate the approach by generating an attack for a small period of time frequently. The attack has been generated for 12 s with 0.08 intervals and run this script 8 times. The normal traffic is generated continuously for 360 s with 0.2 intervals. Here we aim to evaluate the detection accuracy by checking for how many times the attack alarm is displayed. Detection accuracy depends on the values of true positive, true negative, false positive, and false negatives.

During the simulation, we observed a total of 63 windows, with 23 windows free from attacks and the remaining 40 windows experiencing attacks. In previous works, the attack alarm is triggered when the current entropy is lower than the threshold for 5 consecutive windows or when the current flow rate exceeds the threshold for 5 consecutive windows. However, in the case of a DDoS attack that occurs over a short period, the parameter values may not vary significantly across multiple consecutive windows. To address this, the proposed system incorporates both entropy and flow rate parameters simultaneously in a single window for detecting attacks. This approach is suitable for detecting attacks of both short and long duration. The attack alarm is activated when the current entropy falls below the threshold and the current flow rate surpasses the threshold.

Our proposed system detects attacks in single windows, leading to more alarms than previous methods as depicted in Fig. 7. We analyzed 63 windows, 40 of which contained attack traffic. We focuses on the number of attack alarms generated to assess the detection accuracy. Previous methods only triggered an alarm if the entropy is less than the threshold or the flowrate is greater than the threshold in five consecutive windows. This leads a false negative conclusion on attack traffic and generates small number of attack alarms. Because the attack that send the traffic in less than five consecutive windows didn't detect by the previous works. But the proposed system detects the attack

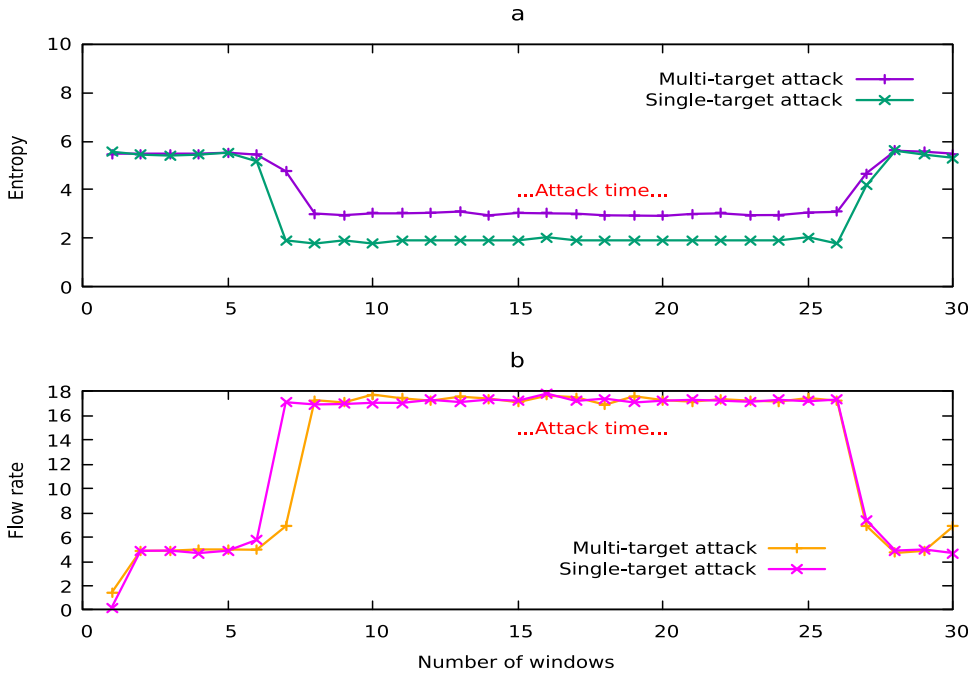

Fig. 5 Entropy and flow rate variation

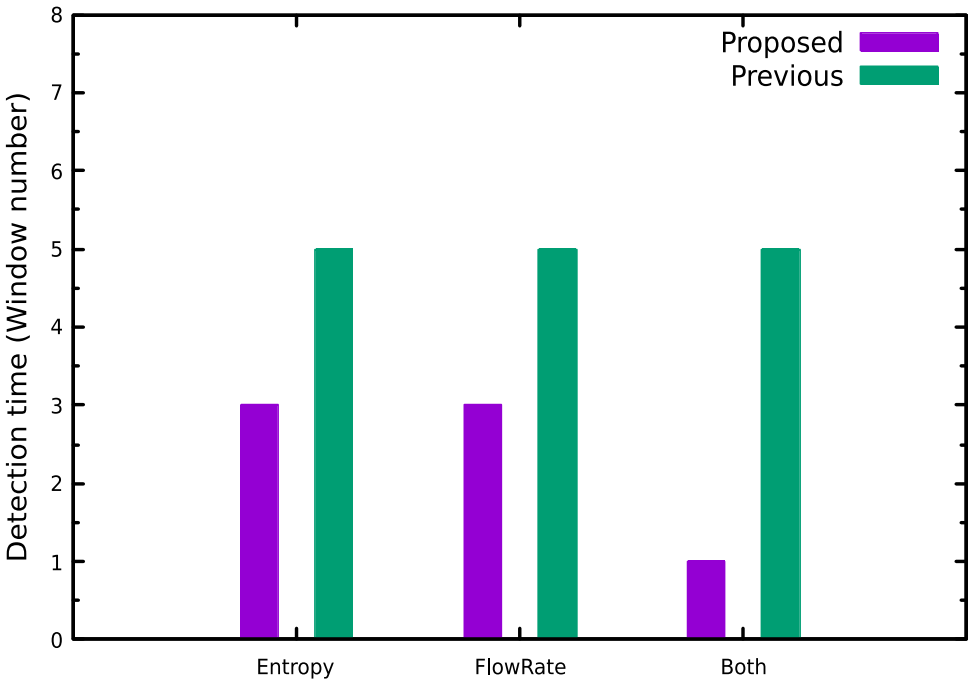

Fig. 6 Detection time

with in a single window by using both entropy and flow rate at a time. Due to this the proposed system generates larger attack alarm than the previous.

In Table 2, the simulation results are listed, and Fig. 7 represents the number of attack alarms displayed in the proposed and baseline paper.

To assess the effectiveness of the proposed system's mitigation capabilities, we employs two different approaches to mitigate attacks: the first approach involves blocking all packets associated with the destination IP address identified during the detection phase, while the second approach entails blocking packets with both the specified destination IP and protocol number having a higher flow along with smaller packet payloads. The first approach is applied to traffic directed outside of the controller's network or to invalid addresses, whereas the second approach is implemented for traffic destined for the

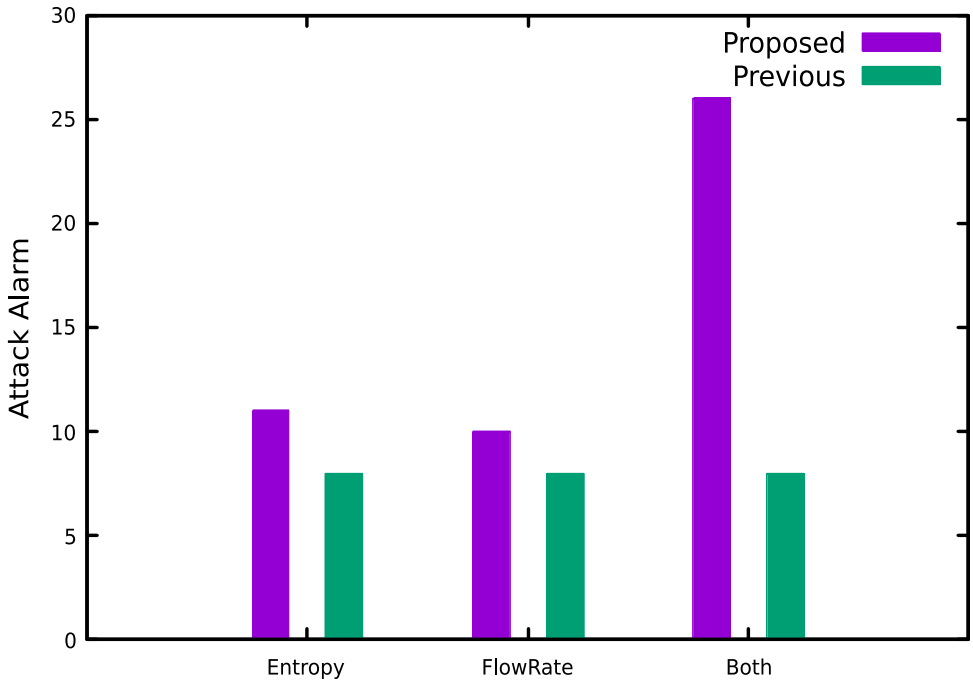

Fig. 7 Number of attack alarms to check detection accuracy

Table 2 Detection accuracy

Total windows = 63, Attack free = 23, With attack = 40												
	Entropy only				Flow rate only				Both entropy and flow rate			
	TP	TN	FP	FN	TP	TN	FP	FN	TP	TN	FP	FN
Previous in [24]	8	23	0	32	8	23	0	32	8	23	0	32
Proposed	11	23	0	29	10	23	0	30	26	23	0	14

controller's network or valid addresses. In the second approach the mitigation module incorporates packet payload as an additional parameter alongside entropy and flow rate.

The mitigation module can be invoked in three distinct conditions after an attack alarm is triggered by a detection module. These ways are:

1. *Warning of entropy*: Here the mitigation module is invoked when the entropy value remains below its threshold for three consecutive windows.
2. *Warning of flow rate*: Here the mitigation module is activated when the flow rate exceeds its threshold for three consecutive windows. And
3. *Both entropy and flow rate*: In this particular conditions, the mitigation module is invoked when both the entropy value falls below its threshold and the flow rate exceeds its threshold within the same single window.

We assess the efficiency of the mitigation by measuring the controller load and OFSwitch-to-controller link bandwidth consumption during both attack and non-attack periods. The evaluation encompasses four scenarios: a single target within the controller network, a single target outside the controller network, multiple targets within the controller network, and multiple targets outside the controller network

4.2.4 Controller load evaluation

The packet-in messages that the controller received in a window were closely related to the controller load. The controller becomes overloaded if it receives a large number of packet-in messages. An attacker sends massive packet-ins during the attack period to overload the controller. By measuring the amount of packet-in messages the controller receives during normal hours and during an attack, it is possible to evaluate how well the proposed methods may reduce the burden on the controller.

In the proposed system, once a detection occurs, a mitigation module is triggered to reduce the controller load by applying specific mitigation rules on an OpenFlow switch. As the mitigation rule is applied to a specific protocol at a time, it takes longer for the controller load to recover to its normal state in cases where the attack traffic is targeted towards the controller network. However, for attack traffic targeting destinations outside the controller

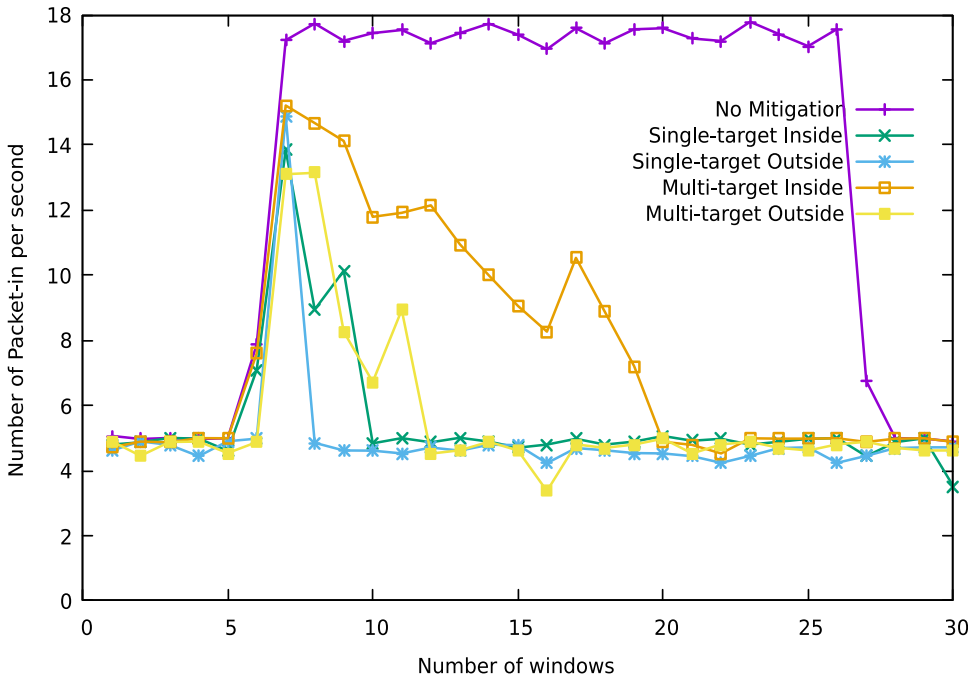

Fig. 8 Controller load evaluation

network, mitigation is carried out based on the destination IP address without specific protocol number selection. This enables a quicker recovery of the controller load to its normal state. The generation of attack traffic initiates in the sixth window, resulting in a rapid increase in the number of packet-in messages received by the controller. This attack traffic generation persists until the twenty-seventh window. As a consequence, the controller becomes overloaded, raising the probability of dropping legitimate packet-in messages. However, after implementing the mitigation measures as illustrated in Fig. 8, the number of packet-in messages received by the controller decreases. This shows that the controller load returns to its normal level, indicating a reduction in the overall burden on the controller. The evaluation involved four scenarios, covering both attack targets within the controller (multi and single) and targets outside the controller (multi and single) network.

4.2.5 OFSwitch-to-controller link bandwidth consumption evaluation

The connection between the controller and OpenFlow switches, known as the core link, is a critical component that can become a single point of failure if redundancy measures are not in place. Ensuring the accessibility of this link is crucial. In situations where attackers generate a large number of packet-in messages, the link may become congested, leading to potential issues such as loss, drop, and delay for legitimate traffic.

We are evaluating how the proposed mitigation system effectively manages the bandwidth consumption of packet-in messages. When a significant amount of data is transferred through the link, exceeding the normal range, it can lead to congestion and potentially indicate an attack. In such cases, the mitigation system intervenes to mitigate the message, restoring the consumption to its normal level. To evaluate the performance of the proposed mitigation system in minimizing link congestion, we conducted experiments by generating normal traffic for a duration of 3 min (180 s). Within this time frame, we introduced an attack traffic period of 1 min (60 s). We measured the amount of data transferred during both normal and attack times and observed the reduction in transferred bytes after applying the mitigation rule during the attack period. This reduction in transferred bytes indicates that the mitigation system effectively reduces the probability of congestion occurrences.

In Fig. 9, we evaluated the performance of the mitigation system using different scenarios involving targets within and outside the controller's network, as well as single-target and multi-target attacks. The attack traffic was generated starting from around the fifty-fifth second and continued until approximately the one hundred fifteenth second. During this period, the bandwidth consumption increased up to 1900 Bytes. However, upon applying the mitigation rule, the bandwidth consumption decreased.

Multi-target attacks targeting hosts within the controller's network require multiple mitigation rules to filter out the attack traffic mixed with normal traffic. This filtration process is based on the protocol number with high flow. The adaptive threshold computation also affects the mitigation of multi-target attack traffic. In some cases, if

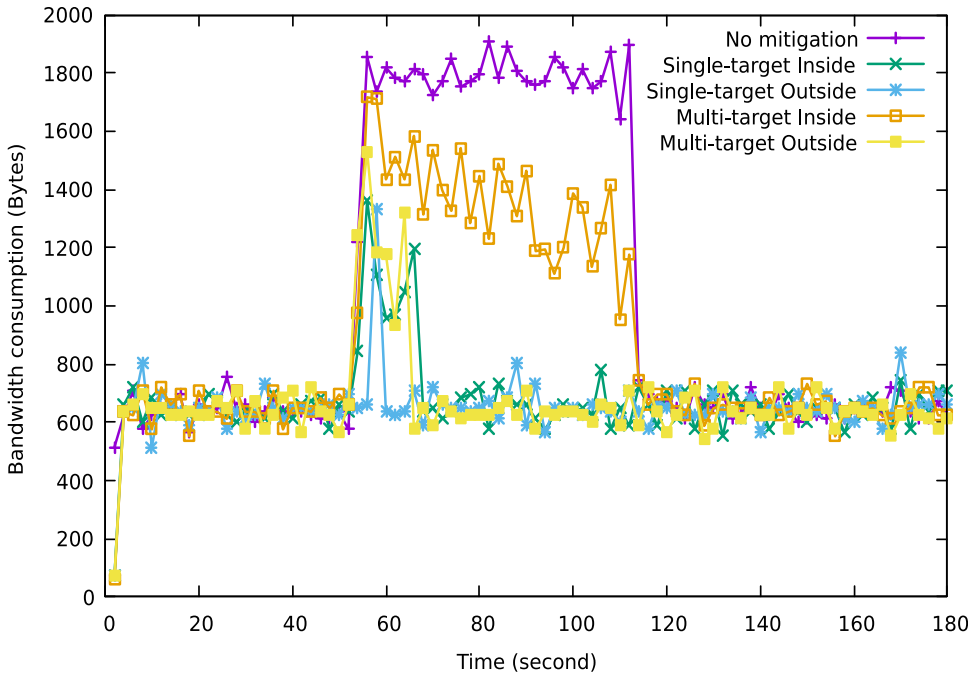

Fig. 9 OFSwitch-to-controller link bandwidth consumption evaluation

multiple windows are needed to fully mitigate the traffic, the adaptive threshold computation may consider the attack traffic as normal during those windows. As a result, the brown-colored attack traffic, representing multi-target attacks within the controller's network, takes longer to return to normal bandwidth consumption. On the other hand, the light blue-colored attack traffic, representing single-target attacks outside of the controller's network, returns to normal immediately. This is because the mitigation system only considers the destination IP address for mitigating in the case of attacks outside the controller's network, assuming that such traffic is unlikely to be legitimate and aiming to protect the controller from overload.

5 Conclusion

SDN is a cutting-edge technology revolutionizing networking by separating the data and control planes, allowing for centralized network management. However, traditional network management struggles in the dynamic and diverse environment of the Internet of Vehicles (IoV). Integrating SDN with IoV, termed SD-IoV, offers significant benefits in managing IoV centrally. Yet, this fusion brings challenges, notably the central controller becoming a vulnerable point for attacks. Attackers could overwhelm the controller with packet floods, causing congestion and disrupting legitimate traffic. In this study, we have proposed an efficient and lightweight detection and mitigation algorithm in order to enhance the security of the controller within highly dynamic SD-IoV networks. The algorithm is designed by using entropy values and flow rate of packet-ins for detection and payload length analysis and protocol number filtration for mitigation. The proposed system aims to detect a DDoS attack with in a single window by simultaneously check the flow rate and entropy of a single windows. We then simulated the proposed system using Mininet-WiFi SDN emulation tool for performance evaluation. We have also used Scapy tool for generating both normal and attack traffic. At the end, we have compared the proposed system with the previous works by taking detection time and detection accuracy as a metrics. The mitigation capability of the proposed system has been assessed by taking the variation of controller load and link congestion during attack traffic and after applying mitigation rule. The system showcases as it minimizes the controller load and link congestion by mitigating the attack traffics. As future work, we recommend leveraging machine learning and deep learning to enhance detection accuracy and mitigation performance. Furthermore, we suggest utilizing edge and fog computing to minimize delays.

Author contributions Mr. Behaylu Tadele Alemu wrote the main manuscript text and Dr. Alemu Jorgi Muhammed deeply reviewed the manuscript and guides us how to write the manuscript. Mr.Habtamu Molla Belachew and Mr.Mulatu Yirga Beyene prepares the figures and table. All authors reviewed the manuscript.

Funding The authors declare that they did not receive any funds, grants, or other support during the preparation of this manuscript.

Data availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. Alemu, B.T., Muhammed, A.J.: Controller-targeted DDoS attack detection and mitigation in software-defined internet of vehicles (SD-IoV). In: 2023 International Conference on Information and Communication Technology for Development for Africa (ICT4DA), pp. 138–143. IEEE (2023)
2. Hakimi, A., Yusof, K., Azizan, M., Azman, M., Hussain, S.: A survey on internet of vehicle (iov): applications and comparison of vanets, IoV and SD-IoV. ELEKTRIKA-J. Electr. Eng. **20**(3), 26–31 (2021). https://doi.org/10.11113/elektrika.v20n3.291
3. Mohanty, S., Sahoo, B., Apat, H., Sahoo, K.: An overview of software-defined internet of vehicles. In: SDN-Supported Edge-Cloud Interplay for Next Generation Internet of Things, 71–86 (2022) https://doi.org/10.1201/9781003213871-4
4. Yang, Y.: A sdn-based traffic estimation approach in the internet of vehicles. Wirel. Netw. **1**, 1–12 (2021)
5. Zhang, T., Xu, C., Zou, P., Tian, H., Kuang, X., Yang, S., Zhong, L., Niyato, D.: How to mitigate DDoS intelligently in SD-IoV: a moving target defense approach. IEEE Trans. Ind. Inform. **19**(1), 1097–1106 (2022). https://doi.org/10.1109/TII.2022.3190556
6. Aladaileh, M.A., Anbar, M., Hasbullah, I.H., Sanjalawe, Y.K.: Information theory-based approaches to detect DDoS attacks on software-defined networking controller a review. Int. J. Educ. Inf. Technol **15**, 83–94 (2021)
7. Aladaileh, M.A., Anbar, M., Hintaw, A.J., Hasbullah, I.H., Bahashwan, A.A., Al-Amiedy, T.A., Ibrahim, D.R.: Effectiveness of an entropy-based approach for detecting low-and high-rate DDoS attacks against the sdn controller: experimental analysis. Appl. Sci. **13**(2), 775 (2023)
8. Rahouti, M., Xiong, K., Xin, Y.: Secure software-defined networking communication systems for smart cities: current status, challenges, and trends. IEEE Access **9**, 12083–12113 (2020). https://doi.org/10.1109/ACCESS.2020.3047996
9. Ominike Akpovi, A., Adebayo, A., Osisanwo, F.: Introduction to software defined networks (sdn). Int. J. Appl. Inf. Syst. **1**, 10–14 (2016). https://doi.org/10.5120/ijais2016451623
10. Singh, J., Behal, S.: Detection and mitigation of DDoS attacks in sdn: a comprehensive review, research challenges and future directions. Comput. Sci. Rev. **37**, 100279 (2020). https://doi.org/10.1016/j.cosrev.2020.100279
11. Khairi, M., Syed Ariffin, S.H., Abdul Latiff, N.M., Abdullah, A.S.: A review of anomaly detection techniques and distributed denial of service (DDoS) on software defined network (SDN). Eng. Technol. Appl. Sci. Res. **8**(2), 2724–2730 (2018). https://doi.org/10.48084/etasr.1840
12. Abbas, T., Muhammad, A., Song, W.-C.: SD-IoV: Sdn enabled routing for the internet of vehicles in road-aware approach. J. Ambient Intell. Hum. Comput. **11**, 1265–1280 (2020). https://doi.org/10.1007/s12652-019-01319-w
13. Jiacheng, C., Haibo, Z., Ning, Z., Peng, Y., Lin, G., Xuemin, S.: Software defined internet of vehicles: architecture, challenges, and solutions. J. Commun. Inf. Netw. **1**(1), 14–26 (2016). https://doi.org/10.1007/BF03391543
14. Ramalakshmi, R., Kavitha, D.: DDoS attack mitigation using distributed sdn multi controllers for fog based iot systems. Int. J. Intell. Syst. Appl. Eng. **12**(4s), 57–69 (2024)
15. Aladaileh, M., Anbar, M., Hasbullah, I., Sanjalawe, Y., Chong, Y.: Entropy-based approach to detect DDoS attacks on software defined networking controller. Comput. Mater. Contin. **69**, 373–391 (2021). https://doi.org/10.32604/cmc.2021.017972
16. Kareem, M.I., Jasim, M.N.: Entropy-based distributed denial of service attack detection in software-defined networking. Indones. J. Electr. Eng. Comput. Sci. **27**(3), 1542–1549 (2022). https://doi.org/10.11591/ijeecs.v27.i3.pp1542-1549
17. Abdullah, M., Al-awad, N., Hussein, F.: Implementation of entropy-based distributed denial of service attack detection method in multiple pox controllers. Rev. Comput. Eng. Stud. **6**(2), 29–38 (2019). https://doi.org/10.18280/rces.060201
18. Aladaileh, M.A., Anbar, M., Hasbullah, I.H., Bahashwan, A.A., Al-Sarawn, S.: Dynamic threshold-based approach to detect low-rate DDoS attacks on software-defined networking controller. Comput. Mater. Contin. **73**(1), 1–10 (2022)
19. Bhayo, J., Shah, S.A., Hameed, S., Ahmed, A., Nasir, J., Draheim, D.: Towards a machine learning-based framework for DDoS attack detection in software-defined iot (sd-iot) networks. Eng. Appl. Artif. Intell. **123**, 106432 (2023)
20. Khedr, W.I., Gouda, A.E., Mohamed, E.R.: Fmdadm: a multi-layer DDoS attack detection and mitigation framework using machine learning for stateful sdn-based iot networks. IEEE Access **11**, 28934–28954 (2023)
21. Li, Z., Kong, Y., Jiang, C.: A transfer double deep q network based DDoS detection method for internet of vehicles. IEEE Trans. Vehic. Technol. (2023)
22. Javanmardi, S., Ghahramani, M., Shojafar, M., Alazab, M., Caruso, A.M.: M-rl: a mobility and impersonation-aware ids for DDoS udp flooding attacks in iot-fog networks. Comput. Secur. **1**, 103778 (2024)
23. Todorova, M.S., Todorova, S.T.: DDoS attack detection in sdn-based vanet architectures. Master Appl. Sci. **175**, 1–10 (2016)
24. Valizadeh, P., Taghinezhad-Niar, A.: DDoS attacks detection in multi-controller based software defined network. In: 2022 8th International Conference on Web Research (ICWR), pp. 34–39. IEEE (2022)
25. Sahoo, K., Puthal, D., Tiwary, M., Rodrigues, J., Sahoo, B., Dash, R.: An early detection of low rate DDoS attack to SDN-based data center networks using information distance metrics. Future Generat. Comput. Syst. **89**, 685–697 (2018). https://doi.org/10.1016/j.future.2018.07.017
26. Biasi, G., Vieira, L.F., Loureiro, A.A.: Sentinel: Defense mechanism against DDoS flooding attack in software defined vehicular network. In: 2018 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2018)
27. Omar, T., Ho, A., Urbina, B.: Detection of DDoS in sdn environment using entropy-based detection. In: IEEE International Symposium on Technologies for Homeland Security (HST), pp. 1–4 (2019)
28. Mousavi, S.M., St-Hilaire, M.: Early detection of DDoS attacks against software defined network controllers. J. Netw. Syst. Manag. **26**, 573–591 (2018)
29. Cabarkapa, D., Pronić-Rančić, O., Rancic, D.: DDoS attack detection in software-defined networks based on multiple entropy. (2022)
30. Ahmad, F.: Detection and mitigation of malicious DDoS floods in software defined networks (2023). https://doi.org/10.21203/rs.3.rs-2421818/v2
31. Sumantra, I., Gandhi, S.I.: DDoS attack detection and mitigation in software defined networks. In: 2020 International Conference

on System, Computation, Automation and Networking (ICSCAN), pp. 1–5. IEEE (2020)
32. Anil, A., Rufzal, T., Adat Vasudevan, V.: DDoS detection in software-defined network using entropy method. In: Proceedings of the Seventh International Conference on Mathematics and Computing: ICMC 2021, pp. 129–139. Springer (2022)
33. Biron, Z.A., Dey, S., Pisu, P.: Real-time detection and estimation of denial of service attack in connected vehicle systems. IEEE Trans. Intell. Transp. Syst. **19**(12), 3893–3902 (2018)
34. Nayak, R.P., Sethi, S., Bhoi, S.K., Sahoo, K.S., Jhanjhi, N., Tabbakh, T.A., Almusaylim, Z.A.: TbDDoSa-md: trust-based DDoS misbehave detection approach in software-defined vehicular network (sdvn). CMC-Comput. Mater. Contin. **69**(3), 3513–3529 (2021)
35. Aladaileh, M.A., Anbar, M., Hintaw, A.J., Hasbullah, I.H., Bahashwan, A.A., Al-Sarawi, S.: Renyi joint entropy-based dynamic threshold approach to detect DDoS attacks against sdn controller with various traffic rates. Appl. Sci. **12**(12), 6127 (2022)
36. Dhaliwal, A.: Detection and mitigation of syn and http flood DDoS attacks in software defined networks. (2017)

Behaylu Tadele Alemu received the M.Sc.degree in Computer Networks and Communications from Wollo University, KIoT, in 2023,and B.Sc. degree in Computer Science from Samara University, Ethiopia. He is currently a Lecturer with the Computer Science department, Debark University. His current research interests include Internet of Things, Software Defined Networks (SDN), SD-IoV, Machine Learning, Wireless Communication, Deep Learning, Computer Network Security.

Alemu Jorgi Muhammed (Member, IEEE) received the B.Sc. and M.Sc. degrees in information technology and information science from Addis Ababa University, Ethiopia, in 2005 and 2010, respectively, and the Ph.D. degree in information and communication engineering from Southwest Jiaotong University (SWJTU), Chengdu, China, in 2020. He is currently an Assistant Professor with the Postgraduate Program in Computer Networks and Communications, School of Informatics, and also worked as the Director of ICT Services, Wollo University, Ethiopia. He has authored research papers in high quality journals and conferences. He has also served as a Reviewer for numerous journals and conferences, such as IEEE Access, IEEE Transactions on Wireless Communications, IEEE Transactions on Vehicular Technology, China Communications, Journal of Computer Communications, IEEE Transactions on Cognitive Communications and Networking, IEEE ICC, and WCSP. His main research interests target on resource allocation and optimization in the 5G/6G wireless networks, focusing on non-orthogonal multiple access (NOMA), millimeter-wave communications, cooperative communications, heterogeneous small cell networks, cognitive networks, wireless resource allocation and management in V2X communication, and machine learning for wireless communication, such as DRL, mobile edge computing, and the IoT.

Habtamu Molla Belachew received the M.Sc. degree in Computer Networks and Communications from Wollo University, KIoT, in 2023, and B.Sc. degree in Information Technology from Bule Hora University, Ethiopia. He is currently a Lecturer with the Information Technology department, Debark University. His current research interests include Internet of Things, Computer Networking, VANET Network Architecture, Machine Learning, Wireless Computing, Computer Networks Security, Cloud Computing, Network Administration, Network technology, SDN.

Mulatu Yirga Beyene received the M.Sc. degree in Computer Networks and Communications from Wollo University, KIoT, in 2023, and B.Sc. degree in Information Technology from Mizan Tepi University, Ethiopia. He is currently a Lecturer with the Information Technology department, Debark University. His current research interests include Internet of Things, Software Defined Networks (SDN), SD-IoV, Machine Learning, Wireless Communication, Deep Learning, Computer Network Security.