Article

Not peer-reviewed version

Mitigating Distributed Denial of Service Attacks in Software-Defined Networking

Abdullah Alnajim , Faisal Alotaibi , Sheroz Khan [*]

Posted Date: 9 September 2024

doi: 10.20944/preprints202409.0641.v1

Keywords: Software-Defined Networking (SDN); Distributed Denial of Service (DDoS) attack; sampling Flow (sFlow); OpenFlow; OpenDaylight controller

Article

Mitigating Distributed Denial of Service Attacks in Software-Defined Networking

Abdullah M Alnajim [1], Faisal Alotaibi [2] and Sheroz Khan [3,*]

1 Department of Information Technology, College of Computer, Qassim University, Buraydah 51452, Saudi Arabia
2 Department of Computer Science, College of Engineering and Information Technology, Onaizah Colleges, Qassim 56447, Saudi Arabia
3 Department of Electrical Engineering, College of Engineering and Information Technology, Onaizah Colleges, Qassim 56447, Saudi Arabia
* Correspondence: cnar32.sheroz@gmail.com

Abstract: Distributed Denial of Service (DDoS) is among the commonly used type of network attacks, particularly aimed at the IoT devices in the emerging future wireless services specially for strictly latency-limited applications. DDoS attacks pose serious threats for many entrepreneurial businesses as they can prevent legitimate customers from accessing network resources of corporate websites, demanding intelligent analytics prior to entertaining a service request. DDoS involves exploiting the vulnerability of IoT devices by launching distributed multi-point attacks to generate high traffic by flooding the network of victims. The consequences of DDoS attacks could be worst in any Software-Defined Networking (SDN) than in traditional networks because SDN is configured as centralised architecture, failing to address the weaknesses current paradigm. However, SDN can play a crucial role in mitigating the effects of DDoS attacks as it bears the features of easy programmability, manageability, flexibility, and scalability. In this manuscript preliminary framework to detect DDoS attacks by mitigating their threats in an SDN architecture is presented. The authors propose a defense mechanism that uses OpenFlow and sFlow to mitigate the effects of DDoS attacks. The initial results show that the proposed model can minimize efficiently the consequences of DDoS attacks in an SDN type of architectural paradigm.

Keywords: Software-Defined Networking (SDN); Distributed Denial of Service (DDoS) attack; sampling Flow (sFlow); OpenFlow,; OpenDaylight controller

---

1. Introduction

Since its inception, the users of Internet have increased exponentially making it to assume the architectures where one can justifiably call it as the Internet of Things (IoT) or INternet of Everything (IoE), enabling merger of physical and digital worlds, ushering an era wherein the sensations by objects and humans across the oceans are being felt remotely similar to the feelings of users interacting among them directly. This architectural structure involves internet-connected devices, smart objects, sensors, and associated web-based services demanding processing before communication - all together make what is referred to as IoT or IoE today. Currently, more than three billion users surf the Internet for banking transactions, shopping from vendors around the globe, web-based monitoring and control, social media, and so many other activities [1]. The so emerging web of everything has been becoming heterogeneous in configuration, and has gotten diverse requirements of intelligent, seamless, and ubiquitous connectivity for end-users. These requirements are impossible to be addressed by the intervention from network administrators, operators or end-users, rather they can possibly be met only if the internet is made to be self-sustaining and self-responding. The increase in the number of IoT-based services in smart homes, smart healthcare, smart cities, smart agriculture, and smart supply chain, are all out there to add value to living conditions. Such a large number of devices is associated with many issues in the current legacy, a kind of outdated networking paradigm, for instance, network manageability is still a challenge. Network devices programmable feature is also an issue restricting developers and researchers from moving onward. In addition, to meet such highly dynamic and extreme requirements there is the need of interoperability and integration of edge computing, block-chain and wireless channel resources to ensure that services are instantly provided in the current network architecture, in which many physical systems or devices connected to the Internet can easily be operated and managed remotely. Concurrently developments in Software Defined Networks (SDN) provide the users with opportunity of embedded intelligent services in computer networks. But at the same time SDN is definitely a parading shift from proprietary platforms of hardware switches in legacy network systems relaying more on the skills and expertise of software community platforms to develop much of the control plane soft-wares and protocols [2]. Finally, information security and privacy remain prominent areas of concerns. All Internet-connected systems are prone to viruses, hacking, and sniffing to face an equally diverse kind of the dangers of threats. Hackers are motivated by various factors such as financial gains, mere fun, showing-off revenge, proving itself a point, damaging other people's businesses, and stealing information, etc. The so developing network of billions of hand-held or stick-on sensors and devices as well as computers is running anytime loaded with confidential and valuable personal and professional information through wired and wireless setups of extremely heterogeneous nature. Such devices need to be user friendly and hence are required with minimal or nonexistent security mechanisms to provide astonishing web-based applications. They are by default are more vulnerable to all sorts of threats and accordingly network security breaches can lead to massive financial losses both for businesses and consumers [3,4].

There have been no significant changes in the conventional networking architectures, consisting of mainly sets of routers and switches alongside devices manufactured by diverse vendors. These devices are supposed to be used together to build heterogeneous networks. Given that these devices may be from different vendors, and coupled with different technicians of equally varying brands, and hence are required to build heterogeneous networks. Different systems from different vendors might be incompatible, making heterogeneous networks expensive to configure and difficult to operate, monitor, and maintain. The process of configuring different systems also increases the vulnerability of the hardware and their software resources. This demands for improved networking architecture designed to combat the challenges of these developing networks by making them easily controllable, and dynamic enough to accommodate different devices from various vendors and to meet the demands of changing workloads by scaling up or down resources with the help of cloud infrastructure. Software-Defined Networking (SDN) is among the most promising approaches for improving current hardware-based networking systems. Hence, this attempt is to develop a DDoS detection model for dynamic and heterogeneous IoT or IoE systems, which can be implemented in a smart environments to conclude that the geo-spatial distribution of attacking sources follow certain patterns [5,6].The IoT-based environments make lifestyle more productive by adding value to living conditions of a digitally networked society. However, as the number of connected devices increases, the complexity of systems has also grown in proportionate manners, making these systems exposed to several types of malfunctions and threats making the DDoS attacks using the computational power of compromised hosts as attack platforms.

The most popularly known reasons for degrading the availability of services on the Internet is that from Distributed Denial of Service (DDoS) attacks. Furthermore, DDoS attacks can range from the misuse of application-level vulnerabilities to high-volume flooding on a network. For instance, flooding a network with high volumes of irrelevant data to create traffic congestion issues or attacking some vulnerable applications to render them useless. Although it is easy to probe the service availability and to ease traffic by de-congesting the network, the most significant challenge lies in differentiating between legitimate congestion based on the traffic flow characteristics and attacker-initiated congestion as a result of nodes flooding, because the two are similar in kind and effects. SDN as an emerging form of network that addresses these shortcomings by separating network control and management from the data plane to minimise complexity by enhancing increased network management through implementing flow-rules in thousands per server rack [7]. One way of combating such DDoS menaces is through identifying for network users the legitimate IoT traffic from anomalous DDoS-generated traffic, bearing thus the SDN with the ability to detect in order to respond to abnormalities in the network in timely manner [8].

This work is mainly motivated by the emergence and importance of SDN networks as a future networking paradigm primarily meant for addressing security concerns. An SDN is basically an IoT-based setup in which the devices are networked together with software applications deployed at the SDN controller level. SDN technology opens a new paradigm for networks capable of managing network efficiently by transforming complex network architectures into flexible, extensible, and feasible, by providing intelligence on networks through possible adoption of machine learning programs. It bears the essential features of resource pooling and on demand-selfservice through integration with cloud computing that has attracted the attention of researchers in academia and industry [9].

Collecting network deploy-ability information is much easier in SDN environment, which may support improved algorithmic designs for detecting attacks. A new generation of applications can take advantage of the well-informed SDN agents to improve policy enforcement and to manage traffic congestion detection with required mitigation. These applications could block malicious intruders before they enter the critical regions of networks. SDN-based security approaches have been rated as trends for future network security solutions. The control plane which is decoupled from the data plane in the SDN environment can dynamically enforce flow rules [1] when required by the data plane. It allows efficient network control implementations. Therefore, it will help to detect and mitigate DoS and DDoS attacks through realizing programmable features.

In this framework, the authors propose an SDN-based security system that aims at protecting the SDN network from DDoS attacks. The proposed algorithm in this system uses statistical data collected by the sFlow and OpenFlow protocols to detect and mitigate the DDoS attacks. Also, the proposed algorithm can identify high network traffic to block it based on selected threshold values. The main contribution of this work is proposing a system that uses both sFlow to detect DDoS attacks and OpenFlow to mitigate these attacks. Besides, the authors to present preliminary results and evaluation of the proposed model.

The rest of the paper is organized with Section 2 presenting general background about Software-Defined Network (SDN), DDoS attacks, defense approaches, and related work. Section 3 presents problem statement and modeling while Section 4 explains the proposed models, and Section 5 concludes the paper with results and concluding observations.

2. Background

This section presents a review of Software Defined Networking (SDN) and Distributed Denial of Service attack (DDoS) supported by related work and its discussions.

2.1. Software Defined Networking (SDN)

SDN paradigm is introduced to make up for the shortcomings of traditional network by introducing programmability, compatibility, cost, manageability, and so many other distinctive features. The main idea in an SDN is to separate in network switches data plane and the centrally located device of control plane. The plane that determines where to send traffic is called control plane, whereas the plane that executes these traffic sending decisions to forward the traffic onward is known as data plane, as shown in Figure 1.

Furthermore, SDN architectures generally have three components of functionality, consisting of: i) Applications: SDN applications are programs that communicate sensations and behaviors and needed resources with the SDN Controller via application programming interface (API), ii) Controller: it is a logical entity that receives instructions or requirements from the SDN application layer and relays them to the networking components, and iii) Networking Devices: SDN networking devices control the forwarding and data processing capabilities for the network that includes forwarding and processing of the data packet paths [10]. DDoS attacks transform the vulnerable IoT devices into

[1] Flow can be defined as a set of consecutive packets that have equivalent properties traveling in the same network.

becoming botnets, which then cause severe disruptions to the IoT system by compromising the nodes to work as execution platform for attackers [11].

OpenFlow is a communication protocol in an SDN environment that is used by control (SDN controller) and data plane of network devices to communicate with each other. Furthermore, SDN controller sets the path of the packets traveling around the SDN network routes and then it configures the network devices with its flow-rules. Besides, OpenFlow protocol allows communication between network devices from different vendors, such that devices from any vendor can be controlled. Finally, OpenFlow protocol can be used to control layer-three switches' packet flow tables remotely [12].

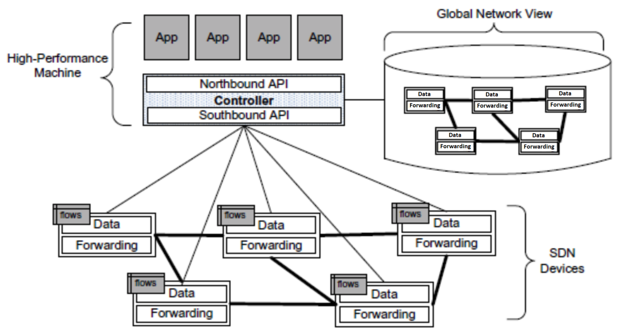

Figure 1. Software Defined Networking (SDN) [7].

2.2. Distributed Denial of Service (DDoS) Attack

2.2.1. Definition

DDoS attacks are undoubtedly among the leading causes of concern for many companies, organizations, and institutions. The DDoS attacks are maliciously coordinated against the online services such as commercial websites, banks' websites, government websites, etc. Moreover, the DDoS attacks are usually performed by a massive number of autonomous programs aimed at compromising system resources for a specified period of time until the victim target becomes unable to offer services on the malicious agents' behalf.

2.2.2. How DDoS Attacks Are Performed

The first step of launching DDoS attacks, not primarily meant to steal data, is to recruit an army of bots or zombies. Furthermore, in pursuance of converting a computer into a zombie, hackers design, develop and implement a specialized Malware, which can be installed on a large number of machines that have visited some unsafe websites containing this Malware. It can be spread in targeted computers via email attachments, compromised websites, or organization networks. As a result, legitimate users can get Malware and turn into a bot unintentionally by running this Malware. This kind of Malware provides access for attackers to get computers affected. Once a computer becomes a zombie, it gets connected to the attackers' commands that starts controlling the server and begins to accept orders from the attackers. The orders may contain information about the target server, time of the attack, attack method, and duration of firing. In addition, an army of bots named as Botnet is formed that usually consists of thousands of bots, making IoT devices insecure and vulnerable to DDoS attacks as shown in Figure 2. Each time the attacker wants to launch an attack, they send a command to the Command and Control server. As a result, the command and control (C & C) server sends commands to every bot connected to the server to launch the attack on the victim server as shown in Figure 3 [13].

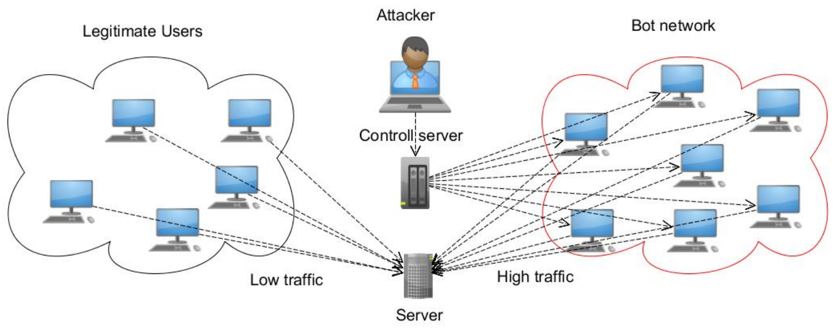

Figure 2. Distribute Denial of Service (DDoS).

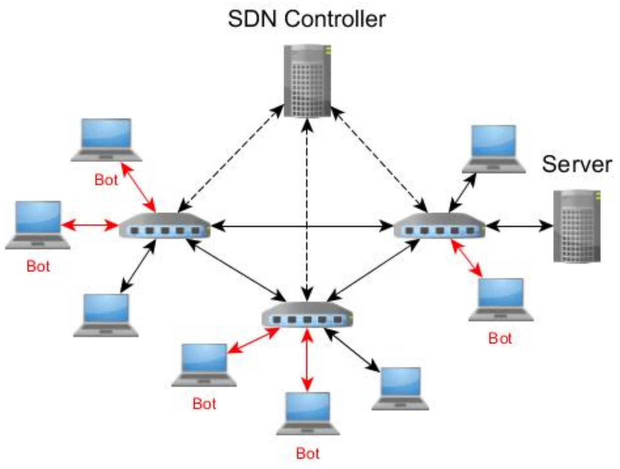

Figure 3. Distributed Denial of Service (DDoS) attack in Software Defined Networking (SDN).

2.2.3. Target of DDoS Attack

The DDoS attacker, in general, targets these network components in order to cut the services off the legitimate users: (i) Routers, (ii) Links, (iii) Firewalls and defense systems, (iv) Victims infrastructure, (v) Victims OS, (vi) Current communications, and (vii) Victims applications [14,41]. Also, in the list of potential targets are vulnerable IoT devices which when attacked form botnets thus threatening the security of the IoT system via DDoS attacks.

2.2.4. DDoS Attack Types

DDoS attacks can be classified into three main types: i) Value Based Attacks, which include, User Datagram Protocol (UDP) floods, Ping Flood, and spoofed-packet floods, ii) Protocol-Based Attacks such as SYN Flood, fragmented packet attacks, Ping of Death, and Smurf DDoS, and iii) Application layer Attacks, for example, low-and-slow attacks, GET/POST floods, attacks that target Apache, and attacks that target Windows, and OpenBSD vulnerabilities) [15]. In addition, the authors in [16] classify DDoS attacks into: (i) application attack, where the attackers target the applications, while the (ii) resource and host attacks make the resources unavailable, (iii) Network attacks that target the network bandwidth, and the iv) infrastructure attacks that target the Domain Name Server (DNS).

2.2.5. Challenges of DDoS Mitigation

There are many challenges to build SDN controller that can detect and mitigate DDoS attacks. These challenges are: i) Size of Botnet, for example, if the size of Botnet is large, it becomes a very difficult to deal with, ii) Abnormal traffic detection with accuracy, iii) Long-term attacks that last longer, and iv) Large-scale testing to meet the needs of multi Gbps networks [15].

2.2.6. Defense Approaches

According to Holl in study [16], there are three main defense mechanisms to detect and mitigate the DDoS attacks. The first approach is called Proactive Defense Mechanisms framework [17] that has proposed a technique that does not mitigate a DDoS attack directly. Nevertheless, it designs and efficiently builds the infrastructure so that it can survive an impending DDoS attack, referred to as Cloud hosting. The second approach is called Reactive Defense Mechanisms, in which the main idea behind reactive defense mechanisms is to mitigate DDoS attacks or to stop them if possible when a DDoS attack happens. Finally, the third approach is called Post-Attack Analysis: this technique analyzes DDoS attacks and find their patterns for information to trace back the attackers for blocking them [18–20]. Advances in the manufacturing of various miniaturized sensor systems and the development of numerous web services along with cloud computing have made it possible to make virtually any isolated system communicate with similar devices, which are expected to run in billion at the moment on the Internet. Such complicated systems are subject to several challenges with Distributed Denial of Service (DDoS) as the most commonly occurring attacks [21,22].

3. Problem Modeling Modeling and Statement

Mitigating DDoS attacks is one of the main causes of concern for network administrators. SDN as shown in Figure 3, is a promising networking paradigm that can be used to tackle DDoS attacks. Furthermore, it is assumed that the network consists of a set of nodes, $\mathcal{N} \neq \emptyset$, where the number of network nodes is $N = \mathcal{N}$, $N \in \mathbb{Z}$, and $N \geq 4$. In addition, there is a subset of these nodes as hosts $\mathcal{H} \subset \mathcal{N}$ such that the number of these hosts is $H = \mathcal{H}$, $H \in \mathbb{Z}$ and $H \geq 2$, and some of these hosts (or users) are legitimate in the sense $\mathcal{U} \subset \mathcal{H}$, the number of these legitimate users $U = \mathcal{U}$ and some of them are Bot or Zombie users $\mathcal{B} \subset \mathcal{H}$, the number of these bots are $B = \mathcal{B}$, and $B \in \mathbb{Z}$. Moreover, some of these nodes are switches $\mathcal{S} \neq \emptyset$, $\mathcal{S} \subset \mathcal{N}$, $S = \mathcal{S}$, and $S \in \mathbb{Z}_{>0}$. Finally, there is at least one controller in the network $\mathcal{C} \neq \emptyset$, $\mathcal{C} \subset \mathcal{N}$, the number of controller in the network is $C = \mathcal{C}$, and $C \in \mathbb{Z}_{>0}$. Each host can send traffic with rate $\gamma \geq 0$ and $\gamma \in \mathbb{R}$. The bandwidth of the link between switches and between users and switches is D, and $D > 0$. The main problem here is to detect bots in $\mathcal{B}$ that are generating high traffic, and stop them from overwhelming the switches and bandwidth. Further, to allow only legitimate users in $\mathcal{U}$ to send packets through links and switches. The proposed model is explained in the section that follows how it is designed to tackle the problem at hand by mitigating DDoS attack.

4. The Proposed Method Proposed

he authors have proposed in this framework a defense approach to detect and mitigate DDoS attacks in an SDN environment.

4.1. DDoS Detection and Mitigation System Using sFlow

The suggested DDoS detection and mitigation system is based on real-time traffic monitoring in Software Defined Network (SDN) network environment as a defense model against DDoS attacks. The security system uses statistics maintained while monitoring the traffic passing through SDN OpenFlow switches to detect malicious traffic. Once the attack is detected, the proposed defense system tracks the hosts causing them in order to block them for a specific period of time, allowing only legitimate users to use network facilities by sending packets through OpenFlow switches and links in the SDN network.

Traffic monitoring is crucial for DDoS attacks detection and mitigation by using tools, one of the available tools is sampling Flow (sFlow). Furthermore, sFlow is a sampling technique that can be used to monitor network traffic. The sFlow can be utilized to detect different type of DDoS attacks such as SYN flood, UDP user datagram protocol, etc. Moreover, the OpenFlow and sFlow protocols have provided SDN researchers with an integrated flow monitoring and control system the OpenFlow controller can be utilized to specify the flows to be monitored by sFlow, making controller the layer that sends packet-out messages to program forwarding rules to switches. In this context, in order to get the most out of the sFlow, we will use the powerful analytic engine called sFlow-RT to collect real-time statistics of the switches. In general, the sFlow-RT can be used for traffic monitoring by recording related-time statistics. It is used for load balancing, DDoS attack detection and mitigation. In addition, there are three main components of sFlow which are: i) sFlow collector in external controller or in SDN controller, ii) sFlow agents that are built into the Openvswitch, and iii) sFlow protocol. [23].

The main idea is that the sFlow agents are built into the OpenFlow switches in an SDN network. The primary task of sFlow agents is to send samples of the network traffic from a particular network device such as switches or routers to sFlow collector in the external controller. Then, the sFlow collector can utilize tools like inMon sFlow-RT to automatically detect and mitigate the huge long-queue flows in real-time, see Figure 4 for more details. The sFlow agents embedded in the OpenFlow switches communicate with sFlow collector via sFlow protocol, whereas the OpenfFow switches communicate with OpenDaylight over OpenFlow protocol as shown in Figure 5. The sFlow agent encapsulates sample flow and interfaces counters into sFlow data-grams. These data-grams are sent by sFlow agents to sFlow collector in the external controller via sFlow protocol.

The sFlow agents sample traffic packets based on probability specified by network administrators, network operators, or the end-users. These agents also send interface counters to sFlow collector. Network users can also set threshold for traffic and sampling rates. The primary roles of the SDN collector are to analyze the traffic samples and take measures against DDoS attacks in order to mitigate or stop these attacks if possible.

In addition, the REST APIs, also known an architectural style for an application program interface (API) help each application to retrieve metrics, configure flows, receive notifications, and set appropriate threshold values using *GET*, *PUT*, *POST* and *DELETE* data types, which refers to the reading, updating, creating and deleting of operations concerning resources. The sFlow agents can be embedded into OpenFlow switches by using the following code:

```
sudo ovs-vsctl – –id=@sflow create sflow agent=eth1
target=\"192.168.56.102:6343\" sampling=300 polling=15
– – set bridge s1 sflow=@sflow
```

The above instruction codes will configure or enable sFlow in OpenvSwitch. The sFlow agent of *eth*1 is embedded into the IP address (192.168.56.102) of the controller. Besides, the sampling rate is set at 300 which means for every 300 packets captured by sFlow agent, one packet is sent to sFlow collector. The polling interval is set at 15 seconds, which means that the agent will send a data-gram to sFlow collector every 15 seconds. Figure 5 shows flow diagram of sFlow agent processing.

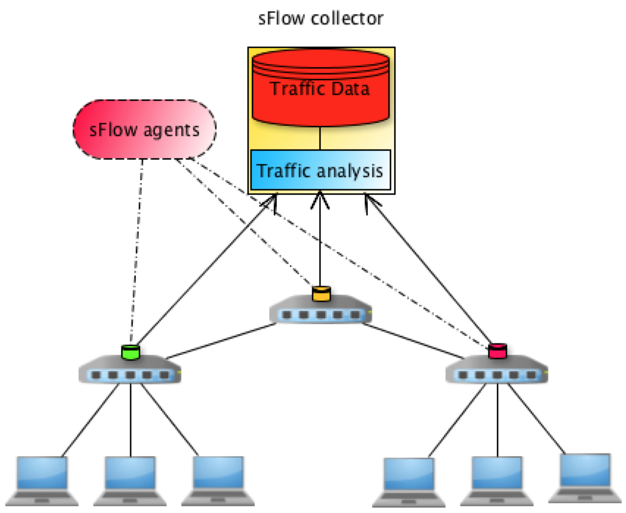

Figure 4. sFlow agents and sFlow collector.

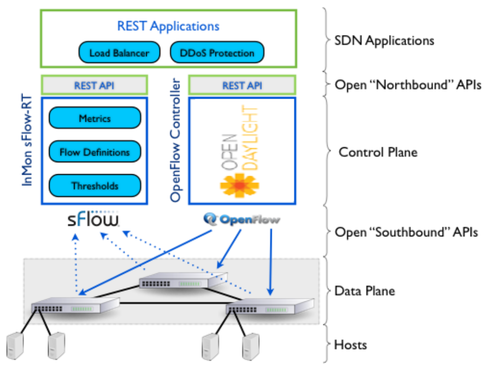

Figure 5. sFlow Agent Embedded in Switch [?].

All deployed sFlow agents in OpenFlow switches will send traffic samples to the sFlow Real-time (RT) analytic engine. This engine sends a real-time notification of the potential DDoS and information about Bots and the victim server to the relevant application. This Application triggers the controller by using $RESTAPI$ to send instructions to switches to mitigate the DDoS attack. The listed Algorithm 1 demonstrates a pseudo-code designed for DDoS detection and mitigation. The algorithm checks the traffic flows λ at any time t in all switches in the SDN network through sFlow that allows to check the usage of the bandwidth accordingly. Once the traffic reaches a pre-set threshold value θ that identifies to mean a DDoS attack is detected. In this paper θ is kept equal to 10% of the bandwidth (D), $\theta = \frac{D}{100} \times 10$. The utilization of bandwidth is u. It is noteworthy to note that θ could take any other value, depending on network administrator. The algorithm then blocks the users who behave like a Bot by keeping them on hold for T seconds.

5. Testbed and Evaluation

In this work Mininet has been used as network emulator and OpenDaylight as the SDN Controller to develop the testbed to show how it may potentially function as a reference topology for security applications. The computational system used for simulation purposes has been a Core i7-4790@3.60GHz CPU and 32GB of RAM processor. Also, has been used Oracle VM Box to create a virtual machine for accommodating the concerned SDN network devices.

5.1. Network Emulator

The SDN has five main components consisting of 1) application plane (AP), 2) northbound interface (NBI), 3) control plane (CP), 4) southbound interface (SBI) and 5) data plane (DP). The

Algorithm 1 Flow Monitoring and DDoS Attack Detection

```
1:  procedure ATTACK DETECTION
2:      Set all parameters.
3:      Start:
4:      for each switch $S \in \mathcal{S}$ do                      ▷ For all switches
5:          for each packet $P_i$ do                                  ▷ For all packets
6:              Add source IP address of new flow to the flow table.
7:              Add OpenFlow rules to the flow table.
8:              Monitor packet flow (rate and duration) using sFlow.
9:              if $\lambda > \theta$ then                            ▷ Check bandwidth usage
10:                 The traffic is malicious (DDoS attack).
11:                 Block the attackers for a period of time, $T$.
12:             else
13:                 The traffic is legitimate, no action needed.
14:             end if
15:         end for
16:     end for
17:     goto Start.
18: end procedure
```

control plane coupled with SDN controllers interacts with applications to transmit the data flow to its destination to ensure device sharing and QoS-aware data routing among the devices connected. There are many network emulators that have been used in research such as Network Simulator 3 (NS3), OPNET, OMNeT++, NetSim, REAL, QualNet, and J-Sim [24]. However, the authors have decided to use a Mininet as a network emulator [25] for the reasons of simplicity and easy implementation. Furthermore, Mininet can emulate a complete network of end-hosts, controller, links for large SDN networks deployed on limited resources of a Virtual Machine of a single Computer. Also, Mininet switches support OpenFlow protocols.

5.2. SDN Controller

SDN controllers are also called network operating systems (NOS), which perform the decisions in routing the packets. These controllers have global surveillance control in the SDN network from the packet flow viewpoint. The controllers can monitor all the network devices within its administrative domain. There are many open source SDN controllers available for use in research, including but not limited to the following: a) Open Daylight, b) ONOS, c) Project Calico, d) the Fast Data Project, e) Project Floodlight, f) Beacon, g) NOX/POX, h) vneio/sdnc, i) Ryu Controller, j) Cherry, k) Faucet, and f) OpenContrail. However, in this work OpenDaylight SDN controller has been adopted as an open source of the SDN architecture because it provides a comprehensive platform on overall design aspects for our framework.

5.3. Network Traffic Generator

Numerous software-based packet generating tools can be used to flood the SDN with random packets or allow the user to construct customized packet exchanges between hosts in the SDN network. Such tools, though not limited, include: AnetTest, Pktgen, IP Sorcery, Pierf, and Scorpy. However, in this work the authors have decided to use Hping3 for being simple and easy to implement. It specializes in the generation, analysis, and transmission of Malicious IP-based packets for DDoS attacks.

5.4. Case Study

The proposed method for detecting and mitigating the DDoS attacks in an SDN environmental setup is implemented by utilizing sFlow and OpenFlow protocols for performance evaluation.

The star topology as shown in Figure 6 has been used. The number of nodes in the SDN network is 13 nodes, there are 9 hosts in this network, $H = 9$, four of these hosts are legitimate users, that is, $U = 4$ and four of them are bots, $B = 4$ and one host is a server. Furthermore, there is also one controller $C = 1$ and four OpenFlow switches, $S = 4$. We will use an OpenDaylight controller (Nitrogen version) and OpenvSwitch to receive/forward commands using OpenFlow protocols. In addition, the legitimate user sends traffic at rate $\lambda = 10$ to 80 Kbps, whereas the bot sends traffic at rate

λ between 1 Mbps and 10 Mbps. Table 1 lists more details about users and traffic rates for generating SDN traffic jam.

Table 1. SDN Traffic details.

Host number	Host type	Traffic rate
1	legitimate user	10Kbps
2	legitimate user	30Kbps
3	legitimate user	20Kbps
4	legitimate user	80Kbps
5	bot	3Mbps
6	bot	4Mbps
7	bot	5Mbps
8	bot	10Mbps

The sFlow agents have been configured for all OpenFlow switches of Figure 6, which are described in Table 1. All the OpenFlow switches have been configured automatically using Python.The malicious traffic is generated by using the following $Hping3$ command:

The top four charts in Figure 7 shows regular traffic that is generated by legitimate users, whereas the high traffic generated by the $Hping3$ command to mimic the behavior of the DDoS attack in the SDN network generated by four Bots hosts as demonstrated in the bottom four charts of the same figure.

Figure 8 illustrates the SDN network traffic of the device with IP (192.168.11.35) with and without the DDoS mitigation algorithm. As shown in Figure 9, the traffic exceeds the pre-set threshold of 1.25Mbps in the beginning, which happens because the detection and mitigation algorithms are not running as the controller had not been enabled. Then, after running the detection and mitigation algorithm, the traffic is kept below the threshold value. The suspension of IP address has been set to last for around 60 seconds. Subsequently, the mitigation algorithm removes the suspension after 60 seconds to re-trigger provided the Bot has been still attacking.

Figure 8 shows the DDoS attack performed by just one Bot (Host number 5) with IP address of 192.168.11.35. The Figure confirms that the proposed detection and mitigation algorithm (sFlow-based) is sufficient and can mitigate a large number of DDoS attacks on SDN networks. Figure 9 shows when the sFlow controller is active shown in blue color or pending shown in orange color.

The results also show that the detection time is around 3 seconds in this setting. The authors believe that the results are not fixed as as they may vary based on the value and on the type of DDoS attacks. In addition, the main advantage of using sFlow-based algorithm is that it does not overwhelm the controller with huge mass of overhead data that can delay the traffic by increasing the latency of packets in SDN network because it sends just a sample of the traffic.

Comparing our initial results with the result proposed in framework [26–28], our system can detect and mitigate any type of DDoS attack, whereas the work in [26] can only detect SYN DDoS attacks. In addition, the work provided by the authors in manuscript [27] can only detect massive traffic whereas our method can detect low traffic DDoS attacks and mitigate them. The authors in [28] have thwarted the DDoS attacks by remapping controller to send access control back to the data plane, causing latency issues. However, in this paper the increase in the network size is not affecting the overhead traffic between the controller and OpenFlow switches adversely by keeping the sampling rate lower to keep the overhead traffic down.

```
Bot name hping3 –flood –tcp -k -s 80 Server name
```

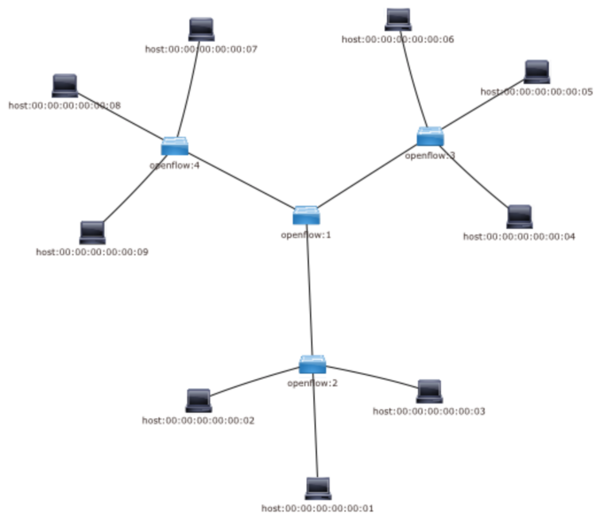

Figure 6. Star topology (OpenDaylight interface).

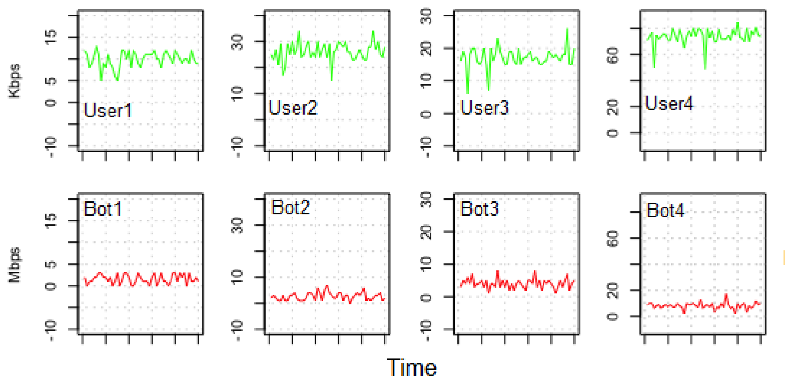

Figure 7. SDN network traffic (Kbps): The top four charts depict the sample of traffic generated by legitimate users, whereas the bottom four charts demonstrate the DDoS attack.

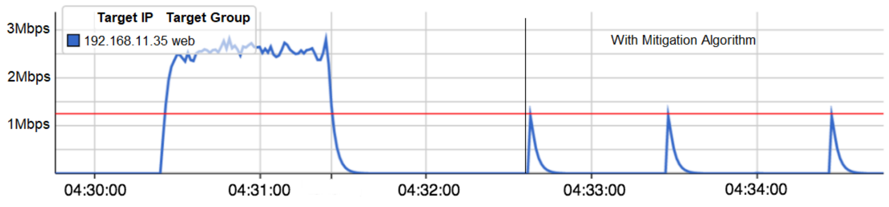

Figure 8. Real-time values of DDoS attack generated by the first Bot that triggers the mitigation algorithm when the threshold value is exceeded.

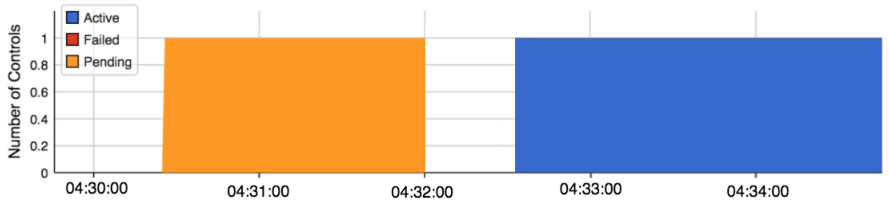

Figure 9. Shows time when Mitigation algorithm controller is Active (in blue), Pending (in yellow).

6. Related Work

Several solutions are currently available for mitigating DDoS attacks, which could be application level attack, control level or volumetric in nature. One popular method used by large enterprises and service providers to protect against the DDoS threats is BGP Remote Triggered Black Hole (RTBH). It instructs routers to block the traffic at the edge before it enters the protected target network, thus educing network overload [29]. Various hardware devices can also be used to mitigate DDoS attacks by filtering all types of attacks to exhaust resources disabling thus several services, and the network performance is accordingly downgraded. SDN technologies such as Open Flow provide multiple methods for controlling routing and switching, making current networks more rich with features such that to modify network resources to the arising requirements [30].

With the development of computer networks, existing network systems and data centers have become increasingly feature-rich, complex and data-intensive, so that system designers often need to modify network software and coordinate computer and network resources according to the specific requirements. However, traditional network architectures are not suitable to meet the above requirements from the viewpoints of enterprises, carriers, and end users. For instance, the decision-making capability of legacy networks is distributed across various network components, making adding any new devices or services to the network a daunting task. [31,32]

SDN plays a crucial role as it helps in classifying incoming requests using different algorithms that help detect to identify incoming requests [33,40]. The operating system uses defense system based on machine learning algorithms to either allow or drop different packets at the switch level [34]. Such algorithms include the use of PATMOS [35], a novel hybrid flow-based handler with a novel protocol that uses a gossip-style approach to identify attacks [36], and an SDN schedule algorithm that ensures that the SDN is made unavailable during some attack [37]. By using different algorithms, the SDN can learn different DDoS techniques and counter attacks before they cause harm to the server.

SDN is useful as it helps in blocking automated attacks using Bot-net [33], and can detect DDoS attacks in an early stage so the DDoS attack falls behind bringing the servers down [38]. Also, SDN networks can function automatically with minimal supervision for handling DDoS attacks such as UDP, HTTP, etc. In essence, an effective method should be utilized to ensure the detection of the DDoS attacks, through proper management of packet traffic in the SDN network, and prevention of a different type of DDoS attacks [39].

The authors in [26] proposes an SDN-based security system for the Internet of Things (IoT)-base networks. The authors have used a combination of three techniques, which are Intrusion detection system called Snort, sampling Flow standard called sFlow, and OpenFlow protocol. They have used Snort for fast detection of a DDoS attacks. Although the authors have not provided details for their system, however the researchers in this work believe the proposed method is complicated, and it may cause packets latency concerns in such small networks.

Framework [28] has suggested a defense system based on sFlow and OpenFlow. Nevertheless, the proposed model is not, and the authors did not include their algorithm. They managed to present some preliminary results to report that their proposed system could detect only massive DDoS attacks. The authors in [29] offered defense system against SYN DDoS attacks using sFlow and OpenFlow. However, the suggested model could not deal with other types of DDoS attacks. Having looked to those mentioned above most of the proposed model can detect and mitigate massive attack only. Also, some of the recommended modules, such as [29] have focused to dedicate to a specific type of DDoS attacks.

7. Conclusion

In conclusion, the findings of this framework have examined and evaluated as a defense system for detection and mitigating DDoS attacks in SDN networks. The preliminary simulation results have confirmed that the proposed defense system can be efficiently used to detect and mitigate DDoS attacks in a matter of few of seconds. Also, the results show that the proposed method does not affect the overhead traffic adversely. Finally, by using the suggested defense system, the scalability would not be serious issues as the network administrator can reduce the sampling rate to keep overhead traffic between the controller and OpenFlow switch down.

8. Future Work

This work is planned to continue into three future directions using ODL controller for mitigating DDoS attacks. Firstly, a multi-level controller (global and local) will be tried to mitigate the DDoS attack and overcome the issue of overhead traffic between the data plane of switches and control plane as a controller, in single controller architecture. Secondly, the authors have planned to investigate running the OpenDaylight on Raspberry Pi 3. The preliminary idea is to write DDoS mitigation script (mitigation algorithm) and implement it on Raspberry Pi-3 device architecture. Finally, the sampling rate is set randomly in the proposed system; the author will try to find ways to select the optimal sampling rate that guarantees fast detection time and minimised packet latency.

Author Contributions: Conceptualization, F.A. and A.M.A.; methodology, F.A.; software, F.A.; validation, F.A., A.M.A. and S.K.; formal analysis, F.A., A.M.A, S.K; investigation, F.A., A.M.A, S.K; resources, A.M.A; data curation, F.A.; writing—original draft preparation, F.A., A.M.A.; writing—review and editing, F.A., A.M.A, S.K.; visualization, F.A.; supervision, F.A., S.K., A.M.A.; project administration, A.M.A.; funding acquisition, Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Acknowledgments: The authors would like to thank Saudi government for supporting the second author financially during his research study for a Ph.D. in computer science. Also, the author acknowledge the support of Muhammad Haris Kaka Khel, PhD Scholar at the Atlantic Technological University Donegal, Ireland to template the paper in Latex and for technical support of functional plotting.

References

1. Khan, Latif U., Zhu Han, Walid Saad, Ekram Hossain, Mohsen Guizani, and Choong Seon Hong. "Digital twin of wireless systems: Overview, taxonomy, challenges, and opportunities." IEEE Communications Surveys and Tutorials (2022).
2. Goransson, Paul, Chuck Black, and Timothy Culver. Software defined networks: a comprehensive approach. Morgan Kaufmann, 2016.
3. Sarwar, Asima, Abdullah M. Alnajim, Safdar Nawaz Khan Marwat, Salman Ahmed, Saleh Alyahya, and Waseem Ullah Khan. "Enhanced anomaly detection system for iot based on improved dynamic SBPSO." Sensors 22, no. 13 (2022): 4926.
4. Dhruba Kumar Bhattacharyya and Jugal Kumar Kalita. 2016. DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance. CRC Press.
5. Harish Barapatre, Jai Deshmukh, Vikrant Kapse, and Pritam Patil. DDoS Spyware for Cloud Computing Environment.
6. Wang, An, Wentao Chang, Songqing Chen, and Aziz Mohaisen. "Delving into internet DDoS attacks by botnets: characterization and analysis." IEEE/ACM Transactions on Networking 26, no. 6 (2018): 2843-2855.
7. Hussain, Mudassar, Nadir Shah, Rashid Amin, Sultan S. Alshamrani, Aziz Alotaibi, and Syed Mohsan Raza. "Software-defined networking: Categories, analysis, and future directions." Sensors 22, no. 15 (2022): 5551.
8. Neelam Dayal and Shashank Srivastava. 2017. Analyzing Behavior of DDoS Attacks to Identify DDoS Detection Features in SDN. In 2017 9th International Conference on Communication Systems and Networks (COMSNETS), pp. 274-281. IEEE. DOI: 10.1109/COMSNETS.2017.7945387.
9. Aladaileh, Mohammad A., Mohammed Anbar, Iznan H. Hasbullah, Yung-Wey Chong, and Yousef K. Sanjalawe. "Detection techniques of distributed denial of service attacks on software-defined networking controller–a review." IEEE Access 8 (2020): 143985-143995.
10. Rawat, Danda B., and Swetha R. Reddy. "Software defined networking architecture, security and energy efficiency: A survey." IEEE Communications Surveys and Tutorials 19, no. 1 (2016): 325-346.
11. Open Network Foundation. (2015, May). OpenFlow Switch Specification (Version 1.5.1). [Online]. Available: https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf.

12. Tr, O. (2015). Principles and Practices for Securing Software-Defined Networks. Open Networking Foundation. Palo Alto CA, USA.
13. Bhattacharyya, D. K., and Kalita, J. K. (2016). DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance. CRC Press.
14. Douligeris, C., and Mitrokotsa, A. (2004). DDoS attacks and defense mechanisms: classification and state-of-the-art. Computer Networks, 44(5), 643-666.
15. Holl, P. (2015). Exploring DDoS Defense Mechanisms. Network, 25.
16. Mousavi, S. M., and St-Hilaire, M. (2015). Early detection of DDoS attacks against SDN controllers. In 2015 International Conference on Computing, Networking and Communications, ICNC 2015 (pp. 77-81). IEEE. doi:10.1109/ICCNC.2015.7069319.
17. Keromytis, A. D., Misra, V., and Rubenstein, D. (2002). Using overlays to improve network security. In Proceedings of SPIE ITCom Conference on Scalability and Traffic Control in IP Networks II (Vol. 2002).
18. Wang, H., Zhang, D., and Shin, K. G. (2002). Detecting SYN flooding attacks. In INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE (Vol. 3, pp. 1530-1539). IEEE.
19. Lersak Limwiwatkul and Arnon Rungsawang, "Distributed denial of service detection using TCP/IP header and traffic measurement analysis," in Communications and Information Technology, 2004. ISCIT 2004. IEEE International Symposium on, vol. 1, pp. 605-610, 2004, doi: 10.1109/ISCIT.2004.1386790.
20. Rodrigues, Bruno, Eder Scheid, Christian Killer, Muriel Franco, and Burkhard Stiller. "Blockchain signaling system (BloSS): cooperative signaling of distributed denial-of-service attacks." Journal of Network and Systems Management 28 (2020): 953-989.
21. Haider, Shahzeb, Adnan Akhunzada, Iqra Mustafa, Tanil Bharat Patel, Amanda Fernandez, Kim-Kwang Raymond Choo, and Javed Iqbal. "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks." Ieee Access 8 (2020): 53972-53983.
22. Doshi, Keval, Yasin Yilmaz, and Suleyman Uludag. "Timely detection and mitigation of stealthy DDoS attacks via IoT networks." IEEE Transactions on Dependable and Secure Computing 18, no. 5 (2021): 2164-2176.
23. Aslam, Muhammad, Dengpan Ye, Aqil Tariq, Muhammad Asad, Muhammad Hanif, David Ndzi, Samia Allaoua Chelloug, Mohamed Abd Elaziz, Mohammed AA Al-Qaness, and Syeda Fizzah Jilani. "Adaptive machine learning based distributed denial-of-services attacks detection and mitigation system for SDN-enabled IoT." Sensors 22, no. 7 (2022): 2697.
24. Alsaeedi, Mohammed, Mohd Murtadha Mohamad, and Anas A. Al-Roubaiey. "Toward adaptive and scalable OpenFlow-SDN flow control: A survey." IEEE Access 7 (2019): 107346-107379.
25. Gupta, Neelam, Mashael S. Maashi, Sarvesh Tanwar, Sumit Badotra, Mohammed Aljebreen, and Salil Bharany. "A comparative study of software defined networking controllers using mininet." Electronics 11, no. 17 (2022): 2715.
26. Nugraha, M., Paramita, I., Musa, A., Choi, D., and Cho, B. (2014). Utilizing OpenFlow and sFlow to Detect and Mitigate SYN Flooding Attack. Journal of Korea Multimedia Society, 17(8), 988-994.
27. Lin, H., and Ping Wang. "Implementation of an SDN-based security defense mechanism against DDoS attacks." In Proceedings of the 2016 Joint International Conference on Economics and Management Engineering (ICEME 2016) and International Conference on Economics and Business Management (EBM 2016). 2016.
28. Wang, Yang, Tao Hu, Guangming Tang, Jichao Xie, and Jie Lu. "SGS: Safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking." IEEE Access 7 (2019): 34699-34710.
29. Wani, Azka, and Rubeena Khaliq. "SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL)." CAAI Transactions on Intelligence Technology 6, no. 3 (2021): 281-290.
30. Varghese, Josy Elsa, and Balachandra Muniyal. "An Efficient IDS framework for DDoS attacks in SDN environment." IEEE Access 9 (2021): 69680-69699.
31. Mousavi, Seyed Mohammad, and Marc St-Hilaire. "Early detection of DDoS attacks against SDN controllers." In 2015 international conference on computing, networking and communications (ICNC), pp. 77-81. IEEE, 2015.

32. Mousavi, Seyed Mohammad, and Marc St-Hilaire. "Early detection of DDoS attacks against SDN controllers." In 2015 international conference on computing, networking and communications (ICNC), pp. 77-81. IEEE, 2015.
33. Lim, Sharon, J. Ha, H. Kim, Y. Kim, and S. Yang. "A SDN-oriented DDoS blocking scheme for botnet-based attacks." In 2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 63-68. IEEE, 2014.
34. Phan, Trung V., Nguyen Khac Bao, and Minho Park. "A novel hybrid flow-based handler with DDoS attacks in software-defined networking." In 2016 Intl IEEE Conferences on Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), pp. 350-357. IEEE, 2016.
35. Eliyan, Lubna Fayez, and Roberto Di Pietro. "DoS and DDoS attacks in Software Defined Networks: A survey of existing solutions and research challenges." Future Generation Computer Systems 122 (2021): 149-171.
36. Macedo, R., de Castro, R., Santos, A., Ghamri-Doudane, Y., and Nogueira, M. (2016). Self-organized SDN controller cluster conformations against DDoS attacks effects. In 2016 IEEE Global Communications Conference, GLOBECOM 2016 - Proceedings (pp. 1-6). IEEE. doi:10.1109/GLOCOM.2016.7842259.
37. Yan, Q., Q. Gong, and F. Richard Yu. "Effective software-defined networking controller scheduling method to mitigate DDoS attacks." Electronics Letters 53, no. 7 (2017): 469-471.
38. Zhang, P., Wang, H., Hu, C., and Lin, C. (2016). On Denial of Service Attacks in Software Defined Networks. IEEE Network, (December), 28-33. doi:10.1109/MNET.2016.1600109NM.
39. Xu, Yang, and Yong Liu. "DDoS attack detection under SDN context." In IEEE INFOCOM 2016-the 35th annual IEEE international conference on computer communications, pp. 1-9. IEEE, 2016.
40. Aladaileh, Mohammad A., Mohammed Anbar, Iznan H. Hasbullah, Yung-Wey Chong, and Yousef K. Sanjalawe. "Detection techniques of distributed denial of service attacks on software-defined networking controller–a review." IEEE Access 8 (2020): 143985-143995. 12
41. Hussain, M., Shah, N., Amin, R., Alshamrani, S.S., Alotaibi, A. and Raza, S.M., 2022. Software-defined networking: Categories, analysis, and future directions. Sensors, 22(15), p.5551.
42. (2018, March). [Online]. Available: https://sflow-rt.com (Accessed: Mar. 2018).
43. Controlling Large Flows with OpenrFlow. [Online]. Available: https://blog.sflow.com/2013/05/controlling-large-flows-with-openflow.html: https://blog.sflow.com/2013/05/controlling-large-flows-with-openflow.html (Accessed: May. 2018).