Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Review

New-flow based DDoS attacks in SDN: Taxonomy, rationales, and research challenges

Maninder Pal Singh [*], Abhinav Bhandari

Department of Computer Science and Engineering, Punjabi University, Patiala, India

ARTICLE INFO

Keywords:
Software-defined networking
SDN
DDoS
New-flow
OpenFlow
SDN security
DDoS detection and, mitigation

ABSTRACT

Software-defined Networking (SDN) is an emerging technology that revolves around the fundamental notion of setting up a network with a decoupled control plane and data plane. It brings numerous benefits such as improved network manageability, flexibility, and programmability to measure up to the enormous demands of future networking. However, it also comes with security concerns that are intrinsically present in SDN's architecture. In classic SDN, a switch acts as a forwarding device that has to forward a packet towards the centralized controller for every new flow that comes into the network. Out of this design philosophy, a new-flow based distributed denial-of-service (DDoS) attack is born, which presents this internal flow-based policy as a critical security vulnerability to confiscate the scarce resources of the control plane and data plane in an SDN network. In this paper, we propose a classification of such security vulnerabilities exposed by SDN architecture and leveraged by a new-flow based DDoS attack. We also provide an analysis of the latest developments made in recent years on DDoS detection and mitigation research works to overcome these security vulnerabilities. Finally, we discuss SDN security-related research challenges that can be valuable for the research community and academics for carrying out further research and investigation.

Contents

[*] Corresponding author.
E-mail address: manips88@yahoo.com (M.P. Singh).

https://doi.org/10.1016/j.comcom.2020.02.085
Received 11 October 2019; Received in revised form 26 January 2020; Accepted 29 February 2020
Available online 7 March 2020

1. Introduction

The expeditious growth of the number of network devices in traditional networking infrastructure promotes complexity in management and imposes innovation barriers to the future Internet. In traditional networks, the control logic and forwarding capability of a network device are tightly coupled as shown in Fig. 1. This design reduces flexibility, impedes innovation and in addition involves higher operational costs. As a result, halting the momentum of next-generation emerging technologies such as IoT, Big data and Cloud, etc. that increasingly demand more bandwidth, adaptability, and better manageability.

Software-defined Networking (SDN) has emerged as a revolutionary networking paradigm that is able to meet these escalating demands of future networking. The core aspect of SDN architecture in contrast with traditional networking architecture is that the control plane is separated from the data plane. As shown in Fig. 2, a simplified SDN's architecture typically consists of three layers or planes: data plane, control plane, and application plane. The data plane is comprised of network switches acting as forwarding devices. The control plane contains at least one software controller that is logically centralized to manage the configurations and behavior of these forwarding devices in the network. Lastly, the application plane is where controller applications reside. Using a vendor-agnostic interface, e.g., OpenFlow [1] – between the network switches and controller – SDN provides the ability to install unforeseen experimental features and protocols. Moreover, it yields a global view of the network which facilitates the research communities and network administrators, etc. to actively monitor and reconfigure their networks. Therefore, this centralized system enables the network to be flexible, cost-effective, programmable to promote innovation, being an ideal solution for high bandwidth on demand, and dramatically simplifies network management.

SDN is often associated with OpenFlow for providing a centralized and global view of the network. The OpenFlow is a widely deployed and de-facto protocol, mainly responsible for the communication between forwarding devices such as OpenFlow switches and the software-based controller in the SDN's architecture [1]. An OpenFlow switch consists of at least one flow table that has a set of flow entries. Flow entries are made up of matching rules, counters, and action fields. Based on the match rules of a particular flow, the specified actions are applied, and counters record the information. OpenFlow-based switches also reduce operational costs; in addition, their behavior can be configured to function as a router, switch, or firewall as instructed by the corresponding application installed on the controller.

Despite these standout benefits, there are some essential security challenges on which only a few research efforts have been made over the last several years. The centralized nature of SDN is revealed to be a single point of failure that can be leveraged by one of the Internet's most brutal and dominant security threat known as Distributed Denial of Service (DDoS) Attack. A DDoS attack is a distributed and coordinated attack that originates from multiple network sources. Fundamentally, the strategy of this attack is to send a sheer volume of spoofed IP packets from disparate sources in order to make the network resources unavailable to legitimate users.

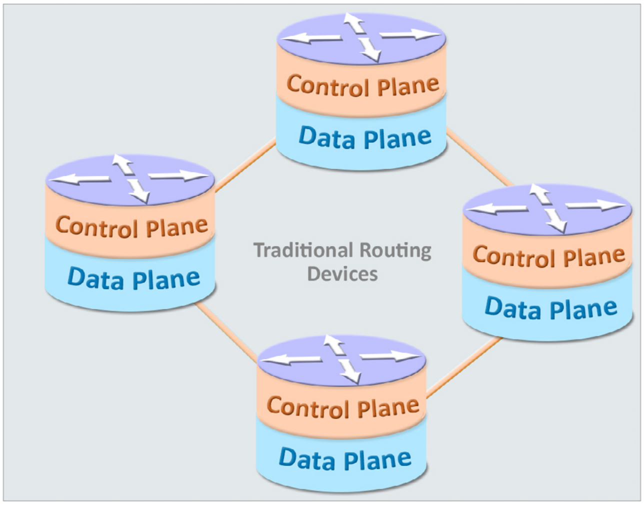

Fig. 1. Traditional networking devices with an embedded control plane and data plane.

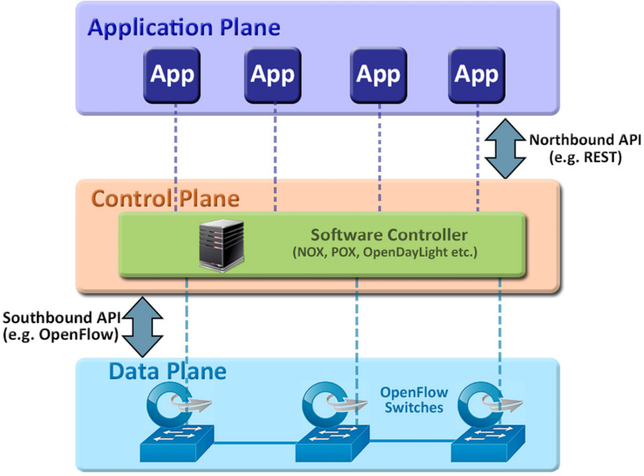
Application Plane
App
App
App
App
Northbound API (e.g. REST)
Control Plane
Software Controller (NOX, POX, OpenDayLight etc.)
Southbound API (e.g. OpenFlow)
Data Plane
OpenFlow Switches

Fig. 2. Software-defined Networking (SDN) architecture showing the decoupled data plane and control plane along with the application plane.

Over recent years, the attackers have got smarter and have been continuously improving and using advanced DDoS attack techniques to inflict more economical and financial damages. For example, a recently reported record-breaking DDoS attack [2] is said to have misconfigured thousands of Memcached servers to launch a massive attack comprising the bandwidth of 1.7 Tbps using reflection/amplification-based attack

Security Issues In SDN
Intrinsic (Security for SDN)
Extrinsic (Security By SDN)

Fig. 3. Classification of Security issues in SDN.

techniques. According to 2017's Arbor's security survey report [3], the DDoS attacks have increased exponentially in size in recent years and have been causing a tremendous amount of damage to the enterprise networks and data centers. Furthermore, 57% of enterprise, government, and education (EGE) respondents encountered the attack and suffered from the saturation of Internet bandwidth indicating an increase of 15% compared to the previous year. The same report also shows that due to the rapid rise of IoT bots, DDoS attackers have been able to devise new ways to launch more devastating attacks. These attacks are capable of causing further significant damage to the largest enterprises, cyber–physical systems [4], data centers, fog computing [5] and service providers where the SDN technology has just begun to sprout and take its shape.

While in [6–9] research has shown that SDN's ability to have a global authority over OpenFlow devices can be utilized for defending against these attacks, but a major concerning fact is that due to the separation of the control plane from the data plane, SDN also needs security for itself. Fig. 3 classifies these security issues as intrinsic and extrinsic. Under intrinsic SDN security, there exist multiple open attack vectors and threats in SDN architecture which can be potentially vulnerable to DDoS attacks. A DDoS attacker, for example, can easily target OpenFlow switches in the infrastructure layer, and as well as a controller in the control layer of SDN architecture. Forwarding devices such as switches have scarce memory resources for flow tables and minimal control channel throughput compared to the data plane channels between the other switches [10]. Therefore, a DDoS attacker can easily exploit this fact and overload their storage and processing capabilities by sending a stream of unmatched flows. The switch acknowledges these unmatched flows as new-flows, and by default, they are always forwarded towards the controller. By exploiting this design protocol, attackers can aim to exhaust the controller to result in catastrophic effects such as saturation at the southbound channel, which ultimately leads to crashing down the entire network.

Since SDN offers a modern networking paradigm and revolutionary architecture, we believe that it needs first and foremost protection against these attacks before it can be reliably deployed in data centers and cloud computing environments to provide its own security benefits. Therefore, our survey falls under the group of intrinsic security issues.

In Table 1, we provide a comparison of our work with other related surveys published in recent years that closely match our article. We have chosen several parameters for the basis of comparison, as shown in the table. Some existing studies have generally focused on overall security issues regarding SDN. However, in this article, we focus on the security issues of SDN architecture uncovered and exploited by new-flow attacks. To the best of our knowledge, it is the first survey that provides a taxonomy of security vulnerabilities exposed by new-flow based DDoS attack.

Our main contributions are summarized as follows:

- To propose a taxonomy of various kinds of security vulnerabilities exposed by a new-flow based DDoS attack targeting the different planes and modules of SDN architecture.
- To provide a state-of-the-art review of defense solutions along with rationales used for detection against DDoS attacks on the SDN architecture.
- To discuss some of the essential open research issues and challenges related to security in SDN faced by researchers.

The rest of this paper is structured as follows: Section 2 discusses the background on OpenFlow Forwarding; Section 3 presents a taxonomy of new-flow based DDoS attacks exploiting the vulnerabilities in SDN. The strategies of DDoS detection and mitigation related to SDN's Security are given in Section 4. Section 5 highlights open issues and research challenges associated with SDN's security. Finally, Section 6 concludes the paper with a discussion on future directions.

2. Background on openflow forwarding

The underlying network communication in a software-defined network is supported by OpenFlow protocol which allows switches to communicate directly with the controller. OpenFlow protocol initially began as a research project at Stanford University in 2008 with its deployment in campus area networks. The goal of OpenFlow was to break the limitations posed by traditional network systems by providing the flexibility of control over multiple switches through a controller software, ease of network configuration management, and innovation of new applications and protocols through programmability. It was adopted by ONF (Open Networking Foundation) in 2011, a non-profit organization founded and funded by Deutsche Telekom, Facebook, Google, Microsoft, Verizon, and Yahoo! [16]. Presently in 2018, there are over 150 companies that are a member of the Open Networking Foundation [17]. Several standards of OpenFlow protocol specifications have been released to the date starting from version 1.0 to version 1.5. Since its first standardization, OpenFlow protocol has considerably evolved and gained unprecedented momentum over the years resulting in commercialization of a wide range of OpenFlow devices. Furthermore, its evolution has increased the number of applications and uses cases such as Cloud-enabled data centers, enterprise networks, ISPs, and wireless networks, etc.

The OpenFlow specification [18] allows an OpenFlow-enabled switch to operate in two modes for the installation of forwarding rules. These are known as proactive mode and reactive mode. Under the proactive mode, the flow rules are inserted into a switch before the network traffic arrives and under the reactive mode, a flow rule is inserted into a switch after the traffic arrives at the switch. For installing flows reactively, the controller has to be involved, and the flow entries in the switch will populate dynamically as the packets arrive. This process is termed as "OpenFlow Forwarding". Fig. 4 illustrates this process with a sequence of steps:

1. To send the packet to the destination host 'B,' source host 'A' has to forward it to the switch (S1) that is connected via Port 1.
2. Upon receiving the packet, the switch performs a lookup in its flow table to find a match for the packet. If the packet has no matching flow entry in the switch (S1) then by the default behavior of pre-defined OpenFlow rules, the switch forwards it to the controller as an OF_PACKET_IN message via OpenFlow (Southbound) channel.
3. The controller receives the OF_PACKET_IN and after calculating the optimal forwarding route, it sends it back to switch (S1) as OF_PACKET_OUT message.
4. The switch (S1) receives the packet and updates its flow table with a flow entry and forwards it to the next node.
5. Next, the packet gets received by the switch (S2). It also performs a lookup in its flow table, and upon not finding a matching entry, it forwards it to the controller as OF_PACKET_IN message.
6. Upon receiving the packet, the controller calculates the next forwarding route and replies to the switch (S2) with OF_PACKET_OUT message. The message instructs the switch (S2) to forward it along the path towards the destination host.

Table 1
Comparison of our work with other related surveys.

Reference	Year	New-Flow based DDoS taxonomy	Classification of security issues in SDN		Illustration of DDoS attack vectors on SDN layers			Classification of DDoS attack detection methods	Tables with rationales for detection	Issues & challenges
			By SDN	For SDN	None	Partial	Full			
[11]	2016	✗	✓	✓	✗	✗	✓	✓	✗	✓
[12]	2016	✗	✓	✓	✗	✗	✓	✓	✗	✗
[13]	2017	✗	✓	✗	✗	✗	✗	✓	✗	✓
[14]	2017	✗	✗	✓	✗	✓	✗	✓	✗	✗
[15]	2018	✗	✗	✓	✗	✓	✗	✓	✗	✗
Our Survey	2019	✓	✗	✓	✗	✗	✓	✓	✓	✓

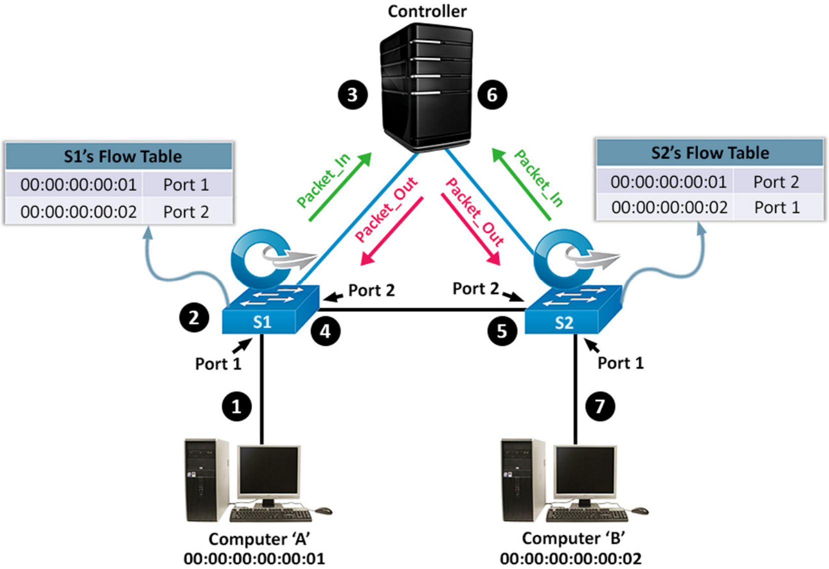

Fig. 4. OpenFlow Forwarding.

7. The destination host 'B' finally receives the packet.
8. For the next the packet delivery from host 'A' to 'B,' the packet does not need to consult the controller unless the flow entries in the switches get timed-out and expire or the topological changes occur in the network [19].

While using proactive approach requires switches to be filled with a large number of flows ahead of the arrival of network traffic but with the reactive approach, it is not the case and it allows dynamic control over the network. However, installing flow rules reactively also comes with a specific drawback which makes network switches vulnerable to DDoS attacks [20]. For example, a new-flow based DDoS attack can easily exploit the reactive flow installation mechanism by causing a surge of spoofed OF_PACKET_IN flows sent towards the controller from the switch. As a result, the OpenFlow channel between the switch and the controller will become congested, and the services will be denied to the legitimate users.

3. Classification of new-flow based DDoS attacks on SDN architecture

This section proposes a taxonomy for a new-flow based DDoS attack on SDN architecture and shows how various components of SDN architecture are vulnerable to these attacks. Fig. 5 shows the taxonomy classified by switch vulnerabilities, classification by attack type on different SDN modules, classification by the impact of the attack and by attack strength.

3.1. Classification by switch vulnerabilities

In this section, the attacks are classified according to the specific vulnerabilities found in the victim modules of an OpenFlow switch. The vulnerabilities exposed by the switch comprise of OFA (Open Flow Agent) overloading, manipulating the duration of flow entries, packet buffer overflow, and flow table overloading. All these vulnerabilities are greatly interlinked with each other since a new-flow based DDoS attack can target and exploit more than one these at the same time.

3.1.1. OFA overloading

OFA (Open Flow Agent) is a software agent inside the switch [10, 21] which enables communication between the switch and the controller over a secure connection. In reactive mode, whenever a new packet arrives at the switch, OFA performs a lookup in the flow table to check if there is a match for an incoming packet in the flow table. If there is no match found then OFA encapsulates it into a *Packet_In* message before forwarding the packet to the controller via a secure channel. The controller processes the packet and upon computing an optimal path, sends it back to the switch via the same control path as a *Packet_Out* message. OFA will receive this packet, and it will also install a new-flow rule into the flow table based on the reply from the controller. In short, OFA serves as the intermediary between the switch and controller and allows the behavior of a switch to be controlled by a controller.

In contrast with OpenvSwitches [22], physical switches generally have a low-powered CPU [23] compared to the controller because of their cost and limited capabilities. Due to this limitation, OFA can only process and forward a limited number of packets. A DDoS attacker sees this limitation as a vulnerability to exploit, and if it sends *Packet_In* messages at an increasing rate than OFA can forward to the controller, then it can cause overloading of OFA module.

Wang et al. [10] performed an experiment to evaluate the impact a DDoS attack can inflict on the packet forwarding abilities of a switch. For each of their three trials, they used a different switch. They

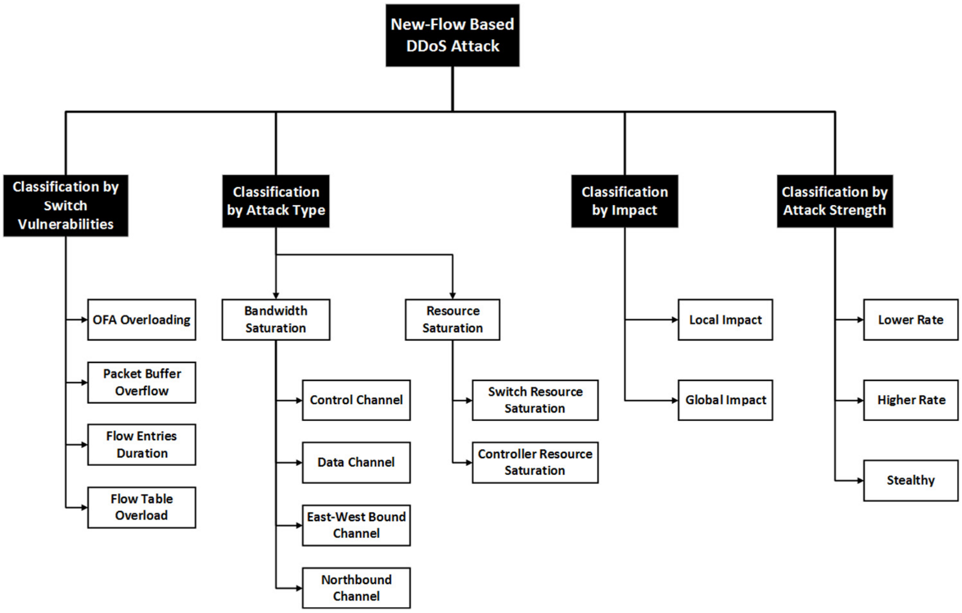

Fig. 5. Taxonomy of SDN vulnerabilities exposed by new-flow based DDoS Attack.

observed that in the case of all three switches, as the attack rate was increased, the legitimate client would start to lose connectivity with the server in terms of flow-rate failure. The experiments concluded that OFA is a bottleneck that resulted in slowing down the overall performance of the switch during the flooding attack. Therefore, the impact of this attack will be local to the switch, and thus, it will also be experienced by the clients connected to the switch.

3.1.2. Packet buffer overflow

Packet buffering mechanism is yet another module of the switch that a new-flow based DDoS can exploit. Typically, when a switch first receives a new packet, it buffers the packet and then forwards only the header of the packet to the controller using a *Packet_In* message. However, under DDoS attack, if this packet buffer becomes full, then *max_len* field in *ofp_action_output* is set to OFPCML_NO_BUFFER which indicates that now a complete packet must be forwarded to the controller [18]. As a consequence, a high number of packets can extensively start to use control channel bandwidth and controller's resources, resulting in increased latency and response time from the switch and as well as causing loss of packets for end-users who are connected to the switch [24]. According to a recent study [25] in 2018, the size of packet buffer plays a vital role in providing an optimal level of performance and response time.

The impact of this attack can be local to the switch in some cases since the switch will neither be able to install any new-flow entries nor direct traffic with new-flows towards the controller. However, since this attack also causes heavy congestion at the control plane [26,27], it can potentially leverage the switch's vulnerability to inflict damage to a controller and disconnect it altogether from switches. Thus, the network can experience a global impact due to this exploited vulnerability.

3.1.3. Flow entries duration

Every flow table in an OpenFlow switch supports a timeout mechanism for flow entries. A timeout defines the lifetime i.e. the duration of a flow entry in the switch. There are two kinds of timeouts associated with flow entries: *idle_timeout* and *hard_timeout*. When *idle_timeout* is non-zero, the flow entry expires after the specified *idle_timeout* value if no traffic is received. When *hard_timeout* is non-zero, the flow entry must expire after the specified *hard_timeout* value independent of whether the packets on flow entry arrived or not. Based on this, a novel stealthy DDoS attack that exploits the lifetime of flow entries was explored by [28]. From their analysis of flow table behavior under a flooding attack, the authors devised a DDoS attack which continuously sends attack flows with minimal duration. They validated their findings through simulation experiments that the attack can go unnoticed and remain undetected by some traditional detection mechanisms. It is also able to imitate the behavior of flash crowds and inflict long-term financial damages. Furthermore, the impact of this attack will be local to the switch since it exploits the timeout mechanism within a switch while staying under the radar.

3.1.4. Flow table overload

As already mentioned previously, the limited size of the flow table attracts the attention of potential DDoS attackers, and it also makes switch an easy target compared to the controller. Moreover, DDoS attackers need fewer resources to overload the switch's flow table compared to attack resources required to exhausting the controller in a network. A new-flow based DDoS attack on a flow table in SDN aims to fill up and exhaust its limited capacity which results in disruption of switch's operation to potentially damage the network services. According to the OpenFlow Specification, when there is not enough space in the flow table for new entries, the switch sends *ofp_error_msg* of type OFPET_FLOW_MOD_FAILED containing the error code OFPFMFC_TABLE_FULL [18]. In the worst case, as soon as legitimate flow entries are dropped from the flow table due to the time-out mechanism, the attack flow entries take up their place resulting in damage to network resources and services. Similar to other attacks on a switch, this attack has a local impact since it overloads the flow table of a particular victim switch in an SDN network. The research studies in references [29–31] further explore and present the behavior of flow tables under new-flow attacks.

In [29], a recent study demonstrates an experiment to prove the existence of flow table overload attacks. The authors used a typical commodity switch with less than a thousand entries and observed that as attack rate increases, the time required to overload the flow table decreases. In another related research [30], analysis of flow table overflow attack has been done from two perspectives in which the attacker may be present inside the network and outside the network. The researchers have shown the impact of a flooding attack on the flow table which results in degradation of performance of the network in terms of packet loss and bandwidth consumption.

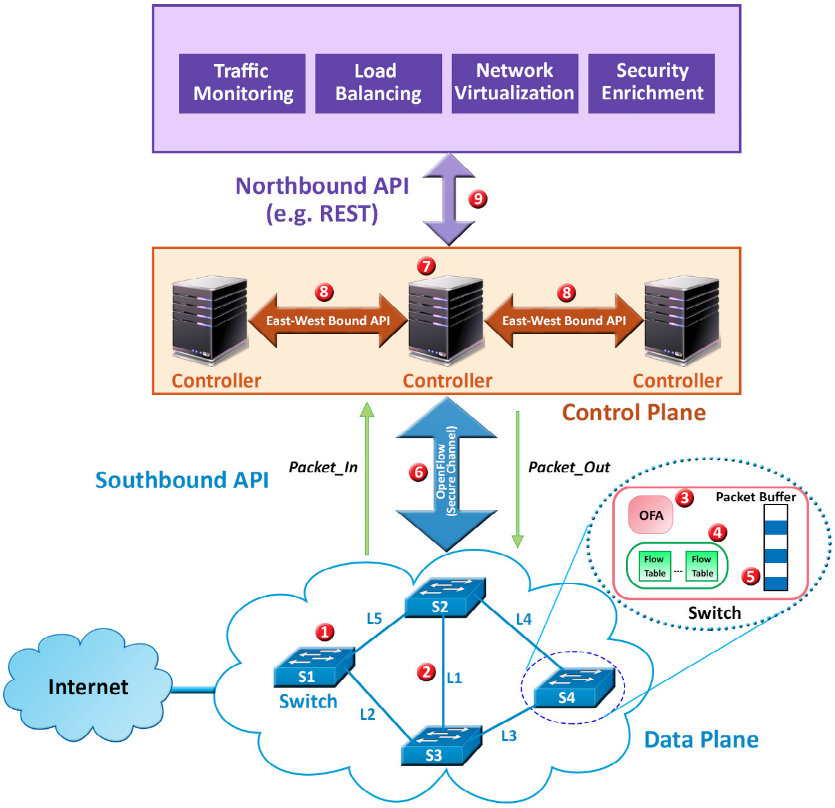

Fig. 6. New-flow based DDoS threat vectors in SDN. (1) Attack on a switch in data plane (2) Attack on data channel between switches (3) Attack on OFA of a switch (4) Attack on flow table of a switch (5) Attack on packet buffer of a switch (6) Attack on OpenFlow interface (7) Attack on SDN controller (8) Attack on East–West Bound Interface (9) Attack on Northbound Interface.

Another research in 2017 [31] referred to this attack as a "TCAM exhaustion attack''. It explores a novel variant of this attack known as (Slow-TCAM) Slow TCAM Exhaustion Attack. When compared to the TCAM exhaustion attack that attempts to quickly occupy the space in the flow table by sending a vast flood of UDP packets, this new attack (Slow-TCAM) works its way rather slowly with the same motive to fill up the flow table and disrupt the services of legitimate users.

3.2. Classification by attack type

In this section, a new-flow based DDoS attack is classified as mainly into two types: Bandwidth saturation attack and resource saturation attack. Bandwidth saturation attack aims to victimize the various interfaces under SDN architecture by sending a large volume of spoofed packet floods and thereby consuming their channel's bandwidth capacity. It is known to target the control channel (southbound API), data channel, east–west bound channel, and northbound channel of an SDN architecture. On the other hand, the resource saturation attack consumes the resources (e.g., CPU, Memory) of SDN network devices such as switches and controllers. The resource saturation attacks will be discussed first in the Section 3.2.1 and 3.2.2 and then the bandwidth saturation attacks which will be discussed from Section 3.2.3 to 3.2.6.

3.2.1. Switch's resource saturation

In classic SDN, switches are simply forwarding devices. They communicate with the controller using a secure OpenFlow channel. A commercial OpenFlow switch available in the market typically supports from a few hundred to thousand flow entries [32]. It is provided with TCAM (Ternary Content-Addressable Memories) to support optimal performance. TCAM, however, is expensive and consumes a considerable amount of power. Therefore, it is only available for a limited amount in a switch to reduce costs.

Furthermore, the processing capabilities of a switch are also a limitation when compared with the number of flow requests per second the controllers can handle [33]. In one of SDN's most current applications such as in data centers, scalability is a notable issue where flow rates can arrive toward a switch at around 72K per second [34]. For these reasons, switches easily become a target of DDoS attackers.

As already mentioned and illustrated in Section 2, one of the ways in which a DDoS attacker can leverage the limitations of storage capacity is by exploiting the mechanism of reactive rule installation of flow entries. Fig. 6 shows an outline of the OpenFlow switch components and highlights the associated threat vectors (numbered as 3, 4, 5).

3.2.2. Controller's resource saturation

A controller is a necessary and crucial component in an SDN network. Fig. 6 shows the placement of the controller in an SDN network where it logically connects with switches through a secure channel. It is a single most network authority responsible for directing and managing flows, enforcing network configuration and installing forwarding rules for all the devices. The controller in SDN also brings many benefits such as providing a global view of a network, dynamic network control, and device management, which can contribute to implementing security measures. However, despite these advantages, centralized controllers are seen as a single point of failure. They can easily be recognized as central points in the network for attackers to exploit.

New-flow based DDoS attack can cause the most significant amount of havoc on an SDN network at a global scale by charging towards the controller with a resource saturation attack [35–38]. This attack accomplishes its goal by exploiting the inherent weakness in SDN's classic design similar to the bandwidth saturation attack on the control channel. Since every new packet has to arrive at the controller, a spoofed flooding attack occurring at a high enough rate can ultimately waste the controller's system resources such as its processing capabilities (CPU), physical memory (RAM), etc. into handling these malicious packets [39]. Even though the attack may require a considerable amount of attack resources compared to the attack on the switch, but at the same time, the effectiveness of this attack on the controller is increased once the packet buffer of the switch becomes full.

Furthermore, having a backup controller will not be enough against this attack after the primary controller's resources are exhausted because that controller would also need to confront the same offense and potentially meet the similar consequences [37]. In [38] Dong et al. studied the effect of a novel variant of this attack that involves the attackers to deplete the resources of a controller by generating low-traffic flows. In this case, the incoming packets will be distributed equally among the flows making their detection by IDS systems even more challenging.

Once the controller is under one of these discussed attacks, its impact can affect the whole network causing high latency and response time with a complete degradation and unavailability of network services for legitimate users. Scalability is also another primary concern regarding controllers [40] which must be resolved effectively to overcome their processing limitations such that they will not be an easy target for DDoS attackers. For these reasons, the controller is perceived as one of the most vulnerable and attractive targets for attackers.

3.2.3. Control channel saturation

The Southbound API, also known as the "control channel" connects the switches with the controller, as shown in Fig. 6. It provides a secure interface between the switches and controller via OpenFlow protocol. It is crucial that the controller should remain connected with the switches in the network all the time to provide them with decision making for routing and controlling the network traffic. Since the controller manipulates the switches through a control channel, a new-flow based DDoS attack on the controller is seen as one of the most pivotal [41–44] drawbacks which lies in the behavior of OpenFlow protocol. It reveals an inherent critical vulnerability in its design and, that is, for every packet which arrives at the ingress port of an OpenFlow switch, the switch allocates memory in the flow buffer. When the packet fails to find a match rule in the flow table, then the switch has to forward it to the controller to get feedback on how to handle this packet. The classic design of SDN follows this ingrained rule in which switches itself are considered to be dumb devices, and it applies to every new packet the switch receives.

Malicious users can easily exploit this scalability drawback of the OpenFlow between the data plane and control plane by launching a new-flow based Distributed Denial of Service attack. It is also traditionally known as a bandwidth saturation attack against the target network accomplished by sending a surge of spoofed IP address attack packets—this kind of attack results in the saturation of OpenFlow protocol link between the switch and the controller under SDN context. In the worst-case scenario, if the switch's packet buffer is full, then it forwards the entire packet to the controller which can result in extensive utilization of the switch–controller interface. Furthermore, It can potentially cause the controller to become unavailable to underlying switches and network hosts. In the worst-case scenario, if the DDoS attack's strength is huge compared to the victim's resources, it can halt the communication of an entire centralized network and render it unusable.

3.2.4. Data channel saturation

The data channel is the communication link between at least two OpenFlow switches. It forwards the network traffic between switches in order for it to reach towards a destination network device such as a controller, a server, or any host connected to a switch. Unlike the control channel, the data channel of a commodity hardware switch has a higher capacity, and it also remains comparatively under-utilized [45].

Despite this, and the fact that the performance impact caused by a new-flow DDoS attack on the data channel is still an insufficiently researched domain, there are multiple ways in which a new-flow DDoS attack can choose to target this channel. (1) When a new-flow attack enters the victim's network, it may take different paths through switches to make its way towards the controller. As a result, some data channels or links between the switches will be occupied by the malicious packets, and they will be more or less congested than the other. (2) If multiple attacks are targeting different switches in a coordinated DDoS attack, then links between the switches will become inaccessible. For example, consider the switches S2 and S3 in Fig. 6. If the attackers under these switches carry out a coordinated attack, they can potentially take out the direct link L1 between these switches.

The impact of this attack will be local to the network switches, data channels, and it will further be experienced by the legitimate clients connected to the switches as well. If the bottleneck of the data channel becomes excessively more significant, the switches may not be able to forward any packets to the controller at all. This scenario could lead to a potential breakdown of an entire network depending on the volume of attack traffic and the number of switches in the victim's network. Furthermore, since there is no secure channel protocol involved at the forwarding plane, it causes trust issues between the switches [46] and leaves an open window for any unencrypted and sensitive information present in network traffic to be easily confiscated by the attackers.

3.2.5. East–west bound channel saturation

In a distributed SDN architecture, the east–west bound APIs act as interfaces between multiple controllers. Multi-controller setups in SDN are used to scale the network size when a single centralized controller is unable to handle the network traffic due to an increase in the number of network switches. It is a challenging task for the distributed SDN controllers to coordinate and synchronize information with each other through east–west bound APIs to maintain global visibility of the entire network [47].

Under a new-flow DDoS attack, such controllers may act as a backup to support the network to avoid a single point of failure. The new flows can travel in either direction towards the additional backup controllers causing the east–west bound interfaces to be saturated with a massive number of incoming attack packets. As a result, the controllers might malfunction, and information exchange between them could be interrupted. It should also be noted that like the northbound API, east–west bound API also has no standardized interface, and each controller is responsible for enforcing its own policies and routing protocols [48]. This limitation further narrows the interoperability domain between different SDN controllers and makes them insecure against this attack.

3.2.6. Northbound channel saturation

SDN promotes innovation and programmability through northbound APIs which connects the controller to the programmable applications running over the network. It allows software developers to build their applications to make the network programmable. Unlike the Southbound API, each platform has its own proprietary and open-source northbound APIs—that is, there is no commonly adopted standard [49,50]. This absence of standardization is believed to expose susceptibilities to several security threats stemming from the lack of trust issues, application permissions, and unauthorized application access.

Due to these security issues, a new-flow attack directed towards the controller can potentially cause congestion at the northbound interface. For example, if a traffic monitoring application listening to *Packet_In* events arriving from switch towards the controller suddenly bombards the network at a high rate, then this leads to the occupation of bandwidth of northbound channel caused by a new-flow based DDoS attack. Moreover, a malicious application can also let the attacker gain unauthorized access and cause havoc by uncontrollably sending back the requests to the controller to enlarge the volume of the attack. The attack's impact could detach the controller from the applications and in the worst-case scenario, bringing the entire network down. The fact that security mechanisms are not standardized increases the risk of malicious threats and as well as reduces the integrity and trust between controller and applications.

3.3. Classification by impact

This section classifies the attacks based on the impact of attacks on the various victim modules. The consequences of an attack can be either local or global to the victim.

3.3.1. Local impact

The impact of the DDoS attack is considered local if the attack does not cause malfunctioning of the entire network. Thus, it will be experienced locally by the forwarding devices in the data plane. It also implies that only those hosts which are directly or indirectly connected to the switches will be affected. As a result, this type of attack could go unnoticed and unrepaired for a prolonged period of time since merely a portion of users will be affected. In the long run, it can potentially inflict a huge amount of financial and reputational damages to the organization given that legitimate users will eventually switch and change their service provider.

3.3.2. Global impact

The impact of this DDoS attack is considered global if it can cause the failure of an entire network. The strength of the DDoS attack is much higher in the control plane than in the data plane since the control plane is a driving portion of the network and attack packets travel towards the controller along this path. Consequently, it results in total disruption of services to the end-users.

3.4. Classification by attack strength

In this section, the attacks are classified based on their strength, i.e. the rate of transmission of attack packets sent towards the victim. Depending on the attack's method, it can be launched at a higher rate, lower rate or it can be a stealthy attack.

3.4.1. Higher rate

A DDoS attack with higher attack strength than the victim's network is able to disrupt its services by sending a surge of spoofed IP packets. Since all the new packets will be forwarded towards the controller from the switch, the attack will be able to damage the system resources of the switch while congesting the control channel at the same time. Therefore, it puts heavy processing load the controller, destroys the legitimate user bandwidth and de-link the switches from the controller causing the services of the entire network to fall apart.

3.4.2. Lower rate

A new-flow based DDoS attack with lower attack strength than the target network will degrade its quality of services (QoS) provided to the end-users. It can unobtrusively hijack the network resources including servers and consume more than a half or a substantial portion of the bandwidth allocated to the legitimate users to result in much lower response times, latency, and degrade performance.

These types of attacks are often very challenging to detect. For example, the attacks launched using mobile botnets [51] have a considerably lower attack rate. They can consume a portion of a network's performance for legitimate users and remain unidentified and active for a long time.

3.4.3. Stealthy attack

As studied by [28] a DDoS attack based on new flows can exploit the switch's underlying mechanism and impact its operations while staying stealthy and undetected. To launch this attack, the attacker can potentially send malicious attack flows lasting in the flow table for a minimum *idle_timeout* value. Thereby the lifetime of these flow entries in the switch immediately expires before they are even detected by some traditional defense mechanisms configured with pre-defined detection thresholds. Since the attack rate is minimized to the point that it is able to go on unnoticed without demonstrating any signs of flow table overflow attack or controller overload, the authors have declared it as a stealthy attack.

The performance impact of this attack was evaluated with the conclusion that it will potentially be able to cause long-term financial loss to the systems. The main reason behind it can both be stealthy and still inflict damage is that the malicious flow entries will occupy some amount of switch's scare memory resources, and short bursts of *Packet_In* events will be generated towards the control plane.

4. Strategies of DDoS detection and mitigation for SDN's security

Distributed denial-of-service (DDoS) attacks are a severe threat to SDN's security. The dynamic nature of a large number of spoofed packets coming from disparate sources poses a tricky challenge to distinguish them from legitimate packets. While SDN's capabilities can be utilized to defend against these attacks, but first of all, it is crucial that the centralized composition of SDN must itself be shielded from the DDoS attacks. Researchers have formulated a number of techniques for this purpose which are based on statistical analysis, information entropy and machine learning methods. In this section, we review the most recent and some popular DDoS detection and mitigation solutions used for SDN security.

In Tables 2–4, we present a classification of the DDoS defense strategies which have been proposed in the last few years and highlight their rationales to provide a clear understanding of the basis on which the anomaly behavior in the network traffic is detected. By doing this, it will provide the researchers with an insight into understanding the attack behavior and methods used by DDoS attackers. Moreover, we also classify the DDoS mechanisms by packet-based and flow-based which are the two different ways to inspect and examine the network traffic. Under packet-based inspection, the network packet features such as header and payload are investigated. On the other hand, in flow-based inspection, the aggregated flow statistics are collected from the OpenFlow switches. A flow is a combination of packet field values representing a connection between source and destination device. Using packet-based inspection, it is possible to deeply analyze its data and discard the malicious packets. Furthermore, in order to make connections with our proposed classification of DDoS attacks, we further include the type of vulnerabilities resolved by each of these DDoS defense solutions.

4.1. Statistical-analysis based DDoS defense for SDN

Statistical-analysis based DDoS defense methods detect a DDoS attack based on the nominal profile data collected during an attack-free period. To achieve attack detection, if an incoming attack packet does not comply with the behavioral profile of normal traffic, the DDoS defense mechanism marks it as malicious and sets off the alarm. For mitigating the effects of an attack, it typically either throttles the bandwidth [52] for malicious packets or completely drops them. Each of these mitigations strategies have their pros and cons. The authors in [53] propose one such statistical-based approach using the IP filtering technique which utilizes the features of network traffic to prevent DDoS attack in order to protect the controller in an SDN network. For this approach to work, the authors have implemented a table and placed it inside the controller to keep track of the source IP addresses of incoming packets. For each IP address, a counter is assigned and incremented to count the number of connections it has made. Based on their survey, they have observed two characteristics to identify a DDoS attack. First, If the number of connections from an IP address is less than k, then no action is required, and the packet is assumed to be a DDoS attack packet. Second, If the counter n representing the number of packets transmitted per connection is less than five, then the packet is classified as malicious, and a drop rule is transferred to the switch for corresponding flow entries. To mitigate the attack, the controller running this defense algorithm classifies the packets and tells the switches what to do, i.e., whether to block or keep the flow entry. While this method offers a fast and straightforward way to prevent a DDoS attack based on two parameters to profile the normal user behavior but this mechanism is not suitable for massive bandwidth attacks. Moreover, this work does not address the possibility of false-positives and collateral damage caused by dropping flows (see Table 2).

References [41,42] proposed packet-based frameworks which are directly implemented into the OpenFlow switches. In [41] Shin et al. tackle the scalability issues introduced by the behavior of OpenFlow as well as the response issues which arise during a DDoS attack. To achieve this, they introduce intelligence into the switch by implementing two extensions for flow management: connection migration and actuating triggers. With connection migration, they aim to protect the SDN data plane and control plane communication against a TCP SYN flooding DDoS attack. Connection migration allows the forwarding of only those flows to the controller that complete the TCP handshake. Otherwise, it drops them. While it stands out well in terms of blocking the half-opened connections, but it also introduces a small overhead and delay involved in the classification process. Actuating triggers, on the other hand, facilitate in dynamically inserting the flows into the switch to mitigate the potential effects of an attack in progress. Together these extensions offer a resilient security model for threats; however, the lack of DDoS attacks covered makes it unsuitable for defending against a wide range of sophisticated attacks faced today in the commercial and enterprise networks.

Kalkan et al. presented statistical and packet-based mechanism ‘SDNScore’ in 2016 [42]. In this approach, switches are provided with some level of intelligence to collect flow statistics. They configured the switch to detect and take action to mitigate the effect of a DDoS attack while working in collaboration with a controller. One rationale behind the approach is to keep the traffic in the data plane as much as possible which can make the control plane less susceptible to DDoS attacks. SDNScore’s implementation comprises five modules in total to detect and mitigate the attack in SDN. The four modules namely: Actuator, Comparator, Scorer and Profiler are loaded on the switch. The actuator module manages the activation of comparator and profiler modules when it detects a sudden burst in bandwidth. As the comparator module gets activated, it requests the profiler to generate a current profile while the attack is under progress. It then requests the nominal pair profile from the controller. The ProfilePair module, which is implemented on the controller, generates this profile during an attack-free period. After receiving these profiles, the comparator module then filters out two such parameters from the attack profile which deviate the most from profile generated during the attack-free period, i.e., a suspicious pair based on a comparison. The scorer module takes this suspicious pair and calculates a packet scoring value to classify the packets as malicious if the packet’s score value surpasses the accepted threshold value. Apart from the detection of well-known DDoS attacks, SDNScore also offers protection against unknown attacks. Furthermore, it aims to reduce collateral damage with the help of packet-based inspection by selectively discarding the attack packets based on a packet scoring technique.

In another recent and novel statistical-based approach [28], researchers investigated that due to the centralized control logic of SDN, most of the existing works have focused on protecting the control plane and the vulnerabilities of the data plane have been widely overlooked. The first aim of this study is to examine the limitations of the data plane to assert that DDoS attacks are very much possible on the switches of an SDN network. In this proposal, the authors thoroughly analyzed the behavior of a flow table size and miss-rate to prove that attackers can vandalize the performance of the SDN network concerning response time with a minimum amount of attack resources. They further developed a novel and stealthy DDoS attack which targets the data plane by exploiting the lifetime of flow entries in flow tables based on *idle_timeout*. However, it does not consider the *hard_timeout* parameter of flow entries. Several analytical and simulation experimentation results confirmed that a stealthy attack could go unnoticed and at the same time cause prolonged financial damages at a slow rate. To propose a suitable countermeasure for this stealthy attack, they have identified two significant differences between normal and stealthy attack traffic based on their analysis. (1) For normal traffic, the flow entries remain in the flow table for a comparatively short duration. (2) For attack flows, their last visit fields are usually small compared to the normal flows. Based on these observations, the two parameters used were: duration of the flow and the “last visit field” of the flow. For the detection of this attack, they placed a monitoring agent at the switch which was used to trace flow entry lifetime and last visit field (time counter) to find the attack. This decision might also make the switches somewhat more complicated, and besides, it provided no performance metrics for the usage of CPU and Memory.

In [31] researchers propose a novel defense against TCAM exhaustion attacks known as **S**electI**v**e De**F**ense for **T**CAM (SIFT). It protects the switches in SDN against Slow-TCAM exhaustion attacks. They point out several existing defenses that will not be able to guard against this attack successfully because of its slow and dynamic nature. The proposed method for protection against the attack is lightweight, and results indicate its high success rate. However, researchers believe that much research on the work has left undone in this area in order to improve and apply effective mitigation strategies. In another related work, Yuan et al. [29] present a peer support strategy to mitigate the effects of flow table against overloading attacks. The proposed strategy behind the approach is to combine and utilize the vacant space of flow tables available in all the switches to tolerate and reduce the effects of the attack. This work employs a mathematical modeling based system to devise conditions for defeating the attack. The researchers also validated the proposed method through several experimentations and found the capacity of SDN can be increased to a certain degree such that it can overcome the size of an attack bandwidth. Their fundamental strategy here is that whichever side has more resources, succeeds to overpower the opposite side. While the authors were able to build a resilient SDN network against the attacks by improving the resource management of flow tables, there are several issues and limitation which must be further scrutinized. These include such as performance overhead, the complexity involved in switches, calculation of optimal global route for peer support, and avoiding the consumption of the flow table resources in case the attack rate is enormous.

Recently, the emerging trends in network and communication technologies like the Internet of Things (IoT) and Cloud-based management services have integrated with SDN to adopt better management and reap the benefits of SDN. Cloud computing has been one of the widely used on-demand service technologies which significantly benefits from centralized control and manageability which SDN brings. Being one of the primary applications of SDN, it is no surprise that the DDoS attacks have also victimized it. Recently, Abdulqadder et al. [54] have proposed a framework to provide a secure SDN-based cloud architecture under the name of ‘SecSDN-Cloud.’ The goal of this framework is to shield the architecture against DDoS attacks which are able to cause flow table overloading, control plane saturation and byzantine attacks responsible for causing the controller failure. It analyzes network traffic based on several features such as port number, source IP, destination IP, source ethernet, destination ethernet, source TCP port, and destination TCP port to segregate malicious traffic from legitimate traffic. Based on the traffic features, policies are defined into switches to avoid flow overloading attacks. Control plane saturation attacks are mitigated by introducing more tolerance and load balancing provided by the use of architecture involving multiple controllers. Researchers in [44] proposed a smart security mechanism (SSM) to protect the IoT devices and servers in SDN-based IoT against new-flow attacks. These attacks occur by exploiting the inherent SDN vulnerabilities of control plane and data plane. Their framework consists of a detection module which employs a low-cost monitoring method. By determining the hit rate of flow entries, the detection module can distinguish between the normal

Table 2
Solutions based on Statistical-analysis for DDoS defense in SDN.

Packet or flow-based	References	Rationales used for detection	Detection plane	Validation	Dataset
Packet	[41]	Only complete TCP Connections are allowed to be forwarded to the controller, and half-opened connections are dropped.	Switch	Real-time	–
Packet (*Packet_In* message)	[53]	Two characteristics are used to identify a DDoS attack: 1. If the number of connections from an IP address is less than 'k,' then no action is taken, and the packet is assumed to be a DDoS attack packet. 2. Transmission of fewer than five packets 'n' per connection.	Controller	Simulation	–
Flow	[52]	FlowFence aims to provide a fair share of bandwidth for legitimate users under attack conditions. Therefore, it identifies a DDoS attack whenever the average-usage rate of bandwidth is higher than 80%. Under a DDoS attack scenario, the switch notifies the controller with a message which includes attack details.	Switch	Simulation	Simulated (MAGI agents)
Flow	[28]	To pinpoint a stealthy attack, they have identified two key differences between normal and stealthy attack traffic: 1. For some normal traffic, the flow entries remain in the flow table for a comparatively small duration. 2. For attack flows, last visit fields are small compared to the normal flows.	Switch	Simulation	Synthetic Dataset
Flow	[29]	By implementing a "peer support strategy", when the switch signals an attack condition after most of its flow table's space is taken by attack flow entries, it alerts the controller which directs the traffic to peer switches.	Switch	Mininet Emulator	–
Packet	[42]	The DDoS attack was identified based on any two traffic attributes with the most deviation from the 'pair profile' of legitimate traffic. Then, score values are computed and compared against the threshold to mark malicious packets.	Switch	Simulation	MAWI Working Group Traffic Archive dataset.
Flow	[44]	New-flow based DDoS attack was detected based on the hit rate of flow entries to distinguish them from normal flows.	Controller (ODL)	Simulation	Traces of normal traffic from campus network.

flows and new flows involved in the attack. They also proposed a mitigation method for a new-flow attack based on a dynamic access control technique. It migrates the malicious flows off the victim port and forwards them towards a security device to discover anomalies in results before applying dynamic access control.

4.2. Information entropy based DDoS defense for SDN

Information entropy is one of the most common and widespread methods to detect a DDoS attack. Many researchers over the years have adopted it for its simplicity and effectiveness. Entropy is a "measure of randomness" under the field of information theory. It is a lightweight method and makes it possible to identify a DDoS attack at the early stages under certain conditions. For a DDoS attack detection, entropy uses two main components: a window size based on either time or packet size and a baseline of normal traffic often referred to as a threshold value. It is possible to apply entropy to several network traffic features such as source IP, port, and destination IP to calculate the amount of randomness in their distribution in window size. The higher the computed value of entropy, the greater will be the randomness in the network traffic. For example, In [37] the controller monitors the destination IP address of the incoming packets. When a victim host is under DDoS attack, it gets suddenly bombarded with a large volume of inbound packets which causes the increase in the number of unique destination IP addresses and fills up the window size. As a result, entropy value falls sharply below the threshold and sets off the alarm as the detection of a DDoS attack occurs.

Over the years, many researchers have attempted to bring more intelligence into the OpenFlow switch. In [55], Wang et al. proposed a distributed DDoS detection technique based on entropy with an aim to reduce flow collection overload on the controller at the cost of making the switches intelligent and complicated. The authors implemented both the flow statistics collection and detection module and placed inside OpenFlow Edge switches to enable distributed monitoring and detection of DDoS attacks. For the attack detection, it calculates entropy on the destination IP addresses of incoming attack packets. The switch detects DDoS attack when there is an abrupt rise in the number of incoming packets with an identical destination IP address. It causes entropy to decrease under a proposed threshold limit, and the controller is notified of the attack. For feasible mitigation, they have pointed out the possibility of tracing back to the victim switch node by utilizing the benefits of SDN's centralized control capabilities. However, the method to separate malicious and legitimate packets is not given.

sFlow is a scalable and low-cost monitoring technique based on a sampling mechanism for high-speed networks. InMon technologies [56] proposed sFlow as a solution to reduce the flow collection overhead caused by the traditional OpenFlow monitoring approach. Since then it has been broadly accepted and implemented in many commercially available OpenFlow devices. sFlow offers a centralized architecture in which sFlow agents are installed on switches and sFlow collector is at a central location within the controller. sFlow agents capture the traffic using the sampling mechanism to construct statistical traffic data and immediately forward it to the sFlow collector. However, with sFlow, there is a trade-off involved between the accuracy of detection and overloading the control plane. Therefore, it is necessary to configure the sampling rate properly. A poorly configured sampling rate [57] can degrade and stagnate the performance of a detection mechanism. For example, if the sampling rate is configured to be too high, then its impact will be inflicted on the controller. However, if the sampling rate was set too low then the performance and accuracy of detection decreases (see Table 3).

In [58] Giotis et al. validated the performance of the sFlow monitoring approach against traditional OpenFlow (OF) monitoring technique. The research results indicated that OF statistics collection is neither a suitable choice for high-speed traffic and nor does it scale well. Therefore, in order to apply anomaly detection and mitigation in SDN, researchers proposed a modular design using the combination of OpenFlow and the sFlow-based approach. It utilized the sFlow-based approach for traffic monitoring and used OF for mitigation. To evaluate the research work, they performed several experiments that employed the entropy-based detection technique with the following parameters: source IP address, source port, destination IP address and destination port. With sFlow-based monitoring, the results showed drastic improvements in scalability and detection accuracy was approximately matched with experiments performed using native OF monitoring.

Moreover, the sFlow-based approach has much less overhead over control plane and system resources compared to the native OF monitoring technique. [59] proposed a source-based DDoS defense mechanism promoting the benefits of both SDN and sFlow. They deployed it as an application at the controller and validated their solution based on real traffic using a Mininet emulator. While the authors believe that it will be viable in defending against both inside and outside of network DDoS attacks, however, it is not suitable for large networks with thousand nodes and where the attack can come from different networks.

In the classical SDN approach for dealing with DDoS attacks, the controller handles the collection of flows from the switches. As a result, it puts the controller under a heavy processing load and congests the control channel. To overcome this scalability problem, researchers proposed the StateSec [60] framework to provide switches with the capability to monitor a DDoS attack without having to resort to the controller for collection and monitoring of traffic statistics. StateSec offers a novel technique based on stateful monitoring to increase the reaction time for attack detection. It is implemented inside switches to collect the necessary traffic features for attack detection with the help of state tables used in conjunction with flow tables. A state in the table is said to be updated when the count for the number of times a feature appears in network traffic is incremented. The detection mechanism resides in the controller; however, they have strongly suggested the possibility of moving it on the switch using the Extended Finite State Machine (XFSM) approach. Overall, it demonstrates high accuracy when compared with traditional monitoring approaches such as OpenFlow monitoring as well as sFlow monitoring.

References [61,62] employ *Packet_In* based detection of DDoS in SDN. This approach was introduced to reduce the further overhead of the controller caused by the collection of flow statistics, especially during which the DDoS attack is in progress.

In [61] You et al. extracted the relevant flow features of *Packet_In* message required to achieve detection in real-time. To provide the detection of a DDoS attack in real-time, they have applied a trigger mechanism instead of a loop-based detection mechanism. The authors implemented the overall detection module on the controller. It used three parameters to compute entropy on namely: Destination IP address, destination IP port and source IP address. The rationale for the detection of a DDoS attack is based on the premise that when the attack occurs, a victim host will receive an enormous amount of packets from a botnet and therefore, the entropy of source IP address will increase while the entropy of destination IP address and destination IP port will decrease. The drawback of this research work is that it validated the results using a simulated environment and no real network traffic was not used. In [62] Jiang et al. presented a two-stage Entropy-based DDoS Defense Mechanism (EDDM) based on the research findings to characterize flash traffic from attack traffic with a similar purpose to detect DDoS attack in an SDN environment. Their proposed model for detection runs on the SDN controller.

In contrast with existing approaches, it prevents the loss of legitimate packets and minimizes collateral damage. It is also possible to traceback [63] to the attacker and block its traffic to reduce further damage to network resources. For evaluation, it used three scenarios to examine the effectiveness of EDDM: 1. DDoS without any defense mechanisms 2. DDoS with EDDM, 3. Analysis of flash crowd and DDoS with EDDM. For mitigation, the controller directly blocks the attack packets instead of throttling their bandwidth. The authors studied that

Table 3
Solutions based on Information Entropy for DDoS defense in SDN.

Packet or flow-based	References	Rationales used for detection	Detection Plane	Validation	Dataset
Flow	[58]	The anomaly behavior of network flows under DDoS attack generated a surge of attack packets targeted towards the victim. As a result, a high rate of packets containing destination IP and destination port caused the entropy to decrease significantly.	Controller (NOX)	Emulation	Three different datasets from real traffic from Campus locations and Scapy tool for attack traffic.
Flow	[37]	The destination IP of incoming packets is monitored. If the attack is targeted towards a host, a large number of packets will fill out the packet window size reducing the entropy value until it falls below the chosen threshold.	Controller (POX)	Mininet Emulator	Traffic Simulation Tools (Scapy)
Flow	[55]	They have taken the destination IP address of the incoming packets as their basis for attack detection. When there is a sudden increase in the DDoS packets, the entropy of packets with the same destination IP steeply falls below the threshold.	Edge switch	Mininet Emulator	Center for Applied Internet Data Analysis (CAIDA) DDoS Attack Dataset 2007
Packet (*Packet_In message based)*	[61]	The authors identified DDoS attack based on the rationale that when a victim host is under an attack, it will receive an enormous amount of *Packet_In* messages from a botnet and therefore, the entropy of destination IP address and destination IP port will decrease while the entropy of IP Source address will increase.	Controller (Ryu)	Mininet Emulator	Traffic Simulation Tools (TFN 2K)
Packet	[60]	Various kinds of anomaly behavior patterns are taken into account for the detection of various types of attacks. 1. A denial of service attack is revealed as distributed if there is an increase in entropy of source address. Otherwise, there is a sharp decrease in entropy of both the destination address and destination port. 2. For spoofing attacks, the entropy of the source port field was used.	Controller (Ryu)	Mininet Emulator	"BigFlows" trace from a real network for legitimate traffic and hping3 for attack generation.
Packet (*Packet_In message based)*	[62]	The authors investigated two distinguishing characteristics of flash event traffic which separates it from DDoS attack traffic. 1. First of all legitimate hosts have a real IP address, and their flash event packets are distributed in the entire network. 2. On the other hand, DDoS attack packets have a huge number of spoofed IP addresses coming from specific attack bots.	Controller (Floodlight)	Mininet Emulator	–

banning the attack's source IP address is the best way to deal with DDoS attacks to reduce the damage on network bandwidth and SDN controller. Moreover, the second reason they chose to block malicious packets directly is that unlike the previous studies, the EDDM approach can also traceback to the attack sources.

4.3. Machine learning (ML) based DDoS defense for SDN

Machine learning provides defense systems with an ability to self-learn from a large set of data to identify hidden patterns and make decisions requiring no explicit instructions. Some of the machine learning-based techniques include neural networks, Bayesian networks [64], Self-organizing map (SOM) [65], Naïve Bayes, and Support Machine Vector (SVM) [66] to segregate the attack traffic and legitimate traffic. Machine learning algorithms have been widely used over the years in traditional intrusion detection systems [67] but recently they also started to appear under the SDN context. In [68] Nanda et al. proposed a machine learning method in order to predict the possible attack connections and as well as locations where possible attacks can happen. The authors in this research work analyzed three historical network datasets to train various machine learning algorithms namely C4.5, Bayesian Network (BayesNet), Decision Table (DT), and Naïve Bayes. They used these trained models to classify the real-time traffic and install blocking rules on the SDN controller accordingly. The research showed that even the slightest possibility of attack could sometimes contribute to defending against a more significant impact. It also compared the performance of these results to determine their accuracy of prediction.

Similarly, in [69] the researchers perform various experiments to evaluate the performance of machine learning-based algorithms such as J48, Random Forest, Random Tree, Decision Table, MLP, Naïve Bayes, and Bayes Network. The classifiers and experiments used the KDD-99 dataset. The work in [70] employs a refined version of the KDD-99 dataset known as NSL-KDD for their intrusion detection system (IDS) based on machine learning. This solution adopts a flow-based technique built on signature-based architecture and utilizes the backpropagation algorithm for training the model. They validated the proposed research work by conducting simulation experiments on a virtual testbed with multiple attacks including denial of service attacks.

In [71], the researchers proposed a lightweight DDoS detection mechanism based on traffic flow features using a self-organizing map (SOM) classifier. The detection method comprises three modules: Flow collector, flow extractor and classifier. These modules are installed on the controller which executes them inside a detection loop to recognize a DDoS attack. The flow collector module gathers flow entries from the OpenFlow switches during a pre-defined interval of time. The flow extractor module extracts the features from flow entries which are considered essential for DDoS attack detection. It gathers the extracted features into 6-tuples. Lastly, the classifier module receives the features data gathered in 6-tuple and analyses whether it represents an attack or normal traffic by utilizing SOM. In order to achieve attack detection, several anomalies in network traffic are taken into account to detect DDoS attack such as (a) Average of Packets for each flow (ADf), (b) Average of Bytes for each flow (ABf), (c) Average of Duration for each flow (ADf), (d) Percentage of Pair-flows (PPf), (e) The growth of single-flows (GSf), and (f) The growth of Different Ports (GDP). They have used three types of attacks: TCP/SYN Flood, UDP Flood, and ICMP Flood attacks for training and testing the model. By taking advantage of the benefits of SDN architecture, the researchers were also able to traceback to the OpenFlow switches in the SDN network where the DDoS attack was detected. To make the scheme lightweight, it collects per-flow statistics without having to inspect every packet.

Furthermore, it extracts features with small overhead compared to other KDD-99 dataset-based approaches. The low overhead comes from the fact that SDN provides a programmatic interface for retrieving the information from OpenFlow switches. Although it achieves a high rate of detection with low false alarms, it is challenging to select an optimal detection loop interval time. Also, the flow collector module does not consider the controller overload caused by the collection of flow entries from all the switches (see Table 4).

In another machine learning-based solution [72] the authors proposed an unsupervised machine learning-based mechanism with an aim to detect and protect SDN architecture against DDoS attacks. The proposed approach utilizes a stochastic and self-learning algorithm known as the Restricted Boltzmann Machine (RBM) algorithm to identify DDoS attacks in the network traffic. Their detection module takes the most "reliable" network features which include the energy consumption rate of switches, hit count of incoming *Packet_In* messages going towards the controller, and modified entries in the flow table. In order to identify and classify the DDoS attack, Restricted Boltzmann Machine (RBM) algorithm was trained using Contrastive Divergence (CD) algorithm. For the training purposes and dataset generation, the learning model has taken the following parameters: source IP, source port, destination IP, destination port and type of protocol. For detection and mitigation, the algorithm compares the hit count with self-learned average threshold values, tests whether energy consumption is higher for a particular MAC address, and inspects the sudden deviations in flow entries. They concluded that the proposed solution was able to achieve a high detection rate with low false positives.

Cui et al. [43] propose SD-Anti-DDoS mechanism to defend against DDoS attacks effectively. It comprises four modules namely, an attack detection trigger, detection module, the traceback module, and mitigation module. It introduces a novel strategy for attack detection which relies on *Packet_In* message based attack detection trigger module. The purpose of this module is to reduce the heavy workload of switches and the controller which comes in effect if a periodic trigger is used. As a consequence, it also reduces the response time against the attack. It is based on neural networks and uses a backpropagation neural network (BPNN) classifier for identifying the attack and characterizing legitimate and attack flows. The attack traceback module aids the mitigation module to block the attack sources and clean up the malicious flow entries deposited inside victimized switches. However, there is still an overhead with this approach in case high-speed DDoS traffic was received. Since the controller requests all flows from the switches for detection purposes, it results in heavy congestion at the control plane and response time will be affected. The authors in [73] propose a framework for the detection and mitigation of DDoS attacks to protect the SDN control plane and data plane resources. To achieve attack detection, they use a combination of techniques that include entropy to compute variations in network traffic features and then employ an SVM classifier to identify whether an attack condition exists. In order to neutralize the impact of the attack, the authors introduced a whitelist-based mitigation module which incorporates a mitigation agent installed at a network switch level to directly migrate the legitimate flows and drop the attack flows at the same. The accuracy of their model is improved under moderate traffic by utilizing *Packet_In* based method, and in addition, the overhead is also reduced by adopting sFlow-based monitoring for a vast amount of incoming packets under high network traffic.

Recently, researchers have also devised defense models based on deep learning methods. It is a subset of the machine learning approach which has also gained noticeable attention in identification and detection of DDoS attacks in SDN. It offers improved classification and detection rate accuracy over existing machine learning techniques. In 2017, Niyaz et al. proposed a multi-vector DDoS detection system using deep learning approach [74]. The deep learning technique applied in this work utilizes a layered model of Stacked Auto-Encoder (SAE) and softmax classifier for training and classification. Their detection system is implemented as a POX controller application to monitor the entire network from a single point. It separately identifies various kinds of DDoS attacks which offers more control in mitigating a specific type of attack. While the classification module used to segregate the malicious and normal traffic achieves high accuracy, however, the control

Table 4
Solutions based on Machine Learning (ML) methods for DDoS defense in SDN.

Packet or flow-based	References	Rationales used for detection	Detection plane	Validation	Dataset
Flow	[71]	This work has employed several statistical network traffic parameters to form their basis of detection of DDoS attack: 1. Average of Packets for each flow (ADf) 2. Average of Bytes for each flow (ABf) 3. Average of Duration for each flow (ADf) 4. Percentage of Pair-flows (PPf) 5. Growth of Single-flows (GSf) 6. Growth of Different Ports (GDP)	Controller (NOX)	Emulation	Mixed legitimate traffic and Attack generation tools.
Both (*Packet_In* message and flow based)	[43]	Abnormal arrival of *Packet_In* messages towards the controller was used as an initial indicator of a possible attack. Several anomalies in the network were taken into account based on the following parameters: 1. Number of packets matched per flow entry, 2. Number of bytes matched per flow entry, 3. Duration of each flow entry, 4. Packet rate per flow entry 5. Byte rate per flow entry.	Controller (Ryu)	Mininet Emulator	Normal and Attack traffic was simulated using D-ITG and TFN2K Tool.
Flow	[72]	Their algorithm compares the hit count with self-learned average threshold values and tests whether energy consumption is higher for a particular MAC address, and inspects the abrupt deviations in flow entries.	Controller (POX)	Mininet Emulator	Training dataset was generated using given parameters and attack traffic simulation tools *hping3* was used
Packet	[74]	The training dataset was used to generate multiple classifier models such as neural network-based, soft-max, and SAE in order to identify various kinds of DDoS attacks.	Controller (POX)	SDN Testbed on VMWare ESXi Host	Real HWN traffic of three days was captured and replayed. For attack generation, *hping3* was used.
Both (*Packet_In* message and flow based)	[73]	Several network features (src IP, dest IP, src port, dest port) are extracted, and their entropy is computed for variations. Then, a supervised learning-based (SVM) model is used to automatically identify anomalies in the network by classifying the flows as normal (−1) and abnormal (+1).	Controller (POX)	Mininet Emulator	Dataset generated from capture of traffic from the given laboratory; and TFN2K as an attack simulation tool.

plane can experience congestion and performance overload because the feature extractor module has to extract features from every incoming packet collected by the traffic collector module.

5. Open research issues and challenges

SDN promises to simplify network management and offers numerous benefits. However, it also brings many security-related research challenges that are needed to be addressed. This section lists the security challenges brought by SDN such as scalability, centralization, continuous availability of services, security for SDN and as well as challenges related to defending against DDoS attacks which include characterizing (flash events) FEs and DDoS traffic, simulation-based evaluations, lack of benchmarked datasets and mobile and IoT botnets.

5.1. Scalability

While the decoupling of the control plane and data plane introduces plenty of benefits, it also brings scalability issues between them. A centralized controller will require a huge amount of computation and processing power for the increasing size of the network to avoid performance, latency, and response time issues. According to a recent report [33] from 2018 on controllers in SDN, it has shown that centralized controllers have a limited capacity of flow rate per second for handling the communication with devices in the data plane. For example, one of the most popularly used controllers, e.g., NOX is only able to handle 30K flow requests, whereas enterprise-level large network organizations and data centers require much higher flow rates which can go above 10 million flows per second depending on the size and service requirements. So, as the network grows in size by the number of switches, the centralized controller becomes a bottleneck. Moreover, the switches in data plane have hardware limitations, e.g., limited TCAM size which result in network and resource bottlenecks of their own.

Among the possible ways to overcome the scalability issues include: (1) to adopt a distributed controller design and (2) to optimize the centralized controllers itself by exploiting parallelism techniques. Regarding distributed controllers, the proposed solutions such as Kandoo [75], DIFANE [76], and Onix [77] can increase the scalability levels in data centers and enterprise-level organizations where the requirement is to handle millions of flows per second within a minimum latency range. While the distributed controller design offers load balancing and provides a solution to single controller failure, but communication overhead between the controllers, synchronization issues, and further minimizing the latency are among the open research challenges.

In the approach involving optimization of a centralized controller, parallelism techniques can increase its processing power to serve a high number of flow requests per second. For example, Beacon [78] is among one of these controllers to handle high flows/sec (up to 12.8M) with an average latency of 24.7 μs. NOX-MT [79] is a successor of NOX capable of achieving a 1.8M flow throughput rate and average response time of 2 μs. These controllers have their own requirements, performance benchmarks, benefits, and disadvantages depending on the application [80].

5.2. Centralization

Due to the radical change brought by SDN to the traditional network paradigm, network devices have become simple forwarding devices, whereas the SDN controller is a centralized software component. While it is much easier to update the network policies, firewalls, and monitor the whole network from a logically centralized location but on the other hand, SDN controller is also regarded as a single point of failure which can result in crashing down the entire network under a DDoS attack. In traditional networks, if one or more network devices or a link between them fails, the network traffic could take an alternative path to maintain the continuity of flow. It is not so much the case with SDN, the packets traveling on the failed link will be dropped until they get recognized by the controller, and their flow entries are updated. Furthermore, if switches lose contact with the controller, they lose their decision control ability altogether.

Under the context of DDoS attacks, the problem of centralization has been addressed by several research articles cited in Section 4. Their proposals suggest statistical-based, information entropy, and machine learning-based techniques in order to detect and mitigate the impact of an attack and, therefore, to minimize the effects of centralization. Distributed controllers and multi-threaded centralized controllers are also among the possible solutions to avoid centralization, improve load distribution and reliability. However, in [81,82], authors have indicated that using a distributed SDN controller is not itself a solution to this problem and proposed a defense framework to cope with controller failures.

5.3. Continuous availability of services

Internet services such as web, communications, cloud computing cannot exist without the availability of network resources. DDoS attacks are the primary threat and are targeted towards exhausting the availability of these services. The benefit of a traditional network is that because of its distributed nature, it can survive an attack if a switch or router fails [83] to function. In contrast to this, with an SDN-based network, the communication of forwarding devices with the controller must be maintained in order to ensure availability. As previously mentioned in Section 3, the centralized controller in the SDN network is the driving force of the entire network. DDoS attacks on SDN architecture such as on OpenFlow (Southbound API) channel, data channel link between the switches, or on the multiple switches can make the controller unavailable for the rest of the devices in the same network.

One approach to confront this problem is to introduce backup or standby controllers in the network [84,85]. The key idea is that in case the central controller goes out of service or gets attacked, a secondary backup controller can take its place. This method requires frequent synchronization and consistent network states to be maintained among the controllers. In addition, given the massive impact and bandwidth of DDoS attacks, the backup controller(s) might potentially face similar consequences. Hence, this dependency on the controller possesses a primary challenge for the availability of services provided by SDN networks.

5.4. Security for SDN

Large organizations, data centers, ISPs, and other enterprise-level domains have high-security requirements for providing continuous services to legitimate users. Several new security reports [3,86,87] have shown that they are the most attractive domains and get often targeted by DDoS attacks. These security issues associated with SDN must be resolved before it gets properly deployed, able to gain a valuable reputation, and deliver uninterrupted services.

The taxonomy provided in Section 3 of this article illustrates that the SDN architecture suffers from many security threats induced by new-flow based attacks on the controller, switches, data channels, OpenFlow channel, etc. and a variety of solutions have been proposed in Section 4 with their own inherent advantages and disadvantages. The idea that the first new packet arriving at switch must be forwarded to the controller in a reactive mode has itself caused a major security concern that is responsible for giving birth to a new kind of DDoS attack. Therefore, it is widely seen as an inherent vulnerability that allows a DDoS attacker to take control, exhaust the resources, and consequently shut down the services of a network for legitimate users.

The hardware limitation of switches and their incapability to make decisions are also among the principal security challenges which openly encourage the DDoS attackers. While the SDN infrastructure promises to have reduced hardware and operational costs, it has unquestionably raised new security concerns and brought several research issues. It remains to be seen whether deploying the extra security mechanisms to protect the centralized nature of SDN would increase costs and evolve the switches to the same point where they regain intelligence and make networking decisions.

5.5. *Characterizing (flash events) FEs and DDoS traffic*

Unlike a DDoS attack which is an intentional attempt to degrade a server's network performance and inflict both financial and reputational damages, a flash event refers to a scenario in which legitimate users are suddenly trying to send access requests in a large number. Thereby, the performance and quality of service of a targeted server is degraded or completely disrupted.

In order to successfully identify and mitigate the effects of a DDoS attack, it is very crucial that flash events must not be misjudged as a DDoS attack. However, due to their similar characteristics, distinguishing FEs and DDoS traffic has been one of the most difficult and challenging aspects of network security [88]. Another important reason is that there is an unavailability of such reliable and real datasets which contain a combination of FE and DDoS traffic which can be utilized for conducting experiments [89].

Therefore, because of their elusive nature and the fragile line between their behavior, most of the researchers have overlooked and neglected to include this area in their research and only a few of them considered to classify them [58,62].

5.6. *Simulation-based evaluations*

Although several DDoS detection and mitigation solutions have been proposed in academic research, the rise in recent DDoS attack incidents in the real world indicates the ineffectiveness of these solutions. The major reason has been inappropriate validations of these solutions. Most of these solutions have been validated using emulation and simulation tool in SDN environment for e.g. EstiNet [90], Mininet [91], OpenvSwitch [22], OpenDayLight [92], Floodlight [93], NOX [94], POX [95], and RYU [96] etc. The downside is that the results are neither accurate nor realistic compared to those which are done on the actual hardware in a realistic environment. In some of the works, researchers are still relying on legacy datasets from traditional networks to assess their DDoS defense methods. They are mapped using a simpler simulated SDN network often using a single virtualized host machine under minimal resources. These small test networks cannot accurately represent the Internet over which tons of high-bandwidth DDoS attacks are launched every once in a while. Therefore there is a need strong need to put substantial efforts to develop state-of-the-art and large-scale SDN-enabled testbeds to validate DDoS detection and mitigation in realistic scenarios.

5.7. *Lack of benchmarked datasets*

Most of the research works have been validated using synthetic datasets generated using simulation-based experiments. Some of them used publicly available traditional datasets. The traditional datasets which are mostly used include KDD-99 and CAIDA (2007) dataset [97]. These datasets seem to be obsolete for conducting experiments for a rapidly changing network paradigm like SDN.

It is possible that if an organization has been a victim of DDoS attacks, it might feel reluctant to publicly publish these datasets potentially containing the landscape of their organizational network activity records. Thereby, one of the primary reasons behind the unavailability of DDoS attack benchmark datasets could be that these organizations may feel that it could inflict damage to their reputation if they release these datasets. Moreover, the organizations could possibly lose their customers over privacy concerns causing them as well as financial damages.

5.8. *Mobile and IoT botnets*

With the evolution of technology, recent years have witnessed increased growth and demand for smart devices. Modern DDoS attackers have also adopted sophisticated mechanisms to launch distributed attacks using mobile and IoT botnets. IoT botnets, due to their growing size are capable of delivering high volume impacts that the Internet has ever seen. The largest attack on record generated a throughput of 1.2 Tbps victimizing the servers of DYN using Mirai botnet [98]. A recent 2018 DDoS threat report from NexusGuard [86] observed a dramatic rise in the size of DDoS attacks was due to large-scale botnets created from insecure IoT devices. It shows that attacks were increased by 29.02% since the second quarter of 2017.

On the other hand, Mobile botnet, for example, is comprised of smartphones and mobile devices with internet service capabilities. Due to their distinguishing characteristics, these attacks are difficult to detect by traditional defense mechanisms. Apart from the statistical-based detection techniques [99], machine learning techniques have also been proposed to detect mobile botnet-based attacks. Examples of these techniques include in the research works recently proposed by authors in [100] and [101]. Mobile botnets are developed to inflict long-term effects, consume low bandwidth, low battery power, and designed to operate silently in the background. For example, the study in [102] shows that low-rate DDoS attacks using mobile botnet are challenging to detect because they consume low bandwidth and resources. Moreover, mobile bots are able to use stealthy communication mechanisms which further complicates their detection [103].

Thus, the increasing amount of IoT devices and the added sophistication involved in attack behaviors introduce prominent research challenges and open research questions on how to effectively defend against them in real-time.

6. Conclusion and future directions

SDN has a promising future as an emerging network technology with security being one of the most prominent and compelling need. DDoS attacks have evolved in sophistication, and each year massive attacks are launched with growing frequency and size responsible for targeting large organizations, data centers, and other enterprises. New-flow based DDoS attacks have raised significant security concerns in SDN by posing numerous security challenges and threats to the architecture and provoking the need for continuous research efforts in the development of novel security defenses.

In this review paper, we presented a classification of security issues in SDN architecture concerned with new-flow based DDoS attacks. By focusing on the intrinsic security issues, we devised a taxonomy of the major design vulnerabilities in SDN architecture through which DDoS attackers are able to gain leverage to initiate a new-flow based DDoS attack. The proposed taxonomy aims to provide a clear hierarchical view of the possibilities where a new-flow based DDoS attack can victimize the SDN architecture. Moreover, it will help the research community to find effective DDoS defense solutions by understanding the dimensions, interdependencies, and impact of the DDoS attack problems. Furthermore, we provided a state-of-the-art review of DDoS defense solutions including the latest developments and techniques used by researchers under the SDN environment. We also presented rationales for the detection of DDoS to gain insight into the novel strategies used over recent years to identify how the attack patterns of DDoS attackers have changed and developed over time. Finally, we looked at some perplexing research challenges concerned with SDN's security.

Since SDN brings flexibility and innovation to the networking, therefore it has the potential to integrate with unforeseen future technologies to bring many research opportunities. This review study in our opinion is the first to focus on the impact of a new-flow based DDoS attack on different planes of SDN. In the future, it remains to be seen how the developments in DDoS defense techniques tackle these SDN's security-related research challenges present in the architecture. Below, we have identified some future directions and gaps related to SDN security:

- Most of the researchers have tried to create defense solutions using a single controller involved in the network topology. However, the centralized nature of SDN is viewed as a critical vulnerability point to DDoS attacks while a distributed controller design offers much better load distribution, processing power, and reliability. These capabilities of a distributed control architecture can be utilized to increase the performance where a centralized controller becomes a bottleneck as the overhead of DDoS attack increases. Therefore, to the best of our knowledge and based on the literature review, employing such a multi-controller setup to defend against these attacks is still a very new research area among researchers. Future work can involve creating DDoS defense framework solutions based on distributed SDN controllers to protect the network while minimizing communication overhead, failures, and balancing the traffic load between the controllers.
- We observed that many researchers are interested in bringing intelligence into the switch. One of their motivations behind adopting this strategy is to reduce the overload of a centralized controller and letting the switches to process the flows locally. As long as the controller remains the network authority, it does not seem to break the underlying design principle of SDN. However, as switches incorporate more control and security mechanisms, both their marketing cost and complexity are likely to rise. Therefore, in order to efficiently deploy security mechanisms on switches, this trade-off needs careful research investigation to develop defense solutions against DDoS attacks.
- More than 95% of popular and latest research works we reviewed in this paper have used emulation, simulation and other virtualized environments to conduct experiments. Often these experiments involved the use of a more straightforward and smaller network. As a result, the obtained results do not accurately validate the methodology used by researchers. In contrast, by employing real hardware and testbed facilities, better results can be achieved. Hence, experimentation and validation of future SDN-based security applications using such real testbeds is another important future direction.
- Out of the three DDoS defense techniques reviewed, we believe that machine learning has a great potential to evolve in future network applications. DDoS attackers have been continuously upgrading their weaponry and techniques to inflict vast amounts of financial damages while trying to remain undetected by most traditional defense mechanisms. Supervised learning techniques under machine learning provide the ability to detect unknown attacks, and they have been successfully applied to achieve better detection accuracy. Moreover, it enables the ability to fully automate the process of detection with decidedly less human intervention. Therefore, we firmly believe that machine learning provides a future research direction for SDN security applications.

As a part of our future work, we are motivated to propose a machine learning-based detection and mitigation framework by evaluating the performance of various machine learning algorithms.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, ACM SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74.

[2] No sooner did the ink dry: 1.7tbps ddos attack makes history, 2018, https://www.netscout.com/blog/security-17tbps-ddos-attack-makes-history, (Accessed 23 January 2020).

[3] Arbor's worldwide security report, 2017, https://pages.arbornetworks.com/rs/082-KNA-087/images/13th_Worldwide_Infrastructure_Security_Report.pdf, (Accessed 23 January 2020).

[4] P. Kathiravelu, L. Veiga, SD-CPS: taming the challenges of cyber-physical systems with a software-defined approach, in: 2017 Fourth International Conference on Software Defined Systems, SDS, 2017, pp. 6–13, http://dx.doi.org/10.1109/SDS.2017.7939133.

[5] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, J.P. Jue, All one needs to know about fog computing and related edge computing paradigms: A complete survey, J. Syst. Archit. 98 (2019) 289–330, http://dx.doi.org/10.1016/j.sysarc.2019.02.009, http://www.sciencedirect.com/science/article/pii/S1383762118306349.

[6] S. Shin, G. Gu, CloudWatcher: Network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?), in: Network Protocols (ICNP), 2012 20th IEEE International Conference on, IEEE, 2012, pp. 1–6.

[7] P. Manso, J. Moura, C. Serrão, SDN-Based intrusion detection system for early detection and mitigation of ddos attacks, Information 10 (3) (2019) 106, http://dx.doi.org/10.3390/info10030106, https://www.mdpi.com/2078-2489/10/3/106.

[8] J. Zheng, Q. Li, G. Gu, J. Cao, D.K. Yau, J. Wu, Realtime DDoS defense using COTS SDN switches via adaptive correlation analysis, IEEE Trans. Inf. Forensics Secur. 13 (7) (2018) 1838–1853.

[9] Y. Xu, Y. Liu, Ddos attack detection under SDN context, in: INFOCOM 2016-the 35th Annual IEEE International Conference on Computer Communications, IEEE, IEEE, 2016, pp. 1–9.

[10] A. Wang, Y. Guo, F. Hao, T. Lakshman, S. Chen, Scotch: Elastically scaling up sdn control-plane using vswitch based overlay, in: Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies, ACM, 2014, pp. 403–414.

[11] Q. Yan, F.R. Yu, Q. Gong, J. Li, Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges, IEEE Commun. Surv. Tutor. 18 (1) (2016) 602–622, http://dx.doi.org/10.1109/COMST.2015.2487361.

[12] N. Dayal, P. Maity, S. Srivastava, R. Khondoker, Research trends in security and DDoS in SDN, Secur. Commun. Netw. 9 (18) (2016) 6386–6411, http://dx.doi.org/10.1002/sec.1759, https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.1759.

[13] N.Z. Bawany, J.A. Shamsi, K. Salah, DDoS Attack detection and mitigation using SDN: Methods, practices, and solutions, Arab. J. Sci. Eng. 42 (2) (2017) 425–441, http://dx.doi.org/10.1007/s13369-017-2414-5.

[14] K. Kalkan, G. Gur, F. Alagoz, Defense mechanisms against ddos attacks in SDN environment, IEEE Commun. Mag. 55 (9) (2017) 175–179, http://dx.doi.org/10.1109/MCOM.2017.1600970.

[15] M.M. Joëlle, Y.-H. Park, Strategies for detecting and mitigating DDoS attacks in SDN: A survey, J. Intell. Fuzzy Syst. 35 (6) (2018) 1–13, http://dx.doi.org/10.3233/JIFS-169833, http://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/JIFS-169833.

[16] Open networking foundation, 2018, https://www.opennetworking.org/, (Accessed 23 January 2020).

[17] What is openflow? definition and how it relates to SDN, 2013, https://www.sdxcentral.com/networking/sdn/definitions/what-is-openflow/, (Accessed 23 January 2020).

[18] Openflow switch specification (version 1.5.1), 2015, https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf, (Accessed 23 January 2020).

[19] A. Azzouni, R. Boutaba, N.T.M. Trang, G. Pujolle, sOFTDP: Secure and efficient OpenFlow topology discovery protocol, in: NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, 2018, pp. 1–7, http://dx.doi.org/10.1109/NOMS.2018.8406229.

[20] K. Benton, L. Camp, C. Small, OpenFlow Vulnerability assessment, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ACM, 2013, pp. 151–152.

[21] K.-y. Chen, A.R. Junuthula, I.K. Siddhrau, Y. Xu, H.J. Chao, SDNShield: Towards more comprehensive defense against DDoS attacks on SDN Control plane, in: Communications and Network Security (CNS), 2016 IEEE Conference on, IEEE, 2016, pp. 28–36.

[22] Open vSwitch, https://www.openvswitch.org/, (Accessed 23 January 2020).

[23] S.H. Yeganeh, A. Tootoonchian, Y. Ganjali, On scalability of software-defined networking, IEEE Commun. Mag. 51 (2) (2013) 136–141.

[24] Z. Guo, Y. Xu, R. Liu, A. Gushchin, K. yin Chen, A. Walid, H.J. Chao, Balancing flow table occupancy and link utilization in software-defined networks, Future Gener. Comput. Syst. 89 (2018) 213–223, http://dx.doi.org/10.1016/j.future.2018.06.011, http://www.sciencedirect.com/science/article/pii/S0167739X18306666.

[25] A. Mondal, S. Misra, I. Maity, Buffer size evaluation of openflow systems in software-defined networks, IEEE Syst. J. 13 (2) (2019) 1359–1366, http://dx.doi.org/10.1109/JSYST.2018.2820745.

[26] Y. Jarraya, T. Madi, M. Debbabi, A survey and a layered taxonomy of software-defined networking, IEEE Commun. Surv. Tutor. 16 (4) (2014) 1955–1980.

[27] Z. Shu, J. Wan, D. Li, J. Lin, A.V. Vasilakos, M. Imran, Security in software-defined networking: threats and countermeasures, Mob. Netw. Appl. 21 (5) (2016) 764–776, http://dx.doi.org/10.1007/s11036-016-0676-x.

[28] X. Wu, M. Liu, W. Dou, S. Yu, DDoS Attacks on data plane of software-defined network: are they possible? Secur. Commun. Netw. 9 (18) (2016) 5444–5459.

[29] B. Yuan, D. Zou, S. Yu, H. Jin, W. Qiang, J. Shen, Defending against flow table overloading attack in software-defined networks, IEEE Trans. Serv. Comput. 12 (2) (2019) 231–246, http://dx.doi.org/10.1109/TSC.2016.2602861.

[30] Y. Qian, W. You, K. Qian, Openflow flow table overflow attacks and countermeasures, in: Networks and Communications (EuCNC), 2016 European Conference on, IEEE, 2016, pp. 205–209.

[31] T.A. Pascoal, Y.G. Dantas, I.E. Fonseca, V. Nigam, Slow TCAM exhaustion DDoS attack, in: IFIP International Conference on ICT Systems Security and Privacy Protection, Springer, 2017, pp. 17–31.

[32] J. Leng, Y. Zhou, J. Zhang, C. Hu, An inference attack model for flow table capacity and usage: exploiting the vulnerability of flow table overflow in software-defined network, CoRR abs/1504.03095 (2015) http://arxiv.org/abs/1504.03095.

[33] M. Paliwal, D. Shrimankar, O. Tembhurne, Controllers in SDN: A review report, IEEE Access 6 (2018) 36256–36270.

[34] C.-H. Hung, C.-W. Huang, L.-C. Wang, C. Chen, Scalable topology-based flow entry management in data center, in: Consumer Communications & Networking Conference (CCNC), 2016 13th IEEE Annual, IEEE, 2016, pp. 85–90.

[35] C. Gkountis, M. Taha, J. Lloret, G. Kambourakis, Lightweight algorithm for protecting SDN controller against ddos attacks, in: Wireless and Mobile Networking Conference (WMNC), 2017 10th IFIP, IEEE, 2017, pp. 1–6.

[36] N.G. Dharma, M.F. Muthohar, J.A. Prayuda, K. Priagung, D. Choi, Time-based DDoS detection and mitigation for SDN controller, in: Network Operations and Management Symposium (APNOMS), 2015 17th Asia-Pacific, IEEE, 2015, pp. 550–553.

[37] S.M. Mousavi, M. St-Hilaire, Early detection of DDoS attacks against SDN controllers, in: Computing, Networking and Communications (ICNC), 2015 International Conference on, IEEE, 2015, pp. 77–81.

[38] P. Dong, X. Du, H. Zhang, T. Xu, A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows, in: 2016 IEEE International Conference on Communications, ICC, 2016, pp. 1–6, http://dx.doi.org/10.1109/ICC.2016.7510992.

[39] S. Li, Y. Cui, Y. Ni, L. Yan, An effective SDN controller scheduling method to defence DDoS attacks, Chinese J. Electron. 28 (2019) https://digital-library.theiet.org/content/journals/10.1049/cje.2019.01.017, 404–407(3).

[40] H. Yang, J. Ivey, G.F. Riley, Scalability comparison of SDN control plane architectures based on simulations, in: Performance Computing and Communications Conference (IPCCC), 2017 IEEE 36th International, IEEE, 2017, pp. 1–8.

[41] S. Shin, V. Yegneswaran, P. Porras, G. Gu, Avant-guard: Scalable and vigilant switch flow management in software-defined networks, in: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, ACM, 2013, pp. 413–424.

[42] K. Kalkan, G. Gür, F. Alagöz, SDNScore: A statistical defense mechanism against DDoS attacks in SDN environment, in: Computers and Communications (ISCC), 2017 IEEE Symposium on, IEEE, 2017, pp. 669–675.

[43] Y. Cui, L. Yan, S. Li, H. Xing, W. Pan, J. Zhu, X. Zheng, SD-Anti-DDoS: Fast and efficient DDoS Defense in software-defined networks, J. Netw. Comput. Appl. 68 (2016) 65–79.

[44] T. Xu, D. Gao, P. Dong, H. Zhang, C.H. Foh, H.-C. Chao, Defending against new-flow attack in sdn-based internet of things, IEEE Access 5 (2017) 3431–3443.

[45] C. Hu, K. Hou, H. Li, R. Wang, P. Zheng, P. Zhang, H. Wang, SoftRing: TAming the reactive model for software defined networks, in: 2017 IEEE 25th International Conference on Network Protocols, ICNP, IEEE, 2017, pp. 1–10.

[46] C. Gong, D. Yu, L. Zhao, X. Li, X. Li, An intelligent trust model for hybrid ddos detection in software defined networks, Concurr. Comput.: Pract. Exper. (2019) e5264, http://dx.doi.org/10.1002/cpe.5264, https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5264, e5264 CPE-18-1402.R2.

[47] M. Alsaeedi, M.M. Mohamad, A.A. Al-Roubaiey, Toward adaptive and scalable OpenFlow-SDN flow control: A survey, IEEE Access 7 (2019) 107346–107379, http://dx.doi.org/10.1109/ACCESS.2019.2932422.

[48] T. Hu, Z. Guo, P. Yi, T. Baker, J. Lan, Multi-controller based software-defined networking: A survey, IEEE Access 6 (2018) 15980–15996, http://dx.doi.org/10.1109/ACCESS.2018.2814738.

[49] D.L.P. Sarwar Raza (HP), Open networking foundation northbound interface working group (NBI-WG), 2013, https://www.opennetworking.org/images/stories/downloads/working-groups/charter-nbi.pdf, (Accessed 23 January 2020).

[50] Z. Latif, K. Sharif, F. Li, M.M. Karim, Y. Wang, A Comprehensive Survey of Interface Protocols for Software Defined Networks, 2019, arXiv:1902.07913.

[51] A. Karim, S.A.A. Shah, R.B. Salleh, M. Arif, R.M. Noor, S. Shamshirband, Mobile botnet attacks-an emerging threat: Classification, review and open issues, TIIS 9 (4) (2015) 1471–1492.

[52] A.F.M. Piedrahita, S. Rueda, D.M. Mattos, O.C.M. Duarte, Flowfence: a denial of service defense system for software defined networking, in: Global Information Infrastructure and Networking Symposium, GIIS, IEEE, 2015, pp. 1–6.

[53] N.-N. Dao, J. Park, M. Park, S. Cho, et al., A feasible method to combat against ddos attack in SDN network, in: 2015 International Conference on Information Networking, ICOIN), IEEE, 2015, pp. 309–311.

[54] I.H. Abdulqadder, D. Zou, I.T. Aziz, B. Yuan, W. Li, SecSDN-Cloud: defeating vulnerable attacks through secure software-defined networks, IEEE Access 6 (2018) 8292–8301.

[55] R. Wang, Z. Jia, L. Ju, An entropy-based distributed ddos detection mechanism in software-defined networking, in: Trustcom/BigDataSE/ISPA, 2015 IEEE, vol. 1, IEEE, 2015, pp. 310–317.

[56] P. Phaal, S. Panchen, N. McKee, RFC3176: InMon Corporation's SFlow: A Method for Monitoring Traffic in Switched and Routed Networks, RFC Editor, USA, 2001.

[57] V. Mann, A. Vishnoi, S. Bidkar, Living on the edge: Monitoring network flows at the edge in cloud data centers, in: Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on, IEEE, 2013, pp. 1–9.

[58] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, V. Maglaris, Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments, Comput. Netw. 62 (2014) 122–136.

[59] Y. Lu, M. Wang, An easy defense mechanism against botnet-based DDoS flooding attack originated in SDN environment using sFlow, in: Proceedings of the 11th International Conference on Future Internet Technologies, ACM, 2016, pp. 14–20.

[60] J. Boite, P.-A. Nardin, F. Rebecchi, M. Bouet, V. Conan, Statesec: Stateful monitoring for ddos protection in software defined networks, in: Network Softwarization (NetSoft), 2017 IEEE Conference on, IEEE, 2017, pp. 1–9.

[61] X. You, Y. Feng, K. Sakurai, Packet in message based DDoS attack detection in SDN network using openflow, in: Computing and Networking (CANDAR), 2017 Fifth International Symposium on, IEEE, 2017, pp. 522–528.

[62] Y. Jiang, X. Zhang, Q. Zhou, Z. Cheng, An entropy-based DDoS defense mechanism in software defined networks, in: International Conference on Communicatins and Networking in China, Springer, 2018, pp. 169–178.

[63] K. Kumar, A.L. Sangal, A. Bhandari, Traceback techniques against DDOS attacks: A comprehensive review, in: 2011 2nd International Conference on Computer and Communication Technology, ICCCT-2011, 2011, pp. 491–498, http://dx.doi.org/10.1109/ICCCT.2011.6075132.

[64] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Mach. Learn. 29 (2) (1997) 131–163, http://dx.doi.org/10.1023/A:1007465528199.

[65] T. Kohonen, The self-organizing map, Proc. IEEE 78 (9) (1990) 1464–1480, http://dx.doi.org/10.1109/5.58325.

[66] L. Boero, M. Marchese, S. Zappatore, Support vector machine meets software defined networking in IDS domain, in: 2017 29th International Teletraffic Congress, ITC 29, vol. 3, 2017, pp. 25–30, http://dx.doi.org/10.23919/ITC.2017.8065806.

[67] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, W.-Y. Lin, Intrusion detection by machine learning: A review, Expert Syst. Appl. 36 (10) (2009) 11994–12000.

[68] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa, B. Yang, Predicting network attack patterns in SDN using machine learning approach, in: Network Function Virtualization and Software Defined Networks (NFV-SDN), IEEE Conference on, IEEE, 2016, pp. 167–172.

[69] M. Almseidin, M. Alzubi, S. Kovacs, M. Alkasassbeh, Evaluation of machine learning algorithms for intrusion detection system, in: Intelligent Systems and Informatics (SISY), 2017 IEEE 15th International Symposium on, IEEE, 2017, pp. 000277–000282.

[70] A. Abubakar, B. Pranggono, Machine learning based intrusion detection system for software defined networks, in: Emerging Security Technologies (EST), 2017 Seventh International Conference on, IEEE, 2017, pp. 138–143.

[71] R. Braga, E. Mota, A. Passito, Lightweight ddos flooding attack detection using NOX/openflow, in: Local Computer Networks (LCN), 2010 IEEE 35th Conference on, IEEE, 2010, pp. 408–415.

[72] P. MohanaPriya, S. Shalinie, Restricted Boltzmann machine based detection system for ddos attack in software defined networks, in: Signal Processing, Communication and Networking (ICSCN), 2017 Fourth International Conference on, IEEE, 2017, pp. 1–6.

[73] D. Hu, P. Hong, Y. Chen, FADM: DDoS Flooding attack detection and mitigation system in software-defined networking, in: GLOBECOM 2017-2017 IEEE Global Communications Conference, IEEE, 2017, pp. 1–7.

[74] Q. Niyaz, W. Sun, A.Y. Javaid, A deep learning based ddos detection system in software-defined networking (SDN), EAI Endorsed Trans. Secur. Safety 4 (12) (2017) http://dx.doi.org/10.4108/eai.28-12-2017.153515.

[75] S. Hassas Yeganeh, Y. Ganjali, Kandoo: A framework for efficient and scalable offloading of control applications, in: Proceedings of the First Workshop on Hot Topics in Software Defined Networks, HotSDN '12, Association for Computing Machinery, New York, NY, USA, 2012, pp. 19–24, http://dx.doi.org/10.1145/2342441.2342446.

[76] M. Yu, J. Rexford, M.J. Freedman, J. Wang, Scalable flow-based networking with DIFANE, in: Proceedings of the ACM SIGCOMM 2010 Conference, SIGCOMM '10, Association for Computing Machinery, New York, NY, USA, 2010, pp. 351–362, http://dx.doi.org/10.1145/1851182.1851224.

[77] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, T. Hama, et al., Onix: A distributed control platform for large-scale production networks, in: OSDI, vol. 10, 2010, pp. 1–6.

[78] D. Erickson, The beacon openflow controller, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, HotSDN '13, Association for Computing Machinery, New York, NY, USA, 2013, pp. 13–18, http://dx.doi.org/10.1145/2491185.2491189.

[79] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, R. Sherwood, On controller performance in software-defined networks, in: 2nd USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services, Hot-ICE 12, USENIX Association, San Jose, CA, 2012, https://www.usenix.org/conference/hot-ice12/workshop-program/presentation/tootoonchian.

[80] L. Zhu, M.M. Karim, K. Sharif, F. Li, X. Du, M. Guizani, SDN controllers: Benchmarking & performance evaluation, CoRR abs/1902.04491 (2019) http://arxiv.org/abs/1902.04491.

[81] S. Hamid, N.Z. Bawany, J.A. Shamsi, Recsdn: Resilient controller for software defined networks, Int. J. Adv. Comput. Sci. Appl. 8 (8) (2017) 202–208.

[82] Y. Wang, T. Hu, G. Tang, J. Xie, J. Lu, SGS: safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking, IEEE Access 7 (2019) 34699–34710, http://dx.doi.org/10.1109/ACCESS.2019.2895092.

[83] How the internet was born: The network begins to take shape, 2016, http://theconversation.com/how-the-internet-was-born-the-network-begins-to-take-shape-67904, (Accessed 23 January 2020).

[84] N.J. Kong, Design concept for a failover mechanism in distributed sdn controllers, (Master's thesis), San Jose State University, 2019, https://scholarworks.sjsu.edu/etd_projects/548.

[85] F. Botelho, A. Bessani, F.M.V. Ramos, P. Ferreira, On the design of practical fault-tolerant SDN controllers, in: 2014 Third European Workshop on Software Defined Networks, 2014, pp. 73–78, http://dx.doi.org/10.1109/EWSDN.2014.25.

[86] DDoS threat report 2018 Q2, 2018, https://www.nexusguard.com/threat-report-q2-2018 (Accessed 23 January 2020).

[87] Kaspersky report on DDoS attacks in Q3 2018, 2018, https://securelist.com/ddos-report-in-q3-2018/88617/, (Accessed 23 January 2020).

[88] A. Bhandari, A.L. Sangal, K. Kumar, Characterizing flash events and distributed denial-of-service attacks: an empirical investigation, Sec. Commun. Netw. 9 (13) (2016) 2222–2239, http://dx.doi.org/10.1002/sec.1472.

[89] S. Behal, K. Kumar, M. Sachdeva, Characterizing DDoS attacks and flash events: Review, research gaps and future directions, Comp. Sci. Rev. 25 (2017) 101–114.

[90] S. Wang, C. Chou, C. Yang, EstiNet Openflow network simulator and emulator, IEEE Commun. Mag. 51 (9) (2013) 110–117, http://dx.doi.org/10.1109/MCOM.2013.6588659.

[91] An instant virtual network on your laptop (or other PC), http://mininet.org/, (Accessed 23 January 2020).

[92] OpenDaylight (ODL), https://www.opendaylight.org/, (Accessed 23 January 2020).

[93] Project floodlight, http://www.projectfloodlight.org/floodlight/, (Accessed 23 January 2020).

[94] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, NOX: towards an operating system for networks, ACM SIGCOMM Comput. Commun. Rev. 38 (3) (2008) 105–110.

[95] POX controller, https://openflow.stanford.edu/display/ONL/POX+Wiki.html, (Accessed 23 January 2020).

[96] Ryu SDN framework, https://osrg.github.io/ryu/, (Accessed 23 January 2020).

[97] CAIDA Ddos attack dataset, 2007, https://www.caida.org/data/passive/ddos-20070804_dataset.xml, (Accessed 23 January 2020).

[98] Oct 2016 DYN / DDoS attack, 2016, http://www.red5security.com/, (Accessed 23 January 2020).

[99] B. Cusack, R. Lutui, R. Khaleghparast, Detecting slow ddos attacks on mobile devices, in: The 27th Australasian Conference on Information Systems, Australasian Conference on Information Systems (ACIS), 2016.

[100] V.G.T.D. Costa, S. Barbon, R.S. Miani, J.J. Rodrigues, B.B. Zarpelão, Mobile botnets detection based on machine learning over system calls, Int. J. Secur. Netw. 14 (2) (2019) 103–118.

[101] G. Kirubavathi, R. Anitha, Structural analysis and detection of android botnets using machine learning techniques, Int. J. Inf. Secur. 17 (2) (2018) 153–167.

[102] P. Farina, E. Cambiaso, G. Papaleo, M. Aiello, Are mobile botnets a possible threat? The case of SlowBot Net, Comput. Secur. 58 (2016) 268–283.

[103] M.R. Faghani, U.T. Nguyen, Mobile botnets meet social networks: design and analysis of a new type of botnet, Int. J. Inf. Secur. 18 (4) (2019) 423–449.