Computers and Electrical Engineering 99 (2022) 107706

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

A survey on DoS/DDoS mitigation techniques in SDNs: Classification, comparison, solutions, testing tools and datasets ☆

Bushra Alhijawi [a,*], Sufyan Almajali [a], Hany Elgala [b], Haythem Bany Salameh [c,d], Moussa Ayyash [e]

[a] School of Computing Sciences, Princess Sumaya University for Technology, Amman, Jordan
[b] Electrical and Computer Engineering Department, University at Albany-SUNY, Albany, NY, USA
[c] College of Engineering, Al Ain University, Al Ain, United Arab Emirates
[d] Telecommunication Engineering Department, Yarmouk University, Irbid, Jordan
[e] Department of Computing Information and Mathematical Sciences and Technology, Chicago State University, Chicago, IL, USA

ARTICLE INFO

Keywords:
Software-defined networking
SDN
DoS
DDoS
SDN tools
SDN datasets

ABSTRACT

Software-Defined Networking (SDN) is a modern network approach that replaces the conventional network architecture with a flexible architecture by separating the control plane from the data plane. SDN simplifies network management and monitoring through logically centralized intelligence, programmability, and abstraction. SDN architectures are vulnerable to attacks, such as the Denial of Service (DoS) attack. This article reviews and classifies the research efforts on SDN and DoS. We categorized the research efforts into two groups: solutions to cope with DoS attacks on SDN and SDN-based solutions to tackle DoS attacks on networks. The first group of solutions includes six categories: Table-Entry, Scheduling, Architectural, Flow Statistics, Machine Learning, and Hybrid solutions. Furthermore, the article surveys the tools and datasets considered by the reviewed contributions. The article presents a detailed comparison among reviewed approaches in terms of network devices, network layers involved, DoS attack's types, and targets of attacks.

1. Introduction

The classical architecture of existing networks consists of multiple connected hosts using switches and routers. Switches process and transmit data packets among hosts within the same Local Area Network (LAN). Whilst, routers enable the transfer of data packets between networks. Those forwarding devices are responsible for both the data transmission process and control process. The configuration and network policies in these devices are established and updated by the network administrator. However, the conventional network architecture is not designed to handle complex networks with a huge number of devices connected to the Internet which requires laborious upkeep. According to the Open Networking Foundation (ONF), the classical network architecture has four main limitations:

- **Complex static architecture**. Network administrators must manually reconfigure the parameters in multiple network devices such as switches, routers, and firewalls whenever changes are presented in the network.

☆ This paper is for regular issues of CAEE. Reviews were processed by Area Editor Dr. G. Martinez Perez and recommended for publication.

* Corresponding author.
E-mail address: b.alhijawi@psut.edu.jo (B. Alhijawi).

https://doi.org/10.1016/j.compeleceng.2022.107706
Received 9 April 2020; Received in revised form 11 December 2021; Accepted 6 January 2022
Available online 3 February 2022

- **Inconsistent policies**. In large networks, network administrators may have to update policies and reconfigure thousands of devices that can take a long time to reconfigure them across the whole network.
- **Scalability**. The size of networks has been growing rapidly. Thus, more switches and transmission capacity are added. This by itself is difficult to accomplish due to design and cost factors.
- **Vendor dependence**. The network needs to rapidly employ new services and capabilities to meet users' and business requirements. However, the development process of new services and capabilities is relatively slow since it depends on the product cycle considered by the vendors.

The SDN is adopted as a promising solution to the conventional network architecture limitations related to the fact that the network controlling functions are physically separated from the network forwarding functions. In SDN, the network controller is responsible for the control functions and switches handle the actual data forwarding. Three main features characterize the SDN architecture: logically centralized intelligence, programmability and abstraction [1]. The centralized intelligence enables a global clear view of the network to facilitate network management, maintenance, and enhance the control process [1]. The programmability attribute enables the automated replacement of the network management paradigm whereas the SDN is controlled by software that is provided by either vendors or the network operators [1]. The decoupling of the control plane from the forwarding plane abstracts the network infrastructure toward higher flexibility. Hence, it simplifies the configuration of network devices. In addition, the abstraction attribute enables applications to operate and benefit from network services and capabilities oblivious to the implementation details [1].

The features that characterize the SDN architecture lead to network security challenges and concerns. As the conventional network, the SDN is vulnerable to common attacks such as message modification attack, DoS, spoofing and information disclosure. Further, a new set of attacks are possible in SDN such as unauthorized access of network infrastructure and controller, data leakage (*i.e.*, flow rule awareness and the analysis of packet processing), and malicious applications that install fake rules. The DoS attack has a catastrophic impact on the SDN. It makes either the entire or parts of the network unavailable by crippling the controller or disabling switches. The potential of DoS attacks on SDN has increased since it depends on centralized control and a flow-table [1]. The DoS attack can be launched by flooding the network with packets to drain the control plane bandwidth or by filling up the flow-table of switches.

Several solutions to prevent, mitigate or detect DoS attacks in SDNs have been proposed. Similarly, numerous contributions have utilized SDN features for preventing, mitigating, or detecting DoS attacks on the network. However, to the best of our knowledge, a few systematic reviews are available to properly shape this field and position existing research works and recent progress. Although these systematic reviews have explored the DoS/Distributed DoS (DDoS) mitigation mechanisms in SDN and have attempted to formalize this research field, they have focused on reviewing the contributions to DoS/DDoS mitigation mechanisms in SDN that are deployed in specific environments. Recently, several survey articles reviewed the contributions that have been made in this area. Most authors have focused on reviewing the contributions to DDoS SDN-based mitigation solutions in cloud computing. Whereas, Dong et al. [2] reviewed DDoS attacks in SDN and cloud computing scenarios. They classified the DDoS attacks in SDN according to attack's target; data layer DDoS attacks, control layer DDoS attacks, and application layer DDoS attacks. Besides, they summarized a very few numbers of contributions (*i.e.*, eight articles) that developed to mitigate DDoS attacks against SDN. Sultana et al. [3] reviewed SDN-based network intrusion detection systems that use Machine Learning (ML). Imran et al. [4] provided a comprehensive overview of DoS mitigation methods in SDN and classified the methods into three classes (*i.e.*, blocking, delaying, and resource management methods) based on the methodology used to handle the malicious traffic. Swami et al. [5] focused on studying on DDoS threats prevalent in SDN. Singh and Sunny [6] reviewed and classified the DDoS detection and mitigation methods in SDN according to the utilized techniques for developing the proposed solutions. Table 1 shows an overview of the survey papers that have been published in this research area. Table 2 shows a detailed comparison between this research work and other review papers in terms of covered topics.

In this context, this survey aims to thoroughly review the literature on the advances of DoS/DDoS attacks against SDN and on mitigating DoS/DDoS attacks. It provides a panorama through which readers can quickly understand and step into the field of DoS/DDoS mitigation mechanisms in SDN. This survey directly serves the researchers and professionals who are interested in and involved in this research area. To summarize, the main contributions of this survey are four-folds: (1) We conduct a systematic review for the DoS/DDoS mitigation mechanisms in SDN and propose a classification scheme to position and organize the research works during the period 2013–2020; (2) We provide an overview and summary for the state-of-the-art; (3) We survey the testing tools and datasets used to implement and test the solutions that have been developed; (4) We identify the new future directions in this research area to share the vision and expand the horizons of DoS/DDoS mitigation mechanisms in SDN research.

In this survey, we reviewed sixty-one (61) related papers published during the period 2013–2020 and classified them systematically based on the relationship between SDNs and DoS/DDoS attacks:

- Defense Methods of DoS/DDoS attacks against SDN.
  - Table-Entry-based solutions.
  - Scheduling-based solutions.
  - Architectural-based solutions.
  - Flow Statistics-based solutions.
  - ML-based solutions.
  - Hybrid solutions.

Table 1
An overview of the reviewed survey articles.

Article	Year	#Covered papers	Covered period	Description
[2]	2019	124	2004–2019	– Survey focused on DDoS attacks in SDN and cloud computing scenarios. – Reviewed DDoS detection mechanisms in SDN and cloud computing.
[3]	2019	48	2009–2016	– Survey focused on intrusion detection systems using ML. – Reviewed SDN-based network intrusion detection systems using deep learning.
[4]	2019	32	2013–2018	– Comprehensive survey focused on mitigation mechanisms for DoS attack in SDN. – Classified mitigation mechanisms into three groups; blocking, delaying, and resource management methods.
[5]	2019	56	2016–2018	– Review DDoS detection and mitigation method for defending SDN. – Classified DDoS defence methods into two classes: defense by SDN and defense for SDN.
[6]	2020	70	2014–2020	– Comprehensive survey focused on detection and mitigation mechanisms for DDoS attack in SDN. – Classified mitigation mechanisms into four groups; Information theory-based, ML-based, Artificial Neural Networks (ANN)-based, and other miscellaneous methods.
This research paper	–	91	2013–2020	– A systematic review for the DoS/DDoS mitigation techniques in SDN. – Classified mitigation methods into two classes: defense methods to cope with DoS/DDoS attacks in SDN and DoS/DDoS attacks mitigation solutions using SDN. – Compared the DoS/DDoS attacks mitigation techniques against SDN networks in terms of five aspects: solution type, attack target, tested attack, switch intelligence, and solution layer. – Collected and grouped the testing tools according to usage. – Presented the used datasets in all related research work of SDN and DoS. – Suggested a promising future research directions in this field.

Table 2
Contributions of the reviewed survey articles.

Covered topic	Article					
	[2]	[3]	[4]	[5]	[6]	This research paper
DoS mitigation in SDN	✓	✓ (ML-based)	✓		✓	✓
Using SDN as a solution for DoS				✓		✓
Solution layer						✓
Attack target layer	✓		✓	✓		✓
Tested attack			✓	✓		✓
Switch intelligence						✓
Used Testbed	✓ (Three types)	✓ (Partially)		✓		✓
Used SDN controller		✓ (Partially)		✓	✓	✓
Used SDN switch		✓ (Partially)				✓
Used generating, capturing, and analyzing tools						✓
Used measurements and monitoring tools						✓
Used dataset					✓	✓
Suggest future research directions		✓	✓	✓	✓	✓

- SDN-based solutions for DoS/DDoS attacks.
  - Applications.
  - Used SDN characteristics.

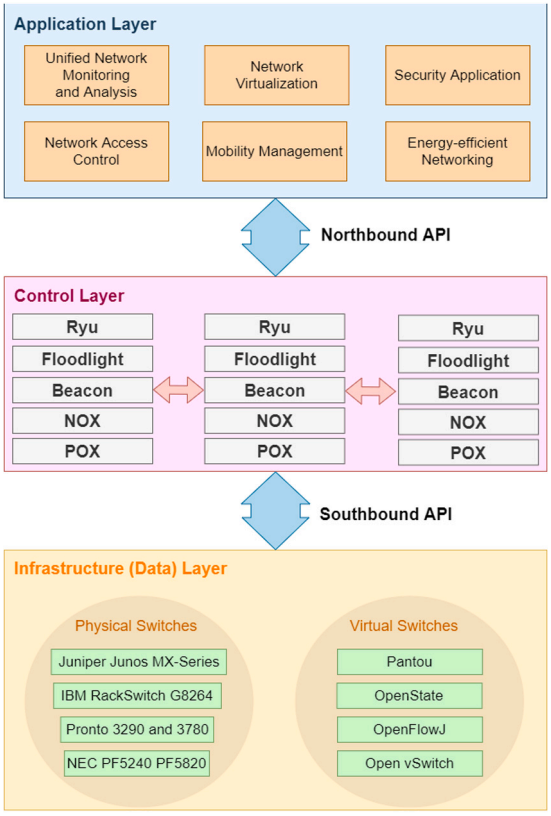

Fig. 1. The SDN architecture.

Furthermore, this survey compared the defense techniques of DoS/DDoS attacks against SDN in terms of five aspects:

- Solution type: Detect, Mitigate or Prevent solutions.
- Attack target: Flow Table, Controller Bandwidth and Controller Resources.
- Tested attack: TCP SYN flood, UDP flood, ICMP flood, etc.
- Switch intelligence: Capable switch or Dumb switch.
- Solution layer: Data layer, Control layer, or Application layer.

Besides, the collected testing tools are grouped into four types according to usage: emulators and experimentation environments, SDN controllers, SDN switches and traffic generation, capturing, analyzing, and performance monitoring tools.

The rest of this article is organized as follows. Section 2 presents a brief overview of SDN architecture and components. Section 3 describes the types of DoS attacks and how to launch them in SDNs. Section 4 presents the methodology followed in this survey. The solutions to prevent, mitigate, or detect DoS attacks on SDN are discussed in Section 5. Section 6 reviewed the contributions that benefited from the SDN architecture for preventing, mitigating or detecting DoS attacks. The tools (*i.e.*, emulators, test-beds, controllers, switches, traffic generators, and analyzer and performance monitors) and datasets utilized in the reviewed articles are presented in Section 7. Finally, Section 8 concludes the article and shows a promising future research directions in this field.

2. SDN architecture and components

SDN is a new network paradigm that has been designed to overcome the problems of conventional network architecture by enabling more agile and cost-effective networks [1]. Fig. 1 presents the SDN architecture model.

The SDN architecture model consists of three distinct layers that communicate through Application Plane Interface (APIs):

- **Application layer**: This layer consists of the end-user applications that exploit the SDN services (e.g. traffic engineering, network security, load balancing, access control management, quality of service, and virtualization) provided by the controller [1]. The control layer communicates with the application layer and vice versa via the Northbound API [1].
- **Control layer** : The control layer is considered the core of network intelligence. The controller is responsible for managing the traffic flow, data handling policies and collecting network information [1,7]. In other words, the controller abstracts the

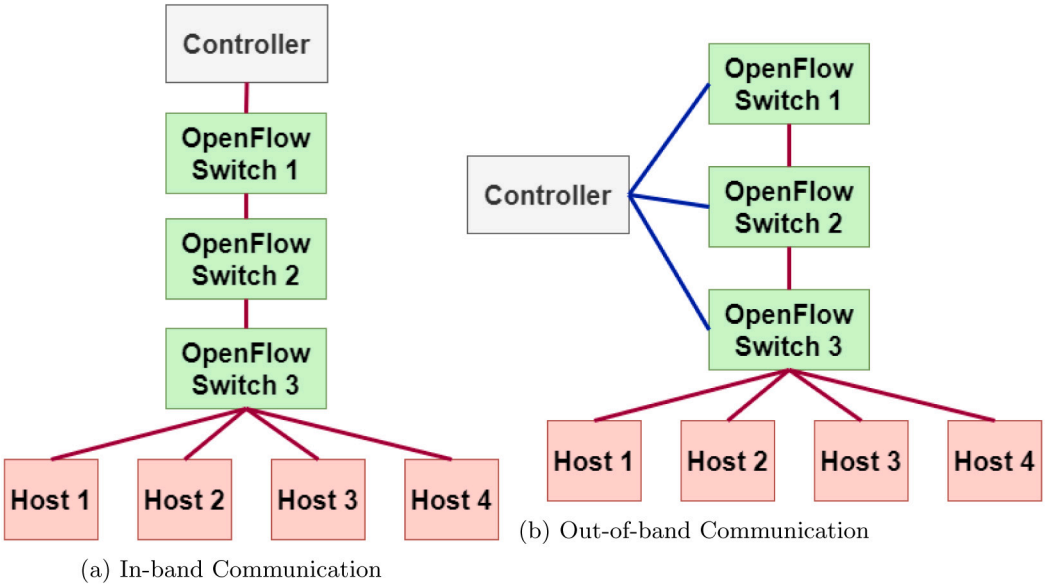

Fig. 2. The Southbound Communication Deployment Scenarios.

network logic, performs several network management tasks and provides the ability to implement new functionality to the network through a programmatic interface.

- **Infrastructure layer (*i.e.*, Data Layer)**: The data layer consists of the network devices (*i.e.*, switches, routers, etc.). The layer is responsible for packet switching and forwarding [1]. An SDN device utilizes the Southbound API to communicate with the controller that abstracts the device as flow tables. Two deployments scenarios of Southbound communication can be found: in-band (Fig. 2(a)) and out-of-band (Fig. 2(b)) communications [7]. In the in-band communication, the traffic between the controller and the SDN devices depends on the dictated flow rules [7]. In the out-of-band communication, the traffic between the controller and the SDN devices does not follow the flow rules. Such communication deployment requires VLAN implementation in order to separate the traffic flow from the communications that depend on OpenFlow rules [7]. The most popular Southbound API is the OpenFlow protocol.

3. Denial of service attack

3.1. Classification and types of DoS

The availability of the network is one of the essential security requirements that ensures timely and reliable access to the network's resources and services. The DoS attack results in losing the availability of the network. The target of this attack may be a specific networking device (e.g. switch) or the entire network. However, the ultimate goal of the DoS attack is to disable the network as well as degrade its performance by overflowing it with a large number of packets. Indeed, a DoS attack is considered a simple powerful attack. In general, the attack packets do not have explicit features that can help in separating a malicious packet from a legitimate one.

The DoS attacks can be classified based on the number of attackers into single-source DoS and multiple-source DoS (*i.e.*, DDoS). A Single-source DoS attack is launched using a single machine. While DDoS attack is conducted by compromising many distributed hosts (*i.e.*, Zombie computers or Botnet) to generate malicious traffic. In addition, the DoS attacks can be classified in terms of several aspects such as network-based attack (TCP SYN flood, UDP flood), target type (host, router, OS, application), the impact of attack (degrading, disruptive) and depletion type (bandwidth, resources).

The most popular DoS attacks are TCP SYN flood, UDP flood, ICMP flood, Smurf attack, Fraggle attack, SNMP amplification attack, CoremeIt attack, and HTTP flood. TCP SYN flood attack exploits the weakness of TCP. The server replies to a large number of SYN requests which are sent by the attacker and waits for the ACK from the attacker. However, the attacker does not send the ACK which leaves the server waiting. This will consume the limited buffer queue of the server that leads to rejecting the new valid SYN requests. UDP flood attack sends a massive number of UDP packets to the server to exhaust it. It is harder to detect UDP flood attack compared with TCP SYN flood since there is no end-to-end communication process between the communicating hosts as in TCP. ICMP flood attack overloads the server with a large number of ICMP echo requests from a spoofed source IP. A smurf attack broadcasts a large number of ICMP packets with the target's spoofed source IP to the network. Other network devices will respond to those packets by sending a reply to the target source IP which exhausts the target resources. The Fraggle attack is similar to the Smurf attack except it uses UDP packets instead of ICMP packets. In a CoremeIt attack, the spoofed bots are divided into two groups.

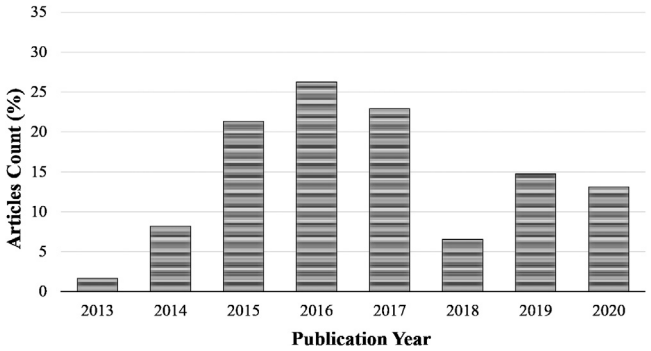

Fig. 3. Temporal Distribution for the Reviewed Papers.

The attacker instructs one of the groups to communicate with the other group leading to generate huge network traffic. HTTP flood attack overloads a web server with a huge number of HTTP requests.

3.2. DoS/DDoS Attacks against SDN

Several vulnerabilities and DoS attacks for SDN have been recently introduced to target switch memory, controller bandwidth, and resources. Switches have limited memory to store flow rules; thus, the attackers focus on filling rules for malicious requests. The switch cannot forward buffered packets due to a full flow table. This attack has a local impact on the switch. Heavy consumption of controller bandwidth and resources results in a large number of new packets. Such an attack increases the delay of installing new flow rules; thus, the switch will not be able to forward traffic from new flows. Hence, this attack has effects on all switches connected to the impacted controller.

In addition, packet injection and Know Your Enemy (KYE) attacks can be conducted against the SDN. A malicious user can inject packets with forged header fields, forcing the switches to send Packet-In messages to the controller. The packet injection attack targets controller bandwidth and resources. Amplified Control Plane Saturation Attack (CPSA) is a powerful DDoS attack on the control plane. In the case of a big SDN network, CPSA generates new unique flows quickly; thus, forcing the switch to route the packet through other network switches, each of which will send a flow rule request to the controller. The KYE allows an attacker to gather information about the SDN configuration of security tools. The KYE attack involves two phases: probing and inference phases. The attacker attempts to trigger the installation of flow rules on the entry switch during the probing phase. Later in the inference phase, the attacker analyzes the correlation between the crafted traffic sent during the probing phase and the installed rules. The attacker can utilize KYE to obtain information related to DoS detection methods and generate DoS floods through the SND during the probing.

4. Methodology

This survey focused on articles published within 2013–2020. The covered literature included various research databases (*i.e.*, IEEExplore, Google Scholar, ACM Digital Library, Springer Link, and ScienceDirect) to identify relevant literature for our survey. Initially, 91 papers were selected. These research papers were sought using a combination of keywords such as; DoS, DDoS, SDN, Flow table, OpenFlow and flooding attack. At this stage, keywords and titles gained the most emphasis while the abstracts gained less emphasis. Next, based on the abstract scanning, 21 articles were excluded resulting in 70[1] research papers considered in this survey. The selected papers were published within the period 2013–2020 most of them between 2015–2017 (*i.e.*, 70.5%). Fig. 3 shows the temporal distribution for the reviewed papers. 65.6% of the selected articles are conferences papers and 49.2% have been published in journals. In term of citation counts (Table 3), 21.5% of the research papers had a high citation rate and most research works had more than 60 citations. The mean citation count was 52.2 and the median was 31. Google Scholar was used for citations counts in April/2020. In general, the reviewed research papers were full papers with (42.9%) of 4 and 7 pages (Table 4) and 22.9% of 12 and 15 pages.

This survey classified the contributions published in this research area into two main categories; solutions for DoS/DDoS attacks against SDN and SDN-based DoS/DDoS attack solutions. The first main category of solutions for DoS/DDoS attack against SDN includes the contributions that proposed solutions for DoS/DDoS attack against SDN main components (*i.e.*, switch and controller). The articles belong to this group are classified into six sub-categories according to the nature of the solutions. Sub-categories include Table-Entry-based, Scheduling-based, Architectural-based, Flow Statistics-based, ML-based, and Hybrid solutions. For each sub-category, the contributions are compared based on solution type (Detect/Mitigate/Prevent), attack target (Flow Table/Controller Bandwidth/ Controller Resources), tested attack (TCP SYN flood/UPD flood/etc..), switch intelligence (Capable/Dumb) and solution

[1] The complete list of references that were used in this study can be found in website: https://doi.org/10.13140/RG.2.2.28931.63527.

Table 3
The distribution of researches based on citation count.

Citation count	Percentage of researches
0–10	24.29
11–20	18.58
21–30	8.58
31–40	18.58
41–50	4.29
51–60	4.29
>60	21.43

Table 4
The distribution of researches based on page count.

Page count	Percentage of researches
1–3	5.72
4–7	42.86
8–11	8.58
12–15	22.86
16–19	11.43
20–23	2.86
>23	5.72

layer (Data/Control/Application). The second main category of SDN-based DoS/DDoS attack solutions includes the proposed solutions which exploit the SDN architecture to defeat the DoS/DDoS attack against network services or other environments such as cloud computing and cross-domain networks. Those articles are compared and classified according to the application and the used SDN characteristic.

In addition, the tools and datasets utilized by the reviewed research works were captured and detailed. Four types of tools are considered in this survey: emulators and experimentation environments, SDN controllers, SDN switches and traffic generation, capturing, analyzing, and performance monitoring tools.

5. Solutions to cope with denial of service attacks in SDNs

The flow table stores the flow rules generated by the controllers for forwarding and controlling the packets in SDN. Thus, the flow table has a critical role in SDN. However, this table has a limited size that makes it vulnerable to DoS/DDoS flooding attacks. In SDN, each flow does not match an existing rule in the SDN switch table needs a new flow entry. Therefore, the flow table is vulnerable to overflow. At the same time, for each unknown flow, the data layer will communicate with the controller to install a rule that matches the unknown flow. This creates a bottleneck during a DoS/DDoS attack as it exhausts the controller bandwidth and network resources. Several solutions were proposed in order to detect, mitigate, and prevent DoS/DDoS attack on SDN. We classified those solutions based on the nature of the solution into six categories: Table-Entry-based, Scheduling-based, Architectural-based, Flow Statistics-based, ML-based and Hybrid solutions. In addition, the solutions belong to each category are analyzed and compared in terms of solution type (Detect/Mitigate/Prevent), attack target (Flow Table/Controller Bandwidth/Controller Resources), tested attack (TCP SYN flood/UPD flood/etc..), or switch intelligence (Capable/Dumb) and solution layer (Data/Control/Application). Table 5 presents a comparison between the reviewed articles.

5.1. Table-entry-based solutions

Several solutions have been proposed to control flow rule installation and table entry replacement policies. Floodguard is a scalable, efficient, and lightweight DoS prevention framework that is protocol-independent. This system consists of two modules (*i.e.*, proactive flow rule analyzer and packet migration) which are added to the existing control layer modules. The proactive flow rule analyzer is responsible for generating and updating the proactive flow rules and dynamically loading the rules into the flow tables. Proactive flow rules can be populated by a policy based on the controller status. The packet migration is responsible for handling the flooding packets without sacrificing legitimate packets. Once the packet migration module detects a DoS attack, the unknown flows are temporarily cached after installing the proactive flow rules. Qian et al. [8] introduced a systematic method to mitigate flow table overflow attack. Their solution consists of two components; general strategy and eviction algorithm. Two general strategies are adopted to control flow entries in the flow table that are rate-limiting and timeout adjustment. They considered three rate-limiting cases; port-based where the traffic rate limit is per port, controller-based where packets rate limit is per controller, and rules count-based where the controller is limited in the number of rules it can install into the flow table in a short period time. The eviction algorithm is a reactive and dynamic event-driven method that validates the traffic source by querying the log. In case the traffic source does not pass the validation step, it is added to the blacklist and its flows that exceed the rate limit threshold is removed. The core idea behind the PacketChecker is to give the controller the ability to distinguish between malicious packets

Table 5
Summarization of DoS/DDoS against SDN Solutions.

Ref	Class	Goal	Attack target			Tested attack/s	Switch intelligence	Solution layer	Comments
			DL	CB	CR				
[8]	Table-Entry	M	✓			*	Dumb	Control	
[9]	Table-Entry	M	✓			*	Dumb	Application	Malicious flows are directly dropped
[10]	Scheduling	M			✓	UDP flood	Dumb	Control	No need for any modifications of switches
[11]	Scheduling	M			✓	*	Dumb	Control	The flow requests of each switch are isolated and different allocation strategies is assigned
[12]	Architectural	D, M			✓	*	Dumb	Control	Backup controllers pool
[13]	Architectural	M		✓	✓	*	Dumb	Control	Parallel flow rule installation
[14]	Flow Statistics	P	✓			TCP SYN flood	Dumb	Data	
[15]	Flow Statistics	D	✓			TCP SYN flood	Capable	Data	Reduce the overhead of flow collection
[16]	Flow Statistics	D		✓		TCP SYN flood	Capable	Control	Use the ratio between the capacity and the number of flows of a link
[17]	Flow Statistics	D, M	✓			*	Dumb	Control	Identifying the attacker the header fields
[18]	Flow Statistics	D, M		✓	✓	TFTP, HTTP flood	Capable	Data	Use the monitoring ability of stateful SDN
[19]	ML	D	✓		✓	TCP SYN, UDP,ICMP flood	Dumb	Application	
[20]	ML	D, M	✓			TCP SYN, UDP, ICMP flood	Dumb	Application	– No need to add hardware – A detection scheduling mechanism
[21]	ML	D	✓		✓		Dumb	Control	Access control mechanism
[22]	Hybrid	D	✓			*	Dumb	Control	– Flow Statistics –ML
[23]	Hybrid	D, M	✓		✓	TCP SYN, ICMP flood	Dumb	Control	– Table-Entry –ML (SVM,SOM)
[24]	Hybrid	D, M	✓			TCP SYN, UDP, Smurf, ARP flood Land, Teardrop	Dumb	Control	– Table-Entry –Flow Statistics –ML

* Not Available, D: Detect, M: Mitigate, P: Prevent, DL: Data Layer, CB: Controller Bandwidth, CR: Controller Resources.

from normal ones, thus, it can drop the malicious packets before being processed by other modules in the controller. PacketChecker creates a mapping table for the MAC address and switch port (*i.e.*, switch data path ID and the source packets port number). This table is dynamically maintained in a real-time manner. Using this table, the controller can validate the received packet via querying the table. The controller flags a packet as legitimate if the table contains the MAC addresses and port information of this packet. Otherwise, the controller flags the packet as malicious and drops it. The packet-drop action is controlled by setting a hard and idle timeout for the flow rules. Sharing the unused space in the flow table of other switches with a switch under DoS/DDoS attack is one of the utilized strategies to mitigate the effect of the attack. In that context, Bhushan and Gupta [9] exploited this strategy to introduce new solutions. Bhushan and Gupta [9] implemented DDoS solution that exploits the queuing theory-based mathematical model to represent the flow table-space during the normal condition. Their solution maintains two databases; flow table status and blacklist. The flow table status stores data related to the current status of flow tables for all SND switches in the network. The blacklist includes records related to the IP addresses of attack sources which previously launched DDoS attack on the SDN. The controller installs a redirection rule at the attacked switch to redirect particular flows to the nearest switch with a sufficient amount of unused flow table space. The switch to receive redirected flows is determined based on the analysis of the flow table status. Rules with wildcards are utilized to reduce the number of redirection rules through redirecting an aggregation of multiple flows. At the same time, the controller installs a wildcard rule at the attacked switch to directly drop the malicious flows. The proposed solution by Bhushan and Gupta [9] redirects only the legitimate flows to the peer switches and the malicious flows are directly dropped. Other solutions focused on redirecting both the legitimate and malicious flows to the peer switches. DosDefender is a prevention mechanism for DoS attacks that involves three modules; Port Management, Attack Detection, and Flow Rule. The Port management module is responsible for preparing the input (*i.e.*, identifies the incoming packet) for the attack detection module.

The attack detection module validates the incoming packet, drops the malicious packet, and triggers the flow rule installing module. The flow rule installing module is responsible for mitigating the DoS attack by installing flow rules to the switch that connects the malicious hosts.

5.2. Scheduling-based solutions

Prioritizing and scheduling flow requests to be processed by the controller is one of the proposed strategies to defeat DoS/DDoS attacks on SDN. The reviewed solutions implemented a scheduler module in the control layer to manage flow requests processing.

Another scheduling-based solution is a multi-layer fair queueing method proposed by Zhang et al. [10]. Their idea is to maintain a small number of queues, where each queue is corresponding to a switch in the normal condition. In case the queue size exceeds a threshold, it is dynamically expanded multiple sub-queues. Once the all sub-queues sizes drop below another threshold, they are aggregated into a single queue. Yan et al. [11] presented a multi-queue scheduling method depending on the time slice allocation strategy. Their solution consists of two modules: DDoS detection and MultiSlot algorithm modules. They required to modify the flow processing mechanism of the controller by en-queuing the flow requests of each switch into its corresponding logical queue. The DDoS detection module detects DDoS attack on SDN and the extent of a switch being attacked. Once DDoS is detected, the controller activates the time slice allocation strategy to determine flow requests' processing power for each logical queue based on the decision of DDoS detection module. Other scheduling-based solutions used a single request processing queue logically divided into sub-queues, each corresponding to a switch; thus, the allocation of controller processing capacity can be isolated for each switch.

5.3. Architectural-based solutions

The set of defeating solutions against DoS/DDoS attack related to the hierarchies and roles of SDN components are called architectural solutions. This category includes the least number of reviewed research works.

Etaiwi et al. [12] proposed a solution to consider backup controllers pool to be used in case of any working controller has been attacked. Their solution is designed to handle DoS attack against the SDN of distributed controllers. Each back-controller monitors at least one primary online controller. The communication between the back-controller and the active controllers depends on protocols including a mapping algorithm, heartbeat messages, synchronization messages, take-over process, and protective mode. Imran et al. [13] proposed a parallel flow installation model by modifying the controller to be capable of installing a parallel flow rule instead of the linear installation. Specifically, they modified the module that is responsible for handling incoming packets to (1) generate a list of all switches that lie on the route from the source to the destination, (2) send Flow-Add messages to them without receiving requests to install flow rules from them. Their model addressed several problems such as additional hardware requirements, switch modification, additional delay in route establishment, information loss, and additional control packets. Other solutions adopted hierarchical role-based controller architecture and the Moving Target Defense (MTD) strategy that consists of a multi-controller pool. The core idea behind the hierarchical role-based controller architecture is to have multiple controller architectures to boost the controller's capacity and decrease flow processing capacity. while, the multi-controller pool includes one master controller (*i.e.*, working as online) and equal controllers (*i.e.*, working as offline). MTD manager monitors the bandwidth and traffic load of the master controller. Once a blind DDoS attack is detected, the master controller populates defense configuration instructions to the switches.

5.4. Flow statistics-based solutions

The contributions [14–18] developed DoS/DDoS defeating mechanisms based on statistical models that depend on frequency of different captured features, correlation matrix, entropy, and chi-squared goodness-of-fit test.

Dao et al. [14] introduced a solution that depends on the statistical analysis of network users' behavior. In the normal condition, the solution computes the minimum number of packets per connection and the mean number of connections for active users. In addition, a temple table is added to the controller to store the flow's source IP address with a corresponding counter to track the number of arrived packets from this source. In the attack condition, the controller treats all incoming packets as malicious packets by creating a new specific entry with a hard and idle timeout. The timeout of this entry is set to values smaller than those of legitimate flows. The controller updates the information related to the flow's source IP address in the temple table. When the received packet counter of a particular source IP reaches the defined average number of connections, the controller analyzes its flow traffic characteristic by requesting the average number of packet counter. Later, the average number of packet counter is compared with the minimum number of packets per connection of the source IP. In case it is greater, the source address transmitted legitimate flows and the controller modifies all hard and idle timeout of its existing flow entries to normal value. Otherwise, the flow traffic is considered as malicious and the controller installs a dropped rule for the IP address. The basic idea of the proposed solution by Piedrahita et al. [16] is the switches monitor their traffic to detect congestion conditions and notify the controller. Based on the received flow statistics, the controller command to limit the bandwidth usage of the attacked switches. Another solution depends on the consumption bandwidth called Flowsec. Flowsec consists of an application that requests flow statistics from the switch in order to compute controller bandwidth usage dynamically. Once the bandwidth consumption exceeds a predefined threshold, the flow rate is limited. Durner et al. [17] presented a solution that utilizes a table of counters with the fixed header fields as columns. These fixed header fields are extracted from the malicious flows. During the attack, the malicious flows are analyzed to extract the header fields and increased its corresponding entries in the table. Note that the values of these entries will enormously increase which is a clear

indication of DoS attack. The controller installs flow rules to treat the malicious flows that contain the fixed headers of the attacker. Also, Daisy is a DDoS detection and prevention method that installs a specific flow rule to blocks the suspicious traffic for a short time. Later, the flow is considered as malicious and blocked it for a longer period when the attack continues. Avant-guard includes an extension to the data layer in order to mitigate the impact of DDoS on the controller bandwidth. Their solution consists of connection migration and actuating trigger modules. Connection migration module provides the switches with capabilities to differentiate the sources that complete a TCP connection from sources that will not. Only the flow requests that complete handshakes are forwarded to the control layer. Actuating trigger module depends on data layer statistics module to activate a flow rule under some predefined conditions. Those predefined conditions help the controller in managing flows without delay.

Several research works employed entropy in their proposed solutions. Entropy measures randomness. The core idea behind using entropy for DoS/DDoS attack detection is to measure randomness in the incoming flows. The lower the randomness the higher possibility of attack condition. Wang et al. [15] solution includes designing a flow statistics process and an entropy-based model for detecting DDoS attacks. The flow table is extended to include the packet counter which facilities flow monitoring process. The anomaly detection technique depends on entropy calculation and runs in every edge switch. The entropy is computed based on the frequency of each IP address. Boite et al. [18] proposed a solution for detecting and mitigating DDoS using in-switch processing capabilities. The traffic is monitored to capture pertinent features (e.g. IP src, IP dst, port src, port dst). Later, those features are used as input to an entropy-based anomaly detection algorithm. Other entropy-based solutions classified the incoming packets based on the destination IP address and window size, exploited the collected destination IP per each observation window to compute the probability of occurrence of the destination IP for each packet in the observation window, and analyzed the distribution of the frequency of occurrence of source and destination IPs.

5.5. Machine learning-based solutions

ML methods focus on automatically building and training a model to be used later for detecting and mitigating DoS/DDoS attacks in SDNs. This model is trained using a training dataset. The dataset includes a collection of data instances that are defined using a set of features and the associated labels. Several ML methods such as Support Vector Machines (SVM), ANN, graph-based, K-Nearest Neighbor (KNN) and clustering methods have been utilized in order to detect and/or mitigate DoS/DDoS attack in SDNs.

SVM is one of the most popular supervised learning methods for classification. One of the key features of the SVM is the ability to learn from a few samples. Basically, SVM tries to find an optimal linear separation (largest margin) that separates the considered classes. ANN is a bio-inspired supervised learning method for classification. Generally, ANN learns to perform tasks though considering training instances without requiring any task-specific rules. An ANN depends on a collection of connected units called artificial neurons. The ANN classifier has been widely employed for DoS/DDoS detection in conventional networks as well as in SDN. Deep learning is deep structured learning classifier based on ANN. Several solutions employed classification methods such as KNN, SVM, ANN and Navie Bayes to compare the current flow traffic features with existing attacks patterns modeled as a graph, classify the connections into legitimate and malicious instead of classifying traffic flow, . Niyaz et al. [19] developed a DDoS detection solution using deep learning. Their solution consists of three modules; traffic collector and flow installer, feature extractor, and traffic classifier. Traffic collector and flow installer modules are responsible for extracting several packet's header fields to identify flow. Based on a timer function, the feature extractor module is triggered to extract features from the packet headers list collected by the traffic collector and flow installer modules. Traffic classifier module utilizes the extracted features set as input to the deep learning model to classify the traffic into normal or malicious flows. Back Propagation Neural Network (BPNN) is a type of ANN that consists of computed information forward propagation process and error information backpropagation process. Cui et al. [20] adopted the BPNN to detect the DDoS attack. Their solution includes attack detection, trigger, traceback and mitigation modules. Initially, any abnormal representation of packet-in flows enforces the trigger module to activate the detection module. The detection module captures the suspicious flows and extracts its features to be used in the detection process. Once the DDoS attack is detected, the traceback module traces the attack path and the originating switch. Subsequently, the mitigation module deletes the malicious flow entries. Self-Organizing Map (SOM) is a type of ANN that produces a low dimensional space through unsupervised learning. The main difference between SOM and other ANN types are that SOM applies competitive learning instead of error-correction learning. SOM has been employed in the solution proposed by Wang and Chen [21]. Their solution incorporates an access control mechanism and a classification mechanism. The access control module is responsible for identifying the traffic source to forward only the legitimate flows. The classification module extracts the most representative features from the flow entries of all switches and ranks them to select features, which improves the classification accuracy.

The clustering-based DoS/DDoS mitigation methods monitor the performance of nodes instead of analyzing the flow traffic. In addition, it reduces the impact of DoS/DDoS impact without using any resource out of the cluster. A DDoS detection methods using KNN and k-means for SDN is proposed in the literature.

5.6. Hybrid solutions

The hybrid solutions combine at least two of the aforementioned solutions types. The flow statistics-based mechanisms are widely employed in hybrid solutions.

FlowSec and Blackbox are two DoS defense methods rely on the flow statistics either to detect or mitigate DoS attacks against SDN. FlowSec is a detection and mitigation module that dynamically monitors the controller bandwidth usage. In the case of DoS detect, then the controller installs a rate-limiting rule on one or more switches. Blackbox is a strategy adopting Finite State Machine

Table 6
SDN-based DoS/DDoS Solutions.

Ref.	Goal	Application	Used SDN characteristics
[25]	D, M	General	– Traffic analysis – A centralized network view using SDN controller
[26]	D	Multiple Network Domains	Decoupling of Control plane from the data plane
[27]	D, M	Smart City	– A centralized network view using SDN controller – Network Programmability

* Not Available, D: Detect, M: Mitigate, P: Prevent.

(FSM) to determine a suitable response. The flow statistics are analyzed and fed to FSM. Note that Blackbox produces different responses, which fools and blocks attackers. Flooddefender is a scalable and protocol-independent DoS attack solution. Their method includes three techniques (*i.e.*, table-miss engineering, packet filter, and flow rule management) to mitigate DoS attack against flow table, controller bandwidth and resources. The table-miss engineering technique shares the unused space in the flow table of neighbor switches with the victim switch. The main objective of the table-miss engineering technique is to save the bandwidth of the attacked switch. The packet filter technique exploits the flow statistics and extracted features to filter the packets using two filters; frequency-based and traffic-based filters. The traffic-based filter employs SVM to classify the traffic. The packet filter technique is responsible for protecting the controller computational resources. The flow rule management technique aims to defeat the DoS attack against the flow table by installing monitoring rules in the attacked switch to eliminate most of the useless flow entries in the flow table.

A hybrid solution combines the flow statistics and ML is proposed by Jankowski and Amanowicz [22]. Their method consists of four modules; flow bundle, integrator, features generator and ML-based classifier. The flow bundle module captures and extracts flow statistics form the incoming traffic. The integrator analyzes and processes the extracted flow statistics in order to generate additional features. The flow bundle and Integrator modules form the features generator module. The output of the features generator module is used as input to SOM classifier which is responsible for detecting the malicious flow. Phan et al. [23] presented a hybrid solution that combines table-entry-based method and ML-based method to protect switch and controller resources. The core idea is that the SVM classifies the incoming flow to either normal or malicious flow. In case the traffic is classified as normal, the idle-timeout adjustment algorithm sets flow idle-timeout for each new flow entry. Otherwise, the policy enforcement module enforces the controller to populate drop and delete rules. Li et al. [24] presented a detection and mitigation solution of DDoS attack. The proposed method is comprised of several modules. This includes a features extraction module to extract features from network traffic and prepares a feature matrix depending on the extracted features set. The deep learning DDoS detector module is trained to detect malicious packets based on the feature matrix. The information statistics module extracts the features of the detected attack traffic along with the frequency of the features. According to the collected statistics, the flow table generator module updates the SDN switches with the flow entries and priorities of which ones to drop. Another solution included an adaptive pooling based sampling approach in the data plane to manage heavy traffic flows. Also, a snort and stacked auto-encoders method has been developed to detect the DDoS attack.

6. DoS Mitigation solutions using SDN

Recently, SDN architecture has become one of the promising solutions for the future Internet security issues. Several contributions have been exploited SDN to develop detection, mitigation and prevention mechanisms against DoS/DDoS attacks that are targeted towards a specific service such as cloud computing, data centers, legacy networks, and smart city. Table 6 presents a comparison between the reviewed articles it terms of SDN characteristics.

Chin et al. [25] presented a DDoS detection and mitigation collaborative method that employs multiple monitoring sensors distributed over a network and attack correlators. OrchSec is a Orchestrator-based architecture that employs network monitoring and SDN control functions for developing security applications. This architecture consists of orchestrator, orchestrator agent, data layer, network monitoring and control layer. The orchestrator component is security applications. While, the orchestrator agent is an application installed on each SDN controller to connect the orchestrator with the controller. Zhu et al. [26] proposed a cross-domain DDoS attack detection method using cryptography and KNN classifier. The cryptography is utilized to ensure the confidentiality (*i.e.*, data privacy) during the transmission process between two servers and SDNs controllers. Chen et al. [27] presented a statistics-based DDoS traceback scheme for smart city using SDN. Their method analyzes the fluctuation of network traffic to build traceback scheme based on an anomaly tree. Other solutions reduced the impact of DDoS attack against service by adaptively redirecting the attack traffic across multiple resource replicas in the network, blocked botnet-based attacks by monitoring the flows in each flow table, benefited from the centralized control and programmability features of the SDN to detect and mitigate DDoS attacks against the cloud, and compared the new flow with legitimate traffic instances. Other solution benefited from the SDN characteristics to mitigate DDoS attacks in SDN-based internet service provider networks and IoT traffic by employing ML algorithms and entropy-based methods.

Table 7
Classification of Reviewed Articles based on the Utilized Tools.

Tool		Table-Entry-based	Scheduling-based	Architectural-based	Flow Statistics-based	ML-based	Hybrid	SDN-based
Testbed	Mininet	[8,9]		[12,13]		[20,21]	[22]	[27]
	OPNET				[14]			
	NetFPGA				[15]			
	FITS				[16]			
	MAGI framework				[16]			
	Spirent contracting tools (TestCenter (C1))						[24]	
	GENI Experimenter							[25]
	Metasploitable2		[10]					
	OMNeT++				[17]			
Controller	NOX					[21]		
	POX	[9]		[13]	[16]	[19]	[23]	
	Floodlight		[10]	[15]			[28]	
	Beacon	[8]						
	Ryu			[13]		[20,21]		[27]
	OVS			[12]				
	OpenDayLight		[10]					
	FlowVisor	[8]						
Switch	Open vSwitch	[8]		[12]	[15,18,29]	[19]	[23,24]	[25]
	OpenState				[18]			

7. Tools and datasets

The reviewed articles tested their proposed methods using several tools that are testbeds, controllers, switches, traffic generation and analysis tools, and performance monitoring and reporting tools. Ten emulators and experimentation environments were used in the referenced papers (Section 7.1). In addition, the research techniques discussed in this survey used eight types of SDN controller presented in Section 7.2. Two types of SDN switches were considered in the reviewed articles (Section 7.3). Table 7 presents the distribution of the reviewed research paper according to the utilized testbed, controller and switch. Ten traffic generators and analyzer tools and two performance monitoring tools were utilized in their experiments (Section 7.4). In addition, the datasets that have been utilized to train and test ML based methods are detailed in Section 7.5.

7.1. Emulators and experimentation environments

Mininet[2] is a network emulator that allows virtualizing several network devices such as hosts, SDN switches, SDN controllers, and links. Several controllers are integrated with Mininet such as POX, NOX, Floodlight and Ryu. Mininet supports OpenFlow switches, Open vSwitch. Mininet is considered as one of the most popular network emulators that support research, development, learning, prototyping, testing, and debugging the network. Compared with other network virtualization tools, Mininet is more scalable, faster to boot, and provides more bandwidth. Mininet has the best characteristics of other system virtualization tools, hardware testbeds and simulators:

- Provides simple, flexible, quickly reconfigurable, restartable and inexpensive network experimentation environment for developing OpenFlow-based applications.
- Allows multiple developers to work concurrently and separately on the same network.
- Connects easily to real networks.
- Provides an extensible Python-based API for developing OpenFlow-based network.
- Supports several types of tests such regression tests.
- Enables virtually testing of the complex network topologies.
- Enable easy, simple and quick creation of arbitrary custom topologies.
- Enables sharing the created prototypes easily with other collaborators.

OPNET[3] is a tool to simulate the behavior and performance of any network type including SDN networks. It uses a hierarchical model to define the network. The top level includes network topology. The data flow models and network nodes are defined in the next level. OPNET supports C and C++ languages and only available in commercial form.

NetFPGA[4] is an open source platform that enables students and researchers to network systems prototypes. It focuses on high-performance systems that are accelerated via hardware. The NetFPGA uses standard computer-aided design tool flows to implement

[2] http://mininet.org/.
[3] https://www.riverbed.com/gb/products/steelcentral/opnet.html.
[4] https://netfpga.org/.

the circuits that run on field programmable gate arrays. This platform consists of three components: hardware, gateware, and software. The hardware component includes Xilinx Virtex-II Pro FPGA, Xilinx Spartan FPGA, two Static RAMs, double data rate SDRAM devices. The official NetFPGA is integrated with two gatewares, IPv4 router and 4-port NIC.

Future Internet Testbed with Security (FITS)[5] is a testbed for network experimentation that brings together the most renowned universities. FITS provides an experimentation environment infrastructure for network testing using a hybrid system based on two different virtualization approaches, Xen and OpenFlow. It allows researchers and developers to configure, monitor and test their network in different environments which facilitates results comparison, analysis and choosing the new protocols and mechanism for network modification. FITS tool is based on a service-oriented architecture that is available through a web interface. FITS creates a geographically distributed network of experimentation environment, measurements and performance evaluation among 14 universities. Each participating university builds its own set of nodes with local policies to run experiments (*i.e.*, island). FITS interconnects the islands of participating universities.

Montage AGent Infrastructure (MAGI) framework[6] is a system that allows for expressing and completely automating processes and procedures for deterministic control over the various components in an experiment (e.g. networked elements and agents). MAGI framework, developed by DeterLab, provides scalable and highly controllable testbeds for experimentation.

Spirent contracting tools (TestCenter (C1))[7] is a testbed that delivers high performance with deterministic answers. Spirent C1 enables emulation of complex network topologies and traffic conditions. Also, it minimizes the risk, testing time and cost by mirroring actual network scenarios and traffic patterns. Spirent C1 tool enables researchers and developers to customize network traffic and to simulate common attacks such as DDoS attack (e.g. SynFlood, PingofDeath, Smurf, UDPFlood and Teardrop).

Global Environment for Network Innovations (GENI)[8] is an open infrastructure that has been designed to support at-scale networking and distributed systems research and education. GENI has been utilized for research in future internet architectures, SDN, novel protocol suites, cloud computing and 4G wireless networks. GENI is suitable for large-scale experiment infrastructure and detailed programmability of all network resources and instrumentation and measurement tools.

Metasploit framework,[9] an open source project, for extensive security testing and auditing. Metasploitable 2 is a virtual machine with the Ubuntu operating system for testing security tools and demonstrating common vulnerabilities. This virtual machine enables the users conducting security training, security tools testing, and practice common penetration testing techniques.

Objective Modular Network Testbed in C++ (OMNeT++)[10] is a network simulation platform based on C++ simulation library and consists of hierarchically nested modules. Mainly, OMNeT++ allows modeling network systems and traffic according to need. In addition, it allows evaluating the performance of the proposed software and network systems. OMNeT++ runs on all common operating systems. OMNeT++ was designed to meet the following requirements:

- Large-scale and hierarchical models simulation.
- Facilitate visualizing and debugging of simulation models.
- Modular and customizable simulator.
- Open data interfaces.
- Provide an integrated development environment.

7.2. SDN controllers

NOX controller is the original OpenFlow-SDN controller that supports C++ language. Initially, Nicira Networks developed the NOX controller and it is currently developed by the open source community. This controller is pre-installed and integrated with Mininet. NOX supports only Linux platforms with fast asynchronous I/O. Three main sample components are included in NOX controller; topology discovery, learning switch and network-wide switch.

POX controller[11] is an open source OpenFlow-SDN controller that supports Python language. This controller can be used by Mininet remotely since it is integrated with Mininet. POX is utilized for faster development, implementing and prototyping of network applications. In addition, it provides the users with the ability to run several applications (e.g. switch, firewall, load balancer, and hub). Several communication protocols such as OpenFlow, ForCES and OVSDB can be utilized to carry the communication between POX controller and switches. POX is supported on different platforms, specifically, Linux, Mac OS and Windows. Moreover, POX controller can be utilized as a basic SDN controller simply through using the POX stock components[12]. POX stock components consist of stock components that provide core functionality. However, POX controller has not a GUI interface.

Floodlight controller[13] is an Apache-licensed open source OpenFlow-SDN controller that supports Java language. As NOX controller, Floodlight supports only Linux platforms. Floodlight controller can be used by Mininet remotely. This controller includes

[5] https://www.gta.ufrj.br/fits/.
[6] https://deter-project.org/.
[7] https://www.spirent.com/products/testcenter.
[8] https://www.geni.net/.
[9] https://www.metasploit.com/.
[10] https://omnetpp.org/.
[11] http://www.noxrepo.org/pox/about-pox/.
[12] https://openflow.stanford.edu/display/ONL/POX+Wiki#POXWiki-StockComponents.
[13] http://Floodlight.openflowhub.org/.

a set of modules in which each module provides a service for both the other modules and the control logic application via Java API or a REST API. The modular architecture of the Floodlight controller facilitates the task of enhancing and extending it. In addition, Floodlight supports networks which consist of OpenFlow-enabled switches connected through non-OpenFlow switches.

Beacon controller[14] is an open source OpenFlow-SDN controller that supports Java language. The Beacon consists of three main core functions; topology discovery, device manager and routing. This controller showed a high performance since it is a multithreaded implementation. One key feature of Beacon controller is the support of compile-time modularity, start time modularity and runtime modularity without shutting down the Beacon process.

Ryu controller[15] is an open source component-based OpenFlow-SDN controller that supports Python language. Ryu includes a collection of built-in components. These components are developed with well-defined API that provides the developers with the ability to change, extend and compose the components for creating new customized controller applications. Ryu supports several SDN protocols such as OpenFlow, Netconf and OF-config.

OVS controller is a simple OpenFlow-SDN controller able to manage any number of switches. This controller utilizes OpenFlow-protocol to carry communication with switch casing them to function as L2 MAC-learning switches. OVS controller is not eligible for a OpenFlow production deployment. However, the OVS controller is suitable for initial testing of OpenFlow-SDN.

OpenDayLight controller[16] is the largest open source OpenFlow-SDN controller platform that supports Java language. OpenDayLight supports various communication protocols to carry the communication between OpenDayLight controller and switches such as OpenFlow, OVSDB, NETCONF, and BGP. This controller is integrated with Mininet, thus, it can be utilized remotely. The OpenDayLight controller can be deployed with any platform; Linux, Mac OS, and Windows.

FlowVisor is an SDN controller that allows virtualization of network. In order to virtualize a physical network, FlowVisor slices the network into abstracted units of bandwidth, topology, traffic and network device CPUs. FlowVisor divides the switches and resources between controllers and ensures that each one accesses and controls only the resources assigned to it.

7.3. SDN switches

Open vSwitch[17] is an open source switch that enables effective and massive network automation through software extensions and supports standard SDN interfaces and protocols. Currently, Open vSwitch supports the following features:

- Security features: Supports 802.1Q VLAN model.
- Monitoring: Open vSwitch enables NetFlow, sFlow, SPAN, RSPAN and mirroring for increasing visibility and monitoring.
- Quality of Service (QoS): Open vSwitch supports traffic queuing, traffic shaping and policies to ensure QoS.
- Automated control: Open vSwitch supports OpenFlow 1.0 plus numerous extensions and OVSDB management protocol.

OpenState switch[18] allows for abstracting the data plane abstraction via eXtensible finite state machines.

7.4. Traffic generation, capturing, analyzing and performance monitoring

Tcpreplay[19] is an open source suite of utilities that allows editing and replaying previously captured network traffic. Basically, tcpreplay resends all previously captured packets at a specified data rate or at the speed it was captured up to the hardware speed capability. In addition, tcpreplay provides the researchers and developer with the ability to test firewalls, network intrusion detection systems, NetFlow and other network devices. Also, it provides results reporting, Flow statistics including flows per second and flows analysis and fine-tuning of flow expiry timeouts.

Scapy[20] is a Python-based tool that is capable to send, sniff and dissect and forge network packets. Scapy is an interactive packet manipulation program that can easily handle several tasks such as scanning, attacks, probing, unit tests, tracerouting, attacks and network discovery. This tool has a flexible model and enables users to build their own packets. In the context of SDN, Scapy has been employed to emulate a DDoS attack [8,15,26] and normal traffic.

Distributed Internet Traffic Generator (D-ITG)[21] platform allows for producing IPv4 and IPv6 traffic. The traffic generation process can be based on various probability distributions such as Constant, Uniform, Normal, Exponential, Gamma, Cauchy, Poisson and Pareto. Currently, D-ITG supports Linux, Windows, OSX and FreeBSD platforms and supports UDP, TCP, DCCP, SCTP protocols.

Static Flow Pusher API is a Floodlight controller module that enables manual addition of flows into OpenFlow-based SDN network. This API is useful for proactive flow insertion. The Forwarding module (*i.e.*, loaded by default) does reactive flow pushing and must be disabled in order to use the static flow pusher module.

Tcpdump[22] is a widely used command-line network packet analyzer. Tcpdump captures detailed network traffic including timestamps. It reports a set of statistics:

[14] https://openflow.stanford.edu/display/Beacon.
[15] http://osrg.github.com/ryu/.
[16] https://www.opendaylight.org/.
[17] https://www.openvswitch.org/.
[18] http://openstate-sdn.org/.
[19] https://tcpreplay.appneta.com/.
[20] https://scapy.net/.
[21] http://www.grid.unina.it/software/ITG/.
[22] https://www.tcpdump.org/.

Table 8
Classification of Reviewed Articles based on the Traffic Generation and Monitoring tools.

Tool type	Tool name	Research works
Generating, Capturing, Analyzing Tools	Tcpreplay	[19]
	Scapy	[8,15,26]
	D-ITG	[15,20]
	Tcpdump	[19]
	Hping3	[10,22]
	BigFlows Trace	[18]
	Nping	[22]
	BoNesi	[23]
	TFN2K	[20]
	sFlow-RT	[28]
Measurements, Monitoring Tools	iperf	[16,29,30]
	Httperf	[16]

- Counts of captured packets (*i.e.*, received and processed).
- Counts of packets that were received by the filter.
- Counts of packets that were dropped by kernel.

Hping3[23] is a command-line that sends custom TCP/IP packets and displays target replies like ICMP replies. Hping3 package supports a variety of protocols; specifically, TCP, UDP, ICMP, and RAW-IP protocols. Mainly, Hping3 tool has been utilized as a security tool for testing firewall and network, scanning ports, tracing routes, and auditing TCP/IP stacks. In the context of SDN, Hping3 has been used in order to generate a DoS flooding attack.

BigFlows Trace[24] is another tool for capturing real traffic on a busy private network. It is designed to capture a large number of flows (*i.e.*, 40686) within 5 min across various applications (*i.e.*, 132 applications).

Nping[25] is a highly customizable open source network packet generation, response analysis and response time measurement tool. Nping can be utilized to generate a raw packet for DoS attacks and route tracing and ARP poisoning. Nping is a cross-platform traffic generator.

BoNesi[26] allows simulating a special type of DDoS attack, Botnet DDoS attack. This is accomplished via generating ICMP, UDP and TCP (HTTP) flooding attacks from a predefined botnet size. BoNeSi is considered as the first tool that simulates HTTP-GET floods from large-scale bot networks.

Tribal Flood Network 2000 (TFN2K)[27] is a DDoS attack tool that generates UDP flood, ICMP flood and TCP SYN flood. TFN2K is compatible with UNIX and Windows platforms. It aimed to attack systems with IP stack vulnerabilities and weak routers. TFN2K enables controlling any number of remote machines to generate on-demand, anonymous DDoS attack and remote shell access.

sFlow-RT[28] is an asynchronous analytics technology that allows testing and validating of the performance aspect of SDN networks and applications.

A widely used tool for active performance measurements and tuning of IP networks is iperf.[29] The iperf tool reports the bandwidth loss and various parameters related to timing, buffers and protocols. This tool is supported on several platforms such as Linux, Windows, Mac OS, Android, FreeBSD, VxWorks, OpenBSD, NetBSD and Solaris.

Httperf is a testing tool to measure the performance of web servers. Httperf enables the performance analysis of new server features and enhancements. Also, this tool is capable to generate server overload and supports the HTTP/1.1 protocol. In addition, Httperf is extensible towards new workload generators and statistics collectors.

sFlowTrend[30] is a free graphical and monitoring network tool. sFlowTrend utilizes sFlow protocol and its extensions in order to generate real-time statistics about network bandwidth usage, monitor physical and virtual server performance.

Table 8 presents the distribution of referenced research works according to the utilized traffic generation and monitoring tools.

7.5. Datasets

This section describes and details the specification and features of the datasets have been utilized to train and test the ML-based solutions. Table 9 presents the distribution of the reviewed articles based on the used dataset.

[23] http://www.hping.org/hping3.html.
[24] http://tcpreplay.appneta.com.
[25] https://nmap.org/nping/.
[26] https://github.com/markus-go/bonesi.
[27] https://github.com/poornigga/tfn2k.
[28] https://sflow-rt.com/.
[29] https://iperf.fr/.
[30] https://inmon.com/products/sFlowTrend.php.

Table 9
Classification of Reviewed Articles based on the Utilized Dataset.

Dataset	Research works
CAIDA DDoS Attack 2007	[23,26]
CAIDA Anonymized 2008 Internet traces	[26]
CAIDA Anonymized Internet Traces 2015	[23]
1999 Darpa Intrusion Detection Evaluation	[26]
2000 Darpa Intrusion Detection Evaluation	[23,26]
KDD Cup 1999 traces	[26]
ISCXIDS2012	[24]

Center for Applied Internet Data Analysis (CAIDA) datasets[31] a center that aims to collect, analyze, and share data for scientific analysis of Internet traffic, topology, routing, performance, and security-related events. It provides to researchers two types of data public datasets and restricted datasets required to request access to it. CAIDA DDoS Attack 2007 is one of common restricted CAIDA datasets that contains approximately one hour of anonymized traffic traces from a DDoS attack. Another restricted dataset called CAIDA Anonymized 2008 Internet traces consists of one-hour anonymized passive traffic trace collected during the DITL 2008 measurement event. CAIDA Anonymized Internet Traces 2015 is one of CAIDA dataset that contains anonymized passive traffic traces.

MIT Lincoln Laboratory[32] provides the researches with many experimental datasets for scientific analysis in several research areas such as communication systems, cybersecurity and information system and air traffic control. 1999 Darpa Intrusion Detection Evaluation dataset divided into training and testing data. Training data consists of three weeks of traffic. 2000 Darpa dataset is another data provided by Lincoln Laboratory that includes two attacks scenarios: LLDOS 1.0 and LLDOS 2.0.2. LLDOS 1.0 includes a DDoS attack run by a novice attacker over multiple network and audit sessions.

KDD Cup 1999 traces[33] is the dataset utilized for The Third International Knowledge Discovery and Data Mining Tools Competition. This dataset contains a total of 24 training attack types and 38 the testing attack types. UNB Canadian Institute for Cybersecurity (CIC)[34] provides universities, private industry and independent researchers with 13 different datasets for scientific analysis of cybersecurity. ISCXIDS2012 dataset is one of 13 datasets that consist of labeled real traffic. This dataset includes several intrusion scenarios with full packet payloads.

8. Discussion, future research directions, and conclusion

This research paper includes two main contributions. First, the paper presented a new classification of DoS attacks and mitigation techniques related to SDN. The classification included mitigation methods to cope with DoS attacks on SDN and SDN-based solutions for mitigating DoS attacks in general. The first category includes detection, mitigation, and prevention methods for DoS attacks against the SDN switch and controller. The second category includes solutions that utilized the SDN characteristics to handle DoS in other network environments (e.g., cloud computing, legacy networks, and data center). The paper analyzed the methods to cope with DoS attacks in SDN and classified them into six types: Table-Entry-based, Scheduling-based, Architectural-based, Flow statistics-based, ML-based, and Hybrid solutions.

There are several promising future research directions. First, while existing research works have established a good level of progress in ML-based detection solutions for DoS/DDoS, more contributions can be developed using a graph, KNN, and clustering methods. Second, a promising future direction would be to develop additional SDN-based solutions for DoS/DDoS mitigation against other environments such as IoT, cloud computing paradigms, and ISP networks. Third, designing and developing optimization-based methods for detecting DoS/DDoS against SDN and in SDN-based solutions to minimize the false positive and maximize the true positive is an additional potential area of research. Fifth, authors generally used different options for selecting baselines, datasets, and evaluation methodology, and their selections were seemingly arbitrary, which led to inconsistent and unreliable reporting of scores. There is no control over the evaluation process and its settings. Also, many evaluation parameters need unified tuning to ensure reliable results. The authors believe that a more unified evaluation methodology is needed to ensure the reliability of the results and facilitate the comparison between approaches.

CRediT authorship contribution statement

Bushra Alhijawi: Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Drafting the manuscript, Revising the manuscript critically for important intellectual content. **Sufyan Almajali:** Conception and design of study, Analysis and/or interpretation of data, Drafting the manuscript, Revising the manuscript critically for important intellectual content. **Hany Elgala:** Drafting the manuscript, Revising the manuscript critically for important intellectual content. **Haythem Bany Salameh:** Drafting the manuscript, Revising the manuscript critically for important intellectual content. **Moussa Ayyash:** Drafting the manuscript, Revising the manuscript critically for important intellectual content.

[31] http://www.caida.org.
[32] https://www.ll.mit.edu/.
[33] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
[34] https://www.unb.ca/cic.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.compeleceng.2022.107706.

Acknowledgments

All authors approved the version of the manuscript to be published.

References

[1] Brief ONF-OS. Sdn security considerations in the data center. 2013.
[2] Dong S, Abbas K, Jain R. A survey on distributed denial of service (ddos) attacks in sdn and cloud computing environments. IEEE Access 2019;7:80813–28.
[3] Sultana N, Chilamkurti N, Peng W, Alhadad R. Survey on sdn based network intrusion detection system using machine learning approaches. Peer-To-Peer Netw Appl 2019;12(2):493–501.
[4] Imran M, Durad MH, Khan FA, Derhab A. Toward an optimal solution against denial of service attacks in software defined networks. Future Gener Comput Syst 2019;92:444–53.
[5] Swami R, Dave M, Ranga V. Software-defined networking-based ddos defense mechanisms. ACM Comput Surv 2019;52(2):1–36.
[6] Singh J, Behal S. Detection and mitigation of ddos attacks in sdn: A comprehensive review, research challenges and future directions. Comp Sci Rev 2020;37:100279.
[7] Feghali A, Kilany R, Chamoun M. Sdn security problems and solutions analysis. In: 2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS). IEEE; 2015, p. 1–5.
[8] Qian Y, You W, Qian K. Openflow flow table overflow attacks and countermeasures. In: 2016 European Conference on Networks and Communications (EuCNC). IEEE; 2016, p. 205–9. http://dx.doi.org/10.1109/eucnc.2016.7561033.
[9] Bhushan K, Gupta B. Distributed denial of service (ddos) attack mitigation in software defined network (sdn)-based cloud computing environment. J Ambient Intell Humaniz Comput 2018;10(5):1–13. http://dx.doi.org/10.1007/s12652-018-0800-9.
[10] Zhang P, Wang H, Hu C, Lin C. On denial of service attacks in software defined networks. IEEE Netw 2016;30(6):28–33. http://dx.doi.org/10.1109/mnet.2016.1600109nm.
[11] Yan Q, Gong Q, Yu FR. Effective software-defined networking controller scheduling method to mitigate ddos attacks. Electron Lett 2017;53(7):469–71.
[12] Etaiwi W, Biltawi M, Almajali S. Securing distributed SDN controllers against DoS attacks. In: 2017 International Conference on New Trends in Computing Sciences (ICTCS). IEEE; 2017, p. 203–6. http://dx.doi.org/10.1109/ictcs.2017.52.
[13] Imran M, Durad MH, Khan FA, Derhab A. Reducing the effects of dos attacks in software defined networks using parallel flow installation. Human-Centric Comput Inf Sci 2019;9(1):16.
[14] Dao N-N, Park J, Park M, Cho S. A feasible method to combat against ddos attack in sdn network. In: 2015 International Conference on Information Networking (ICOIN). IEEE; 2015, p. 309–11. http://dx.doi.org/10.1109/icoin.2015.7057902.
[15] Wang R, Jia Z, Ju L. An entropy-based distributed DDoS detection mechanism in software-defined networking. In: 2015 IEEE Trustcom/BigDataSE/ISPA. IEEE; 2015, p. 310–7. http://dx.doi.org/10.1109/trustcom.2015.389.
[16] Piedrahita AFM, Rueda S, Mattos DM, Duarte OCM. Flowfence: a denial of service defense system for software defined networking. In: 2015 Global Information Infrastructure and Networking Symposium (GIIS). IEEE; 2015, p. 1–6. http://dx.doi.org/10.1109/giis.2015.7347185.
[17] Durner R, Lorenz C, Wiedemann M, Kellerer W. Detecting and mitigating denial of service attacks against the data plane in software defined networks. In: 2017 IEEE Conference on Network Softwarization (NetSoft). IEEE; 2017, p. 1–6.
[18] Boite J, Nardin P-A, Rebecchi F, Bouet M, Conan V. Statesec: Stateful monitoring for ddos protection in software defined networks. In: 2017 IEEE Conference on Network Softwarization (NetSoft). IEEE; 2017, p. 1–9.
[19] Niyaz Q, Sun W, Javaid AY. A deep learning based ddos detection system in software-defined networking (sdn), http://dx.doi.org/10.4108/eai.28-12-2017.153515, arXiv preprint arXiv:1611.07400.
[20] Cui Y, Yan L, Li S, Xing H, Pan W, Zhu J, Zheng X. Sd-anti-ddos: Fast and efficient ddos defense in software-defined networks. J Netw Comput Appl 2016;68:65–79.
[21] Wang T, Chen H. Sguard: A lightweight sdn safe-guard architecture for dos attacks. China Commun 2017;14(6):113–25. http://dx.doi.org/10.1109/cc.2017.7961368.
[22] Jankowski D, Amanowicz M. On efficiency of selected machine learning algorithms for intrusion detection in software defined networks. Int J Electron Telecommun 2016;62(3):247–52. http://dx.doi.org/10.1515/eletel-2016-0033.
[23] Phan TV, Van Toan T, Van Tuyen D, Huong TT, Thanh NH. Openflowsia: An optimized protection scheme for software-defined networks from flooding attacks. In: 2016 IEEE Sixth International Conference on Communications and Electronics (ICCE). IEEE; 2016, p. 13–8.
[24] Li C, Wu Y, Yuan X, Sun Z, Wang W, Li X, Gong L. Detection and defense of ddos attack–based on deep learning in openflow-based sdn. Int J Commun Syst 2018;31(5):e3497. http://dx.doi.org/10.1002/dac.3497.
[25] Chin T, Mountrouidou X, Li X, Xiong K. An sdn-supported collaborative approach for ddos flooding detection and containment. In: MILCOM 2015-2015 IEEE Military Communications Conference. IEEE; 2015, p. 659–64.
[26] Zhu L, Tang X, Shen M, Du X, Guizani M. Privacy-preserving ddos attack detection using cross-domain traffic in software defined networks. IEEE J Sel Areas Commun 2018;36(3):628–43. http://dx.doi.org/10.1109/jsac.2018.2815442.
[27] Chen W, Xiao S, Liu L, Jiang X, Tang Z. A ddos attacks traceback scheme for sdn-based smart city. Comput Electr Eng 2020;81:106503.
[28] Buragohain C, Medhi N. Flowtrapp: An sdn based architecture for ddos attack detection and mitigation in data centers. In: 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN). IEEE; 2016, p. 519–24.
[29] Oktian YE, Lee S, Lee H. Mitigating denial of service (DoS) attacks in OpenFlow networks. In: 2014 International Conference on Information and Communication Technology Convergence (ICTC). IEEE; 2014, p. 325–30. http://dx.doi.org/10.1109/ictc.2014.6983147.
[30] Wang H, Xu L, Gu G. Floodguard: A dos attack prevention extension in software-defined networks. In: 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. IEEE; 2015, p. 239–50. http://dx.doi.org/10.1109/dsn.2015.27.

Bushra Alhijawi is an assistant professor in the Data Science department at Princess Sumaya University for Technology (PSUT). She received her Ph.D. degree in computer science from PSUT, Jordan in 2021 and graduated with 2nd class Honor. She received her Master's degree in Information Systems and Innovation from Hashemite University, Jordan in 2017 and graduated with 1st class Honor. She has a B.Sc. degree in Computer Information Systems from Al-Albayt University, Jordan, in 2015 and graduated with 1st class Honor. She is pursuing research on web intelligence, recommender system, machine learning, and optimization.

Sufyan Almajali received the Ph.D. degree in computer science from the Illinois Institute of Technology, Chicago, USA. He is an Associate Professor at Princess Sumaya University for Technology, Jordan. He has 19 years of academic and industrial experience. His research interests include IoT security, edge computing, and network security.

Hany Elgala received the Ph.D. degree from Jacobs University, Germany, in 2010. He is currently an Assistant Professor with the Electrical and Computer Engineering Department, University at Albany–State University of New York, Albany, NY, USA. He focuses on the areas of wireless networks, digital signal processing, and embedded systems. He has authored over 80 journals and conference publications.

Haythem A. Bany Salameh received his Ph.D. degree in electrical and computer engineering from the University of Arizona, Tucson in 2009. He is currently a professor of telecommunication engineering with Yarmouk University, Irbid, Jordan. His research is in wireless communications technology and computer networking.

Moussa Ayyash received his B.S., M.S. and Ph.D. degrees in Electrical and Computer Engineering. He is currently a professor at Chicago State University, Chicago, IL. His current research interests span digital and data communication areas, wireless networking, visible light communications, network security, Internet of Things, ML, and interference mitigation.