IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 9, 2021, accepted May 1, 2021, date of publication May 6, 2021, date of current version May 17, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3078065

An Efficient IDS Framework for DDoS Attacks in SDN Environment

JOSY ELSA VARGHESE AND BALACHANDRA MUNIYAL, (Member, IEEE)
Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India
Corresponding author: Balachandra Muniyal (bala.chandra@manipal.edu)

ABSTRACT The rapid usage of the Internet for the last few decades has lead to the deployment of high-speed networks in commercial and educational institutions. As network traffic is increasing, security challenges are also increasing in the high-speed network. Although the Intrusion Detection System (IDS) has a significant role in spotting potential attacks, the heavy traffic flow causes severe technical challenges relating to monitoring and detecting the network activities. Moreover, the devastating nature of the Distributed Denial-of-Service (DDoS) attack draws out as a significant cyber-attack regardless of the emergence of Software Defined Network (SDN) architecture. This paper proposes a novel framework to address the performance issues of IDS and the design issues of SDN about DDoS attacks by incorporating intelligence in the data layer using Data Plane Development Kit (DPDK) in the SDN architecture. This novel framework is named as DPDK based DDoS Detection (D3) framework, since DPDK provides fast packet processing and monitoring in the data plane. Moreover, the statistical anomaly detection algorithm implemented in the data plane as Virtual Network Function (VNF) using DPDK offers fast detection of DDoS attacks. The experimental results of the D3 framework guarantee both efficiency and effect of the novel IDS framework. The publicly available CIC DoS datasets also ensure the detection effect of a single statistical anomaly detection algorithm against the DDoS attack.

INDEX TERMS Data plane development kit (DPDK), denial of service attack (DoS), DPDK based DoS detection (D3) framework, high-speed network, intrusion detection system (IDS), software defined network (SDN), virtual network function (VNF).

I. INTRODUCTION

Distributed Denial of Service (DDoS) has been one of the evergreen attacks for a few decades preventing legitimate users from accessing services, incapacitating the target, and causing high revenue loss. Recently the Amazon Web Services (AWS) was attacked by DDoS attack with a peak traffic volume of 2.3 Tbps in February 2020 [1] and GitHub was targeted by 1.35 Tbps in February 2018 [2]. There has been an exponential increase in the power, frequency, severity, and volume of DDoS attacks despite the existence of all detection and mitigation solutions. It is hard to detect DDoS attacks without adversely affecting network resources. Thus, the inevitable need of the research community is to focus on developing an efficient Intrusion Detection System (IDS) framework against DDoS attacks with high detection power. The middlebox-based DDoS detection used in conventional systems offers good accuracy, but it causes communication overhead and rigidity. The requirement of customized hardware with software in the middlebox defense technique is incompatible with adaptable network architecture and fails to maintain a global network intelligence [3]–[8]. So researchers addressed this issue by a programmable network paradigm called Software Defined Network (SDN) for challenging security threats of DDoS [9]–[14] that delivers network intelligence to incorporate the rapid change of network configuration in today's data centers, industry, academic, and IoT era. It helps to provide a holistic, cost-effective, lightweight approach against DDoS attacks without any additional hardware requirements, which is ideal for a modern changing network scenario.

The introduction of network programmability, the global network intelligence, the decoupling of data plane and control plane, traffic engineering with dynamic forwarding rules of network traffic in SDN paved a secured and adaptable innovation in the network architecture. But the centralized

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Shen.

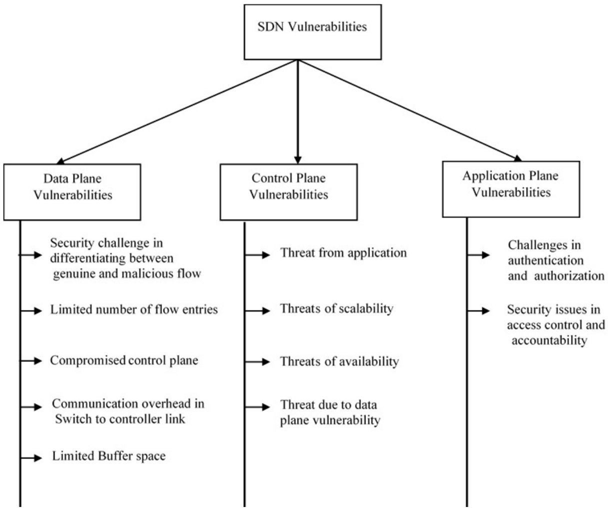

SDN Vulnerabilities
Data Plane Vulnerabilities
Control Plane Vulnerabilities
Application Plane Vulnerabilities
Security challenge in differentiating between genuine and malicious flow
Limited number of flow entries
Compromised control plane
Communication overhead in Switch to controller link
Limited Buffer space
Threat from application
Threats of scalability
Threats of availability
Threat due to data plane vulnerability
Challenges in authentication and authorization
Security issues in access control and accountability

FIGURE 1. **SDN vulnerabilities.**

SDN controller causes potential threats due to its single point failure. The major vulnerabilities in SDN can be categorized into three major attacks based on the different plane of SDN architecture [15], which is shown in Figure 1.

1) *Data Plane Attacks*: The space constraint in the data plane results in buffer saturation and flow table overflow, which are the main reasons for the security challenges in data plane attacks. Due to the presence of dump switches and the decision-making role of the controller, the identification of genuine and malicious flow in the data plane is a challenging task. As the data traffic increases, the congestion in the control-data plane link causes disconnection of the control plane and data plane. Moreover, the data plane resources will be compromised as the SDN controller is compromised. Hence, the data plane attacks depend on the security of the control plane.
2) *Control Plane Attacks*: The control plane is the targeted plane for most of the attacks due to the centralized nature of the controller. It includes threats from the application, threats of scalability, and threats of availability. Most of the untreated data plane problems cause saturation attacks in the control plane. The controller is responsible for a customized security check of different applications with authentication of applications and authorization of resources, which have not been established yet. Moreover, today's SDN controllers are not able to handle the network traffic in a high-speed network having a 10 Gbps link [16]. The lack of scalability of the SDN controller and the unavailability of network resources create a conducive environment for DoS attacks in SDN. Multi-controller is not a good solution for DDoS attacks as it can result in cascading failure of all controllers.
3) *Application Plane Attacks*: It includes challenges concerned with authentication and authorization, issues related to access control and accountability. It is important to authenticate every request from the application to access network resources. However, the authentication of a large number of SDN applications is challenging. Moreover, the malicious application can bypass the SDN network due to the lack of access control and accountability.

The DDoS attacks are categorized under the availability threat of the controller. The reasons for DDoS vulnerabilities are as follows: a) Buffer saturation due to the limited memory space to buffer the information, b) Controller saturation is the overhead in the controller due to the centralized architecture of the controller, c) Flow Table overflow due to limited TCAM memory, and d) Communication overhead of control-data plane link causes bottleneck to the legitimate users.

Even though the global view of the SDN controller is beneficial to DDoS detection, the flow statistics of switches received from the data plane lead to a significant detection delay and communication overhead of South Bound Interface (SBI), which results in bottleneck [17]–[20] and saturation attack on the controller [16], [21]. Moreover, the contradictory relationship of the centralized architecture of SDN and the distributed nature of DDoS attacks cause several design issues for building an efficient intrusion detection system in SDN [22]. The design issues can be resolved by recommending any two alternatives. The first option is the participation of the data plane in anomaly detection [23]–[26] which reduces channel congestion and overloading of the SDN controller. The second option is the introduction of lightweight statistical anomaly detection algorithms [17], [18], [27], [28] which is the best suit for the SDN controller. Radware Defense Flow [29] is an example of a commercial solution for statistical anomaly detection. The statistical approaches can identify new attacks, low rate attacks, and high rate attacks with faster response time and minimum overhead to the controller since it can detect attacks by a minimum number of features [18], [30]–[32]. Moreover, the detection time reduces with the fewer number of features [26] and with the low complexity in the traffic characterization phase [33].

Even though these existing approaches deliver good accuracy and reduce SDN southbound communication overhead, the performance of actuators in the data plane is lower compared to the hardware path. Moreover, the deployment of a high-speed network in today's data centers, commercial and academic institutions reveals the challenges of IDS concerning network monitoring and network security [34]. Since IDS is unable to handle the huge network traffic, it results in huge packet drop and low detection rate. This problem can be solved by scaling the network resources or by increasing the bandwidth to handle it. However, it increases CAPEX and OPEX expenditure. This problem of scaling of network resources can be solved by introducing NFV technology but it cannot handle the problem of bandwidth with OVS switches, which can be resolved through DPDK. Hu *et al.* [35] have recommended an IDS in a high-speed network using DPDK capturing mechanism and SDN technology.

Thus, the objective of this work is to build an efficient and cost-effective IDS in a high-speed network against DDoS attacks. The rationale of the proposed IDS framework named DPDK based DDoS Detection (D3) framework in SDN environment is to address the design issues of SDN environment for DDoS defense and the limitation of IDS in a high-speed network. The main contribution of this paper is outlined as follows:

1) The proposed D3 framework is considered the first DPDK based DDoS defense framework built on single feature anomaly detection in SDN to the best of our knowledge. The single point failure of the centralized SDN controller and the incapability of dump switches are eluded by using an integrated D3 framework of NFV and SDN technology. Here, SDN abstracts network control functions from network forwarding functions, whereas the NFV abstracts IDS functions from the hardware on which it runs.
2) A lightweight statistical anomaly detection D3 algorithm based on a single feature is introduced, which is the best fit for the SDN environment for fast DDoS detection.
3) The effect and efficiency of the D3 framework are analyzed. The detection effect of the proposed D3 algorithm is evaluated for the D3 framework and CIC DoS datasets, whereas the efficiency of the D3 framework is compared with other IDS alternatives. The performance and detection time of the D3 framework is good enough since it is implemented in OVS-DPDK.
4) This is a cost-effective approach since the framework doesn't involve any extra physical devices for its assistance. Moreover, the network baseline created for normal network scenarios helps to track the abnormal traffic, which can be applied in significant areas like data centers, educational institutions, corporate, government, military, etc.

The abbreviations used in the proposed paper is listed in Table 1 and the rest of the paper is systematized as follows: Section 2 discusses the recent related works. Section 3 mentions the importance of DPDK for DDoS detection. The proposed architecture and methodology of the D3 framework are explained in Section 4. Section 5 presents the experimental setup. The results and discussions are explained in Section 6. Finally, Section 7 is summarized with the conclusion and future scope.

TABLE 1. Acronyms.

Acronyms	Description
SDN	Software Defined Networking
DDoS	Distributed Denial of Service
IDS	Intrusion Detection System
DPDK	Data Plane Development Kit
NFV	Network Function Virtualization
D3	DPDK based DDoS Detection
TCAM	Ternary Content Addressable Memory
VNF	Virtual Network Function
CNF	Containerized VNF
OVS	Open Virtual Switch
OVS-DPDK	DPDK integrated OVS
OF controller	OpenFlow controller
Pktgen-DPDK	Packet Generator in DPDK framework
PacketGen	Packet Generator CNF for normal traffic
AttackGen	Attack Generator CNF for abnormal traffic
DUT	Device Under Test
Host-VM	Host Virtual Machine
CAPEX	Capital Expenditures
OPEX	Operating Expenditures
EAL	Environment Abstraction Layer
PCI	Peripheral Component Interconnect
PMD	Poll Mode Driver
SBI	South Bound Interface
Huge TLB	Huge Page Table
WMA	Weighted Moving Average

II. RELATED WORK

A. PERFORMANCE OF IDS IN HIGH-SPEED NETWORKS

The data transfer on the internet is growing at a fast pace which resulted in the deployment of high-speed networks in commercial and educational institutions. In today's network, the role of IDS for identifying potential attacks is inevitable. The heavy traffic causes major technical challenges for the IDS about monitoring and detecting the network activities. The incapability of IDS to process the large diverse traffic causes the dropping of packets and low detection accuracy. This limits the usage of IDS in the high-speed network.

Extensive studies are conducted [35]–[38] on the performance of IDS in high-speed networks due to the potential challenges that occur during heavy traffic. Hu *et al.* [36] presented the various challenges of packet capturing systems in high-speed networks, which can be solved by using multithreaded architectures. Since the overloading of the IDS misses malicious activities in high-speed networks, the multithreaded architecture optimizes IDS, and maximizes its performance by reducing the overloading of IDS which in turn decreases packet drop and increases CPU utilization. Moreover, the studies [39], [40] describe that the detection capacity of the IDS will decrease with the increase of the packet drop rate. It depicts that the effectiveness of the IDS will degrade with the packet loss. Thus, IDS performance can be influenced by two aspects, namely, packet capturing mechanism and packet detection mechanism.

Wu *et al.* [41] demonstrated the packet processing at 100 Gbps using DPDK packet capturing mechanism in user space with no packet drop, wherein DPDK bypasses the existing network stack for packet processing. Hu *et al.* [35] presented a comprehensive study of the performance of two open-source IDS namely Suricata and Snort in the high-speed network. The objective of this study was to improve the performance of IDS in the high-speed network by incorporating packet capturing and data processing approaches. It also discussed the vital factors like memory utilization, CPU utilization, packet drop rate, and detection accuracy which limits IDS application in high-speed networks. This study concluded with challenges of open source IDS in the high-speed network and provided its recommendation by developing a new IDS in a high-speed network using DPDK capturing mechanism and SDN technique.

B. DIFFERENT DEFENSE MECHANISM IN SDN AGAINST DDOS ATTACKS

The dissonant relationship of SDN architecture and the nature of DDoS attacks underline the relevance of an efficient framework in SDN architecture to handle DoS attacks. The main security issues in the design of SDN architecture are single-point failure of the controller, the existence of dump switches, and limited flow table memory. So, the framework in SDN against DoS attack should address the defense strategies for both data plane attacks and controller plane attacks.

1) DEFENSE AGAINST DDoS ATTACK ON DATA PLANE

In SDN, switches act as forwarding devices without any intelligence, and all the decisions are carried out by the centralized controller. When heavy traffic comes, it will increase the bandwidth of the controller plane and reduce the performance of the controller. So these dump switches [42] increase the vulnerability of the data plane, for it can be easily targetted by attackers because of the incapability of OpenFlow SDN switches to handle threats on their own. Moreover expensive and power-hungry characteristics of TCAM results in the limited TCAM size of SDN switches [43], [44] [45], which increases the risk of rapid overloading by flooding attacks [46], [47]. This results in normal communication breakdown, flow table overflow, and higher energy consumption [48], [49] [50]. Thus the defensive mechanism should be quick and cost-effective as updating OF switches or adding additional appliances are costly.

Xu *et al.* [51] described a mathematical model for table overflow attacks and pinpointed the potential victim in the network topology. This paper also suggested three traffic features that aid to identify attacks through monitoring mechanisms and the mitigation is performed using a token bucket based algorithm. Thus, the proposed work provided a defense against table overflow attacks in the target switch which causes memory exhaustion. It also ensures stable transmission for a normal client and limits the rate of transmission for attackers. It works effectively by reducing attack rate but the routing complexity and overhead increase with topology size.

Durner *et al.* [52] introduced a statistical model and lightweight approach for DoS defense in the data plane to counter table overflow attacks due to flooding attacks. The detection mechanism depends on the analysis of the header field in the flow table and the attackers are identified using hashing techniques which can be handled by defining new rules. This method gives a good detection rate with fewer false positives. Yet, the statistical method failed to identify attackers whose header field change alternatively but performance can be increased by selecting good features.

Since the SDN routing system is exhausted by low traffic flows, which resulted in resource consumption in both data and control plane, Dong *et al.* [53] proposed SPRT (Sequential Probability Ratio Test) for DDoS detection in controller and switches to negate false positive and false negative due to low traffic flows. SPRT is a statistical tool obtained from the ratio of normal flow to low traffic flow. This proposed methodology has outstanding accuracy, versatility, and promptness compared to other detection techniques like percentage, count, and entropy of the flows. But the setting of the threshold value in the real network scenario is challenging.

Yuan *et al.* [54] proposed a QoS mitigation approach based on a peer support strategy that guards the SDN against flow table overflow attacks. It is performed by integrating the available idle resources (switches) in the SDN environment to prevent an attack against the victim switch. Even though redirecting the attack flows from saturated switches to idle switches distribute the traffic to peer switches effectively, the redirection action has no control over the rate of attack traffic. The performance goes down when the attack rate becomes high and there are no adequate resources (switches) for redirection. Moreover, there is no detection mechanism for identifying the attack which also makes it an ineffective methodology for a complete solution against DDoS attacks.

2) DEFENSE AGAINST DDOS ATTACK ON CONTROL PLANE

The controller is the brain of the SDN network, which provides complete visibility and intelligence to the network. The centralized SDN controller is the most attractive target for DDoS attacks. So the controller must be properly protected by fast DDoS detection and mitigation strategy. Most of the defense mechanisms of the controller are focused to avoid resource saturation quickly.

Mousavi and St-Hilaire [55] proposed an entropy-based early DDoS detection method for both bandwidth and memory exhaustion. It is a lightweight fast approach against flooding attacks by calculating the entropy of the destination IP address in the SDN controller. The attack is detected when the entropy value is less than the experimental entropy threshold over the 5 consecutive windows. But the attack against the whole network is not identified and can support only a single controller architecture. Sahoo *et al.* [27] proposed a generalized entropy approach in SDN controller using information distance as detection metric to find the difference in the probability distribution of low rate attack and normal traffic, which is more accurate than detection method described by Mousavi and St-Hilaire [55].

Zhang *et al.* [45] introduced a dynamic queue management approach to prevent resource saturation attacks in the control plane. The queues are expanded dynamically when a UDP flooding attack occurs and can be aggregated during normal traffic by a multi-layer fair queuing (MLFQ) based method which does not need any extra appliances in the data plane. But this method can only handle specific attacks.

Shin *et al.* [16] presented a defense framework called AVANT-GUARD against controller bandwidth saturation threat caused by TCP-SYN flooding attack. The large TCP connections initiated by attackers resulted in a large number of packets to the controller. So AVANT-GUARD allows forwarding plane to handle failed TCP connections, and the flow messages are not sent to the controller until the handshake process is completed successfully. These TCP connections introduce an unavoidable and significant delay. Moreover, this framework is suitable only for TCP-SYN attacks, and the necessity of switch modification is undesirable in a real deployment.

Wang *et al.* [56] introduced a DDoS defense named Flood-Guard against control plane DDoS attack. Instead of controller, the proactive flow rule analyzer monitors new incoming attacks packets from the data plane cache and automatically changes the flow rules when an attack occurs. Even though it reduces the overloading of the controller due to flooding attacks, it increases the delay in the data processing. Moreover, Flood-Guard requires the deployment of supplement devices in the data plane.

Cui *et al.* [28] proposed a SD-Anti-DDoS defense framework by introducing an attack detection trigger for quick response against DDoS attack and to reduce the overhead of the SDN controller. It also traceback the attack source and mitigate it. This framework falls short in the performance of different OpenFlow version.

3) DEFENSE BY INTEGRATING INTELLIGENCE IN SWITCHES

As the controller is responsible for every decision-making of switches in the SDN environment, switches are just forwarding devices. This result isn controller overloading and channel congestion. The characteristic of SDN switches as simple forwarding device increases the communication overhead, delay in attack detection, and congestion in the controller. These problems can be solved by incorporating intelligence in the switches and thereby reducing overload in the controller and its bandwidth. Thus the detection of malicious activities can be detected quickly at the switch level.

Kalkan *et al.* [24] proposed SDNscore, a statistical approach against the DoS attack. This is a packet-based approach, where a score value is calculated to find an unknown DDoS attack which is performed at switch level and the verification module is handled by the controller. Even though this method outperforms the entropy-based model, this is not yet implemented in an SDN environment.

Biote *et al.* [25] presented a stateful approach called StateSec against DDoS attacks in the SDN environment. To achieve this goal, the switches handle the monitoring function and detection using finite state machines, whereas the mitigation is handled by the controller. This system provides improved reactivity and good detection by offloading the controller. The implementation of in-switch processing for monitoring traffic is integrated but the implementation of detection algorithm in the switch-level is not yet implemented.

Han *et al.* [23] introduced a collaborative intelligence in both the data plane and control plane using a cross-plane DDoS framework named OverWatch, where the data plane detection is performed by a coarse-grained sensor using 4 statistical parameters and the controller plane detection using an autoencoder ML algorithm. This is a good solution for both attacks on the data plane and controller. It gives good accuracy and reduces SDN southbound communication overhead. But the performance of actuators in the data plane is low when compared to the hardware path. This can be improved by using DPDK.

Tan *et al.* [26] proposed a new framework for DDoS detection and defense, where the trigger mechanism of DDoS detection is performed in the data plane of the SDN environment and the SDN controller is responsible for the detection and mitigation. The detection of the suspicious traffic is performed by using a combined machine learning algorithm of K-Means and KNN using 5 statistical features. The combined detection method of both the data plane and control plane improves its detection ability and efficiency. However, the DDoS detection in large network traffic is yet to be solved.

In short, the limitation of existing systems include SDN design problem like the overhead of controller, single-point failure, the existence of dump switches, limited flow memory, and other problems including a selection of good feature for detection, early detection, handling all common DDoS attacks, IDS performance in high-speed network, setting baselines, reduce the delay in data processing, and cost of additional hardware for data plane security. Moreover, most of the studies are focused on the detection effect than efficiency. These issues are addressed by the proposed D3 framework in the SDN environment.

III. IMPORTANCE OF DPDK IN DDOS DETECTION

1) As the NIC faces a bottleneck in the high-speed network due to the large overhead of data buffering, copying and interruptions, DPDK offers fast network packet processing in user space by avoiding the overhead caused by kernel function, which advances the DDoS detection rate [57].
2) Even though OpenFlow (OF) provides a programmable data plane in SDN, the programmability with high performance can be guaranteed only through the DPDK framework [58].
3) OVS-DPDK guarantees the performance enhancement of flow forwarding and network management along with the efficiency of real-time packet processing with full CPU utilization [59].
4) The main issue of DDoS attacks in the SDN environment is the saturation of the controller due to the arrival of a large number of packets. This can be solved by adding intelligence in the data plane using the DPDK framework which is logically the same as OpenFlow [23].
5) DPDK handled the issues of commercial off-the-shelf (COTS) based hardware used for intrusion detection [60].

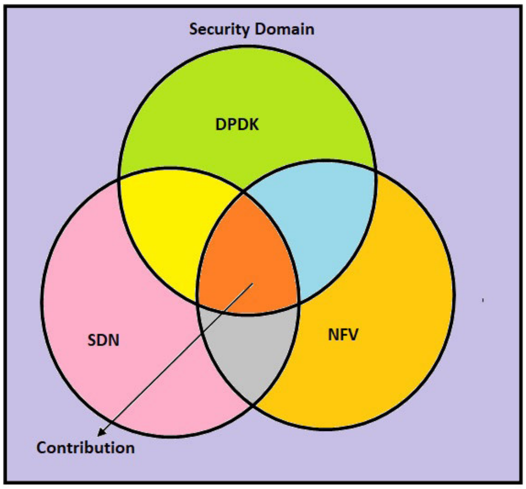


FIGURE 2. Contribution in the security domain.

6) The challenges of IDS in a high-speed network can be swept away by using the DPDK packet capturing mechanism [35].
7) The amalgamation of DPDK with SDN switches offers high performance with a lower performance cost and can reduce overheads of the network traffic [35].

The contribution of the proposed work in the security domain is shown in Figure 2.

IV. METHODOLOGY

A. SYSTEM ARCHITECTURE AND NETWORK MODEL

The virtual switches have a big role in connecting VNF hosted in the same application or across multiple applications. OVS is the most known virtual switch solution. However, the performance limitation of OVS can be overcome by porting OVS to DPDK called OVS-DPDK. The proposed framework is called the D3 (DPDK based DDoS Detection) framework since it uses the DPDK framework for DDoS detection.

1) SYSTEM ARCHITECTURE

The different schematic design of the DDoS detection framework in the SDN environment is shown in Figure 3, wherein Figure 3(a) depicts DDoS attack detection in SDN controller plane described in sections II-B1,VI-A, Figure 3(b) depicts the collaboration of data plane in DDoS detection with the controller plane mentioned in section VI-B, and Figure 3(c) depicts the proposed system architecture which is the integration of DPDK in the Data plane for the fast processing of packets and high performance.

The DPDK framework of the proposed architecture facilitates the data processing in a fast manner, as the data processing occurs in user space using PMD without any kernel interrupts. Thus, it delivers fast switching as PMD polls data directly from NIC which in turn improves the performance of the OVS switch. Thus it addresses the problem of a high-speed network regarding packet capturing. Apart from the advantage of OVS-DPDK, there is another level of optimization performed to increase the performance of network function by running DPDK inside CNF. This will take advantage of DPDK inside the application of DDoS attack detection in addition to the DPDK accelerated OVS. This helps the framework to detect the DDoS attack in a fast manner. Thus it solves the issues of DDoS attack detection in a SDN environment.

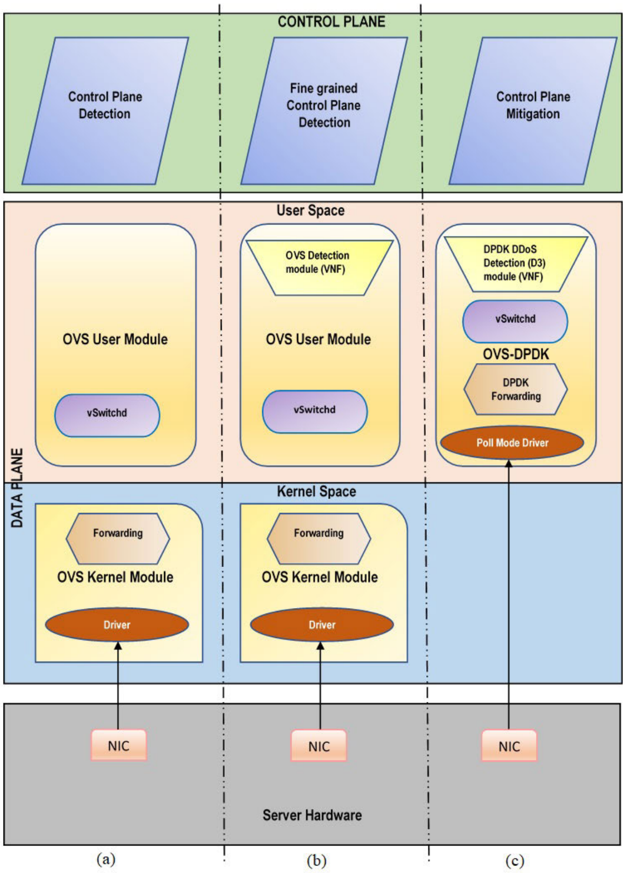


FIGURE 3. System design of DDoS detection framework in SDN environment: (a) Controller-based framework (b) Cross-plane framework (c) DPDK based DDoS Detection (D3) framework.

2) NETWORK MODEL

The network model consists of OVS-DPDK as a bridge, PacketGen and AttackGen are Pktgen-DPDK application [61] for normal and attack traffic respectively, DUT symbolizing data server enabled by a fast packet processing framework of DPDK and DPDK based intrusion detection system for DDoS attacks, and faucet SDN controller [62] are responsible for monitoring and configuring network based on the detection of attacks. Thus, the experimental SDN test network is a simple star topology having five hosts and a switch connected to the SDN controller and the implementation details are provided in Section V. Figure 4 and Figure 5 depict the network model of the proposed system and its notation respectively, where PacketGen (contains two hosts) and AttackGen (contains one host) send packets to

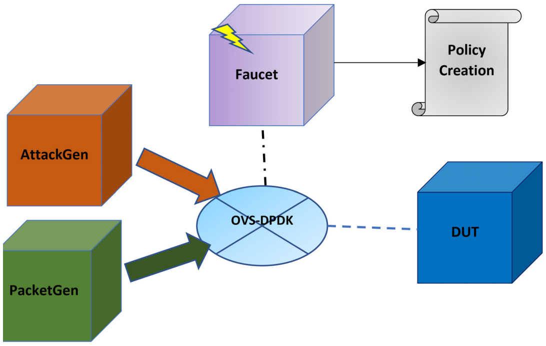


FIGURE 4. Network model of the proposed system.

Sl.No	Notation	Meaning
1	□	Docker Container
2	⚡□	Docker Container for SDN Controller
3	⊗	OVS-DPDK switch
5	📜	Configuration file
6	- - - -	Data Traffic
7	➡	Normal Data Traffic
8	➡	Attack Data Traffic
9	-·-·-	Control Traffic

FIGURE 5. Notation of the network model.

the DUT (contains two hosts) through OVS-DPDK, and the faucet controller changes the network configuration (policy creation) based on attack detection. Thus, DPDK enables fast data processing and improves the performance of network functions in the D3 framework along with high performance and flexibility. The network intelligence in the switch level lessens the overloading of the controller and reduces channel congestion. This is a cost-effective approach since no external resources are needed for implementing the model.

B. PROPOSED METHODOLOGY

The functionalities of the D3 framework are classified into two main modules, namely the DPDK based detection module and the Control plane mitigation module, which is depicted in Figure 6. The DDoS detection takes place at the data plane and the DDoS mitigation is managed by the control plane. A detailed explanation of each module is described below.

1) DPDK BASED DETECTION MODULE

This module integrates network intelligence for DDoS detection using the DPDK framework in the data plane and the functionalities are described below.

a: DATA PRE-PROCESSING

The data preprocessing module is built on the 'Testpmd' DPDK application, which consists of two methods namely packet capturing and traffic aggregation. Packet capturing in the D3 module uses the 'ethdev' library for forwarding packets between ethernet port and PMD features supported by NIC. The 'iofwd' is the forwarding engine used for the D3 application, which is the simplest and fastest forwarding mode [63]. This DUT performs detection of an anomaly for each port which is considered as a different destination. Here, the network packets for each destination address are considered as network flows.

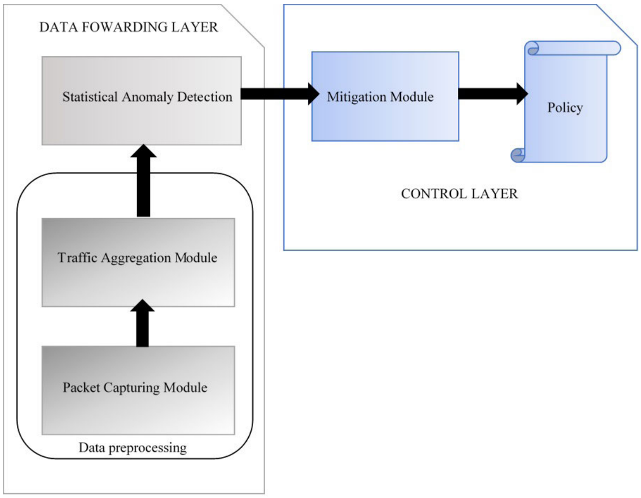


FIGURE 6. Modules in the D3 framework.

The traffic aggregation module in the D3 application collects the average throughput value '*Th*' for each network flows in an interval of 'δt' called *stat_period*. Consider X(t) be $X_1, X_2, X_3, \ldots\ldots. X_n$; represents '*Th*' of '$n$' different network flows at the time interval of 'δt'. These '*Th*' values are considered as the single statistical parameter for the anomaly detection, which represents bandwidth utilization per flow [64], [65].

b: ANOMALY DETECTION MODULE

The anomaly detection performed in the D3 framework is named as D3 Algorithm, wherein a single statistical feature '*Th*' is extracted from the traffic aggregation module for detecting anomalies. The rate of change of throughput corresponding to normal traffic and attack traffic is different. The idea behind DDoS detection is that as a DDoS attack progresses, a massive rate of change occurs. Moreover, the DPDK framework with a single statistical metric makes detection faster.

A lightweight statistical flow monitoring and anomaly detection approach is described in *Algorithm* 1. In the D3 Algorithm, throughput values are collected for every *stat_period* of δt in a moving window size of n where $n > 10$ [66]. Here δt is assigned as 1 second for fast detection and a baseline is established for the minimum and maximum throughput during training.

Moving average is a technical indicator that refers to an average throughput for a network system over a specified

Algorithm 1 D3 Algorithm(The Outlier Detection)

```
INPUT: V[n] ← Throughput value of each destination, t[n] ← current time, w ← window size.
OUTPUT: Triggering the controller based on the detection of the outlier values.
procedure IDENTIFY_THE_OUTLIER_VALUES
    Extract the single feature throughput during a time interval and append it to '$V[w]$'.
    Iterate and sort the values inside each 'w' and apply the weighting moving average (WMA) formula to the window, where the largest throughput value gets the highest weight.
    Continue it for p training index to calculate the maximum throughput as $V_n^{max}$ and minimum throughput as $V_n^{min}$
    for each flow metric $V_{n+1}$ at $t_{n+1}$ do
        if $t_{n+1} = t_n + \delta$ t then
            Add the new entry as an actual value $V_{n+1}^{actual}$ in history records in $V[w]$.
            Calculate the prediction value $V_{n+1}^{predict}$ using WMA, where the largest value gets the highest weight.
                $V_{n+1}^{predict} = \sum_{i=1}^{n} \lambda V_{n+1}^{actual}; \quad where \sum_{i=1}^{n} \lambda = 1$    (1)
            Evaluate the ratio metric $R^{predict}$ to compare prediction value and actual value.
                $R_{n+1}^{predict} = \frac{V_{n+1}^{actual}}{V_{n+1}^{predict}}$    (2)
            Calculate the mean and standard deviation for ratio metrics as $R^{mean}$ and $\sigma$
            Use Pauta criterion to evaluate the prediction range
                $R_u \leftarrow R^{mean} + 3 * \sigma$    (3)
                $R_l \leftarrow R^{mean} - 3 * \sigma$    (4)
            if $(R_{n+1}^{predict} > R_u \ \&\& \ V_{n+1}^{actual} > V_n^{max}) \parallel (R_{n+1}^{predict} < R_l \ \&\& \ V_{n+1}^{actual} < V_n^{min})$ then
                Trigger alert to controller
                $boolTrigger \leftarrow 1$
            else
                Normal Traffic
                $boolTrigger \leftarrow 0$
                Append ratio metrics $R_{n+1}$ and actual value $V_{n+1}^{actual}$ to the sliding window $R[w]$ and $V[w]$ respectively.
            end if
        end if
    end for
    return $boolTrigger$
end procedure
```

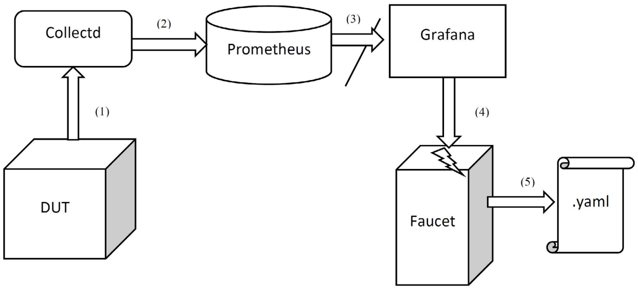


FIGURE 7. Block diagram of mitigation module.

TABLE 2. Ports listening to the services.

Sl.No:	Service	Port
1	Faucet OpenFlow Channel	6653
2	Gauge OpenFlow Channel	6654
3	Prometheus Node exporter	9100
4	Collectd Exporter	9103
5	Prometheus Faucet	9302
6	Prometheus Gauge	9303
7	Prometheus Web Interface	9090
8	Grafana Web Interface	3000

period. It helps to keep track and identify trends by smoothing normal fluctuations. Thus, the moving average acts as an analytical tool to identify the current trend and the potential for a change in an established trend. Assuming that when attacks occur, throughput values increase. As a result, the value of λ in *Algorithm* 1 is adaptively adjusted, with the highest throughput value receiving the highest weight in predicting the value for the next time slot. So, the ratio metrics derived from the actual throughput value and the predicted throughput value helps in identifying the instabilities from the normal trend using the 3 sigma criterion (68-95-99.7 rule) of the gaussian distribution. The value that breaks the normally distributed metrics is considered as outliers or anomalies [67]. This helps to find the deviation from the historical records and current value in a better way. Moreover, the normal traffic is added into the moving window to the next iteration for creating the baselines, which helps to reduce the false positives.

2) CONTROL PLANE MITIGATION MODULE

Early detection of DDoS attacks helps in corrective action by changing the configuration (.yaml) file based on the trigger received from the D3 algorithm using components like Collectd, Prometheus, and Grafana, wherein Collectd is a daemon for gathering statistics from an application, Prometheus is a time-series database and monitoring system based on the pull approach connected to Gauge controller and Collectd, and Grafana is an open-source monitoring dashboard for Prometheus databases used for data analysis and visualization which notifies the SDN controller using Grafana alert notification. The network faucet controller is responsible for network configuration against DDoS attacks. The Figure 7 depicts the high-level block diagram of the mitigation module and the ports listening to services are shown in Table 2.

1) Collectd gathers trigger indication from the DUT running D3 algorithm. The trigger indicator includes the alert flag of each port

TABLE 3. Advantage of the proposed system.

Issues	Sub Issues	Addressed	Not Addressed	Status of Proposed System	Proposed Solution
Design issues of SDN	Single point failure or Overloading of Controller	[53] [18] [30] [23] [24] [25] [27] [69] [70]		Addressed	Network functionalities are distributed to both data plane and control plane
	Existence of dump switches	[23] [24] [25] [26]	[42]	Addressed	Intelligence is added in switch level
	Channel Congestion	[16] [57]		Addressed	SBI communication is reduced by the involvement of data plane functionalities
	Limited flow table memory	[51] [52] [18] [54]	[46] [47]	Addressed	Using statistical method of detection
Issues of an efficient detection system against DDoS attack	Selecting good features	[33]	[52]	Addressed	The single feature detection method is introduced
	Lightweight and early detection	[18] [24] [53] [71] [72]		Addressed	Anomaly detection based on a single feature in the DPDK framework
	Specific DDoS attacks	[45] [16]		Addressed	A single feature anomaly detection is capable of detecting most DDoS attacks
	Delay in data processing	[16] [57] [23] [26]		Addressed	DPDK framework is used for the high data processing
	No control on the rate of attacks	[54]		Addressed	Attacks are controlled by changing the network configuration file
	The need for additional hardware		[16] [57] [32]	Addressed	CNF technology is used
	Integrating intelligence in switch level	[23] [26]	[24] [25]	Addressed	Detection mechanism is integrated in switch level
	Challenges of IDS in high-speed network	[35] [41] [73]	[26] [28]	Addressed	DPDK packet capturing mechanism
	Mitigation mechanism	[26] [28] [69]	[71]	Addressed	Using faucet configuration file
	Cost-effective		[35] [36]	Addressed	No need for any hardware, opensource platforms are used
	Integrating time-based database and Monitoring dashboard	[35]		Addressed	Used opensource prometheus database with grafana

2) Prometheus receives the metrics from the Collectd exporter through port 9103
3) Grafana listening at port 3000 receives the indicators from the Prometheus exporter through port 9100
4) Grafana trigger alert to the SDN controller using Grafana alert notification.
5) SDN controller generates network configuration file in response to the alert.

The network configuration file can mitigate the malicious traffic by dropping the malicious packet, rate-limiting the flow towards the detected destination, blocking the port, or redirecting the traffic to scrubbing centers for further analysis. Here dropping of packets is performed since we are more focused on DPDK based anomaly detection. The overhead of the controller reduced significantly due to the involvement of the data plane for anomaly detection.

C. ADVANTAGES OF THE PROPOSED SYSTEM

The advantage of the proposed system is shown in Table 3. For result analysis and comparison of the D3 anomaly detection algorithm, the OverWatch algorithm [23] is used. Both [23] and D3 algorithms are lightweight algorithms implemented in the data plane for the detection of DDoS attacks using ratio metric. The main difference between the [23] algorithm and the D3 algorithm is shown in Table 4.

TABLE 4. Comparison of OverWatch with D3 algorithm.

Sl No	OverWatch [23]	D3 algorithm
1	Implemented using OVS hardware switch	Implemented using OVS-DPDK virtual switch which offers high performance than OVS
2	Using 4 metrics for detection	Using a single metric makes univariate detection faster.
3	Using both normal and attack traffic in the moving window which increases false negatives	Appending normal traffic in the window for next time series prediction, which decreases false negatives
4	No baseline creation in the data plane	Network baselines for maximum and minimum throughput are maintained, which reduces false positive
5	Control layer is responsible for both detection and mitigation	The Control layer offers only anomaly mitigation, which reduces controller overhead

D. FLOWCHART

The flowchart of the main phases of D3 modules is shown in Figure 8, wherein basic functionalities of the 'Testpmd' application are shown in blue colored blocks, and additional functionalities for DDoS detection in the D3 application are shown in green colored blocks. The basic Testpmd functionalities include:

(i) Initialization of DPDK invokes environment abstraction layer through *rte_eal_init* function to initialize DPDK

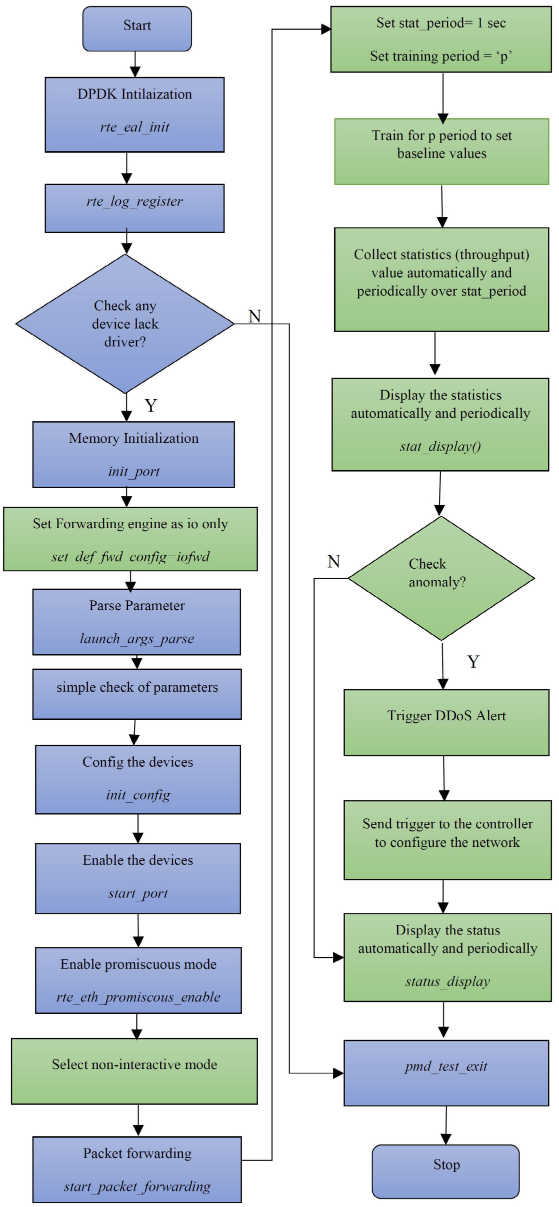

Start
DPDK Intilaization
rte_eal_init
rte_log_register
Check any device lack driver?
N
Y
Memory Initialization
init_port
Set Forwarding engine as io only
set_def_fwd_config=iofwd
Parse Parameter
launch_args_parse
simple check of parameters
Config the devices
init_config
Enable the devices
start_port
Enable promiscuous mode
rte_eth_promiscous_enable
Select non-interactive mode
Packet forwarding
start_packet_forwarding
Set stat_period= 1 sec
Set training period = 'p'
Train for p period to set baseline values
Collect statistics (throughput) value automatically and periodically over stat_period
Display the statistics automatically and periodically
stat_display()
Check anomaly?
N
Y
Trigger DDoS Alert
Send trigger to the controller to configure the network
Display the status automatically and periodically
status_display
pmd_test_exit
Stop

FIGURE 8. Flowchart of D3 module.

runtime environment including buffer management, memory management, PCI, load device driver, and map the kernel mode to user mode.

(ii) Configuration process includes forwarding configuration through *set_def_fwd_config* function, recording the information related to logical cores and socket using *set_default_fwd_lcores_config* function, initializing the ethernet address as destination address through *set_def_peer_eth_addrs* and logging of the analyzed port details using *set_default_fwd_ports_config* function.

(iii) Parsing the parameters of CLI to acquire the configuration details of logical cores, queues, ports, memory distribution, acquiring devices, setting offload, and initializing forward engine using *launch_args_parse* function.

(iv) Configuration of parsed parameters are initialized through the *init_config* function.

(v) Launching device under test (DUT) using *start_port* function. The 'iofwd' forwarding mode is used where *pkt_burst_receive* handles the function for receiving and releasing the packets.

The additional functionalities of the D3 module comprise two main sections: training and testing. In the training phase collects a single statistical feature '*Th*' from each port of DUT for a fixed *stat_period* of 1 second. After the training period, evaluate the WMA of the sorted array, find the $V_{n+1}^{predict}$, $R_{n+1}^{predict}$ by maintaining a baseline of minimum and maximum throughput value. In the testing phase, check the anomaly by using the 3-sigma criterion. Based on the DDoS trigger faucet configure the network file. Thus, the DPDK based DDoS Detection (module) uses a single metric predictive approach for anomaly detection.

V. EXPERIMENTAL SETUP

The experimental setup includes a host machine (Host-VM) which contains five containers namely *OVSdaemon*, *PacketGen*, *AttackGen*, *DUT*, and *Faucet* connected to the OVS-DPDK switch. Even though the container has a fast boot-up time, low overhead and is easy to deploy, yet the container networking can be accelerated by running DPDK inside containers [73]. The initialization steps include installation of OVS-DPDK, the binding of *IGB_UIO* driver in DPDK to NIC (NIC sends packets directly to user space), allocation of the huge pages (4096*2MB), initialization of the OVS database server and OVS controller, and creation of OVS-DPDK bridge. Each container is configured with one memory bank with socket memory of 512 MB and uses virtual devices instead of PCI devices that are connected to the net_virtio-user driver. The hardware and the software requirements for this lab environment are depicted in Table 5 and Table 6.

TABLE 5. Hardware requirements.

Sl.No:	Hardware Requirements	Version
1	RAM	32 GB
2	Hard drive	1TB*2
3	Intel Xeon Processor	8 core 2.4 GHz*2
4	NIC	1GbE*2

TABLE 6. Software requirements.

Sl.No	Software Requirements	Version
1	Ubuntu	18.04
2	Kernel	5.0.0-23 generic
3	DPDK	18.11.5
4	OVS	2.12.0
5	Faucet	1.9.43
6	Gauge	1.9.43
7	Collectd	5.11
8	Prometheus	2.1.0+ds
9	Grafana	v7.0.2
10	Pktgen-DPDK	3.2.4

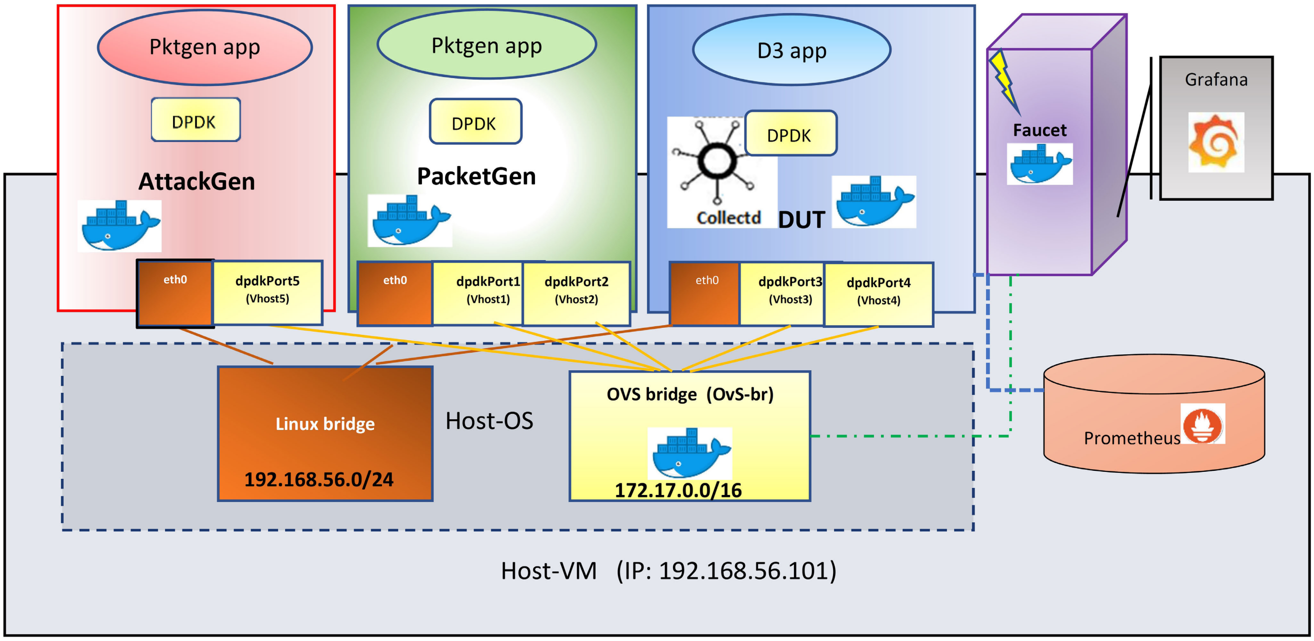


FIGURE 9. Test environment.

The OVS-DPDK bridge is named ‘OvS-br’ has five virtual ports of *dpdkvhostuser* type; two virtual ports for PacketGen, one port for AttackGen, and two virtual ports for DUT. The PacketGen container and the AttackGen containers hold the Pktgen-DPDK (Packet Generator in DPDK framework) application used for generating network traffic through Vhost1 (dpdkPort1), Vhost2 (dpdkPort2) for normal traffic, and Vhost5 (dpdkPort5) for attack traffic by regulating the rate of traffic. The DUT container receives packets from the Pktgen-DPDK application through Vhost3 (dpdkPort3) and Vhost4 (dpdkPort4) and Vhost5 (dpdkPort5) for executing statistical anomaly detection named D3 (Dpdk based DDoS Detection) module. The OvS-br is also connected to the faucet controller for managing the network configuration, which is attached to a time-series database called Prometheus and Grafana dashboard for visualization. The entire setup is shown in Figure 9.

The entire lab setup has 10 cores where core 1 handles ovs-switchd daemon and core 2 is responsible for DPDK PMD functionalities. PacketGen uses cores 0,3,4 to generate ‘packets in an interactive promiscuous mode, wherein core 3 is the master core for invoking command-line interface and managing slave cores whereas slave core 0 and slave core 4 are responsible for generating packets in dpdkport1 and dpdkport2 respectively. AttackGen uses cores 8,9 to generate attacks wherein core 8 is the master core and core 9 is the slave core responsible for attack generation in dpdkport5. The DUT uses another three cores 5, 6, and 7, wherein core 5 is the master core for managing slaves and detecting attacks while core 6 and core 7 are slave cores that are functioning in ‘iofwd’ forwarding mode with a burst of 64 packets and 2048 descriptors in Rx and Tx rings. The distribution of CPU resources on Host-VM is shown in Table 7.

VI. RESULTS AND DISCUSSION

To evaluate the performance of the IDS, the experiments are broadly classified into two categories. The Framework evaluation is performed to test the efficiency of the IDS whereas the Algorithm evaluation is executed to check the detection effect.

A. FRAMEWORK EVALUATION

The framework evaluation is performed by checking efficiency in three ways.

1) Initially, test cases are conducted to compare the performance and latency of OVS and OVS-DPDK at different scenarios in our system configuration.
2) Secondly, the performance of packet capturing mechanisms in various IDS depicted in [35] is compared with the D3 framework under a controlled environment of 10 Gbps TCP flows.
3) Finally, the CPU utilization of the controller in the D3 framework is also evaluated to illustrate the virtue of the D3 framework compared with other SDN based DDoS defense framework [26], [28].

1) TEST CASES FOR THE PERFORMANCE COMPARISON BETWEEN OVS AND OVS-DPDK

Inspired by [74], the performance test of OVS-DPDK on two parameters namely throughput and latency for a virtualized network architecture is conducted. The host machines are configured to the same subnet address 172.17.0.0/16, whereas the client machine is configured as 172.17.0.2 and the server machine as 172.17.0.3 with a gateway of 172.17.0.1. Thus the difference in network performance to OVS and OVS-DPDK are evaluated by network test cases.

TABLE 7. Distribution of CPU resources on host-VM.

Core (0-7)	Core Mask	Functionality	Application
Core 1	0b0000 0010	OVS-daemon	Open vSwitch
Core 2	0b0000 0100	OVS DPDK PMD	
Core 3	0b0000 1000	DPDK master lcore managing GUI and messages	PacketGen (A DPDK packet Generator using Pktgen)
Core 0	0b0000 1000	DPDK PMD – dealing with dpdkport1 (vhost1)	
Core 4	0b0001 0000	DPDK PMD – dealing with dpdkport2 (vhost2)	
Core 5	0b0010 0000	Run master D3 thread	D3 (DPDK based DoS Detection)
Core 6	0b0100 0000	D3 DPDK PMD(Lock D3 to run on cores 6 and 7)	
Core 7	0b1000 0000		
Core 8	0b0001 0000 0000	DPDK master lcore managing GUI and messages	AttackGen (A DPDK packet Generator using Pktgen)
Core 9	0b0010 0000 0000	DPDK PMD dealing with dpdkport5 (vhost5)	

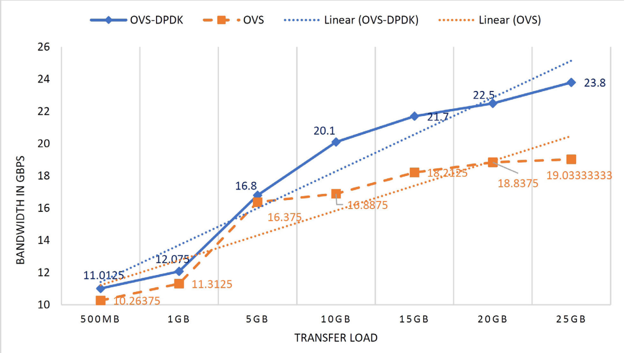


FIGURE 10. Performance comparison between OVS and OVS-DPDK with varying loads.

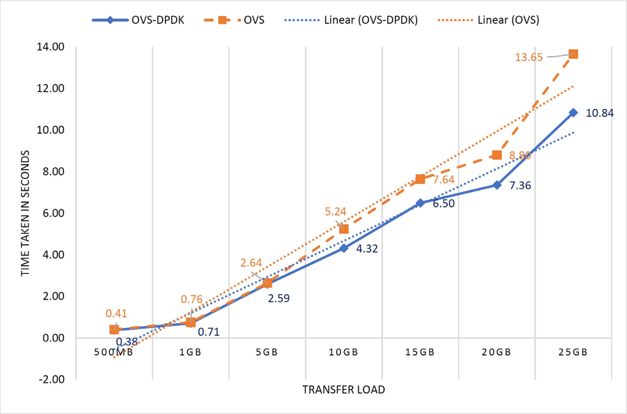


FIGURE 11. Time interval comparison between OVS and OVS-DPDK with varying load transfer.

Throughput Comparison: The throughput test of the framework with OVS and OVS-DPDK is conducted using iperf3 with varying transfer load. The throughput comparison of OVS and OVS-DPDK with varying transfer loads is shown in Figure 10, wherein X-axis represents varying loads and Y-axis represents bandwidth in Gbps. Even though the OVS-DPDK has high performance in bandwidth utilization and the maximum bandwidth transfer compared to OVS throughput, the clear distinction in performance starts from a 10 GB load. Thereafter the variation between OVS and OVS-DPDK increases from 16% to 20% due to polling, huge pages, pinned CPU, and user space IO in OVS-DPDK. Similarly, the time interval comparison between OVS and OVS-DPDK with varying load transfer is shown in Figure 11. Even though the OVS-DPDK has taken less time compared to OVS for load transfer, the clear distinction in time interval starts from 10 GB load, which is the same for throughput comparison. Thereafter the variation between OVS and OVS-DPDK increases around 15% to 20.6% due to context switching overhead in OVS. Hence, the overall network performance of OVS-DPDK is 1.21 × greater than OVS, whereas the time interval used for OVS is 7.25 × greater than OVS-DPDK. The CPU pinning and Huge page tables (Huge TLB) support are the main reason for the performance of the OVS-DPDK.

Latency Comparison: To measure the delay, the latency test of the framework with OVS and OVS-DPDK is performed. The latency of OVS and OVS -DPDK with varying packet size is depicted in Figure 12, wherein X-axis shows varying packet size and Y-axis shows latency in milliseconds. The average of multiple latency test runs is taken into consideration to find the performance variation. The result depicts a parallel trendline that shows clear evidence of lower latency of OVS-DPDK compared to OVS. The latency of OVS-DPDK for various packet size of 64, 128, 256, 512 1024, and 1500 bytes has decreased by 51.92%, 51.37%, 53.57%, 54.20%, 48.11%, and 63.70% respectively. The results clearly show the average latency of OVS is 2.23 × greater than OVS-DPDK. The reason for low latency is the 'fastpath' provided by OVS-DPDK by ignoring kernel, which results in fast packet processing compared to OVS.

2) PERFORMANCE COMPARISON OF DIFFERENT IDS UNDER 10GBPS TCP FLOWS

The vital performance factors for evaluating the efficiency of IDS include CPU utilization, memory utilization, and packet drop rates. The IDS performance of various versions of Snort and Suricata for different packet capturing mechanisms are investigated by Hu *et al.* [35]. It is compared with the D3 framework under the same default configuration of 10 Gbps TCP flow for 1800 seconds, which is depicted in Figure 13. The X-axis shows the name of IDSs with packet capturing mechanism and the Y-axis depicts the performance

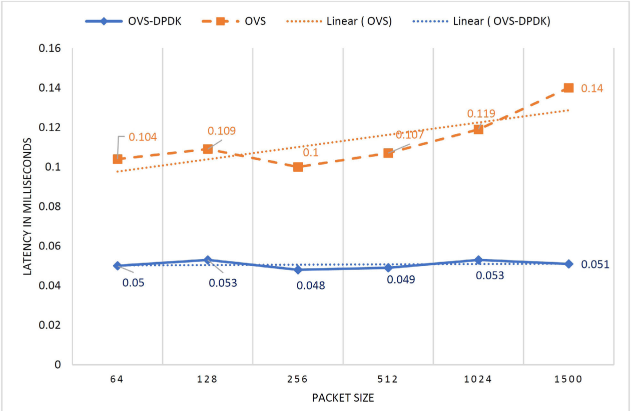
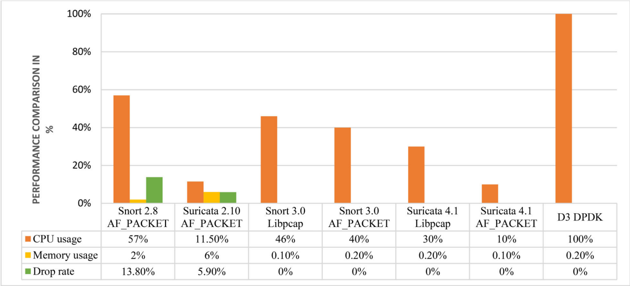


FIGURE 12. Latency comparison between OVS and OVS-DPDK with varying packet size.

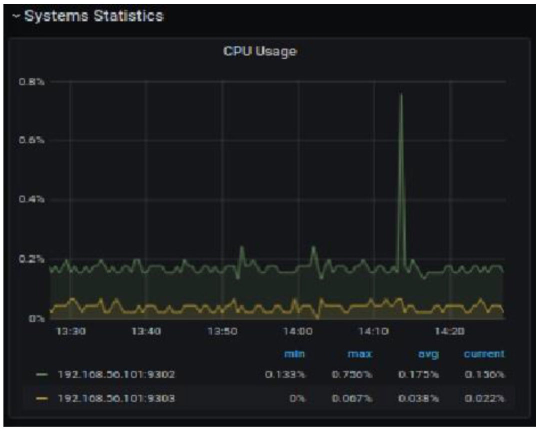


	Snort 2.8 AF_PACKET	Suricata 2.10 AF_PACKET	Snort 3.0 Libpcap	Snort 3.0 AF_PACKET	Suricata 4.1 Libpcap	Suricata 4.1 AF_PACKET	D3 DPDK
CPU usage	57%	11.50%	46%	40%	30%	10%	100%
Memory usage	2%	6%	0.10%	0.20%	0.20%	0.10%	0.20%
Drop rate	13.80%	5.90%	0%	0%	0%	0%	0%

FIGURE 13. Performance comparison of different IDS under 10Gbps TCP flows.

in terms of CPU utilization, memory utilization, and the packet drop rate. The result shows that the D3 framework using DPDK as a packet capturing mechanism improves CPU utilization, reduces packet drop with optimum usage of memory. The multithreaded architecture of the DPDK framework increases performance. Regardless of its performance improvements compared to other IDS in [35] D3 framework incurs low-performance cost since it is executed in a container test environment and requires no hardware. The discussions of Figure 13 are

- *The newer versions of Snort and Suricata are better than the older versions.* The newer version of Snort is enabled by a multithreading framework which gives better performance, whereas the newer version of Suricata is integrated with extended BSD Packet Filter and XDP support, which delivers packet capturing process immediately after the reception from the hardware that increases the performance of Suricata 4.1. The enabling of eBPF and XDP decreases packet drop rates.
- *AF_PACKET packet capturing mechanism is more satisfactory than Libpcap.* Even though both are Linux native network sockets, AF_PACKET configure memory buffer for capturing packet compared to kernel. This packet capturing mechanism saves both CPU resources and time.
- *Integration of DPDK packet capturing in the D3 framework makes it better than other IDS in terms of better utilization of CPU with low overhead of memory and zero drop rate of packets.* DPDK bypasses the existing network stack for fast packet processing which boosts the performance of network application by a large margin with a set of libraries, processing techniques, and fast I/O forwarding. The DPDK multithreaded architecture optimizes IDS and maximizes its performance by reducing the overloading of IDS and by enhancing CPU utilization with a zero drop rate. It also provides low latency and zero-copy packet handling with a low-performance cost. Moreover, a lightweight statistical anomaly detection used in the D3 framework makes the detection easier and faster.



	min	max	avg	current
— 192.168.56.101:9302	0.133%	0.750%	0.175%	0.150%
— 192.168.56.101:9303	0%	0.067%	0.038%	0.022%

FIGURE 14. CPU overhead of controller in D3 framework.

3) CPU UTILIZATION OF CONTROLLER IN D3 FRAMEWORK

Figure 14 shows the CPU utilization of the controller indicating the overhead of the controller during DDoS attack flows. As per the analysis in [26], the CPU utilization increases when the attack occurs, where SD-Anti-DDoS [28] increases to 35%, and NewFramework [26] increases to 15%. The proposed D3 framework lingers on 0.2% of CPU utilization. The large number of flows during attack detection in SDN controller increase CPU utilization in both [28] and [26] framework. In the [28] method, the controller is responsible for collecting traffic information from the switches, processing the data, detecting suspicious traffic, and mitigating attacks, which increases the CPU utilization by 20% than [26]. In [26] defense mechanism, the data collection and triggering of suspicious traffic is performed by the data plane which reduces the controller overhead. The feature extractions, detection, and mitigations are performed by the controller once the trigger is generated. In the proposed D3 framework, the traffic collection, feature extraction, and attack detection with a trigger are performed on the data plane DPDK framework, which reduces the overhead of the controller to 0.02% drastically.

B. ALGORITHM EVALUATION

The packet detection mechanism is evaluated by taking the average of the 8 different test cases conducted in the

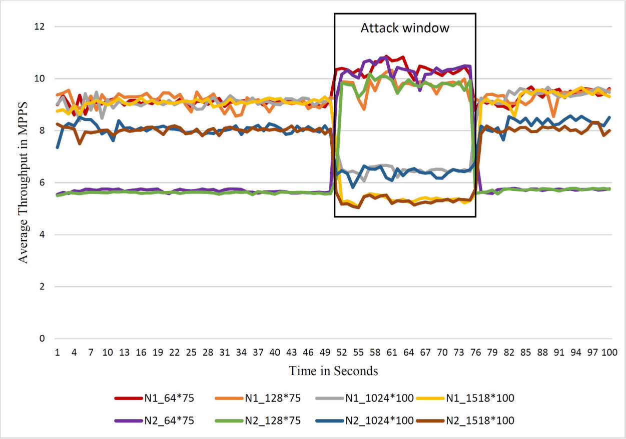


FIGURE 15. Attack window in the time series.

D3 framework. Since the test cases have to be conducted in high-speed network scenarios, both the normal and attack packets are generated by Pktgen-DPDK, wherein the normal traffic consists of N1 and N2 series having packet size of 256 and 512 with rate control of 25% and 50% respectively; and the attack series consist packet size of 64, 128,1024, 1518 with rate control of 75% and 100%. By maintaining a balanced dataset for both attack and normal series, the attack traffic is initiated from the 51^{st} second of the time series to the 75^{th} second of the time series, as shown in Figure 15, after a fixed training time interval.

The detection time and memory utilization are the two important parameters in the IDS framework against DDoS detection. The detection time (also known as detection power) indicates how fast the IDS can detect the attack without overwhelming the resources, whereas the memory utilization indicates the lightweight of the algorithm. The results of the D3 algorithm are compared with OverWatch [23] in the D3 framework since both are the predictive based algorithm.

The algorithm evaluation is performed in the D3 framework and validated by publicly available CIC DoS dataset using three performance metrics namely accuracy, F1-measure, and α-error in IDS which are shown from Eq. 5 and Eq. 7. The accuracy is defined as the ratio of the correctly classified instances to the total instances. The F1-measure shows the harmonic relationship between precision and recall, where the highest F1-measure indicates the high flow detection accuracy. The α-error is caused when there is no detection of attacks even though attacks exist. It is also termed as the false-negative rate, which is considered the most hazardous attack in IDS. So, the IDS should be efficient enough to detect α-error as quickly as possible before it corrupts the entire network or system.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$\alpha{-}error = \frac{FN}{FN + TP} \tag{6}$$

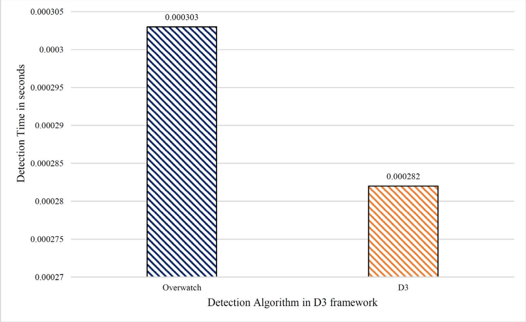


FIGURE 16. Detection time comparison.

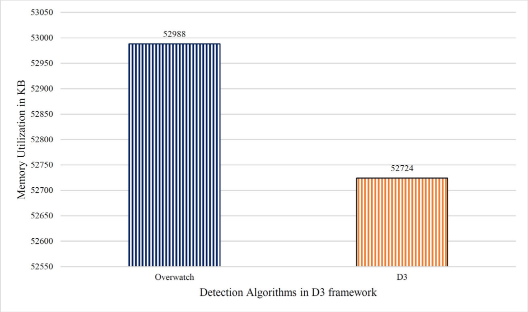


FIGURE 17. Memory utilization in D3 framework.

$$F1 - measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

where Precision $= \frac{TP}{TP+FP}$, Recall $= \frac{TP}{TP+FN}$

1) USING D3 FRAMEWORK

a: DETECTION TIME

As per the result analysis in [23], the detection time of the OverWatch is shown as 0.001 seconds. Figure 16 depicts the detection time of algorithms in the D3 framework, wherein the X-axis shows different detection algorithms and the Y-axis shows the detection time in seconds. The detection time of [23] in the OVS framework is reduced by 69.7% when the D3 framework is used. The proposed D3 algorithm is 71.8% faster than [23] in the OVS framework and 6.9% faster than OverWatch in the D3 framework. The detection power of the D3 framework is due to the advantage of OVS-DPDK and the usage of one metric rather than 4 metrics in [23].

b: MEMORY UTILIZATION

In the D3 framework, the memory utilization of the D3 algorithm is reduced by 0.5% than OverWatch as shown in Figure 17, wherein the X-axis shows different detection algorithms and the Y-axis shows the memory utilization in KB. A slight difference in memory execution is due to the single metric utilization in the D3 algorithm.

c: ACCURACY

The accuracy of the detection algorithms in the D3 framework is shown in Figure 18, wherein the X-axis shows different

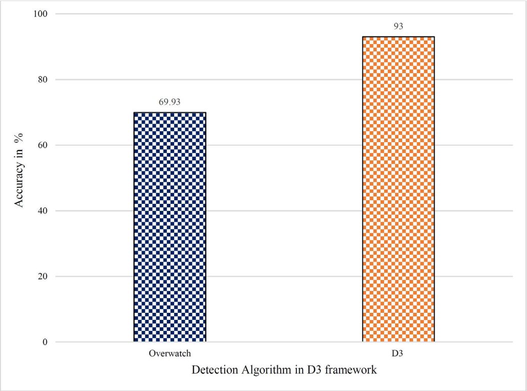


FIGURE 18. Accuracy comparison in D3 framework.

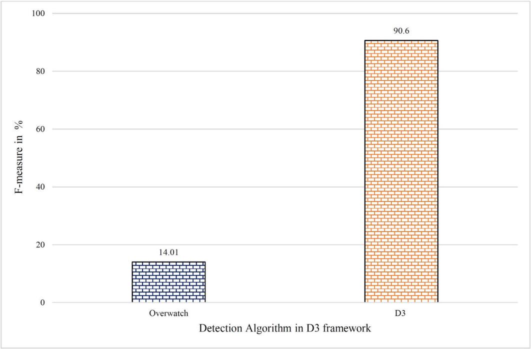


FIGURE 19. F1-measure comparison in D3 framework.

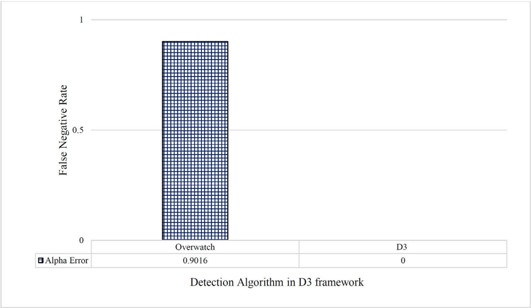


FIGURE 20. α-error comparison in D3 framework.

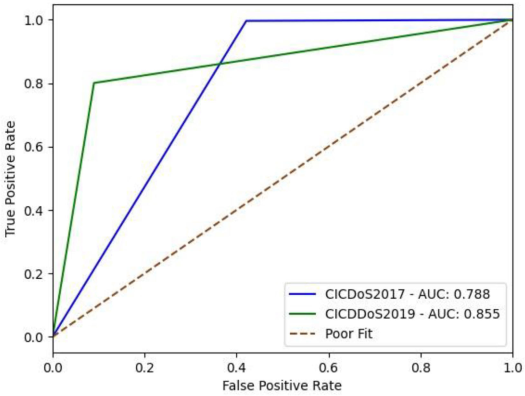


FIGURE 21. ROC of OverWatch.

detection algorithms and the Y-axis shows the percentage of accuracy. The D3 algorithm increases the accuracy by 24.81% than the [23] detection algorithm. The D3 algorithm is better than [23] since it uses network baseline and appended normal traffic for the next time series prediction.

d: F1-MEASURE

The F1-measure of detection algorithms in the D3 framework is shown in Figure 19, wherein the X-axis shows different detection algorithms and the Y-axis shows the percentage of F1-measure. The result depicts that the D3 algorithm has improved F1-measure by 84.54% compared to [23] algorithm in the D3 framework. The single elite feature in the D3 algorithm and prediction metric evaluation from the normal traffic delivers a better F1-measure compared to the [23] algorithm.

e: α-ERROR

The α-Error of the detection algorithms in the D3 framework is shown in Figure 20, wherein the X-axis shows different detection algorithms and the Y-axis shows the α-Error rate ranging from 0 to 1. The D3 algorithm decreases the α-error to zero, which shows that the D3 algorithm is 100% better than the [23] detection algorithm. The reason is that the attack traffics are excluded from the network baseline, which gives better prediction metrics for the next time series by reducing false negatives. Although the α-error is zero, the D3 algorithm has false alarm (FP) rate of 0.0939, which correspondingly lowers the F1-measure.

2) USING CIC DoS DATASETS

To validate the performance of the detection algorithm, the performance metrics of the detection algorithms are also analyzed with different publicly available CIC DoS datasets namely $CICDoS2017$ [75] and $CICDDoS2019$ [76]. The $CICDoS2017$ dataset comprises of low-volume application layer DoS attack dataset with 8 different attacks and has a total size of 4.6GB. The $CICDDoS2019$ dataset containing volumetric attacks generated on March 11^{th}, 2019 recorded 7 attacks [77].

a: ROC GRAPH

AUC value shows the degree of separability between classes. The ROC curve of the OverWatch algorithm and D3 algorithm for the different datasets are shown in Figure 21 and Figure 22 respectively. The X-axis depicts the False Positive Rate and the Y-axis depicts the True Positive Rate of the algorithm. The AUC value of OverWatch for CICDoS2017 and CICDDoS2019 are 0.788 and 0.855 respectively, whereas the AUC value of the D3 algorithm for CICDoS2017 and CICDDoS2019 are 0.855 and 0.9 respectively. Here, the AUC value of the D3 algorithm increases 5% than the OverWatch algorithm in both datasets, which displays the efficiency of the

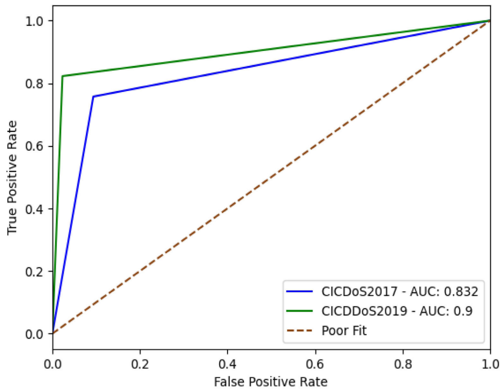


FIGURE 22. ROC of D3.

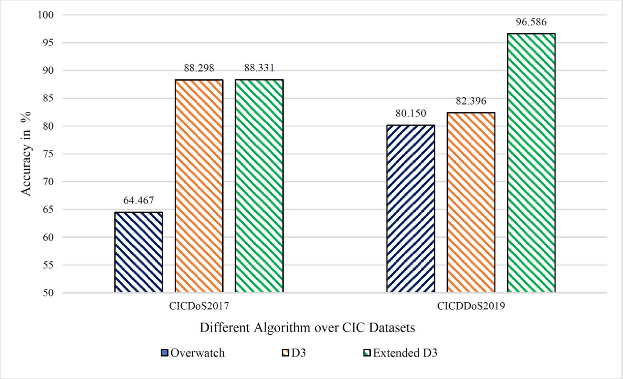


FIGURE 23. Accuracy of CIC DoS datasets.

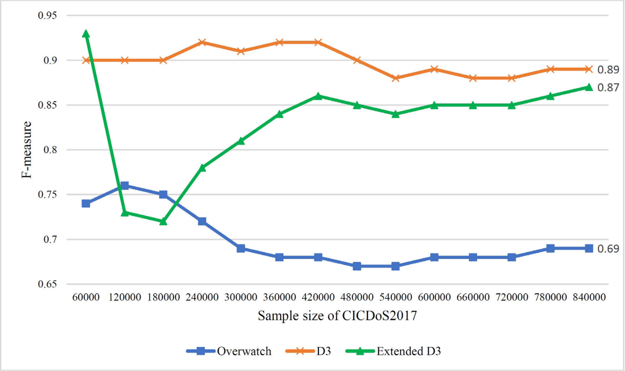


FIGURE 24. F1-measure of CICDoS2017.

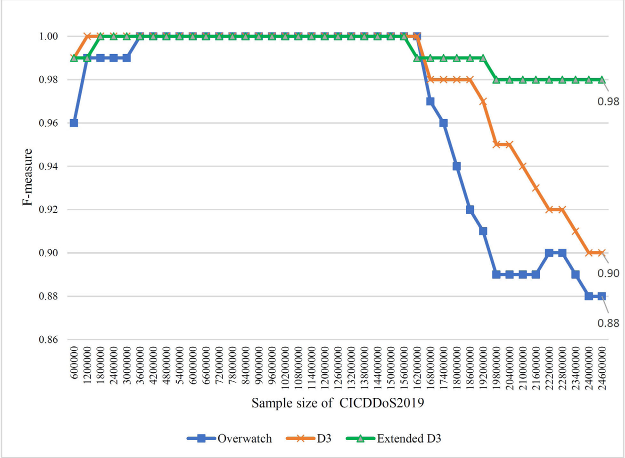


FIGURE 25. F1-measure of CICDDoS2019.

D3 anomaly detection algorithm to classify between classes. Moreover, there is a difference in the AUC value between datasets, where the AUC value of CICDoS2017 is lesser than the CICDDoS2019, as the CICDoS2017 consists of low volume application DoS attack, which is difficult to identify than volumetric attacks in the CICDDoS2019 dataset.

b: ACCURACY

The accuracy of the detection algorithms using CICDoS2017 datasets and CICDDoS2019 datasets is shown in Figure 23, wherein the X-axis shows different detection algorithms and the Y-axis shows the percentage of accuracy. Initially, both datasets are trained for 100 samples. For the CICDoS2017 dataset, the D3 algorithm increases the accuracy by 27% than the [23] detection algorithm. Similarly, for the CICDDoS2019 dataset, the D3 algorithm increases the accuracy by 2.72% than [23] detection algorithm. The D3 algorithm is better than [23] because the next time-series prediction is based on network baseline and appended normal traffic.

To find the effect of training samples over the D3 algorithm, we increase the training set from 100 to 500 samples, which is named as 'Extended D3', wherein the CICDoS2017 increases the accuracy by 27% than OverWatch which is the same as D3, whereas the CICDDoS2019 increases the accuracy by 17% which is 14.69% more than the D3. This gives two conclusions that the D3 detection is always superior to the OverWatch algorithm in accuracy and the nature of the datasets determines whether to extend the training set.

c: F1-MEASURE

To analyze the nature of flow detection accuracy over CIC DoS datasets, the F1-measure of the CICDoS2017 dataset and CICDDoS2019 dataset with varying sample sizes are shown in Figure 24 and Figure 25 respectively. The X-axis represents the sample size of the dataset and the Y-axis represents F1-measure, where the D3 algorithm has the highest F1-measure than OverWatch.

The extensive training of the D3 algorithm brings different effects in the F1-measure of CICDoS2017 datasets and CICDDoS2019 datasets. The CICDoS2017 dataset comprises of low-rate attacks which increase the false positive rate in extensive training of the D3 algorithm, whereas the CICDDoS2019 dataset comprises of mainly high rate attacks which decrease false-positive rate and increase the F1-measure during extensive training of D3. The high volumetric attacks in the CICDDoS2019 dataset have maintained a distinct boundary for both normal and attack sce-

TABLE 8. Comparison of different DDoS detection model in SDN with D3 framework.

Reference [Year]	DDoS detection in SDN (Yes/No)	Lightweight strategy (Yes/No)	Number of Features	Triggering Alert in data plane (Yes/No)	Feature Extraction in data plane (Yes/No)	Introducion of DPDK (Yes/No)	DDoS Solution in high-speed network (Yes/No)	Mitigation solution in control plane (Yes/No)
[17] [2010]	Yes	Yes	6 features	No	No	No	No	Not explained
[23] [2018]	Yes	Yes	4 features	Yes	Yes	No	No	Yes
[26] [2020]	Yes	Yes	5 features	Yes	No	No	No	Yes
D3 (Proposed Model)	**Yes**	**Yes**	**1 feature**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

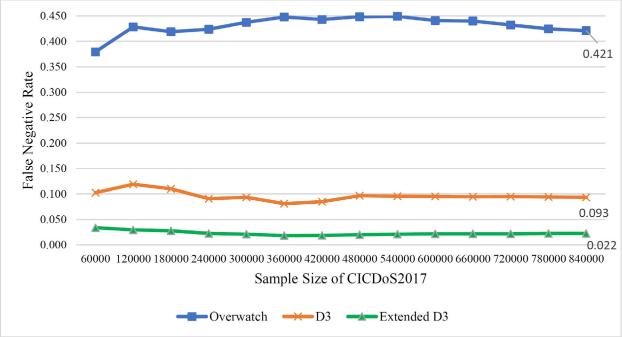


FIGURE 26. α-error of CICDoS2017.

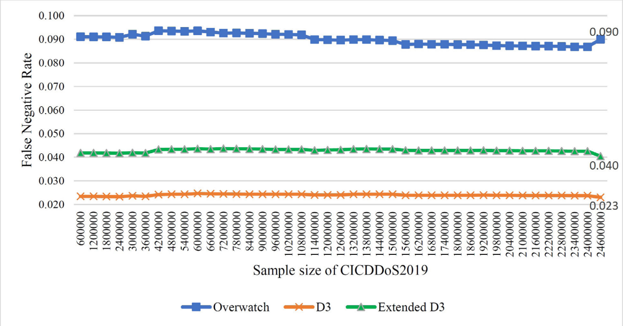


FIGURE 27. α-error of CICDDoS2019.

narios, whereas the low rate attacks in CICDoS2017 are unable to fix a boundary for the normal scenario. This increases the false positive rate during extensive training in CICDoS2017. Thus D3 algorithm with a small training set is optimum for unknown scenarios. The F1-measure of the CICDoS2017 dataset shows that the F1-measure of OverWatch has decreased by 22.47% compared to the D3 algorithm, whereas the extended D3 algorithm decreased by 2.24% due to the wrong boundary selection of extensive training. The F1-measure of the CICDDoS2019 dataset of OverWatch shows that the F1-measure of OverWatch has decreased by 2.22% compared to the D3 algorithm, whereas the extended D3 algorithm improved F1-measure by 8.89% due to good boundary selection of extensive training.

d: α-ERROR

The false-negative rate of the CICDoS2017 dataset and the CICDDoS2019 dataset with changing sample size are shown in Figure 26 and Figure 27 respectively, wherein the X-axis represents the sample size of the CIC DoS dataset and the Y-axis represents α-error.

For the CICDoS2017 dataset, the α-error rate of OverWatch is 0.4, whereas the α-error rate of the D3 algorithm is 0.1 but for the CICDDoS2019 dataset, the α-error rate of OverWatch and D3 algorithm are 0.09 and 0.04 respectively. The α-error rate of the CICDDoS2019 dataset is lesser than CICDoS2017 due to the difference in the detection rate of the volumetric attacks over low-rate attack detection. In both CIC DoS datasets, the D3 algorithm works better than OverWatch in finding the α-error, since the normal traffic is only considered for the traffic prediction.

Even though the D3 algorithm works better than OverWatch in both CIC DoS datasets, the extended training of D3 gives better results in the CICDoS2017 dataset, whereas the extended training of the D3 algorithm shows a negligible increase of false-negative rate in the CICDDoS2019 dataset. It is due to the inconsistency of network traffic caused by a port scan attack in the CICDDoS2019 dataset. In the CICDoS2017 dataset, the D3 algorithm decreases the α-error by 77.91%, whereas the D3 algorithm with extended training decreases the α-error by 94.77% than the OverWatch algorithm. Similarly, in the CICDDoS2019 dataset, the D3 algorithm decreases the α-error by 74.44%, whereas the D3 algorithm with extended training decreases α-error by 55.55% than OverWatch algorithm.

e: COMPARISON OF DIFFERENT DDoS DETECTION IN SDN WITH D3 FRAMEWORK

The proposed work is compared with three lightweight DDoS strategies in the SDN network, which is shown in Table 8. The DDoS detection methods used in [17], [23], and [26] are unsupervised self-organizing map (SOM), predictive anomaly detection, and combined machine learning algorithm of K-Means and KNN respectively. The comparison results show that the D3 algorithm is superior to all other detection methods, where it uses the advantage of DPDK to provide the DDoS solution in high-speed networks. Table 9 depicts the simulated detection results of CIC DoS datasets for different DDoS detection techniques. The simulation result shows that the proposed D3 algorithm offers the highest accuracy and lowest α-error rate in CICDDoS2019 compared to other detection models, whereas the accuracy of the [26] in CICDoS2017 has a negligible increase of 0.43% than proposed D3 algorithm, but its α-error rate increases by

TABLE 9. Comparison of accuracy versus α-error of D3 algorithm with related works in CIC DoS datasets.

CIC DoS Dataset	Reference	Accuracy	alpha-error
CICDoS2017	[17]	67.08%	0.646
	[23]	64.47%	0.421
	[26]	88.72%	0.708
	D3 (Proposed work)	**88.33%**	**0.022**
CICDDoS2019	[17]	89.62%	0.104
	[23]	80.15%	0.090
	[26]	87.66%	0.124
	D3 (Proposed work)	**96.59%**	**0.040**

96.89% than proposed D3 algorithm. Thus, the comparison results show the efficacy of the D3 algorithm to achieve maximum accuracy and minimum α-error rate with single feature extraction in both CIC DoS datasets.

The key findings of all the above experiments can be summarized as follows: The D3 framework is highly efficient compared to the existing systems in the SDN environment as it offers an ideal packet capturing (DPDK) mechanism compared to other IDS, enhances the performance than the OVS framework, and provides low controller overhead in a high-speed network. Moreover, the detection ability of the D3 algorithm is superior in a high-speed network while comparing its detection time, memory utilization, accuracy, F1-measure, and α-error. The algorithm evaluation is also validated by three different DDoS detections in SDN using publicly available CIC DoS datasets.

VII. CONCLUSION & FUTURE WORK

DDoS detection is a hard problem in the cyber world to be quickly identified without overwhelming the resources. The proposed approach presents a fast DDoS Detection framework using a single statistical parameter in the DPDK framework of SDN architecture. This D3 framework solves the problem regarding (i) the discordant relationship of DDoS attack and SDN architecture (ii) the limitation of IDS in the high-speed network. Moreover, the D3 detection algorithm provides a good prediction of attacks with good detection performance. The experimental results show that the D3 framework is successful in building a trade-off between the detection effect and efficiency of the framework in a high-speed network. It ensures a low α-error rate with a high detection rate and detection power. Furthermore, it provides a cost-effective approach with no external hardware usage. This proof concept of IDS against DDoS attack is ideal for the application areas like data centers, cooperation, government, educational institution, etc.

This is the initial phase of the D3 framework. Due to the limitation of the experimental environment, the scaling of the framework is not performed. In future, the proposed system can be advanced by (i) scaling up the framework with more destination ports and for larger attacks, (ii) introducing an adaptive threshold detection algorithm, (iii) optimization technique in the DPDK detection framework, (iv) expanding the mitigation module with various mitigation solutions, and (v) dynamic resource management with load balancing technique.

REFERENCES

[1] C. Cimpanu. (2020). *AWS Said it Mitigated a 2.3 TBPS DDoS Attack, Largest Ever*. Accessed: Aug. 3, 2020. [Online]. Available: https://www.zdnet.com/article/aws-said-it-mitigated-a-2-3-tbps-ddos-attack-the-largest-ever/

[2] Cloudflare. (2018). *Famous DDoS Attacks | The Largest DDoS Attacks All Time*. Accessed: Mar. 2, 2018. [Online]. Available: https://www.cloudflare.com/learning/ddos/famous-ddos-attacks/

[3] D. Geneiatakis, G. Portokalidis, and A. D. Keromytis, "A multilayer overlay network architecture for enhancing IP services availability against DoS," in *Proc. Int. Conf. Inf. Syst. Secur.* Berlin, Germany: Springer, 2011, pp. 322–336.

[4] X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: Network-layer DoS defense against multimillion-node botnets," in *Proc. ACM SIGCOMM Conf. Data Commun.*, 2008, pp. 195–206.

[5] H. Wang, Q. Jia, D. Fleck, W. Powell, F. Li, and A. Stavrou, "A moving target DDoS defense mechanism," *Comput. Commun.*, vol. 46, pp. 10–21, Jun. 2014.

[6] P. Mittal, D. Kim, Y. Hu, and M. Caesar, "Mirage: Towards deployable DDoS defense for Web applications," *CoRR*, vol. abs/1110.1060, Oct. 2011. [Online]. Available: http://arxiv.org/abs/1110.1060

[7] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 2046–2069, Mar. 2013.

[8] A. Mahimkar, J. Dange, V. Shmatikov, H. M. Vin, and Y. Zhang, "dFence: Transparent network-based denial of service mitigation," in *Proc. NSDI*, vol. 7, 2007, pp. 327–340.

[9] A. Akhunzada, E. Ahmed, A. Gani, M. K. Khan, M. Imran, and S. Guizani, "Securing software defined networks: Taxonomy, requirements, and open issues," *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 36–44, Apr. 2015.

[10] Z. Shu, J. Wan, D. Li, J. Lin, A. V. Vasilakos, and M. Imran, "Security in software-defined networking: Threats and countermeasures," *Mobile Netw. Appl.*, vol. 21, no. 5, pp. 764–776, Oct. 2016.

[11] N. Z. Bawany, J. A. Shamsi, and K. Salah, "DDoS attack detection and mitigation using SDN: Methods, practices, and solutions," *Arabian J. Sci. Eng.*, vol. 42, no. 2, pp. 425–441, Feb. 2017.

[12] T. Mahjabin, Y. Xiao, G. Sun, and W. Jiang, "A survey of distributed denial-of-service attack, prevention, and mitigation techniques," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 12, 2017, Art. no. 1550147717741463.

[13] K. Benzekki, A. El Fergougui, and A. E. Elalaoui, "Software-defined networking (SDN): A survey," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5803–5833, Dec. 2016.

[14] N. Dayal, P. Maity, S. Srivastava, and R. Khondoker, "Research trends in security and DDoS in SDN," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 6386–6411, Dec. 2016.

[15] J. Singh and S. Behal, "Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions," *Comput. Sci. Rev.*, vol. 37, Aug. 2020, Art. no. 100279.

[16] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 413–424.

[17] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in *Proc. IEEE Local Comput. Netw. Conf.*, Oct. 2010, pp. 408–415.

[18] S. M. Mousavi and M. St-Hilaire, "Early detection of DDoS attacks against SDN controllers," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2015, pp. 77–81.

[19] S. R. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba, "PayLess: A low cost network monitoring framework for software defined networks," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, May 2014, pp. 1–9.

[20] J. Seo, C. Lee, T. Shon, K.-H. Cho, and J. Moon, "A new DDoS detection model using multiple SVMs and TRA," in *Proc. Int. Conf. Embedded Ubiquitous Comput.* Berlin, Germany: Springer, 2005, pp. 976–985.

[21] M. Ambrosin, M. Conti, F. De Gaspari, and R. Poovendran, "LineSwitch: Tackling control plane saturation attacks in software-defined networking," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 1206–1219, Apr. 2017.

[22] R. Swami, M. Dave, and V. Ranga, "Software-defined networking-based DDoS defense mechanisms," *ACM Comput. Surveys*, vol. 52, no. 2, pp. 1–36, May 2019.

[23] B. Han, X. Yang, Z. Sun, J. Huang, and J. Su, "Overwatch: A cross-plane ddos attack defense framework with collaborative intelligence in SDN," *Secur. Commun. Netw.*, vol. 2018, Jan 2018, Art. no. 9649643, doi: 10.1155/2018/9649643.

[24] K. Kalkan, G. Gür, and F. Alagöz, "SDNScore: A statistical defense mechanism against DDoS attacks in SDN environment," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2017, pp. 669–675.

[25] J. Boite, P.-A. Nardin, F. Rebecchi, M. Bouet, and V. Conan, "Statesec: Stateful monitoring for DDoS protection in software defined networks," in *Proc. IEEE Conf. Netw. Softwarization (NetSoft)*, Jul. 2017, pp. 1–9.

[26] L. Tan, Y. Pan, J. Wu, J. Zhou, H. Jiang, and Y. Deng, "A new framework for DDoS attack detection and defense in SDN environment," *IEEE Access*, vol. 8, pp. 161908–161919, 2020.

[27] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. P. C. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics," *Future Gener. Comput. Syst.*, vol. 89, pp. 685–697, Dec. 2018.

[28] Y. Cui, L. Yan, S. Li, H. Xing, W. Pan, J. Zhu, and X. Zheng, "SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks," *J. Netw. Comput. Appl.*, vol. 68, pp. 65–79, Jun. 2016.

[29] Radware, NEC Corporation of America—Whitepaper. (2012). *Denial-of-Service (DoS) Secured Virtual Tenant Networks (VTN)*. Accessed: Oct. 21, 2019. [Online]. Available: https://www.necam.com/Docs/?id=0078a286-d99d-4d2c-ab54-d18814b59dd

[30] M. Kia, "Early detection and mitigation of DDoS attacks in software defined networks," M.S. thesis, Appl. Sci., Ryerson Univ., Toronto, ON, Canada, 2015.

[31] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to DDoS attack detection and response," in *Proc. DARPA Inf. Survivability Conf. Expo.*, vol. 1, 2003, pp. 303–314.

[32] N. Hoque, H. Kashyap, and D. K. Bhattacharyya, "Real-time DDoS attack detection using FPGA," *Comput. Commun.*, vol. 110, pp. 48–58, Sep. 2017.

[33] X.-D. Zang, J. Gong, and X.-Y. Hu, "An adaptive profile-based approach for detecting anomalous traffic in backbone," *IEEE Access*, vol. 7, pp. 56920–56934, 2019.

[34] W. Bulajoul, A. James, and M. Pannu, "Network intrusion detection systems in high-speed traffic in computer networks," in *Proc. IEEE 10th Int. Conf. e-Bus. Eng.*, Sep. 2013, pp. 168–175.

[35] Q. Hu, S.-Y. Yu, and M. R. Asghar, "Analysing performance issues of open-source intrusion detection systems in high-speed networks," *J. Inf. Secur. Appl.*, vol. 51, Apr. 2020, Art. no. 102426.

[36] Q. Hu, M. R. Asghar, and N. Brownlee, "Evaluating network intrusion detection systems for high-speed networks," in *Proc. 27th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, Nov. 2017, pp. 1–6.

[37] S. Campbell and J. Lee, "Intrusion detection at 100G," in *Proc. State Pract. Rep. (SC)*, 2011, pp. 1–9.

[38] J. Yang, L. Jiang, X. Bai, H. Peng, and Q. Dai, "A high-performance round-robin regular expression matching architecture based on FPGA," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2018, pp. 1–7.

[39] L. Schaelicke and J. C. Freeland, "Characterizing sources and remedies for packet loss in network intrusion detection systems," in *Proc. IEEE Int. Workload Characterization Symp.*, Oct. 2005, pp. 188–196.

[40] T. H. Ptacek and T. N. Newsham, "Insertion, evasion, and denial of service: Eluding network intrusion detection," Tech. Rep. ADA391565, 1998.

[41] X. Wu, P. Li, Y. Ran, and Y. Luo, "Network measurement for 100 GbE network links using multicore processors," *Future Gener. Comput. Syst.*, vol. 79, pp. 180–189, Feb. 2018.

[42] G. Bianchi, M. Bonola, A. Capone, and C. Cascone, "OpenState: Programming platform-independent stateful openflow applications inside the switch," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 44–51, Apr. 2014.

[43] N. Katta, O. Alipourfard, J. Rexford, and D. Walker, "CacheFlow: Dependency-aware rule-caching for software-defined networks," in *Proc. Symp. SDN Res.*, Mar. 2016, pp. 1–12.

[44] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[45] P. Zhang, H. Wang, C. Hu, and C. Lin, "On denial of service attacks in software defined networks," *IEEE Netw.*, vol. 30, no. 6, pp. 28–33, Nov. 2016.

[46] R. Kandoi and M. Antikainen, "Denial-of-service attacks in OpenFlow SDN networks," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 1322–1326.

[47] H. T. N. Tri and K. Kim, "Assessing the impact of resource attack in software defined network," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2015, pp. 420–425.

[48] D. B. Rawat and S. R. Reddy, "Software defined networking architecture, security and energy efficiency: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 325–346, 1st Quart., 2017.

[49] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 623–654, 1st Quart., 2016.

[50] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in software defined networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2317–2346, 4th Quart., 2015.

[51] T. Xu, D. Gao, P. Dong, C. H. Foh, and H. Zhang, "Mitigating the table-overflow attack in software-defined networking," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 4, pp. 1086–1097, Dec. 2017.

[52] R. Durner, C. Lorenz, M. Wiedemann, and W. Kellerer, "Detecting and mitigating denial of service attacks against the data plane in software defined networks," in *Proc. IEEE Conf. Netw. Softwarization (NetSoft)*, Jul. 2017, pp. 1–6.

[53] P. Dong, X. Du, H. Zhang, and T. Xu, "A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[54] B. Yuan, D. Zou, S. Yu, H. Jin, W. Qiang, and J. Shen, "Defending against flow table overloading attack in software-defined networks," *IEEE Trans. Services Comput.*, vol. 12, no. 2, pp. 231–246, Mar. 2019.

[55] S. M. Mousavi and M. St-Hilaire, "Early detection of DDoS attacks against software defined network controllers," *J. Netw. Syst. Manage.*, vol. 26, no. 3, pp. 573–591, Jul. 2018.

[56] H. Wang, L. Xu, and G. Gu, "FloodGuard: A DoS attack prevention extension in software-defined networks," in *Proc. 45th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Jun. 2015, pp. 239–250.

[57] (2018). *Dpdk*. Accessed: Mar. 3, 2018. [Online]. Available: http://www.dpdk.org/

[58] G. Pongracz, L. Molnár, and Z. L. Kis, "Removing roadblocks from SDN: OpenFlow software switch performance on Intel DPDK," in *Proc. 2nd Eur. Workshop Softw. Defined Netw.*, Oct. 2013, pp. 62–67.

[59] X. Zhao, "Study on DDoS attacks based on DPDK in cloud computing," in *Proc. 3rd Int. Conf. Comput. Intell. Commun. Technol. (CICT)*, Feb. 2017, pp. 1–5.

[60] B. Yi, X. Wang, K. Li, S. K. Das, and M. Huang, "A comprehensive survey of network function virtualization," *Comput. Netw.*, vol. 133, pp. 212–262, Mar. 2018.

[61] (2019). *DPDK PKTGEN*. Accessed: Jun. 23, 2020. [Online]. Available: https://github.com/Pktgen/Pktgen-DPDK

[62] (2016). *Faucet Opensource SDN Controller for Production Network*. Accessed: Aug. 2, 2018. [Online]. Available: https://faucet.nz/

[63] (2019). *Debug Test DPDK Applications*. Accessed: Sep. 24, 2019. [Online]. Available: https://software.intel.com/en-us/articles/debug-and-test-dpdk-applications-with-testpmd

[64] (2016). *DPDK Testpmd*. Accessed: Nov. 7, 2018. [Online]. Available: https://github.com/ceph/dpdk/blob/master/app/test-pmd/testpmd.c

[65] W. Queiroz, M. A. M. Capretz, and M. Dantas, "An approach for SDN traffic monitoring based on big data techniques," *J. Netw. Comput. Appl.*, vol. 131, pp. 28–39, Apr. 2019.

[66] C. Wang, J. Caja, and E. Gómez, "Comparison of methods for outlier identification in surface characterization," *Measurement*, vol. 117, pp. 312–325, Mar. 2018.

[67] M. Abdurohman, D. Prasetiawan, and F. A. Yulianto, "Improving distributed denial of service (DDOS) detection using entropy method in software defined network (SDN)," *ComTech, Comput., Math. Eng. Appl.*, vol. 8, no. 4, pp. 215–221, 2017.

[68] Y. Wang, T. Hu, G. Tang, J. Xie, and J. Lu, "SGS: Safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking," *IEEE Access*, vol. 7, pp. 34699–34710, 2019.

[69] Y. Yu, L. Guo, Y. Liu, J. Zheng, and Y. Zong, "An efficient SDN-based DDoS attack detection and rapid response platform in vehicular networks," *IEEE Access*, vol. 6, pp. 44570–44579, 2018.

[70] H. Peng, Z. Sun, X. Zhao, S. Tan, and Z. Sun, "A detection method for anomaly flow in software defined network," *IEEE Access*, vol. 6, pp. 27809–27817, 2018.

[71] Y. Xu, H. Sun, F. Xiang, and Z. Sun, "Efficient DDoS detection based on K-FKNN in software defined networks," *IEEE Access*, vol. 7, pp. 160536–160545, 2019.

[72] W. Yang, B.-X. Fang, B. Liu, and H.-L. Zhang, "Intrusion detection system for high-speed network," *Comput. Commun.*, vol. 27, no. 13, pp. 1288–1294, 2004.

[73] T. Rang, "NFV performance benchmarking with OVS and Linux containers," Karlstad Univ., 2017. [Online]. Available: http://kau.divaportal.org/smash/get/diva2:1111361/FULLTEXT01.pdf

[74] S. Shanmugalingam, A. Ksentini, and P. Bertin, "DPDK open vSwitch performance validation with mirroring feature," in *Proc. 23rd Int. Conf. Telecommun. (ICT)*, May 2016, pp. 1–6.

[75] C. I. for Cybersecurity. (2017). *CIC DoS Dataset (2017)*. Accessed: Aug. 2, 2019. [Online]. Available: https://www.unb.ca/cic/datasets/dos-dataset.html

[76] (2019). *DDoS Eval. Dataset (CIC-DDoS2019)*. Accessed: Jan. 8, 2020. [Online]. Available: https://www.unb.ca/cic/datasets/ddos-2019.html

[77] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2019, pp. 1–8.

JOSY ELSA VARGHESE received the B.Tech. degree in information technology from the Cochin University of Science and Technology, Kerala, India, in 2009, and the M.E. degree in computer science and engineering from Satyabama University, Chennai, India, in 2012. She is currently pursuing the Ph.D. degree with the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal. Her research interests include network security, intrusion detection systems, software defined networks, and DoS attacks.

BALACHANDRA MUNIYAL (Member, IEEE) received the B.E. degree in computer science and engineering from Mysore University and the M.Tech. and Ph.D. degrees in computer science and engineering from the Manipal Academy of Higher Education, Manipal, India. He worked with Manipal International University, Malaysia, in 2014. He is currently working as a Professor with the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal. He did his M.Tech. Project work in T-System Nova GmBH, Bremen, Germany. He has 27 years of teaching experience in various institutes. He has more than 50 publications in national and international conferences/journals. His research interests include network security, crypotography, and intrusion detection systems.