IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 August 2024, accepted 29 October 2024, date of publication 7 November 2024, date of current version 15 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3493384

RESEARCH ARTICLE

Survey: Intrusion Detection System in Software-Defined Networking

AHMED H. JANABI[1], (Member, IEEE), TRIANTAFYLLOS KANAKIS[2], (Member, IEEE), AND MARK JOHNSON[2]

[1]Computer Techniques Engineering Department, College of Engineering and Technology, Al-Mustaqbal University, Hillah, Babylon 51001, Iraq
[2]Department of Computing, University of Northampton, NN1 5PH Northampton, U.K.

Corresponding author: Ahmed H. Janabi (Ahmed.Janabi@uomus.edu.iq)

ABSTRACT In the rapidly evolving field of network architecture, Software-Defined Networking (SDN) has emerged as a transformative approach, providing unprecedented flexibility and control over network resources. While SDN enhances efficiency and programmability, it also introduces various security vulnerabilities, primarily due to its architecture, which distinctly separates the control plane from the data plane. This division enables dynamic and adaptable network management but also exposes networks to sophisticated cyber threats, including Distributed Denial of Service (DDoS) attacks, SQL injections, and other forms of intrusion targeting the centralised SDN controllers and open interfaces of its switches. This paper explores the complex security landscape of SDN, identifying critical vulnerabilities within this modern networking model. By analysing prevalent network attacks such as DDoS, DoS, Probe, and SQL Injection, we underscore the pressing need for resilient intrusion detection systems (IDS) that are specifically designed to meet the unique security challenges of SDN environments. Our investigation highlights significant gaps in current research, particularly in the development of real-time traffic processing and system overload mitigation strategies, both of which are vital for establishing durable and resilient SDN architectures. This study contributes to the discourse on SDN security by proposing a strategic framework for developing sophisticated IDS solutions that can adapt to the evolving dynamics of network threats. Our findings emphasise the importance of continuous innovation and a focus on sustainable, secure infrastructure within Software-Defined Networking, supporting its role as a safe and efficient foundation for future network developments.

INDEX TERMS Software-defined networking (SDN), intrusion detection system (IDS), cybersecurity, deep learning (DL), dataset, machine learning (ML).

I. INTRODUCTION

The advent of SDN marks a significant evolution in the domain of network architecture, heralding a new era of efficiency, flexibility, and programmability. At the heart of SDN lies the decoupling of the control plane from the data plane, a revolutionary concept that allows network administrators unprecedented control over network traffic flow and management. This paradigm shift, while facilitating agile network configurations and optimisations, also brings to light a spectrum of security vulnerabilities inherent to the SDN architecture. The centralised nature of the SDN control plane, coupled with the programmability and openness of network interfaces, makes SDN environments particularly susceptible to a range of cyber threats, from Distributed Denial of Service (DDoS) attacks to sophisticated intrusion attempts.

In light of these challenges, the need for robust, dynamic, and adaptive security measures is more pronounced than ever. Among the plethora of security solutions, IDS stands out as a critical line of defence, designed to detect, analyse, and mitigate malicious activities within the network. However, the unique characteristics of SDN require a re-evaluation and redesign of traditional IDS approaches to effectively address

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

the specific vulnerabilities of SDN infrastructures. The increasing sophistication of cyber threats further exacerbates this need, calling for innovative solutions that not only can detect known attack vectors, but also anticipate and neutralise emerging threats.

This paper embarks on an in-depth exploration of the security vulnerabilities associated with SDN, with a keen focus on understanding the nature and impact of various network attacks that threaten the integrity and reliability of SDN environments. Drawing from a review of existing literature and identifying critical gaps in current research, we identified gaps in intrusion detection frameworks tailored to the unique demands of SDN. Our research is motivated by the pressing need to fortify SDN networks against an ever-evolving landscape of cyber threats, ensuring the resilience and sustainability of this transformative networking paradigm.

As we navigate through the intricacies of SDN security challenges and solutions, our study aims to contribute significantly to the body of knowledge on network security. This research not only addresses the identified research gaps but also lays the groundwork for future innovations in network security. Our ultimate goal is to enhance the security posture of SDN environments, paving the way for their safe and effective deployment across various sectors.

II. BACKGROUND AND THEORETICAL FRAMEWORK

A. SDN OVERVIEW

SDN, short for Software-Defined Network, embodies an innovative paradigm in network architecture. This technology enables the network to be centrally and intelligently managed by applying several applications, such as traffic classification and security applications [1]. The rapid expansion of the Internet presents challenges for traditional networks, which are inherently static and possess limited capacity to adapt to evolving organisational needs. SDN emerges as a new architecture that addresses these difficulties and offers promising solutions. Nevertheless, these centralised and programmable techniques face various challenges and issues that require contemporary security solutions, such as IDS [2]. In recent times, the majority of security solutions have integrated ML techniques. In particular, DL algorithms have been increasingly used to improve accuracy and efficiency. In this section, an overview of SDN is provided, along with its security vulnerabilities and existing solutions. In addition, various types of attacks are presented.

SDN enables networks to be controlled by several applications that grant fewer networking devices with simple physical connectivity and configurations. As a result, network operators can tailor the behaviour of the network to support modern services and security applications [3].

SDN is managed by using lower-level functionality abstraction. The main characteristic of SDN is the separation of the control and data planes using the Application Programming Interface (API). SDN decouples the forwarding and control functions of the network [4]. The forwarding devices, representing switches and routers, are separated from the control logic and moved to the logical controller. This controller is considered the centralisation point of the network [4]. Thus, the controller organises the data plane. The decoupling of the control and data planes allows services and applications within the network to become programmable.

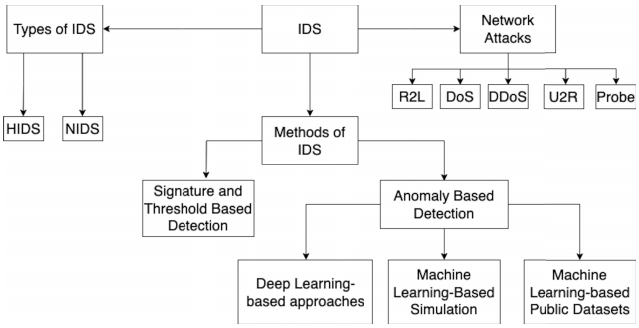

FIGURE 1. Taxonomy of IDS in SDN.

In other words, SDN disaggregates the traditionally vertical stacks of networking to customise and enhance scalability to suit the new technological environments. The primary goal of SDN is to allow network engineers and administrators to react quickly, allowing them to effectively deal with the dynamic needs of businesses [5].

Figure 1 demonstrates the taxonomy of IDS in SDN networks. Figure 2 presents the interrelations between the layers and planes of SDN and the SDN reference architecture. As indicated, by adopting the control plane, a network administrator can establish the network security policies and strategies through the application plane while also redirecting network traffic to various applications or systems [6].

This section provides an overview of the SDN and IDS. Details of the types of IDS that threaten SDN networks will be included in this section. The potential points that attackers target will be discussed thoroughly. Furthermore, the effects of attacks on the SDN environment are described in detail.

1) OpenFlow PROTOCOL

OpenFlow constitutes the communication interface between the control plane and the data plane in SDN networks. Consequently, data plane devices require the OpenFlow protocol to connect to the control plane [7]. It allows administrators to manage network traffic flow by decoupling the control plane from the data plane. OpenFlow was introduced for the first time by the Open Networking Foundation (ONF) in 2008 and has been widely accepted and adopted in both research and industry [8]. Moreover, the OpenFlow controller provides a precise, undisputed perspective of the network topology, providing straightforward access to the controller to network vulnerabilities and facilitating the definition of intrusions [9].

At its core, OpenFlow provides a standardised interface between the control and data planes. In traditional networking, these planes are tightly interconnected. However,

OpenFlow separates the control plane from the data plane, enabling a more flexible and programmable network infrastructure. This separation simplifies the implementation of security policies within SDN [10]. Many industry players are moving towards SDN technology to revolutionise network design and operation [11]. The following subsections will detail the main functions of the control and data planes.

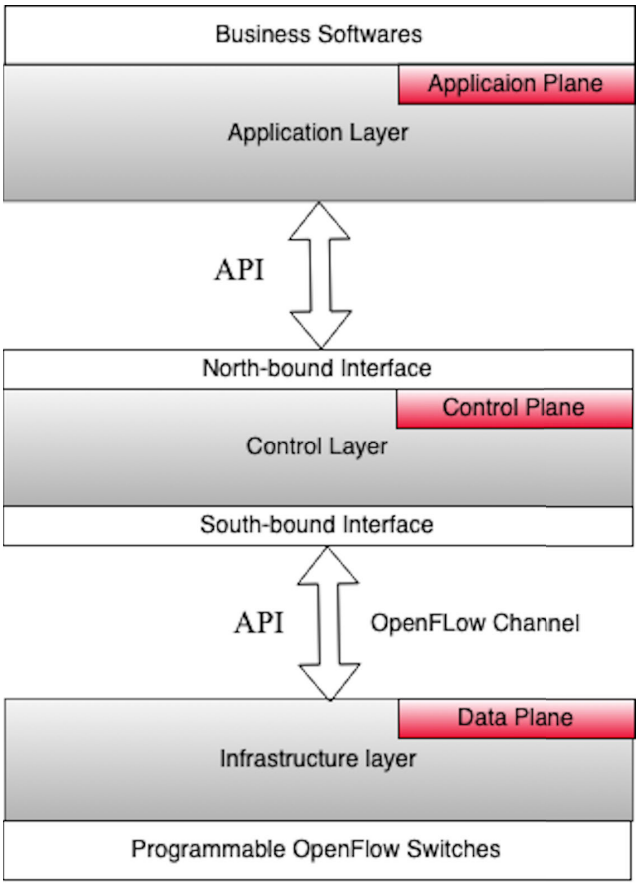

FIGURE 2. Application plane, control plane, and data plane.

2) DATA PLANE

The data plane is essential for transporting and processing network packets in SDN. It contains physical and virtual parts that implement forwarding functions. The data plane enables efficient traffic flow within the network. This plane is managed by the SDN controller, which ensures the correct dispatch of packets to their intended destinations [12].

The data plane processes incoming packets, making forwarding decisions based on established rules and redirecting them to their destinations. Switch features play a critical role in implementing forwarding actions in adherence to the specified rules. This separation of the control and data planes is the main difference between SDN and traditional networking paradigms, offering greater flexibility, programmability, and scalability [13].

In an SDN network, the data plane functions as a workhorse, its inherent programmability and dynamic nature adapting to varying network traffic conditions. Consequently, it facilitates systematic and intelligent routing, traffic engineering, quality of service (QoS) control, and other vital network functions. The advent of the data plane in SDN has revolutionised network management, providing administrators with unparalleled control and agility to meet the ever-evolving network demands [12].

3) CONTROL PLANE

The control plane in SDN is implemented through a centralised controller, which functions as the brain of the SDN networks. The control plane intercommunicates with the network devices in the data plane using a standardised protocol such as OpenFlow [6]. The main function of the control plane is traffic engineering. It determines the optimal paths for network traffic based on several factors, such as network congestion, bandwidth usage, and quality of service conditions. This plane optimises network resource management by dynamically modifying routing directions based on current conditions [14].

The control plane is responsible for implementing network policies. It allows administrators to define and enforce policies that govern the behaviour of network traffic with greater flexibility. These policies include access control rules and traffic prioritisation. The responsibility of this plane is to ensure that these policies are applied uniformly throughout the network [15].

Furthermore, the control plane enables network monitoring and troubleshooting. Collect real-time traffic information from the data plane and analyse it to detect anomalies, identify performance bottlenecks, and pinpoint vulnerable areas across the network. Furthermore, this information can be used to make informed decisions to maintain the highest possible network reliability and performance [16].

B. SECURITY FEATURES OF SDN

SDN possesses several characteristics that correspond to such modules. Its design features distinguish it from traditional network architectures. SDN's features enhance network security by allowing greater flexibility while being efficient and rapid [17]. However, SDNs can be inherently vulnerable and face threats that exploit such weaknesses and introduce overhead. Thus, descriptions of the SDN design features are considered from two points of view. The first is the aspect in which components protect the SDN framework against various threats. The second relates to the features that may make it vulnerable. Applications, including IDS, can be installed in the application plane of the SDN structure, as shown in Figure 3.

1) RESILIENCE CHARACTERISTICS IN SDN's STRUCTURE AGAINST ATTACKS

SDN offers many strategic features to deal with, for example:

1) **Centralising the monitoring of malformed traffic flows:** The controller manages the network's data. Hence, the controller can observe all suspicious activities throughout the network [18].
2) **Programmable configuration:** An essential advantage offered by SDN is its programmable features. When malicious behaviour is detected on the network, the new configuration of a programme acts instantaneously to address the identified anomalies [18].

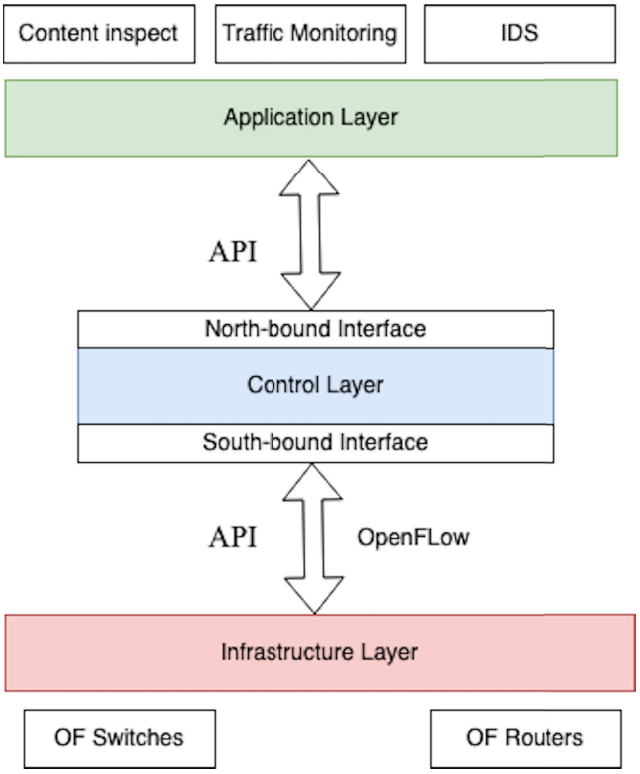

FIGURE 3. The architecture of SDN when using security applications.

2) VULNERABILITY TRAITS IN SDN's STRUCTURE TO ATTACKS

SDN's design has vulnerabilities that make it susceptible to various security threats.

1) **The Separation of the Planes:** The separation of the control and data planes introduces vulnerabilities to various attacks. Both planes transfer data to each other by employing the OpenFlow protocol. Therefore, an intruder can exploit these vulnerabilities by executing DoS, DDoS, and saturation attacks, leading to congestion in the channel bandwidth between the switch and the controller [19].
2) **The Controller Suffers from Cascading and Single-Point Failures:** In any SDN-based infrastructure, the controller is a primary target for intruders. As the SDN design is based on a centralised entity, it is prone to a single point of failure. If the controller crashes, most of the network functionalities will halt and security properties will be compromised [20]. A single controller is ineffective at handling significant network traffic; hence, the deployment of multiple controllers may address this issue. However, this can impact the integrity, scalability and consistency of privacy rules across each controller's domain, potentially causing multiple controllers to fail in a cascading manner.
3) **A Limited Ternary Content Addressable Memory (TCAM):** OpenFlow switches use TCAM to store flow rules in their flow tables. TCAM enables high-speed searches in applications. However, the flow tables of the OpenFlow switches in SDN have limited storage capacity, rendering the network vulnerable to several attacks [21].

C. NETWORKS ATTACKS

Network attacks represent any unauthorised attempts to access data, potentially compromising network security or disrupting services. These attacks are categorised into eight major groups, which are explained below.

1) **Distributed Denial of Service (DDoS) Attack:** DDoS is a kind of cyber-attack where numerous compromised systems are utilised to flood a target network or server with a large amount of traffic, blocking responses to normal requests. These attacks are organised by employing a network of computers, which is called a botnet, that is managed by the attacker. The size and distributed nature of the attack make it difficult to mitigate the impacts and identify the source of the attack [22].
   DDoS attacks manipulate the basic principles of network communication and can target different layers of the SDN network, including the control and data planes. Commonly, DDoS floods the victim with an excessive volume of packets, consuming network resources, such as bandwidth or server processing capacity, or exploits vulnerabilities in network protocols to disrupt communication among other parts of the network [23].
   The motivations behind DDoS attacks vary, ranging from malicious intent to seeking financial profit or advancing ideological agendas. In such attacks, the targets can be businesses or institutions with the aim of crippling their economy, causing financial losses, tarnishing their reputation, or facilitating data breaches. Common forms of DDoS attacks include Smurf, TCP SYN flooding, and teardrop [24].
2) **Denial of Service (DoS) Attack:** A DoS attack follows the same concept as a DDoS attack, with the main goal of both being to disrupt the resources of a targeted system or service. However, DoS attacks are implemented by utilising a single source, rather than multiple sources as in DDoS attacks [25].
3) **Port Scan Attack:**
   A port scan attack is defined as a mechanism used by an attacker to probe systems for open ports. This method

enables attackers to identify potential vulnerabilities and gain illegal access to the target system. The attack systematically queries a range of network ports on a given host to determine which ports are open, closed, or filtered. The attacker uses the obtained information to compromise sensitive services and threaten the victim. It is difficult to prevent this attack due to its stealthy nature [26].

4) **SQL Injection:** An SQL injection attack is a malicious utilisation that targets the vulnerabilities in SDN controllers and associated databases employing SQL applications. Due to the significant impact of SDN controllers in governing the network's policies, an SQL injection attack can seriously affect network security and functionality [27]. By injecting malicious SQL queries into user-supplied data, attackers can manipulate the network's policies, compromise the integrity of network flows, and gain unauthorised access to sensitive data. SDN architectures should implement robust security measures, such as IDS-based ML, to prevent this threat [28].
5) **Document Infiltration:** A document infiltration attack is a sophisticated and targeted cyberattack aimed at obtaining sensitive information stored in documents, whether in physical or electronic form. Hackers use complex intrusion strategies to breach a system's security and gain illegal access to documents [29].
These types of attacks exploit vulnerabilities in information systems, such as human errors, or by using advanced hacking techniques, including phishing, malware, or social engineering. Once the attacker gains access to the system, the intruder employs techniques to smuggle the targeted documents out, such as copying, downloading, or transmitting them to remote servers [30]. Document infiltration attacks pose a high risk to the confidentiality and integrity of sensitive information. Institutions must implement critical security measures, such as encryption techniques, access controls, and continuous monitoring, to detect and prevent such attacks [31].
6) **Probe Attack:** The hacker scans the entire network to gather information about the target machine. Using port sweeps, which identify the services running on the host machine, and Ping Sweep, which scans a wide range of IP addresses, the hacker maps out the live hosts [32].
7) **User to Root (U2R):** This attack is used to enable unauthorised logins to a host machine to gain superuser privileges. It is usually launched to obtain the root privileges of a user account. The basic types include input modification and buffer overflow [33].
8) **Remote to Local (R2L):** The R2L attack is an initial or traditional attempt to imitate offline users. In this method, attackers penetrate the system by sending data packets over the network. For the attack to succeed, the victim and the hacker must be able to reach each other or be on the same network. Unauthorised access to a user account can be gained through social engineering or password sniffing methods [34].

D. THE TARGETED POINTS FOR ATTACKS IN THE SDN

The SDN architecture comprises three planes: data, control, and application. The nature of SDN makes the network vulnerable to several types of attacks, including DDoS and DoS. The points targeted by attacks are described below:

1) **The SDN Switches:** The primary function of OpenFlow switches is to forward newly received traffic to the controller for action and processing. However, each switch has a flow table with a limited size of memory. Consequently, this limitation is a significant security concern, as attackers can flood the switch with a massive volume of malicious packets [35].
2) **The SDN Switch Links:** Packets transfer between switches before reaching the controller. These packets are not encrypted during transfer over the links, making it easy for attackers to intercept the packets, especially in wireless environments [36]. This type of attack is known as man-in-the-middle. DoS and DDoS attacks can also be executed at this juncture, where the attacker may send a large volume of malicious traffic to interrupt the transmission of regular traffic, thereby halting services.
3) **The SDN Controller:** As the core of the network, the controller executes critical operational functions. Any anomalies can halt network activity. The significant influence of the controller on network functionality makes it an attractive target for attackers. Using a single controller can create a single point of failure, representing a considerable security risk that needs to be mitigated [37].
4) **Controller and Switch Communication:** If a newly received packet does not match flow table records, it is redirected to the controller for further processing. The controller then inserts appropriate forwarding rules into the flow table entries of the specific switch. At this point, an attacker can intercept the packet to inject malicious rules or modify existing ones, ultimately causing many packets to be misdirected [37].
5) **Applications:** Applications in the SDN control layer, which include traffic monitoring and classification, are usually developed by third parties and are designed to bolster essential security measures. Attackers may target these applications to steal sensitive information or to inject malicious rules into the controller. An unauthenticated user could perform such actions during API communication, making SDN applications a direct point of attack for disrupting the controller's services [22].

E. THE ATTACKS EFFECTS ON THE SDN ENVIRONMENT

The most concerning threats in SDN networks include DDoS, DoS, R2L, U2R, and Probe attacks. These threats occur by

overloading the controller through bandwidth congestion of the communication between the data and control planes. The most significant threats and the potential effects of these attacks are discussed as follows:

1) **Saturation of the Controller Resources:** The controller is regarded as the core of the SDN network. Thus, if the controller fails, it can significantly impact network performance. The controller's resources may become depleted due to processing a high volume of flood requests from DDoS attacks. When the controller is overwhelmed, it cannot manage all the incoming flows [38]. Consequently, a substantial amount of regular traffic could be delayed or miss crucial processing [38].
2) **Switch Overloading:** The primary threats to SDN architectures are DoS and DDoS attacks. Such attacks can generate a multitude of malicious packets to flood the switches. When a switch fails to locate a matching entry for a malicious packet in the flow table, there is a designated buffer for all unmatched entry requests, which are then forwarded to the controller for rule application [39]. However, since the switch is limited by TCAM, not all incoming packets can be addressed. As a result, the incoming flow and requests might surpass the flow table's memory capacity. Hence, regular traffic could miss the required processing [40].
3) **Bandwidth Congestion between Switch and Controller:** Missing events arise during two key actions with new incoming packets. The first is when incoming packets are stored in a buffer within the flow table. The second takes place when an OpenFlow request, containing packet header data, is sent to and received by the controller once the buffer is at full capacity. Consequently, packets may clash at another bound interface, leading to a disruption of services for users [18].

F. INTRUSION DETECTION SYSTEM (IDS)

An IDS is essential, especially when a network enterprise has security concerns or handles sensitive data. The IDS provides defensive capabilities for the network by controlling and monitoring its traffic [41]. The IDS releases an alert to the administrator when it identifies malicious traffic and either redirects or filters this traffic according to requirements, or it can install special policies [42]. In network-based IDS, typical components include IDS management, IDS collector, and IDS classification. IDS management defines the policies and rules of the IDS, while the collector gathers traffic or flows from the flow table database. The classification component applies techniques for predictive purposes [43].

IDS examines every packet that arrives through the switches via the southbound API and then proceeds to the application plane through the northbound API. This process is carried out on the standard switch by configuring the controller. Thus, all network traffic passing through the switches can be observed and analysed by the controller [44].

1) TYPES OF INTRUSION DETECTION SYSTEM

IDS can be categorised based on their targets as follows:

1) **Host-based Intrusion Detection System (HIDS):** The IDS must be installed on a computer or personal device such as a mobile phone or tablet. It is an important application that monitors all the activities on the device. It detects and prevents attacks if there are intruders [45].
2) **Network-based Intrusion Detection System (NIDS):** This system observes the traffic to detect malicious activities by examining traffic behaviour at various stages throughout the entire network and raises an alarm when an anomaly is identified [46].

G. METHODS OF INTRUSION DETECTION SYSTEM IN SDN

The IDS analyses the data; the data source is the entire packet's data, which includes monitoring all flows crossing the network. This information includes the packet header, the number of bytes, and the number of packets for both destinations [2]. The IDS uses multiple analysis and detection methods to evaluate and monitor packets moving across the network [47]. These methods can be classified depending on how the mechanism detects an intrusion. The main techniques are signature-based and anomaly-based detection [48]. The main differences between them are shown in Table 1 [49].

TABLE 1. IDS methods comparison.

#	Signature-Based Methods	Anomaly-Based Methods
1	Implementation is easy	Difficult to implement
2	Reliable	Less Reliable
3	Speed is high	Speed is low
4	Less Robust	More Robust
5	Low rate of Alarm	High rate of Alarm
6	Low scalability	High Scalability

1) SIGNATURE AND THRESHOLD-BASED DETECTION

This detection method is identified as misuse detection or knowledge-based detection [50]. It detects attacks or abnormal behaviours through patterns or rules and compares the observed behaviour against these regulations or patterns [51]. Rules are applied to help decide whether a given activity pattern is malicious or normal [52]. This technique is used when there is a need to detect an attack with a unique and clear signature [51].

2) ANOMALY-BASED DETECTION

This detection method is behaviour-based, statistical anomaly-based, or relies on baselining. It collects data on normal user behaviour during a specific period [53]. Statistical tests then efficiently determine whether the user behaviour is normal or indicative of an attack by employing ML or DL approaches. There are three sub-categories of this type of detection: Machine Learning-based simulation, ML-based public datasets, and DL approaches.

1) **Machine Learning-Based Simulation:** This method utilises ML approaches to identify the attacks. The

mechanism employs a simulation technique to generate a dataset for training and testing the ML classifiers. These classifiers categorise the traffic as abnormal or normal. Figure 4 illustrates the sequential stages involved in creating a simulation dataset. In these particular methods, researchers constructed a network topology that included regular hosts to generate normal traffic and bot hosts to generate abnormal traffic [54]. Scapy and Wireshark, open-source tools, were employed to simulate and generate DDoS, DoS, Probe, Portscan, U2R, and R2L attacks. Features such as protocol type, source IP address speed, source port rate, and flow packets are extracted from both normal and abnormal traffic. Following the preprocessing of these features, they are stored in a CSV file as raw data, which will train the proposed models [55]. Upon completing the model training, the model will be prepared to apply ML algorithms to classify normal and malicious packets/flows in the SDN environment accordingly.

2) **Machine Learning-based Public Datasets:** The work proposed in this method utilises the ML approach with public datasets to train and test models in the SDN network. Selecting the datasets is significant for the efficiency and accuracy of IDS. However, most publicly available datasets are not realistic and do not encompass a wide range of attacks; thus, they may negatively affect the accuracy and performance [56]. The primary reasons for these data deficiencies relate to privacy and legal issues. Furthermore, many of these datasets are outdated and lack representations of contemporary behaviours. Additionally, such datasets often contain a high number of duplicate records, which consequently may result in lower accuracy and performance [56]. The available public datasets were collected from traditional networks, not from SDN networks. As a result, these datasets include some features that are not applicable to SDNs [57]. Numerous published datasets that feature certain attacks include KDD'99, NSL-KDD, CICIDS2017, ISCX2012, Kyoto, and CSE-CIC-IDS2018 [58].

3) **Deep Learning-based approach:** DL is an approach belonging to the neural network algorithm, where the nodes can be considered devices built to defend. DL algorithms are modern updates to Artificial Neural Networks (ANN) that exploit swarming and rational computation. DL allows an algorithmic model to learn representations of data with varying levels of abstraction. These methods are applied to visual perception, object detection, network intrusion, and many domains [59]. DL algorithm can be trained as either supervised or unsupervised [60]. DL algorithms include CNN and ANN, which are typically trained in a supervised manner. CNN is currently the benchmark model for computer vision applications [61].

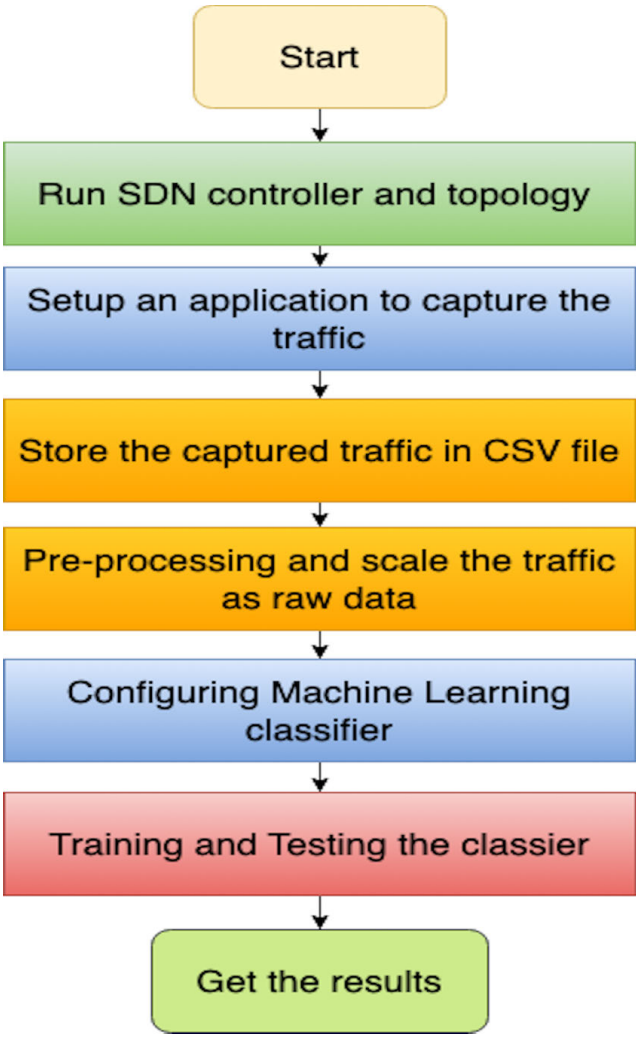

FIGURE 4. Sequential stages of creating a simulation dataset and training an ML model.

H. THE COMMON PUBLIC DATASETS USED IN IDS APPROACHES

Common public datasets utilised in IDS approaches play a crucial role in the development, training, and testing of IDS solutions. These datasets contain a variety of simulated or real network traffic patterns, including both normal operations and malicious activities, to aid in the effective training and evaluation of IDS models. These datasets include the KDD Cup 99, NSL-KDD, and CICIDS2017 datasets, each offering unique scenarios and challenges for IDS development. They provide a standardised basis for comparing the effectiveness of different IDS methodologies and enable researchers to address gaps and opportunities for innovation in the cybersecurity domain. Table 2 summarises the common public datasets used in training and testing IDS-based ML/DL approaches.

1) **KDD'99:** KDD'99 dataset is widely used for evaluating anomaly detection methods. However, its

limitations led to the development of NSL-KDD, a refined version of the KDD'99 dataset [62].

2) **CICIDS2017:** This dataset presented by the Canadian Institute for Cybersecurity, includes modern attack behaviours and offers a balanced mix of network intrusions and normal behaviours, facilitating comprehensive IDS evaluation [63].
3) **ISCX2012:** This dataset was created by the Information Security Centre of Excellence and is known for its broad coverage of normal and attack traffic [64].
4) **Kyoto University's Honeypot:** This dataset encompasses a wide variety of attacks, including less common ones, making them invaluable for the research of intrusion detection [43].
5) **CSE-CIC-IDS2018:** This dataset is a comprehensive dataset that has been presented by the Canadian Institute for CyberSecurity. Provides a current 'snapshot' of Internet traffic for IDS development and training [65].
6) **ISOT Cloud IDS:** This dataset focuses on cloud-based environments, providing diverse attack scenarios and normal traffic patterns, making it a valuable resource for testing and evaluating intrusion detection systems in cloud settings [124].
7) **HIKARI-2021:** This dataset is specifically designed for studying DDoS attacks, consisting of simulated traffic that allows researchers to develop and test DDoS detection mechanisms effectively [125].

I. EVALUATION METRICS AND TOOLS

Evaluation metrics provide quantitative measures to assess the efficiency and effectiveness of IDS, covering various aspects such as accuracy, precision, recall, and the F1 score. These metrics help gauge the ability of an IDS to accurately identify and classify normal and abnormal traffic. In addition, evaluation tools play a crucial role in collecting and analysing data, generating performance reports, and visually representing results.

Using evaluation metrics and tools allows researchers and practitioners to evaluate the performance of IDS and compare different systems effectively. This helps researchers choose the most suitable IDS for their specific needs. This section will delve into the details of various evaluation metrics and tools commonly used in the assessment of IDS. This will provide us with a deeper understanding of their significance and practical application in network security.

J. EVALUATION METRICS

IDS discern normal and abnormal traffic patterns by monitoring the system's overall input [66]. The detection algorithms are used for the classification of input traffic on the network, and these algorithms are also responsible for sounding the alarm. In the following are some of these alarms [67]:

- False Positive (FP): This alarm occurs when regular traffic is wrongly classified as attack traffic.
- False Negative (FN): This alarm occurs when attack traffic is wrongly classified as regular traffic.

TABLE 2. Summary of common public datasets used in IDS approaches.

Dataset	Year	No. of Features	Attack	Data Collection Method
KDD'99	1999	41	U2R, R2L, DoS, Probe	Simulated network traffic
NSL-KDD	2009	41	U2R, R2L, DoS, Probe	Improved KDD'99
CICIDS2017	2017	80	Brute Force, DoS, etc.	Real network traffic
ISCX2012	2012	283	HTTP, DoS, scanning, etc.	Simulated network traffic
Kyoto	2006	Varied	Wide array, including less common	Honeypots and real traffic
CSE-CIC-IDS2018	2018	78-85	Brute Force, DoS, etc.	Real network traffic
ISOT Cloud IDS	2021	Varied	Various attacks	Real cloud environment data
HIKARI-2021	2021	Varied	DDoS	Simulated DDoS traffic

- True Positive (TP): This occurs when attack traffic is classified correctly as attack traffic.
- True Negative (TN): This occurs when normal traffic is classified accurately as regular traffic.

Security metrics are generally categorised into two main groups: basic metrics and evaluation metrics. Basic metrics involve the identification of an optimal IDS or the comparison of several IDS events. Moreover, Table 3 identifies the performance parameters in the confusion matrix [68]. Equations 1, 2, and 3 are used to compute the accuracy, recall, precision, and F1 score metrics [67], as explained below:

To define the accuracy rate of an ML algorithm, the percentage of the total amount of traffic correctly predicted is calculated. Equation 1 was used for this calculation:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \times 100 \quad (1)$$

Precision (P) is used as a metric for evaluating the accuracy of correctly identifying attack traffic. This measure calculates the proportion of true positive identifications out of all positive identifications made. Equation 2 was used to determine precision.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2)$$

Recall (R) evaluates the proportion of correctly identified attacks out of the total actual attack traffic. Equation 3 is used

to calculate R, providing an estimate of the rate at which true attacks are correctly predicted in relation to the total number of actual attacks.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

The F1 score is a metric used to evaluate the classifier's accuracy, combining both Precision and Recall into a single measure. This metric balances the trade-offs between Precision and Recall, providing a comprehensive evaluation of the classifier's performance. It is particularly useful when classes are imbalanced. Equation 4 is used to calculate the F1 score.

$$F1\,\text{score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{4}$$

1) TESTING AND IMPLEMENTATION TOOLS

Effective IDS becomes paramount in SDN environments due to the increasing complexity and challenges of network attacks [69]. This section delves into the diverse array of testing and implementation tools available for IDS within SDN networks. The following are among the most frequently utilised tools in this domain:

1) **Mininet Emulator:** Mininet is a widely-used open-source network emulator that facilitates the design of virtual SDN networks. Mininet enables researchers to assess the performance of IDS under controlled conditions by simulating network topologies and traffic patterns. Provides an emulated environment for the deployment of initial IDS designs and to evaluate their effectiveness in detecting known attacks [70]. Mininet is accessible online at [71].
2) **Scapy:** Scapy is an effective packet generation tool that allows for the development of custom packets to simulate network traffic, including both regular and malicious traffic [72]. It equips cybersecurity researchers with the ability to test IDS within SDN networks by generating packets with varied payloads and headers, thus imitating real-world attack scenarios. Scapy enables the validation of IDS functionalities and the evaluation of their capability to detect, respond to, and mitigate various network attacks [73]. Scapy is available online at [74].
3) **Open vSwitch (OvS):** Open vSwitch is a prominent open-source virtual switch that can emulate components of SDN architectures. OvS enables administrators to filter packets and facilitate traffic monitoring, acting as a practical tool for implementing IDS in SDN networks. It allows researchers to design custom IDS modules that incorporate packet inspection methods, implementing real-time traffic analysis and detection strategies [75]. OvS is available online at [76].
4) **Snort:** Snort is an established open source IDS that can be integrated into SDN networks. It offers various detection policies and the ability to recognise and thwart network attacks [47]. By installing Snort alongside SDN controllers such as OpenDaylight or POX, researchers can tailor the Snort configuration to adapt to network traffic flows and improve IDS performance in dynamic SDN environments [77]. Snort is available online at [78].
5) **Bro/Zeek:** Bro, now known as Zeek, is a powerful network security monitoring framework suitable for SDN networks. It provides real-time traffic analysis capabilities, offering clear insights into network operations and enhancing the functionality of IDS [79]. With its programmable scripting language, Zeek allows researchers to detect specific malicious traffic, thereby delivering more precise and bespoke attack detection capabilities [80]. Zeek is available online at [81].
6) **Slowloris:** Slowloris, developed using the Python programming language, is an open-source tool designed to execute HTTP DoS/DDoS attacks that incapacitate targeted servers [82]. The operation of Slowloris is characterised by:
   - Generating voluminous HTTP requests, leading to a substantial surge in server traffic.
   - Dispatching packets at regular intervals, approximately every 15 seconds, to maintain persistent connections.
   - Continuation of the connection until the server itself initiates closure; should the server close the connection, a new connection is promptly established to sustain the ongoing process.

   This methodical pattern of activity exerts pressure on the server's thread pool, rendering it incapable of responding to requests from legitimate users. The Slowloris tool exploits this by inundating the server with HTTP connections, thus obstructing its functionality. Slowloris is available online at [83].
7) **Wireshark:** Wireshark is a software network protocol analyser widely used for network analysis, troubleshooting, and the acquisition of packets and flows from network devices [84]. As an open-source packet sniffer tool, Wireshark enables administrators and researchers to capture and scrutinise network traffic in situ. Its sophisticated features and wide-ranging capabilities make it an indispensable tool for scholarly and professional endeavours. Wireshark can be found online at [85].

III. LITERATURE REVIEW AND RESEARCH GAPS

This section explores existing research on IDS methods, aiming to identify gaps in this domain. This section focuses on investigating three distinct subcategories of anomaly-based detection methods: ML-based simulation, ML-based public datasets, and DL-based approaches. A critical examination of these areas provides a comprehensive understanding of current state-of-the-art methodolo-

gies, their limitations, and potential avenues for further exploration.

In the field of cybersecurity, IDS methods have gained significant prominence as organisations pursue to protect their sensitive data and networks against evolving threats. ML techniques have emerged as a promising approach to improving IDS detection abilities, by employing algorithms that learn from patterns and anomalies present in network traffic data. However, within this overarching framework, different approaches and methodologies have been employed, giving rise to various subcategories of IDS detection methods.

The first subcategory, ML-based simulation, involves the creation of artificial network environments to train and evaluate IDS algorithms [86]. The objective is to simulate realistic network scenarios, enabling researchers to develop and refine ML models for accurate intrusion detection. This subcategory encompasses a range of simulation techniques, such as Monte Carlo simulation and network traffic generation, which have demonstrated promising results in identifying and mitigating network intrusions.

The second sub-category, ML-based public datasets, revolves around the utilisation of publicly available datasets for IDS training and evaluation purposes. These datasets, often collected from real-world network environments, offer valuable insight into the characteristics and patterns of network traffic. Using these data sets, researchers enhance the robustness and generalisability of IDS algorithms, ultimately contributing to improved intrusion detection accuracy.

Lastly, the emergence of DL approaches has revolutionised the field of IDS by enabling the utilisation of complex neural networks for intrusion detection. DL techniques, such as CNN and recurring neural networks (RNN), exhibit great potential to capture intricate patterns and extract meaningful features from network traffic data. These approaches introduce new dimensions to IDS detection, which present novel opportunities for improved accuracy and real-time threat detection.

However, despite advancements within these IDS subcategories, notable research gaps persist, warranting further investigation. These gaps encompass issues related to the scalability and efficiency of IDS algorithms, the interpretability and explainability of ML models in IDS, and the incorporation of contextual information to improve detection accuracy. Additionally, the adaptation and evaluation of IDS methods in dynamic and evolving network environments pose significant challenges that need addressing In this section, a critical review of the existing literature on IDS methods within the aforementioned subcategories is conducted. Through an in-depth analysis of the strengths, weaknesses, and limitations of these approaches, it is possible to identify the gaps that require attention and propose potential avenues for future research. Addressing these gaps will contribute to advancing IDS methodologies, ultimately resulting in more robust and effective intrusion detection systems in cybersecurity.

A. MACHINE LEARNING-BASED SIMULATION

ML-Based simulation approaches employ ML techniques to identify network attacks by creating and utilising simulatively generated datasets for the purpose of training and testing ML classifiers. These classifiers are then responsible for recognising between normal and abnormal traffic patterns. This process involves several sequential stages.

In this particular method, researchers construct a network topology containing regular hosts, which are used to generate standard, non-malicious traffic, alongside 'bot' hosts that produce abnormal traffic, indicative of network attacks. To simulate and generate a variety of attack patterns, including DDoS, DoS, Probe, Portscan, U2R, and R2L, researchers utilise open-source tools such as Scapy and Wireshark [87].

Critical features, such as protocol type, source IP address speed, source port rate, and flow of packets, are extracted from both normal and abnormal traffic. Following the preprocessing stage, these features are stored in CSV files as raw data, ready for use in training the proposed models [55]. Once the training phase is completed, the model becomes adept at applying ML algorithms to accurately classify packets as either normal or malicious within the SDN environment. Below are explanations and discussions regarding some works that have employed such approaches and datasets:

The authors in [88] employed the Support Vector Machine (SVM) algorithm for identifying DDoS attacks in the SDN architecture. The methodology shows the implementation of ''Flow Status Collection''; however, there is no explanation of this system or how it works. Although the algorithms are described briefly, there is no code provided to analyse the implementation, which creates uncertainty. It is necessary to implement a feature selection method to analyse better results for this type of attack, despite it being a learning model designed for training with little data. From the results, it can be noticed that the highest achieved detection rate is 98%. However, given that this approach provides some non-real values, these false positives can become alarming. If the false positives are not 5.88%, they could be up to 7 or 8 times higher, depending on the type of infrastructure, as previously mentioned. This method faces the challenge of implementing solutions to DDoS attacks, and although it has achieved some success, it lacks various datasets to improve the training model and integrate with multiple frameworks and publicly available trials.

In another paper [20], the researchers constructed a simulation platform based on Mininet in the SDN network to identify DDoS attacks. They used an SVM classifier to classify each new incoming packet. The flow table collection is taken through characteristic value extraction to feed the classifier during the attack detection process. However, no feature selection process is performed, which can affect the classifier's performance and accuracy. Efficient packet processing is also a challenge since DDoS attacks involve high volumes of data within short periods, but this aspect

is not covered in the DDoS attack practices. Therefore, the detection results can be helpful, but the performance may be insufficient in certain test environments and may cause a bottleneck at the controller. According to the results, the best accuracy rate is achieved by testing with a 600 Transmission Control Protocol (TCP) packet size dataset, obtaining 96.83% accuracy. The TCP protocol provides better results compared to User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP), as it has more network fields to classify. UDP, on the other hand, is less but more representative based on the analysed application. Finally, ICMP has representative characteristics in the payload that allow individual behaviour during attacks. The proposed system has achieved a detection accuracy rate of 95.24% and a false alarm rate of 1.26%.

Myint Oo et al. [89] designed a detection model for the SDN network using an improved version of the SVM to identify DDoS attacks. The challenges faced by this study that have not yet been solved include performance degradation and efficient packet processing. The detection model involves packets traversing the entire SDN network, including OpenFlow switches, flow entries, and OpenDaylight, before reaching the API of the Advanced SVM (ASVM), which then classifies the packet as an attack or not. The testing process is challenged by the use of realistic datasets or environments, as it was tested with 1,000 packets, a number deemed average under normal conditions. However, DDoS attacks are anything but normal; they stress the infrastructure to its limits. Latency does increase considerably—considered to be 0.1 seconds under typical conditions in the testing scenario—but the model fails to consider obfuscated information, a technique attackers use to bypass the security controls' fingerprinting process. An accuracy of around 97% was achieved with minimal training time required; however, this does not necessarily indicate that the algorithm was optimal. The study lacks a detailed comparison of features that show improvements over a standard SVM and the ASVM, with the objective to achieve better results than those using a regular version.

In their research, Silva et al. [90] presented a management framework that combines information theory techniques with ML algorithms, aiming to categorise traffic analysis. They addressed the challenge of efficient packet processing and proposed a comprehensive view of the SDN networks. However, the use of SDN poses challenges with human intervention, being manageable at a high level but problematic at a low level. The extraction of network traffic profiles is resource-intensive, demanding efficient feature selection methods for managing high volumes of data. The study implemented K-means for clustering and SVM for the classification of anomalous traffic. Their use of a clustering algorithm to create the subsets could have impacted performance. During testing, they described the behaviour of the DDoS and port scanning attacks. The achieved accuracy of 88.7% and precision of 82.3% are considered low when compared to other studies. The reasons for the low accuracy rates were not discussed, leading one to speculate that a primary issue could be the juxtaposition of high-level instructions executed within a low-level technical environment, even when the time taken for collection and analysis was 0.075 seconds in a topology with 100 switches.

In summarising the various machine learning-based simulations for DDoS detection in SDN architectures, it is evident that SVM and enhanced versions such as ASVM have been popular choices due to their robust classification capabilities. Each study demonstrates promising detection rates, with accuracies ranging between 88.7% and 98%. However, challenges such as high false-positive rates, lack of realistic and diverse datasets, and inadequate feature selection methods persist, impacting the models' performance under real-world conditions. Key issues, including SDN switch memory limitations and controller bottlenecks during high-volume attacks, remain largely unaddressed. Efficient packet processing and feature selection are essential to enhance these models' resilience and detection accuracy, as demonstrated by the notable decline in performance when simulating realistic attack scenarios. Improvements in these areas would be instrumental in refining the ability of SDN-based systems to manage and detect large-scale DDoS attacks effectively.

B. MACHINE LEARNING-BASED PUBLIC DATASETS

In the ML-based public datasets approach, researchers employ widely available datasets to train and test models within an SDN network. The selection of these datasets is crucial to developing an efficient and accurate IDS. However, many public datasets are criticised for their lack of realism and incomplete representation of contemporary network attacks. This shortcoming can lead to diminished accuracy and performance in IDS [56]. Privacy and legal concerns are primary reasons for these data deficiencies, and many datasets are outdated, lacking modern attack behaviour and containing a high number of duplicate records [91].

The publicly available datasets are often derived from traditional network setups rather than SDN networks, which may contain features not present in SDN environments [56]. Some of the known datasets in this area are KDD'99, NSL-KDD, CICIDS2017, ISCX2012, Kyoto, and CSE-CIC-IDS2018, each presenting a mixed bag of challenges and benefits when applied to ML approaches in IDS [56]. Below are explanations and discussions regarding some works that have employed such approaches and datasets:

The authors in this paper [92] focused on the fundamental need and presented a solution called Eunoia. The proposed model is an IDS that is ML-based in the SDN network. Eunoia targets the monitoring, detection, and control of any malicious or suspicious traffic in SDN that could harm its internal operations, thereby causing network intrusion. The presented solution comprises three sub-processes: data

pre-processing (filters out irrelevant data traffic to provide valuable data), data modelling (applies the chosen algorithm to predict new audit data), and decision-making and response (enables SDN to respond to analysis results through an active learning process and reactive routing in SDN). However, it faces many challenges, such as having sufficient computational power to process the enormous amount of data entering the SDN-based network intrusion system as malicious traffic. The features extracted in the model filter the valuable data from the non-valuable data. The active learning and reactive routing of data further examine the analysis results and store the implemented outcomes. Another significant challenge may be the efficient processing of packets to avoid causing bottlenecks that could lead to system crashes or force the system into standby.

Authors in [93] improved the SVM with a behaviour-based approach to enhance the learning algorithm's ability to monitor and categorise threats. The feature extraction they described utilises an information gain approach for each variable. However, they do not clearly define the process and selection of these variables. Although the selection method is characterised as being based on the top-ranked features, one of the most significant challenges for NIDS is establishing effective feature selection. The authors have outlined the hyperparameters of the SVM algorithm without explaining the rationale for such value ranges, a problem faced by NIDS, which must ensure consistent and accurate evaluations. The implementation of the SVM is described ambiguously, without indicating whether it involves optimised frameworks or custom modifications over other implementations. The primary objective of this paper is to identify a range of attacks; however, the tests conducted focused solely on DoS attacks. Thus, while the algorithm can identify such attacks with a high accuracy of 97.63%, the proposed model faces challenges when applying to more complex scenarios and is not suitable for large-scale networks due to the need for an efficient method to process each packet. Moreover, this model requires a feature selection technique capable of distinguishing the relevant features that effectively represent the behaviours of the attacks under study.

In this research paper [94], the researchers presented a model to ensure the SDN structure is self-adaptive while responding to network events by analysing misbehaviour and new flow attacks. The analysis is made by implementing ML algorithms to classify such behaviours. The proposal has more than one strategic point where analysis has to be done; the first one is when the traffic is new, it has to be analysed if it is an attack or not, and the second one is when the traffic is not known, it is going to be analysed in-depth. However, the time-lapse of waiting for the client to request the resource is a challenge related to efficient packet processing, which is not covered in the paper. Since DDoS attacks generate large amounts of abnormal traffic, some hours under attack impact the algorithm's performance. The number of features is 41, which challenges the problem of the proper use of feature selection, and it is not covered in the paper. Therefore, it affects algorithm performance and executions for each incoming sample from the network traffic. On the other hand, 20% of the total samples were used for training but usually used 60 and 40 for testing. Still, the paper does not present any analysis of about 20%, even though the highest accuracy is 99.4% for Self-Organising Map (SOM) classification. Finally, the equation to identify misbehaviour attacks is just the computation of distances between values; therefore, there is no in-depth analysis of correlation or variance.

The researchers in [55] conducted a benchmarking analysis of existing ML approaches to recognise malicious traffic in SDN environments. The evaluation identified the limitations of each algorithm and the execution of experiments based on a public database. The analysed algorithms included SVM, NB, k-Nearest Neighbors (KNN), ASVM, Hidden Markov Model (HMM), K-means, Random Forest (RF), Decision Tables (DT), K-medoids, and fuzzy control models. They were analysed based on previous research, considering features, classes, the datasets used, and size. However, such a comparison is not balanced, since each algorithm has advantages and disadvantages in different areas. For example, the paper does not address the intended use of K-means in finding clusters as compared to SVM, which is used for classifying information. A direct comparison is not viable because one algorithm excels at separating a dataset based on patterns, while the other is adept at separating data with already defined patterns, thus presenting the problem of appropriate selection. Principal Component Analysis (PCA) is useful for reducing the number of parameters in algorithms requiring manual parameter analysis, but this is not presented alongside the challenges. The evaluation metrics for the algorithms are precision, recall, F-score, and accuracy. However, a high or low score in each metric does not convey the same advantages or disadvantages across different algorithms, and the paper does not offer an in-depth analysis of such differences. Lastly, the paper mentions that DL is the optimal approach to identifying unknown attacks, but it tested ML algorithms, not DL algorithms. The highest accuracy achieved was 81.5% for the J48 algorithm. The authors did not provide a technique to mitigate the controller's overhead when installing the IDS.

The authors in [95] introduced an inference-based IDS to detect the DoS attack in SDN. The proposed IDS is tasked with managing the separation of network structural information from the control plane. The approach is grounded in Graph Theory, which centres on the relationship of context to predict attacks. The authors utilised the CAIDA dataset and a specific dataset containing labels per connection to test the system. The IDS was evaluated using precision, recall, and F-score measurements; the results were 0.84, 0.78, and 0.81, respectively. This method can also mitigate the effects of DoS attacks in SDN environments. However, this approach requires processing a high volume of data,

which can affect the controller's performance. Moreover, the extracted features were taken from the header information, and some were not relevant or used to identify the DoS attack, such as the node type. Additionally, the selected features do not address the behaviours characteristic of a DoS attack. Hence, an attacker could easily circumvent this model when the attack commences.

The authors in [96] presented a network-based IDPS method in SDN, targeting DoS and Probe attacks. The proposed system utilised the DT methodology, employing the C4.5 algorithm and the 1999 Defense Advanced Research Projects Agency (DARPA) dataset. The authors assert that the C4.5 algorithm is capable of preventing overfitting and can handle missing attribute values in the training data. They claim that this method can mitigate the effects of DoS and Probe attacks within SDN. The model was evaluated using precision and recall metrics, where the results for the DoS attack were 0.989 and 0.964, respectively, and for the Probe attack, they were 0.984 and 0.921, respectively. However, the researchers used the 1999 DARPA dataset for training, which many researchers have since abandoned due to its lack of features associated with newer types of DoS attacks. Attackers continually develop new behaviours for DoS and Probe attacks. Furthermore, the integration of the SDN controller with their IDPS required monitoring a high volume of control packets, potentially leading to an overload on the controller. This presents another challenge that needed addressing in their research.

In the study by Braga et al. [97], the researchers developed an IDS to pinpoint DDoS attacks within SDN environments, employing a NOX controller and a Flow Collector for traffic data acquisition. The classification of traffic as either normal or malicious was performed using the SOM algorithm, with the system's efficacy being assessed via detection rate and false alarm rate measurements, yielding results of 98.61% and 0.59%, respectively. However, it's observed that a limited dataset was used for training, which raises concerns about the accuracy of detection in real-world scenarios. An additional observation is that the feature extraction process was restricted to packet header data, potentially omitting critical indicators of DDoS attack patterns. As a result, the system might be vulnerable to evasion techniques that involve disguising malicious packets as legitimate traffic. Furthermore, the study only addresses DDoS attacks, suggesting a need for expansion to include other attack vectors such as Probe and U2R. Another notable observation is the absence of any proposed countermeasures for the attacks once they are detected.

The authors in [98] proposed an IDS on SDN using the SVM algorithm with a kernel function to classify the traffic into normal and abnormal. Kernel functions are commonly used to support linear classifiers by taking the dataset and transforming it into a higher dimension. The proposed system detects Internet Protocol (IP) sweep, Probe, and DDoS attacks in the control plane. To evaluate the system, they used datasets from the 1998 and 2000 DARPA to train and test the model, where each set includes different types of attacks. The system achieved a 94.81% accuracy rate and a 0.11% false alarm rate. However, the number of extracted features is inadequate to understand the behaviours of attacks fully, and some features are not associated with the patterns of attacks. Also, the SVM classifier requires more time during the training phase. Additionally, the controller will examine all packets passing through to classify them, a process that can generate significant overload on the controller, leading to flooding and congestion.

The researchers in [99] implemented five IDS models in the SDN network using several ML algorithms: SOM and Learning Vector Quantization (LVQ1), as well as their modified editions. The ML algorithms used in this work are as follows: SOM, Multi-pass Self-Organising Maps (M-SOM), LVQ1, Multi-pass Learning Vector Quantization (M-LVQ1), and Hierarchical Learning Vector Quantization (H-LVQ1). These approaches are considered types of ANN. The proposed models detect multi-level attacks such as Probe, U2R, R2L, and DoS by classifying the traffic. All implemented models achieved an average True Positive Rate of 94%. However, the authors generated a dataset containing features that are very basic and easily extracted from the packet header. Consequently, the implemented models may not detect actual network attacks and do not cover the behaviours of the attacks. The authors did not consider managing the vast packet processing flows in large-scale networks. Additionally, the integration of the SDN controller with the IDS incurs overhead on the controller, leading to potential controller flooding. This is another challenge that needs addressing.

This paper [100] introduced a detection method based on anomalies. This method operates by integrating with OpenFlow switches. The proposed model aims to prevent and detect both known and unknown attacks in SDN. The J48-tree algorithm, which is a version of the C4.5 decision tree developed for classification purposes, has been utilised. The proposed model has been implemented using the NetFPGA-10G board. The system achieved a 91.81% detection rate and a 0.55% false alarm rate, employing the KDD'99 public dataset for the training and testing stages. However, the researchers did not consider the high volume of data in large networks, data that require time and energy to be processed efficiently. This oversight may result in an overload on the controller and the switches, as the OpenFlow switches need to check each incoming packet and subsequently forward it to the controller for appropriate action, which can lead to flooding. Additionally, the proposed system extracts an excessive number of features during the packet processing in the investigation stage, leading to the consumption of the network's resources.

The authors in [101] introduced a method to address the dynamic nature of SDN and to detect DDoS attacks in the application plane by classifying incoming traffic through ML

algorithms. The employed ML algorithms include NB, KNN, K-means, and K-medoids. The experiment utilised a private dataset derived from a trace file that belongs to a real network to train and test the models. The Detection Rate metric was used to measure the accuracy of the implemented algorithms, with the results being 94%, 90%, 86%, and 88%, respectively. However, 50 features were used to train and test the models, and the controller is required to extract these features for each new flow of incoming traffic. This high volume of features demands more memory and a lengthy processing time. Consequently, when an attack begins, the controller is likely to suffer from an excessive overhead shortly thereafter.

The paper by Abubakar & Pranggono [102] introduced an approach to identify attacks using ML methods. The system is a flow-based IDS developed in response to the limitations of signature-based IDS. A neural network algorithm has been applied to classify each packet by the controller. The NSL-KDD public dataset was employed to implement and train the proposed IDS. The model's aim is to detect DoS, U2R, R2L, and Probe attacks. The proposed model has achieved a Detection Rate of 97.4%. However, the use of a small database size during the training stage negatively affects the detection accuracy in real tests. The features extracted have been obtained only from the packet headers, failing to encapsulate the behaviour of the attacks adequately. Additionally, the dataset utilised contains redundant records. The processing operation conducted by the controller for each packet can create a bottleneck for the controller and a single point of failure, which represents a significant challenge in implementing ML approaches within IDS. Furthermore, the proposed model requires a feature selection method to identify the most effective features when an attack occurs.

Lataha and Toker in [103] aimed to demonstrate that SDN can contribute to a solution for DoS attacks. The proposed model is divided into two phases of intrusion detection: the first one is flow-based, and the second is packet-based. A significant disadvantage arises with the high resource consumption when filtering and analysing packets in two states, similar to both stateful and stateless firewalls. These issues have resulted in performance degradation during periods of high incoming data rates. The proposed intrusion detection system differentiates malicious from legitimate flows using the KNN approach. However, in the SDN environment, high-level management limits the number of features available for manipulation. Meanwhile, the detection of malicious packets classifies both legitimate and malicious traffic using neural networks. Although such an algorithm is adept at separating two classes as proposed, since the malicious traffic was already classified based on the flow, the packet detection should incorporate properties not considered by the preceding algorithm. Consequently, the proposed approach has an accuracy of 91.27% and precision of 0.99%, relative to algorithms such as KNN utilising the NSL-KDD dataset, neural networks, and others in the same environment. Despite an improvement in the false positive rate, the processing time has not been enhanced, failing to address the issues of packet processing and the controller's potential bottleneck.

The authors in [104] proposed an IDS in SDN. The model is based on AI with two stages of processing. The proposed system utilised the Random Forest algorithm to classify traffic, and a feature selection stage that used a Bat algorithm with swarm division and binary differential mutation. The system identifies DoS, Probe, DDoS, U2R, and R2L attacks. The authors employed the KDD Cup 1999 dataset to evaluate the system and to train and test the model. The system achieved a 96.3% accuracy rate. However, the proposed classifier requires more time in the training stage. Additionally, the controller examines all the packets passing through to classify them, which may cause an overhead on the controller, leading to a bottleneck. Furthermore, the limited size of the raw dataset used to train the model negatively impacts the algorithm, which requires more data to achieve higher detection accuracy.

The researchers in [105] presented an IDS utilising an AI algorithm. Implemented in the context of SDN, the IDS detects DDoS attacks in Home and Small Office/Home Office (SOHO) networks. This approach employed the Threshold Random Walk with Credit-Based rate limiting (TRW-CB) and Rate Limiting to classify real-time traffic. The authors collected the dataset using the Mergepcap tool from three locations: Home Network, SOHO, and Internet Service Provider (ISP), which was then used to train the model. The proposed model focused on the essential features for classification, primarily extracted from the packet header, which will be obtained for each packet at the SDN controller. The NOX controller was used with this model. The experiment achieved a detection rate accuracy of 90% and a 70% false positive rate. However, the limited number of extracted features may lead to low detection efficiency. These features, having been extracted from the packet's basic header information, are not sufficient to cover attack behaviours. Thus, an attacker could easily bypass the IDS by altering the packet header to resemble regular traffic. Additionally, the authors did not consider the controller's potential bottleneck, which could hinder the controller's efficiency in a larger network. Consequently, the system requires a method to process packets efficiently that is also lightweight.

The research by Jiaqi et al. [106] proposed an ML approach in the SDN 5G environment to identify DDoS, DoS, U2R, and R2L attacks in the SDN controller. The K-means++ and AdaBoost algorithms have been used to classify the traffic, while the RF algorithm has been applied in feature selection. The authors employed the KDD Cup 1999 dataset, which is widely used in IDS, to evaluate the proposed system. The model has achieved an average classification accuracy of 84%. However, the RF algorithm has failed to select related features that cover the behaviours of U2R and R2L attacks, as it has achieved low accuracy in detecting such attacks. These features have been extracted from the

packet's basic header information and are insufficient to cover the attacks' behaviours. The authors did not consider the controller's potential bottleneck, which could result in inefficiency at the large-scale network level. Consequently, the system requires an efficient, lightweight method to process packets.

Sathya and Thangarajan in [107] focused on security violations in the SDN environment and how to identify and prevent attacks using anomaly-based detection methods. The authors adopted the use of an IDS to recognise DoS, Probe, U2R, and R2L attacks. The proposed model utilised the NSL-KDD dataset, which includes four types of attack packets: DoS, Probe, U2R, and R2L. The Feature Selection stage selected 27 features for DoS attacks, 26 for U2R attacks, 33 for Probe attacks, and 33 for R2L attacks. The system achieved detection rates of 90.9%, 91.1%, 80.2%, and 98.1% and false alarm rates of 0.111%, 0.249%, 0.69%, and 0.887% for DoS, Probe, R2L, and U2R attacks, respectively. However, this system did not achieve the highest accuracy or the minimum false alarms compared to other approaches. The system struggled to minimise the features selected by the Binary Bat algorithm, where too many features used could have been reduced and were not relevant to the detected attacks. The large number of features extracted for each traffic request results in excessive processing time and memory consumption at the controller.

The authors in [108] proposed an effective IDS in SDN environments to identify DDoS attacks using a Sequential Probability Ratio Test (SPRT). The proposed IDS has been tested with datasets from DARPA for intrusion detection and compared the proposed technique against others. Classification is a statistical approach that uses the Bernoulli random variable to preprocess the training section. However, this type of algorithm requires datasets with the same features as the environment to analyse, facing the problem of using accurate data. Since the datasets are from DARPA, they are not sufficient due to rapid technological changes, leading to the datasets' obsolescence. The attacks used to test the SDN environment include DDoS, Neptune, Smurf, IPsweep, and Port sweep. However, not all of these are targeted at SDN. The values used for the SPRT parameters were set manually, but most of these algorithms must be tested under a combination and variation of parameters. This could result in good performance and overfitting when choosing a suitable threshold for the algorithm. The most significant problem in this paper is identifying the DDoS attacks in the SDN controller using an acceptable threshold. Yet, the proposal is ineffective when detecting DDoS attacks against a host, outcomes that produce false positives since the rules differ for these two types of targets. This shows that the proposed method is ineffective for expected flows over distributed environments as was concluded. Moreover, this approach faces the issue of the controller's overhead in large networks, which requires more CPU and memory consumption to extract the features and classify each piece of traffic.

The authors in [2] presented a novel system called HFS-LGBM IDS for detecting attacks in SDN. The HFS model merges the benefits of two techniques: Correlation-Based Feature Selection and RF Recursive Feature Elimination. Mininet has been used to evaluate and test their proposal. The NSL-KDD dataset was employed to test and train the Light Gradient Boosting Machine (LightGBM), a Gradient-Boosting Framework classifier based on Decision Trees. The accuracy obtained with this model was 98.72%. However, the NSL-KDD dataset has been thoroughly analysed and deemed outdated, thus failing to represent real-world network traffic accurately. Consequently, this impacts the detection accuracy in actual tests. Furthermore, only eight features were considered, which are insufficient to cover the behaviour of attacks, leaving the system unable to predict the flows accurately. In addition, integrating the SDN controller with their IDS required a high volume of flows to check traffic, which would generate a considerable overload on the controller—another challenge that needs to be addressed in their work.

The researchers in [109] presented the OpenFlowSIA security system for SDN. The system incorporates an SVM classifier and Idle-timeout Adjustment (IA) algorithms to protect the controller and OpenFlow switches from DDoS attacks. The proposed IDS includes five modules: Flow Collector, Feature Extractor, SVM, Policy Enforcement, and IA Algorithm. The Flow Collector gathers traffic from the flow tables of all the OpenFlow switches, preparing it for the next phase by the Feature Extractor module. This collected traffic is then processed to extract features, which will be input for the SVM classifier. The SVM classifies the traffic by protocol type TCP, UDP, ICMP, or ARP. Finally, the Policy Enforcement and IA algorithm classify the packet as normal or malignant. The authors used CAIDA datasets to train and test the system. The evaluation revealed that the proposed system consumes about 50% of the CPU usage of the controller, indicating that the method is inefficient as it uses excessive memory and CPU; this system can also cause congestion at the controller, affecting response time. The proposed model is not scalable, as it is tailored to identify DDoS attacks and consumes about 50% of the CPU capacity. The proposed system lacks a feature selection method to identify suitable features that cover the behaviours of DDoS attacks. Additionally, the authors did not employ IDS evaluation metrics to determine the detection or accuracy rate when evaluating the model.

The reviewed literature on ML-based IDSs in SDN emphasises the potential of leveraging public datasets to enhance network security. However, several challenges persist, including the outdated and unrealistic nature of commonly used datasets like KDD'99 and DARPA, which limits their applicability to contemporary attack scenarios. Additionally, computational demands associated with processing large volumes of traffic can lead to performance bottlenecks in IDS implementations, particularly during high-traffic events such as DDoS attacks. Effective feature selection

remains a critical issue, with many studies lacking clear methodologies to identify relevant features that accurately represent attack behaviors. Furthermore, the limited scope of evaluation in existing research often neglects diverse attack vectors, reducing the robustness of proposed solutions. These challenges underscore the need for further exploration and innovation to enhance the reliability and effectiveness of ML-based IDS in dynamic SDN environments.

C. DEEP LEARNING-BASED APPROACHES

DL is an advanced approach rooted in ANN algorithms, where nodes emulate defensive devices in a network [110]. It represents an evolutionary leap by leveraging intricate structures and high-level data abstraction. With the capacity for swarming and intelligent computation, DL empowers algorithms to learn representations of data across various layers of abstraction [111]. These powerful methods are employed for tasks such as visual perception, object recognition, and network intrusion detection across diverse domains. DL models can be trained using both supervised and unsupervised learning techniques. Notable DL algorithms include CNN and ANN, which are predominantly trained in a supervised manner [112]. Specifically, CNNs have become the gold standard in computer vision tasks, shaping the cutting edge of visual computational models [61]. Below are explanations and discussions regarding some IDSs that have employed DL approaches for attack detection:

B. Sarra and G. Mohamed in [113] proposed a DL approach in the SDN context to identify DDoS and DoS attacks that occur between the controller and end-user devices. Utilising the Rectified Linear Unit (ReLU) and Softmax functions, the traffic is classified as malignant or normal within the SDN controller. The public dataset CICIDS2017 was utilised in the experiments during the training and testing stages. The authors employed the logarithm function, which uses the Min/Max scalar technique to normalise the extracted features for the classification step. The proposed model utilised five basic features for classification, which are extracted for each packet at the SDN controller in real-time. The model has achieved an accuracy rate of 99.6%. However, the limited number of extracted features could lead to low efficacy in detection. These features, extracted from the packet's basic header information, are not sufficient to cover the behaviour of the attacks; thus, an attacker could easily evade the IDS by modifying the packet header to resemble regular traffic. Additionally, the authors did not address the controller's bottleneck issues, where the controller in an extensive network may not be able to perform its functions efficiently. Therefore, the system requires a lightweight method to process packets efficiently.

Niyaz et al. in [114] presented a DL-based system to recognise DDoS attacks in the SDN environment, focussing on multi-vector attack detection. The system examines every packet at the SDN controller, extracts the features, and then classifies the extracted features as either normal or malignant. The proposed model employs the POX controller. The authors collected the dataset from the home wireless network (HWN) using the tcpdump tool, and the hping3 tool was used to generate the DDoS traffic. After collecting the traffic, the authors divided it into training and testing datasets. At the Feature Extractor stage, the proposed system extracts 68 features for use in the classification of each packet. The models have achieved an accuracy of 95.65%. However, the numerous features extracted require high-level memory and extended processing time, which may lead to a controller's bottleneck. Moreover, most of these extracted features are not relevant to DDoS attack patterns.

The authors in [115] employed a DNN approach to recognise DDoS attacks in SDN networks. The NSL-KDD public dataset was used in the training and testing stages. The proposed model used six basic features for classification, which were extracted for each flow in the controller in real-time. The model achieved an accuracy of 75.75%. However, the small number of extracted features resulted in a low detection accuracy in the detection stage. These features, derived from the basic statistical information of the flow, are insufficient to cover the attacks' behaviours; thus, attackers could easily evade the IDS. After a fixed time-window, the authors suggested that the controller requests flow table entries from all OpenFlow switches, meaning that all flows in the switches would be classified each time. Moreover, the controller processing the requested flows would send other flow statistics requests before finishing the previous request. Therefore, the model needs a technique to confirm the completion of the ongoing process before sending another statistics request, as this would create significant overhead on the controller. Given the nature of deep learning approaches with their cost and computational complexity, this adds extra processing time, underscoring the need for the system to efficiently process flows and remain lightweight.

The research in [116] presented a DL model for SDN to classify traffic as either a DDoS attack or not. This approach utilised the GRU-RNN to classify traffic in real-time. The NSL-KDD public dataset was used in the training and testing stages. The proposed model used six basic features for classification, which were extracted for each flow at the SDN controller in real-time. The POX controller was used with this model. The model achieved a detection rate of 89%. However, the limited number of extracted features led to low detection efficiency. These features were extracted from the packet's basic header information and are insufficient to cover DDoS attack behaviours; therefore, an attacker could easily evade the IDS. Additionally, the authors did not address the controller's bottleneck, which causes issues in a large network where the controller may not perform functionalities efficiently. Consequently, the system requires a lightweight method to process traffic efficiently.

The authors in [117], proposed a hybrid system termed called 'SD-Reg and CNN' for SDN attack detection. The model utilises standard deviation, referred to as SD-Reg,

to resolve the overfitting issue in the CNN predictor. The InSDN dataset was deployed for testing and training the CNN classifier. This model achieved accuracies of 99.28% for binary classification and 98.92% for multi-class classification. Nonetheless, relying solely on the InSDN dataset to train the model proves inadequate as it does not cover the majority of high-risk attacks, thus affecting the validity of the test results. Additionally, the approach presented lacks a solution to the overhead created on the controller, which stems from the CPU consumption that occurs when merging CNN and SD-Reg.

In the OpenFlow controller, Dey & Rahman in [118] proposed a flow-based anomaly detection approach using DNN algorithms. The proposed model is termed Gated Recurrent Unit Long Short-Term Memory (GRU-LSTM DNN). A feature selection technique (ANOVA F-test) has been employed to achieve high-performance classification. The NSL-KDD public dataset was used in the training and testing stages of the experiment. By classifying each packet, the proposed models detect multi-level attacks such as Probe, U2R, R2L, and DoS. The proposed model utilised 52 features for classification, which were extracted for each flow at the SDN controller in real-time using the ANOVA F-test method. The model has achieved an accuracy of 87% with a false alarm rate of 0.76%. However, the controller's bottleneck was not addressed.

Y. Hande and A. Muddana in [119] addressed issues regarding the development of anomaly-based NIDS in SDN networks. The model utilised a DL approach to improve performance and accuracy, aligning with previous research using a CNN model to identify various types of attacks within the SDN network traffic. The sniffer IDS module, which feeds the detector, is positioned efficiently for packet processing at strategic points through the use of SDN. This raises an issue, as SDN eliminates physical network devices, and consequently, security no longer resides in such devices, allowing for the potential manipulation of network traffic through the ANIDS. Moreover, the paper lacks a thorough analysis of the features—how and why they were extracted and selected. It offers scant details about the design of critical components of the system, particularly the detector. Not explained is the rationale for having precisely two CNN layers, nor is it clarified why these layers cover the same type of information. The question arises as to why features were manually selected instead of utilising an unsupervised CNN approach, which is one of the key advantages of employing CNN and remains a significant challenge for DL-based systems. Finally, the sensing module's description omits the kinds of classes they have modelled for an attack and how it detects not well-known attacks.

DL, an advanced form of ANN, utilises intricate structures and high-level data abstraction to enhance network security through intelligent computation and feature extraction. Recent studies, including those by Sarra and Mohamed and Niyaz et al., highlight the efficacy of DL models in detecting DDoS attacks within SDN. However, these models often employ a limited number of features, which can hinder their detection capabilities and allow attackers to evade detection by manipulating packet headers. Moreover, issues such as high computational demands and controller bottlenecks emerge due to the extensive processing requirements of these algorithms. For instance, while some models achieve high accuracy rates (up to 99.6%), their reliance on basic packet features may result in insufficient coverage of attack behaviors. Furthermore, hybrid models, like the one proposed by Dey & Rahman, emphasise the need for effective feature selection techniques to mitigate false alarms while addressing the controller's limitations. Overall, while DL approaches show promise in enhancing intrusion detection within SDNs, they face significant challenges, including the need for lightweight processing methods and a comprehensive feature extraction strategy to accurately reflect the dynamic nature of network threats.

Table 3 and 4 summarises the related works that used ML and DL approaches for threat detection.

D. THE FINDINGS AND RESEARCH GAPS

Due to the flexibility and innovation of SDN in networking environments, there are uncovered gaps that can be integrated with future technologies to offer more exciting research. This review study is the first to make an effort to solve the problem of attack impact on different SDN layers. In the future, it will still be necessary to analyse and measure how attack-defined techniques affect the security components and requirements of SDN research challenges. The following points have been identified as emerging directions and unaddressed gaps related to the security of the SDN architecture.

1) **An Effective Way to Process Network Traffic**
   Most researchers propose models that require real-time traffic monitoring, necessitating the analysis of each packet or flow to determine if it is normal or malicious. Consequently, this process demands time, memory, and increased CPU usage to gather, process, and classify the traffic. As a result, it can lead to overload and congestion in both the controller and the switches. Hence, researchers should carefully consider striking the optimal balance between efficiency and avoiding overload.
2) **Distribute the Processing Stages Over OpenFlow Devices**
   The majority of researchers utilise the 'of_stats_ request' to obtain the flow tables of all switches to extract the selected features [120]. However, this method retrieves the flow tables of switches without sorting them based on time, size window, etc. As a result, this data needs to be processed by the controller to sort and extract the selected features. Some of these features are not taken directly from the flow tables, and complex computational operations are necessary for their calculation [55]. Consequently,

the OpenFlow channel can become busy and congested when installing an IDS. Additionally, due to 'ofp_flow_stats' returning all the switches' flow tables, the message size will significantly impact the channel and increase the processing time. Therefore, this method is inefficient for handling massive data processing, causing more overhead and congestion on the controller. Hence, researchers need to consider the use of efficient approaches in distributing the processing stages.

3) **Detect Slow DDoS/DoS Attack**
In this survey, it was observed that the majority of researchers focus on high-rate DDoS/DoS attacks, which involve sending large, irregular packets to the target [121]. However, there should be a greater emphasis on detecting slow DDoS and DoS attacks that specifically target web services, as they mimic regular traffic. These attacks gradually deplete resources and disrupt the target service over time. Hence, it is necessary for researchers to address this issue.
4) **Employing Outdated Datasets for Training and Testing the Proposed Models**
It is evident from the survey that many researchers have used outdated datasets to test and evaluate their proposed techniques [122]. Selecting appropriate datasets is crucial for an efficient and accurate detection system. Most publicly available datasets need to be realistic and encompass a wide range of attacks, as their absence adversely impacts accuracy and performance [56]. The main reasons for the lack of suitable data are privacy and legal concerns. Many of these datasets are outdated and require inclusion of updated behaviours. Moreover, such datasets often contain a large number of duplicate records. Consequently, systems that rely on such datasets tend to exhibit low accuracy and poor performance [123]. The publicly available datasets were primarily collected from traditional networks, not specifically from SDN networks. As a result, these datasets may contain certain features that are not applicable to SDN networks [23]. Some of the commonly mentioned datasets include KDD'99 NSL-KDD, CICIDS2017, ISCX2012, and Kyoto.
5) **Getting High Accuracy with Limited Raw Features by Using DL or ML Approaches**
The primary issues with ML/DL approaches lie in feature extraction and selection. Current systems often face challenges due to excessive utilisation of features in classification or reliance on basic features. The inclusion of a large number of features can lead to network overload and high latency. Conversely, using a limited set of basic features fails to accurately detect attacks as they do not encompass the full range of attack behaviours. Therefore, achieving high accuracy in attack detection through DL/ML approaches requires increased attention from researchers in leveraging a limited set of raw features.
6) **Test the Proposed Models with the Real Network Environments**
Most recent research reviewed in this section has made use of simulation, emulation, and virtualised environments for testing. However, these experiments typically utilise a simplistic and smaller network structure. Consequently, the results may not be fully validated for the methodologies as they have not been applied to real data. On the other hand, the use of actual hardware can yield more reliable outcomes. Therefore, testing and validating security applications based on SDN in real-world environments represents another direction for future research.
7) **Adaptability of IDS to Evolving Threats:**
The landscape of network security threats is continually evolving, with new types of attacks emerging regularly. Many existing IDS approaches in SDN rely heavily on predefined threat models and signatures, rendering them ineffective against novel or sophisticated cyber-attacks. This gap highlights the urgent requirement for more adaptable and learning-oriented IDS frameworks that can evolve in tandem with the threat landscape.
8) **The Scalability of Using More Than One Controller**
Researchers in the cybersecurity area have attempted to propose solutions using individual controllers within a network topology. Nonetheless, SDN's centralised nature is vulnerable to several attacks. Using a single controller can create a bottleneck both during an attack and in the process of handling incoming packets. Therefore, the design of distributed controllers significantly enhances reliability, load distribution, and processing power efficiency. However, the integration of IDS with multiple controllers is still under investigation and requires further exploration by researchers. The emerging research direction involves designing new IDS systems that are compatible with the distributed approach of SDN controllers to reduce overhead and balance traffic across multiple controllers. Nevertheless, when employing multiple controllers, networks can suffer from interoperability issues, as various controllers may operate with different routing algorithms and policies. This challenge has yet to be addressed, with no substantial research currently considering such issues.
9) **Security Policy Management and Automation:** The management of security policies within SDN is predominantly manual, making it prone to errors and inefficiencies. Automated security policy management systems that leverage AI and ML for decision-making could significantly enhance the effectiveness and efficiency of security measures in SDN. Research in this area is nascent, and there exists a substantial opportunity for innovation and development.

TABLE 3. IDSs machine learning based.

Author & Year	Approach	Features No.	Detected Attack	Results
(Li, et al., 2018)	SVM	N/A	DDoS	Detection Rate of 98%, False Positive Rate of up to 8%
(Jin, et al., 2018)	SVM	6	DDoS	Detection Rate of 95.24% and False Alarm of 1.26%
(Myint Oo, et al., 2019)	ASVM	5	DDoS	Accuracy 97%
(Silva, et al., 2016)	K-means and SVM	7	DDoS and port scanning	Accuracy of 88.7% and Precision of 82.3%
(Alshamrani, et al., 2017)	SOM	41	DDoS	Accuracy of 99.4%
(Wang, et al., 2016)	SVM	29	DoS	Accuracy of 97.63%
(Elsayed, et al., 2019)	Analysed algorithms including SVM, Naïve Bayes, KNN, ASVM, HMM, K-means, Random forest, Decision tables, K-medoids, and fuzzy	5	DDoS	Highest accuracy is 81.5% for the J48 algorithm
(AlEroud & Alsmadi, 2017)	Graph Theory	6	DoS	Precision of 0.84, Recall of 0.78, and F-score of 0.81
(AnLe, Phuong Dinh, Hoa Le, and Ngoc Cuong Tran, 2015)	C4.5	none	DoS and Probe	Precision: 0.989 (DoS), 0.984 (Probe), Recall: 0.964 (DoS), 0.921 (Probe)
(A, et al., 2015)	C4.5	52	DoS and Probe attacks	Precision of 0.989 and Recall of 0.964
(Braga, et al., 2010)	SOM	4	DDoS	Detection Rate of 98.61% and False Alarm of 0.59%
(Jiaqi Li, et al., 2018)	Random Forest	none	DoS, Probe, DDoS, U2R, R2L	Accuracy: 96.3%
(RT, et al., 2014)	SVM algorithm with a kernel function	8	IPsweep, Probe, and DDoS	Accuracy Rate of 94.81% and False Alarm of 0.11%
(Van, et al., 2016)	J48-tree	41	Known and unknown attacks	Detection Rate of 91.81% and False Alarm of 0.55%
(Barki, et al., 2016)	NB, KNN, K-means, and K-mediods	52	DDoS	Detection Rate of 94%, 90%, 86%, and 88%
(Abubakar & Pranggono, 2017)	NN	6	DoS, U2R, R2L, and Probes	Detection Rate of 97.4%
(Majd & Toker, 2020)	KNN	23	DoS	Accuracy Rate of 91.27% and Precision of 0.99%
(Jiaqi, et al., 2018)	RF	5	DoS	Detection Rate of 90% and False Alarm of 70%
(Akbar, et al., 2011)	TRW-CB and Rate Limiting	4	DoS, Probe, DDoS, U2R, and R2L	Accuracy Rate of 96.3%

TABLE 3. *(Continued.)* IDSs machine learning based.

Author & Year	Approach	Features No.	Attacks	Results
(Jiaqi, et al., 2017)	K-means++, AdaBoost, and Random Forest (RF)	6	DDoS, DoS, U2R, and R2L	Accuracy Rate of 84%
(Sathya & Thangarajan, 2015)	J48 algorithm	41	DoS, Probe, U2R, and R2L	Detection Rate of 90.9% for DoS
(Dong, et al., 2016)	Bernoulli random	2	DDoS	None
(Phan, et al., 2016)	IA	6	DDoS	None
(Logeswari, et al., 2023)	LGBM	8	DDoS	Accuracy of 98.72%

TABLE 4. IDSs based on the DL approaches.

Authors & Year	Approach	Features	Attack	dataset	Limitations	Results
(BOUKRIA & GUER-ROUMI, 2019)	Relu Softmax function	5	DDoS and DoS	CICIDS2017	Must reduce the bottle-neck controller and low number of used features	99.6%
(Tang, et al., 2016)	DNN	6	DoS	KDD	The accuracy must be increased. Must reduce the bottleneck controller	75.75%
(Niyaz, et al., 2016)	SAE	Unknown	DDoS	Traffic Dataset	Must reduce controller bottleneck	95.65%
(Tang, et al., 2018)	Recurrent Neural Network	6	DoS	NSL-KDD	Model optimisation is required in features se-lection and extraction	89%
(Dey & Rah-man, 2018)	GRU-LSTM	52	Prop, U2R, R2L, and DoS	NSL-KDD	Model optimisation re-quired	87%
(ElSayed, et al., 2021)	CNN	48 and 9	Prop, U2R, R2L, DoS, etc.	InSDN	Model optimisation re-quired	98.92%
(Hande & Muddana, 2019)	CNN	Not specified	Anomaly De-tection	SDN	Lacks thorough feature analysis and rationale for design choices; po-tential for traffic manip-ulation due to the ab-sence of physical de-vices.	N/A

IV. CONCLUSION AND FUTURE DIRECTION

This study has delved into the evolving landscape of SDN, with a particular focus on its security vulnerabilities and the spectrum of network attacks that exploit these weaknesses. Through a comprehensive literature review, we have highlighted the current state of IDS in SDN, assessing both signature-based and anomaly-based approaches, and explored the promising potential of ML and DL techniques in enhancing network security. Despite significant advancements, our analysis identifies critical research gaps that necessitate further exploration to fortify SDN against emerging threats.

The centralisation of control in SDN, while streamlining network management, introduces a single point of failure, making it imperative to develop robust security protocols that safeguard the controller. Furthermore, the open nature of SDN architectures exposes them to sophisticated attack vectors, such as DDoS and port scan attacks, underscoring the need for advanced detection and mitigation strategies. Our review of ML and DL applications in IDS reveals an evolving

frontier in network security, offering nuanced insights into traffic patterns and anomalies indicative of cyber threats. However, challenges in real-time threat detection, scalability, and the handling of encrypted traffic persist, marking pivotal areas for future research.

A. FUTURE DIRECTIONS

The landscape of SDN security presents a fertile ground for future investigations, particularly in the development of adaptive, scalable IDS frameworks that can seamlessly integrate with SDN architectures without impeding network performance. Emphasis should be placed on the following areas:

1) **Enhanced Real-Time Detection:** Future research should aim at leveraging ML and DL models that excel in real-time analysis, offering immediate detection and mitigation of network threats without significant latency.
2) **Scalability and Performance:** Investigating IDS solutions that maintain high levels of accuracy and efficiency, even as network scale and complexity increase, remains a critical need. This includes minimizing the impact on the SDN controller's performance and ensuring seamless network operation.
3) **Encrypted Traffic Analysis:** With the increasing prevalence of encrypted traffic, developing techniques that enable effective threat detection without decryption is paramount. This includes exploring privacy-preserving ML and DL approaches that can analyse encrypted data streams for anomalies.
4) **Cross-Layer Security Approaches:** Given the multi-layered architecture of SDN, exploring cross-layer security mechanisms that provide comprehensive protection across the control, data, and application layers offers a promising research avenue.
5) **Autonomous Security Mechanisms:** Advancing towards fully autonomous IDS that can dynamically adapt to new threats and network configurations without human intervention could significantly enhance SDN security.

In conclusion, while SDN offers transformative potential for network management and operation, securing this dynamic architecture against cyber threats remains a paramount challenge. The continuous evolution of network attack vectors necessitates ongoing research and innovation in network security strategies, with a keen focus on leveraging emerging technologies to develop resilient, intelligent security solutions for SDN environments.

ACKNOWLEDGMENT

The author would like to express his sincere gratitude to Al-Mustaqbal University for their unwavering support and sponsorship throughout his Ph.D. journey. Their commitment to fostering academic research has been instrumental in enabling him to pursue his studies and achieve his goals.

REFERENCES

[1] Y. Maleh, Y. Qasmaoui, K. El Gholami, Y. Sadqi, and S. Mounir, "A comprehensive survey on SDN security: Threats, mitigations, and future directions," *J. Reliable Intell. Environments*, vol. 9, no. 2, pp. 201–239, Jun. 2023.

[2] G. Logeswari, S. Bose, and T. Anitha, "An intrusion detection system for SDN using machine learning," *Intell. Autom. Soft Comput.*, vol. 35, no. 1, pp. 867–880, 2023.

[3] R. Palanikumar and K. Ramasamy, "Software defined network based self-diagnosing faulty node detection scheme for surveillance applications," *Comput. Commun.*, vol. 152, pp. 333–337, Feb. 2020.

[4] A. Abdulghaffar, A. Mahmoud, M. Abu-Amara, and T. Sheltami, "Modeling and evaluation of software defined networking based 5G core network architecture," *IEEE Access*, vol. 9, pp. 10179–10198, 2021.

[5] M. V. O. Assis, L. F. Carvalho, J. Lloret, and M. L. Proença, "A GRU deep learning system against attacks in software defined networks," *J. Netw. Comput. Appl.*, vol. 177, Mar. 2021, Art. no. 102942.

[6] A. Shaghaghi, M. A. Kaafar, R. Buyya, and S. Jha, "Software-defined network (SDN) data plane security: Issues, solutions, and future directions," in *Handbook of Computer Networks and Cyber Security: Principles and Paradigms*. Cham, Switzerland: Springer, 2020, pp. 341–387, doi: 10.1007/978-3-030-22277-2_14.

[7] C. M. G. de Toledo, "Secure IT-SDN: A secure implementation of software defined wireless sensor network," Ph.D. thesis, Escola Politécnica, Universidade de São Paulo, São Paulo, Brazil, 2020, doi: 10.11606/D.3.2020.tde-11012022-120120.

[8] A. Liatifis, P. Sarigiannidis, V. Argyriou, and T. Lagkas, "Advancing SDN from OpenFlow to p4: A survey," *ACM Comput. Surveys*, vol. 55, no. 9, pp. 1–37, Sep. 2023.

[9] A. Khraisat and A. Alazab, "A critical review of intrusion detection systems in the Internet of Things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges," *Cybersecurity*, vol. 4, no. 1, pp. 1–27, Mar. 2021.

[10] A. Yazdinejadna, R. M. Parizi, A. Dehghantanha, and M. S. Khan, "A kangaroo-based intrusion detection system on software-defined networks," *Comput. Netw.*, vol. 184, Jan. 2021, Art. no. 107688.

[11] S. A. Darade and M. Akkalakshmi, "Load balancing strategy in software defined network by improved whale optimization algorithm," *J. High Speed Netw.*, vol. 27, no. 2, pp. 151–167, Jul. 2021.

[12] F. Hauser, M. Häberle, D. Merling, S. Lindner, V. Gurevich, F. Zeiger, R. Frank, and M. Menth, "A survey on data plane programming with p4: Fundamentals, advances, and applied research," *J. Netw. Comput. Appl.*, vol. 212, Mar. 2023, Art. no. 103561.

[13] C. D. Bhowmik and T. Gayen, "Traffic aware dynamic load distribution in the data plane of SDN using genetic algorithm: A case study on NSF network," *Pervas. Mobile Comput.*, vol. 88, Jan. 2023, Art. no. 101723.

[14] V. Huang, G. Chen, P. Zhang, H. Li, C. Hu, T. Pan, and Q. Fu, "A scalable approach to SDN control plane management: High utilization comes with low latency," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 682–695, Jun. 2020.

[15] Z. A. Bhuiyan, S. Islam, M. M. Islam, A. B. M. A. Ullah, F. Naz, and M. S. Rahman, "On the (in)Security of the control plane of SDN architecture: A survey," *IEEE Access*, vol. 11, pp. 91550–91582, 2023.

[16] D. Espinel Sarmiento, A. Lebre, L. Nussbaum, and A. Chari, "Decentralized SDN control plane for a distributed cloud-edge infrastructure: A survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 1, pp. 256–281, 1st Quart., 2021.

[17] M. Pradhan, C. K. Nayak, and S. K. Pradhan, "Intrusion detection system (IDS) and their types," in *In Securing the Internet of Things: Concepts, Methodologies, Tools, and Applications*. Hershey, PA, USA: IGI Global, 2020, pp. 481–497.

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

[19] C. Birkinshaw, E. Rouka, and V. G. Vassilakis, "Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks," *J. Netw. Comput. Appl.*, vol. 136, pp. 71–85, Jun. 2019.

[20] J. Ye, X. Cheng, J. Zhu, L. Feng, and L. Song, "A DDoS attack detection method based on SVM in software defined network," *Secur. Commun. Netw.*, vol. 2018, pp. 1–8, Aug. 2018.

[21] S. Xu, X. Wang, G. Yang, J. Ren, and S. Wang, "Routing optimization for cloud services in SDN-based Internet of Things with TCAM capacity constraint," *J. Commun. Netw.*, vol. 22, no. 2, pp. 145–158, Apr. 2020.
[22] 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.
[23] S. Karnani, N. Agrawal, and R. Kumar, "A comprehensive survey on low-rate and high-rate DDoS defense approaches in SDN: Taxonomy, research challenges, and opportunities," *Multimedia Tools Appl.*, vol. 83, no. 12, pp. 35253–35306, Sep. 2023.
[24] M. Rahouti, K. Xiong, and Y. F. Xin, "Secure software-defined networking communication systems for smart cities: Current status, challenges, and trends," *IEEE Access*, vol. 9, pp. 12083–12113, 2020.
[25] D. Tang, Y. Yan, S. Zhang, J. Chen, and Z. Qin, "Performance and features: Mitigating the low-rate TCP-targeted DoS attack via SDN," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 428–444, Jan. 2022.
[26] D. M. Brandão Lent, M. P. Novaes, L. F. Carvalho, J. Lloret, J. J. P. C. Rodrigues, and M. L. Proença, "A gated recurrent unit deep learning model to detect and mitigate distributed denial of service and portscan attacks," *IEEE Access*, vol. 10, pp. 73229–73242, 2022.
[27] T. Han, S. R. U. Jan, Z. Tan, M. Usman, M. A. Jan, R. Khan, and Y. Xu, "A comprehensive survey of security threats and their mitigation techniques for next-generation SDN controllers," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 16, p. e5300, Aug. 2020.
[28] F. M. Alotaibi and V. G. Vassilakis, "Toward an SDN-based web application firewall: Defending against SQL injection attacks," *Future Internet*, vol. 15, no. 5, p. 170, Apr. 2023.
[29] G. Engelen, V. Rimmer, and W. Joosen, "Troubleshooting an intrusion detection dataset: The CICIDS2017 case study," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, 2021, pp. 7–12.
[30] A. Mahfouz, A. Abuhussein, D. Venugopal, and S. Shiva, "Ensemble classifiers for network intrusion detection using a novel network attack dataset," *Future Internet*, vol. 12, no. 11, p. 180, Oct. 2020.
[31] D. He, J. Dai, X. Liu, S. Zhu, S. Chan, and M. Guizani, "Adversarial attacks for intrusion detection based on bus traffic," *IEEE Netw.*, vol. 36, no. 4, pp. 203–209, Jul. 2022.
[32] M. Elshrkawey, M. Alalfi, and H. Al-Mahdi, "An enhanced intrusion detection system based on multi-layer feature reduction for probe and dos attacks," *J. Internet Services Inf. Secur. (JISIS)*, vol. 11, no. 4, pp. 40–57, 2021.
[33] B. S. Bhati, C. S. Rai, B. Balamurugan, and F. Al-Turjman, "An intrusion detection scheme based on the ensemble of discriminant classifiers," *Comput. Electr. Eng.*, vol. 86, Sep. 2020, Art. no. 106742.
[34] Y. Otoum, D. Liu, and A. Nayak. "Dl-IDS: A deep learning-based intrusion detection framework for securing IoT," *Trans. Emerging Telecommun. Technol.*, vol. 33, no. 3, p. e3803, 2022.
[35] Y.-C. Wang and P.-Y. Su, "Collaborative defense against hybrid network attacks by SDN controllers and P4 switches," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 2, pp. 1480–1495, Mar. 2024.
[36] R. Xie, J. Cao, Q. Li, K. Sun, G. Gu, M. Xu, and Y. Yang, "Disrupting the SDN control channel via shared links: Attacks and countermeasures," *IEEE/ACM Trans. Netw.*, vol. 30, no. 5, pp. 2158–2172, Oct. 2022.
[37] J. D. Gadze, A. A. Bamfo-Asante, J. O. Agyemang, H. Nunoo-Mensah, and K. A.-B. Opare, "An investigation into the application of deep learning in the detection and mitigation of DDOS attack on SDN controllers," *Technologies*, vol. 9, no. 1, p. 14, Feb. 2021.
[38] J. Xu, L. Wang, and Z. Xu, "An enhanced saturation attack and its mitigation mechanism in software-defined networking," *Comput. Netw.*, vol. 169, Mar. 2020, Art. no. 107092.
[39] S. Iqbal, K. N. Qureshi, F. Shoaib, A. Ahmad, and G. Jeon, "Minimize the delays in software defined network switch controller communication," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 13, p. e5940, Jun. 2022.
[40] H. Zhong, J. Xu, J. Cui, X. Sun, C. Gu, and L. Liu, "Prediction-based dual-weight switch migration scheme for SDN load balancing," *Comput. Netw.*, vol. 205, Jul. 2022, Art. no. 108749. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128621005946
[41] J. E. Varghese and B. Muniyal, "An efficient IDS framework for DDoS attacks in SDN environment," *IEEE Access*, vol. 9, pp. 69680–69699, 2021, doi: 10.1109/ACCESS.2021.3078065.
[42] M. Hanselmann, T. Strauss, K. Dormann, and H. Ulmer, "CANet: An unsupervised intrusion detection system for high dimensional CAN bus data," *IEEE Access*, vol. 8, pp. 58194–58205, 2020, doi: 10.1109/ACCESS.2020.2982544.
[43] A. Thakkar and R. Lohiya, "A review of the advancement in intrusion detection datasets," *Proc. Comput. Sci.*, vol. 167, pp. 636–645, Jul. 2020.
[44] R. S. Krishnan, E. G. Julie, Y. H. Robinson, R. Kumar, L. H. Son, T. A. Tuan, and H. V. Long, "Modified zone based intrusion detection system for security enhancement in mobile ad hoc networks," *Wireless Netw.*, vol. 26, no. 2, pp. 1275–1289, Feb. 2020.
[45] W. Meng, W. Li, L. T. Yang, and P. Li, "Enhancing challenge-based collaborative intrusion detection networks against insider attacks using blockchain," *Int. J. Inf. Secur.*, vol. 19, no. 3, pp. 279–290, Jun. 2020.
[46] W. Li, W. Meng, and L. F. Kwok, "Surveying trust-based collaborative intrusion detection: State-of-the-art, challenges and future directions," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 280–305, 1st Quart., 2022.
[47] A. Waleed, A. F. Jamali, and A. Masood, "Which open-source IDS? Snort, suricata or zeek," *Comput. Netw.*, vol. 213, Aug. 2022, Art. no. 109116.
[48] K. A. P. Da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, "Internet of Things: A survey on machine learning-based intrusion detection approaches," *Comput. Netw.*, vol. 151, pp. 147–157, Jul. 2019.
[49] M. A. Ferrag, L. Shu, O. Friha, and X. Yang, "Cyber security intrusion detection for agriculture 4.0: Machine learning-based solutions, datasets, and future directions," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 3, pp. 407–436, Mar. 2022.
[50] H. Asad, S. Adhikari, and I. Gashi, "A perspective-retrospective analysis of diversity in signature-based open-source network intrusion detection systems," *Int. J. Inf. Secur.*, vol. 23, pp. 1–16, Dec. 2023.
[51] Y. Otoum and A. Nayak, "As-IDS: Anomaly and signature based IDS for the Internet of Things," *J. Netw. Syst. Manage.*, vol. 29, pp. 1–26, Aug. 2021.
[52] M. C. van Rossum, L. B. Vlaskamp, L. M. Posthuma, M. J. Visscher, M. J. M. Breteler, H. J. Hermens, C. J. Kalkman, and B. Preckel, "Adaptive threshold-based alarm strategies for continuous vital signs monitoring," *J. Clin. Monitor. Comput.*, vol. 2, pp. 1–11, Apr. 2021.
[53] A. A. Neghabi, N. J. Navimipour, M. Hosseinzadeh, and A. Rezaee, "Nature-inspired meta-heuristic algorithms for solving the load balancing problem in the software-defined network," *Int. J. Commun. Syst.*, vol. 32, no. 4, p. e3875, Mar. 2019.
[54] A. Amouri, V. T. Alaparthy, and S. D. Morgera, "A machine learning based intrusion detection system for mobile Internet of Things," *Sensors*, vol. 20, no. 2, p. 461, Jan. 2020.
[55] M. S. Elsayed, N.-A. Le-Khac, S. Dev, and A. D. Jurcut, "Machine-learning techniques for detecting attacks in SDN," in *Proc. IEEE 7th Int. Conf. Comput. Sci. Netw. Technol. (ICCSNT)*, May 2019, pp. 277–281.
[56] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, "InSDN: A novel SDN intrusion dataset," *IEEE Access*, vol. 8, pp. 165263–165284, 2020.
[57] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016.
[58] L. Dhanabal and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 4, no. 6, pp. 446–452, 2015.
[59] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100379.
[60] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning-based text classification: A comprehensive review," *ACM Comput. Surv.*, vol. 54, no. 3, pp. 1–40, 2021.
[61] 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.
[62] A. Devarakonda, N. Sharma, P. Saha, and S. Ramya, "Network intrusion detection: A comparative study of four classifiers using the NSL-KDD and KDD'99 datasets," *J. Phys., Conf. Ser.*, vol. 2161, no. 1, Jan. 2022, Art. no. 012043.
[63] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset," *IEEE Access*, vol. 9, pp. 22351–22370, 2021.
[64] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," *Comput. Netw.*, vol. 188, Apr. 2021, Art. no. 107840.

[65] University of New Brunswick. (2018). *CSE-CIC-IDS2018 Dataset*. Accessed: Dec. 1, 2021. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2018.html

[66] A. A. Salih and A. M. Abdulazeez, "Evaluation of classification algorithms for intrusion detection system: A review," *J. Soft Comput. Data Mining*, vol. 2, no. 1, pp. 31–40, 2021.

[67] Y. Hande and A. Muddana, "A survey on intrusion detection system for software defined networks (SDN)," in *Research Anthology on Artificial Intelligence Applications in Security*. Hershey, PA, USA: IGI Global, 2021, pp. 467–489.

[68] R. Kaur, R. Kumar, and M. Gupta, "Deep neural network for food image classification and nutrient identification: A systematic review," *Rev. Endocrine Metabolic Disorders*, vol. 24, no. 4, pp. 633–653, Aug. 2023.

[69] P. T. Duy, L. K. Tien, N. H. Khoa, D. T. T. Hien, A. G.-T. Nguyen, and V.-H. Pham, "DIGFuPAS: Deceive IDS with GAN and function-preserving on adversarial samples in SDN-enabled networks," *Comput. Secur.*, vol. 109, Oct. 2021, Art. no. 102367.

[70] P. Karthika and K. Arockiasamy, "Simulation of SDN in mininet and detection of DDoS attack using machine learning," *Bull. Electr. Eng. Informat.*, vol. 12, no. 3, pp. 1797–1805, 2023.

[71] Mininet. (2017). *Announcing Mininet 2.2.2*. Accessed: Dec. 3, 2021. [Online]. Available: http://mininet.org/blog/2017/03/17/announcing-mininet-2-2-2/

[72] A. Majumdar, S. Raj, and T. Subbulakshmi, "ARP poisoning detection and prevention using scapy," *J. Phys., Conf. Ser.*, vol. 1911, no. 1, May 2021, Art. no. 012022.

[73] R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno, "Machine learning algorithms to detect DDoS attacks in SDN," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 16, p. e5402, Aug. 2020.

[74] (2023). *Scapy*. Accessed: Jul. 21, 2019. [Online]. Available: https://scapy.net/

[75] P. Gao, Y. Xu, and H. J. Chao, "OVS-CAB: Efficient rule-caching for open vSwitch hardware offloading," *Comput. Netw.*, vol. 188, Apr. 2021, Art. no. 107844.

[76] (2023). *Open Vswitch*. Accessed: Dec. 20, 2021. [Online]. Available: https://www.openvswitch.org/download/

[77] N. Alsharabi, M. Alqunun, and B. A. H. Murshed, "Detecting unusual activities in local network using snort and wireshark tools," *J. Adv. Inf. Technol.*, vol. 14, no. 4, pp. 616–624, 2023.

[78] (2023). *Snort*. [Online]. Available: https://www.snort.org/downloads

[79] A. Mudgal and S. Bhatia, "Experimental-based comparative study on open-source network intrusion detection system," *Int. J. Internet Technol. Secured Trans.*, vol. 12, no. 5, p. 462, 2022.

[80] A. Tiwari, S. Saraswat, U. Dixit, and S. Pandey, "Refinements in zeek intrusion detection system," in *Proc. 8th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, vol. 1, Mar. 2022, pp. 974–979.

[81] *Zeek*. Accessed: Sep. 14, 2023. [Online]. Available: https://docs.zeek.org/en/master/install.html

[82] I. Mardianto, D. Sugiarto, and K. A. Ashari, "The elastic stack ability test to monitor slowloris attack on digital ocean server," *Ultimatics, Jurnal Teknik Informatika*, vol. 13, no. 2, pp. 120–126, Jan. 2022.

[83] G. Yaltirakli. (2015). *Slowloris*. Accessed: Aug. 12, 2022. [Online]. Available: https://github.com/gkbrk/slowloris

[84] R. Jawaharan, P. M. Mohan, T. Das, and M. Gurusamy, "Empirical evaluation of SDN controllers using mininet/wireshark and comparison with cbench," in *Proc. 27th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2018, pp. 1–2.

[85] *Wireshark*. Accessed: Jul. 12, 2022. [Online]. Available: https://www.wireshark.org/

[86] N. Sahani, R. Zhu, J.-H. Cho, and C.-C. Liu, "Machine learning-based intrusion detection for smart grid computing: A survey," *ACM Trans. Cyber-Phys. Syst.*, vol. 7, no. 2, pp. 1–31, Apr. 2023.

[87] S. H. Brahmanand, N. D. Lal, D. S. Sahana, G. S. Nijguna, and P. Nayak, "A systematic approach of analysing network traffic using packet sniffing with scapy framework," in *Proc. Comput. Netw. Inventive Commun. Technol.* Singapore: Springer, 2022, pp. 811–820, doi: 10.1007/978-981-16-3728-5_60.

[88] D. Li, C. Yu, Q. Zhou, and J. Yu, "Using SVM to detect DDoS attack in SDN network," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 466, Sep. 2018, Art. no. 012003.

[89] M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, and S. Vasupongayya, "Advanced support vector Machine- (ASVM-) based detection for distributed denial of service (DDoS) attack on software defined networking (SDN)," *J. Comput. Netw. Commun.*, vol. 2019, pp. 1–12, Mar. 2019.

[90] A. Santos da Silva, J. A. Wickboldt, L. Z. Granville, and A. Schaeffer-Filho, "ATLANTIC: A framework for anomaly traffic detection, classification, and mitigation in SDN," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp.*, Apr. 2016, pp. 27–35.

[91] A. Kenyon, L. Deka, and D. Elizondo, "Are public intrusion datasets fit for purpose characterising the state of the art in intrusion event datasets," *Comput. Secur.*, vol. 99, Dec. 2020, Art. no. 102022.

[92] C. Song, Y. Park, K. Golani, Y. Kim, K. Bhatt, and K. Goswami, "Machine-learning based threat-aware system in software defined networks," in *Proc. 26th Int. Conf. Comput. Commun. Netw. (ICCCN)*, May 2017, pp. 1–9.

[93] P. Wang, K.-M. Chao, H.-C. Lin, W.-H. Lin, and C.-C. Lo, "An efficient flow control approach for SDN-based network threat detection and migration using support vector machine," in *Proc. IEEE 13th Int. Conf. e-Business Eng. (ICEBE)*, Nov. 2016, pp. 56–63.

[94] A. Alshamrani, A. Chowdhary, S. Pisharody, D. Lu, and D. Huang, "A defense system for defeating DDoS attacks in SDN based networks," in *Proc. 15th ACM Int. Symp. Mobility Manage. Wireless Access*, Nov. 2017, pp. 83–92.

[95] A. AlEroud and I. Alsmadi, "Identifying cyber-attacks on software defined networks: An inference-based intrusion detection approach," *J. Netw. Comput. Appl.*, vol. 80, pp. 152–164, Feb. 2017.

[96] A. Le, P. Dinh, H. Le, and N. C. Tran, "Flexible network-based intrusion detection and prevention system on software-defined networks," in *Proc. Int. Conf. Adv. Comput. Appl. (ACOMP)*, Nov. 2015, pp. 106–111.

[97] 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.

[98] R. T. Kokila, S. T. Selvi, and K. Govindarajan, "DDoS detection and analysis in SDN-based environment using support vector machine classifier," in *Proc. 6th Int. Conf. Adv. Comput. (ICoAC)*, Dec. 2014, pp. 205–210.

[99] D. Jankowski and M. Amanowicz, "On efficiency of selected machine learning algorithms for intrusion detection in software defined networks," *Int. J. Electron. Telecommun.*, vol. 62, no. 3, pp. 247–252, Sep. 2016.

[100] N. T. Van, H. Bao, and T. N. Thinh, "An anomaly-based intrusion detection architecture integrated on OpenFlow switch," in *Proc. 6th Int. Conf. Commun. Netw. Secur.*, Nov. 2016, pp. 99–103.

[101] L. Barki, A. Shidling, N. Meti, D. G. Narayan, and M. M. Mulla, "Detection of distributed denial of service attacks in software defined networks," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2016, pp. 2576–2581.

[102] A. Abubakar and B. Pranggono, "Machine learning based intrusion detection system for software defined networks," in *Proc. 7th Int. Conf. Emerg. Secur. Technol. (EST)*, Sep. 2017, pp. 138–143.

[103] M. Latah and L. Toker, "Minimizing false positive rate for DoS attack detection: A hybrid SDN-based approach," *ICT Exp.*, vol. 6, no. 2, pp. 125–127, Jun. 2020.

[104] J. Li, Z. Zhao, R. Li, and H. Zhang, "AI-based two-stage intrusion detection for software defined IoT networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2093–2102, Apr. 2019.

[105] S. A. Mehdi, J. Khalid, and S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking," in *Proc. Int. Symp. Recent Adv. Intrusion Detection*, Menlo Park, CA, USA. Springer, Sep. 2011, pp. 161–180.

[106] J. Li, Z. Zhao, and R. Li, "Machine learning-based IDS for software-defined 5G network," *IET Netw.*, vol. 7, no. 2, pp. 53–60, Mar. 2018.

[107] R. Sathya and R. Thangarajan, "Efficient anomaly detection and mitigation in software defined networking environment," in *Proc. 2nd Int. Conf. Electron. Commun. Syst. (ICECS)*. Berlin, Germany: Springer, Feb. 2015, pp. 479–484, doi: 10.1007/978-3-642-23644-0_9.

[108] 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.

[109] T. V. Phan, T. Van Toan, D. Van Tuyen, T. Thu Huong, and N. H. Thanh, "OpenFlowsia: An optimized protection scheme for software-defined networks from flooding attacks," in *Proc. IEEE 6th Int. Conf. Commun. Electron. (ICCE)*, 2016, pp. 13–18.

[110] V. Ravi, "Deep learning-based network intrusion detection in smart healthcare enterprise systems," *Multimedia Tools Appl.*, vol. 83, no. 13, pp. 39097–39115, Oct. 2023.

[111] H. Sadia, S. Farhan, Y. U. Haq, R. Sana, T. Mahmood, S. A. O. Bahaj, and A. R. Khan, "Intrusion detection system for wireless sensor networks: A machine learning based approach," *IEEE Access*, vol. 12, pp. 52565–52582, 2024.

[112] S. Y. Ali, U. Farooq, L. Anum, N. A. Mian, M. Asim, and T. Alyas, "Securing cloud environments: A convolutional neural network (CNN) approach to intrusion detection system," *J. Comput. Biomed. Inform.*, vol. 6, no. 2, pp. 295–308, 2024.
[113] S. Boukria and M. Guerroumi, "Intrusion detection system for SDN network using deep learning approach," in *Proc. Int. Conf. Theor. Applicative Aspects Comput. Sci. (ICTAACS)*, vol. 1, Dec. 2019, pp. 1–6.
[114] Q. Niyaz, W. Sun, and A. Y. Javaid, "A deep learning based DDoS detection system in software-defined networking (SDN)," 2016, *arXiv:1611.07400*.
[115] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Proc. Int. Conf. Wireless Netw. Mobile Commun. (WINCOM)*, Oct. 2016, pp. 258–263.
[116] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep recurrent neural network for intrusion detection in SDN-based networks," in *Proc. 4th IEEE Conf. Netw. Softwarization Workshops (NetSoft)*, Jun. 2018, pp. 202–206.
[117] M. S. ElSayed, N.-A. Le-Khac, M. A. Albahar, and A. Jurcut, "A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique," *J. Netw. Comput. Appl.*, vol. 191, Oct. 2021, Art. no. 103160.
[118] S. K. Dey and Md. M. Rahman, "Flow based anomaly detection in software defined networking: A deep learning approach with feature selection method," in *Proc. 4th Int. Conf. Electr. Eng. Inf. Commun. Technol. (iCEEiCT)*, Sep. 2018, pp. 630–635.
[119] Y. Hande and A. Muddana, "Intrusion detection system using deep learning for software defined networks (SDN)," in *Proc. Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Nov. 2019, pp. 1014–1018.
[120] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, M. Ghogho, and F. El Moussa, "DeepIDS: Deep learning approach for intrusion detection in software defined networking," *Electronics*, vol. 9, no. 9, p. 1533, Sep. 2020.
[121] E. M. Bårli, A. Yazidi, E. H. Viedma, and H. Haugerud, "DoS and DDoS mitigation using variational autoencoders," *Comput. Netw.*, vol. 199, Nov. 2021, Art. no. 108399.
[122] G. Kumar and H. Alqahtani, "Machine learning techniques for intrusion detection systems in SDN-recent advances, challenges and future directions," *Comput. Model. Eng. Sci.*, vol. 134, no. 1, pp. 89–119, 2023.
[123] J. Liang and M. Ma, "Co-maintained database based on blockchain for IDSs: A lifetime learning framework," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 2, pp. 1629–1645, Jun. 2021.
[124] W. Elbakri, M. M. Siraj, B. A. S. Al-Rimy, S. N. Qasem, and T. Al-Hadhrami, "Adaptive cloud intrusion detection system based on pruned exact linear time technique," *Comput., Mater. Continua*, vol. 79, no. 3, pp. 3725–3756, 2024.
[125] B. S. Sharmila, B. M. Nandini, S. S. Kavitha, and A. Srivatsa, "Performance evaluation of parametric and non-parametric machine learning models using statistical analysis for RT-IoT2022 dataset: Parametric and non-parametric machine learning models," *J. Sci. Ind. Res. (JSIR)*, vol. 83, no. 8, pp. 864–872, 2024.

AHMED H. JANABI (Member, IEEE) received the B.Eng. degree (Hons.) from the University of Babylon, in 2017, and the Ph.D. degree in cybersecurity from the University of Northampton, U.K., in 2024. He is currently the IT Unit Manager with the College of Engineering and Technology, Al-Mustaqbal University, Babylon, Iraq. With a rich background in academia and research, he has contributed extensively to the fields of software-defined networks, the Internet of Things, and network-level cybersecurity. His academic journey includes roles as a researcher and a teaching assistant, alongside being a respected reviewer for prestigious journals, such as *Nature* (Springer), IEEE Access, and IEEE Transactions on Intelligent Transportation Systems. His current work at Al-Mustaqbal University encompasses leading IT operations, fostering technological advancements, and guiding the next generation of engineers and technologists.

TRIANTAFYLLOS KANAKIS (Member, IEEE) received the B.Eng. degree from the University of Northampton (UoN), U.K., the M.Sc. degree from the King's College of London, the M.B.A. degree from UoN, and the Ph.D. degree from the University of Greenwich. He is currently the Programme Leader of Computer Networks Engineering at UoN. He has more than 15 years of experience in cellular and wireless communications engineering, multiple antenna systems, and cooperative communications. He has numerous publications in leading journals and conferences, and he is the co-author of the Best Paper Award-winning work in forward error correction presented in IEEE CEEC 2016. He is the main organizer of the UoN IoT Workshop and an academic advisor of the IEEE Student Branch Manager at UoN. He has long experience in the industry where he has worked as a network specialist at AT&T, a technical consultant, and a technical trainer in the LTE4G era, and more recently as a technology consultant in the constructions sector while he has served the Hellenic Navy as part of his national military service. He is a fellow of the Higher Education Academy (FHEA). He has been a keynote speaker at numerous events worldwide and has acted as a technical committee member of various conferences in the field of telecommunications engineering while he has served as a for to numerous conferences and journals, including IEEE Access.

MARK JOHNSON received the Ph.D. degree from the University of Northampton (UoN), in 2007. He is currently the Programme Leader of the Software Engineering Course, UoN. He is also the Deputy Subject Leader of Technology. He is a Key Consultant at Nyx Technologies Ltd. and works on a variety of contemporary software and hardware-based projects. He has more than 20 years of experience as a software engineer and an academician. He has a significant research interest in pedagogic research within computing and has worked with several commercial organizations (including, Barclaycard, Triad Group PLC, and KPMG) exploring the viability of student engagement with industry as an essential component of undergraduate course provision. As an academic has supported the development of over 20 computing course programmes (at undergraduate and postgraduate level) during his 22 years in higher education. He is a fellow of the Higher Education Academy.