Preprints.org

Review

Not peer-reviewed version

A Survey of Intrusion Detection Systems based Machine Learning Approaches Applied to Software-Defined Networks (SDN): Research Issues and Challenges

Ahmed Hasan Kadhim Janabi *, Triantafyllos Kanakis , Mark Johnson

Posted Date: 19 December 2023

doi: 10.20944/preprints202312.1449.v1

Keywords: Software-Defined Network (SDN), Intrusion Detection System (IDS), Machine Learning (ML), Deep Learning (DL), non-Deep Learning, OpenFlow, Control Plane, Data Plane, attack, security, and challenges.

Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Review

A Survey of Intrusion Detection Systems Based Machine Learning Approaches Applied to Software-Defined Networks (SDN): Research Issues and Challenges

Ahmed Hasan Kadhim Janabi[1,2,*], Triantafyllos Kanakis [1] and Mark Johnson [1]

[1] Department of Computing, The University of Northampton, Northampton NN1 5PH, United Kingdom
[2] Department of Air Conditioning and Refrigeration, Al-Mustaqbal University, Babylon 51001, Iraq
* Correspondence: author: Ahmed H. Janabi (E-mail: Ahmed.Janabi@northampton.ac.uk)

Abstract: Cybersecurity has become a critical area in the digital field in recent years. The expansion of networks has revolutionised the way network structures are organised and managed. However, with increased connectivity and the growing complexity of modern networks, the threat of cyber-attacks has become more intense. As technology continues to advance, it brings both opportunities and challenges. One of the major challenges is the need to secure networks and sensitive data from various malicious activities. Traditional networks have evolved to include Software Defined Network (SDN), which offers a more flexible and programmable framework. Researchers should focus on detecting attacks in SDN because SDN networks are becoming more popular and attractive targets for clients due to their programmability and dynamic nature. The centralised controller, known as the backbone of an SDN, becomes a single point of failure and a potential target for attackers if not properly secured. Researchers need to emphasise the detection of attacks in SDN in order to mitigate these risks. By understanding the potential vulnerabilities and attack methods specific to SDN, researchers can develop effective detection approaches and propose countermeasures. This methodology helps protect the network from potential threats and minimises the impact of successful attacks. Therefore, this survey paper provides a review of Intrusion Detection Systems (IDSs) in Software-Defined Networks (SDN) to provide a thorough understanding of SDN security issues.

Keywords: software-defined network (SDN); intrusion detection system (IDS); non-deep learning (non-DL); deep learning (DL); openflow; control plane; data plane; attack; security; and challenges

1. Introduction

In recent years, the rapid development of network technologies has led to the emergence of Software-Defined Networks (SDN) as a promising paradigm for managing and controlling network infrastructures. With its centralised control and programmability, SDN offers enhanced flexibility and scalability compared to traditional network architectures. However, alongside these advantages come new challenges, particularly in terms of security.

The motivation behind this survey stems from the critical need to address the escalating security concerns within the dynamic landscape of SDNs. The exponential growth in network interconnectivity and the pivotal role played by SDNs in modern networking underscore the necessity for robust security measures. The objective of this survey is to meticulously examine and elucidate the crucial role of Intrusion Detection Systems (IDS) employing machine learning approaches within SDNs.

Our aim is to not only comprehensively outline the existing security vulnerabilities inherent in SDNs but also to explore and evaluate the efficacy of IDS mechanisms in mitigating evolving security threats. By delving into machine learning-based approaches for IDS implementation within SDNs, this survey strives to contribute significantly to the fortification of network security paradigms.

Our survey begins by introducing SDN and providing an overview of its key components, including the OpenFlow Protocol, which enables centralized network control. We also discuss the

© 2023 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

security features of SDN and highlight the vulnerabilities that make the SDN environment susceptible to various types of attacks.

Furthermore, we explore the different types of network attacks that commonly target SDNs, including Distributed Denial of Service (DDoS) attacks, Denial of Service (DoS) attacks, Portscan attacks, SQL injection, document infiltration, probe attacks, and penetration attacks such as User to Root (U2R) and Remote to Local (R2L). Understanding these specific attacks is crucial for devising effective intrusion detection mechanisms within SDNs.

In the subsequent sections, we focus on the methods of intrusion detection in SDNs, including signature and threshold detection methods, as well as anomaly-based detection approaches. Specifically, we delve into non-Deep Learning-based, and Deep Learning (DL) approaches as potential techniques for improving intrusion detection capabilities within SDNs. DL-based approaches and non-DL-based approaches are both subsets of Machine Learning (ML) techniques used in various domains, including intrusion detection.

Finally, we discuss the important findings of existing research and identify research gaps that need to be addressed. These include the need for effective network traffic processing, distributed processing stages over OpenFlow devices, accurate detection of slow DDoS/DoS attacks, the usage of up-to-date datasets for training and testing proposed models, achieving high accuracy with limited raw features through Deep Learning and non-Deep Learning approaches, testing models in real network environments, and investigating the scalability of using multiple controllers.

In conclusion, this survey aims to highlight the research issues and challenges associated with developing effective IDS based on machine learning approaches for SDNs. By addressing these challenges, we can enhance the security and resilience of SDN environments against evolving network attacks. SDN allows networks to be controlled by multiple applications, reducing the number of networking devices needed and simplifying physical connectivity and configurations. Consequently, network operators can customize the network's behaviour to accommodate modern services and security applications [1]. Figure 1 illustrates the classification of Intrusion Detection Systems (IDS) in SDN networks.

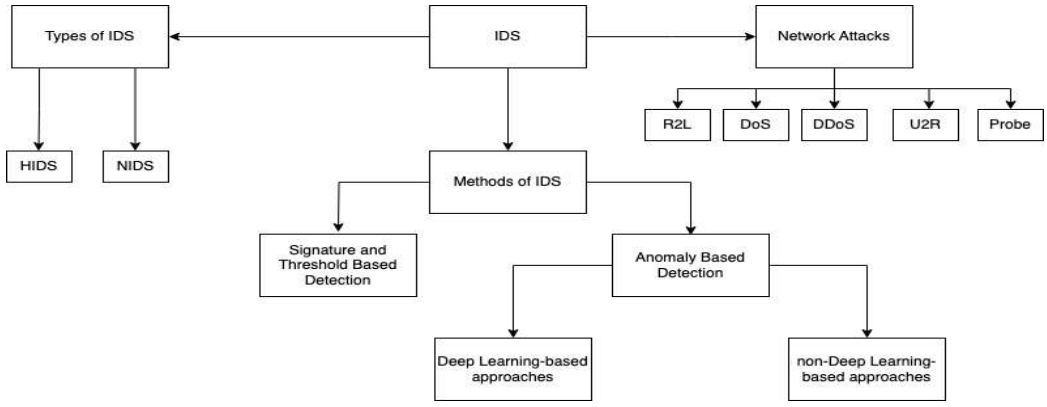

Figure 1. Taxonomy of IDS in SDN.

The main contribution of this paper is introducing a literature review of Intrusion Detection Systems (IDSs) in SDN to present a full understanding of SDN security issues. This survey includes the most recent research related to SDN security. Moreover, this paper discussed the tools used in IDS evaluations. Furthermore, the review paper contains a critical review and research gaps regarding security issues.

The rest of this paper is organised as follows: Section 2 provides an overview of Software-Defined Networks (SDN), including the OpenFlow Protocol and the security features of SDN. Section 3 explores the different types of network attacks that target SDNs and discusses their effects on the SDN environment, as well as the points vulnerable to attacks. Section 4 focuses on the methods of intrusion detection in SDNs, covering signature and threshold detection methods, as well as anomaly-based detection approaches. Section 5 and 6 discusses the evaluation metrics and tools

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

available for IDS in SDNs. Section 7 presents the findings of existing research and identifies research gaps that need to be addressed. Finally, Section 8 offers a conclusion summarising the key insights and recommendations from this survey.

2. Software-Defined Networks (SDN)

SDN stands for Software-Defined Network. It is an innovative approach to networking architecture. This technology allows for centralised and intelligent management of networks through applications like traffic classification and security measures. The internet is expanding rapidly, posing challenges for traditional networks that lack flexibility and capacity to meet organisational needs. SDN addresses these difficulties and offers promising solutions. However, there are challenges and issues in implementing centralised and programmable techniques, requiring contemporary security solutions such as Intrusion Detection Systems (IDS). Recently, security solutions have utilised Machine Learning techniques, particularly Deep Learning algorithms, to enhance accuracy and efficiency [1].

The 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 Programmable Interface (API). SDN decouples the forwarding and control functions in the network [2]. The forwarding devices, such as switches and routers, are separated from control logic and moved into the logical controller. This controller aims to centralize the network [2]. Therefore, the controller organizes the data plane. The separation of the data and controller planes allows the network's services and applications to be programmable. In other words, SDN breaks down traditionally vertical stacks of networking to customize it and improve scalability to adapt to new technology environments. The SDN's primary goal is to permit the networks' engineers and administrators to react rapidly, which leads them to deal with dynamic businesses' necessities effectively [3].

The connections between the layers and planes of SDN, as well as the SDN reference architecture, are depicted in Figure 2. As mentioned, by utilising the control plane, a network administrator can address the security and tactics of the network across the application plane. Additionally, they can redirect network traffic to multiple applications or systems [4].

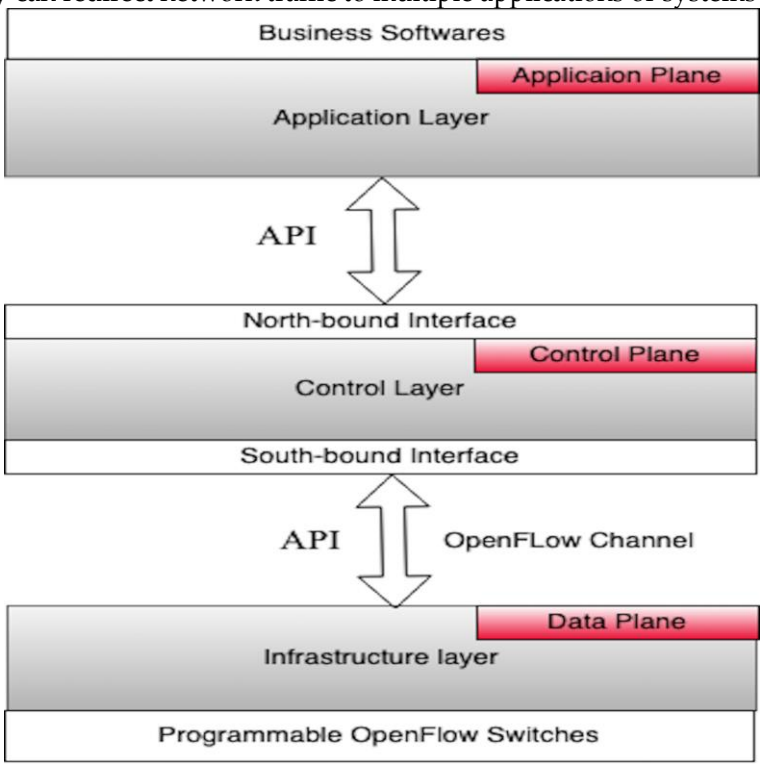

Figure 2. Application Plane, Control Plane, and Data Plane.

2.1. OpenFlow Protocol

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

OpenFlow serves as the communication interface between the data and control plane in SDN networks. As a result, the data plane devices need to adhere to the OpenFlow protocol in order to connect to the control plane [5]. This protocol allows administrators to effectively manage and control the flow of network traffic by separating the control plane from the data plane. OpenFlow was initially introduced by the Open Networking Foundation (ONF) in 2008 and has since garnered widespread acceptance and adoption within the research and industry communities. Additionally, the OpenFlow controller provides a precise and unambiguous network plan, enabling easier controller access to network vulnerabilities and intrusion detection.

At its core, OpenFlow provides a standardised interface between the control and data planes. In conventional networking, these planes are tightly connected. However, OpenFlow splits the control plane from the data plane, allowing a more flexible and programmable network infrastructure. The splitting between these planes makes it easier for the SDN to implement security policies. Many of these players are moving toward SDN technology to revolutionise the design of networks and operations [6]. The following subsections explain the main functions of the data and control planes.

2.1.1. 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 allows traffic flow within the network efficiently. This plane is managed by the SDN controller, which provides the correct dispatch of packets to their targeted destinations [7].

The data plane processes incoming packets, making forwarding decisions based on allocated rules and redirecting them to their destinations. The properties of switches play critical roles in implementing forwarding actions in adherence to the specified forwarding rules. This separation of control and data planes determines the main difference between SDN and traditional networking paradigms, presenting more significant flexibility, programmability, and scalability [8].

In an SDN network, this plane works as a workhorse, where its inherent programmability and dynamic nature adjust to varying traffic conditions on the network. Therefore, it allows for systematic and intelligent routing, traffic engineering, quality of service (QoS) control, and other vital network functions. The invention of the data plane in SDN has revolutionised network administration, offering administrators outstanding control and agility in dealing with ever-evolving network requirements [7].

2.1.2. Control Plane

The control plane in SDN is implemented through a centralised controller, which works as the brain of 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 specifies the best paths for network traffic based on several factors, such as network congestion, bandwidth utilisation, and quality of service conditions. This plane optimises network resource management by dynamically modifying the routing directions based on the current conditions [7].

The control plane is responsible for implementing network policies. It allows administrators to define and enforce policies that govern network traffic behaviour more flexibly. These policies include access control rules and traffic prioritisation. The responsibility that falls on this plane is to ensure these policies are applied across the network [9].

Furthermore, the control plane enables network monitoring and troubleshooting. It collects real-time traffic information from the data plane and analyses it to detect anomalies, specify performance bottlenecks, and analyse the vulnerable points across the network. In addition, this information may be used to make accurate decisions to keep the highest possible network reliability and performance [10].

2.2. SDN Security Features

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

As it is commonly known, software has essential features in its design and implementation, often referred to as architecture. SDN has several features that correspond to these modules. The design characteristics of SDN make it a distinct approach from typical network architecture. SDN features enhance network security, enabling greater flexibility and efficiency. However, SDNs can also be vulnerable by design, susceptible to threats that exploit their weak points and overhead [11]. Hence, the descriptions of SDN design features have been approached from two perspectives. The first one focuses on the components that safeguard the SDN framework against various threats. The second aspect relates to the features that can potentially make it vulnerable.

2.2.1. To achieve resilience of the SDN infrastructure to several attacks

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

(1) **Centralising the monitoring of malformed flow:** The controller manages the network's data. Hence, the controller is able to observe all the suspicious activity throughout the network [12].

(2) **Programmable configuration:** An essential advantage covered by SDN is the programmable features. When any malicious behaviour in the network is detected, the new configuration of a program acts instantaneously to deal with the identified anomalies [11].

2.2.2. Features that make the SDN environment vulnerable to various attacks

SDN's design has vulnerabilities that make it weak against various security threats.

(1) The Separation of the planes: The separation of planes causes vulnerabilities to various attacks. Both planes start transferring the data between each other by employing the OpenFlow protocol. Therefore, an intruder can take advantage of attacking the channel by executing DoS, DDoS, and saturation attacks. Thus, congestion will occur at the channel bandwidth between the switch and the controller [10].

(2) The controller suffers from cascading and single-point failures: In any SDN-based infrastructure, the controller is the primary target for intruders. When the SDN design relies on a centralized entity, it becomes vulnerable to a single point of failure. If the controller crashes, most network functionalities, including traffic monitoring, will be disrupted, and security measures will be compromised [8]. The nature of a single controller makes it ineffective in handling significant network traffic. To address this issue, multiple controllers can be deployed. However, this approach may impact the authenticity, scalability, and consistency of privacy rules within each controller domain. This behaviour can lead to a cascading failure of more than one controller.

(3) A limited TCAM: The OpenFlow switches retain the flow rules in order to store the received packets within the flow tables. SDN switches utilise the Ternary Content Addressable Memory table (TCAM) technology to store the flow rules [9]. TCAM is a memory that enables fast searching within used applications. However, SDN switches have a limitation in terms of storage capacity in their flow tables. As a result, the network becomes susceptible to various attacks [10].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

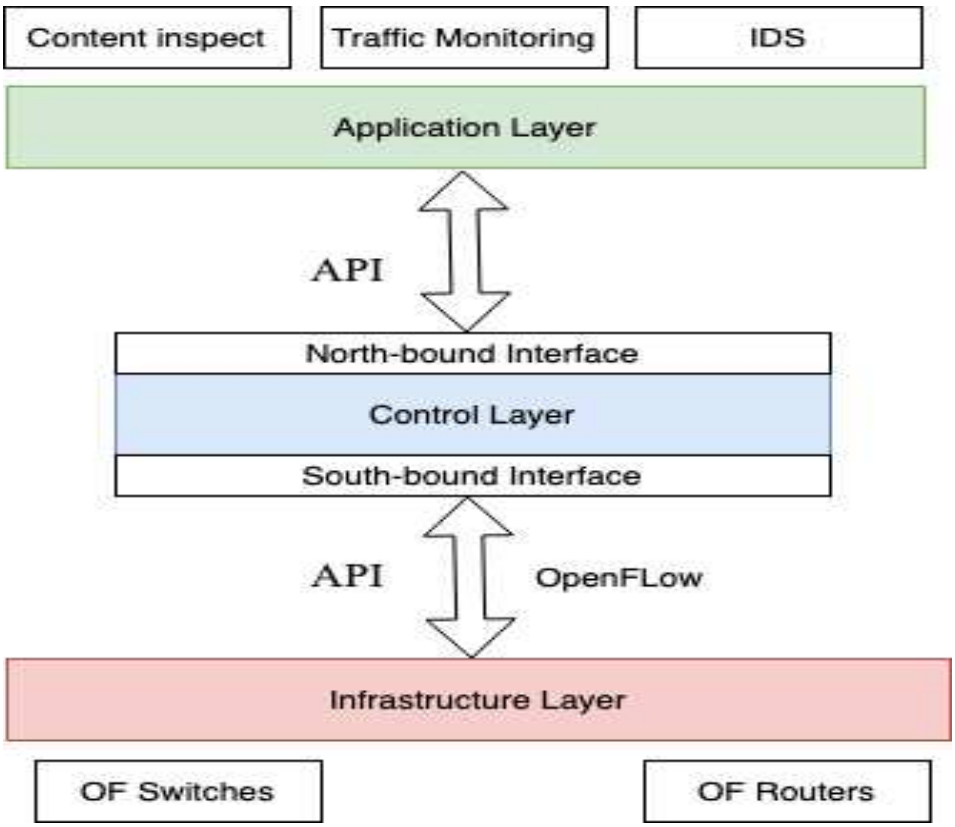

Figure 3. The architecture of SDN when using security applications.

3. Intrusion Detection Systems (IDS) in SDN

IDS play a critical role in safeguarding SDN against various forms of attacks. This section provides an overview of IDS in the context of SDNs. IDS are responsible for monitoring network traffic and detecting potential intrusions or security threats. They analyse network packets, anomalies in traffic patterns, and other indicators to identify suspicious activities. Within SDNs, IDS face unique challenges due to the dynamic and programmable nature of the network infrastructure. This section delves into the different types of IDS that can be employed in SDNs and highlights their strengths and limitations. Understanding the capabilities and limitations of IDS in the context of SDNs is pivotal for formulating effective strategies to detect and mitigate security threats within these network environments.

3.1. IDS Overview

Intrusion Detection Systems (IDS) are crucial, particularly when a network enterprise is concerned about security or handles sensitive data. IDS is responsible for defending the network by controlling and monitoring its traffic. When malicious traffic is detected, IDS sends an alert to the administrator and acts by filtering or redirecting the traffic based on specific requirements or installing special policies [13]. The components of a network-based IDS typically include IDS management, IDS collector, and IDS classification. IDS management handles the policies and rules of the IDS, while the collector collects traffic or flows from the database of flow tables. The classification component utilizes classification techniques for forecasting purposes.

The IDS inspects each packet. These packets are received through the switches using the south-bound API and then sent to the application plane through the north-bound API. This procedure is conducted on the standard switch by configuring the controller. Consequently, the controller can observe and analysing all network traffic that flows through the switches [14].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

3.2. Type of IDS

IDS can be categorised upon on their targets as below:

1. **Host-based Intrusion Detection System:** The IDS must be installed on a computer or personal device like a mobile or tablet. It is an important application that examines all the activities on the device. It detects and prevents attacks if there are intruders [15].
2. **Network-based Intrusion Detection System:** This system observes the traffic to detect malignant activities by checking the traffic behaviour at various stages throughout the entire network and raises the alarm when an anomaly is identified [15].

4. Attacks in SDN

Network attacks represent any unauthorised attempt to access data that may compromise network security or stop some services. An explanation of the network attacks is provided in this section.

4.1. Network Attacks Overview

In the context of SDN, network attacks pose a significant threat to the security and integrity of the network infrastructure. Network attacks in SDN refer to malicious activities that exploit vulnerabilities within the SDN environment to compromise its availability, confidentiality, or integrity. These attacks can range from Distributed Denial of Service (DDoS) attacks, which overwhelm network resources and cause service disruptions, to Denial of Service (DoS) attacks, which target specific network components to render them inaccessible. Other network attacks include Portscan attacks, SQL injection, document infiltration, probe attacks, and various types of penetration attacks like User to Root (U2R) and Remote to Local (R2L). Understanding the different network attacks in SDN is crucial to develop effective IDS that can detect and mitigate such threats, enhancing the security of SDN environments.

4.2. Specific Attacks:

This subsection delves into the various types of attacks that target SDN. These attacks are categorised into eight major groups. These groups are explained below.

1. DDoS (Distributed Denial of Service) Attack

DDoS is a kind of cyber-attack where numerous compromised techniques are utilised to flood a target network or a 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 are 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 [16].

DDoS attacks manipulate the basic principles of network communication and can target different planes of the SDN network, including the control and data planes. Commonly, DDoS floods the victim with an extensive volume of packets, consuming network resources, including bandwidth or server processing capacity, or using vulnerabilities in network protocols to suspend communication between other parts of the network [17].

The motivations behind DDoS attacks vary from malicious intention to performing financial profit or advancing ideological agendas. In such attacks, the targets can be businesses or institutions seeking to stop their economy, causing financial losses, spoiling their reputation, or data breaches. The general forms of DDoS attack include Smurf, TCP SYN flooding, and teardrop [18].

2. DoS (Denial of Service) Attack

DoS attack is the same DDoS concept as the main target for both is to disrupt the resources of a targeted system or service, but the DoS attacks are implemented by utilising a single source rather than multi-source like DDoS attacks.

3. Portscan attack

A *portscan attack* is defined as a mechanism used by an attacker to probe systems for open ports. This method enables the attackers to access potential vulnerabilities and get illegal access to the target

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

system. This attack systematically queries a range of network ports on a given host to define which ports are open, closed, or filtered. The attacker employs the obtained information to breach sensitive services and threaten the victim. It is difficult to prevent this attack due to its stealthy nature [19].

4. SQL Injection

An SQL injection attack is a malignant utilisation that targets the weak points in SDN controllers and related databases employing SQL applications. Due to the significant impacts of SDN controllers in governing the network's policies, an SQL injection attack can seriously affect network security and functionality. By injecting malignant SQL queries into user-supplied data, attackers can exploit the network's policies, compromise the integrity of network flows, and gain unapproved access to sensitive data. SDN architectures should apply robust security measures, such as IDS-based ML, to prevent this threat [20].

5. Document infiltration

A *document infiltration attack* is a refined and targeted cyberattack directed at obtaining sensitive data stored in documents, either in physical or electronic format. Hackers employ complicated intrusion strategies to breach the security of a system to get unlawful access to documents.

These types of attacks use vulnerabilities in information systems, like human errors, or utilising advanced hacking approaches, including phishing, malware, or social engineering. Once the attacker logs into the system, the Intruder uses techniques to contraband the targeted documents, such as copying, downloading, or sending them to remote servers. Document infiltration attacks involve a high risk to the confidentiality, and integrity of secret information. Institutions must utilise vital security measures such as encryption techniques, access controls, and continuous monitoring, to detect such attacks [21].

6. Probe Attack

The hacker scans the whole network to get information about the target machine. With the help of port sweeps responsible for running the host machine's services, Ping Sweep launches an enormous IP address range, mapping for live hosts [18].

7. User to Root (U2R)

This attack is used to allow the unauthorised log to a host machine to access the superuser privileges. This attack is usually launched to get the root privileges of the user account. The basic types include input modification and buffer overflow [18].

8. Remote to Local (R2L)

The R2L attack is an initial or traditional attempt to plagiarise offline users. The attackers enter the system by sending data packets over the network in this method. The victim and the hacker need to reach others or be on the same network. Unauthorised access to the user can be gained through Social Engineering or password sniffing methods [18].

4.3. Effects of Attacks on the SDN Environment

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

(1) Saturation of the controller resource: The controller is seen as the central component of the SDN network. As a result, if the controller crashes, it can have a significant impact on network performance. The controller's capabilities may be overwhelmed by the processing of many flooded requests generated by DDoS attacks. When the controller is overloaded, it becomes impossible to manage all the incoming data flows effectively. Consequently, a substantial amount of regular traffic may experience delays or fail to undergo crucial processing [18].

(2) Switch overloading: The primary threats to SDN architectures are Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks. These attacks involve flooding the switches with numerous harmful packets. When a switch cannot find a corresponding entry in the flow table for a malicious packet, all unmatched entry requests are stored in a single buffer and then sent to the controller for a specific rule application. However, because the switch has limitations in terms of TCAM, it cannot process all incoming packets. As a result, the flow table's memory can be exceeded

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

by the incoming flow and requests. This leads to regular traffic being unable to undergo the necessary processing [22].

(3) Congestion bandwidth between switch and controller: The missing events happen when two actions occur during the arrival of new packets. The first one takes place when incoming packets are stored in a buffer in the flow table. The second happens when an OpenFlow request includes information from the packet header. The controller is notified and receives this information when the buffer reaches its maximum capacity. As a result, packets might collide with another associated interface, leading to users experiencing a service blockage [23].

4.4. Vulnerable Points in the SDN Environment

The SDN structure consists of three planes: data, control, and application. This characteristic of SDN makes the network susceptible to multiple attacks, including DDoS and DoS. The specific points that attack target are described below:

(1) The SDN switches: The primary purpose of OpenFlow switches is to send the newly received traffic to the controller for further action and processing. However, the switch has a flow table with limited memory capacity. Consequently, this becomes a significant concern in terms of security as it enables attackers to flood the switch with an enormous number of harmful packets [12].

(2) The SDN switches links: The packets transfer between the switches before they reach the controller. The packets are not encrypted when they are transferred over the links. As a result, an attacker can easily capture the packets, particularly in wireless environments. This type of attack is referred to as man-in-the-middle. Additionally, at this stage, DoS and DDoS attacks can be carried out, where the attacker can send a large amount of malicious traffic to disrupt the transmission of normal traffic, thereby halting the services.

(3) The SDN controller: As it represents the central component of the network, it performs crucial environmental actions. Any abnormality can halt network operations. The controller affects the functionality of a network, making it an attractive target for attackers. However, if only one controller is used, the network becomes vulnerable to a single point of failure. Therefore, this significant security concern must be addressed [23].

(4) Controller and switch communication: When a newly received packet cannot match any records in the flow table, it will then be redirected to the controller for further processing. As a result, the rules used for forwarding will be added to the flow table entries in the particular switch. At this stage, an attacker has the opportunity to intercept the packet and inject harmful rules or modify the existing ones. Because of this, the majority of packets may be sent in the wrong direction [17]

(5) Applications: The applications that are implemented in the SDN control layer include traffic monitoring and classification. These applications are created by third parties to add necessary security requirements. Attackers focus on these applications in order to access sensitive information or introduce harmful rules to the controller. An unauthorized user could carry out this process while communicating with the API. Consequently, SDN applications are seen as a direct target to disrupt the service of controllers [19].

5. Methods of Intrusion Detection System in SDN

The Intrusion Detection System (IDS) analyses the data from the packet, which is the entire data of the packet obtained by monitoring all network flows. This information consists of a packet header, as well as the number of bytes and packets in both the source and destination. The IDS employs various analysis and detection methods to assess and monitor the movement of packets throughout the network. These methods can be categorized based on how they detect potential intrusions. The primary techniques include Signature-based and Anomaly-based methods. Table 1 illustrates the key distinctions between these techniques. Additionally, other approaches are explained below.

Table 1. IDS methods Comparison [24].

#	Methods Signature - Based	Methods Anomaly - Based

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

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	A high rate of Alarm
6	Low scalability	High Scalability

5.1. Signature and Threshold Detection Method

This method of detection is known as knowledge-based detection. It identifies attacks or abnormal behaviours by examining patterns or rules and comparing the observed behaviour against these regulations or patterns. The rules are used to determine whether a particular activity pattern is malicious or normal. This approach is employed when there is a need to detect an attack with a distinctive and unambiguous signature [24].

5.2. Anomaly - Based Detection

This detection method can be categorised as behaviour-based, statistical anomaly-based, or baselining. It involves collecting data on the normal behaviour of users during a specific period [6]. Statistical tests are then used to efficiently determine whether a user is exhibiting normal behaviour or being attacked, using Machine Learning techniques. Within Machine Learning type of detection, there are two sub-categories: non-DL-based approaches, and DL-based approaches, which are both belong to Machine Learning approaches [23].

Non-DL-based approaches: Non-deep learning (DL) based approaches refer to machine learning techniques other than deep learning algorithms. These approaches include traditional machine learning algorithms such as decision trees, random forests, support vector machines, and naive Bayes classifiers. Non-DL-based approaches are often used in behaviour-based and statistical anomaly-based detection methods. They are effective in detecting known attack patterns but may struggle with detecting complex or evolving attack techniques [19].

DL-based approaches: DL-based approaches utilise deep learning algorithms, such as artificial neural networks, convolutional neural networks (CNNs), or recurrent neural networks (RNNs). These approaches have the ability to automatically learn complex patterns and features from data, making them suitable for detecting sophisticated and evolving attack techniques. DL-based approaches are particularly effective in identifying unknown or zero-day attacks. However, they often require a large amount of labelled training data and computational resources [20].

Overall, both non-DL-based and DL-based approaches are part of the broader category of machine learning approaches used for intrusion detection. The choice of approach depends on the specific requirements of the system, the available data, and the types of attacks to be detected.

Table 2. Comparison of non-DL and DL Techniques in IDS within SDNs: Pros and Cons.

Aspect	non-DL Approaches within IDS in SDNs	Deep Learning (DL) Approaches within IDS in SDNs
Pros	- **Interpretability and Explainability:** Non-DL models offer explicit rules, aiding in understanding decision-making [24]. **- Efficiency with Moderate-Sized Datasets:** Techniques like SVMs perform well without excessive computational demands [22]. **- Adaptive Learning:** non-DL models can adapt to changing behaviours effectively [17]. **- Availability of Off-the-Shelf Algorithms:** wide range of established non-DL algorithms are available.	**- Complex Feature Extraction:** DL architectures excel in extracting intricate features from raw data [24]. **- Superior Accuracy in Complex Scenarios:** DL models often achieve higher accuracy [23]. **- Adaptability to Diverse Data Structures:** ADL methods can process diverse data formats without explicit feature engineering [19].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

		- Potential for Real-Time Decision-Making: Optimised DL architectures enable rapid decisions [11].
Cons	**- Struggles with Complex Data Patterns:** Traditional non-DL methods might struggle to discern intricate attack patterns. **- Limited Scalability:** Some non-DL models face challenges in handling large-scale SDNs effectively [9]. **- Dependency on Feature Engineering:** Some non-DL techniques require manual feature engineering [11].	**- High Computational Demands:** DL models require substantial computational resources [4]. **- Black-Box Nature and Interpretability:** DL architectures often result in opaque models [8]. **- Data Dependency and Overfitting Risks:** DL models are highly data-dependent and prone to overfitting [11].

Table 3 provides an overview of the advantages and disadvantages of the classifiers that are widely used for anomaly detection.

Table 3. Advantages and disadvantages of non-DL and DL approaches.

Algorithms	Type	Advantages	Disadvantages
SVM	non-DL	• High accuracy [25-27]. • SVM can reduce the data redundancy [27]	• Cost computationally [27] • It takes a long time for the training. • SVM is used in single-class 1 or 0. To use it in multiple classes will need more computational complexity [27] [28]. • SVM needs more memory and CPU for the implementation [26] • It is improper with a large number of features [29] • SVM does not fast enough in the real-time classification [25]
DT	non-DL	• It works probably with a large number of features [27] • DT able to work with a small number of features [29-30] • It could be used with numerical and categorical data [26]	• It has complexity computational [27] • Giving low accuracy compared with other classifiers depends on the dataset [31] • Requires a long time for the training [29] • It becomes too complicated in numeric data [29]
NB	non-DL	• Low-cost computational [26] [29] [30] • It gives high accuracy in dependencies absence case in the features [27] [29] • It works probably with a large and low number of features [26] [31] • It can be used in multi-class classification [31] • It can classify continuous and discrete data [30]	• It gives low accuracy only if there is a high dependency between the features [26].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

		• It is swift and lightweight for the real-time classification [26] [30]	
RF	non-DL	• It is considered a non-linear classifier with high accuracy [26] [32] • It can classify the continuous and categorical data. • It is a very stable classifier [26].	• High cost and complex computational [32]. • It needs more memory and CPU for the implementation [32]. • It takes a long time for the training [32].
K-NN	non-DL	• It is a fast classifier [30]. • Does not need to a long time to the training and it is easy to execute [27]. • It does not affect the dependency of the data [30].	• Requires selecting the number of neighbours' K' manually [30] • It is not efficient with a large number of features [26]. • Uses more storage [30]
LR	non-DL	• High accuracy [26] • It has less complexity than other classifiers. • Low cost computational	• It gives high accuracy only if there is a high dependency between the features [26]. • Uses only the numerical value in the outputs [26] • It takes a long time for the training [26] • LR used in single-class 1/0 [27]
LogR	non-DL	• High accuracy [26] • It has less complexity than other classifiers [27] • Low cost computational [26]	• It gives high accuracy only if there is a high dependency between the features [26]. • It is very sensitive to noisy data. • Uses only the categorical value in the outputs [26] • It takes a long time to train. • LR used in single-class 1/0
ANN	DL	• High accuracy [33]. • It is not sensitive to noisy data [28] • It can be used in multiple classes. • It could be used with numerical and categorical data. • It can easily define the complicated relationship between the independent and dependent features.	• High cost and complex computational [26] • Requires a long time for the processing. • Requires a long time for the training [28] • Needs big data for the training [33]. • Needs more memory and CPU for the implementation [33]. • Does not fast enough in the real-time classification [28].
CNN	DL	• High accuracy [33] • It can be used in multiple classes [33]	• High cost and complex computational [33]. • Requires a long time for the process [33] • Requires a long time for the training [34] • It needs big data for the training [34]

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

			• Needs more memory/storage and CPU for the implementation [33] • CNN needs to change the dataset to be like the image data before the classification [33]. • It is not fast enough in the real-time classification [34].
GRU	DL	• Highest accuracy than the other DL approaches [35]. • It can be used in multiple classes [35]. • It can be used with a low number of features on the training. • It can be supervised or unsupervised. • Can easily define the complicated relationship between the independent and dependent features [35].	• It has a high cost and complex computational [35]. • It gives lousy prediction when using a dataset with a longer sequence [35]. • Requires a long time for the training [33]. • Uses more storage [35]. • It does not fast enough in the real-time classification [28].

The non-DL and DL based approaches need a dataset to train and test the models to be ready for attack detection. There are two methods to train such models, which are simulation-based and public datasets-based.

The simulation-based mechanism uses a simulation method to generate the dataset in order to train and test the non-DL classifiers. These classifiers categorise the traffic into abnormal or normal. Figure 4 shows sequential stages in creating a simulation dataset. In these specific methods, the researchers built a network topology that included regular hosts to generate normal traffic and other bot hosts to generate abnormal traffic. Scapy and Wireshark are open-source tools that were used by researchers to simulate and generate DDoS, DoS, Prob, Portscan, U2R, and R2L attacks. The features, such as protocol type, source IP address speed, source port rate, and flow packets, will be extracted from normal and abnormal traffic. After pre-processing these features, they will be stored in a CSV file as the raw data, which will train the proposed models [36]. After learning the model, it will be ready to apply non-DL algorithms to classify the normal and malicious packets in the SDN environment accordingly.

The proposed work in the public datasets-based method utilises public datasets to train and test models. The selection of these datasets is crucial for achieving efficient and accurate Intrusion Detection Systems (IDS). However, it should be noted that most publicly available datasets are not realistic and lack the inclusion of most types of attacks. Consequently, this can have a negative impact on the accuracy and performance of IDS [37]. The main reasons behind the inadequacy of these datasets are related to privacy and legal concerns. Additionally, these datasets tend to be outdated and may not encompass the latest behaviours. Furthermore, such datasets often contain many duplicate records. Consequently, using such datasets achieves low accuracy and performance [37]. The public datasets that are currently available were gathered from conventional networks rather than the SDN network. These datasets contain certain characteristics that cannot be accessed in SDN networks [37]. Numerous published datasets also incorporate various attacks like KDD'99 NSL-KDD, CICIDS2017, ISCX2012, Kyoto, and CSE-CIC-IDS2018 [37].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

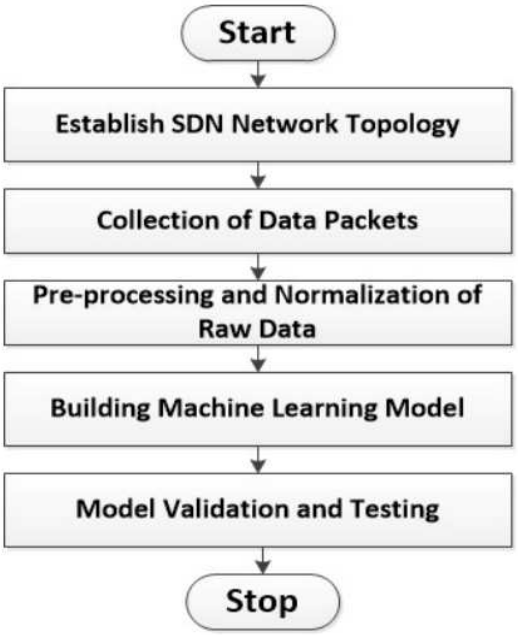

Figure 4. Sequential stages in creating a simulation dataset [36].

1. **The KDD'99** is a widely used data set for the evaluation of anomaly detection methods. Some pitfalls and problems of KDD'99 led to the creation of the NSL-KDD, which is a refined version of the KDD'99 dataset.

2. **CICIDS2017:** Another dataset from the Canadian Institute for Cybersecurity, this dataset is much newer and includes modern attack behaviours. It is highly regarded for having a balance of multiple types of network intrusions and normal behaviours which allows for a comprehensive evaluation of IDS.

3. **ISCX2012:** Created by the Information Security Centre of Excellence (ISCX), this dataset is recognized for its extensive coverage of both normal and attack traffic.

4. **Kyoto:** The Kyoto University's Honeypot datasets are some of the most comprehensive ones for IDS as well. These datasets include a wide variety of attacks, including fewer common ones, making them quite useful for intrusion detection research.

5. **CSE-CIC-IDS2018:** This dataset is another product of the Canadian Institute for Cybersecurity, and it is one of the most recent and comprehensive datasets for developing and training IDS. It contains a wide array of updated attacks which provide researchers with a current 'snapshot' of internet traffic to assist in IDS development and training.

Table 4. A summary of the used datasets in IDS approaches.

Dataset	Year	No. of Features	Notable Attack types	Method of Data Collection
KDD'99	1999	41	U2R, R2L, DoS, Probe	Simulated network traffic
NSL-KDD	2009	41	U2R, R2L, DoS, Probe	Improved version of KDD'99 with unnecessary duplicates removed
CICIDS2017	2017	80	Brute Force, DoS, Heartbleed, Web Attack, Infiltration, Botnet	Real network traffic
ISCX2012	2012	283	HTTP, DoS, scanning, infiltration	Simulated network traffic
Kyoto	2006	Depends on varied releases	Wide array, including less common and recent attacks	Honeypots and real network traffic

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

CSE-CIC-IDS2018	2018	78-85 depending on the scenario	Brute Force, DoS, Heartbleed, Infiltration, Web Attacks	Real network traffic

5.2.1. non-Deep Learning-based Approaches

Non-Deep Learning (non-DL) approaches refer to machine learning techniques that do not involve deep learning algorithms [31]. These approaches are often based on traditional machine learning algorithms, which are shallower models with fewer layers [32]. Here are some examples of non-DL approaches in the context of intrusion detection systems (IDS):

1. Decision Trees: Decision trees are a type of non-DL approach used in IDS. They work by recursively splitting the data into subsets based on the feature values, creating a tree-like structure. The leaves of the tree represent decisions, and the path from the root to a leaf represents a classification or prediction. Decision trees can be used for both anomaly detection and signature-based detection, where they can classify network traffic or user activities based on predefined rules or patterns [29].

2. Random Forests: Random forests are an ensemble learning method that builds multiple decision trees and combines their predictions to improve the overall accuracy and robustness of the model. They are effective in IDS due to their ability to handle high-dimensional data and different feature types. Random forests can be used for both anomaly detection and signature-based detection, as well as for predicting the severity of detected threats [28].

3. Support Vector Machines (SVM): SVMs are a powerful non-DL approach for classification and regression tasks. They work by finding the optimal hyperplane that separates different classes of data points. In the context of IDS, SVMs can be used to classify network traffic or user activities based on known attack patterns or features. SVMs are known for their effectiveness and efficiency in handling large datasets and high-dimensional feature spaces [33].

4. Naive Bayes Classifiers: Naive Bayes classifiers are a family of probabilistic classifier algorithms based on Bayes' theorem. They are "naive" because they assume that the features are conditionally independent, given the class label. Naive Bayes classifiers are often used in IDS for both anomaly detection and signature-based detection. They are known for their simplicity, efficiency, and effectiveness in handling high-dimensional data and text data, such as logs or network traffic [34].

5. K-Nearest Neighbours (KNN): K-Nearest Neighbours (KNN) is a non-DL approach that is used for classification and regression tasks. KNN works by finding the k-nearest data points (neighbours) to a given data point and assigning the majority class label or predicting the mean value based on these neighbours. In the context of intrusion detection systems, KNN can be used for anomaly detection or signature-based detection by comparing the features of known attack patterns or network traffic/user activities to known normal or malicious patterns [30][26].

6. Linear Regression: Linear regression is a non-DL approach used for predicting the relationship between a dependent variable and one or more independent variables. In the context of intrusion detection systems, linear regression can be used to model the relationship between different features of network traffic or user activities and the likelihood of an attack. This can be useful for detecting anomalous behaviour or predicting the severity of detected threats. Although linear regression is not typically used for anomaly detection, it can still be applied to other aspects of intrusion detection systems [27].

The paper in [24] presents an approach to identify non-DL-based attacks. The system is a flow-based IDS designed with the limitations of signature-based IDS in mind. The controller uses a neural network algorithm to classify each packet. The proposed IDS uses the NSL-KDD public dataset for implementation and training, aiming to define DOS, U2R, R2L, and Probes. The model has achieved a Detection Rate of 97.4%. However, during the training stage, using a small database negatively impacts the detection accuracy of real tests. The features are extracted solely from the header packet, lacking coverage of attack behaviours. Moreover, the dataset used contains redundant records. Implementing non-DL approaches with IDS poses the challenge of a bottleneck and single point of

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

failure in the controller due to the processing operation for each packet. Additionally, the proposed model requires a feature selection method to choose effective features during an attack.

The researchers in reference [36] conducted a comprehensive analysis of the existing non-DL approaches for detecting malicious traffic in SDN environments. The evaluation involved identifying the limitations of each algorithm and performing experiments using a publicly available database. The algorithms analysed included Support Vector Machine (SVM), Naïve Bayes, K-Nearest Neighbours (KNN), Adaptive Support Vector Machine (ASVM), Hidden Markov Model (HMM), K-means, Random Forest, Bayesian Networks, Decision Tables, K-medoids, and fuzzy control models. These algorithms were assessed based on previous research, considering their features, classes, datasets, and sizes. However, it should be noted that the comparison of these algorithms is not entirely fair, as each algorithm has its own advantages and disadvantages in different areas. For instance, the paper fails to mention the specific use case of K-means in clustering compared to SVM in classification. These algorithms cannot be directly compared because K-means is effective at separating datasets based on patterns, whereas SVM excels at separating data with predefined patterns, thereby facing the challenge of proper selection. Principal Component Analysis (PCA) is used to reduce the number of parameters required for algorithms that necessitate manual parameter analysis, although it does not address this challenge. The evaluation process for these algorithms involves measuring precision, recall, F-score, and accuracy. However, the scores attained by each algorithm do not necessarily imply the same advantages and disadvantages across the board. Unfortunately, the paper lacks an in-depth analysis of these differences. Lastly, the paper mentions that deep learning (DL) is the most effective method for detecting unknown attacks. However, only non-DL algorithms were tested, and DL algorithms were not considered. Among the non-DL algorithms tested, the J48 algorithm achieved the highest accuracy of 81.5%. The authors did not provide a technique to mitigate the overhead caused by the Intrusion Detection System (IDS) when installed on the controller.

The authors in [38] used the SVM algorithm to identify DDoS attacks in the SDN architecture. The methodology includes the implementation of a system called "Flow Status Collection," but there is no explanation of how this system works. The algorithms are briefly described, but there is no code provided for analysing the implementation, creating uncertainty. Despite being a learning model designed for training with limited data, it is necessary to implement a feature selection method to achieve better results for this type of attack. The results show that the highest detection rate achieved is 98%. However, since this approach yields some non-real values, these false positives can be concerning. If the false positive rate is not 5.88%, it could be up to 7 or 8 times higher due to the type of infrastructure, which is known to have the aforementioned issues. This method faces the challenge of implementing solutions to DDoS attacks. Although successful, it lacks various datasets for a better training model. Integration with datasets from multiple frameworks and public trials for research purposes is already missing.

In this paper [39], the researchers created a simulation platform using Mininet in the SDN network in order to identify DDoS attacks. They employed an SVM classifier to classify each incoming packet. The flow table collection was subjected to extraction of characteristic values in order to provide input for the classifier during the attack detection process. However, this flow table collection did not undergo a feature selection process, which means that certain features could impact the performance and accuracy of the classifier. This presents a challenge concerning the appropriate selection method. Another challenge addressed was the efficient processing of packets, as DDoS attacks involve processing large volumes of data within short periods. However, these particular features were not covered by the practices employed for detecting DDoS attacks. Consequently, while the detection results may be helpful, the performance could be insufficient depending on the testing environment, potentially resulting in a bottleneck at the controller. Based on the results, the highest accuracy rate was achieved through testing using a dataset consisting of 600 TCP packets, resulting in an accuracy of 96.83%. The overall best results were obtained for the TCP protocol compared to UDP and ICMP, as the TCP protocol provides more network fields for classification. Conversely, UDP offers fewer fields but is more representative in terms of the analysed application. Lastly, ICMP

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

exhibits distinctive characters in the payload which allow for individual behaviour during attacks. The proposed system achieved a detection accuracy rate of 95.24% and a false alarm rate of 1.26%.

The authors in [40] have designed a detection model for the SDN network. They used an improved version of SVM to identify DDoS attacks. However, there are still some challenges that have not been solved. These challenges include performance degradation and efficient packet processing. In this model, packets go through the entire SDN network, including OpenFlow switches flow entries, Open Daylight, and finally the API of ASVM. This allows ASVM to classify the packet as an attack or not. The testing process faces the challenge of using realistic datasets or environments. The testing was done with 1000 packets, which is an average number under normal conditions. However, a DDoS attack is not a normal condition. It is the worst condition where the infrastructure is heavily stressed. Additionally, latency increases significantly, although it is considered to be 0.1 seconds under typical conditions in the testing scenario. Furthermore, the model does not consider obfuscated information, which is a technique attackers use to bypass security controls' fingerprinting process. The accuracy of the model was around 97% with training with the minimum required time. However, this does not mean that the algorithm was optimal. More features related to improvements over SVM and a comparison with a normal implementation of SVM and ASVM are missing. The goal of the study was to achieve better results than using a regular version.

In the study in [41], the authors introduced a management framework that combines techniques from information theory with non-DL algorithms. The objective of this research is to address the categorization of traffic analysis. The main challenge lies in efficiently processing packets, so the authors proposed a comprehensive approach to understanding SDN networks. The problem starts with the involvement of human intervention, as using SDN is manageable at a broad level but not at a detailed level. Moreover, extracting network traffic profiles requires significant memory usage, which introduces the issue of selecting appropriate features when dealing with large amounts of data. On a different note, the authors implemented K-means for clustering and SVM (Support Vector Machine) for variety in the classification of abnormal traffic. However, since a clustering algorithm was used to create subsets, there might be an impact on performance. In the testing phase, the authors described the behaviour of DDoS (Distributed Denial-of-Service) and port scanning attacks. The results achieved an accuracy of 88.7% and a precision of 82.3%, which is considered low compared to previous works. However, the cause of this low accuracy percentage is not discussed. Nevertheless, one could speculate that the main issue lies in integrating directives from a user-friendly interface executed in a technical and low-level environment, even when the collection and analysis time is only 0.075 in a topology involving 100 switches.

The authors of the paper in [42] focused on the essential requirement and presented a solution called Eunoia. The proposed model is an IDS based on non-DL in the SDN network. Eunoia aims to monitor, detect, and control any malicious or suspicious traffic in SDN that could harm its internal operations, resulting in network intrusion. The presented solution consists of three subprocesses: data pre-processing (filtering irrelevant data traffic to provide valuable data), data modelling (applying chosen algorithms to predict new audit data), and decision-making and response (helping SDN respond to analysis results through an active learning process and reactive routing in SDN). However, it faces numerous challenges, such as the need for sufficient computational power to handle and process the large amount of data entering the SDN-based network intrusion system as malicious traffic. The features extracted in the model filter valuable data from non-valuable data. The active learning and reactive routing data further examine the analysis results and store the implemented results. Another relevant challenge could be efficient packet processing to avoid causing bottlenecks and deteriorating the system, leading to crashes, or going on standby.

The authors in reference [43] enhanced the SVM (Support Vector Machine) by adopting a behaviour-based approach to integrate the learning algorithm's functionality for monitoring and categorizing threads. The feature extraction method was based on an information gain approach, which described each variable accordingly. However, the process and selection of variables were not defined. While the selection method was described as based on top-ranked features, setting a proper feature selection remains a significant challenge for NIDS (Network Intrusion Detection System). The

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

SVM algorithm mentioned its hyperparameters but did not explain why they were set within specific ranges. This presents a challenge for NIDS, as it aims to ensure consistent and accurate evaluations. The SVM implementation was neither optimized frameworks nor custom-made modifications to other implementations. The main objective of this paper was to identify various attacks, yet the tests conducted focused solely on DoS (Denial of Service) attacks. Consequently, the algorithm demonstrated a high accuracy rate of 97.63% in identifying such attacks. However, the proposed model may not be suitable for large-scale networks and requires an efficient method for processing each packet. Furthermore, this model needs to incorporate a feature selection technique that captures the behaviours of the selected attack.

In this research paper [44], the researchers presented a model designed to ensure that the SDN structure is self-adaptive while responding to network events. This is done by analysing misbehaviour and new flow attacks. The analysis is conducted using non-DL algorithms to classify such behaviours. The proposal has multiple strategic points where analysis needs to be performed. The first one is when the traffic is new, it must be determined if it is an attack or not. The second point is when the traffic is unknown, it needs to be analysed in detail. However, the challenge of waiting for the client to request the resource in a timely manner is not addressed in the paper. Efficient packet processing is essential, especially during DDoS attacks that generate large amounts of abnormal traffic. The algorithm's performance can be impacted if the attack lasts for several hours. The paper mentions that there are 41 features, which brings up the issue of how to properly select these features. Unfortunately, this aspect is not covered in the paper. Consequently, it affects the performance of the algorithm and its execution for each incoming sample from the network traffic. On another note, only 20% of the total samples were used for training, while typically 60% and 40% are used for testing. However, the paper does not provide any analysis of the 20% that were not used. Despite this, the SMO classification achieves an accuracy of 99.4%, which is impressive. Finally, the equation used to identify misbehaviour attacks is simply a calculation of distances between values. Therefore, there is no in-depth analysis of correlation or variance.

The authors in [45] introduced an inference-based IDS to DoS attacks in SDN. The proposed IDS is responsible for managing the separation of network structure information from the control panel. The proposed approach is based on Graph Theory, which focuses on the relationship of context to predict attacks. The authors used the CAIDA dataset and a specific dataset containing labels per connection to test the system. They evaluated this IDS using Precision, Recall, and F-score measurements, with respective results of 0.84, 0.78, and 0.81. This method can also help mitigate the effects of DoS attacks in SDN. However, this approach uses a large amount of data and requires packet processing, which can impact the performance of the controller. Furthermore, the features extracted were taken from the header information, and some of these features were not relevant or used to identify the DoS attack, such as the Node type. Additionally, the selected features do not cover the behaviours of a DoS attack. As a result, an attacker can easily evade this model when initiating an attack.

In this paper [46], the researchers presented a method for detecting network-based attacks such as DoS and Probe attacks in SDN. The proposed system used the Decision Tree approach with the C4.5 algorithm and the 1999 Darpa dataset. The C4.5 algorithm prevented overfitting of the data and dealt with missing attribute values in the training data. The researchers claim that this method can effectively mitigate the impact of DoS and Probe attacks in SDN. They evaluated the model using Precision and Recall measurements, achieving results of 0.989 and 0.964 for the DoS attack, and 0.984 and 0.921 for the Probe attack, respectively. However, their use of the 1999 Darpa dataset for training is questionable since it does not include features related to new types of DoS attacks. Attackers frequently come up with new behaviours for DoS and Probe attacks. The integration of the SDN controller and the IDPS requires a large number of control packets to monitor traffic, which poses another challenge that needs to be addressed in their work.

In this paper [47], the authors presented an IDS that identifies DDoS attacks in SDN networks using traffic data. The proposed system utilises the NOX controller. The main concept is to use a Flow Collector to retrieve traffic information from the flow table. A Self-Organizing Maps (SOM) algorithm

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

is employed for classifying the traffic as normal or malicious. The system's performance is evaluated based on Detection Rate and False Alarm rate measurements, which yield results of 98.61% and 0.59 respectively. However, it is worth noting that the training stage utilizes a small dataset size, which negatively impacts the accuracy of real tests. Furthermore, the extracted features considered only the packet header, failing to capture the complete behaviour of DDoS attacks. Consequently, attackers can easily bypass the system by modifying the header information to resemble a regular packet. Moreover, it is important to extend the scope of the research to include other attack types such as Prob and U2R. Additionally, the authors did not provide any method for preventing the detected attacks.

The authors in [48] proposed an Intrusion Detection System (IDS) on SDN using the Support Vector Machine (SVM) algorithm. They used a kernel function to classify network traffic into normal and abnormal. These kernel functions are commonly used to transform the dataset into a higher dimension and support linear classification. The proposed system is capable of detecting IPsweep, Probe, and DDoS attacks in the control plane. To evaluate the system, the authors used 1998 DARPA and 2000 DARPA datasets for training and testing. Each dataset contains 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 insufficient for understanding attack behaviours, and some features are unrelated to attack practices. Additionally, the SVM classifier takes longer during the training stage. Furthermore, the controller needs to examine all pass-through packets to properly classify them. This process can overwhelm the controller, leading to flooding and congestion.

This paper [49] presents a detection method that is based on an anomaly. This method functions by integrating with the OpenFlow switch. The proposed model helps to prevent and detect both known and unknown attacks in SDN networking. The J48-tree algorithm, which is a variant of the C4.5 decision tree designed for classification purposes, has been utilized. The implementation of the proposed model has been done using the NetFPGA10G board. The system achieved a 91.81% detection rate and a 0.55% false alarm rate. The training and test stage employed the KDD'99 public dataset. However, the authors failed to consider the large amount of data present in the extensive network, which requires significant time and energy for efficient processing. This can result in overloading the controller and the switches, as the OpenFlow switches inspect each incoming packet and send it to the controller for appropriate action, leading to flooding. Moreover, the proposed system extracts an excessive number of features during the investigation stage, thereby consuming the network's resources.

The researchers in [50] have implemented five Intrusion Detection System (IDS) models in an SDN network using various non-DL algorithms. These algorithms include Self-Organizing Maps (SOM) and Learning Vector Quantization (LVQ1), along with their modified versions. The non-DL algorithms utilised in this study are as follows: Self-Organizing Maps (SOM), Multi-pass Self-Organizing Maps (M-SOM), Learning Vector Quantization (LVQ1), Multi-pass Learning Vector Quantization (M-LVQ1), and Hierarchical Learning Vector Quantization (H-LVQ1). These approaches are considered types of Artificial Neural Networks (ANN). The proposed models aim to detect multi-level attacks such as Prop, U2R, R2L, and DoS by classifying each network traffic. All the implemented models have shown successful results, with an average True Positive Rate of 94%. However, the authors have created a dataset containing features that are highly specific and easily extracted from the packet header. As a result, these implemented models may not be effective in detecting real network attacks or capturing their behaviours. Furthermore, the authors have not taken into consideration the challenges of handling large-scale networks with a high volume of packet processing flows. Moreover, the integration of the SDN controller and the IDS introduces an overhead on the controller, leading to controller flooding. This issue poses another challenge that needs to be addressed.

The researchers in [51] introduced a method to handle the dynamic nature of SDNs in order to detect DDoS attacks in the application plane. They accomplished this by classifying the incoming traffic using non-DL algorithms. The specific algorithms employed were Naive Bayes, KNN, K-means, and K-medoids. For the experiment, a private dataset was utilised, taken from a real

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

network's traced file to train and test the models. The accuracy of the implemented algorithms was measured using the Detection Rate metric, with the results being 94%, 90%, 86%, and 88% respectively. However, in order to train and test the models, 50 features were utilized, which in turn requires substantial memory and leads to a lengthy process. As a result, when an attack begins shortly after, the controller will experience a significant overhead.

Lataha and Toker [52] conducted an analysis to demonstrate that SDN can be utilized as a solution for DoS attacks. Their proposed model consists of two phases: intrusion detection. The first phase is flow-based, while the second phase is packet-based. A drawback of this approach is the high utilization of resources, particularly when filtering and analysing packets in two states, resembling a stateful and stateless firewall. These challenges lead to performance degradation during periods of high incoming data rates. The proposed intrusion detection system detects malicious flows by comparing them to legitimate flows using the Knn approach. However, due to the presence of SDN in the environment, the management at a higher level restricts the manipulation of features. On the other hand, the detection of malicious packets is performed through neural networks, which successfully classify both legitimate and malicious traffic. Nonetheless, this algorithm excels mainly at separating the two classes, as suggested. Since the flow-based detection already classified the malicious traffic, the packet detection should utilize the previous algorithm and consider additional properties not accounted for in the layers. As a result, the proposed approach achieved an accuracy of 91.27% and a precision of 0.99%, outperforming other algorithms such as Knn using the NSL-KDD dataset, as well as neural networks and others, under the same circumstances. Although the false positive rate improved, the processing time did not, as it still had to handle packet processing and the controller's bottleneck.

The authors in [53] proposed an IDS in the SDN environment. Their model is based on Artificial Intelligence (AI) and consists of two stages of processing. In the first stage, the authors utilised the Random Forest algorithm to classify the network traffic. For the features selection stage, they employed a Bat algorithm with swarm division and binary differential mutation. This proposed system can identify various types of attacks, including DoS, Probe, DDoS, U2R, and R2L. To evaluate the system's performance and effectiveness, the authors used the KDD Cup 1999 dataset for both training and testing purposes. The results showed that the system achieved an accuracy rate of 96.3%. However, it is worth noting that the proposed classifier requires more time during the training stage. Additionally, the controller in the system needs to examine all the packets passing through it for classification, resulting in increased overhead and creating a bottleneck for the controller. Moreover, the limited size of the raw dataset used for training negatively impacts the algorithm's ability to achieve high detection accuracy, highlighting the need for a larger and more diverse dataset.

The researchers in [54] presented an IDS using an Artificial Intelligence (AI) algorithm. The IDS is implemented in the context of SDN to detect Distributed Denial-of-Service (DDoS) attacks in Home and Small Office/Home Office (SOHO) networks. This approach utilizes the TRW-CB and Rate Limiting techniques to classify real-time traffic. The authors collected the dataset from three locations: Home Network, SOHO, and Internet Service Provider (ISP) using the Mergepcap tool. Once these datasets are collected, they will be used to train the model. The proposed model focuses on extracting essential features from the packet headers for classification, which are obtained at the SDN controller. The NOX controller has been used in conjunction with this model. In the experiment, a detection rate accuracy of 90% was achieved with a 70% false positive rate. However, the limited number of extracted features hinders the detection efficiency, as they do not cover all possible attack behaviours. Attackers can easily bypass the IDS by modifying the packet headers to resemble regular traffic. Additionally, the authors did not consider the bottleneck of the controller in a large-scale network, where its functionalities are not performed efficiently. Therefore, there is a need for a lightweight and efficient method to process packets in the system.

The researchers in [55] proposed a non-DL approach in the SDN 5G environment to identify DDoS, DoS, U2R, and R2L attacks in the SDN controller. The K-means++ and AdaBoost algorithms were used for traffic classification, while the Random Forest (RF) algorithm was employed for feature selection. The authors evaluated the proposed system using the widely used KDD Cup 1999 dataset

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

for IDS. The model achieved an average classification accuracy of 84%. However, the RF algorithm failed to select relevant features that cover U2R and R2L attack behaviours, resulting in low detection accuracy for these attacks. Similar to the previous study, the selected features were extracted from the packet headers and were insufficient to adequately characterize attack behaviours. The authors also did not address the controller's bottleneck in large-scale networks, where its functionalities are not performed efficiently. Thus, there is a need for a lightweight and efficient packet processing method in the system.

Sathya and Thangarajan in [56] focused on security violations in the SDN environment and how the model can be identified to prevent attacks using anomaly-based detection methods. The authors advocate the use of an Intrusion Detection System (IDS) to recognise the Denial of Service (DoS), Probe, User to Root (U2R), and Remote-to-Local (R2L) attacks. The proposed model utilizes the NSL-KDD dataset, which includes four types of attack packets: DoS, Probe, U2R, and R2L. The Feature Selection stage has selected 27 features for the DoS attack, 26 features for the U2R attack, 33 features for the Probe attack, and 33 features for the R2L attack. 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 and minimum false alarms compared to other approaches. This system failed to minimize the number of extracted features selected by the Binary Bat algorithm, resulting in excessive processing time and memory usage at the controller.

In [57], the authors proposed an effective IDS in SDN environments to identify DDoS attacks using a Sequential Probability Ratio Test (SPRT). The proposed IDS was tested with datasets from the Defence Advanced Research Projects Agency (DARPA) for intrusion detection and compared against other techniques. However, this algorithm requires datasets with the same features as the environment, which can be a problem due to the fast technological changes that render the DARPA dataset obsolete. The attacks used to test the SDN environment include DDoS, Neptune, smurf, ipsweep, and port sweep, but not all of them are targeted specifically at SDN. The SPRT parameters were set manually, and further testing under different parameter combinations is necessary to improve performance. The main problem with this paper is its inability to accurately identify DDoS attacks in the SDN controller using an acceptable threshold. Additionally, the proposed method is ineffective for detecting DDoS attacks against a host, as it generates false positives due to differences in attack rules. This shows that the proposed method is ineffective for expected flows over distributed environments. In [69], the authors presented a novel system called HFS-LGBM IDS for SDN attack detection. The HFS model combines the benefits of correlation-based feature selection and Random Forest Recursive Feature Elimination. The NSL-KDD dataset and Mininet were used to evaluate and test the system. However, the NSL-KDD dataset was found to be outdated and not representative of real-world network traffic, which negatively impacted the accuracy of the system. Moreover, the system only considered eight features as significant, making it unable to accurately predict the flows. Integrating the SDN controller and the IDS resulted in a high volume of required flows to check traffic, causing overload on the controller, and posing another challenge that needs to be addressed.

In [58], the researchers presented the OpenFlowSIA security system in the SDN context. The proposed system utilises an SVM classifier and Idle-timeout Adjustment (IA) algorithms to secure the controller and OpenFlow switches from DDoS attacks. The IDS consists of five modules: Flow Collector, Feature Extractor, SVM, Policy Enforcement, and IA Algorithm. The system collects traffic from the flow tables of OpenFlow switches, processes it to extract features, classifies the traffic using the SVM based on the protocol type, and ultimately determines if the packet is normal or malignant using the Policy Enforcement and IA algorithm. The CAIDA datasets were used for training and testing. However, the proposed system was found to consume a significant amount of CPU usage and memory, leading to congestion, and affecting response time. The system lacks a feature selection method to cover the behaviours of DDoS attacks, and the authors did not use evaluation metrics to assess the detection rate or accuracy of the model.

Table 5. IDSs-based non-Deep Learning.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

Ref and authors	Approach	Number of Features	Detected Attack	Controller	Dataset	Accuracy/False Alarm
[39]	SVM	6	DDoS	SDN controller	Simulation datasets	Detection Rate of 95.24% and False Alarm of 1.26%
[40]	ASVM	5	DDoS	POX controller	Simulation datasets	Accuracy 97%
[43]	SVM	29	DoS	OpenDaylight	KDD99 dataset	Accuracy of 97.63%
[44]	SMO, Naive Bayes, and J48	41	DDoS	SDN controller	NSL dataset	Accuracy of 99.4% for SMO
[36]	The analysed algorithms were SVM, Naïve Bayes, KNN, ASVM, HMM, K-means, Random Forest, Decision tables, K-medoids, and fuzzy	5	DDoS	POX controller	NSL-KDD dataset	The highest accuracy is 81.5% for the J48 algorithm
[45]	Graph Theory	Statistics Flow	DoS	POX controller	CAIDA dataset	The precision of 0.84, Recall of 0.78, and F-score of 0.81
[46]	C4.5	52	DoS and Prob attacks	SDN controller	1999 Darpa dataset	The precision of 0.989 and recall of 0.964
[47]	SOM	4	DDoS	NOX controller	1999 Darpa dataset	Detection Rate of 98.61% and False Alarm of 0.59%
[48]	SVM algorithm with a kernel function	8	IPsweep, Probe, and DDoS	SDN controller	1998 DARPA and 2000 DARPA	Accuracy Rate of 94.81% and False Alarm of 0.11%
[49]	J48-tree	41	known and unknown attacks	NetFPGA10G	KDD'99	Detection Rate of 91.81% and False Alarm of 0.55%
[51]	Naive Bayes, KNN, K-means and K-mediods	52	DDoS	POX controller	privet dataset	Detection Rates of 94%, 90%, 86% and 88%

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

[24]	NN	6	DOS, U2R, R2L, and Probes	SDN controller	NSL-KDD	Detection Rate of 97.4%
[52]	KNN	23	DoS	SDN controller	NSL-KDD	Accuracy Rate of 91.27% and 0.99% of precision
[53]	Random Forest	5	DOS	NOX controller	Specific dataset	Detection Rate of 90% and False Alarm of 70%
[54]	TRW-CB and Rate Limiting	4	DoS, Probe, DDoS, U2R, and R2L	NOX controller	KDD Cup 1999 dataset	Accuracy Rate of 96.3%
[55]	K-means++, AdaBoost, and Random Forest (RF)	6	DDoS, DoS, U2R, and R2L	SDN controller	KDD Cup 1999 dataset	Accuracy Rate of 84%
[56]	J48 algorithm	41	DoS, Probe, U2R, and R2L	SDN controller	NSL-KDD	Detection Rate of 90.9% for DoS
[57]	Bernoulli random	2	DDoS	SDN controller	DARP 1999 dataset	None
[58]	Idle-timeout Adjustment (IA) & SVM	The six Basic features	DDoS	OpenFlow controller	CAIDA	None

5.2.3. Deep Learning-based Approaches

Deep Learning (DL) is an approach that belongs to the neural network algorithm, where the nodes can be considered as devices built for defence. DL algorithms are a modern update to artificial neural networks that utilise swarming and reasonable computation. DL allows an algorithmic program to learn an illustration of data with varying levels of generalisation. These methods are applied to visual perception, object detection, network intrusion, and many other domains. A DL algorithmic program can be trained as either supervised or unsupervised. Deep learning algorithms include Convolutional Neural Networks (CNN) and Artificial Neural Networks (ANN), which are generally trained and supervised. CNN is currently the benchmark model for computer vision purposes [59]. Here are some examples of DL approaches in the context of intrusion detection systems (IDS):

1. Artificial Neural Networks (ANN): ANNs are a type of deep learning-based approach that is inspired by the structure and function of the human brain. ANNs consist of interconnected nodes (neurons) organized in layers, including input, hidden, and output layers. They are designed to learn and recognize patterns in data by adjusting the weights of the connections between neurons. ANNs have been widely used in various applications, including intrusion detection systems, where they can be employed for anomaly detection or signature-based detection [60].

2. Convolutional Neural Networks (CNNs): CNN is a specific type of deep learning-based approach that is particularly effective for processing grid-like data, such as images or network traffic matrices. CNNs are built on a grid-like structure and use convolution operations to scan and analyse local patterns in data. They consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers. CNNs have been successfully applied to network traffic analysis for detection purposes, where they can identify patterns or features related to known attack signatures or detect anomalous traffic patterns [61].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

3. Recurrent Neural Networks (RNNs): RNNs are another type of deep learning-based approach designed to process sequential data, such as time series data or text data. RNNs have a unique feature called "long short-term memory" (LSTM) that allows them to maintain information about previous time steps, making them suitable for detecting patterns, trends, and anomalies in time-series data. In intrusion detection systems, RNNs and LSTMs can be used for anomaly detection and signature-based detection, especially in cases where network traffic or user activities have a sequential relationship [62].

4. Gated Recurrent Units (GRUs): GRUs are a variant of Recurrent Neural Networks (RNNs) that were introduced to address some limitations of the traditional RNN architecture. GRUs are a type of deep learning-based approach that also processes sequential data, such as time series data or text data, making them suitable for detecting patterns, trends, and anomalies in such data [61-63].

B. Sarra and G. Mohamed in [60] proposed a DL approach in the SDN context to identify the DDoS and DoS attacks between the controller and end-user devices. Using the Relu and Softmax functions, the traffic will be classified as malignant or normal inside the SDN controller. The CICIDS2017 public dataset has been used in the experiment in the training and testing stages. The authors used the logarithm function, which uses the Min/Max scalar technique to normalise the extracted features for the classification step. The proposed model used five basic features in the classification, and these features will be extracted for each packet at the SDN controller in real-time. The model has achieved an accuracy of 99.6%. However, the limited number of the extracted features will be caused by the low efficiency in the detection. These features have been extracted from the packet's basic header information and are not enough to cover the attack's behaviours; therefore, the attacker can easily avoid the IDS by modifying the header packet to seem like regular traffic. Also, the authors did not consider the controller's bottleneck, where the controller at the extensive network will not be able to do the functionalities efficiently. Hence, the system needs a lightweight method to process the packet efficiently.

In this paper [61], the researchers presented a DL-based method for detecting DDoS attacks in the SDN environment, specifically focusing on multi-vector attacks. The system examines each packet at the SDN controller and extracts features from them, classifying them as normal or malicious. The proposed model utilizes the POX controller. The authors collected the dataset from a home wireless network using tools like tcpdump and hping3 to generate DDoS traffic. They divided the collected traffic into training and testing datasets. The proposed system extracts 68 features from each packet for classification, achieving an accuracy of 95.65%. However, the extraction of these features requires significant memory and processing time, causing a bottleneck in the controller. Additionally, many of these extracted features are not relevant to DDoS attack practices.

In another study [62], the authors employed a deep neural network approach to recognize DDoS attacks in SDN networks. They utilised the NSL-KDD public dataset for training and testing. The proposed model used six basic features for classification, extracting them in real time for each flow at the SDN controller. The model achieved an accuracy of 75.75%. However, the limited number of extracted features resulted in low detection accuracy during the detection stage. These features were extracted from basic statistics information and were insufficient to cover the behaviours of attacks, making it easy for attackers to evade the intrusion detection system (IDS). The authors suggested that the controller periodically requests flow table entries from OpenFlow switches and that each flow in the switches should be classified every time. This approach adds complexity and overhead to the controller, requiring an efficient and lightweight method to handle traffic processing.

In [63], the authors presented a DL model based by using the Gated Recurrent Unit Recurrent Neural Network (GRU-RNN) to classify traffic as either a DDoS attack or not in SDN. They used the NSL-KDD public dataset for training and testing, extracting six basic features for each flow at the SDN controller in real time. The POX controller was utilized in this model, achieving a detection rate of 89%. However, the limited number of extracted features resulted in low efficiency in detection, as they did not cover the full range of DDoS attack behaviours. Additionally, the authors did not address the bottleneck issue of the controller in handling large networks, requiring a lightweight traffic processing method.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

In [64], the authors proposed a hybrid system called SD-Reg, which combines convolutional neural network (CNN) and SD-Reg to detect SDN attacks. They used the InSDN dataset for training and testing the CNN classifier. The model achieved high accuracies of 99.28% and 98.92% for binary and multi-class classifications, respectively. However, relying solely on the InSDN dataset for training might not cover all high-risk attacks and could lead to poor test results' validity. Additionally, the approach did not address the issue of CPU consumption and generated overhead on the controller when merging CNN and SD-Reg.

In [65], the authors proposed a flow-based anomaly detection approach for the OpenFlow controller using deep neural network (DNN) algorithms. The model, called GRU-LSTM DNN, used 52 features extracted in real-time from each flow at the SDN controller using the ANOVA F-TEST method. The NSL-KDD public dataset was used for training and testing. The model achieved an accuracy of 87% with a false alarm rate of 0.76%. However, the issue of the controller's bottleneck was not addressed.

Y. Hande and A. Muddana [66] addressed the development of Anomaly-based Network Intrusion Detection Systems (NIDS) in SDN networks. They utilized a CNN model to identify various types of attacks in SDN network traffic, with the sniffer IDS module feeding the detector. However, the paper lacks details regarding the selection and extraction of features, as well as the critical components' design for the detector. The authors did not explain why they chose to have two layers in the CNN model or why manually selected features were used instead of an unsupervised CNN approach. The sensing module did not provide information about the classes it had or how it detected unknown attacks. The paper also mentioned setting a boundary value for the IDS based on a threshold to describe the correct behaviour of network traffic. The authors suggested installing the system in the controller, which poses a significant challenge due to the processing and overhead it would introduce. Lastly, the CNN algorithm was not suitable for large-scale networks due to its high computational complexity.

Table 6. IDSs-based Deep Learning.

Ref & Author	Approach	Features	Attack Detected	Controller	Dataset	Limitations	Accuracy/False Alarm
[60]	Relu and Softmax function	5	DDoS and DoS	ONOS	CICIDS2017	Must reduce the bottleneck controller and low number of used features	99.6
[62]	Deep NN	6	DoS	OpenFlow Controller	KDD	The accuracy must be increased. Must reduce the bottleneck controller	75.75%
[61]	SAE	TCP, UDP features	DDoS	POX	Traffic Dataset	Must reduce the bottleneck controller	95.65%
[63]	Recurrent Neural Network.	6	DoS	POX	NSL-KDD	Model optimisation is required in	89%

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

						feature selection and extraction	
[65]	GRU-LSTM DNN	52	Prop, U2R, R2L and DoS	POX	NSL-KDD	Model optimisation required	87%
[64]	CNN	48 and 9	Prop, U2R, R2L, DoS, etc.	OpenFlow	InSDN	Model optimisation required	98.92%

6. Evaluation Metrics

IDS recognise normal and abnormal traffic by observing the system's overall input. The detection algorithms are used for this classification of input traffic on the network, and these algorithms are also responsible for sounding the alarm. Some of these alarms are explained below [13]:

- False Positive (K): This alarm occurs when regular traffic is wrongly classified as attack traffic.
- False Negative (S): This alarm occurs when attack traffic is wrongly classified as regular traffic.
- True Positive (N): This occurs when attack traffic is classified correctly as attack traffic.
- True Negative (P): This occurs when normal traffic is classified accurately as regular traffic.

There are two main groups for measuring security metrics. One is called basic metrics, and the other is known as evaluation metrics. In basic metrics, the identification of optimal IDS or the comparison of several IDS events. Moreover, Table 7 identifies the parameters related to performance in the confusion matrix [13] below are used to compute the metrics of accuracy, recall, precision, and F1 score [13].

Table 7. Confusion Matrix.

The Event	Abnormal	Normal
Abnormal	True Positive (N)	False Negative(S)
Normal	False Positive(K)	True Negative(P)

To define the accuracy rate of a machine learning algorithm, the percentage of the total amount of traffic is divided by correct predictions. Equation 1 was used for this calculation:

$$Accuracy = \frac{N+P}{N+P+K+S} X\ 100 \quad (1)$$

Precision (P), alternatively known as a false alarm, is used as a metric for evaluating the accuracy of catching attack traffic. This measure calculates the capacity to identify attack traffic correctly. Equation 2 was utilised to determine Precision.

$$Precision = \frac{N}{N+K} \quad (2)$$

Recall (R) assess the ratio of correctly recognised attacks out of the total attack traffic. Equation 3 has been used to calculate R, which provides an estimation of the rate of predicted attacks compared to the overall attack traffic.

$$Recal = \frac{P}{N+K} \quad (3)$$

The F1 score is used to evaluate the classifier employed to calculate the accuracy of a model. It includes both the Precision and Recall scores to provide a total evaluation. This metric calculates the number of correct predictions made by the model across the entire dataset and therefore determines the overall accuracy of the classifier. Equation 9 has been used to calculate F-Score.

$$\mathrm{F-Score} = \frac{2}{\frac{1}{P}+\frac{1}{R}} \quad (4)$$

7. Testing and Implementation Tools

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

Effective IDS becomes paramount in SDN environments with network attacks' increasing complexity and challenges. This section explores the different testing and implementation tools available for IDS in SDN networks.

- **Mininet Emulator:** Mininet is a popular open-source network emulator that enables the design of virtual SDN networks. Mininet allows researchers to assess IDS performance under managed conditions by simulating network topologies and traffic patterns. It provides an emulated environment to deploy IDS initial designs and assess their effectiveness in detecting known attacks [14]. Mininet is available online on [14]. However, the performance of the IDS measured inside Mininet is relative and dependent on the hosting machine's capabilities. Some factors that have a huge influence include the processor speed, the amount of RAM, the hard drive speed, and the network adapter's performance. The IDS measurements we get from Mininet might appear more or less efficient than in a non-virtualised environment. This is due to the fact that the virtual network is sharing the resources with the host machine, potentially creating a bottleneck, and affecting the accuracy of the IDS performance data.
- **Scapy:** Scapy, an effective packet generation tool, allows the development of customised packets to simulate network traffic, including regular and attack traffic. It enables cybersecurity researchers to test IDS in SDN networks by generating composed packets with different payloads and headers, imitative real-world attack scenarios. Scapy allows the validation of IDS functionalities and the evaluation of their capability to detect, respond to, and mitigate various network attacks [23]. Scapy is available online on [67].
- **Open vSwitch (OvS):** Open vSwitch is a popular open-source virtual switch that can simulate components of SDN architectures. OvS allows administrators to filter packets and enable traffic monitoring, making it a practical implementation tool for IDS in SDN networks. It allows researchers to design custom IDS modules that employ packet inspection methods, implementing real-time traffic analysis and detection approaches [68]. Open vSwitch is available online on [69].
- **Snort:** *Snort* is a well-established open-source IDS that can integrated with SDN networks. It presents various detection policies and abilities to recognise and prevent network attacks. By installing Snort with SDN controllers like OpenDaylight or POX, researchers can adjust Snort's management to network traffic flows and improve IDS performance in dynamic SDN environments [70]. *Snort* is available online on [71].
- **Bro/Zeek:** Bro, now known as Zeek, is a robust network security monitoring framework for SDN networks. It presents real-time traffic analysis abilities, supplying clear insights into network possibilities and facilitating IDS functionality. With its programmable scripting language, researchers can utilise Zeek to catch specific malignant traffic, supplying more accurate and customised attack detection powers [72]. Zeek is available online on [73].
- **Slowloris:** Slowloris is an open-source tool that has been developed using Python programming language. Slowloris tool is available online on [74]. Slowloris is an HTTP DoS/DDoS attack that affects targeted servers and works as follows:

a) Creating extensive HTTP requests, causing vast traffic to the server.
b) Send packets at regular intervals, about every 15 seconds, to keep open connections continuously.
c) The connection is only closed if the server launches such an action. In the event of connection closure by the server, a new connection is set to maintain the continuous process.

This continuous pattern of activity stresses the server's thread pool, rendering it unable to respond to requests from other users. By using the Slowloris attack, targeted servers face the effects of the attacker flooding the HTTP connections to block the server's functionality.

- **Wireshark:** Wireshark is a software network protocol analyser commonly used in network analysis, troubleshooting, and collecting packets and flows from connected devices [75]. Wireshark is an open-source packet sniffer tool. It authorises administrators and researchers to catch and analyse network traffic in real-time. This software's advanced abilities and comprehensive characteristics make it a useful tool in academic and professional settings. It is available online on [76].

Table 8. The summary of the testing and implementation tools.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

Tool	Description	Link
Mininet Emulator	Mininet is an open-source network emulator that enables the design of virtual SDN networks. It's useful for assessing IDS performance under managed conditions.	[14]
Scapy	Scapy is a packet generation tool that helps develop customised packets to simulate network traffic, aiding in testing IDS in SDN networks.	[16]
Open vSwitch (OVS)	OvS is an open-source virtual switch that simulates components of SDN architectures. It's handy for designing custom IDS modules.	[18]
Snort	Snort is a popular open-source IDS that can be integrated with SDN networks to provide various detection policies and abilities.	[20]
Bro/Zeek	Zeek (formerly known as Bro) is a robust network security monitoring framework for SDN networks that offers real-time traffic analysis facilities.	[22]
Slowloris	Slowloris is an open-source tool developed using Python, known for its implementation in HTTP DoS/DDoS attacks, which can stress a server's thread pool.	[23]
Wireshark	Wireshark is an open-source network protocol analyser used for network analysis, troubleshooting, and collecting packets from connected devices.	[25]

8. The Findings and Research Gaps

Due to the flexibility and innovation of SDN over networking environments, there are uncovered gaps that integrate with future technologies to offer more exciting research. The present review study is the first to make an effort to solve the problem of attack impact over different SDN layers. In the future, it is still required to analyse and measure how attack-defined techniques affect SDN research challenges' security components and requirements. The following points are what we identified as emerging directions and not covered gaps related to the security of SDN architecture:

8.1. An effective way to process network traffic

Most researchers propose models that need to monitor traffic in real-time, so this process needs to analyse each packet or flow passed through it as normal or malignant. Therefore, the processing requires time, memory, and more CPU usage to collect, process, and classify the traffic. This process will cause overload/congestion at the controller and the switches. Therefore, getting the best choice between efficiency and overload needs to be considered by the researchers.

8.2. Distribute the processing stages over OpenFlow devices

All the researchers use of_stats_request to claim the flow tables of all the switches to extract the selected features [80]. This method brings all the switches' flow tables without sorting according to time, size window, etc. This way needs to be processed by the controller to sort and extract the selected features. Some of these features do not take directly from the flow tables as such features need complex computational operations to be calculated [47]. Therefore, the OpenFlow channel will be busy and congested when installing an IDS. Additionally, due to ofp_flow_stats returning all the switches' flow tables, the messages' size will significantly affect the channel and increase the processing time. Consequently, this method is inefficient with massive data processing and causes more overhead and congestion on the controller. Therefore, using efficient approaches in distributing the processing stages needs the attention of the researchers.

8.3. Detect Slow DDoS/DoS Attack

In this survey, it was noticed that the majority of researchers focus on heavy-rate DDoS/DoS attacks that send huge irregular packets to the target. However, there needs to be more interest in detecting the slow DDoS and DoS attacks that target web services, which seem like regular traffic. These attacks consume the resources and stop the target service from such attacks with time. Therefore, this issue needs to be covered by the researchers.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

8.4. Employing outdated datasets for training and testing the proposed models

It is clear from the survey that many researchers have employed out-of-date datasets to test and evaluate their proposed techniques [77]. Selecting the datasets is significant for an efficient and accurate detection system. Most datasets found publicly need to be realistic and include most attacks, negatively affecting accuracy and performance [26]. The main reasons for the deficiency of these data are due to privacy and legal matters. Most of these datasets are old and need to include the updated behavers. Moreover, such datasets have a high number of duplicate records. Consequently, such systems will be involved to obtain low accuracy and poor performance [26]. The available public datasets were collected from the traditional networks, not the SDN network. Such datasets include some features not accessible in the SDN networks [77]. These datasets are published online, including KDD'99 NSL-KDD, CICIDS2017, ISCX2012, and Kyoto.

8.5. Getting high accuracy with limited raw features by using DL/non-DL approaches

The essential issues with non-DL/DL approaches are extraction and feature selection. The currently proposed systems suffer from many used features in classification or using basic features. Using a large number of features causes overload and latency in the network. Using a small number of basic features will not give accurate detection attacks as they do not cover the behaviours of attacks. Therefore, getting high accuracy with limited raw features using DL/non-DL approaches needs more attention from the researchers.

8.6. Test the proposed models with the real network environments

Most recent research reviewed in this survey has taken advantage of simulation, emulation, and virtualised environments for testing. However, such experiments involve using a simple and smaller network. This leads to the outcomes needing to be more accurately validated for the methodologies since they applied real data. On the other hand, using real hardware can achieve better results. Henceforward, testing and validating security applications based on SDN using real environments is another future research direction.

8.7. The scalability of using more than one controller

Researchers in the cybersecurity area tried to propose solutions using individual controllers in a network topology. Nevertheless, SDN has a centralised nature that is critical to several attacks. Using a single controller will cause a bottleneck during the attack and the incoming packet process. Hence, the distributed controllers' design significantly improves reliability, load distribution, and processing power consumption. However, the implementation of IDS with multi-controllers is under research and needs to be covered by the researchers. The new research direction involves designing new IDS flexible with the SDN controllers' distribution approach to minimise overhead issues and balance the traffic between more controllers. However, when using more than one controller, the network suffers from interoperability, where several controllers have different routing algorithms and policies. This issue needs to be addressed where no further research considers such issues.

9. Conclusion

The conclusion derived from all the literature is that the proposed Intrusion Detection Systems (IDSs) play a specific role in identifying and preventing harmful activities in the SDN environment. SDN technology enables programmability, flexibility, manageability, dynamism, and intelligence in the current network architecture by utilizing a centralized controller, a single viewpoint, and reduced configurations to save costs. In contrast, traditional networks rely on switches and routers with their operating systems, offering a limited range of configuration options. The SDN concept manages network services and maintains control using lower-level functional abstraction. It allows for the extraction of multiple features or specific focus on certain types of attacks. The analysis and collection of network flow data are more effective compared to traditional networks. This approach also ensures user privacy, which is increasingly critical in today's world.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

Recent research has concluded that combining SDN technology with non-DL approaches for attack detection produces superior results for its intended purpose. However, it also has limitations, as explained in this review. The transition from conventional networks to SDNs significantly alters network architecture and management. Nevertheless, current solutions introduce new challenges that cause significant overhead, particularly when implementing such IDS on large-scale networks. Researchers need to prioritize intrusion detection in SDN networks to create a secure and integrated system that can contribute to the development of a robust model capable of protecting against various threats. As technology advances, the research community must dedicate attention to devising effective detection approaches to address the current research gaps in SDN networks.

Data Statement: No new data were created during this study.

References

1. P. R. and K. Ramasamy, "Software defined network based self-diagnosing faulty node detection scheme for surveillance applications," *Computer Communications,* vol. 152, pp. 333-337, 2020.
2. G. Yuki, N. Bryan, W. K. Seah and Y. Takahashi, "Queueing analysis of software defined network with realistic OpenFlow–based switch model," *Computer Networks,* vol. 164, 2019.
3. J. A. Herrera and J. E. Camargo, "A survey on machine learning applications for software defined network security," in *In International Conference on Applied Cryptography and Network Security*, Cham, 2019.
4. S. Arash, M. A. Kaafar, R. Buyya and S. Jha, "Software-defined network (SDN) data plane security: issues, solutions, and future directions," *Handbook of Computer Networks and Cyber Security,* pp. 341-387, 2020.
5. A. Shaghaghi, S. S. Kanhere, M. A. Kaafar and S. Jha, "Gwardar: Towards protecting a software-defined network from malicious network operating systems," in *In 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, 2018.
6. 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," *International Journal of Communication Systems,* vol. 32, no. 4, p. e3875, 2019.
7. Hauser, Frederik, 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," *Journal of Network and Computer Applications* , vol. 212, p. 103561, 2023.
8. Bhowmik, C. Dey and T. Gayen, "Traffic aware dynamic load distribution in the Data Plane of SDN using Genetic Algorithm: A case study on NSF network," *Pervasive and Mobile Computing,* vol. 88, p. 101723, 2022.
9. Bhuiyan, Z. Ahmed, S. Islam, M. M. Islam, A. 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, 2023.
10. Sarmiento, D. Espinel, A. Lebre, L. Nussbaum and A. Chari, "Decentralized SDN control plane for a distributed cloud-edge infrastructure: A survey," *IEEE Communications Surveys & Tutorials* , vol. 23, no. 1, pp. 256-281, 2021.
11. M. Pradhan, C. K. Nayak and S. K. Pradhan, "Intrusion detection system (IDS) and their types," *Securing the Internet of Things: Concepts, Methodologies, Tools, and Applications - IGI Global,* pp. 481-497, 2020.
12. R. Swami, M. Dave and V. Ranga, "Software-defined networking-based ddos defense mechanisms," *ACM Computing Surveys (CSUR),* vol. 52, no. 2, pp. 1-36, 2019.
13. 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.
14. K. R. Santhana, J. E. Golden, R. Y. Harold, 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 Networks,* vol. 26, no. 2, pp. 1275-1289, 2020.
15. M. Weizhi, E. W. Tischhauser, Q. Wang, Y. Wang and J. Han, "When intrusion detection meets blockchain technology: a review," *IEEE Access,* vol. 6, pp. 10179-10188, 2018.
16. Ali, T. Emad, Y.-W. Chong and S. Manickam, "Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review.," *Applied Sciences,* vol. 13, no. 5, p. 3183, 2023.
17. Karnani, Suruchi, 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 and Applications,* pp. 1-54., 2023.
18. I. Ahmad, S. Namal, M. Ylianttila and A. Gurtov, "Security in software defined networks: A survey," *IEEE Communications Surveys & Tutorials,* vol. 17, no. 4, pp. 2317-2346, 2015.
19. Lent, D. M. Brandão, M. P. Novaes, L. F. Carvalho, J. Lloret, J. J. 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.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

20. Alotaibi, F. M. and V. G. Vassilakis, "Toward an SDN-Based Web Application Firewall: Defending against SQL Injection Attacks.," *Future Internet,* vol. 15, no. 5, p. 170, 2023.
21. D. He, J. Dai, X. Liu, S. Zhu, S. Chan and M. Guizani, "Adversarial Attacks for Intrusion Detection Based on Bus Traffic," *IEEE Network,* vol. 36, no. 4, pp. 203 - 209, 2022.
22. Lin, H.T. and Wang, P.C., 2023. TCAM-based packet classification for many-field rules of SDNs. Computer Communications, 203, pp.89-98.
23. Santos, Reneilson, D. Souza, W. Santo, A. Ribeiro and E. Moreno, "Machine learning algorithms to detect DDoS attacks in SDN.," *Concurrency and Computation: Practice and Experience,* vol. 32, no. 16, 2020.
24. A. Abubakar and B. Pranggono, "Machine learning based intrusion detection system for software defined networks," in *In 2017 seventh international conference on emerging security technologies (EST)*, 2017.
25. M. Awad and R. Khanna, "Support vector machines for classification- Efficient Learning Machines," *Springer,* pp. 39-66, 2015.
26. T. Divya and S. Agarwal, "A survey on Data Mining approaches for Healthcare," *International Journal of Bio-Science and Bio-Technology,* vol. 5, no. 2, pp. 241-266, 2013.
27. W. Xindong, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda and G. J. McLachlan, "op 10 algorithms in data mining," *Knowledge and information systems,* vol. 14, no. 1, pp. 1-37, 2008.
28. A. Javed and L. Seemab, "Handling intrusion and DDoS attacks in Software Defined Networks using machine learning techniques," in *In 2014 National Software Engineering Conference*, 2014.
29. R. Irina, "An empirical study of the naive Bayes classifier," in *In IJCAI 2001 workshop on empirical methods in artificial intelligence*, 2001.
30. J. S. D. and H. P. Channe, "Comparative study of K-NN, naive Bayes and decision tree classification techniques," *International Journal of Science and Research (IJSR),* vol. 5, no. 1, pp. 1842-1845, 2016.
31. S. L. Ting, W. H. Ip and A. H. Tsang, "Is Naive Bayes a good classifier for document classification.," *nternational Journal of Software Engineering and Its Applications,* vol. 5, no. 3, pp. 37-46, 2011.
32. M. Pal, "Random forest classifier for remote sensing classification," *International journal of remote sensing,* vol. 26, no. 1, pp. 217-222, 2005.
33. J. Kim, J. Kim, H. Kim, M. Shim and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics,* vol. 9, no. 6, p. 916, 2020.
34. S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore and M. Plakal, "CNN architectures for large-scale audio classification," in *In 2017 ieee international conference on acoustics, speech and signal processing (icassp)*, 2017.
35. X.-B. Jin, N.-X. Yang, X.-Y. Wang, Y.-T. Bai, T.-L. Su and J.-L. Kong, "Hybrid deep learning predictor for smart agriculture sensing based on empirical mode decomposition and gated recurrent unit group model," *Sensors,* vol. 20, no. 5, p. 1334, 2020.
36. M. S. Elsayed, N.-A. Le-Khac, S. Dev and A. D. Jurcut, "Machine-learning techniques for detecting attacks in SDN," in *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, Dalian, China, 2019.
37. 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.
38. D. Li, C. Yu, Q. Zhou and J. Yu, "Using SVM to Detect DDoS Attack in SDN Network," in *In IOP Conference Series: Materials Science and Engineering*, 2018.
39. Y. Jin, X. Cheng, Z. Jian, L. Feng and S. Ling, "A DDoS attack detection method based on SVM in software defined network," *Security and Communication Networks,* 2018.
40. 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)," *Journal of Computer Networks and Communications,* 2019.
41. D. Silva, A. Santos, J. A. Wickboldt, L. Z. Granville and A. Schaeffer-Filho, "ATLANTIC: A framework for anomaly traffic detection, classification, and mitigation in SDN," in *In NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*, 2016.
42. A. Song, P. Younghee, G. Keyur, K. Youngsoo, B. Kalgi and G. Kunal, "Machine-learning based threat-aware system in software defined networks," in *In 2017 26th international conference on computer communication and networks (ICCCN)*, 2017.
43. 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 *In 2016 IEEE 13th International Conference on e-Business Engineering (ICEBE)*, 2016.
44. A. Alshamrani, A. Chowdhary, S. Pisharody, D. Lu and D. Huang, "A defense system for defeating DDoS attacks in SDN based networks," in *In Proceedings of the 15th ACM International Symposium on Mobility Management and Wireless Access*, 2017.
45. A. AlEroud and I. Alsmadi, "Identifying cyber-attacks on software defined networks: An inference-based intrusion detection approach," *Journal of Network and Computer Applications,* vol. 80, pp. 152-164, 2017.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

46. L. A, P. Dinh, H. Le and T. N. Cuong, "Flexible network-based intrusion detection and prevention system on software-defined networks," in *In 2015 International Conference on Advanced Computing and Applications (ACOMP)*, 2015.
47. R. Braga, E. Mota and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in *In IEEE Local Computer Network Conference*, 2010.
48. K. RT, S. T. Selvi and K. Govindarajan, "DDoS detection and analysis in SDN-based environment using support vector machine classifier," in *In 2014 Sixth International Conference on Advanced Computing (ICoAC)*, 2014.
49. N. T. Van, H. Bao and T. N. Thinh, "An anomaly-based intrusion detection architecture integrated on openflow switch," in *In Proceedings of the 6th International Conference on Communication and Network Security*, 2016.
50. A. Jankowski and M. Amanowicz, "On efficiency of selected machine learning algorithms for intrusion detection in software defined networks," *International Journal of Electronics and Telecommunications,* vol. 62, no. 3, pp. 247-252, 2016.
51. 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 *In 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016.
52. L. Majd and L. Toker, "Minimizing false positive rate for DoS attack detection: A hybrid SDN-based approach," *CT Express,* vol. 6, no. 2, pp. 125-127, 2020.
53. L. Jiaqi, Z. Zhifeng, R. Li and H. Zhang, "Ai-based two-stage intrusion detection for software defined iot networks," *IEEE Internet of Things Journal,* vol. 6, no. 2, pp. 2093-2102, 2018.
54. M. S. Akbar, J. Khalid and S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking.," in *In International workshop on recent advances in intrusion detection*, Berlin, Heidelberg, 2011.
55. L. Jiaqi, Z. Zhao and R. Li, "Machine learning-based IDS for software-defined 5G network," *ET Networks,* vol. 7, no. 2, pp. 53-60, 2017.
56. R. Sathya and R. Thangarajan, "Efficient anomaly detection and mitigation in software defined networking environment," in *In 2015 2nd international conference on electronics and communication systems (ICECS)*, 2015.
57. 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 *In 2016 IEEE International Conference on Communications (ICC)*, 2016.
58. T. V. Phan, T. V. Toan, D. V. Tuyen, T. T. Huong and N. H. Thanh, "Openflowsia: An optimized protection scheme for software-defined networks from flooding attacks," in *In 2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, 2016.
59. S. Scott-Hayward, S. Natarajan and S. Sezer., "A survey of security in software defined networks," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 1, pp. 623-654, 2015.
60. S. BOUKRIA and M. GUERROUMI, "Intrusion detection system for SDN network using deep learning approach," in *In 2019 International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS)*, 2019.
61. Q. Niyaz, W. Sun and A. Y. Javaid, "A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN)," *Cornell University,* vol. arXiv 2016." arXiv preprint arXiv:1611.07400, 2016.
62. 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 *In 2016 international conference on wireless networks and mobile communications (WINCOM)*, 2016.
63. 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 *In 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, 2018.
64. 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," *Journal of Network and Computer Applications,* vol. 191, 2021.
65. S. K. Dey and M. M. Rahman, "Flow based anomaly detection in software defined networking: A deep learning approach with feature selection method," in *In 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCTP)*, 2018.
66. Y. Hande and A. Muddana, "Intrusion Detection System Using Deep Learning for Software Defined Networks (SDN)," in *In 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2019.
67. Scapy, "Scapy," 2023. [Online]. Available: https://scapy.net/. [Accessed 21 07 2019].
68. H. Ma, X. Luo and D. Xu, "Intelligent queue management of open vSwitch in multi-tenant data center," *Future Generation Computer Systems,* vol. 144, pp. 50-62, 2023.
69. vSwitch, "Open vSwitch," 2023. [Online]. Available: https://www.openvswitch.org/download/. [Accessed 20 12 2021].
70. Alsharabi, Naif, M. Alqunun and B. A. H. Murshed, "Detecting Unusual Activities in Local Network Using Snort and Wireshark Tools," *Journal of Advances in Information Technology,* vol. 14, no. 4, 2023.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2023 doi:10.20944/preprints202312.1449.v1

71. Snort, "Snort," 2023. [Online]. Available: https://www.snort.org/downloads. [Accessed 12 06 2021].
72. Tiwari, Asheesh, S. Saraswat, U. Dixit and S. Pandey, "Refinements In Zeek Intrusion Detection System," in *8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2022.
73. Zeek, "Zeek," 2023. [Online]. Available: https://docs.zeek.org/en/master/install.html. [Accessed 14 September 2023].
74. G. Yaltirakli, "Github," 2015. [Online]. Available: https://github.com/gkbrk/slowloris. [Accessed 12 08 2022].
75. Jawaharan, Ragaharini, P. M. Mohan, T. Das and M. Gurusamy, "Empirical evaluation of sdn controllers using mininet/wireshark and comparison with cbench," in *27th international conference on computer communication and networks (ICCCN)*, 2018 .
76. Wireshark, "Wireshark," 2022. [Online]. Available: https://www.wireshark.org/. [Accessed 12 July 2022].
77. Kumar, Gulshan and H. Alqahtani, "Machine Learning Techniques for Intrusion Detection Systems in SDN-Recent Advances, Challenges and Future Directions," *CMES-Computer Modeling in Engineering & Sciences* , vol. 134, no. 1, 2023.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.