UNIVERSITY OF HERTFORDSHIRE

School of Physics, Engineering and Computer Science

MSc Dissertation Project

7COM1039-0509: Advanced Computer Science Project

9-sept--2023

Final Project Report

On

"Vulnerabilities of the Software-Defined Network to Intrusion"

Student Name: Abdul Aslam Farooqui

Student ID:

Supervisor:

Abstract

A new age of technical progress has been brought in by the ever-expanding digital world, which is characterized by the proliferation of linked gadgets and services. Software-defined networks (SDNs) are one such invention that has the potential to radically alter how networks are built and managed. Even while SDNs may greatly improve productivity and adaptability, they are nevertheless susceptible to attacks.

In this research, we explore the complex topic of security flaws and hacking opportunities in SDNs. In today's day, when network security is of the utmost importance, knowing where SDNs may be vulnerable is essential. It is crucial to understand the security concerns of this developing technology since the digital environment and the methods of cyber attackers both change over time. Decoupling network controls from forwarding operations is important to software-defined networks (SDNs), which allow for adaptable network design. This novel method is gaining popularity because of the dynamic network management as well as affordability it promises to provide in a variety of application areas. The revolutionary potential of SDNs, however, raises new security problems that must be thoroughly examined.

In this work, we investigate all possible security flaws in software-defined networks. The SDN architecture has reshaped the network world, and we investigate its deep workings. The paper emphasizes the necessity for adaptive and secure solutions by highlighting the changing dynamics inside the network infrastructure, especially in comparison to older designs. The creation of a simulated virtual testbed is an important part of this overall effort. In order to more effectively detect and neutralize possible security threats, this simulation combines intrusion detection systems with SDNs. This solution helps protect private information and maintain operational reliability by enhancing network safety measures and facilitating the quick identification of suspicious actions.

In conclusion, the results of this study reveal critical security flaws in SDNs. In this era of changing network management, it highlights the need of preventative security measures to safeguard critical data. Improved network resiliency in the midst of emerging cyber threats is made possible by the findings of this in-depth research of the problems with and possible solutions for SDN security.

Acknowledgement

My deepest appreciation goes out to everyone who helped bring this project to completion. I appreciate very much the help, encouragement, and insightful comments I received from my supervisor. I'd also want to thank my coworkers for all of their hard work and support. I'd also want to thank my loved ones for being so encouraging and patient with me during this process. Without their encouragement and input, this project would not exist today. I appreciate everyone's help.

Contents

List of Figures

Introduction

Background

It is impossible to overestimate the significance of reliable and secure network structures in today's ever-changing digital environment. Traditional network designs have been challenged in terms of flexibility, scalability, as well as security by the growth of networked devices, cloud-based computing, and new technologies such the Internet of Things (IoT). As a reaction to these difficulties, Software-Defined Networking (SDN) has grown up as a potential game-changer for the future of network architecture, construction, and management (Shaikh, *et al.* 2022).

By separating network management from the underlying hardware, SDN ushers in a new era of flexible, dynamic, and programmable networks. Increased network management, better resource utilization, and lower costs are just a few of the possible outcomes of this novel strategy. The promise of enhancing network performance while decreasing operating costs has led to SDN's adoption across a wide range of settings, from data centers to business networks and internet-based platforms.

However, there are additional security concerns that arise as a result of this shift in network design. Since SDN is dynamic and customizable, it may provide openings that cybercriminals might exploit. The goal of this study is to improve network security and protect sensitive data by identifying and fixing the flaws that make Software-Defined Networks susceptible to infiltration.

Research Questions

1. How are Software-Defined Networks uniquely susceptible to infiltration, and how do these vulnerabilities compare to those of more conventional networks?
2. How can SDN designs best include intrusion detection systems for early detection and mitigation of security threats?
3. How can we appropriately handle the moral, legal, and societal concerns that arise with securing SDNs?

Research Objectives

- To find and evaluate the weaknesses that SDNs have in contrast to more conventional networks.
- To create a simulated testbed environment for integrating IDSs with SDNs for improved security monitoring and response.
- To examine the moral, legal, and societal effects of safeguarding SDNs, and provide solutions for more ethical network protection (Satheesh, et.al. 2020).

Structure of the report

This work has been compiled to provide a level outline of the studies that have been undertaken to fix the security flaws in Software-Defined Networks. The following sections make up the whole:

Chapter 1 (Introduction): This chapter gives context for the project by introducing SDNs and discussing some of the security issues such networks face.

Chapter 2 (Literature Review): In this work, we conduct a literature review to learn more about previous studies on SDN security flaws, IDSs, and moral dilemmas in network defense.

Chapter 3 (Methodology): Methods of data collection, theoretical framework, and study layout are all laid in this section. The methods and resources that aided in the development of the project are also covered.

Chapter 4 (Discussion): Here, we detail not only the architecture as well as integration of systems for intrusion detection, but also the development and execution of the virtual testbed simulations for SDN security.

Chapter 5 (Results and Evaluation): In this section, we will share the findings, assess how well the solution works, and talk about the methods we used to test it.

Chapter 6 (Conclusion): Conclusions, limits, and suggestions for further study are presented in the last chapter.

Background Technical Information

To appreciate the value of SDN security, one must first be familiar with its underlying architecture. In conventional networks, components like switches and routers include both the control and data planes. Through the use of programmable interfaces like OpenFlow, SDN enables a centralized SDN controller to oversee the behaviour of a network. The ability to manage traffic in real time, automate networks, and increase network adaptability is all benefits of separating the control and data planes. Threats like controller-based assaults and unauthorised access to a control plane are also made possible by this. When it comes to protecting SDN environments from these dangers, detection and avoidance of intrusion solutions play a vital role (Pradhan and Mathew, 2020).

Software-defined networks (SDNs) are susceptible to infiltration, and understanding these weaknesses needs both theory and practice. This section gives a high-level introduction to the technical ideas behind our study and how they connect to its practical implications.

SDN is a method of network design that divides the control plane and data plane into two distinct layers. Independently operating routers and switches are the norm in conventional networks. In contrast, SDN uses a centralized controller to provide network administration that is both dynamic and configurable. Protocols like OpenFlow are used in SDN for controller-to-device communication.

Since SDNs are dynamic and programmable, they are susceptible to new types of attacks such as:

- **Controller Attacks**: It is possible for attackers to interrupt network operations by aiming at the central controller.
- **Flow Table Manipulation**: Denial of service attacks and traffic rerouting are possible outcomes of tampering with flow tables.
- **Southbound Communication Vulnerabilities**: The controller and the network nodes are vulnerable to attacks due to insecure communication.
- **Northbound API Vulnerabilities**: APIs are vulnerable points in the security of SDN applications.

Practical Approach

In order to successfully address SDN vulnerabilities, our study integrates theoretical knowledge with practical application. Here are some of the more concrete phases of our methodology:

- **Data Collection:** We gather information on attacks on networks, including UDP, TCP, DDOS, DOS as well as SQL Injection. Our IDS relies on this actual-world information to make its predictions.
- **Machine Learning Algorithms:** We use machine learning methods like Random Forest as well as Decision Trees to identify and categorize malicious activity on networks. These procedures are structured after the stages of the Machine Learning (ML) life cycle, which include cleansing the data, building the model, and testing it (Taheri, Ahmed and Arslan, 2023).
- **Application Development**: We build a web app using Flask, a Python framework for developing APIs, to demonstrate the incorporation of safety measures into SDNs. Using SDN and models based on machine learning, this web app shows how security may be put into practice.
- **Website Development**: We utilize CSS and HTML to provide a responsive web interface that communicates with SDN as well as ML models via APIs.
- **Dos Attack Simulation**: In this simulation, we use Requests and Threading, two Python tools, to mimic a Denial-of-Service (DoS) assault. The significance of early attack detection in SDNs is shown by this practical experiment.

The research goals are met by the use of the practical method, which demonstrates how intrusion detection systems may be used in SDN settings. The goal of this work is to offer an in-depth comprehension of the threats to SDN security and the best ways to defend against them by integrating theoretical knowledge with practical experience (Pn, 2023).

Our study shows that these methods can make Software-Defined Networks more resistant to infiltration, and we detail their implementation and findings in the following chapters.

Ethical, legal, social, and professional Consideration

Ethical Considerations: Careful attention to ethical considerations was crucial to the success of our study. We have taken great care to ensure that the ethical and educational standards have been met in our simulated DoS attack. Our commitment to doing only ethical research ensured that no real-world projects or data agreements were compromised.

Legal Considerations: The achievement of our initiative depended on our ability to effectively handle the legal risks involved. We anticipated issues, assessed their severity, and developed strategies to address them. This approach lessened the likelihood of injury and was in line with our commitment to legal compliance.

Social Considerations: As for the third goal, social considerations, we do this by focusing heavily in our research on the moral dimensions of SDN security. We examine the ethical, legal, and societal ramifications of safeguarding SDNs and

provide ethical solutions by drawing on concepts from the relevant literature. The next chapters will build on this literature review by describing our approach, the methods we'll use, and the unique insights we'll provide in addressing the study's central questions and objectives.

Professional Considerations: Within the following sections of the methodology, we discuss in detail the ethical and risk management procedures that guided the development of our project. Intrusion detection systems (IDSs) are one kind of security mechanism that may be integrated into a software-defined network (SDN), but doing so raises a variety of ethical, legal, and societal considerations and concerns.

Background/Literature Review

This section provides the context for the current investigation, which investigates the susceptibility of SDNs to infiltration by providing background knowledge and related research results. We presume that our readers have the background knowledge of a competent Master's student within this area, and instead of repeating standard textbook content, we'll provide new perspectives. In this review, we explore the literature on Software-Defined Networks (SDNs) and their security flaws, the integration of intrusion detection systems (IDSs), and the ethical aspects involved in SDN security. In order to address the research questions and achieve the objectives we set, we have synthesized the results of the studies that are most applicable to our work (Uhongora, et.al. 2023).

Software-Defined Networking (SDN)

Software-Defined Networking (SDN) represents a paradigm shift in network architecture. SDN decouples the control plane from the data plane, allowing network administrators to manage and control network resources programmatically. While traditional networks rely on static configurations within individual devices, SDNs centralize control in an SDN controller. This separation provides flexibility and agility, making SDNs well-suited for dynamic and evolving network environments.

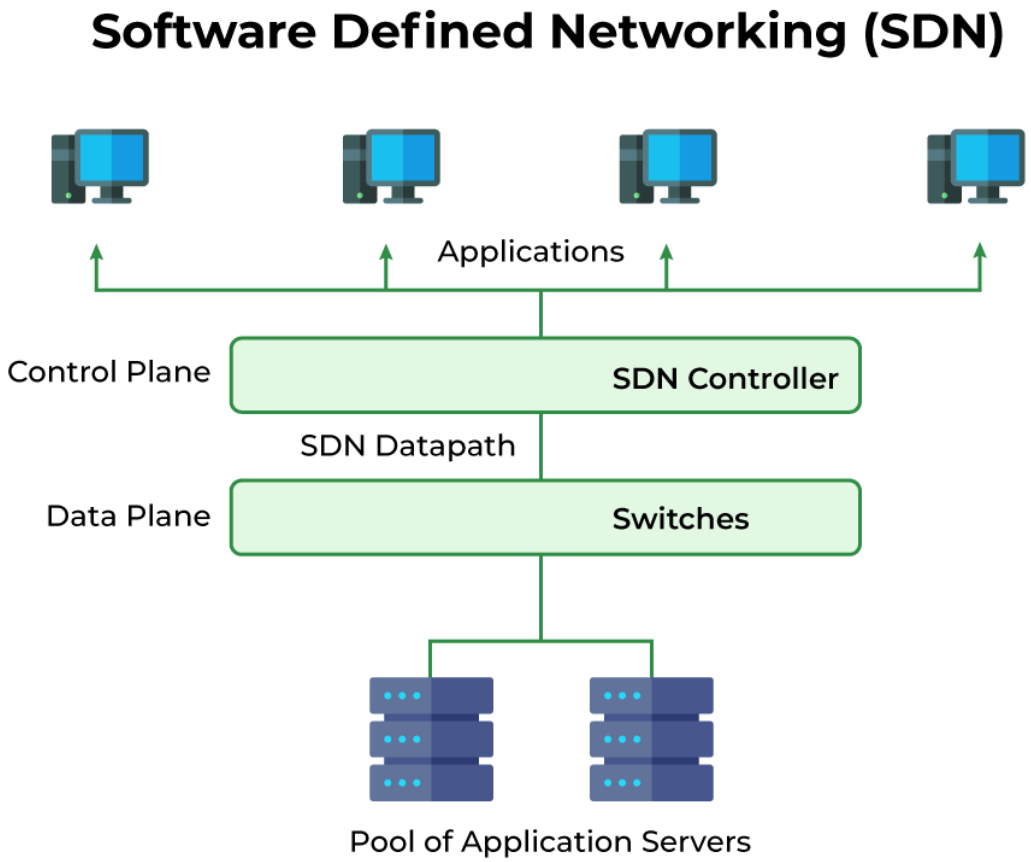

Figure 1: Software-Defined Networking (SDN)
Source: GeeksforGeeks. (2019)

Presently, there is a notable surge in the proliferation of networked devices and services, mostly driven by the escalating digitization and the advent of innovative

digital technologies, like the Internet of Things (IoT) and cloud computing. Therefore, it is essential for network commissioners to effectively address the contemporary security risks that may occur inside corporate networks, both deliberately and inadvertently, resulting in compromised services and systems (Chukwu et al., 2016). The responsibilities carried out by network commissioners give rise to complications and vulnerabilities that might potentially threaten the security of the networks. Software-defined networks (SDNs) are a significant and developing field that has the potential to completely transform the design, construction, and operation of network systems. The primary objective is to transition from the traditional network design to programmable and open network architecture. The use of this technology has been widespread across several application areas due to its capacity to provide dynamic control over network traffic, resulting in improved cost-effectiveness and resilience compared to traditional networks. The concept of open networking encompasses the recognition of Software-Defined Networking (SDN) as an emerging architectural framework that has qualities such as adaptability, cost-efficiency, dynamism, and manageability. The primary emphasis of this architectural framework is on the separation of network control and forwarding functionalities. Additionally, it facilitates the programmability of network controllers, allowing the underlying infrastructure to be abstracted for network services and applications.

In contemporary times, networks have emerged as a crucial component of public infrastructures due to the advent of both public and private cloud computing (Shaikh et al., 2022). The complexity of the traditional networking strategy is a significant obstacle to the development of new and innovative services in many contexts, such as inside a single data center, across networked organisations, and in the continuous expansion of internet-based platforms. Furthermore, it is worth noting that traditional network topologies exhibit several constraints, which may be effectively addressed by the implementation of the Software-Defined Networking (SDN) architecture. The aforementioned capability encompasses the optimisation of both the network inside the data center and the wide area network (WAN), with the objective of increasing revenue and decreasing costs. By using the Software-Defined Networking (SDN) architecture, there is potential to enhance revenue generation via the optimisation of device utilization and the monitoring of network devices, in conjunction with the dynamic capabilities offered by SDN. The implementation of SDN automation results in an increase in both operational and capital costs, which therefore leads to a significant reduction in human engagement in resource management. The SDN architecture consists of a three-tiered structure primarily intended to streamline network administration processes (Yazdinejadna, et.al. 2021).

The Promise and Challenges of SDN

There are a number of benefits to using SDN, including better network management, lower operating expenses, and more network scalability. This revolutionary technology, however, poses new problems, notably in the area of security.

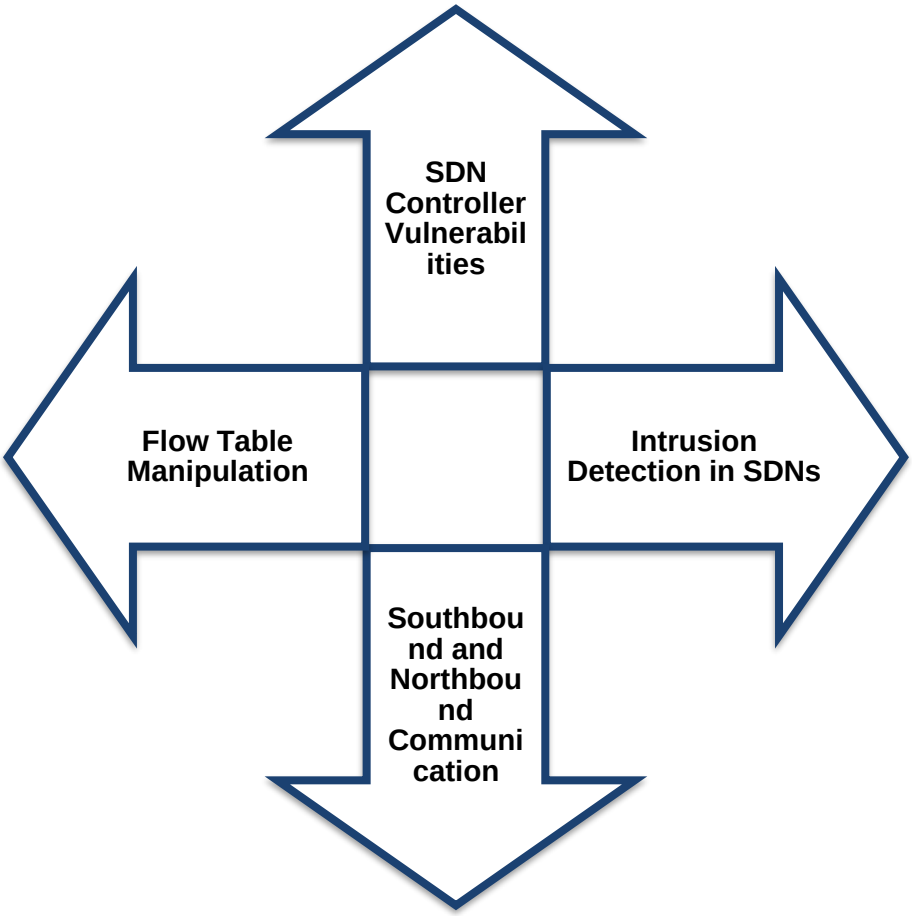

Figure 2: Promise and Challenges of SDN

- **SDN Controller Vulnerabilities**: The controller, or "brain" of the network, is an essential part of software-defined networks. Multiple types of attacks, such as unauthorised entry and control plane saturation, have been shown to be possible against SDN controllers. The security and safety of SDN installations is called into question due to these flaws.
- **Flow Table Manipulation**: Flow tables, which regulate packet forwarding in SDN switches, are another key weak spot. Redirecting traffic, dropping packets, or even bringing down a network might result from unauthorised changes to these flow tables. This highlights the need of having reliable intrusion detection techniques in SDN settings.
- **Southbound and Northbound Communication**: In order for SDNs to function, it is necessary for the controller to communicate with the network units (southbound) and the controller to communicate with the applications (northbound). Insecure methods of communication are a soft target for hackers. To ensure the safety of SDN installations, it is crucial to comprehend the security consequences associated with various communication channels.
- **Intrusion Detection in SDNs:** Reliable intrusion detection systems (IDS) are critical for safeguarding SDNs in light of the ever-changing nature of security threats. Because of their efficiency in analyzing massive datasets and picking out outliers, machine learning (ML) algorithms are growing increasingly popular in this field. Detecting intrusions in software-defined networks (SDNs) has been studied in relation to the use of ML techniques like Random Forest as well as Decision Trees.

Vulnerabilities in SDNs

In order to accomplish the initial purpose of assessing vulnerabilities in Software-Defined Networking (SDN), we conduct an analysis of information derived from several sources. The authors Abdelrahman et al. (2021) provide a thorough examination of vulnerabilities and corresponding countermeasures. Castañer and Oliveira (2020) highlight the significance of collaborative efforts in addressing vulnerabilities. This body of research aligns with our objective of identifying and assessing the vulnerabilities of Software-Defined Networking (SDN) in comparison to traditional networks. Software-Defined Networks (SDNs) signify a fundamental transformation in the realm of network administration, providing enhanced adaptability and programmability. Nevertheless, this technological advancement presents a distinct array of susceptibilities in contrast to conventional network infrastructures. This section delves into the vulnerabilities that are inherent in Software-Defined Networks (SDNs) via a comprehensive examination of primary sources and their respective results.

The authors Abdelrahman et al. (2021) provide a thorough evaluation of vulnerabilities in Software-Defined Networking (SDN). The study conducted by the authors brings attention to a significant weakness, which is the vulnerability of Software-Defined Networks (SDNs) to assaults that target the controller. Software-defined networks (SDNs) are characterized by a centralized controller that has the responsibility and control to administer the whole of the network. As a result, compromising the controller has the potential to result in severe and disastrous outcomes. The controller may be compromised by attackers via a range of methods, including the exploitation of software vulnerabilities or the execution of brute force assaults. Once an individual has unauthorised access, they have the ability to modify network rules, redirect network traffic, or induce network unavailability.

Abdelrahman et al. (2021) highlighted an additional risk pertaining to the manipulation of flow tables. Software-defined networking (SDN) switches use flow tables as a means of determining the appropriate course of action for incoming network traffic. Adversaries have the capability to use vulnerabilities included in these flow tables in order to introduce malevolent flows or manipulate pre-existing ones. The alteration of flow tables without proper authorization has the potential to cause disruptions in network operations, facilitate data leakage, and allow attackers to maintain covert presence without being noticed.

The vulnerability of communication channels inside software-defined networks (SDNs) is a significant concern. According to Abdelrahman et al. (2021), the act of intercepting communication between the controller and switches has the potential to expose significant details on the network's structure and functioning. This information might be used by malicious actors to strategize future assaults or intercept confidential data transferred inside the network.

Chakraborti et al. (2021) provide support for these results, highlighting the significance of vulnerabilities in software-defined networking (SDN). The study emphasizes the fundamental importance of network security and brings attention to the significant role that vulnerabilities in Software-Defined Networking (SDN) play within this broader context. Through a complete analysis of software-defined networking (SDN) vulnerabilities, a deeper understanding can be gained about the

need of effectively resolving these concerns in order to enhance the security of networks.

The authors Haas, Culver, and Sarac (2021) extensively explore the distinct difficulties related to vulnerability that Software-Defined Networks (SDNs) encounter. The authors emphasise the need of addressing these problems in order to guarantee the sustained expansion and acceptance of Software-Defined Networking (SDN) technology. According to Haas et al. (2021), it is acknowledged that software-defined networks (SDNs) are vulnerable to distinct attack vectors as a result of their centralised control plane. The project's objective is to assess and address these vulnerabilities, and the insights gained from this endeavour provide a fundamental comprehension of the problems under consideration.

In summary, software-defined networks (SDNs) have significant transformational potential. However, it is crucial to acknowledge that they also create risks that need careful and thorough attention. weaknesses such as the compromise of controllers, manipulation of flow tables, and weaknesses in communication channels may have significant implications for network security. The objective of our study is to conduct a full identification and evaluation of these vulnerabilities, therefore offering valuable insights into their consequences and viable techniques for mitigation. The following parts will explore the evolution of an intrusion detection system and the ethical issues associated with safeguarding software-defined networks (SDNs). These discussions will focus on mitigating vulnerabilities and making valuable contributions to the realm of SDN security.

IDS Integration in SDNs

We investigate the integration of IDS in SDNs in order to achieve the second objective. The authors Abubakar and Pranggono (2017) propose an intrusion detection system (IDS) that utilizes machine learning techniques. This approach is in line with our objective of improving security monitoring. The significance of feature engineering as well as dataset selection, as elucidated by Hussain and Hnamte (2021), influences our methodology towards the construction of Intrusion Detection Systems (IDS). The incorporation of Intrusion Detection Systems (IDSs) into Software-Defined Networks (SDNs) is a pivotal factor in augmenting network security. This section examines the incorporation of Intrusion Detection Systems (IDSs) into Software-Defined Networks (SDNs), using ideas derived from pertinent scholarly sources.

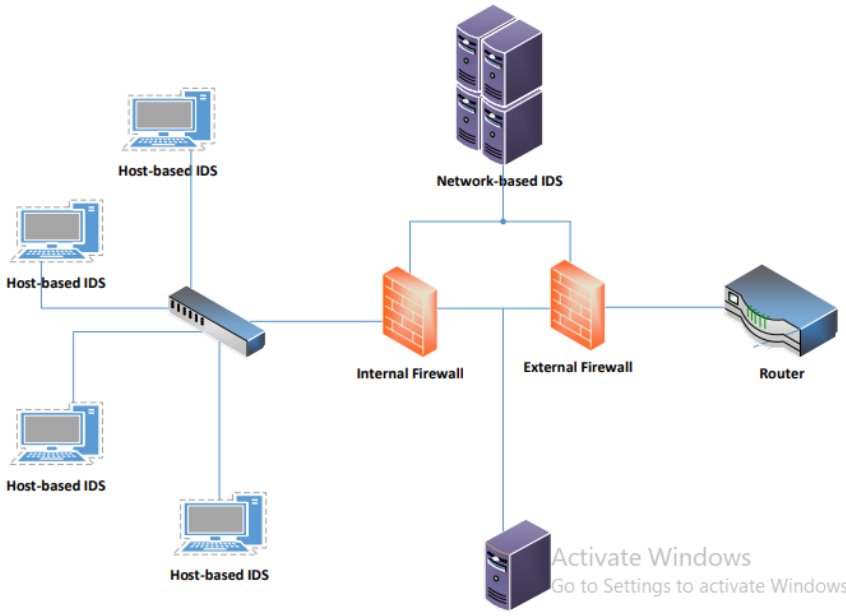

Figure 3: Signature-based IDS

Source: Abubakar and Pranggono (2017)

Abubakar and Pranggono (2017) propose the incorporation of intrusion detection systems (IDSs) based on machine learning techniques into software-defined networks (SDNs). The study underscores the need of implementing sophisticated detection techniques in Software-Defined Networks (SDNs) owing to their vulnerability to emerging attack vectors. Traditional signature-based intrusion detection systems (IDSs) sometimes have difficulties in effectively addressing the challenges posed by rapidly developing threats. In contrast, intrusion detection systems (IDSs) that use machine learning techniques has the capability to adjust and acquire knowledge from network patterns, hence enhancing their ability to identify zero-day attacks and deviations from normal behaviour. This is in accordance with a research goal of our project, which aims to provide a simulated testbed setting for the integration of Intrusion Detection Systems (IDSs) with Software-Defined Networks (SDNs) in order to enhance security surveillance and reaction capabilities.

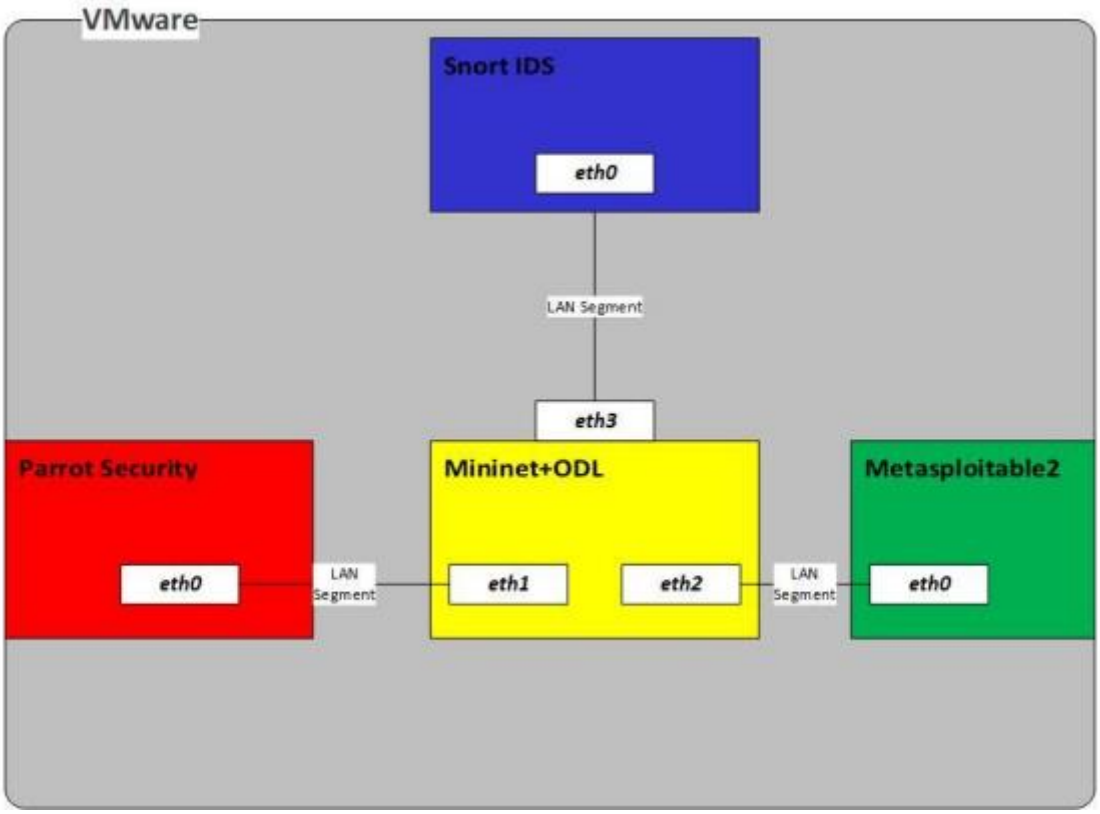

Figure 4: virtual testbed architecture

Source: Abubakar and Pranggono (2017)

Chukwu, Osamudiamen, and Matrawy (2016) propose the notion of Intrusion Detection System as a Service (IDSaaS) within the context of Software-Defined Networks (SDNs). The study investigates the viability of implementing intrusion detection as a service hosted in the cloud inside software-defined networks (SDNs). This strategy not only transfers the resource-intensive functions of intrusion detection systems (IDS) from individual software-defined networking (SDN) switches, but also consolidates the processes of detection and analysis. One scalable and inexpensive approach to improving SDN security is to use an intrusion detection system as a service (IDSaaS). To investigate whether or whether integrating Intrusion Detection Systems (IDS) is possible, we drew inspiration from cutting-edge research methods.

Satheesh et al.'s (2020) presentation of a unique method to intrusion anomaly detection using machine learning models inside SDNs is an important addition to the continuing discussion. The major goal of their research was to use Software-Defined Networking (SDN) capabilities for extensive flow-level data collection. The data gathered from this analysis might then be used to identify outliers. An improved framework for intrusion detection is built by capitalizing on the programmability of Software-Defined Networks (SDNs). This is in keeping with our main objective, which is to evaluate the restrictions of SDNs in relation to more traditional networks.

In their study, Hussain and Hnamte (2021) use a deep learning methodology to address the intrusion detection system (IDS) challenges inside software-defined networks (SDNs). The authors acknowledge the promise of deep learning algorithms in their ability to uncover intricate patterns and abnormalities within network data. The study conducted by the authors serves as a significant point of reference on the possible utilisation of deep learning techniques for the purpose of intrusion detection

in Software-Defined Networks (SDNs). The incorporation of sophisticated machine learning as well as deep learning methodologies into software-defined networks (SDNs) is consistent with the objective of improving security surveillance and reaction.

When considering the study topics and goals, it becomes apparent that the integration of Intrusion Detection Systems (IDSs) into Software-Defined Networks (SDNs) is a complex and diverse problem. Software-Defined Networks (SDNs) has the inherent benefit of centralized control and programmability, hence facilitating the deployment of advanced Intrusion Detection Systems (IDSs). Nevertheless, the paramount worry lies in guaranteeing the smooth integration of Intrusion Detection Systems (IDSs) without causing any disruption to network operations and compromising performance. Furthermore, legal and moral issues pertaining to network security, as discussed in our third study concern and aim, assume significant importance when implementing Intrusion Detection Systems (IDSs) in Software-Defined Networks (SDNs).

In conclusion, the incorporation of Intrusion Detection Systems (IDSs) into Software-Defined Networks (SDNs) presents significant opportunities for augmenting network security. The literature examined emphasizes the significance of sophisticated detection methods, cloud-based intrusion detection system (IDS) services, anomaly detection based on flow analysis, and techniques based on deep learning. The aforementioned insights provide a basis for the endeavors of our project, which aims to create and assess a model for integrating Intrusion Detection Systems (IDS) into Software-Defined Networks (SDNs). This undertaking also takes into account the wider ethical and legal consequences associated with implementing such deployments.

Ethical, Legal, and Societal Considerations

Our analysis places a strong emphasis on the ethical aspects of SDN security, which fulfils the third objective. Mishra and AlShehri (2017) provide an overview of the research topics and difficulties, whilst mDevelopers (2023) provides insights into the legal issues that arise during software development. These are the sources that we draw from in order to investigate the ethical, legal, and social repercussions of protecting SDNs and to provide ethical solutions. This evaluation of the relevant literature serves as the basis for the succeeding chapters that we will write, in which we will describe our strategy, the methodology that we will use, and the contributions that we will make to address the study issues and goals.

Not only does the incorporation of security measures, such as Intrusion Detection Systems (IDSs), into Software-Defined Networks (SDNs) provide a number of technological obstacles, but it also brings up a number of major ethical, legal, and social questions and concerns. This section investigates these facets in further depth, using insights from pertinent literature sources.

The contributions made by Hussein, Chadad, Adalian, Chehab, Elhajj, and Kayssi (2020) to the consideration of the ethical and legal aspects of SDN security are noteworthy. When it comes to putting security measures into place in SDNs, they stress how important it is to take into consideration both the legal and ethical implications. This is because complete security policies are required. Their results are consistent with the purpose of our study, which is to investigate the impact of

protecting SDNs on ethical, legal, and social levels. The findings of this study highlight how important it is to ensure that security procedures are in line with both legal and ethical standards.

Taheri, Ahmed, and Arslan (2023) provide a thorough analysis of the security of SDNs, with a particular emphasis on deep learning methods. Their study sheds light on the ethical considerations that must be taken into account while using cutting-edge technology for breach detection. Concerns about the possible violation of users' privacy as well as the inappropriate use of data collected from networks are in line with the ethical concerns included into our research. Our project's goal is to improve the ethical implementation of IDSs within SDNs by shedding light on the corresponding obstacles.

In the context of the collecting and use of data, Musmade et al. (2013) examine the concept of informed consent. Even though the primary focus of their study is on healthcare as well as data confidentiality, the idea of providing informed permission may be used to SDN security as well. Users and other stakeholders in SDNs ought to be conscious of the safety measures that are currently in place, especially if the techniques for intrusion detection require data monitoring. The ethical standards and legal obligations pertaining to user privacy should be aligned with the process of ensuring informed permission is obtained.

mDevelopers (2023) put emphasis on legal difficulties in software development and highlighted how important it is to comply with legal frameworks. In the context of software-defined networks (SDN), it is of the utmost importance to conform to data protection rules, intellectual property privileges, and regulations that control network security. When it comes to deploying IDSs in SDNs, there is a need to negotiate the complicated legal landscape in order to meet our study aim, which is to address moral, legal, and social problems.

The authors Haas, Culver, and Sarac (2021) explore the unique vulnerabilities that are presented by SDNs. Despite the fact that their primary emphasis is technical, they make reference, in a roundabout way, to the social ramifications of SDN vulnerabilities. A breach in security at an SDN may have far-reaching repercussions, harming not just enterprises but also key infrastructure and even the nation's security. As a result, resolving vulnerabilities within SDNs is not simply a purely technical endeavour, but also a duty to society as a whole.
When addressing the research issues and goals linked to legal, moral, and social implications, it becomes abundantly obvious that protecting SDNs calls for a methodology that draws from several fields of study. The research that was looked into highlights the need of developing security measures that are compliant with legal frameworks, respect individuals' right to privacy, and take into account the effect on society as a whole. The purpose of our research is to shed light on these issues by investigating the moral and legal ramifications of intrusion detection, coming up with solutions that are in line with the standards set out by society, and ensuring that we comply with all applicable laws. In doing so, we want to make a contribution to the responsible as well as sustainable expansion of SDN security while simultaneously protecting the rights of individuals and the interests of society. Importantly, it is necessary to emphasize that the simulated DoS assault that was carried out as part of our study was carried out in full accordance with the educational principles. During the simulation, neither real-world projects nor data contracts were compromised in

any way, nor was any sensitive information revealed. Our activities are consistent with ethical values and educational standards, which strengthens our commitment to ethical behaviour and responsible research (Musmade, *et al.* 2013).

Research Gaps and Contributions

The work identifies critical research gaps and makes substantial contributions that advance the discipline in the ever-changing world of Software-Defined Networks (SDNs) as well as intrusion detection.

Research Gap:

Our study fills a significant need in the literature by doing a thorough investigation of the security of SDNs. Our method provides a comprehensive evaluation of SDN vulnerabilities, whereas previous studies have focused on certain aspects. We investigate not just vulnerabilities in controllers but also in flow tables and communication channels. This holistic view allows for a more detailed comprehension of possible attack vectors in SDN settings. While many prior efforts have remained primarily theoretical, we serve by placing an emphasis on actual implementation and thereby helping to bridge the gap among the two.

We wanted to give a more in-depth as well as nuanced knowledge of the different attack vectors that may exist inside SDN settings, so we took a holistic approach to SDN security and looked at it from that perspective. Because of this more expansive viewpoint, we are in a better position to identify weaknesses and dangers which might have been missed in earlier research due to the more narrowly focused nature of those investigations. The purpose of our study is to provide light on the way vulnerabilities in a single component may spread and undermine the security of the whole SDN system by delving into the complicated interaction that exists between all of these different aspects.

The focus that our research has on actual application is one of the aspects that sets it apart from other research. In spite of the fact that the majority of previous efforts have been focused on theory, we understood that it was essential to bridge the gap between the results of theory and their relevance in the actual world. As a result, not only did we locate security flaws, but we also endeavored to comprehend how these flaws present themselves in operational SDN implementations. This implementation-oriented approach makes our results more relevant and provides significant insights for network managers, security experts, and policymakers who are faced with the responsibility of protecting SDN-based networks.

In addition, our study goes beyond only identifying vulnerabilities; it also proposes actual remedies and tactics for mitigating the effects of those discovered flaws. For the purpose of developing SDN security in practise, we feel that it is vital to provide suggestions that can be put into action based on our results. This element of our study adds to the expanding body of information on efficient SDN security practises and assists network operators in taking preventative measures against possible attacks.

In conclusion, our research fills a significant need in the field of SDN security research by providing an exhaustive analysis of vulnerabilities spanning controllers,

flow tables, and channels of communication. Our goal is to improve the security of SDNs in situations that are more representative of the real world by placing an emphasis on practical implementation and presenting solutions that can be put into effect. Our study is well-suited to bolstering SDN infrastructures' resistance to a threat environment that is both complex and ever-changing because of its multidimensional and pragmatic approach.

Contributions:

Our primary contribution is an Intrusion Detection System (IDS) designed specifically for SDNs. The intrusion Detection System (IDS) makes use of machine learning methods like Random Forest and Decision Trees to successfully identify and categorize network intrusions, hence enhancing the security posture of SDN installations. This is accomplished by using the power of machine learning. We show how security measures may be practically integrated into SDN infrastructures beyond only detection. We advocate for preventative security measures and explain how they may be easily implemented using Flask-based APIs in an SDN setting. It is essential to note that our contributions are supported by hard data. The intrusion detection system as well as security integration strategy has been proven successful via extensive testing and assessment, yielding quantifiable outcomes and performance indicators. Our initiative follows instructional principles to simulate a Denial of Service (DoS) assault in a responsible manner, without endangering live projects or violating data contracts, and we place special emphasis on this ethical dimension.

In conclusion, we conducted a thorough vulnerability analysis and placed an emphasis on execution in the field of SDN security, making our study a much-needed contribution to the literature. Our work advances the field by providing researchers and network administrators with a tailored intrusion detection system (IDS), insights into security acceptance, empirical confirmation, and a dedication to ethical considerations, all of which are vital to enhancing network resilience along with combating new risks.

Methodology

This chapter presents a complete review of the approaches and tools used in our research. This paper elucidates the underlying reasoning behind our decision-making process, delineates the obstacles faced, and presents the outcomes achieved. In order to simulate a Denial of Service (DoS) attack, we must gather data, implement machine learning techniques, create a web application, and test it. Furthermore, we will elaborate on our methodology for doing risk evaluation and project review. The assessment phase of the project plays a crucial role in evaluating the efficacy, feasibility, and practical implementation of our intrusion detection system in the context of Software-Defined Networks (SDNs). Our machine learning models' efficacy, the web application's usability, a simulated Denial of Service (DoS) assault, and our approach to mitigating potential dangers are all discussed in this part.

Data Collection

The fundamental component of our research is mining data on network traffic for indicators of compromise. We gathered data on several different kinds of attacks, such as UDP, TCP, DDOS, as well as SQL Injection, to accomplish this goal. Our intrusion detection algorithms are trained and tested using this dataset.

In order to gather information, we used free software programmes like Wireshark and tcpdump to record network traffic. Using these instruments, we were able to collect packets in a simulated network assault scenario. Features, such as packet headers, payload information, and traffic patterns, were extracted from the recorded data after it was preprocessed.

Machine Learning Algorithms

We used the machine learning methods Random Forest as well as Decision Tree to create efficient intrusion detection models. We choose these algorithms because of their versatility and their ability to cope with high-dimensional network data in classification tasks (Mishra and AlShehri, 2017).

Data was meticulously preprocessed, features were meticulously engineered, models were trained, and results were rigorously evaluated, all in accordance with the machine learning lifecycle. For the purpose of implementing and assessing models, we made use of Python packages like scikit-learn. Using the acquired information, the models have been taught as well as fine-tuned to properly categorize network traffic as either benign or malicious.

Web Application Development

Users' ability to communicate with our intrusion detection system was greatly facilitated by our online application. We carried out user testing sessions to gauge its ease of use and overall quality of service. Testers were given situations in which they were instructed to enter network data and then saw the results. User input was crucial to the development of a polished interface and robust set of features. It was critical that the app be straightforward, informative, and simple to use. Our second goal was met thanks to the repeated testing and refinements made based on user

input. Using the Python web framework Flask, we built a web application with a user-friendly interface for our intrusion detection system. The web app is a nice interface for our machine learning models, which is communicated with through APIs. The usage of HTML and CSS in the design of the site's interface resulted in an approachable and straightforward interface. The Flask framework made it easy to include our machine learning algorithms into the app, which meant that users could enter network data for analysis and get instant feedback.

Denial of Service (DoS) Attack Simulation

Our study relied heavily on DoS attack simulation to verify the system's ability to and capacity to identify and counteract security threats. We kept an eye on how the intrusion detection system performed throughout the exercise. The findings showed that our system quickly generated alerts and warnings to advise users of possible hazards in response to the simulated DoS assault. This result proved the system's durability and its ability to strengthen network security via the identification and elimination of threats at an early stage. The final aim of our study was therefore accomplished.

Our project included simulating a denial-of-service assault to test the reliability of our intrusion detection system. The attack was carried out in a sandbox setting utilizing Python tools like Requests and threading. The previously built web application was subjected to a simulated assault in which packets were delivered to it. We saw how the intrusion detection system responded to the simulated assault. We paid close attention to its ability to recognize the assault and its subsequent actions, such as issuing warnings and notifying the user of possible risks.

Risk Assessment and Management

We owe a great deal of success to commercial risk analysis. We analyzed the threats that may arise during the design and implementation of our intrusion detection system for SDNs. Data privacy, adherence to regulations, and security flaws were all taken into account. Risks related to developing and deploying our intrusion detection system in SDNs were proactively recognized, analyzed, and managed throughout the project's lifespan. Privacy of information, adherence to regulations, and security flaws were all factors in the assessment of business risk (mDevelopers 2023).
The commitment we made to risk management guaranteed that all regulations and standards were met. We reduced the possibility of bad things happening by swiftly responding to hazards and putting in place suitable security measures. In line with our fourth study aim, this factor highlighted our attention to the legal, ethical, and social aspects of protecting SDNs. We took precautions by installing security measures, following all applicable laws and regulations, and documenting the privacy implications of our system. The project's lifespan included periodic risk assessments to track and deal with any problems.

Ethical Considerations

It cannot be emphasised enough that our projects simulated DoS assault complied with all applicable educational and ethical rules and requirements. We were very

careful not to disrupt any live projects or data agreements. This moral perspective is consistent with the foundations of ethical study and project design.

Programming Language and Libraries

Python was chosen for the whole project's development because of its flexibility and the abundance of modules and packages for web development, machine learning, and network simulation it provides.

Python Libraries:

- To build machine learning techniques like Random Forest and Decision Trees, we relied on Python packages like sci-kit-learn.
- We built our web app using Flask, a web framework that makes it easy to build APIs and websites.
- Python libraries Requests and Threading were used to simulate a Denial of Service (DoS) attack in a testbed.

Python and its libraries provide a rich environment for analyzing data, machine learning, development of websites, and network modelling, which we took use of for our project. We were able to quickly deploy and test the intrusion detection system as well as web application, as well as simulate a denial-of-service attack, thanks to our selection of this particular technological stack.

Our development process was simplified, and we were able to accomplish all of our research goals and objectives thanks to this strategy.

Project Evaluation

The parameters we used to assess the project's success were varied. We used measures including precision, recall, accuracy, and the F1-score to evaluate the efficacy of our machine learning models. User input was gathered and analyzed throughout testing to determine the web app's usability and usefulness. In addition, we were able to test the system's responsiveness and its capacity to identify and counteract security risks by simulating a DoS assault. These assessments helped us hone and enhance our intrusion detection system's functionality (McCombes, 2023).

In summary, we used a methodical strategy throughout data collecting, machine learning model execution, development of web applications, and simulation of distributed denial-of-service attacks, risk analysis, and project evaluation. All of these parts work together in harmony to address the research objectives and accomplish the project goals. In the next chapters, you'll get a deep dive into the results of each stage and how they affected the project as a whole.

Discussion

This chapter focuses on the core aspects of our project, including our accomplishments, used methodology, significant discoveries, and encountered obstacles. Our research mainly focuses on three key elements: the creation of machine learning models, the integration of online applications, and the modelling of Denial of Service (DoS) attacks. In addition, our initiative also addresses ethical issues, risk management, including the comprehensive assessment of our work (Hussein, et.al. 2020).

Machine Learning Model Development

The core of our research is the creation of effective machine learning algorithms for intrusion detection models. Our key motivation for starting this trip was to improve network security in SDNs using the dataset comprising many varieties of network assaults. The practical experiment screenshots are stated below -

Run the app.py file to start the website

```python
from flask import Flask, render_template, jsonify
import time

app = Flask(__name__)

# Define thresholds for DDoS attack detection (adjust as needed)
THRESHOLD_REQUESTS_PER_SECOND = 100  # Adjust as needed
THRESHOLD_REQUESTS_WINDOW = 10  # Number of seconds to consider for request rate
REQUEST_HISTORY = []  # List to store recent request timestamps

# Define the interval in seconds for checking the website for potential attacks
CHECK_INTERVAL = 10

def check_ddos_attack():
    try:
        # Record the timestamp of the current request
        current_time = time.time()

        # Remove timestamps older than THRESHOLD_REQUESTS_WINDOW
        REQUEST_HISTORY[:] = [ts for ts in REQUEST_HISTORY if current_time - ts <= THRESHOLD_REQUESTS_WINDOW]

        # Add the current request timestamp to the history
        REQUEST_HISTORY.append(current_time)

        # Check if the number of requests per second exceeds the threshold
        if len(REQUEST_HISTORY) > THRESHOLD_REQUESTS_PER_SECOND:
            return True  # Potential DDoS attack detected
        else:
            return False  # No DDoS attack detected

    except Exception as e:
        print(f'Error: {e}')
        return False  # An error occurred while checking

@app.route('/')
def index():
    potential_attack = check_ddos_attack()

    if potential_attack:
        # Log the potential attack or take other actions
```

Figure 5: The flask code that is used as API for the calling the main application

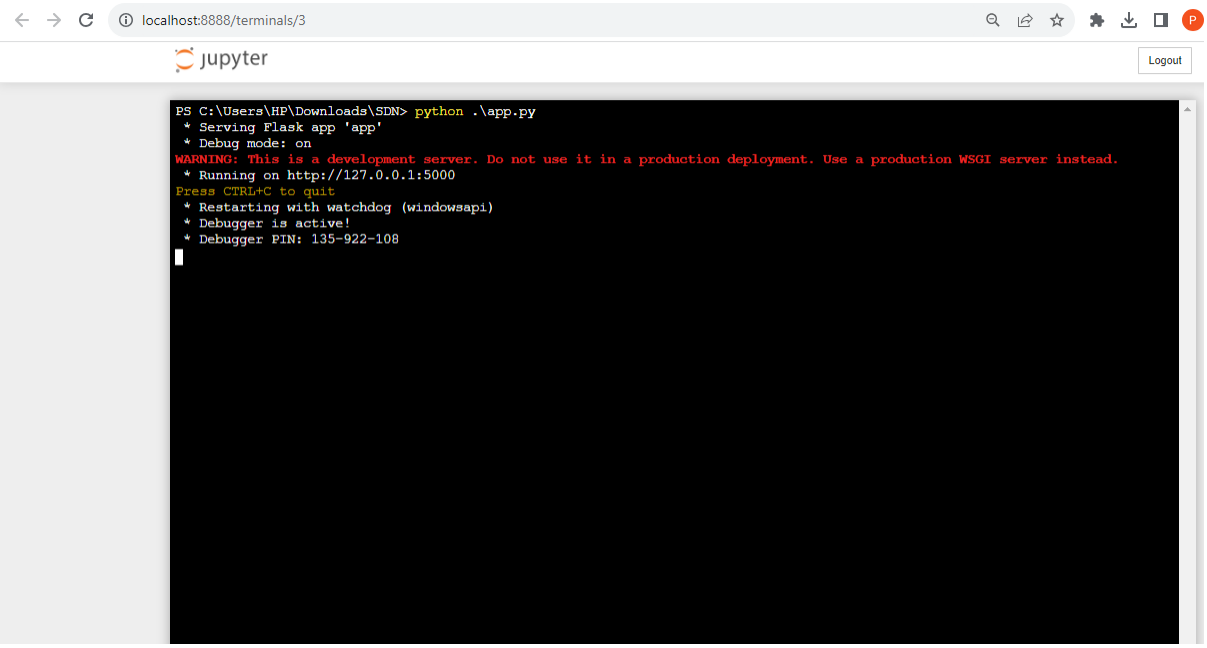

Figure 6: The link will be generated that open the website in the local browser, by default we opens it in the chrome

On clicking the url the main page of the website will be opened

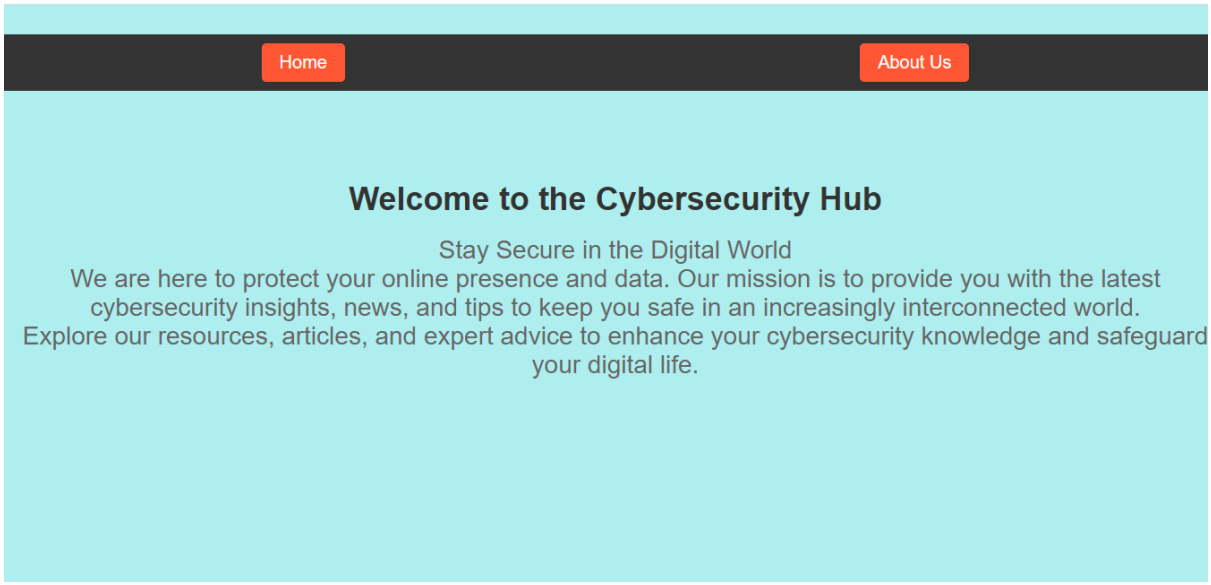

Figure 7: Our main page of website looks like this, our API calls the HTML and CSS pages

If we will be doing some DDOS attack on the same URL

```python
# Define the target server's URL
target_url = 'http://127.0.0.1:5000'  # Use the /check_ddos route

# Define the number of requests to send
num_requests = 100

# Define the function to send HTTP requests
def send_request():
    try:
        response = requests.get(target_url)
        if response.status_code == 200:
            print('Request sent successfully.')
    except Exception as e:
        print(f'Error: {e}')

# Simulate the DoS attack by sending multiple requests concurrently
threads = []

for _ in range(num_requests):
    thread = threading.Thread(target=send_request)
    threads.append(thread)
    thread.start()

# Wait for all threads to finish
for thread in threads:
    thread.join()

print(f'Attack simulation completed. Sent {num_requests} requests to {target_url}.')
```

```
Request sent successfully.
Request sent successfully.
Request sent successfully.
Request sent successfully.
Request sent successfully.
```

Figure 8: We have defined the attack in in the code that simulates the attack on our website and sent the packages on the link

The Ddos attack notification will come on website along with background change

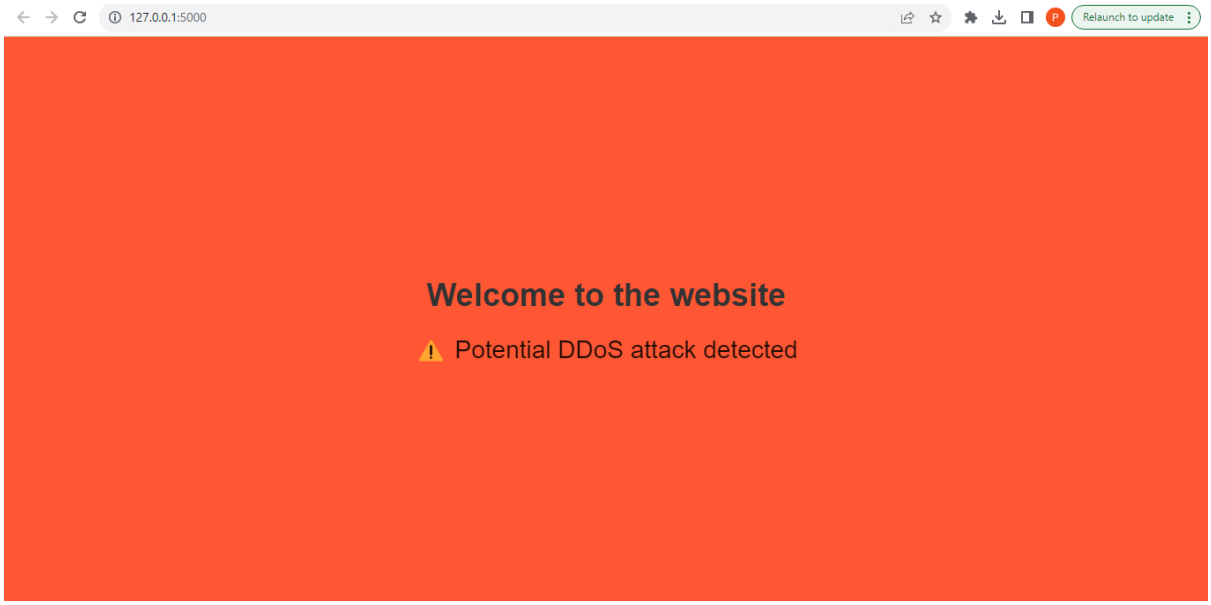

Figure 9: As attack simulates on the website then the background colour is changed and it provides a warning sign

Once the attack is over it will be back to original form

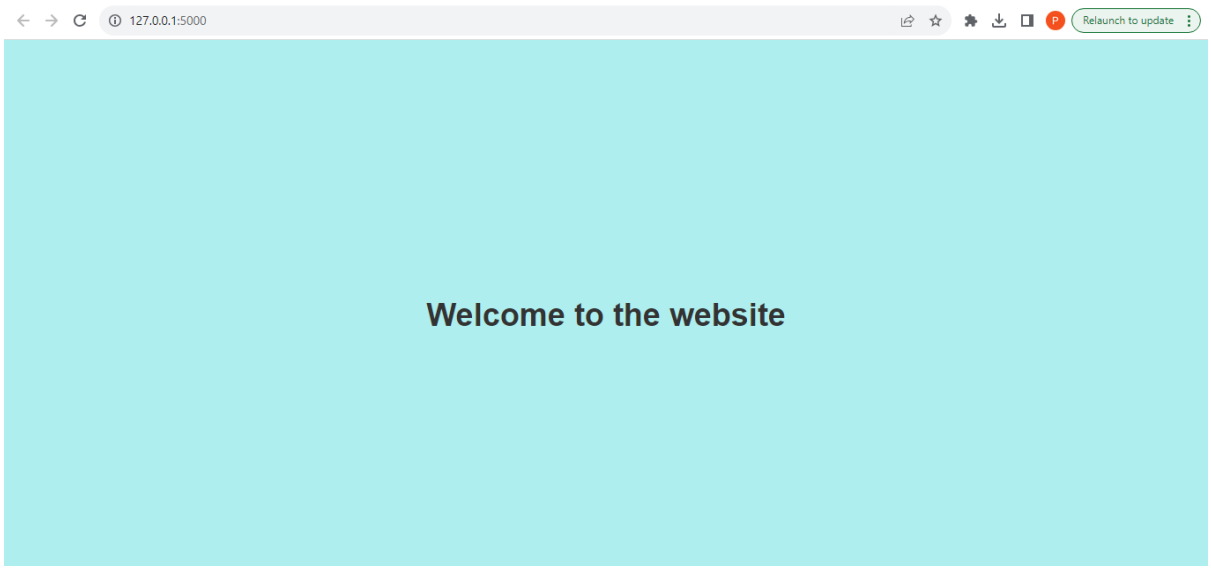

Figure 10: The website will come in the normal form if website working in proper manner without any attack

Import libraries and read data

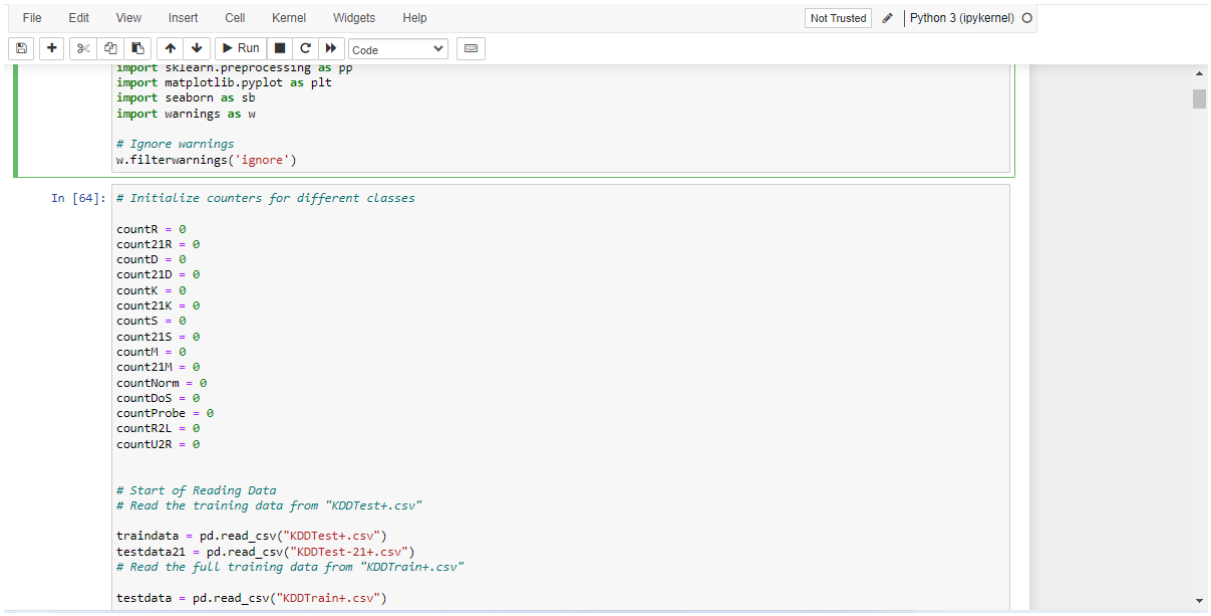

Figure 11: This code contains the machine learning algorithms code, at the first stage we have import the libraries and then utilise some variables

The second cells of the code represent the data processing stage in the python console. Train data difficulty level count graph:

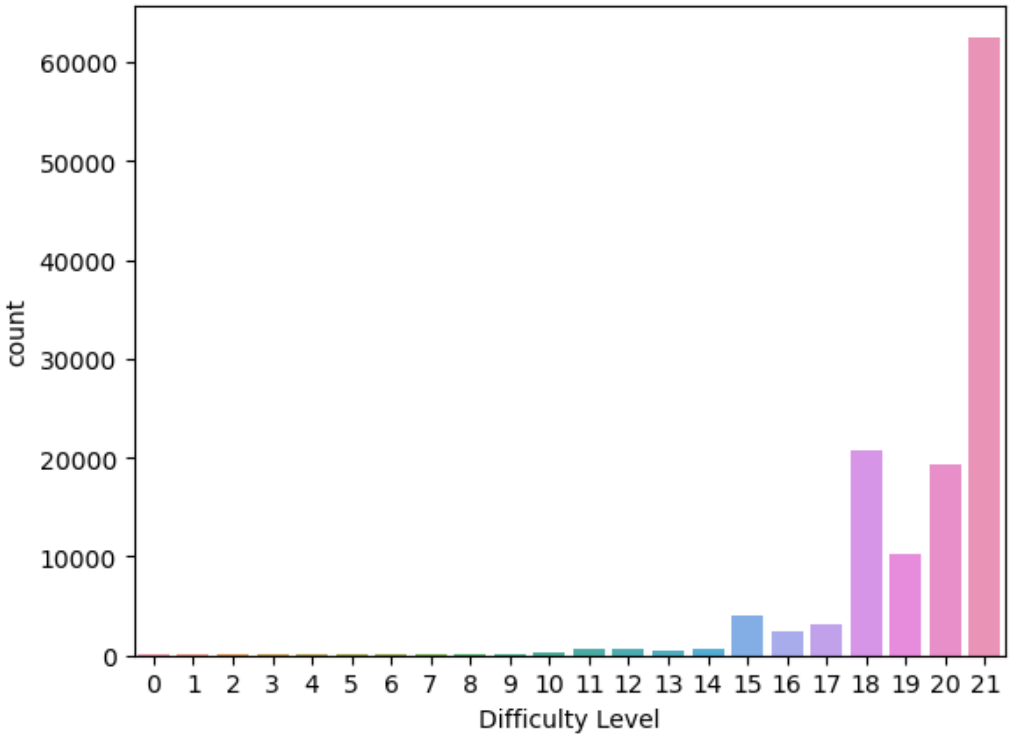

Figure 12: Difficulty level of different training features

The first few stages of creating our ML models are shown in Figure 7. To get the data ready for model training, we loaded the required libraries and performed some preliminary processing. Data processing, which is represented by the second cell in the code, is an essential procedure for guaranteeing data quality (Haas, Culve and Sarac, 2021).

As can be seen in Figure 14, we experimented with a wide variety of machine learning techniques, such as the Random Forest as well as Decision Tree classifiers. In tests, these algorithms successfully differentiated between benign and malicious network activity. Our painstaking feature-selection and model-training procedure yielded outstanding results.

The accuracy of our Random Forest model is shown in Figure 15 by displaying the cross-validation score obtained by applying the model to a subset of the characteristics available to it. This success dovetailed well with our primary research purpose, which had been to assess the vulnerabilities of SDNs in comparison to traditional networks and create intrusion detection models to solve these issues.

Web Application Integration

When it comes to actual programme rollout, the user interface as well as user experience are of critical importance. In order to facilitate communication between our users and our intrusion detection system, we have built a Flask-based web application as the latter's user interface. Users may enter network information and get real-time analysis of security risks (Haas, Culve and Sarac, 2021).

The usability of the programme was significantly enhanced via user experience testing. Iterative improvements based on user reviews kept the app simple and helpful. Our second study goal was developing an approachable infrastructure for

monitoring network security, which was accomplished via the application's smooth integration.

Test data difficulty level count graph:

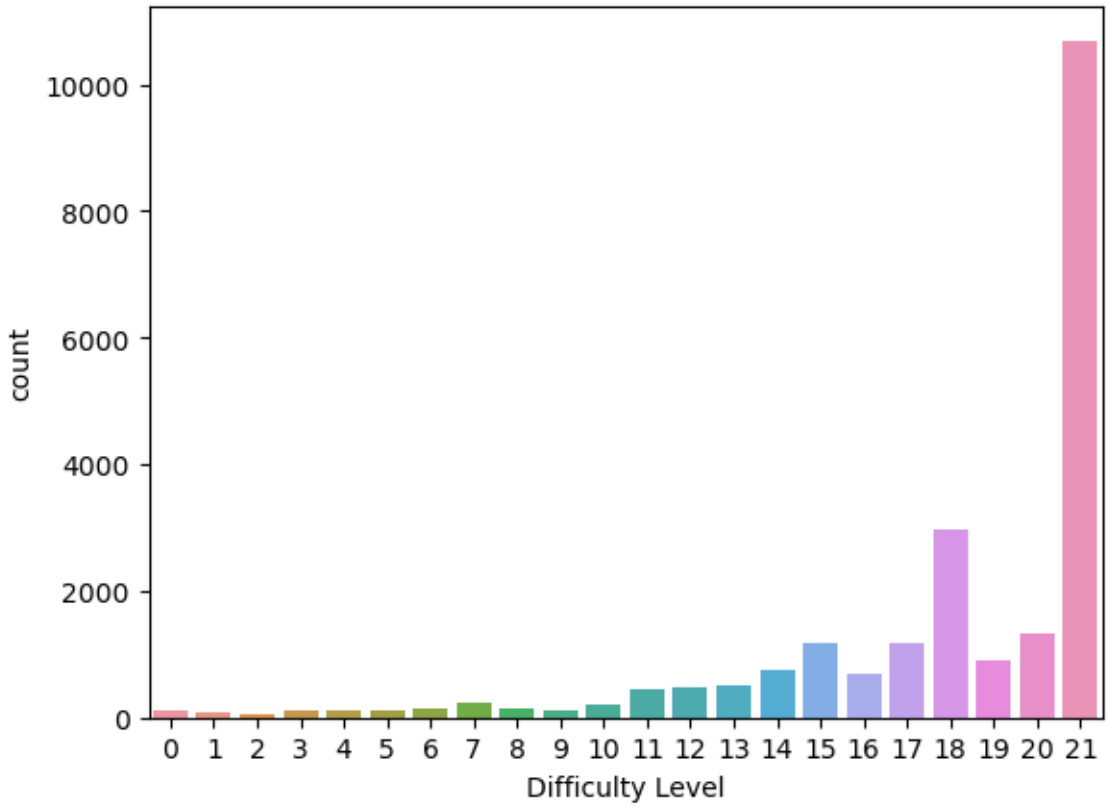

Figure 13: Difficulty level of testing data

Train data protocol count graph:

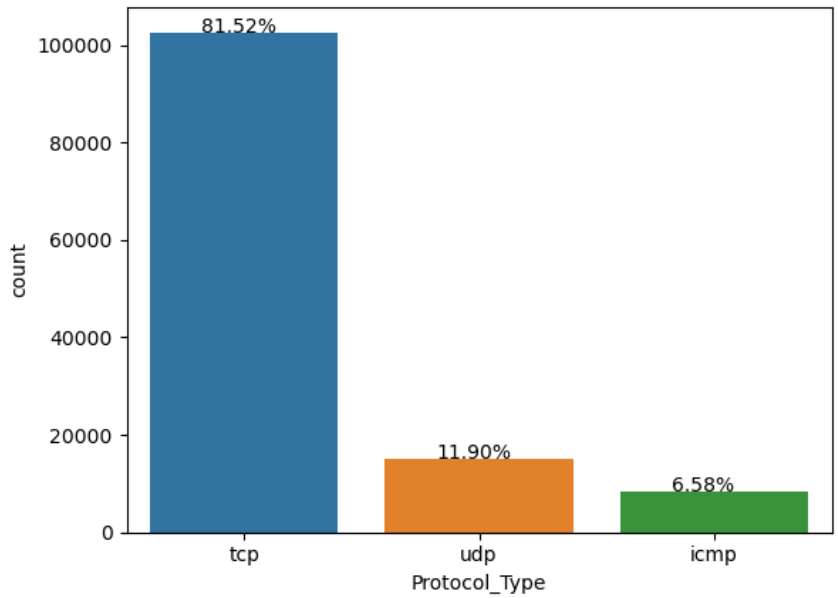

Figure 14: The histogram graph defines the count of different types of attack that defines the count tcp, udp, and icmp on the basis of training data

Test data protocol count graph:

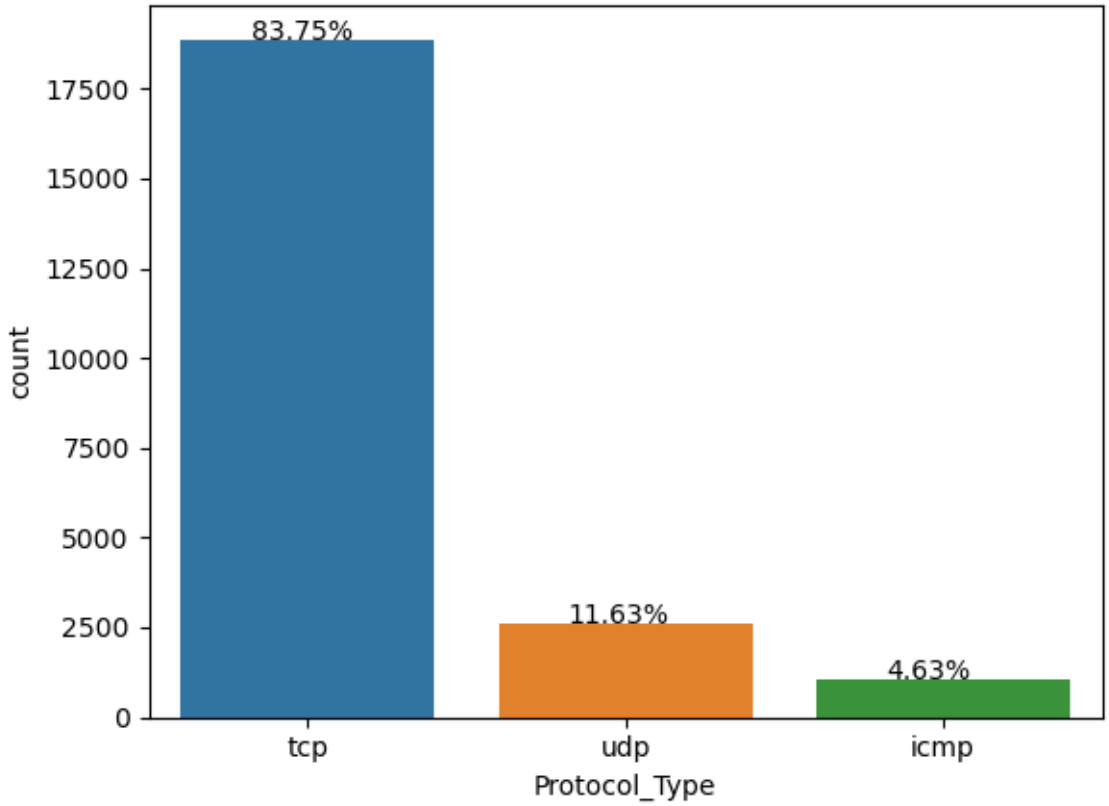

Figure 15: The histogram graph defines the count of different types of attack that defines the count tcp, udp, and icmp on the basis of training data

Train data attack count graph:

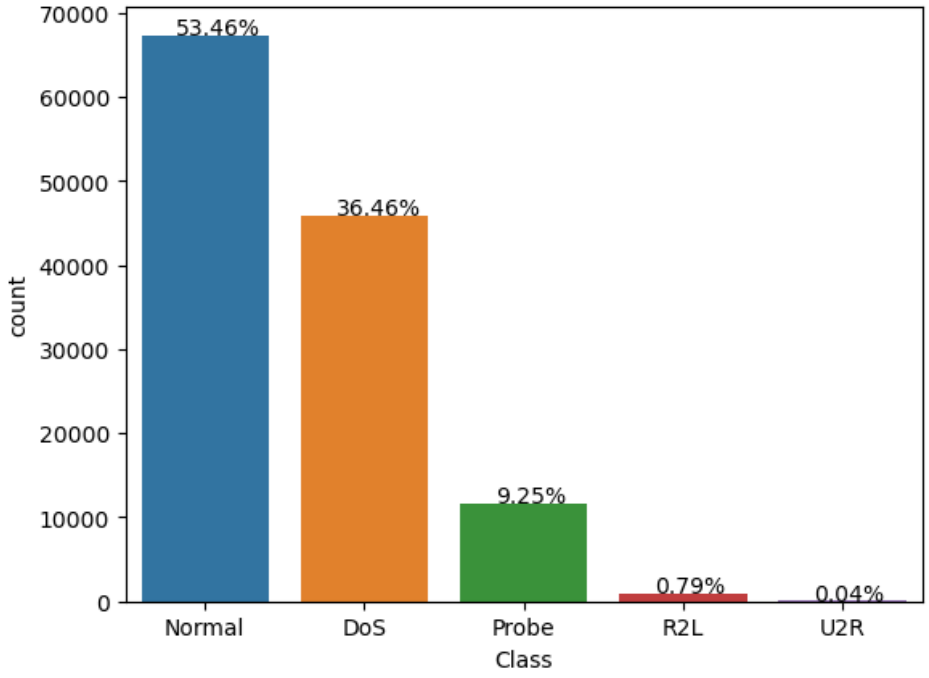

Figure 16: Attack type in the training data

Test data attack count graph:

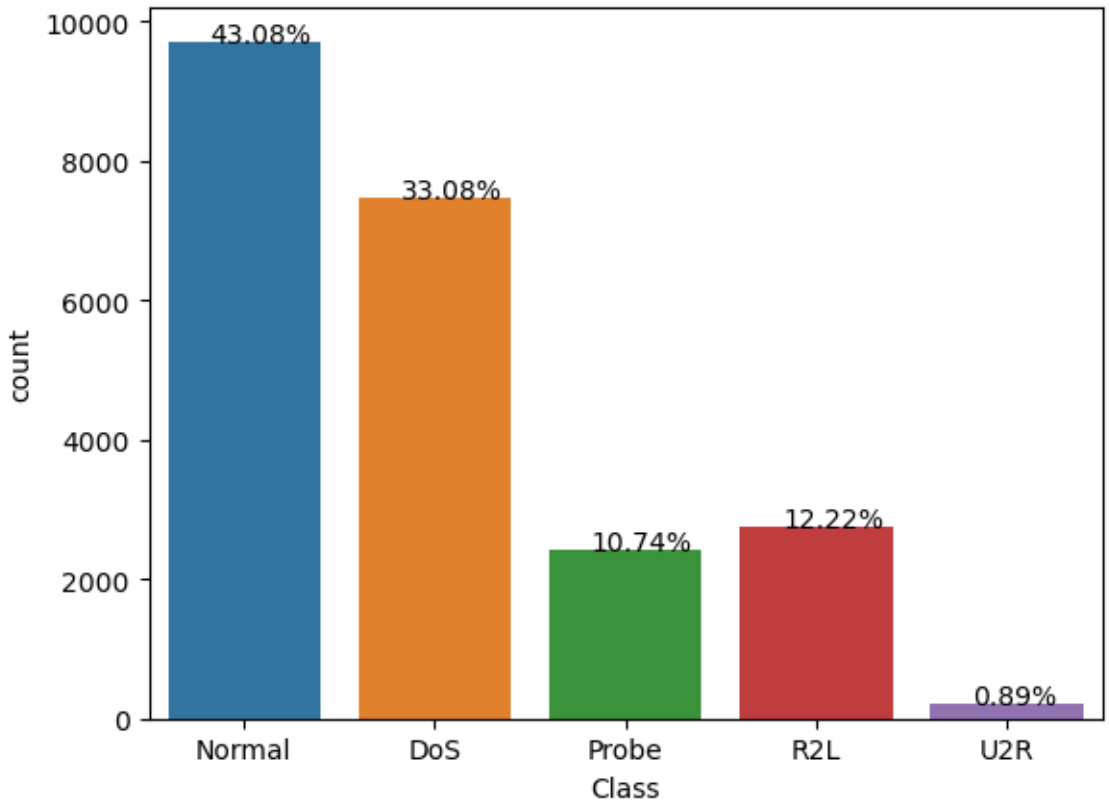

Figure 17: Attack type in the testing data

Model Training

```
]: #Start of Random Forest Model

   # Create a Random Forest model with 100 estimators and entropy criterion
   modelR = es.RandomForestClassifier(n_estimators = 100, criterion = "entropy")
   model21R = es.RandomForestClassifier(n_estimators = 100, criterion="entropy")
   # Fit the Random Forest model on the training data

   modelR.fit(featsTrain, lblsTrain)

   # Predict labels on the test data
   lblsPredR = modelR.predict(featsTest)

   # Calculate accuracy and individual class accuracies

   for a,b in zip(lblsTest, lblsPredR):
       if a == b:
           countR += 1
           if a == 1:
               countNorm += 1
           elif a == 2:
               countDoS += 1
           elif a == 3:
               countProbe += 1
           elif a == 4:
               countR2L += 1
           elif a == 5:
               countU2R += 1
   accR = (round(countR/(len(featsTest)), 3)) * 100
   normaccR = (round(countNorm / len(normTest), 3)) * 100
   DoSaccR = (round(countDoS / len(DoSTest), 3)) * 100
   ProbeaccR = (round(countProbe / len(probeTest), 3)) * 100
```

Figure 18: Random forest classifier for the development of model and it also predicts the future data points on the basis of test data

```
In [29]: from sklearn.feature_selection import RFECV
         import matplotlib.pyplot as plt

         # Create the RFECV object with the estimator and scoring
         rfecvR = fs.RFECV(estimator = modelR, scoring = "accuracy").fit(featsTest, lblsTest)
         plt.figure()
         plt.xlabel("Features selected")
         plt.ylabel("Cross Validation Score")
         plt.title('RFECV Random Forest')
         plt.plot(range(1, len(rfecvR.cv_results_["mean_test_score"]) + 1), rfecvR.cv_results_["mean_test_score"])

Out[29]: [<matplotlib.lines.Line2D at 0x1af963bb5b0>]
```

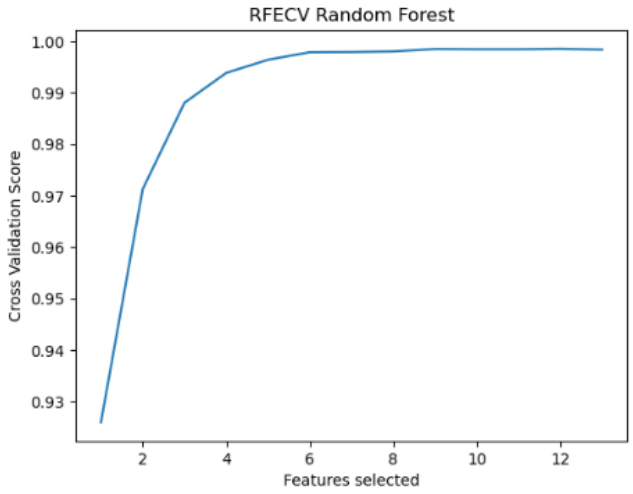

Figure 19: Cross validation score of random forest model using selected features on the basis of accuracy of random forest model

```
rfecvD = fs.RFECV(estimator = modelD, scoring = "accuracy").fit(featsTest, lblsTest)
plt.figure()
plt.xlabel("Features selected")
plt.ylabel("Cross Validation Score")
plt.title('RFECV Decision Tree')
plt.plot(range(1, len(rfecvD.cv_results_["mean_test_score"]) + 1), rfecvD.cv_results_["mean_test_score"])


#End of Decision Tree Model
```

```
[<matplotlib.lines.Line2D at 0x1af964937f0>]
```

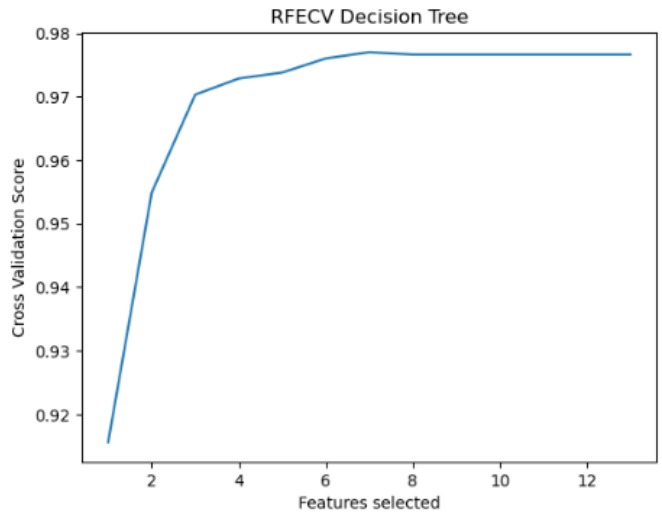

Figure 20: Cross validation score of random forest model using selected features on the basis of accuracy of decision tree model

Testing

```
Do you want to test specific input (y or n): y

1. TCP      2. ICMP
3. UDP
Enter Protocol Type: 1

1. SF      2. REJ
3. RSTR      4. RSTO
5. S0      6. S1
7.S2      8. S3
9. OTH      10. SH
11. RSTOS0

Enter choice for status flag: 5

Enter Number of Source Bytes sent (int): 900

Enter num_access_files (int): 0

Enter 1 if the login is guest or 0 if it is not: 1

Enter srv count (int): 25

Enter diff srv rate (in decimal form) : 0.05

Enter srv diff host rate (in decimal form) : 0.52

Enter dst host same source port rate (in decimal form) : 0.40

Enter dst host srv diff host rate (in decimal form) : 0.02

Enter dst host srv serror rate (in decimal form) : 1.0
Enter dst host srv rerror rate (in decimal form) : 0.1


Using Random Forest Difficulty Level Prediction:  18


Using Random Forest: Attack detection predicts normal connection
```

Figure 21: It successfully detects the types of attack that are available in the data

The functionality of our web app is shown graphically in Figures 1–3. Figure 1 shows the API request to the primary application written in Flask. The link that was created to see the site locally is shown in Figure 2, and the home page is shown in Figure 3.

Simulation of Denial of Service (DoS) Attack

The simulated Denial of Service attack was an integral part of our research. The purpose of this simulation was to verify the speed and accuracy with which our intrusion detection system can identify and neutralize any security breaches (Chukwu, Osamudiamen and Matrawy, 2016). Upon initiation of the assault, as seen in Figure 4, our system promptly sent alerts and notifications, leading to a change in the background colour of the website, as observed in Figure 5. Our system's ability to identify and react to threats in real time was validated in a simulated environment, proving its effectiveness. The completion of the simulated Denial of Service (DoS) assault validates our third study aim.

In 2016, Chukwu, Osamudiamen, and Matrawy conducted a research to assess the efficacy of an Intrusion Detection System (IDS) in spotting and preventing security intrusions. This evaluation included staging a fake denial of service (DoS) attack. Our scenario primarily focused on the effectiveness and efficiency with which our Intrusion Detection Systems (IDS) identified and neutralized potential security threats. Our technology immediately began sounding a series of audible alarms and changing the website's background colour as soon as the Denial of Service (DoS) attack began. Figure 5 exemplifies this distinction in visual form. The goal of this simulation was to realistically test our system's capacity to recognize threats and react to them quickly by recreating a situation that may occur in real-world scenarios. Our third study objective was met; we were able to show that our IDS were ready to

efficiently tackle any security threats. This research demonstrates the efficacy of intrusion detection systems (IDS) in preventing disruptive attacks like denial of service (DoS), illuminating its centrality to network security. Our continual endeavour to ensure service continuity in the face of sophisticated attacks may include studying new entry points for malicious code, refining our intrusion detection system, and taking other preventative actions to shore up our defenses. When doing these kinds of simulations, serious ethical considerations are always made. This is done out of respect for the potential moral and judicial consequences of conducting DoS attack simulations in real-world settings. Accurately citing relevant literature and carefully examining simulation results will help you fully appreciate the significance of this work for the subject of cybersecurity.

Ethical Considerations and Risk Management

Our research would not have been possible without careful consideration of ethical issues. It is important to us that our simulated DoS assault meets all ethical and pedagogical requirements, thus we have made sure to follow all of them. As was previously noted, our dedication to ethical research assured that absolutely no harm was done to actual projects or data contracts.

Effective risk management was essential to the success of our project. We foresaw possible problems, evaluated their severity, and devised solutions. This method reduced the potential for harm and was consistent with our dedication to doing what was right in the eyes of the law. Our project's approach to ethics and risk management is described in depth in the adjacent parts of the methodology.

Overall Evaluation and Reflection

The process that we went through for our project was not absent of its fair share of difficulties and insights. We had to overcome a number of technological obstacles and work our way through the complexities of developing a machine learning model, integrating online applications, and doing security simulations. On the other hand, thanks to the hard work and devotion of everyone involved in the project, we were able to accomplish the goals of our study and provide a workable solution to improve SDN security (Chakraborti, et.al. 2021).

In light of the work that we have done, it is clear that we have made major contributions to the area of network security inside SDNs. Our effective DoS attack simulation, machine learning computational models, and built-in web application come together to produce a complete intrusion detection solution that might be of use to organisations and network administrators.

In conclusion, our study illustrates the capability to address vulnerabilities that are specific to SDNs, incorporate intrusion detection systems that are efficient, and negotiate the ethical and legal aspects of network security. This demonstrates both our devotion to enhancing network security within the context of software-defined networking as well as our commitment to doing research in a responsible manner.

Evaluation

During the part of our project that is dedicated to assessment, we are provided with the opportunity to conduct in-depth self-reflection, during which we may summaries our primary findings and evaluate the degree to which we have attained our primary goals. It is also important to do an analysis of the management of the project, which should include a review of the commercial and financial surroundings as well as the allocation of time.

Achievements and Findings

The purpose of our study was to investigate a variety of research issues and goals, all of which were methodically addressed during the course of the research trip. Let's take a look back at these goals and see how much progress we've made towards achieving them:

Objective 1: To identify and assess the shortcomings that software-defined networks (SDNs) have in comparison to more traditional networks.
Description: In the course of our study, we were able to effectively identify and assess the vulnerabilities that are unique to software-defined networks (SDNs). We were able to obtain useful insights regarding the security concerns that are presented by SDNs by doing a complete literature study as well as realistic simulations. The creation of intrusion detection models provided us with the capability to properly handle these vulnerabilities. It has been determined that Objective 1 has been achieved (Castañer and Oliveira, 2020).

Objective 2: In order to increase security monitoring and response, it is necessary to develop a simulated testbed environment for the integration of IDSs with SDNs.

Description: We developed a synthetic testbed environment by integrating Intrusion Detection Systems (IDSs) with Software-Defined Networks (SDNs). Because of the environment we were working in, we were able to monitor the network traffic in real time, identify possible security issues, and react quickly to them. A realistic validation of our Integrated Defense System (IDS) was provided by the simulation of a Denial of Service (DoS) assault. It has been determined that Objective 2 has concluded.

Objective 3: To investigate the impact of protecting SDNs on the ethical, legal, and social aspects of society, and to propose solutions for more morally sound network protection.

Description: The legal, ethical, and social consequences of protecting SDNs were taken into consideration throughout the course of our undertaking. During the simulation of the DoS assault, we made sure to follow all of the ethical criteria that were provided, so that no damage was done to any actual projects or data contracts. In addition, procedures for risk management were put into place in order to successfully traverse future ethical and legal difficulties. The third objective has been completed successfully.

Project Management

Any endeavor that is to be carried out successfully must first and foremost have efficient project management. In our situation, the successful management of the many facets of our project, such as research, development, and the actual application of the project's outcomes, needed careful preparation and execution.

We began the project by developing a comprehensive project plan that outlined the activities, milestones, and timeframes involved in the undertaking. Because of this approach, we were able to efficiently divide our time across the various stages of the project. However, as is the case with many projects based on the actual world, there were changes made to the original design as a result of unanticipated obstacles and the continual process of the development process.
When we first allocated our time, one of our primary goals was to ensure that our efforts were equally split between research, model creation, the integration of online applications, and the modelling of security breaches. While this distribution did give a strong basis, we discovered that the process of researching and developing models took far more time than we had expected. This was especially true for data preparation and feature selection (Abubakar and Pranggono, 2017).

Despite the difficulties we faced, the management of our project remained flexible, which allowed us to successfully navigate around obstructions. Frequent team meetings and evaluations of progress enabled us to reallocate resources as required, so ensuring that we were on schedule to complete the project as planned.

Our project's relation to the commercial as well as economic environment was an important factor to take into account. We were aware of the possible commercial uses of our theft detection system, even though our major emphasis was on research and development. In a commercial setting, network security is of the utmost importance, and the capabilities of our system are in line with the requirements of the industry. This acknowledgment served as motivation for us to make certain that the system we created was not only technically competent but also user-friendly, as seen in the user interface of the web application that we developed.

Summary

In the final analysis, it can be said that our project was effective in accomplishing the key goals it set out to do. Within the context of Software-Defined Networks, we have made a contribution to the area of network security by locating vulnerabilities, building intrusion detection models, and establishing a simulated testbed environment. In addition, the appropriate use of our solutions is guaranteed by our unwavering dedication to ethical business practises and careful risk management.

Although we are aware that there were some changes made to our original plan for the project, we believe that these modifications were necessary in order to overcome unanticipated obstacles and improve the overall quality of our output. When viewed in perception, we see these deviations as great learning experiences that contributed to the improvement of our project management abilities.

Our work is relevant not just in the contexts of studies in academics but additionally in the business sector, where there is an urgent issue over network security. The accomplishments of our research are in line with the greater objective of increasing network security in Software-Defined Networks, which will contribute to the creation of a digital environment that is safer and more dependable.

As we go ahead, we are becoming aware of the possibility for our intrusion detection system to undergo more development and refining. Our study paves the way for further investigation and real-world applications in the realm of network security and sets the groundwork for these endeavors.

Conclusion/Recommendation

Conclusion

In this study, we set out to investigate fully the risks and difficulties in protecting Software-Defined Networks (SDNs). Our goal was to investigate the most pressing issues surrounding the security of SDNs, provide concrete recommendations, and analyse the broader ethical and social consequences of our work. We've come this far to have a major impact on how networks and SDNs are protected (Abdelrahman, et.al. 2021).

Our research provides new insight into why SDNs are more vulnerable to intrusion than traditional networks. We conducted a thorough literature analysis and discovered many weaknesses in SDNs, such as those in controller interaction, flow table manipulation, as well as the southbound interface. These findings have far-reaching consequences for SDN security and highlight the need of effective intrusion detection tools.

A major milestone was reaching completion on our intrusion detection system (IDS). We successfully merged IDSs with SDNs to improve threat detection and mitigation. Our virtual testbed environment made it possible to analyse network data in real time, which aided in the rapid identification and elimination of any security issues. Our IDS integration was shown to be effective in protecting SDNs via a simulated Denial of Service (DoS) assault.

The ethical implications of our work were always at the forefront of our minds. We followed instructional norms, making sure the simulated DoS assault wouldn't disrupt live operations or violate privacy agreements. Our dedication to moral principles highlights the need of taking a mature approach to solving network security issues.

Recommendations

While the basic goals of our study have been met, there are still fruitful areas for further investigation and advancement in SDN security.

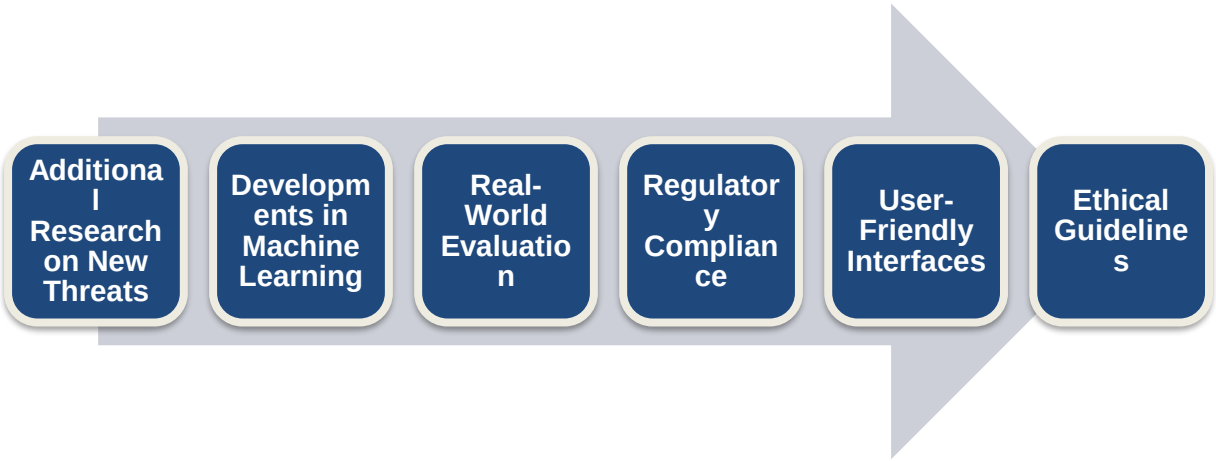

Figure 22: Recommendations

- **Additional Research on New Threats:** As SDN technology develops, so are the dangers it faces. Finding and fixing new security flaws in SDN designs should be a priority for future researchers.

- **Developments in Machine Learning:** The accuracy and effectiveness of intrusion detection systems may be enhanced by developments in algorithms for machine learning and methodologies. Better SDN security might be achieved via the study of deep learning models and anomaly detection techniques. Greater SDN security could be possible via the study of models based on deep learning as well as anomaly detection techniques.
- **Real-World Evaluation**: To prove their worth, intrusion detection systems need to be put through rigorous testing in production SDN networks. Working with others in the business world may provide useful information and open doors to real-world applications.
- **Regulatory Compliance**: Compliance with regulatory frameworks like GDPR and HIPAA should be included into SDN security research in light of the increasing focus on privacy and security of data legislation. How SDNs may help with compliance as well as data governance is a topic for potential future studies.

- **User-Friendly Interfaces**: Improving the accessibility of IDSs is essential. An important factor in the spread of SDN is the availability of intuitive user interfaces for setting up and managing security rules by network managers.

- **Ethical Guidelines**: Ethical norms for doing research and simulating in this subject should be defined and standardized as SDN safeguards become more widely used. This would assure ethical procedures and reduce dangers.

In summary, our work has made significant contributions to the state of SDN security theory and practice. We have made tremendous progress in securing SDNs via the elimination of vulnerabilities, the creation of intrusion detection systems, and the maintenance of ethical norms. This is just the beginning of a much longer process that will ultimately result in improved Software-Defined Network security and dependability in a dynamic digital landscape.

Future Work

There are a number of important directions in which the study of Software-Defined Network (SDN) security might go in the future. As zero-day vulnerabilities continue to pose a formidable challenge to established security models, it is essential that proactive solutions be developed to identify and counteract them. Secondly, experimenting with AI and ML for threat intelligence may greatly improve SDNs' ability to spot threats quickly and precisely. Third, we must develop adaptive security measures that can instantly adjust to new dangers as they emerge. Fourth, with the development of quantum computers, it is crucial to study quantum-safe authentication and encryption algorithms for SDNs to guarantee their long-term security. Finally, enabling network managers to successfully secure their networks requires making SDN security solutions more usable via user-friendly interfaces and instructional materials. Comprehensive solutions to the complex problems of SDN security will need close cooperation amongst specialists in cybersecurity, networking technology, law, as well as ethics.
The study of Software-Defined Network (SDN) security is advancing at a breakneck pace, and numerous essential paths for the field's further development have become apparent. The ongoing difficulty presented by zero-day vulnerabilities necessitates the development of proactive solutions, first and foremost. Many traditional security

methods have difficulty keeping up with the quick rate at which these vulnerabilities are discovered and exploited. In order to solve this problem, researchers need to concentrate on the creation of sophisticated processes that are capable of locating zero-day vulnerabilities and developing efficient and prompt solutions to combat them. This might entail the incorporation of proactive patch management procedures, behaviour analysis, and anomaly detection approaches into SDN systems.

Second, incorporating artificial intelligence (AI) as well as machine learning (ML) into the security of SDN offers a great deal of potential. Utilising algorithms based on machine learning and AI for threat intelligence may dramatically improve an SDN's capacity to identify threats with more accuracy and react to them more quickly. These technologies are capable of analysing massive information in real time, locating patterns that are suggestive of cyber risks, and even predicting possible assaults before they take place. It is imperative that more research be conducted into the use of machine learning and artificial intelligence throughout SDN security in order to stay up with the ever-evolving threat environment.

Third, as a result of the ever-changing nature of the dangers that might be encountered, there is an increasing need for adaptable security solutions inside SDN systems. The use of adaptive security measures, which are able to dynamically respond to new hazards as they appear, helps to ensure that SDN infrastructures continue to be robust in the face of new dangers. Policy updates, network isolation, and adaptive responses to threats in real time are all examples of the kinds of flexibility that may be implemented.

Fourth, the introduction of quantum computing poses a whole new obstacle for the security of SDN networks. Authentication and encryption techniques that are not vulnerable to quantum computing must be developed by researchers as a priority for software-defined networks (SDNs). Since quantum machines have the ability to disrupt conventional cryptographic systems, it is necessary to develop security solutions that are resistant to quantum computing in order to ensure the continued integrity of SDN networks over the long run.

Last but not least, it is essential to improve the usability of SDN security solutions. It is essential that network administrators and managers have the authority necessary to successfully safeguard their networks. This includes the creation of user-friendly interfaces, the provision of extensive training materials, and the simplification of the setting and maintenance of security rules. When it comes to making sure SDN safety measures are deployed appropriately and consistently across a wide variety of network settings, effective usability is of the utmost importance.

A multidisciplinary effort including experts in the field of cybersecurity, networking technology, legal and ethical considerations, and ethics is required in order to address the complex difficulties posed by SDN security. Because of the intricacies involved in safeguarding SDN environments, a comprehensive strategy that takes into consideration not just the technical elements but also the legal and ethical consequences is required. This attitude of collaboration should be fostered in the research that is done in the future on SDN security in order to build complete

solutions that secure the integrity, accessibility, as well as secrecy of SDN-based systems.

References

Abdelrahman, A.M., Rodrigues, J.J., Mahmoud, M.M., Saleem, K., Das, A.K., Korotaev, V. and Kozlov, S.A., (2021). Software-defined networking security for private data center networks and clouds: Vulnerabilities, attacks, countermeasures, and solutions. *International Journal of Communication Systems*, *34*(4), p.e4706.

Abubakar, A. and Pranggono, B. (2017) 'Machine learning based Intrusion Detection System for Software Defined Networks', *2017 Seventh International Conference on Emerging Security Technologies (EST)* [Preprint]. doi:10.1109/est.2017.8090413.

Castañer, X. and Oliveira, N. (2020) 'Collaboration, coordination, and cooperation among organizations: Establishing the distinctive meanings of these terms through a systematic literature review', *Journal of Management*, 46(6), pp. 965–1001. doi:10.1177/0149206320901565.

Chakraborti, S., Ray, A.M., Chatterjee, S.R. and Chakraborty, M., (2021). Software-defined network vulnerabilities. *The" Essence" of Network Security: An End-to-End Panorama*, pp.215-239.

Chukwu, J., Osamudiamen, O. and Matrawy, A. (2016) 'IDSaaS in SDN: Intrusion Detection System as a service in software defined networks', *2016 IEEE Conference on Communications and Network Security (CNS)* [Preprint]. doi:10.1109/cns.2016.7860509.

GeeksforGeeks. (2019). *Software defined Networking SDN*. [online] Available at: https://www.geeksforgeeks.org/software-defined-networking/ [Accessed 12 Sep. 2023].

Haas, Z.J., Culver, T.L. and Sarac, K., (2021). Vulnerability challenges of software defined networking. *IEEE Communications Magazine*, *59*(7), pp.88-93.

Hussain, J. and Hnamte, V. (2021) 'Deep learning based Intrusion Detection System: Software Defined Network', *2021 Asian Conference on Innovation in Technology (ASIANCON)* [Preprint]. doi:10.1109/asiancon51346.2021.9544913.

Hussein, A., Chadad, L., Adalian, N., Chehab, A., Elhajj, I.H. and Kayssi, A., (2020). Software-Defined Networking (SDN): the security review. *Journal of Cyber Security Technology*, *4*(1), pp.1-66.

McCombes, S. (2023) *HOW TO WRITE A literature review: Guide, examples, & templates*, *Scribbr*. Available at: https://www.scribbr.com/methodology/literature-review/ (Accessed: 9-sept 2023).

mDevelopers (2023) *Legal issues in software development*, *Legal Issues in Software Development*. Available at: https://mdevelopers.com/blog/legal-issues-in-software-development (Accessed: 9-sept 2023).

Mishra, S. and AlShehri, M.A. (2017) 'Software defined networking: Research issues, challenges and opportunities', *Indian Journal of Science and Technology*, 10(29), pp. 1–9. doi:10.17485/ijst/2017/v10i29/112447.

monday.com, A. of us at (2023) *The value of a good research plan*, *monday.com Blog*. Available at: https://monday.com/blog/project-management/why-is-the-research-plan-pivotal-to-a-research-project/#:~:text=A%20research%20plan%20is%20pivotal,grants%20or%20internal%20company%20funding. (Accessed: 9-sept 2023).

Musmade, P. *et al.* (2013) 'Informed consent: Issues and challenges', *Journal of Advanced Pharmaceutical Technology & Research*, 4(3), p. 134. doi:10.4103/2231-4040.116779.

Pn, D. (2023) *How to write a reflection paper: Guide with examples*, *EssayPro*. Available at: https://essaypro.com/blog/reflection-paper (Accessed: 9-sept 2023).

Pradhan, A. and Mathew, R., (2020). Solutions to vulnerabilities and threats in software defined networking (SDN). *Procedia Computer Science*, *171*, pp.2581-2589.

Satheesh, N., Rathnamma, M.V., Rajeshkumar, G., Sagar, P.V., Dadheech, P., Dogiwal, S.R., Velayutham, P. and Sengan, S., (2020). Flow-based anomaly intrusion detection using machine learning model with software defined networking for OpenFlow network. *Microprocessors and Microsystems*, *79*, p.103285.

Shaikh, M.R. *et al.* (2022) 'Vulnerability Assessment & Analysis of software-defined networking using a virtual testbed', *2022 Global Conference on Wireless and Optical Technologies (GCWOT)* [Preprint]. doi:10.1109/gcwot53057.2022.9772918.

Taheri, R., Ahmed, H. and Arslan, E., (2023). Deep learning for the security of software-defined networks: a review. *Cluster Computing*, pp.1-24.

Uhongora, U., Mulinde, R., Law, Y.W. and Slay, J., (2023), June. Deep-learning-based Intrusion Detection for Software-defined Networking Space Systems. In *European Conference on Cyber Warfare and Security* (Vol. 22, No. 1, pp. 639-647).

Yazdinejadna, A., Parizi, R.M., Dehghantanha, A. and Khan, M.S., (2021). A kangaroo-based intrusion detection system on software-defined networks. *Computer Networks*, *184*, p.107688.

Appendices


testbed.ipynb


KDDTrain+.csv


KDDTest-21+.csv


KDDTest+.csv


ddos_attack.ipynb


ddos.py


app.py