Research on Security Issues in Software-Defined Networking

A PROJECT REPORT

Submitted by

Kabir Dhruw (20BCY10077)
Apoorv Singh (20BCY10111)
Rishi Kumar (20BCY10052)
Sonalika Bharadwaj (20BCY10061)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

VIT BHOPAL UNIVERSITY

KOTHRIKALAN, SEHORE

MADHYA PRADESH - 466114

MAY 2022

BONAFIDE CERTIFICATE

Certified that this project report titled **"Research on Security Issues in Software-Defined Networking"** is the Bonafide work of **"**Kabir Dhruw, Apoorv Singh, Rishi Kumar and Sonalika Bharadwaj**"** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported at this time does not form part of any other project/research work based on which a degree or award was conferred on an earlier occasion on this or any other candidate.

PROGRAM CHAIR

Dr. R. Rakesh
Assistant Professor
School of Computer Science and Engineering

VIT BHOPAL UNIVERSITY

PROJECT GUIDE

Dr. MD Asdaque hussain,
Senior Assistant Professor
School of Computer Science and Engineering

VIT BHOPAL UNIVERSITY

The Project Exhibition II Examination is held on _____

ACKNOWLEDGEMENT

First and foremost, I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to Dr. Rakesh Kumar, Head of the Department, School of Computer Science and Engineering for much of his valuable support encouragement in carrying out this work.

I would like to thank my internal guide Mr MD Asadeque Hussain. for continually guiding and actively participating in my project, giving valuable suggestions to completethe project work.

I would like to thank all the technical and teaching staff of the School of Computer Science with specialization in cyber security and digital forensics, who extended directly or indirectly all support.

Last, but not least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

LIST OF FIGURES AND GRAPHS

LIST OF TABLES

ABSTRACT

The evolution of the internet, the explosion of information services and the emergence of new technologies sound the knell of the conventional network architecture. The reasons are related to its rigidity, complexity, and cost. Software Defined Networking (SDN) is an emerging paradigm that promises to resolve the limitations of the conventional network architecture. Thanks to its programmability, centralization, federation and externalization, SDN strategy is to make it the backbone of the future internet and an enabler for new information services and technologies The relation between SDN and security is a hot subject that contributes to reaching these goals. SDN and security have a reciprocal relationship. In this thesis, we study and explore two aspects of this relationship. On the one hand, we study security for SDN by performing a vulnerability analysis of SDN. Such security analysis is a crucial process in identifying SDN security flaws and in measuring their impacts. It is necessary for improving SDN security and for understanding its weaknesses. Thus, the thesis provides a generic classification of SDN vulnerabilities. It relies on the Common Vulnerability Scoring System (CVSS) to quantify their severity.

TABLE OF CONTENTS

Chapter 1

Project Description and Outline

1.1 INTRODUCTION

The conventional network architecture is becoming inadequate for the requirements of new technologies such as Cloud Computing, Internet Of Things, Bring Your Own Device and for the expansion of internet services. These technologies and services need large-scale computing, high resource availability, dynamic infrastructure tailoring, automation, resilience, holistic knowledge and other needs. However, conventional architecture cannot satisfy them due to its rigidity, complexity, costs and lack of customization. Its rigidity is a barrier for network innovation because developers are limited to the capacities of proprietary devices and technologies. Its configuration and maintenance are complex and costly because in this architecture network devices are rigid, they use low-level abstractions and do not collaborate with each other. Besides, network devices in the conventional network architecture are vendor dependent, and their evolution relies on the vendor business model. The vendor may hamper network customization when it contradicts its goals.

Software Defined Networking (SDN) promises to resolve the limitations mentioned above. SDN is the softwarization of networks. This softwarization shapes the design and the development of systems and services from a monolithic architecture to a component-based architecture where multiple technologies and stakeholders can deploy their services and collaborate. SDN breaks the vertical integration of the conventional network architecture. As a result, SDN releases network innovation from the vendor dependence paradigm. SDN provides standardized programmable interfaces to network applications that enable them to reprogram the network infrastructure and

customize it according to their needs. Moreover, SDN elevates user network programs from a Command Line Interface(CLI) model to a high abstracted model that federates different applications and users around network policies and holistic network knowledge. SDN reduces network provisioning and maintenance time because it automatizes their tasks. It reduces the manual intervention of network operators and their errors. SDN and security have a particular relation. On the one hand, SDN inherits the vulnerabilities of the conventional network architecture, and it introduces new vulnerabilities that did not exist before.

In this matter, security for SDN is the study of SDN security issues and their resolutions. On the other hand, SDN improves security applications thanks to its advantages. Its programmability automatizes security behavior in the network infrastructure. Its orchestration simplifies the expression and deployment of security policies. Its centralization provides a global knowledge that improves the consistency of security applications. Its programmable interfaces enable the pervasiveness of security policies in all the infrastructure.

1.2 MOTIVATION FOR THE WORK

In the coming future, **SDN will become a technology that will be more responsive, fully automated, and highly secure**. In the near future, you may also get to know about software-defined mobile networking (SDMN), which will make the mobile network controllable from a software.

1.3 PROBLEM STATEMENT

One of the most common assumptions taken in the previous SDN literature is that SDN introduces new vulnerabilities due to its architecture and features. However, there is not a clear piece of work that addresses these vulnerabilities. Also, all the existing vulnerability computation frameworks and models are inadequate for SDN because they do not integrate its specific properties.

1.4 OBJECTIVE OF THE PROJECT

For this project, we study the security of SDN and propose a solution that analyzes SDN vulnerabilities. we propose an approach to assess SDN security vulnerabilities by combining the reversion of security objectives with SDN assets. Then, we quantify the severity of the vulnerabilities using CVSS. CVSS computes the severity based on mathematical metrics that address the intrinsic, temporal and environment characteristics of vulnerabilities. However, we find that there are significant factors, specific to SDN, which are not covered by CVSS. These factors affect the security of SDN and enlarge its vulnerability surface. To cover these factors, we introduce into the severity computation AHP weights to determine the importance of each SDN assets regarding its importance for SDN specific features. AHP compounds four steps. In the first step, we determine the importance of each SDN specific feature in relation to the others. In the second step, we determine the importance of each SDN asset to the rest of the assets according to each SDN feature. In the third step, we combine all the results to determine the weight of each SDN asset. In the fourth step, we integrate the weights to CVSS computations to adapt CVSS to SDN.

1.5 ORGANISATION OF THE PROJECT

This thesis is organized as follows. Chapter 2 covers key background knowledge on SDN. It describes the SDN paradigm, its specific features. It presents its architecture and OpenFlow. It discusses its challenges and benefits. Chapter 3 studies the relationship between SDN and the hardware requirements of this project. The main contributions of this thesis are in chapters 4, 5 and 6. Chapter 4 presents the architectural design of the project and the methodology used. Chapter 5 presents the technical implementation and the code solutions for this project Chapter 6 gives an idea about the possible future developments in the field of networking due to the emergence of SDN Chapter 7 concludes the thesis and outlines future work along with the references.

1.6 SUMMARY

Through this project, we aim to study different vulnerabilities available for SDN and try to provide some possible solution for those vulnerabilities..

CHAPTER-2

RELATED WORK INVESTIGATION

2.1 Introduction

The internet became a conventional architecture for human communication because it leverages human interactions beyond time and space. This technology is largely deployed with a plethora of interconnected devices and services that process, manage and deliver mass information. However, conventional network architecture experiences many issues. It is based on the vertical integration of software and hardware into the same device. The data plane layer, the control layer, and the application layer are vertically integrated together in the same proprietary devices. As a result, services and middle-boxes such as firewalls are costly in terms of price, complexity and maintenance [1]. They cannot be adapted to the client's need without the intervention of their manufacturers; neither can they collaborate with the network devices of different manufacturers. Many middleware products have been developed to enable them to interact. Unfortunately, the adoption of these solutions has not been a success due to their complexity, their costs and the rapid evolution of services. The rigidity of the network devices and their heterogeneity are a barrier for network innovation. It confines developers around the complexity of the hardware and its limitations. Furthermore, the administration of the network devices is costly because it is based on technology proprietary products. Administrators and engineers manually configure the plethora. As a consequence, they prompt errors and affect the security of the network.

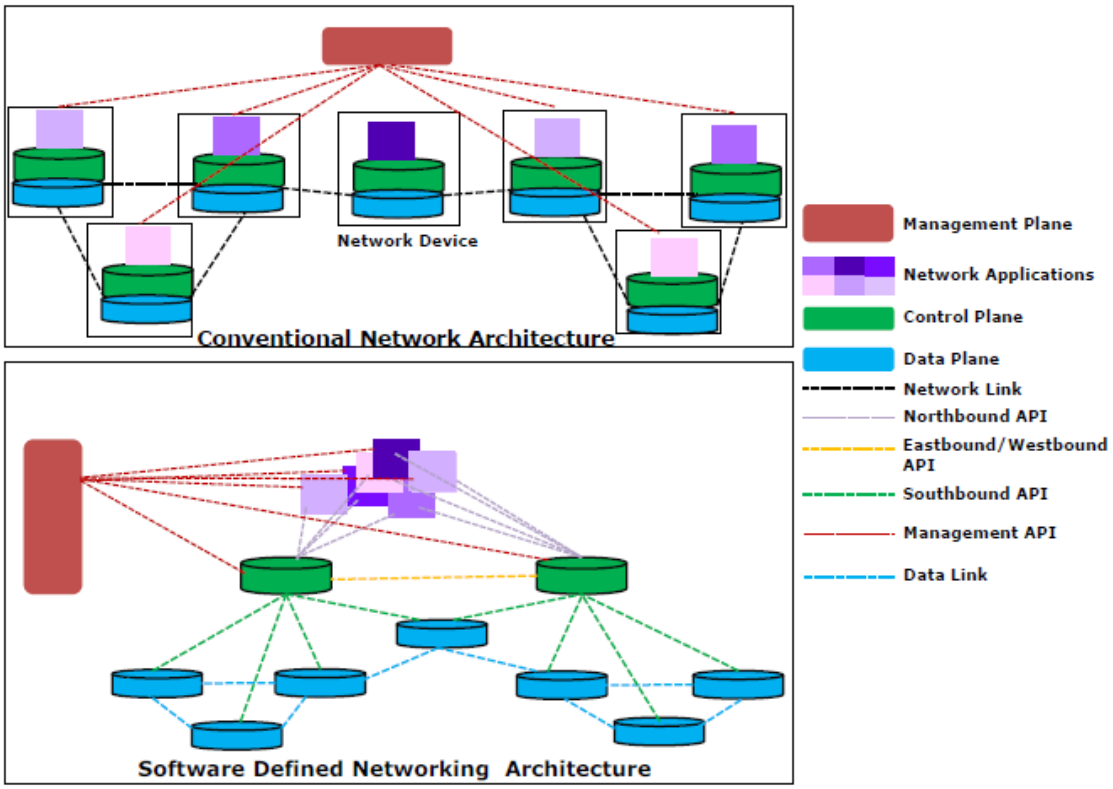


Fig 1 Conventional Network Architecture Vs. SDN Architecture

Software Defined Networking (SDN) brings solutions to the issues mentioned above by breaking the vertical integration of the conventional architecture and standardizing the different networking interfaces. As a result, network devices become network elements that perform only data plane tasks such as forwarding. SDN separates the three layers by externalizing the control layer and the application layer from the data plane layer. Therefore, an independent software entity performs control decisions. Applications become able to reprogram network elements without being limited by the constraints of hardware complexity. This chapter introduces SDN through its principles and its specific features. It presents its architecture. It discusses OpenFlow. It also highlights its advantages and challenges.

2.2 Core area of the project

In this project, we are studying and comparing the past studies done on SDN so to expand our overall knowledge about SDN.

2.3 Existing methods

In recent years, work has been done on security mechanisms and solutions for the availability, confidentiality, and integrity of the elements that compose the SDN architecture, with reviews conducted by some authors. For example, in several security aspects covering problems and solutions of SDN architecture layers are discussed. In [3], the authors conduct a broad and general review of SDN, including architecture security. However, the manuscript does not detail in depth the mechanisms used to solve the mentioned shortcomings. In [4], the authors expose the advances in SDN networks and their various applications, recognizing that SDN has been developed without taking into account several fundamental aspects of security, including both architecture security and measures to prevent and detect attacks. In this regard, the authors focus on the security of controllers. In [5], the process of network topology discovery in SDN and potential security issues are addressed. In [6], the authors try to raise awareness of the vulnerabilities that may exist in stateful data planes. In [7], some security threats to SDN controllers and mitigation techniques are considered. In [8], several security problems of the SDN architecture are presented, with a brief review of some solutions. In [9], the authors focus on the security and privacy aspects of 5G technology, exposing in a general way some problems of the SDN architecture, without detailing possible solutions. In [10], controller architectures are reviewed, taking into account several factors, one of which is security.

2.4 Issues/Observation

To the best of our knowledge, no work has reviewed the most relevant security issues of all interfaces and layers of the SDN architecture (including the differentiation between stateless and stateful data planes) in a single manuscript or classifies in detail the solutions to these issues.

Chapter-3

REQUIREMENT ARTIFACTS

3.1 Introduction

For the elaboration of this survey, articles were selected with the support of search tools, which allowed us to ex- tract information from databases, such as IEEE Xplore, Sco- pus, Springer, and the ACM Digital Library. Although the searches with the tools mentioned yielded exorbitant amounts of papers relating to security solutions in the SDN world, it was important to distinguish between the following: a) secu- rity solutions for the SDN architecture, b) security solutions that use the SDN architecture or some of its components, and c) authors who mention SDN in their titles, although they use only some elements of the architecture to deploy tests, since their solutions are oriented to traditional networks.

3.2 Summary

Considering these particularities, it is necessary to emphasize that this report will solely address security solutions for the SDN architecture. In this sense, it was necessary to perform manual filtering by using keywords according to the interests of this document, until reaching the number of articles referenced in this work. For this purpose, the review of the abstract, research opportunities, and conclusions were used to determine whether or not to use the related articles.

Chapter-4

DESIGN METHODOLOGY AND ITS NOVELTY

4.1 Methodology and goal

Traditional networks have been a significant contribution to the telecommunications world. Nevertheless, due to the high demand for cloud services, traditional networks are becoming decreasingly flexible, programmable, and centrally managed for companies wishing to expand, such as telecommunications operators (TOs). With this background, the SDN paradigm has been pro- posed, which aims to provide alternative solutions to the current limitations of traditional networks. The SDN concept originated approximately two decades ago. Thus, between 1990 and 2000, active networks appeared, which provided programmable functions. Subsequently, between 2001 and 2007, the separation of the data and control planes was observed through open interfaces; since 2007, work has been done on an open protocol to achieve communication between the mentioned layers. Currently, the Open Networking Foundation (ONF), a nonprofit organization, is leading the SDN paradigm project. ONF has approximately 130 members, including companies, such as Facebook, Google, Deutsche Telekom, Microsoft, Yahoo!, and equipment manufacturers, among others. According to the ONF, SDN presents an architecture composed of three planes or layers: the data or forwarding plane, the control plane, and the application plane. The updating and exchange of information between the planes (data, control, application) are via interfaces. There are three interfaces in the SDN architecture: the southbound inter- face, which links the control plane to the data plane; the northbound interface, which links the control plane to the application plane; and the east-/westbound interface,

which interconnects the distributed controllers. Figure 1 shows the SDN architecture explained above. SDN provides a centralized view of the entire network through the controller, which simplifies the management of the nodes that compose it. It also makes it possible to maintain a considerable number of applications or network abstractions that can be developed by controller vendors or by third parties and that can be shared dynamically. All this can be reflected in reduced operating costs (OPEX) and capital expenditure (CAPEX) since it is no longer necessary to have a hardware or software manufacturer specialized in one brand. Due to its advantages, the SDN architecture is currently being used in the deployment of faster services, for example, 5G networks. However, it is recognized that in a large- scale implementation, it is necessary to define limits and security policies in the sharing of information and resources because SDN, like many other technological paradigms, presents security threats, several of which are caused by its layered architecture.

Software Defined Networking applies software engineering concepts such as modularity, abstraction, and separation of concerns to make flexible, scalable and efficient networks. Software evolved from a monolithic bloc of functions, data, and interfaces running on dedicated sealed hardware into decoupled and flexible modules that are hardware agnostic. This evolution was possible because of the introduction of operating systems, the standardization of the different interfaces, the separation of data from functions and the abstraction of hardware complexity. SDN operates with the same idea. It breaks the vertical integration between the control layer and the data layer in networks. It separates the data plane layer from the control layer. It centralizes the control layer logically in a software entity called the controller. Furthermore, it introduces programmable interfaces between the different new layers to offer applications network abstractions

while making them able to reprogram the network devices. SDN is the softwarization of the network. SDN enables network software to change and control the behavior of network infrastructure independently of the underlying hardware. The software can be business applications, management applications, security applications or other network services. The software expresses their logic in high-level languages and policies without the constraints of hardware details. Then, these high-level rationales are interpreted through open programmable SDN interfaces to hardware specifications. The latter can be forwarding behavior, infrastructure configuration, resource requests, and other low-level specifications. Finally, network elements change their behavior and states according to these specifications. SDN is founded on four characteristics. Externalization of control decouples the data plane layer from the control layer. Centralization centralizes the control layer logically in an external software entity. Federation opens and standardizes SDN interfaces. It guarantees the interoperability between applications, controllers, and network elements. Programmability automatizes the decisions of network application and the controller. It enables them to change the states and behaviors of network elements dynamically.

Concerning security, the controller is the most attractive SDN entity for attackers. The reason is related to the role of the control layer in SDN. The controller is the brain of the network. If an attacker takes hold of the control layer; he will seize all the rest of the network. He can corrupt the behavior of network elements. He can modify network traffic. He will be able to expose network applications and even other controllers. An attack can also put down the control layer by targeting its availability. In this case, the entire SDN network availability will be affected. Due to the sturdy dependability and reachability between SDN layers, an attack on a layer can affect the other layers. Depending on its sophistication and complexity, it can open

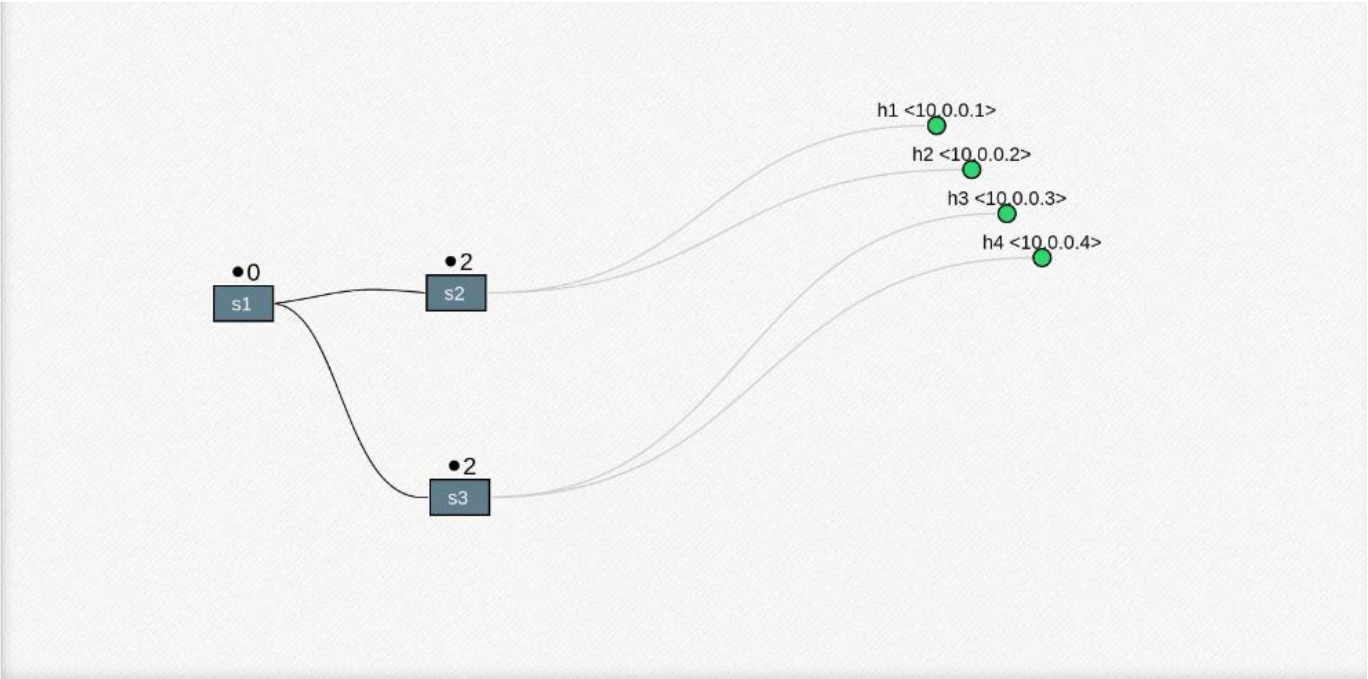
new breaches towards other layers. Also, it can reduce their performance. An attacker can exploit API's vulnerabilities to access SDN resources in different layers. He can modify network traffic and even SDN functionalities. Also, he can insert itself in the middle of two layers to snoop the communications between SDN entities. Therefore, interfaces need to integrate into their heart security mechanisms such as access control, encryption and integrity checks to protect the network. Network elements rely entirely on the control layer. The latter specifies their forwarding behavior. In the case of the connectivity between both layers failed, network elements will not be able to handle network traffic because they do not have any intelligence. In this case, the forwarding and routing capabilities of network elements become inoperative, and they induce the network elements to drop the traffic. As a result, the network becomes unreliable. An attacker can exploit this issue to take control of network elements. For example, it initiates a first threat that disconnects a network element from its controller. Then, it connects an impersonated controller to this network element to command it. As a consequence, the attacker reprograms all the connected network elements.

4.2 Functional modules design and analysis

Modules	Description
Math	Math is imported for performing mathematical functions such as finding log values
Pox	Pox.core module comprises of POX's core API and functionalities
Mininet	Network Emulation tool
Ubuntu	Linux Distro for handling SDN command
core.getlogger	This function returns a logger, which is the root logger of the hierarchy

Table 1. Module Description

4.3 Software Architectural designs




```
rajat@rajat-VirtualBox: ~                    kabir@rajat-VirtualBox: ~/Desktop/security-issues-in-SDN
rajat@rajat-VirtualBox:~$ sudo mn --topo tree,2,2 --controller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=12719>
<Host h2: h2-eth0:10.0.0.2 pid=12721>
<Host h3: h3-eth0:10.0.0.3 pid=12723>
<Host h4: h4-eth0:10.0.0.4 pid=12725>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=12730>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=12733>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None pid=12736>
<RemoteController c0: 127.0.0.1:6653 pid=12711>
mininet> links
s1-eth1<->s2-eth3 (OK OK)
s1-eth2<->s3-eth3 (OK OK)
s2-eth1<->h1-eth0 (OK OK)
s2-eth2<->h2-eth0 (OK OK)
s3-eth1<->h3-eth0 (OK OK)
s3-eth2<->h4-eth0 (OK OK)
mininet>
```

Chapter-5

TECHNICAL IMPLEMENTATIONS AND ANALYSIS

5.1 Outlines

Our project is based on testing and finding issues in Software Defined Networking. This chapter will provide you a detailed information about the different issues in sdn.

5.2 Technical Coding and code solutions

5.2.1 NORTHBOUND INTERFACE & APPLICATION PLANE

The application plane provides a set of programs (applications/abstractions) that are essential to satisfy the system's own needs, making it possible to generate or to respond to the requirements of the SDN environment through the controller. Currently, there is a wide spectrum of applications, including firewalls, routing policies, protocols, etc., which can be provided by the controller's developer itself or by third parties.

To solve authorization problems in applications, finegrained and coarse-grained access controls have been proposed. Coarse-grained access controls are generally used to cover the perimeter vulnerabilities of a standalone system or application; in other words, they can be very useful in invariant environments where habitual behavior is almost predictive. Within this category, RBAC, MAC, and DAC, among others, can be mentioned. On the other hand, there are fine-grained access controls, that is, controls that have greater granularity and detail in application permissions,

which are very useful in more dynamic environments.Something to keep in mind is that having multiple applications coming from third parties can cause interferences between them, generating conflicts in network policies. Such interference can come from an attack or lead to one. Given these scenarios, there are proposals, such as MSAID, which through algorithms used on the same SDN application code, is capable of detecting interference; this is also the case for SAIDE, a proposal to detect and eliminate interference by using mathematical models. With these premises, it can be argued that in the absence of the proper authentication and access control mechanisms that validate the origin of the applications, a northbound interface allows trust relationship attacks to be carried out towards the controller, many of which are derived from impersonation. At the same time, the absence of effective prior authentication and access control mechanisms for SDN applications exposes the network to handle illegitimate requests

5.2.2 SOUTHBOUND INTERFACE & DATA PLANE

The data plane and the control plane communicate via the southbound interface using protocols such as OpenFlow, OVSDB, OpFlex, NETCONF, and ForCes, among others. The protocol most widespread and currently studied is OpenFlow, already considered a standard according to. By itself, OpenFlow has no security mechanisms. Secure communication connections can optionally be established between the OF switch and the OF Controller via TLS or plain TCP for the prime connections or through TLS, DTLS, TCP, or UDP for the secondary connections. The ONF recommends TLS from version 1.2 onwards, despite its

vulnerabilities

Some research has determined the existence of security problems in the communication channel. In and, the authors argue that the lack of use of certificates at handshake time in the authentication process on the client side and the TLS protocol misconfiguration can lead to MiM attacks. On the other hand, Benton et al. indicate that the lack of TLS configuration increases the risk in the switches, since by not having authentication and, in many cases, having the “listening mode” active, the attacker could have access to the forwarding information and rules. In this sense, to mitigate the MiM attack in the southbound interface, in, the authors propose an extension to the TLS protocol. The authors of employ the IBC protocol to secure communications in the Southbound Interface and the data plane.

5.3 Test and Validation

Summary of contributions for northbound interface and application plane.

Ref.	Addressed Problems	Main Solution Approaches	Literature Contribution					
			S	T	R	I	D	E
[30]	Malicious apps	SSBG + Machine learning	✓	✓		✓	✓	✓
[31]	Malicious apps	CFG behavior analysis		✓				
[42]	Lack of access control Cross-domain in multidomain scenarios	Modified RBAC Cross-domain role mapping method						✓
[44]	Illegal modification of essential database information Lack of assigning permissions control	Blockchain	✓	✓	✓	✓	✓	✓
[45]	API abuse	Northbound security extension IDS Policy engine	✓		✓	✓		✓
[46]	Lack of dynamic access control	Data Extraction: Daikon Decision engine: API hooking						✓
[47]	Lack of initial access permissions assignment to apps	Behavior-based access control method + IDS			✓			✓
[48]	API abuse, DoS, Malicious apps	Publish/Subscribe model	✓	✓			✓	✓
[49]	Lack of authentication mechanisms between the app and the Controller	Independent mechanism	✓		✓			✓
[50]	Lack of authentication mechanisms between the app and the Controller	Floodlight-based mechanism	✓					
[51] [52]	Nonencrypted communication between app and controller Lack of management over reliability Lack of access control	Use of REST-like functions	✓	✓	✓	✓		✓
[53]	Lack of third-party app authentication	NTRU encryption algorithm and NSS digital signatures	✓	✓				
[54]	Malicious apps	NTRU encryption algorithm	✓	✓				

S=spoofing, T=tampering, R=repudiation, I= information disclosure, D=denial of service, E=elevation of privilege

Summary of contributions for southbound interface and data plane

Ref.	Addressed Problems	Main Solution Approaches	Literature Contribution					
			S	T	R	I	D	E
[112]	MiM attack communication channel	TLS protocol extension		✓			✓	
[74], [114]	MiM attack communication channel	IBC protocol		✓			✓	
[124]	Network topology information poisoning (Host location hijacking attack and link fabrication attack)	Preconditions/ Postconditions	✓	✓			✓	
[57]	Network topology violation and DoS attacks	Flow Graphs		✓			✓	
[127]	Fake link injection	LLDP payload size check		✓			✓	
[126]	Network topology information poisoning (Port probing and port amnesia)	Preconditions/ Postconditions	✓	✓			✓	
[123]	OFDP vulnerable	BFD	✓	✓		✓	✓	
[125]	Fake packet-in	Switch port association with host MAC	✓				✓	
[135]	Lack of packet-in message authentication	Independent hardware implementation	✓					
[136]	DoS attacks	Statistics					✓	
[137]	DoS attacks	Protocol-independent defense framework					✓	
[128]	Spoofing and DoS attacks	ACL / Machine learning	✓				✓	✓
[155]	Spoofing Route and Dos attacks	Traffic statistics	✓				✓	
[138]	DDoS attacks	Entropy					✓	
[140]	DoS attacks	Entropy					✓	
[141]	DoS attacks	Traffic statistics					✓	
[142]	DDoS attacks	KPCA+GA+ Machine learning					✓	
[143]	DDoS attacks	Blockchain					✓	✓
[144]	Lack of P2P traffic identification	Machine learning					✓	
[145]	HTTP DDoS attacks	Entropy + Hardware					✓	
[149]	DDoS attacks	PCA					✓	
[150]	DDoS attacks	EWMA					✓	
[151]	DDoS attacks	Snort IDS					✓	
[152]	DDoS attacks	Machine learning					✓	
[153]	DDoS attacks	Deep Learning					✓	
[154]	DDoS attacks	Entropy / Machine learning					✓	
[156]	Inference attacks	Randomization of network attributes/ Rate-limiting + Proactive rules Rate-limiting + Proxy					✓	
[160]	DoS attacks (LDoS)	Statistics / LRU					✓	
[130]	Inference attacks	Routing aggregation / TCAM + SRAM					✓	
[161], [162]	Lack of network client access control	EAP / RADIUS						✓
[163]	Lack of network client access control	EAPoL / RADIUS	✓					✓
[164]	DoS attacks	Blockchain + Hardware					✓	
[129]	SYN flooding and ARP spoofing attacks	SYN/ACK and ACK/FIN packets' ratio / P4 cache	✓				✓	
[165]	Traffic overload / Latency	App+P4					✓	
[166]	Traffic overload	Snort IPS + P4					✓	
[167]	DDoS attacks	P4+Entropy+FSM					✓	
[168]	Lack of link protection between stateful switches	MACsec	✓	✓		✓		✓
[169]	States exchange between stateful switches	Digital signatures	✓					✓
[170]	Link flooding attacks stateful data plane	Topology obfuscation					✓	
[171]	DDoS attacks	Machine learning + P4					✓	

S=spoofing, T=tampering, R=repudiation, I= information disclosure, D=denial of service, E=elevation of privilege

5.4 Summary

Existing studies have proposed different solutions for various scenarios and security issues of SDN architecture. Nevertheless, security remains a challenging research area with many unresolved questions. In this section, we will discuss some open issues and future directions that may merit research attention.

Chapter-6

PROJECT OUTCOME AND APPLICABILITY

6.1 Outline

This research paper talks about the Security-issues in software defined network

This chapter will give you an idea about the applicability of the project and its significance in Real-World Applications.

In the upcoming section, we will provide a short guide on how to use it so that anyone can use it effectively and give their feedbacks to us.

6.2 Security Solution

Many security solutions have been developed using the SDN benefits that were mentioned earlier. SDN offers three features to enhance security solutions:

- SDN offers the possibility to plug the northbound API inside the software architecture of a security application. In this case, the developers include in the code of their security applications the northbound API parts that enable their applications to interact with SDN controllers.
- SDN offers Domain Specific Languages (DSL) to applications and users. These DSLs are used to express security policies and re-program network elements according to these policies.
- Some SDN controllers offer programming environments such as development frameworks, libraries, plugins, and resources to develop security applications inside the controller. These security applications behave like controller functions. They use controller services. They access the global controller knowledge. They program the network elements directly.

6.2.1 Authentication & Authorization Solutions

Authentication & Authorization Solutions based on SDN Authentication & Authorization solutions based on SDN have been developed as OpenFlow applications. For example, proposes an authentication and access control mechanism called AUTH Flow. This OpenFlow application authenticates hosts on the control data link layer. A host starts EAP authentication according to IEEE 802.1X standard with AuthFlow's Authenticator. The latter de-encapsulates the authentication messages. It validates the authentication against the Radius server. Then, it sends the results to the AuthFlow application running in POX controller. If the host is successfully authenticated, the AuthFlow application determines its access to SDN components using its credentials and identity.

Flowidentity is another authentication solution based on SDN. It uses the same authentication mechanisms of AuthFlow. Besides, it introduces an access authorization enforcement module based on a role-based firewall. This module can dynamically update the security policies of authenticated sessions. Thanks to SDN global knowledge, automation, and agility, FlowIdentity performs dynamically stateful policy updates to existing sessions unlike in the legacy network.

By contrast, the Authentication and Authorization application in does not use IEE 802.1X software. This SDN application uses the northbound API to perform authentication and authorization of Machine To Machine (M2M) communications. It authenticates hosts based on their credentials. It allocates a bandwidth to each host. It authorizes the host to access other hosts based on an access list. It sends the high-level specification to the controller. The latter interprets them into OpenFlow rules and installs them on the network elements. When the host logs out, it uninstalls all its OpenFlow rules.

6.2.2 Moving target defenses based on SDN

Moving target defense technics defend the network against reconnaissance attacks. They change continuously in time the properties of the network to prevent or make difficult for an attacker to learn the network resources and their vulnerabilities. As a result, the attacker will spend more efforts and time to look for the network vulnerabilities. SDN enables active MTD thanks to the combination of SDN agility and global knowledge. The former enables SDN to adapt its MTD strategy seamlessly with the evolution of the network state and the attack. The SDN global knowledge leverage MTD with a holistic view that improves the reliability of its operations. As a result, MTD strategy is enforced according to

an end-to-end security strategy. For example, OF-RHM is an MTD solution based on SDN. It performs a host transparent IP address replacement by hiding real IP addresses with virtual IP addresses. It associates to the real IP addresses short living virtual IP addresses that are updated according to a time interval. The solution selects randomly from the unused IP address space these virtual IP addresses. Then, it installs the OpenFlow rules that perform the IP address mutation in all the network elements without affecting the end host. By the use of SDN features such as agility, automation, and global knowledge, OF-RHM mutates IP addresses with high unpredictability and speed to distort the attacker knowledge.

6.2.3 Firewall solutions based on SDN

A firewall is a security mechanism that protects a network from unauthorized access. It filters the traffic by matching the packets' headers with a set of security policies to allow only the current traffic to access the network. SDN elevates firewalls with all its features. Simplification enables administrators to manage firewall policies without worrying about the complexity of the underlying infrastructure. Agility adapts firewall rules dynamically according to the network state. Global knowledge enables firewall applications to spread their rules all around the network. Interoperability enables firewall applications to install their rules and collaborate through a common northbound API. Finally, automation enables a firewall application to install its rules autonomously and manage them without the intervention of administrators.

There are two categories of firewalls based on SDN. Full-SDN-Firewalls rely on the control layer to express their behaviors and to collect network state. Hybrid-SDN firewalls are independent of the controller. The details of all these firewalls are as follows:

1. Full-SDN-Firewalls Full SDN firewalls take advantage of SDN to secure the network. Mainly, their behavior is expressed using the southbound API features. This API can reprogram the network elements and collect network events. They do not depend on a specific data plane technology. Besides, their implementation can be independent of the control layer thanks to the northbound interface. However, the controller manages their behaviors and interactions. There are four types of Full-SDN-firewalls. Control function firewalls are firewall functions that are implemented inside the control layer. SDN application firewalls are implemented in the application layer. Extended OpenFlow firewalls extend OpenFlow features to support firewall behavior in OpenFlow. Finally, SDN firewall

frameworks can be used to construct firewall functions in the control layer.

2. Control Function Firewalls: They integrate a firewall module as a controller function. The stateless firewall denies or permits the traffic without managing the dynamic properties of connections such as its state or port allocations. In these Firewalls, the administrator configures the controller with security policies. Then, the firewall function interprets them into OpenFlow rules. It calls the OpenFlow primitives directly. The OpenFlow Agent constructs the appropriate rules and installs them on the network elements. SDN controllers such as RYU, Floodlight and POX integrate an OpenFlow stateless firewall as a control function.

3.Applications layer firewalls: SDN firewall applications are developed independently from the control Layer. Principally, they are integrated into the application layer. They interact with the controller through the northbound interface. They send the security policies (expressed in a high level of abstraction) to controllers that interpret them into OpenFlow rules and install them on the network elements. Their implementation, maintenance, and upgrade neither depend on the source code of the control layer nor affect it.

4.Extended OpenFlow Firewalls: They extend OpenFlow protocol to handle a part of the firewall behavior in the data plane layer. This behavior is mainly related to connection state processing. For example, FleXam extends the OpenFlow with a new artifact that can build firewall functions. It adds to OpenFlow a channel that specifies flows filters, sample schemes (parts of packets that will be sampled) and new actions to filter and sample flows. FleXam forwards all samples to controller systematically to inform it about the results of the sampling. Then, the controller verifies them with firewall policies and installs the associated OpenFlow rules.

5.SDN firewall frameworks: Some researchers propose SDN security frameworks that can implement SDN Firewalls. However, these frameworks are not appropriated for designing stateful Firewalls. They only allow the construction of stateless firewalls. Besides, they neither use nor share the global knowledge. They do not enable multiple firewall applications to collaborate in different domains to reinforce the security policies. Besides, their maintenance is fastidious because it affects the code of the framework and the controller that supports them.

6.Hybrid-SDN-Firewalls: they introduce control function into the network elements or collaborate with the legacy architecture. The controller in these firewalls only installs the OpenFlow rules to direct the traffic towards them and to resubmit the ingress traffic coming from them into the OpenFlow pipeline. As a result, the control layer has neither visibility nor any controllability on the traffic processing inside these firewalls. The different types of these firewalls are as follows.

7.Network element firewalls: They integrate firewall functions in network elements by modifying the data plane architecture. They are powerful because they process the traffic directly in the network elements with dedicated devices. Their hardware structure is adapted to process network traffic and to perform firewall functions. However, the control layer depends on them to know which traffic has been processed and how to manage it. Unfortunately, there is not an SDN controller able to control these firewalls because they require extensions in the southbound API. The solution adds a new action to OpenFlow to send the flows to the Contrack modules. These modules process the traffic according to its pre-configured security policies. The processing steps in this firewall are as follows:

i. The flow that corresponds to an OpenFlow entry with a Contrack action is sent to the Contrack modules.
ii. The Contrack module inspects them to determine their state.
iii. It updates the state connection tracking tables with the new state.
iv. Contrack module forwards the flow with its state to the OF table.
v. The flow is recirculated inside the flow tables to match with another OF Rule.

8.Collaborative firewalls: They connect hardware or virtual firewalls (ex. middlebox) to network elements to filter network traffic. In this case, the controller installs the appropriate OpenFlow rules to forward the traffic to the middlebox and to resubmit the ingress flows coming from the middlebox inside the flow tables. Collaborative firewalls reuse existing legacy firewalls with SDN; however, their behaviors can neither be controlled nor be automatized by SDN. Their architecture induces a hop for each link between a middlebox and a network element. These additional routes increase traffic delays because the network elements send traffic to the middle-box and wait for its processing before forwarding it. Besides, the controller has no visibility on their behavior and their security policies. Thus, they cannot use the global knowledge. Moreover, because they do not collaborate in a distributed environment or a multi-domain network, problems of security policies inconsistencies can arise. For example, a middlebox in a domain A can block traffic which is authorized in a domain B.

6.3 Significant project outcomes

This project reports shows Research on Security Issues in Software Defined Networking

6.4 Project Applicability in real-world

Following are the benefits or **advantages of SDN**:
➡It enables centralized management of networking devices.
➡It helps in automation of networking devices.
➡It provides improvements to end users.
➡It offers flexibility, scalability and efficiency compare to traditional networking.
➡It is widely used by social networking websites (facebook, twitter, google plus etc.) and large database search engines

6.5 Inference

In this chapter, we gave a description of the benefits of advantages of SDN and the impact it might have in the field of networking in the future

Chapter-7

CONCLUSIONS AND RECOMMENDATION

7.1 Outline

For this project, we have explored, studied and developed the relationship between SDN and security. We have presented in chapter 2, SDN principles, concepts, architecture, its challenges, and benefits. We have also introduced the features of OpenFlow. Then, we have studied the literature on the relationship between SDN and security. We have shown on the one hand that SDN features enlarge the attack surface. As a result, SDN assets are vulnerable and can even be even used as attack vectors to attack other SDN assets. We showed that the controller is a single point of failure because it concentrates all the SDN features at the same time. On the other hand, we have seen that SDN improves security applications thanks to its advantages such as simplification, independence, agility, global knowledge, convergence, automation, and orchestration. We showed that SDN is an enabler for security applications. It improves monitoring, detection, prevention, mitigation and defense mechanisms.

7.2 Future Enhancements

Despite the great security challenges presented by the SDN architecture itself, its use by other paradigms and technologies to address security problems is not ruled out. In general, it has been noted that several authors highlight the value of the centralized

control that SDN possesses to become an ally in the detection of attacks. For this reason, there are security solutions that make use of SDN to improve their defense mechanisms, while others opt for the joint participation of Network Functions Virtualization (NFV) and SDN to protect against malicious actions in a world where devices are heterogeneous, as in the case of IoE environments or cloud security. Blockchain has also considered working with SDN either to combine the above paradigms and technologies or to create an SDN trilogy-Blockchain and OpenStack to achieve secure resource and service sharing outcomes, which could be leveraged in multi-vendor environments.

7.3 Inference

At present, the research on the security and vulnerability of SDN network is still in its infancy. This report first analyzed the SDN technical architecture principle and the development present situation. We researched the security characteristic and the vulnerability question in the SDN structure, analyzed its vulnerability in the control level and the application level and proposed the corresponding vulnerability question judgment model. On this basis, the architecture of the protection control architecture is constructed. This architecture explore controllable and manageable problems of network, which is expected to promote the further research on the security and vulnerability of SDN network.

REFERENCES

[1] Lori MacVittie and David Holmes. The new data center firewall paradigm. F5 Networks, Inc., Seattle, 2012.

[2] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, “Security in Software Defined Networks: A Survey,” oct 2015.

[3] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodol- molky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” Proceedings of the IEEE, vol. 103, pp. 14–76, jan 2015.

[4] J. H. Cox, J. Chung, S. Donovan, J. Ivey, R. J. Clark, G. Riley, and H. L. Owen, “Advancing software-defined networks: A survey,” IEEE Access, vol. 5, pp. 25487–25526, oct 2017.

[5] S. Khan, A. Gani, A. W. Abdul Wahab, M. Guizani, and M. K. Khan, “Topology Discovery in Software Defined Networks: Threats, Taxonomy, and State-of-the-Art,” IEEE Communications Surveys and Tutorials, vol. 19, pp. 303–324, jan 2017.

[6] T. Dargahi, A. Caponi, M. Ambrosin, G. Bianchi, and M. Conti, “A Sur- vey on the Security of Stateful SDN Data Planes,” IEEE Communications Surveys and Tutorials, vol. 19, pp. 1701–1725, jul 2017.

[7] T. Han, S. R. U. Jan, Z. Tan, M. Usman, M. A. Jan, R. Khan, and Y. Xu, “A comprehensive survey of security threats and their mitigation techniques for next-generation SDN controllers,” in Concurrency Com- putation, vol. 32, p. e5300, John Wiley and Sons Ltd, aug 2020.

[8] J. C. Correa Chica, J. C. Imbachi, and J. F. Botero Vega, “Security in SDN: A comprehensive survey,” jun 2020.

[9] R. Khan, P. Kumar, D. N. K. Jayakody, and M. Liyanage, “A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions,” jan 2020.

[10] S. Ahmad and A. H. Mir, “Scalability, Consistency, Reliability and Security in SDN Controllers: A Survey of Diverse SDN Controllers,” Journal of Network and Systems Management, vol. 29, jan 2021.

[11] Open Networking Foundation, “SDN architecture,” 2014.

[12] A. Danping, M. Pourzandi, S. Scott-Hayward, H. Song, M. Winandy, and Z. Dacheng, “Threat Analysis for the SDN Architecture,” Tech. Rep. July, 2016.

[13] Open Networking Foundation (ONF), “Principles and Practices

for Se- curing Software-Defined Networks," tech. rep., 2015.

[14] K. Bissell and L. Ponemon, "Ninth Annual Cost of Cybercrime Study. Unlocking the Value of Improved Cybersecurity Protection," tech. rep., 2019.