

Open University of Cyprus

School of Applied Sciences

**Postgraduate Thesis
In Computer and Network Security**



Providing Network resilience through an intelligent SDN network.

Georgios Lialiaris

Supervisor

Dr. Adamantini Peratikou

May 2021

Summary

SDN is an architecture designed to make a network more flexible and easier to manage. SDN consolidates management by removing the control layer from the data forwarding function on discrete networking devices. This architecture allows networks to connect directly to applications via APIs, enhancing application performance and security as mentioned above. In addition, using SDN a network engineer or administrator can configure traffic from a central control console without having to deal with switches or routers on the network. The central controller SDN directs the switches to provide network services where needed, regardless of the specific connections between server and device.

In view of the above, let me mention that traditional networking is based on fixed network devices, such as a switch or router and other physical infrastructure for networking and network operation, as opposed to software-based SDN. Therefore, the most noticeable difference between SDN and traditional networking is that SDN is software based, whereas traditional networking is usually hardware based. Because it is software based it is more flexible, allowing users more control and ease of managing resources at almost the entire control level. In this way, SDN has become a popular alternative to traditional networking, as it allows IT administrators to provide resources and bandwidth without the need for additional physical infrastructure investment. A traditional networking requires new hardware to increase network capacity.

In terms of security, SDN provides a variety of benefits. An administrator can split a network connection between an end user and the data center and have different security settings for different types of network traffic. A network could have a low-security public network that does not touch sensitive information. Another section could have much more detailed remote access control with firewalls based on software and encryption policies to enable sensitive information.

SDN also introduces new capabilities for network management and configuration methods. For example, SDN has the ability to obtain instantaneous network status so that it allows centralized control of a network in real time based on both the instantaneous network status and user-defined policies. This leads to additional benefits in optimizing network settings and improving network performance.

Finally, the Internet of Things (IoT) and Network Defined Network (SDN) are two emerging technologies. IoT aims to connect objects over the Internet, and SDN provides orchestration for network management by disconnecting the control layer and the data layer. The number of connected objects is in the billions, and managing and controlling it is a complex task for a large distributed network. SDN provides flexibility and programming capability on the IoT network without compromising the underlying architecture of existing applications.

Acknowledgment

I would like to thank Dr. Adamantini Peratikou for all the support and the guidance that she provided to me. In addition, her door was always open for solving any inquiries or questions I had about this research.

In addition, I would like to express my sincere gratitude for the comments, suggestions and for all the assistance at every stage of the project.

TABLE OF CONTENTS

List of Figures.....	vi
Introduction to SDN.....	8
1. Components of the SDN.....	12
1.1 Data Plane.....	13
1.2 Application Plane.....	14
1.2.1 Applications and Services.....	14
1.3 Control Plane.....	16
1.4 SDN Southbound Interface.....	17
1.5 SDN Northbound Interface.....	17
1.6 SDN Datapath.....	18
1.7 SDN Control to Data-Plane Interface (CDPI).....	18
2. SDN Architecture.....	19
2.1 Application Layer.....	19
2.2 Control Layer.....	19
2.3 Infrastructure Layer.....	20
3. Traditional Networks and SDN.....	21
3.1 What is Traditional Network.....	21
3.2 What is a Software Defined Network.....	21
3.3 Differences between SDN and Traditional Network.....	21
4. Related Works.....	23
4.1 Operation Checkpoint: SDN Application Control.....	23
4.2 OpenFlow vulnerability assessment.....	24
5.1 FireCol: A Collaborative Protection Network for the Detection of Flooding DDoS Attacks.....	24
4.4 Revisiting Security Aspects of Software-Defined Networking.....	24
4.5 OpenFlow: A Security Analysis.....	24
4.6 Resilience and Security in Software Defined Networking.....	25
4.7 Software defined networking for resilient communications in Smart Grid active distribution networks.....	25
4.8 Implementation challenges for software-defined networks.....	25
4.9 Software-Defined Networking and Distributed Denial of Service.....	26
4.10 Controller placement strategies for a resilient SDN control plane.....	26
5. Methodology.....	Error! Bookmark not defined.
5.1 Research questions.....	Error! Bookmark not defined.
5.2 Purpose of research.....	Error! Bookmark not defined.

5.3	Research method.....	Error! Bookmark not defined.
5.4	SDLC - Spiral Model.....	Error! Bookmark not defined.
6.	Testing and Implementation	27
6.1	Test Environment Preparation.....	27
6.1.1	Ubuntu Virtual Machine	27
6.1.2	ONOS Open Network Operating System	30
6.1.3	Install JAVA	30
6.1.4	Maven.....	31
6.1.5	Mininet.....	31
6.1.6	ONOS.....	33
6.2	SDN Testing and Evaluation.....	36
6.2.1	Crawl and Audit.....	37
6.2.2	Dictionary Attack using Burp Suite.....	41
6.2.3	Network Discovery	43
6.2.4	NMAP	44
6.2.5	Nikto	44
6.2.6	DDOS Attack.....	45
6.3	Traditional Networks.....	47
6.3.1	Nikto	47
6.3.2	Network Discovery	48
6.3.3	NMAP.....	48
6.3.4	DDOS Attack.....	49
6.3.5	Dictionary Attack.....	51
7.	Discussion	52
8.	Conclusion	54
	8.2 Limitations & Future work.....	55
	References.....	56

List of Figures		
Figure	Explanation	Page
1	Components of the SDN	10
2	Data Plane comparison	11
3	How application plane works	13
4	Centralized architecture of a controller	14
5	Southbound and Northbound API	15
6	SDN Datapath	16
7	CDPI	16
8	The three layers in SDN architecture	18
9	Differences between SDN and traditional Network	20
10	Phases of the Spiral model SDLC	25
11	Version of Virtual box	28
12	Creation of Virtual Machine	29
13	Memory required for the Virtual machine	29
14	Hard Disk required for the Virtual machine	29
15	How Virtual Machine looks after installation	30
16	Installation of Java in Ubuntu	31
17	Version of Java installed	31
18	Installation of Maven and version	31
19	Installation of Mininet	32
20	Test functionality of mininet	32
21	Ping hosts over mininet	33
22	ONOS version	33
23	Installation of ONOS	34
24	IP for SDN Controller	34
25	Commands for starting ONOS service	34
26	Service of ONOS started	35
27	Node of SDN controller	35
28	Web Interface of ONOS controller	35
29	Spine Leaf Topology	36
30	Hosts created in topology	36
31	Cluster of controllers	36
32	Scope of the attack	37
33	Results of crawl report	37
34	Cleartext submission of password	38
35	CWE-319	38
36	Password field with autocomplete enabled	39
37	CWE-200	39
40	Cookie without HttpOnly flag set	40

41	CWE-16: Configuration	40
42	Payloads with usernames	41
43	Payloads with passwords	41
44	Username and password match	42
45	Raw data proving the correct username and password	42
46	Netdiscover command	43
47	Network Information about cluster	43
48	Results of NMAP	44
49	Scan results of Nikto	44
50	Performing DDOS attack	45
51	State of the machine before DDOS attack	46
52	State of the machine after DDOS attack	47
53	Results of Nikto in traditional network	47
54	Results of Netdiscover in traditional network	48
55	Results of NMAP in traditional network	49
56	Performing DDOS attack in a traditional network	49
57	State of the machine before DDOS attack traditional network	50
58	State of the machine after DDOS attack traditional network	50
59	Hydra-gtk	51
60	Hydra command line	51

Introduction to SDN

Software-defined networking (SDN) technology is a methodology to network management that allows dynamic, programmatically capable network configuration to improve network performance and monitoring, in order to make it more like cloud computing than traditional network management. [3] SDN was stating that traditional networks are decentralize and complex and that there is place for improvements to be more flexible in order to have easier troubleshooting.

The way to achieve SDN functionality and centralize network intelligence is to isolate the data plane from the routing process. Data plane is the process of the network packets and the control plane is routing process. In a few words, when SDN isolate data plane from the control plane will integrate network intelligence in just one network component. By enabling this cloud computing in network, network team can be more flexible to respond to changes that are required from an organization through the control console that will be separate from the physical hardware. [16] The main reason to enable this is because manage and evolve networks is hard, so we need new ways to achieve that. Briefly, SDN will have a centralized intelligence in order to be able to command the network. [15]

Due to this architecture, SDN can be the perfect model for high-bandwidth application as is manageable and adaptable. Furthermore, the architecture of SDN was been designed to be directly programmable because data plane is isolated from the routing functions. [29] In addition, SDN can help to adjust dynamic flow dynamically and as we mentioned above, SDN can centrally managed. Furthermore, many SDN programs can let administrators configure, secure, or even optimize networks. Considering these characteristics, we can see that SDN architecture can guarantee further technology deployment regarding network resources. [9] [25]

Taking all the above into account, SDN can promise not only technical but even commercial benefits. First, SDN can reduce capital expenditure because there is no need of buying specific purpose network devices as SDN supports pay as you go and scalable models. In addition, most of the switches support SDN technology and OpenFlow. [10] Another business benefit is that SDN can reduce operational expenditure because when using SDN, there it is not essential to change or replace the existing infrastructure if there are demands regarding network. The reason is that there are automations and they can reduce the tasks from network administrators. [21]

The history of SDN starts from the time that engineers tried to make things easier regarding the provisioning and management of the public telephone network by separating the control and data plane. Unfortunately, the tries failed then, because was viewed that splitting control from data could be unsafe mainly if there was a failure to the control pane. Another reason that failed then was that there was a concern about creating APIs among the control and data planes. Their main fear was that APIs would have increase in competition. [21] [27]

At first, SDN was related with the OpenFlow protocol, which is a communications protocol that gives access to the forwarding plane of a network switch or router over the network. This protocol started to flow in 2008 at Stanford University and since then was dedicated to SDN concept. OpenFlow protocol also enables the separation of the control so switches from various vendors can be managed remotely by using only this open protocol. In a few words, OpenFlow is the communication protocol for the entire SDN architecture. Additionally, OpenFlow allows to controllers to regulate what path a packet will take through a network because as we mentioned above, controllers are separated from the switches. [16]

Later on, a non-profit group named Open Networking Foundation (ONF) started to promote SDN through open source business model and this group began to manage OpenFlow protocol also. After that OpenFlow protocol was standardized and SDN was recognized for the innovations through cloud computing. [20]

Besides the above characteristics, we would like to mention that there are several architecture components, such as SDN applications, controllers, NBI and more, which needed for SDN to work as expected. The following chapters will describe comprehensive the main components of the SDN and its architecture.

1. Methodology

In this chapter, the problem statement, the research question and what methodology will be included and explained.

5.1 Research questions

1. Is software-based SDN technology capable of improving network security instead of the traditional hardware-based network (switches, routers and other physical infrastructures to create connections)?
2. How can SDN Network technology be used?
3. What are the existing applications designed to provide enhanced security through SDNs?
4. What are the existing solutions that provide better security?
5. Can SDNs enhance security?

5.2 Purpose of research

SDN (Software Defined Networking) technology will be analyzed to determine if it can offer network security improvements. In network security, research is needed to determine the degree of security of a new technology. Using SDN offers the possibility for more efficient and secure networks. The purpose of this dissertation is to determine whether SDNs can provide the required security in cyber range environments.

5.3 Research method

Quantitative research is used because the research will be mainly of a practical nature. We explored existing SDN-based solutions that provide better network security and studied through literature review the existing SDN architecture. Implement applications in a cyber-range environment for use in cybersecurity exercises.

The reasons that enforced my decision of using quantitative research first was that is based on numbers and it suits better for theoretical facts or assumptions, and it will enabled me to gather in depth information that I need. Also the literature review, as there are few of published works to conduct a survey for understanding this concept. In addition, quantitative is mostly use for numerical and statistical values. In the contrary, qualitative is collecting unstructured data at most cases are investigated in depth. Furthermore, qualitative is not as generalizable as quantitative and is a more subjective method. Quite the opposite, quantitative is objective based and will provide all

the observations that are interpreted from the researcher and usually the research is based on the measurement that are used.

5.4 SDLC - Spiral Model

There are plenty of software development life cycle such as agile model, waterfall, RAD and more. Spiral model has been used for this research, as is a more flexible model with fewer limitations than waterfall. It is progressive and it is a good choice for big projects as you can break this large project into smaller parts. Another fact was that spiral model can accommodate change of requirements during the research and this is expected during a project development life cycle.

Furthermore, spiral model can offer four phases for completing a project, the planning, the design, the implementation and lastly the evaluation. At the first phase, an understanding of the requirements and the objectives must to be identify. In the design phase, a design must be involved i.e. what modules or architecture will be used. During the implementation phase of the project, all the tests and plans will be develop. Lastly, in the evaluation phase, all the results and conclusions will be show. [44]

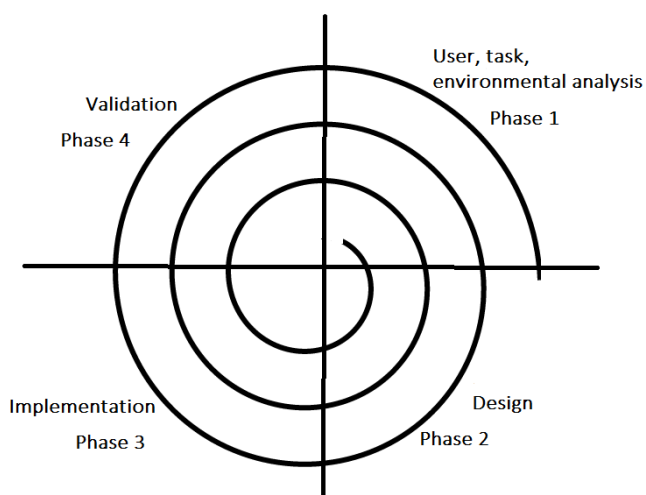


Figure 10 - Phases of Spiral model SDLC

In figure 10 all phases are shown clearly. Starting with phase 1 analysis, phase 2 design, phase 3 implementation and phase 4 validation. In complex projects, there is the possibility to return to another phase without disturbing the course of the project.

2. Components of the SDN

Like every characteristic technology product, SDN needs components in every layer in order to work as expected. In this chapter, we will define and explain the components of a software define network. Later in the chapter SDN architecture, we are going to connect those components in the architecture of the SDN that consists three layers. The application layer, the control layer and the infrastructure layer.

Software defined network is surrounded by several architecture components. We will define and explain the following:

- Data Plane
- Application Plane (SDN Application and Services)
- Control Plane (SDN Controller)
- SDN Southbound Interface
- SDN Northbound Interface
- SDN Datapath
- SDN Control to Data-Plane Interface (CDPI)

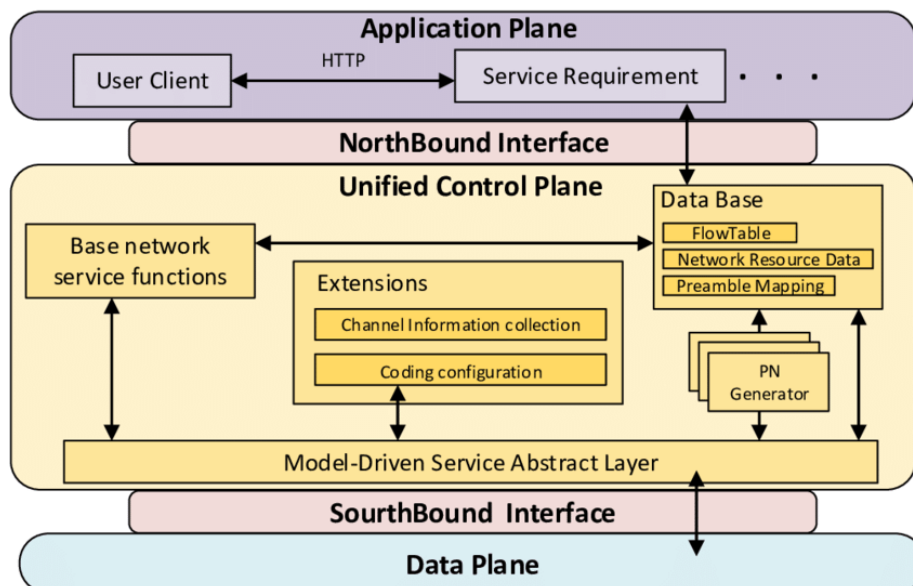


Figure 1- Components of the SDN

In figure 1, we can see how all these components connect to each other. We can see the application plane connecting with the control plane through the northbound interface and how data plane is connected to controller through southbound interface. All the above will be explain at a later stage in detail.

1.1 Data Plane

To begin with, data plane is responsible for forwarding the packets through all the networking equipment i.e. routers and switches. These routers and switches can be physical or virtual and their main duty and specialization is forwarding all the data. Furthermore, this equipment in SDN is dumb, so they are not able to take decisions and there is no some kind of intelligence in those machines as it is in the traditional networking. All the network equipment are communicating with the controller over the OpenFlow interfaces, so controller will certify all the configuration and communication needed between all these machines. [26] It is worth mentioning that software define network can support virtual network infrastructures and data centers through software switches.

Let us see a comparison between the traditional and SDN Data Plane in the figure 2:

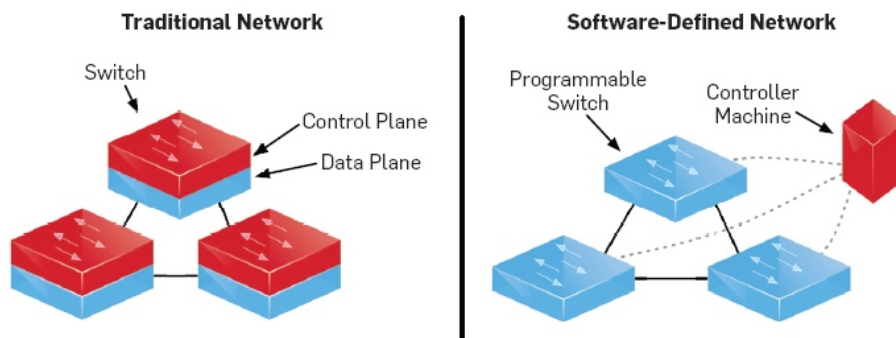


Figure 2- Data Plane comparison

From the figure 2, we can see that traditional network switches have autonomous intelligence and they can proceed with decisions like forwarding the packets. On the contrary, software defined network switches are dumb devices connected from their OpenFlow interfaces to the controller. Every forwarding OpenFlow machine has a forwarding table, which includes rules, actions to be taken about a packet, and counters in order to capture statistics. [30]

To explain further, rules are the fields like source IP, destination IP, port, source MAC, destination MAC, VLAN ID and more. Actions is the forwarding a packet to an outgoing port or the controller, drop, queue and more.

Furthermore, let me mention that when OpenFlow firstly started was a very restrictive standard. Nowadays is a very flexible standard providing rule flexibility, multiple flow tables and can provide stateful capabilities such as fast failover. [26]

1.2 Application Plane

The SDN application plane is the involvement of the network applications i.e. security, and cooperate with the controller in order to utilize the network to the most efficient point. This point depends from the decision making and implementing. In other words, application plane will direct the performance of the entire dump networking devices that connect with the controller. [10]

In a few words, SDN application plane include the programs that will require resources from controller via API's (Application Programming Interfaces). Therefore, the application will give policies to the control plane and the control plane will enforce these policies to the devices in order to implement them.

Many applications are being developed continuously, as there is a lot of dependability on these applications. For example, there are application that are related with the security, monitoring traffic and measurement, perimeter security, network management and much more. For example, you can use an application that was built for monitoring and use it for identifying suspicious networking traffic. All these types of applications are important for upgrading SDN capabilities to the next level that is why there is a lot of research and development around this field. [11]

1.2.1 Applications and Services

There are many applications and services regarding SDN, but in my opinion the below are the most important:

Security services. Nowadays, organizations are digitally transforming that is why the network security layer is vital. [25] More and more attacks are happening daily with ways that are more sophisticated and SDN can help to reduce and respond quickly to those attacks.

Management of the bandwidth. With this kind of SDN applications, administrators can monitor not only how much bandwidth and latency exists, but they can also monitor the requirement of the bandwidth and then to deliver the flow that will be need. With this way user's experience will be better because there will be no buffering or latency.

Performance of the Applications. Performance is a major impact to an infrastructure and as we, all know using the traditional way we will need many bare metal architectures in order to accommodate several heavy workloads. [25] Integrating SDN in the network layer, tasks similar to

heavy traffic, QoS policies, bottlenecks and more, will be better managed and help high performance applications.

Monitoring of the Network. Many infrastructures are complicated and as they are growing, are becoming more and more difficult to control and monitor. [25] Using SDN architecture, all functions as alerting, monitoring and traffic flow can be combine with the SDN technologies and get benefited from them. Besides the above, monitoring through SDN can happen also in cloud and not only in the on premises data center. [32]

Cloud Integration. SDN is capable of extending the on premises networks to the cloud. This can help a lot because you can have an easier movement of data between cloud and on premises network infrastructure. Also due to the fact that is the network is virtualized, you can control just the services that you need and in the location that you need. [25] [33]

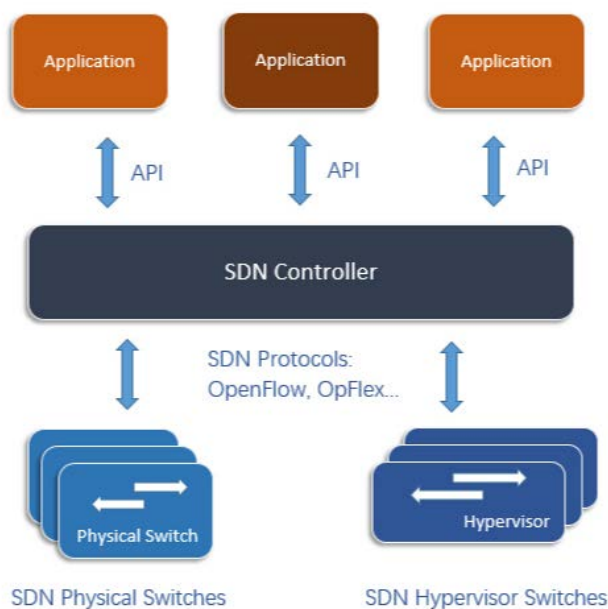


Figure 3 - How application plane works

In figure 3 we can see that the application plane is in the top and connected through API's with the controller and then controller is connected through SDN protocols to the switches.

1.3 Control Plane

The concept of the software defined network is to separate the forwarding process from the routing process. The routing process is consider the control plane and contains one or more SDN controllers. [22] There should be at least two controllers in order to have scalability and failover in case of a disaster recovery.

SDN controller push the settings based on policies, which have set by a network administrator, and we can think the SDN controller as the head of intelligence on the network. Specifically it makes the policies available to the application plane through the services and the API's. It is using northbound API's to connect with the applications and the southbound API's to communicate with physical or virtual network devices. [33]

SDN controller has two architectures, the centralized and the distributed one. Using the architecture of the centralized one there is only one responsible controller for managing all the devices that will forward the packets. [42] Due to the fact that is a single controller, it can cause a single point of failure and lacks scalability. The other architecture, distributed, is safer in a way that can handle logical or physical failures because there are more than one controller and can spread across the network. Both, northbound and southbound API's will be explained at a later stage. [26]

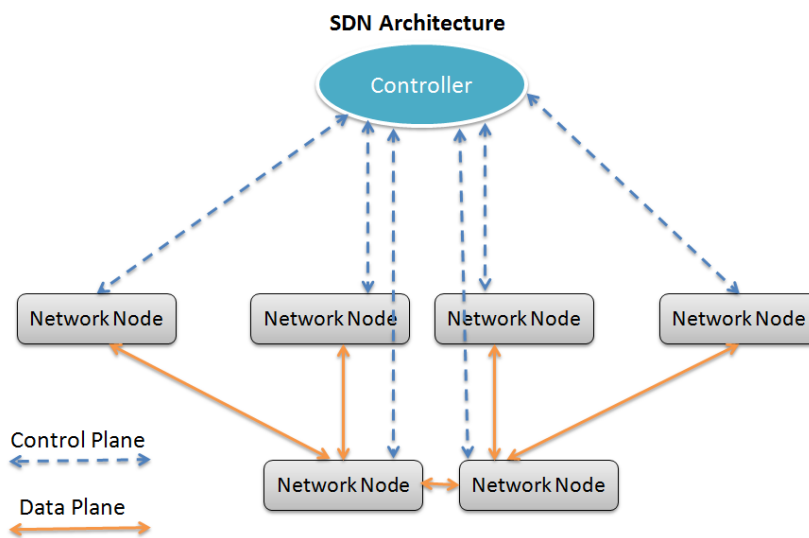


Figure 4 - Centralized architecture of a controller

In figure 4, we can see a single controller that controls all the network devices that are connect to each other via their interfaces. This a centralized architecture.

1.4 SDN Southbound Interface

SDN southbound interface or API is one of the most vital components of SDN technology. It connects the SDN controller with the network devices in a form of the OpenFlow communication. In this way, the controller is able to control the whole network because it will manage the network devices in real time. [31] Because there is a need of various protocols in order to manage both physical and virtual devices, southbound API is providing to controller a common interface so all these protocols to be able to coexist. Southbound API is using the OpenFlow protocol, as is the standard protocol for the southbound.

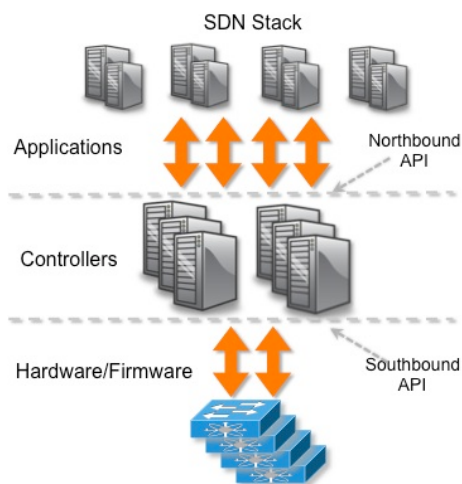


Figure 5 – Southbound and Northbound API

In figure 5, we can see the differences between a Southbound and a Northbound API. Northbound connects controller with the application and Southbound connects the controller with the devices.

1.5 SDN Northbound Interface

As we mentioned above, southbound API is the mail link between the controller and the devices. Now, northbound API is the link or interface between the controller and the applications. There are plenty of network applications such as firewalls, load balancers, application firewalls and more. [9]

1.6 SDN Datapath

Datapath is the representation on a software based for a network device. More example this representation may be a switch that traffic packets. In addition, you can imagine datapath as the logical presentation of all the physical network components that are subsidized from software. [16] In a few words, is the implementation of the devices that will move the packets through the network.

In figure 6, we can see clearly where the datapath exists and that is between the switch and the SDN controller.

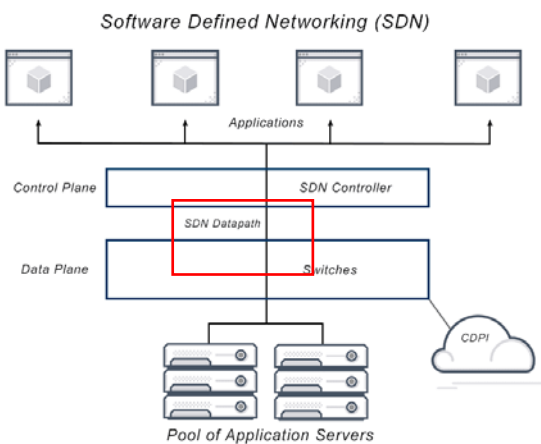


Figure 6 – SDN Datapath

1.7 SDN Control to Data-Plane Interface (CDPI)

This component is the interface that will link the SDN controller with the datapath. The CDPI is expect to be an open vendor and its responsibilities are to deliver all the programmatic control of the operations, reporting and alerting. [30]

In the figure 7 we can see that the CDPI is vital and important in order to link the controller with the datapath.

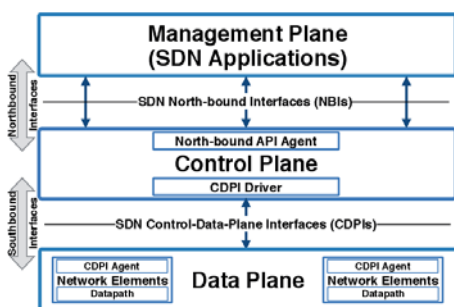


Figure 7 – CDPI

3. SDN Architecture

All technology solutions are produced with an architecture and a solution design. SDN has also an architecture that can be based on. Beginning with, software defined network is based is three layers. The application layer, which are the applications such as IDS, firewalls, load balancers and a variety of more applications. After that is the control layer, which is the brain of the SDN. Lastly is the infrastructure layer where we can find the actual physical or virtual network devices, such as switches.

2.1 Application Layer

As we mention above, the application layer is the layer that the controller is communicating with the applications via the northbound API. It is worth mentioning, that there is no standard for the northbound interface but usually there is a development with REST API.

These applications might include several types such as network management, network monitoring, network security, QoS policies and much more. The main responsibility of this layer is to define rules and services via the applications to the network and to adjust automatically when there is a change in the network. [11]

In figure we can see that clearly the application layer that includes the network application which will define the behavior of the network. Furthermore, it is shown that the control layer is connected through the northbound interface with the network applications. [32]

2.2 Control Layer

The main and most important layer of the software-defined network is the control layer and it is consider the brain of the SDN. This layer is very vital because it includes the controller. As it has been mentioned, the controller is the responsible for pushing all the policies coming from the applications to the network devices. [22] In a few words, the controller defines the routing and the traffic, as it is connect with the network applications and the network devices.

In figure 8, we can see the importance and the way that control layer is working. In addition, we can understand that the controller is connect through northbound API to the application layer and it is connecter via the southbound API to all the network devices. Furthermore, the figure illustrates

that the controller is receiving the commands via the application layer in order to perform them in the network devices. [26]

2.3 Infrastructure Layer

An infrastructure layer in the software define network is shaped from a variety of network devices which their purpose is to forward the network traffic of a network. When we mention network devices, we mean physical or virtual routers and switches in a data room or in the cloud. In order for these devices to be able to communicate with the controller in the control layer, a software must be install so for the southbound API interface to communicate with the controller.

In figure 8 we can see that the network devices that are place in the infrastructure layer need a southbound API interface like OpenFlow to be able to communicate with the control layer.

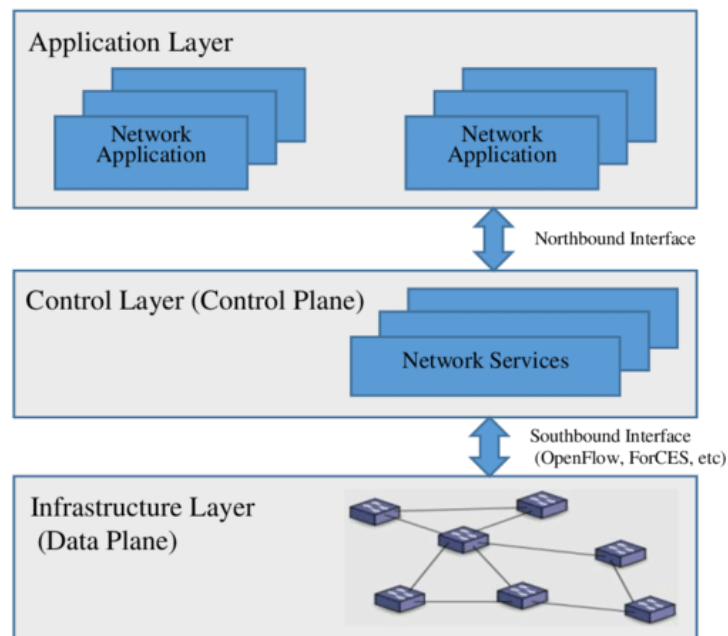


Figure 8 – The three layers in SDN architecture

4. Traditional Networks and SDN

In this chapter, we will describe the two main types of networking, the traditional network and the SDN, and a comparison between them.

3.1 What is Traditional Network

When we mention traditional network we refer to the conventional way that the networks are built. This “old” way of designing networks, its main base is the hardware such as switches and routers. These network devices will function individually as they have certain functions each in order to work well in the network. In order for all these devices to work well and to control network traffic, they are already provision with software that is dedicated for a specific purpose. [1] This has a result the functionality of the device to be dependent from a specific purpose application because a different hardware may has its own application. In addition, all the tasks are being applied from devices like routers or switches that are provisioned with an application.

3.2 What is a Software Defined Network

Software Defined Network is the concept of isolating the data plane from the routing process. By achieving this, networks can connect directly to the applications via API and this is boosting the performance of the applications, the security and the flexibility of a network. In addition, SDN will break down the architecture of the network to the control plane and the data plane because it will use virtualization instead of physical machines. [3] [7] Furthermore, network devices can be physical or virtual can use applications with open protocols and not closed firmware.

3.3 Differences between SDN and Traditional Network

Starting with, the main difference is that SDN is a software based technology as traditional is a hardware based one. This makes the SDN programmable where traditional is not and it is hard to try to replace existing applications that have been installed by the vendors. Moreover, in traditional networks there is a use of physical machines like routers, switches or any other physical networking equipment. SDN is using a virtual approach regarding network infrastructure. Using virtual components, the maintenance and the cost of a network can be reduce, as maintaining a traditional network with a physical infrastructure can be expensive. In addition, using virtual infrastructure is easier to extend the network and there is less complexity in the structure than in the traditional networks.

Furthermore, SDN can be control centralized in contrary with the traditional that has a distributed control. This is helping getting a control and a better understanding of the configuration and the architecture of a network. Similarly being centrally controlled, the SDN is easier to be troubleshoot instead of the traditional that is not. [1]

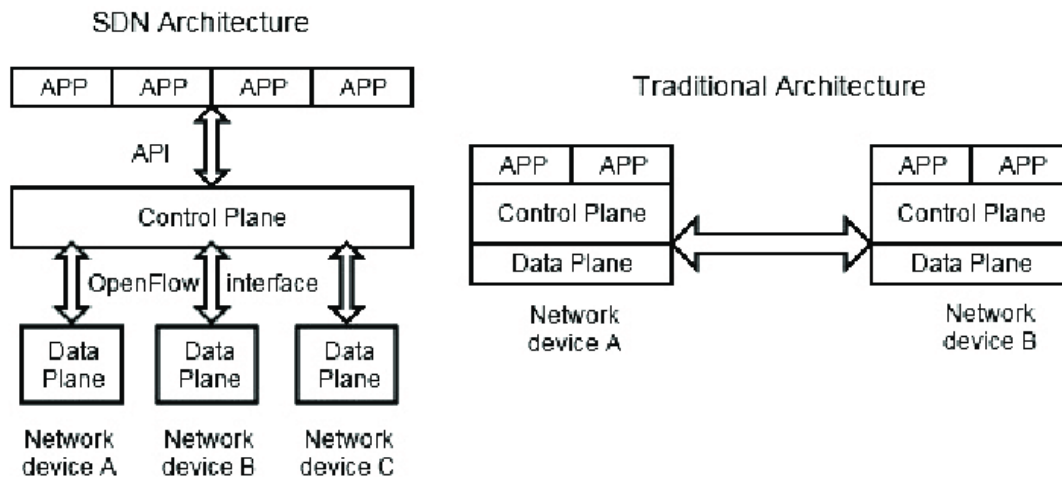


Figure 9 – Differences between SDN and traditional Network

In figure 9, we can clearly see the differences between the two architectures. SDN is centrally controlled by the controller as in traditional every network device has an existing application and the control is distributed.

5. Related Works

This chapter focuses in describing similar work that relate to my research regarding providing network resilience through an intelligent SDN network.

4.1 Operation Checkpoint: SDN Application Control

This research is regarding the development of the applications, which are connected with the network through the SDN controller. Because a number of security issues identified regarding applications, the authors are trying to protect and secure the northbound interface. The main approach of the authors is to try to introduce a permission system so that the controller can run policies only from trusted applications. In addition, they are supporting that with SDN there are many opportunities for innovations regarding new applications to increase the performance and the security of the network. However, we should always consider the security and trusted applications first before we use it to the network. [31]

Therefore, the problem is that northbound interface should allow only trusted applications. Currently this is happening with the action reading network state, which involves HTTP GET request, and writing network policies, which send HTTP POST to the controller. These approaches have vulnerabilities because first there is no confirmation of the relevance of the request for an application. In addition, applications do not have to provide information about their identity so there is no way to regulate their priority. Furthermore, legitimately installed applications may turn malicious without detection. [32]

The solution that they introduce was SE Floodlight. This is an extension of the OpenFlow controller, which has security enforcements as detection of possible rule conflicts between the control and data plane. In addition, there is a digitally authenticated northbound API. [33]

4.2 OpenFlow vulnerability assessment

In this topic, Kevin Benton presents the vulnerabilities of the OpenFlow protocol. Vendors of hardware and software identified the failure of the OpenFlow to adopt Transport Layer Security (TLS) by controller. This had a result the OpenFlow protocol to be vulnerable to man-in-the-middle attack. In addition, he highlights more vulnerabilities regarding control plane and OpenFlow protocol but the investigation is still ongoing regarding the suggestions for solving these vulnerabilities. [34]

5.1 FireCol: A Collaborative Protection Network for the Detection of Flooding DDoS Attacks

The main idea of this research that Jerome supports is to try to discover the attacks earlier as to protect the end users and our network infrastructure. The authors are focusing to the DDoS attack because this attack is a major security vulnerability. FireCol kernel is developed with an intrusion prevention system and they can place it to the level of an internet provider. With this way, the internet service provider will protect the hosts by traffic communication. Nevertheless, this solution looks heavyweight and due to its complexity will limit the flexibility concept of the SDN. [35]

4.4 Revisiting Security Aspects of Software-Defined Networking

In this topic, the authors are investigating threats and vulnerabilities of the software defined network. Their focus are the three most important parts of the security, the CIA. Confidentiality, Integrity and Availability within SDN infrastructure and architecture. It is just investigation without giving any specific solution for these aspects. [36]

4.5 OpenFlow: A Security Analysis

The authors done a security analysis of the OpenFlow protocol using several tools as STRIDE and other attacking methods. Their scenario was that the attacker had already access to the data plane and they wanted to stress out that OpenFlow security is a very important part of the software defined network. At the end, they proposed counter measures to mitigate security issues that related with the SDN. This measure can be adapted in newer versions of the OpenFlow. [37]

4.6 Resilience and Security in Software Defined Networking

This workshop points out issues on the resilience for software defined network. To begin with, data plane resilience is the protection of the data flows as control plane resilience is the connection of the devices to at least one controller. Authors claim that the threads of this resilience that can affect data flows is loss of connectivity, controller outage or even human error. [38]

Regarding the security of the SDN, they focused on the attacks for the three layers. Firstly, if the attacker performs a man in the middle attack and get access to the control plane, then he will control the whole network. They claim that with a secure channel this attack will be prevent. Furthermore, from the usage of malicious or faulty SDN applications, an attacker can gain access to the application plane. Lastly, the authors claim that attacks can also come from the data plane like DDoS, overload a controller, or injecting packets. At the end, this was an overview and a proposal of some theoretical solution that are already mentioned. [38]

4.7 Software defined networking for resilient communications in Smart Grid active distribution networks

Abdullah proposed in this article an SDN infrastructure to communicate with the Smart Grid networks because SDN can provide management, flexibility to large-scale network environments. The focus of this exercise was to implement redundant wireless communication links so in case of emergency the controller to be able to self-recover. The result showed that the performance was not as expected but the reliance was successful. Furthermore, the used the Mininet framework so to be able to capture metrics and simulate. [39]

4.8 Implementation challenges for software-defined networks

Sakir and the rest of the authors ran a very interesting research regarding cloud services and software defined networking. Their focus was to experiment with the security, scalability and in a major grade, the network performance. Because SDN has features like intelligent applications and dynamic nature, the authors are sure that can lower costs in terms of hardware and management. Also are trying to provide network solutions and services for operators and in parallel to increase security. They are still researching to find potential solutions in order to take advantage the services of the virtualization technology. [40]

4.9 Software-Defined Networking and Distributed Denial of Service

Since DDoS attacks are increasing in cloud computing, authors are trying to take a survey and a research in order to examine the challenges of the SDN resilience in cloud computing. They claim that the reliance of the SDN against DDoS attack in more advance in the cloud computing. Based on their experiments, SDN can detect and react quicker in a DDoS attack because it is centrally managed, dynamic rule forwarding and includes traffic analysis. In addition, they are researching for new mechanisms and trends for providing new methods in order to defend SDN in cloud computing. However, they are still researching of how to use SDN advantages so to defeat and prevent DDoS attacks in cloud environments. [41]

4.10 Controller placement strategies for a resilient SDN control plane

This study, claims that if the controller of the SDN needs to be outsource, then it requires a reliable switch to controller connection. They have represent two approaches for this issue to give priority to routing resilience by protecting the controller from node failures using failover scheme and they called it Reliable Controller Placement. The first method is to connect the switches to the controller over two Disjoint Control Paths (RCP-DCP). The second approach is to connect switches to two Different Controller Replicas (RCP-DCR) over two disjoint paths. During their research, both methods were working and they achieved a fast and efficient failover. Furthermore, these approaches improve a lot the resilience of the control plane which is considered the brain of the SDN. [42]

6. Testing and Implementation

Before testing and implementation, a test environment must be build. Then, under this test environment, tests will be develop.

6.1 Test Environment Preparation

To start with, hypervisor VirtualBox used for hosting the virtual machines needed. VirtualBox is a virtualization product and was downloaded from <https://www.virtualbox.org/>. The specifications are the below in figure 11:

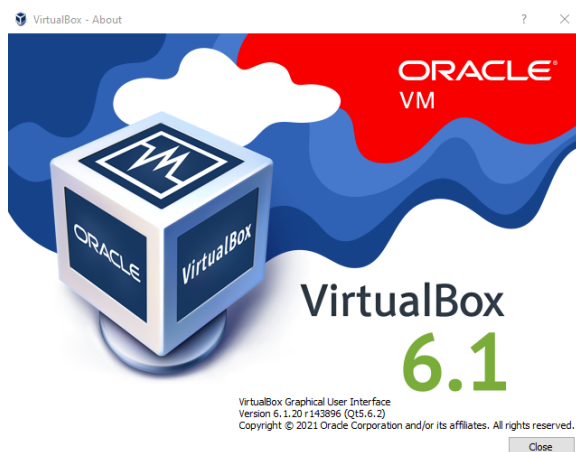


Figure 11- Version of Virtual Box

The main virtual machine that the tests are developed is based on Ubuntu. On this virtual machine, the ONOS and MININET are install. ONOS is a control plane for SDN and MININET creates virtual networks for tests to be developed.

6.1.1 Ubuntu Virtual Machine

Ubuntu is an open source operating system (OS) on Linux. For creating an Ubuntu OS virtual machine, an ISO file downloaded from <https://ubuntu.com>. Then a virtual machine created in VirtualBox to host this Ubuntu OS. After we install the OS from the ISO to the virtual machine named OnosNew, with the steps below:

Create Virtual Machine

? ×


← Create Virtual Machine

Name and operating system

Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:

Machine Folder:

Type: 

Version:

Figure 12 – Creation of Virtual Machine

Memory Size 4GB

? ×

← Create Virtual Machine

Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **1024 MB**.

4096 MB

4 MB 32768 MB

Figure 13 – Memory required for the Virtual machine

10 GB Virtual Hard Disk

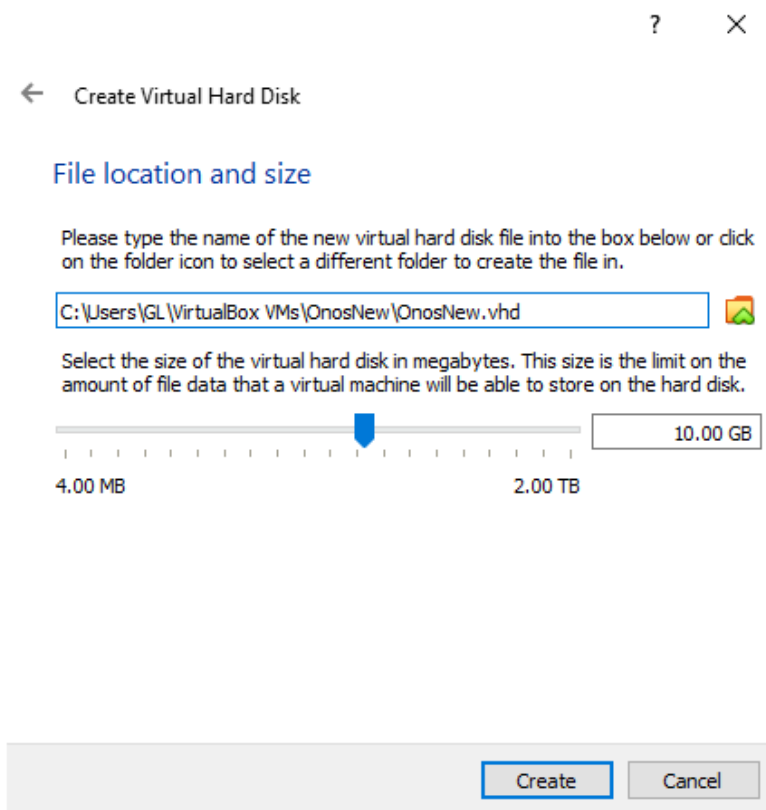


Figure 14 – Hard Disk required for the Virtual machine

Final VM Ubuntu 18.04

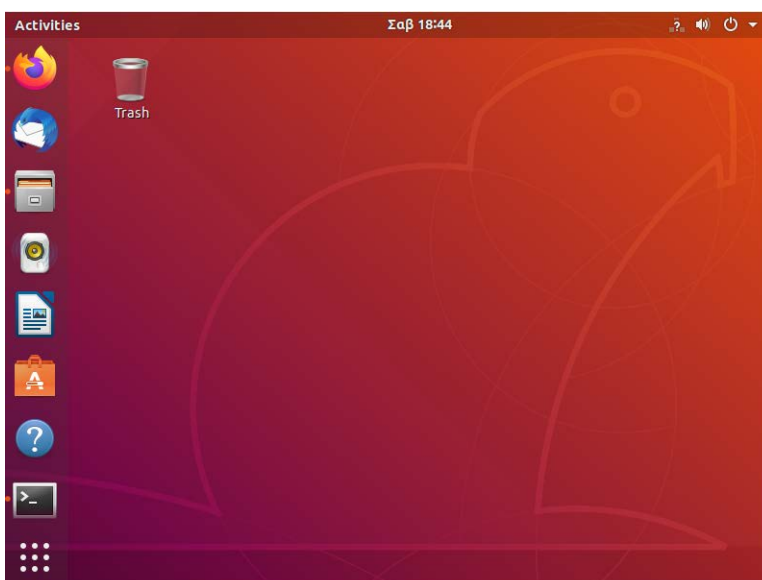


Figure 15 – How Virtual Machine looks after installation

6.1.2 ONOS Open Network Operating System

ONOS is also an open source and is a SDN controller. It can manages the network components and to run applications so to provide communication to end hosts. The below are prerequisites for ONOS installation.

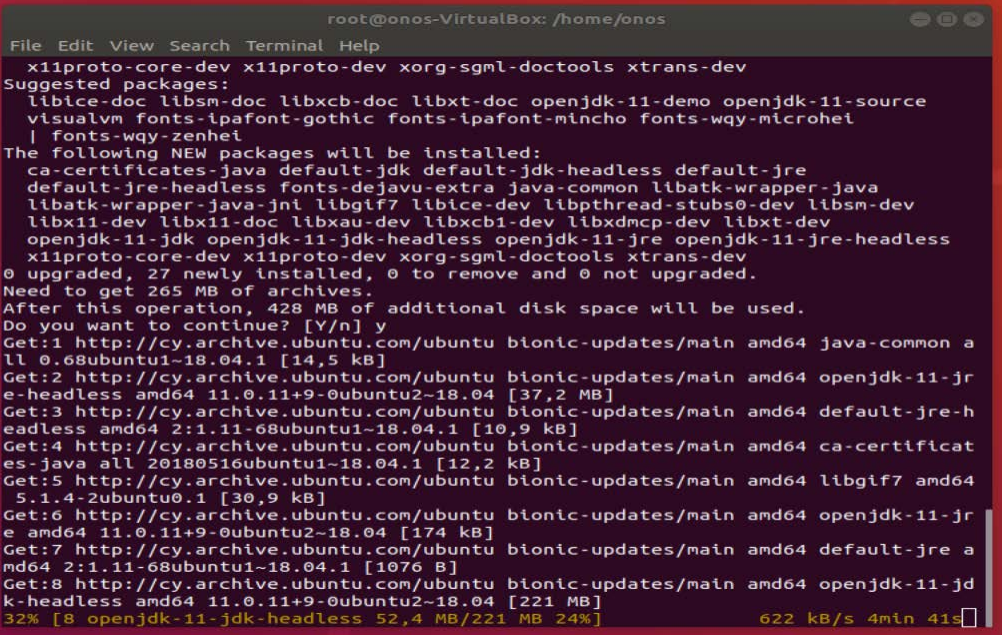
Hardware Requirements:

- 2 core CPU
- 2 GB RAM
- 10 GB hdd
- 1 NIC (any speed)

In order to install ONOS, java must be install on the virtual machine. The way to do it is the following:

6.1.3 Install JAVA

1. sudo su
2. apt-get update && apt-get upgrade
3. apt-get install software-properties-common
4. add-apt-repository ppa:linuxuprising/java
5. apt-get update
6. sudo apt install default-jdk :



```
root@onos-VirtualBox: /home/onos
File Edit View Search Terminal Help
x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
libice-doc libsm-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source
visualvm fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
| fonts-wqy-zenhei
The following NEW packages will be installed:
ca-certificates-java default-jdk default-jdk-headless default-jre
default-jre-headless fonts-dejavu-extra java-common libatk-wrapper-java
libatk-wrapper-java-jni libgif7 libice-dev libpthread-stubs0-dev libsm-dev
libx11-dev libx11-doc libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless
x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 27 newly installed, 0 to remove and 0 not upgraded.
Need to get 265 MB of archives.
After this operation, 428 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 java-common a
ll 0.68ubuntu1-18.04.1 [14,5 kB]
Get:2 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 openjdk-11-jr
e-headless amd64 11.0.11+9-0ubuntu2-18.04 [37,2 MB]
Get:3 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 default-jre-h
eadless amd64 2:1.11-68ubuntu1-18.04.1 [10,9 kB]
Get:4 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 ca-certificat
es-java all 20180516ubuntu1-18.04.1 [12,2 kB]
Get:5 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgif7 amd64
5.1.4-2ubuntu0.1 [30,9 kB]
Get:6 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 openjdk-11-jr
e amd64 11.0.11+9-0ubuntu2-18.04 [174 kB]
Get:7 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 default-jre a
md64 2:1.11-68ubuntu1-18.04.1 [1076 B]
Get:8 http://cy.archive.ubuntu.com/ubuntu bionic-updates/main amd64 openjdk-11-jd
k-headless amd64 11.0.11+9-0ubuntu2-18.04 [221 MB]
32% [8 openjdk-11-jdk-headless 52,4 MB/221 MB 24%] 622 kB/s 4min 41s
```

Figure 16 – Installation of Java in Ubuntu

7. java –version. The below is the java version installed:

```
root@onos-VirtualBox:/home/onos# java -version
openjdk version "11.0.11" 2021-04-20
OpenJDK Runtime Environment (build 11.0.11+9-Ubuntu-0ubuntu2.18.04)
OpenJDK 64-Bit Server VM (build 11.0.11+9-Ubuntu-0ubuntu2.18.04, mixed mode, sharing)
root@onos-VirtualBox:/home/onos#
```

Figure – 17 Version of Java installed

6.1.4 Maven

After java, maven was installed. Is a project management tools that is used for Java projects.

1. sudo apt update
2. sudo apt install maven
3. mvn -version:

```
File Edit View Search Terminal Help
root@onos-VirtualBox:/home/onos# mvn -version
Apache Maven 3.6.0
Maven home: /usr/share/maven
Java version: 11.0.11, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.4.0-72-generic", arch: "amd64", family: "unix"
root@onos-VirtualBox:/home/onos#
```

Figure 18 – Installation of Maven and version

6.1.5 Mininet

Is a network emulator, which creates a network of virtual hosts, switches, controllers, and links. Mininet hosts run standard Linux network software, and its switches support OpenFlow for highly flexible custom routing and Software-Defined Networking. [45]

The commands are very simple to install it and are the following:

1. sudo apt-get update
2. sudo apt-get upgrade
3. sudo apt-get dist-upgrade
4. sudo apt-get install git
5. git clone git://github.com/mininet/mininet
6. cd mininet
7. git tag
8. git checkout -b 2.2.2
9. sudo ~/mininet/util/install.sh -a

```

onos@onos-VirtualBox:~$ git clone git://github.com/mininet/mininet
Cloning into 'mininet'...
remote: Enumerating objects: 10165, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10165 (delta 2), reused 7 (delta 2), pack-reused 10154
Receiving objects: 100% (10165/10165), 3.19 MiB | 3.64 MiB/s, done.
Resolving deltas: 100% (6784/6784), done.
onos@onos-VirtualBox:~$

```

Figure 19 - Installation of Mininet

Below is the way to test the functionality of the mininet. Sudo mn

```

bash: cd: mininet: no such file or directory
root@onos-VirtualBox:/# sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>

```

Figure 20 - Test functionality of mininet

In order to test a simple network is to ping hosts between them like the below example.

h1 ping h2

```

mininet> ifconfig
*** Unknown command: ifconfig
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data:
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=3.46 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.794 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.110 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.040 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.107 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.095 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.321 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.097 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.037 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.109 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.108 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.108 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=0.084 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=0.039 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=0.039 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=0.041 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=0.092 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=0.055 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=0.041 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=0.040 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=0.036 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=0.155 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=0.109 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=0.044 ms

```

Figure 21 - Ping hosts over mininet

6.1.6 ONOS

For this testing, ONOS 2.3.0 was used.:

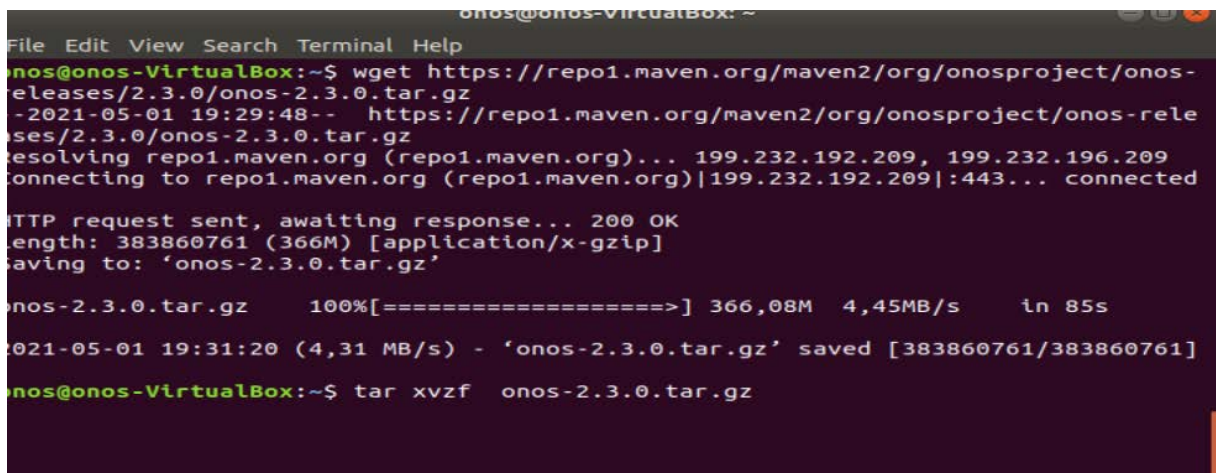


Release Name	Version	API	Release Date	Download	Release Notes	About
Uguisu	2.4.0	API-2.4.0	Jun 5, 2020	tar.gz		About the bird
Toucan	2.3.0	API-2.3.0	Jan 27, 2020	tar.gz	Release Notes	About the bird
Sparrow (LTS)	2.2.0	API-2.2.0	Aug 30, 2019	tar.gz	Release Notes	About the bird
	2.2.1	API-2.2.1	Feb 20, 2020	tar.gz		
	2.2.2	API-2.2.2	Mar 25, 2020	tar.gz		
	2.2.3	API-2.2.3	Jul 3, 2020	tar.gz		

Figure 22 – ONOS Version

The commands that were used to install was the following:

1. `wget https://repo1.maven.org/maven2/org/onosproject/onos-releases/2.3.0/onos-2.3.0.tar.gz`
2. `tar xvzf onos-2.3.0.tar.gz`



```
onos@onos-VirtualBox: ~$ wget https://repo1.maven.org/maven2/org/onosproject/onos-releases/2.3.0/onos-2.3.0.tar.gz
--2021-05-01 19:29:48-- https://repo1.maven.org/maven2/org/onosproject/onos-releases/2.3.0/onos-2.3.0.tar.gz
resolving repo1.maven.org (repo1.maven.org)... 199.232.192.209, 199.232.196.209
connecting to repo1.maven.org (repo1.maven.org)|199.232.192.209|:443... connected
HTTP request sent, awaiting response... 200 OK
Content-Length: 383860761 (366M) [application/x-gzip]
Saving to: 'onos-2.3.0.tar.gz'

onos-2.3.0.tar.gz  100%[=====] 366,08M  4,45MB/s  in 85s
2021-05-01 19:31:20 (4,31 MB/s) - 'onos-2.3.0.tar.gz' saved [383860761/383860761]

onos@onos-VirtualBox: ~$ tar xvzf onos-2.3.0.tar.gz
```

Figure 23 – Installation of ONOS

Then ONOS must be configure. First of all the IP of the VM needed for setting ONOS up. With the command `ifconfig` the IP is found:

```

root@onos-VirtualBox: /
File Edit View Search Terminal Help
h1 h2
*** Done
completed in 120.781 seconds
root@onos-VirtualBox:/# clear

root@onos-VirtualBox:/# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::7846:ec99:1a7:cabe prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:f0:7f:4f txqueuelen 1000 (Ethernet)
    RX packets 152558 bytes 210085419 (210.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 58206 bytes 3565301 (3.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Figure 24 – IP for SDN Controller

Then with the below commands, ONOS controller is ready:

1. cd onos 2.3.0
2. bin/onos-config <IP address>
3. bin/onos-service [start/stop/restart/status]

```

onos@onos-VirtualBox:~$ cd onos-2.3.0/
onos@onos-VirtualBox:~/onos-2.3.0$ ls
apache-karaf-4.2.6  apps  bin  init  VERSION
onos@onos-VirtualBox:~/onos-2.3.0$ cd bin/
onos@onos-VirtualBox:~/onos-2.3.0/bin$ ls
_check-json  onos.bk          onos-log-query      onos-service.bk
_find-node   onos-cfg         onos-netcfg         onos-user-key
onos         onos-compile-yang onos-node-diagnostics onos-user-password
onos-app     onos-diagnostics onos-restore        _rest-port
onos-backup  onos-jenable     onos-service
onos@onos-VirtualBox:~/onos-2.3.0/bin$

```

Figure 25 – Commands for starting ONOS service

After set up is complete, enter ONOS CLI with this command

/bin/onos as we can see in the figure 26.

```

onos@onos-VirtualBox: ~/onos-2.3.0
File Edit View Search Terminal Help
onos@onos-VirtualBox:~$ cd onos-2.3.0/
onos@onos-VirtualBox:~/onos-2.3.0$ bin/onos
Warning: Permanently added '[localhost]:8101' (RSA) to the list of known hosts.
Password authentication
Password:
Welcome to Open Network Operating System (ONOS)!

  ONOS

Documentation: wiki.onosproject.org
Tutorials:     tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.

onos@root >
19:45:47

```

Figure 26 – Service of ONOS started

```
onos@root > nodes
id=10.0.2.15, address=10.0.2.15:9876, state=READY, version=2.3.0, updated=2m27s
ago *
onos@root >
```

Figure 27 – Node of SDN controller

Also after finishing setup, you can enter the GUI of ONOS through the http address: <http://172.17.0.5:8181/onos/ui/login.html> it is in figure 28:

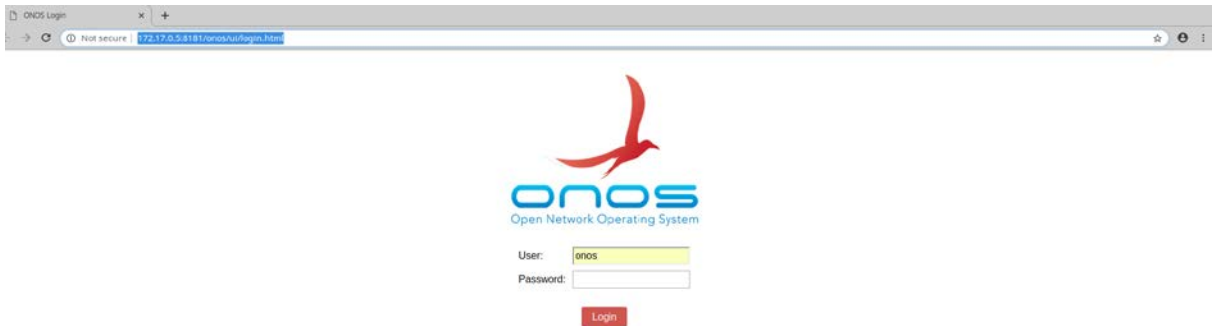


Figure 28 – Web Interface of ONOS controller

6.2 SDN Testing and Evaluation

The purpose of the penetration testing is to evaluate if SDN and traditional networking have the same or similar vulnerabilities with the same penetration tools.

In SDN testing, spine leaf topology is used from the ONOS tutorial through mininet. The Spine leaf topology that is used consists of one cluster of three controllers with IP address 172.17.0.5-7, twenty hosts and six switches. In the figure 29 is how this topology is in a high-level design.

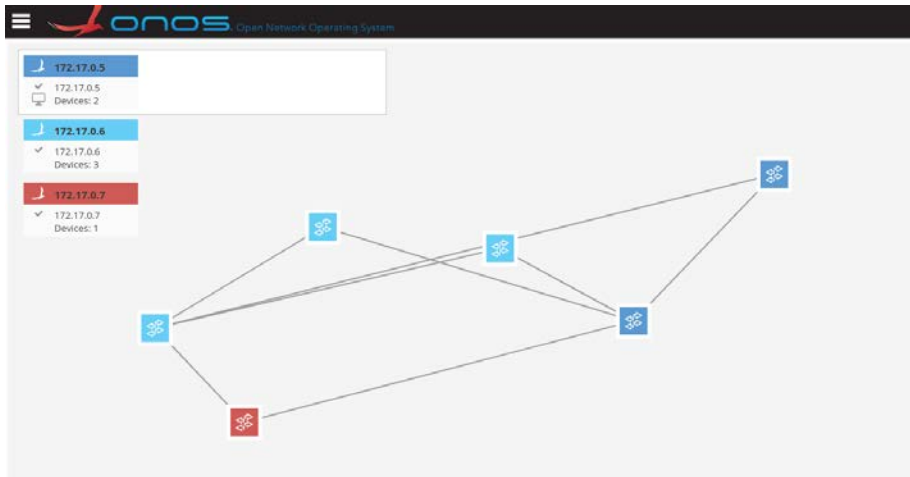


Figure 29 - Spine Leaf Topology

Also in the figure 30, all the hosts and then in figure 31 cluster nodes can be view:

The screenshot shows the ONOS interface with a table of 20 hosts. The table has columns for Friendly Name, Host ID, MAC Address, VLAN ID, Configured status, IP Addresses, and Location. All hosts are listed with IP addresses from 10.0.0.1 to 10.0.0.20.

FRIENDLY NAME	HOST ID	MAC ADDRESS	VLAN ID	CONFIGURED	IP ADDRESSES	LOCATION
10.0.0.1	00:00:00:00:01/None	00:00:00:00:01	None	false	10.0.0.1	of:00000000000000b3
10.0.0.2	00:00:00:00:02/None	00:00:00:00:02	None	false	10.0.0.2	of:00000000000000b4
10.0.0.3	00:00:00:00:03/None	00:00:00:00:03	None	false	10.0.0.3	of:00000000000000b5
10.0.0.4	00:00:00:00:04/None	00:00:00:00:04	None	false	10.0.0.4	of:00000000000000b6
10.0.0.5	00:00:00:00:05/None	00:00:00:00:05	None	false	10.0.0.5	of:00000000000000b7
10.0.0.6	00:00:00:00:06/None	00:00:00:00:06	None	false	10.0.0.6	of:00000000000000c3
10.0.0.7	00:00:00:00:07/None	00:00:00:00:07	None	false	10.0.0.7	of:00000000000000c4
10.0.0.8	00:00:00:00:08/None	00:00:00:00:08	None	false	10.0.0.8	of:00000000000000c5
10.0.0.9	00:00:00:00:09/None	00:00:00:00:09	None	false	10.0.0.9	of:00000000000000c6
10.0.0.10	00:00:00:00:0A/None	00:00:00:00:0A	None	false	10.0.0.10	of:00000000000000c7
10.0.0.11	00:00:00:00:0B/None	00:00:00:00:0B	None	false	10.0.0.11	of:00000000000000d3
10.0.0.12	00:00:00:00:0C/None	00:00:00:00:0C	None	false	10.0.0.12	of:00000000000000d4
10.0.0.13	00:00:00:00:0D/None	00:00:00:00:0D	None	false	10.0.0.13	of:00000000000000d5
10.0.0.14	00:00:00:00:0E/None	00:00:00:00:0E	None	false	10.0.0.14	of:00000000000000d6
10.0.0.15	00:00:00:00:0F/None	00:00:00:00:0F	None	false	10.0.0.15	of:00000000000000d7
10.0.0.16	00:00:00:00:10/None	00:00:00:00:10	None	false	10.0.0.16	of:00000000000000e3
10.0.0.17	00:00:00:00:11/None	00:00:00:00:11	None	false	10.0.0.17	of:00000000000000e4
10.0.0.18	00:00:00:00:12/None	00:00:00:00:12	None	false	10.0.0.18	of:00000000000000e5
10.0.0.19	00:00:00:00:13/None	00:00:00:00:13	None	false	10.0.0.19	of:00000000000000e6
10.0.0.20	00:00:00:00:14/None	00:00:00:00:14	None	false	10.0.0.20	of:00000000000000e7

Figure 30 - Hosts created in the topology

The screenshot shows the ONOS interface with a table of 3 cluster nodes. The table has columns for Active status, Started status, Node ID, IP Address, TCP Port, and Last Updated.

ACTIVE	STARTED	NODE ID	IP ADDRESS	TCP PORT	LAST UPDATED
✓	✓	172.17.0.5	172.17.0.5	9876	8:00:33 PM -09:00
✓	✓	172.17.0.6	172.17.0.6	9876	8:00:34 PM -09:00
✓	✓	172.17.0.7	172.17.0.7	9876	8:00:35 PM -09:00

Figure 31 - Cluster of controllers

6.2.1 Crawl and Audit

As a first test, Burp was used because the IP of the controller pages are known. Burp is an application security tool and it can be very efficient and fast for different tests. As a first step, Burp Suite was installed on the Linux VM according to the following steps:

We downloaded the file from <https://portswigger.net/burp/pro> and created an account in order to get a license for the software. Then using the below two commands, software installed on the VM:

```
chmod +x <package.sh>
```

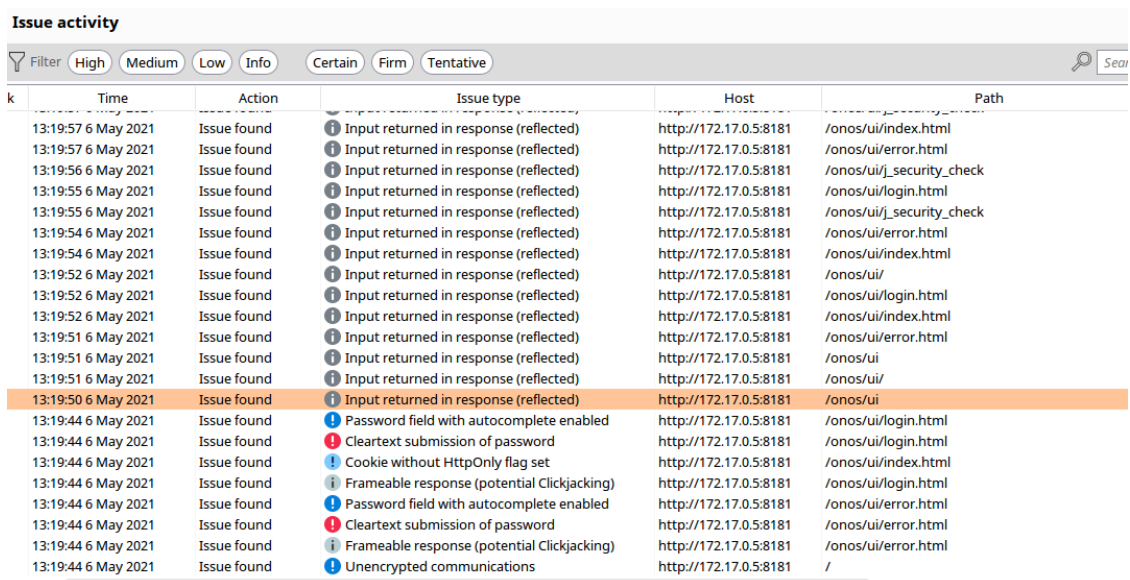
```
chmod +x burpsuite_community_linux_v2020_1.sh
```

As a first test, I scanned the controller for crawl and audit using the burp suite. Firstly, as in figure 32 the scope must be define.



Figure 32 – Scope for the attack

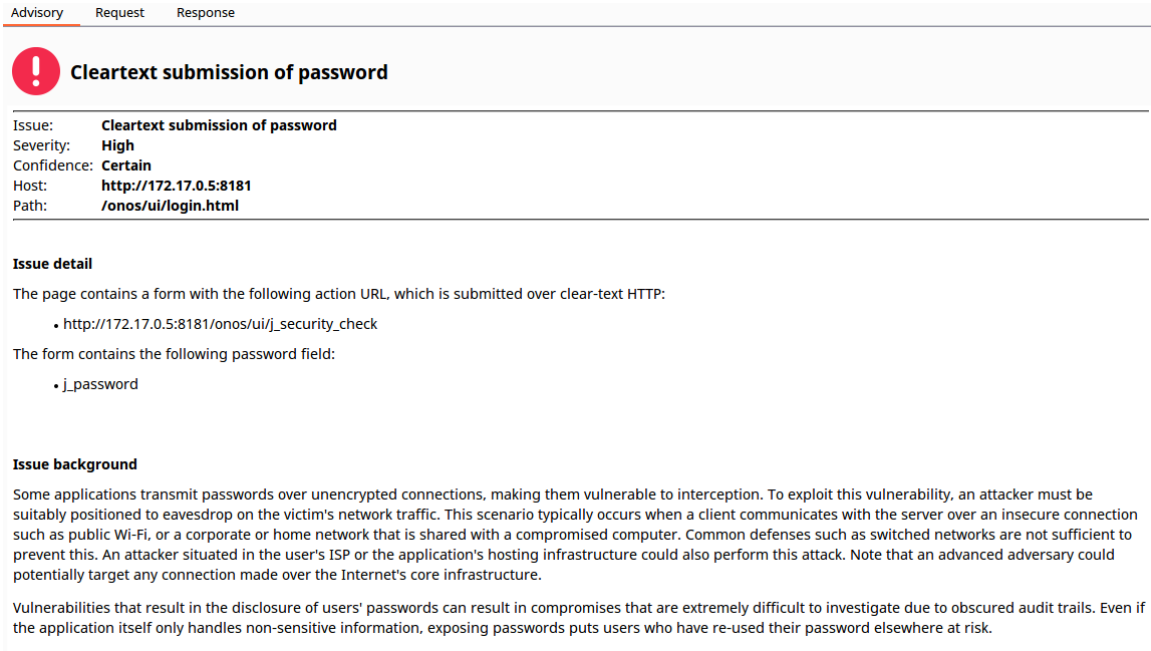
Then a new scan started for the controller <http://172.17.0.5:8181/onos/ui/> and many findings were exposed. In the figure 33 is the state of the results after the scan.



k	Time	Action	Issue type	Host	Path
	13:19:57 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/index.html
	13:19:57 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/error.html
	13:19:56 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/_security_check
	13:19:55 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/login.html
	13:19:55 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/_security_check
	13:19:54 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/error.html
	13:19:54 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/index.html
	13:19:52 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/
	13:19:52 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/login.html
	13:19:52 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/index.html
	13:19:51 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/error.html
	13:19:51 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui
	13:19:51 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui/
	13:19:50 6 May 2021	Issue found	Input returned in response (reflected)	http://172.17.0.5:8181	/onos/ui
	13:19:44 6 May 2021	Issue found	Password field with autocomplete enabled	http://172.17.0.5:8181	/onos/ui/login.html
	13:19:44 6 May 2021	Issue found	Cleartext submission of password	http://172.17.0.5:8181	/onos/ui/login.html
	13:19:44 6 May 2021	Issue found	Cookie without HttpOnly flag set	http://172.17.0.5:8181	/onos/ui/index.html
	13:19:44 6 May 2021	Issue found	Frameable response (potential Clickjacking)	http://172.17.0.5:8181	/onos/ui/login.html
	13:19:44 6 May 2021	Issue found	Password field with autocomplete enabled	http://172.17.0.5:8181	/onos/ui/error.html
	13:19:44 6 May 2021	Issue found	Cleartext submission of password	http://172.17.0.5:8181	/onos/ui/error.html
	13:19:44 6 May 2021	Issue found	Frameable response (potential Clickjacking)	http://172.17.0.5:8181	/onos/ui/error.html
	13:19:44 6 May 2021	Issue found	Unencrypted communications	http://172.17.0.5:8181	/

Figure 33 – Results of crawl report

According to the results, were identified two high severity, four low severity and 18 informational. The issue of the high severity is the clear text submission of the password as this make the controller vulnerable to interception. Of course in order for the attacker to exploit this, must eavesdrop the network traffic. This is usually happens if the client will communicate with the server through a public internet connection or through a compromised device. In the figure 34 is the high severity issue:



The screenshot shows a security advisory page with the following details:

- Issue:** Cleartext submission of password
- Severity:** High
- Confidence:** Certain
- Host:** http://172.17.0.5:8181
- Path:** /onos/ui/login.html

Issue detail

The page contains a form with the following action URL, which is submitted over clear-text HTTP:

- http://172.17.0.5:8181/onos/ui/j_security_check

The form contains the following password field:

- j_password

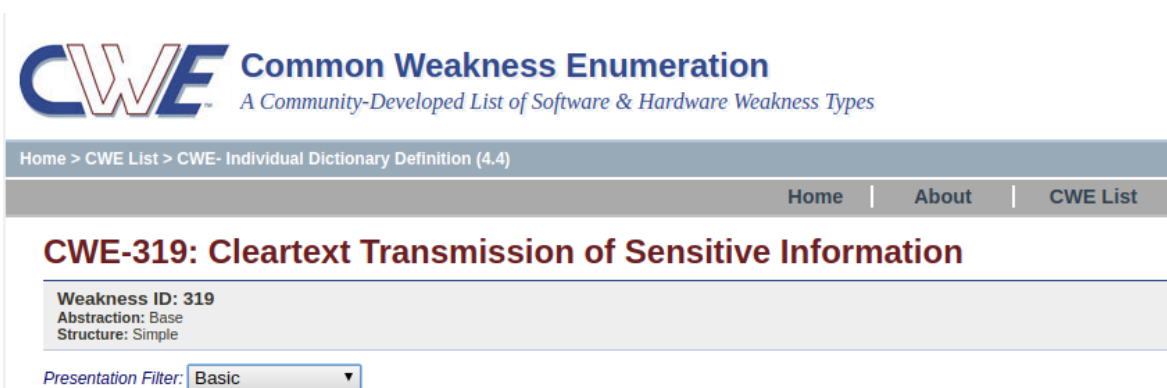
Issue background

Some applications transmit passwords over unencrypted connections, making them vulnerable to interception. To exploit this vulnerability, an attacker must be suitably positioned to eavesdrop on the victim's network traffic. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this. An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. Note that an advanced adversary could potentially target any connection made over the Internet's core infrastructure.

Vulnerabilities that result in the disclosure of users' passwords can result in compromises that are extremely difficult to investigate due to obscured audit trails. Even if the application itself only handles non-sensitive information, exposing passwords puts users who have re-used their password elsewhere at risk.

Figure 34 – Cleartext submission of password

Furthermore, there is a vulnerability classification for this issue and it is called CWE-319: Creartext Transmission of Sensitive information. It is one of the top 25 most dangerous software weaknesses and it shown in the figure 35:



The screenshot shows the CWE-319: Cleartext Transmission of Sensitive Information page. The page header includes the CWE logo and the text 'Common Weakness Enumeration - A Community-Developed List of Software & Hardware Weakness Types'. The breadcrumb trail is 'Home > CWE List > CWE- Individual Dictionary Definition (4.4)'. The page title is 'CWE-319: Cleartext Transmission of Sensitive Information'. The weakness details are:

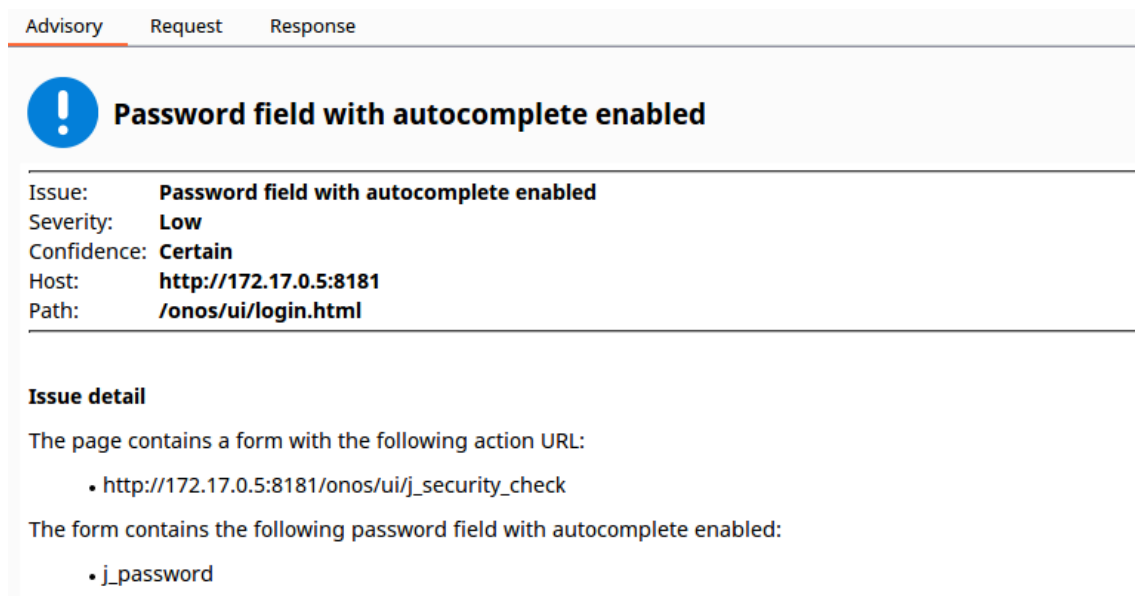
- Weakness ID:** 319
- Abstraction:** Base
- Structure:** Simple

The presentation filter is set to 'Basic'.

Figure 35 – CWE-319

Moreover, the way to solve this issue is applications to use SSL or TLS. The transport-level encryption protects the communication between the server and the client and especially if there are login related functionalities, it is a mandatory to use them.

Another issue that was found, was that the password field with autocomplete enabled. Because many internet browsers have a functionality to remember credentials, this cause a vulnerability for the web applications or applications such SDN controllers. For example, if a user store its credentials an attacker can gain control of the application using a cross-site scripting. Presenting in the figure 36 the advisory warning.



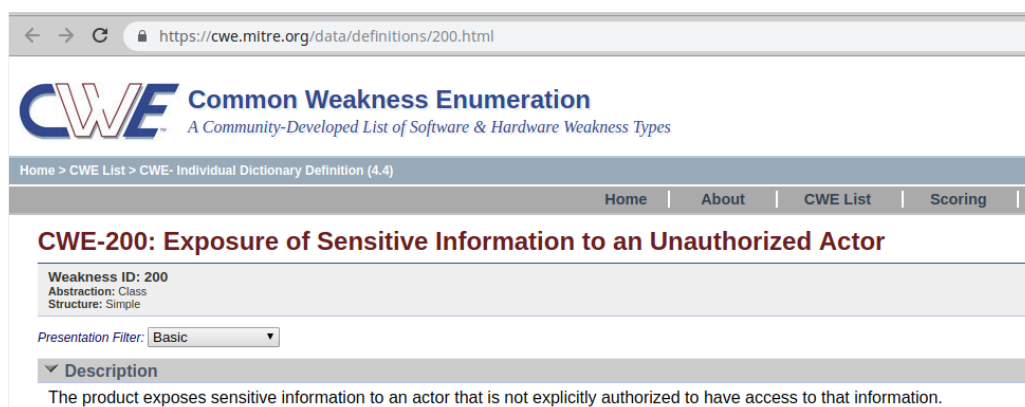
The screenshot shows a security advisory page with a navigation bar at the top containing 'Advisory', 'Request', and 'Response'. The main heading is 'Password field with autocomplete enabled' next to a blue exclamation mark icon. Below the heading, the following details are listed:

- Issue: **Password field with autocomplete enabled**
- Severity: **Low**
- Confidence: **Certain**
- Host: **http://172.17.0.5:8181**
- Path: **/onos/ui/login.html**

Under the heading 'Issue detail', the text states: 'The page contains a form with the following action URL:' followed by a bullet point: '• http://172.17.0.5:8181/onos/ui/j_security_check'. Below that, it says: 'The form contains the following password field with autocomplete enabled:' followed by a bullet point: '• j_password'.

Figure 39 – Password field with autocomplete enabled

The vulnerability classification in this issue it is present in the figure 37 and is the CWE-200: Information Exposure.



The screenshot shows the MITRE website page for CWE-200. The browser address bar shows 'https://cwe.mitre.org/data/definitions/200.html'. The page title is 'CWE-200: Exposure of Sensitive Information to an Unauthorized Actor'. Below the title, the following information is displayed:

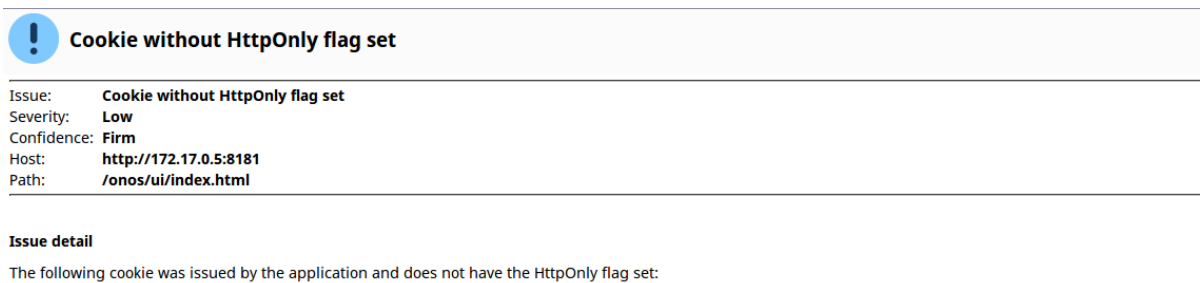
- Weakness ID: 200
- Abstraction: Class
- Structure: Simple

There is a 'Presentation Filter' dropdown menu set to 'Basic'. Below that, a 'Description' section is expanded, showing the text: 'The product exposes sensitive information to an actor that is not explicitly authorized to have access to that information.'

Figure 37 – CWE-200

The way to remediate this issue is to prevent the browsers to store credentials by including an attribute **autocomplete=off** within the form tag. Nowadays most of the browsers may ignore this directive but there must be an option in the menu to disable autocomplete.

Next, several issues with cookies were flagged. The advisory warning is called cookie without HttpOnly flag set and these cookies contains session tokens. The risk in this issue is that the attacker can capture this cookie through an injected script and get the tokens. In figure 40 is the low severity advisory:



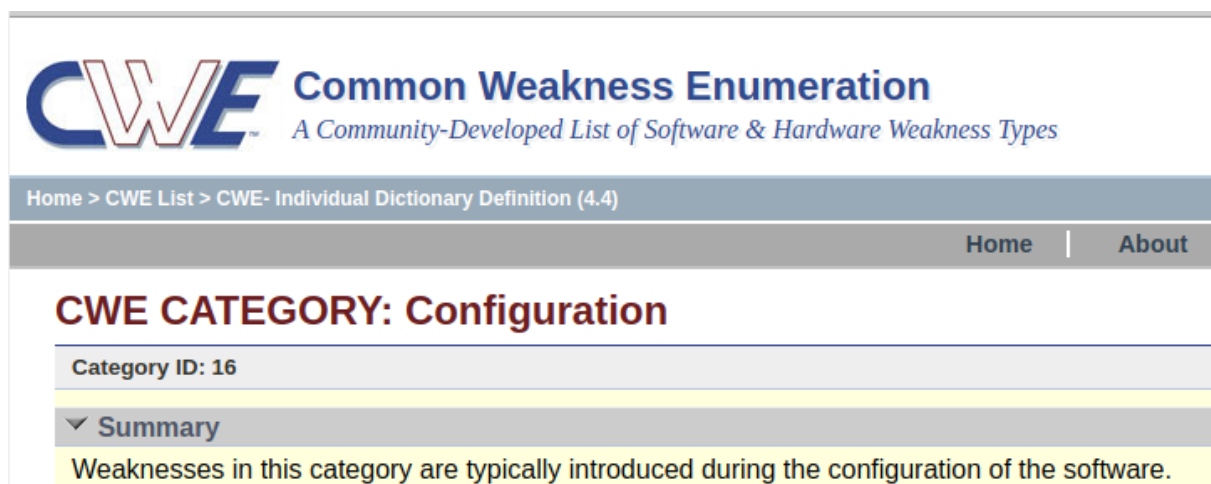
The screenshot shows a security advisory with the following details:

- Issue:** Cookie without HttpOnly flag set
- Severity:** Low
- Confidence:** Firm
- Host:** http://172.17.0.5:8181
- Path:** /onos/ui/index.html

Issue detail
The following cookie was issued by the application and does not have the HttpOnly flag set:

Figure 40 - Cookie without HttpOnly flag set

In addition, there is a vulnerability classification for this issue, which is the CWE-16: Configuration and it is displayed in the below figure 41.



The screenshot shows the CWE-16: Configuration page with the following content:

- CWE Common Weakness Enumeration**
A Community-Developed List of Software & Hardware Weakness Types
- Home > CWE List > CWE- Individual Dictionary Definition (4.4)
- Home | About
- CWE CATEGORY: Configuration**
- Category ID: 16
- Summary: Weaknesses in this category are typically introduced during the configuration of the software.

Figure - 41 CWE-16: Configuration

Because attacks can happen from a simple cookie stealing, http flag must be set. This can be configured through the applications, but in our case, this cannot be configured to set due to JavaScript. SDN controller will not be able to read the cookies if this feature is set and it will not work as expected.

6.2.2 Dictionary Attack using Burp Suite

After crawl, a dictionary attack performed, again through burp suite. Five payloads filled for the attack for sending the information to the tool to attack. For example common passwords or usernames. Burp also offers hundreds of lists with this information but in our case, the payloads filled manually.

In figure, 42 are the payloads with usernames and in figure, 43 are the payloads filled with passwords.

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 7
Payload type: Simple list Request count: 6

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

Paste	admin
Load ...	administrator
Remove	user
Clear	user1
	onos
	User
	rocks

Add

Add from list ...

Figure 42 - Payloads with usernames

Target Positions **Payloads** Options

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type and each payload type can be customized in different ways.

Payload set: 2 Payload count: 7
Payload type: Simple list Request count: 6

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

Paste	password
Load ...	123456
Remove	qwerty
Clear	Password
	Password1
	onos
	rocks

Add

Add from list ...

Figure 43 - Payloads with passwords

After filling the payloads, the attack started and combined all the information that was input. Many attempts failed at the beginning but finally the login credentials exploit. ONOS credentials were the default, so this was a very easy task for an attacker to exploit the login credentials. Usually a dictionary attack uses predefined list of passwords and usernames and if users are using easy or common passwords, then the dictionary attacks will be successful. In figure below are the results of the dictionary attack and all the combinations that burp used. Furthermore in figure is the raw data representing the username and the password of the controller.

Payload4	Payload5
user1	rocks
user1	rocks
user1	rocks
user1	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks
onos	rocks

Figure 44 - Username and password match

```
3 Content-Length: 16
4 Connection: close
5
6 p3=onos&p4=rocks
```

Figure 45 - Raw data proving the correct username and password

It is very easy to protect from a dictionary attack, firstly with changing the default usernames and passwords. Also, common passwords or known usernames should not be in use. In addition, multi factor authentication must be in place for all the application that are using the web and login credentials.

Furthermore, strong passwords must be used and in case of failed logins, the account must lockout. In addition, all passwords must be change regularly for a better security of a possible dictionary attack.

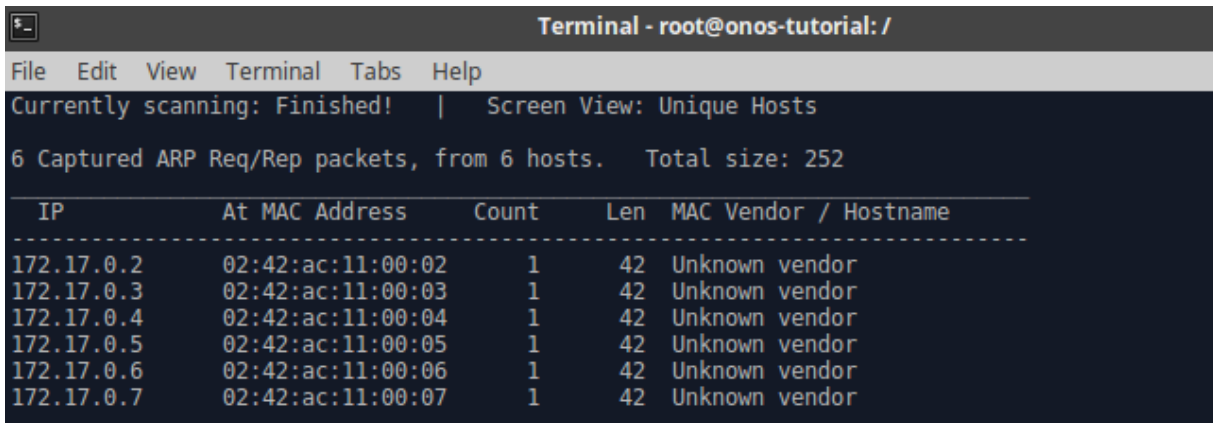
6.2.3 Network Discovery

The main purpose of this test is to discover the network information of the SDN controller. If an attacker is connect to the same network as the controller, he can obtain some of the network information. For this test, netdiscover tool was used with the command as in figure 45. netdiscover -r 172.17.0.5 and is the IP address of the SDN controller.

```
Active scan completed, 0 hosts found.
root@onos-tutorial:/# netdiscover -r 172.17.0.5 -P -N
```

Figure 46 - Netdiscover command

From the results below in figure 47, we can understand that the attacker can get information about the cluster of the controllers but cannot get information about the hosts that the controller manages.



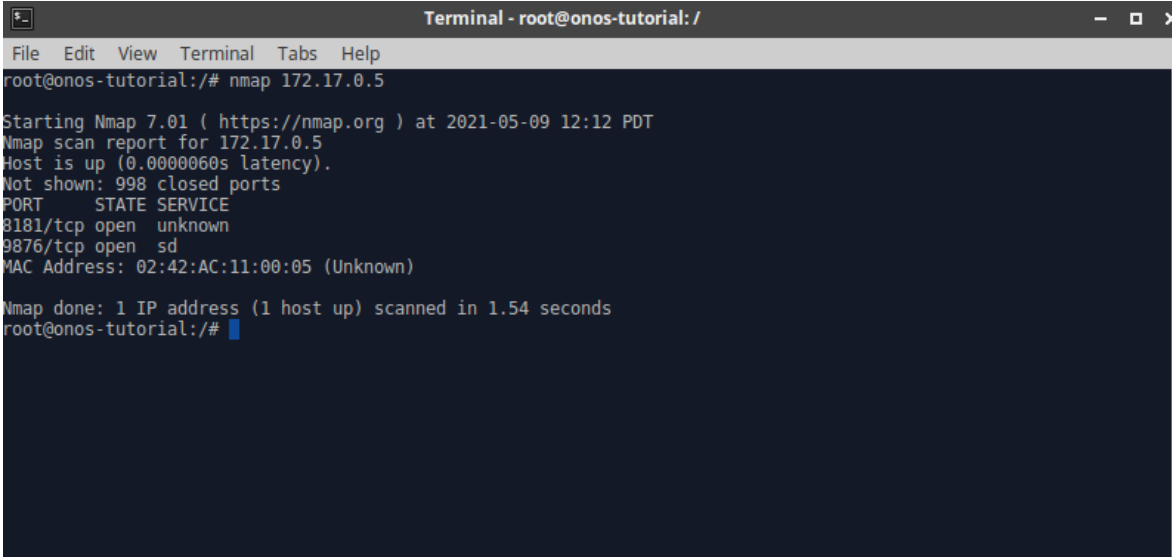
```
Terminal - root@onos-tutorial: /
File Edit View Terminal Tabs Help
Currently scanning: Finished! | Screen View: Unique Hosts
6 Captured ARP Req/Rep packets, from 6 hosts. Total size: 252
-----
IP           At MAC Address      Count  Len  MAC Vendor / Hostname
-----
172.17.0.2   02:42:ac:11:00:02   1      42  Unknown vendor
172.17.0.3   02:42:ac:11:00:03   1      42  Unknown vendor
172.17.0.4   02:42:ac:11:00:04   1      42  Unknown vendor
172.17.0.5   02:42:ac:11:00:05   1      42  Unknown vendor
172.17.0.6   02:42:ac:11:00:06   1      42  Unknown vendor
172.17.0.7   02:42:ac:11:00:07   1      42  Unknown vendor
```

Figure 47 - Network Information about cluster

Examining at these results, the attacker cannot get full information about the content of each machine. He can get only information about the web applications like the MAC address and ip of the cluster nodes of the SDN controller. In case the attacker knows the IP range of the host, easily can run the command **netdiscover -r 10.0.0.0**

6.2.4 NMAP

Another way to get information about the SDN controller is using Nmap command through a device. This tool is to scan open ports and it is suitable for security audits.



```
Terminal - root@onos-tutorial: /
File Edit View Terminal Tabs Help
root@onos-tutorial:/# nmap 172.17.0.5

Starting Nmap 7.01 ( https://nmap.org ) at 2021-05-09 12:12 PDT
Nmap scan report for 172.17.0.5
Host is up (0.0000060s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
8181/tcp  open  unknown
9876/tcp  open  sd
MAC Address: 02:42:AC:11:00:05 (Unknown)

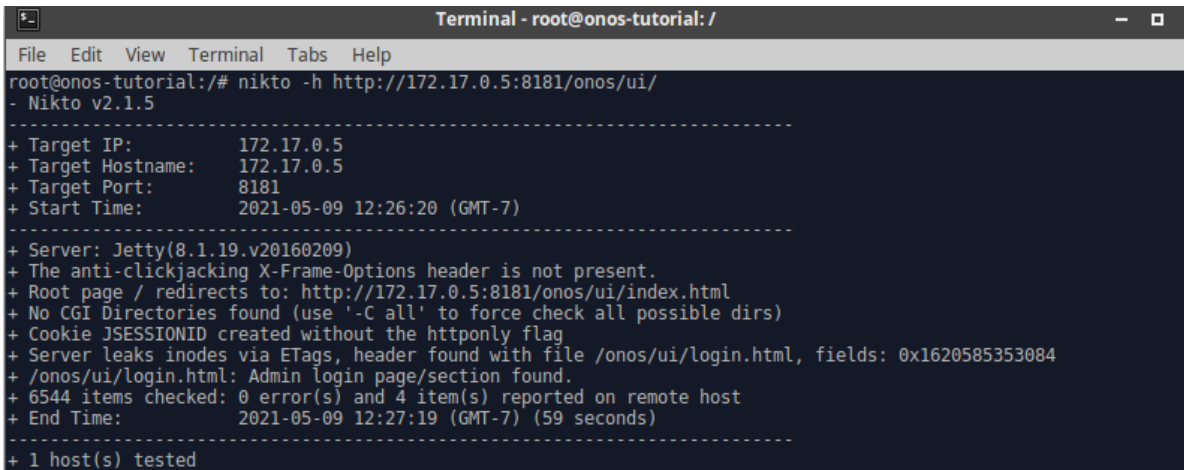
Nmap done: 1 IP address (1 host up) scanned in 1.54 seconds
root@onos-tutorial:/#
```

Figure 48 – Results ofNMAP

In figure 48, we can see that all the ports of the SDN controller are close except default, 8181 and 9876. This is a very good technique for ONOS, because it keeps all the other ports closed and this can avoid further attacks.

6.2.5 Nikto

Nikto is a tool that scans webservers, in our case the SDN controller, for dangerous files or any other outdated software. Using Nikto with the command `nikto -h http://172.17.0.5:8181/onos/ui/` we can view the results in figure 49:



```
Terminal - root@onos-tutorial: /
File Edit View Terminal Tabs Help
root@onos-tutorial:/# nikto -h http://172.17.0.5:8181/onos/ui/
- Nikto v2.1.5
-----
+ Target IP:          172.17.0.5
+ Target Hostname:   172.17.0.5
+ Target Port:       8181
+ Start Time:        2021-05-09 12:26:20 (GMT-7)
-----
+ Server: Jetty(8.1.19.v20160209)
+ The anti-clickjacking X-Frame-Options header is not present.
+ Root page / redirects to: http://172.17.0.5:8181/onos/ui/index.html
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Cookie JSESSIONID created without the httponly flag
+ Server leaks inodes via ETags, header found with file /onos/ui/login.html, fields: 0x1620585353084
+ /onos/ui/login.html: Admin login page/section found.
+ 6544 items checked: 0 error(s) and 4 item(s) reported on remote host
+ End Time:          2021-05-09 12:27:19 (GMT-7) (59 seconds)
-----
+ 1 host(s) tested
```

Figure 48 – Scan results ofNikto

If we examine the above results, we can clearly see that anti-clickjacking x-frame-options header is not present. This means that the SDN controller could be at risk of clickjacking attack. Usually, the attacker will use layers to trick a user to click on a link on a framed page [46]. The way to avoid this is to send the proper X-Frame-Options in HTTP response headers that instruct the browser not to allow framing from other domains [46].

The classification for anti-clickjacking x-frame-options header is not present is low and are the following: CAPEC-103, CWE-693, ISO27001-A.14.2.5, OWASP 2013-A5, OWASP 2017-A6 [46]

Another vulnerability is the cookie jsessionid created without the httponly flag. In this case SDN controller use the cookies to store sensitive information without marking the cookie with the HttpOnly flag. This has a result an attacker to read this content of the information that the cookie has. In order to avoid this there is a configuration and this set can be implement. Its classification is medium and is the CWE-1004: Sensitive Cookie Without 'HttpOnly' Flag. [47]

Furthermore, the admin login page/section login found is a vulnerability also. If the login page is found then an attacker can perform an SQL Injection.

6.2.6 DDOS Attack

DDOS attack performed with the tool hping3. Hping3 is a tool that is used for network testing, firewall testing and much more. Firstly, hping3 was installed using just a single command **apt install hping3**. Then, command, as in figure 50, **hping3 -S -flood -V -p 8181 172.17.0.5** for flooding the SDN controller with packet in order to make it unavailable.

```
root@onos-tutorial:/# hping3 -S --flood -V -p 8181 172.17.0.5
using docker0, addr: 172.17.0.1, MTU: 1500
HPING 172.17.0.5 (docker0 172.17.0.5): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 172.17.0.5 hping statistic ---
```

Figure 50 - Performing DDOS attack

Examining the results, we noticed that this type of attack did not affect much the SDN controller. The results were very decent and the availability of the controller was still up. Although the webpage of the controller had a very slow respond, it was still working.

As we can see from figure 51 this is the machine state that is hosting the SDN controller, before the attack. In figure 52, we can notice that the attack did not affect the resources much. This might be due to packet-based load Balancing Technique that ONOS has already implemented.

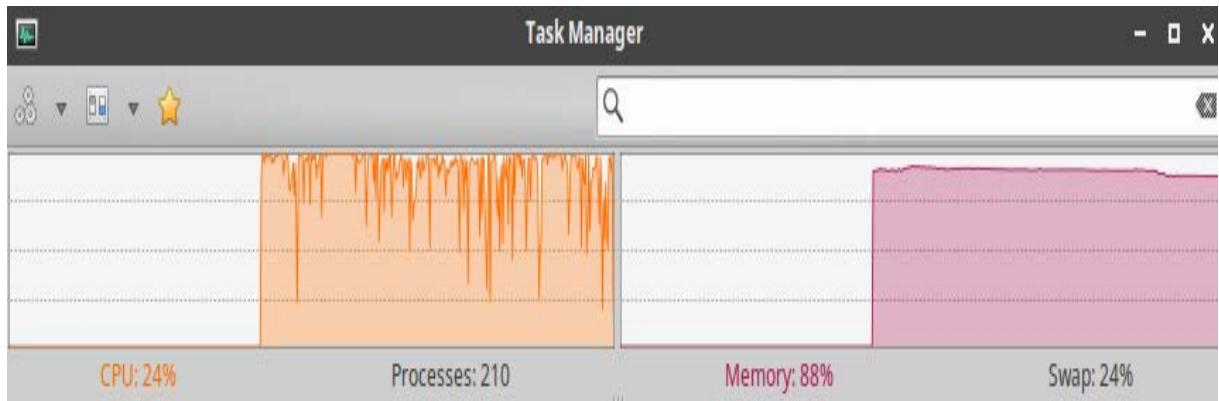


Figure 51 – State of the machine before DDOS attack

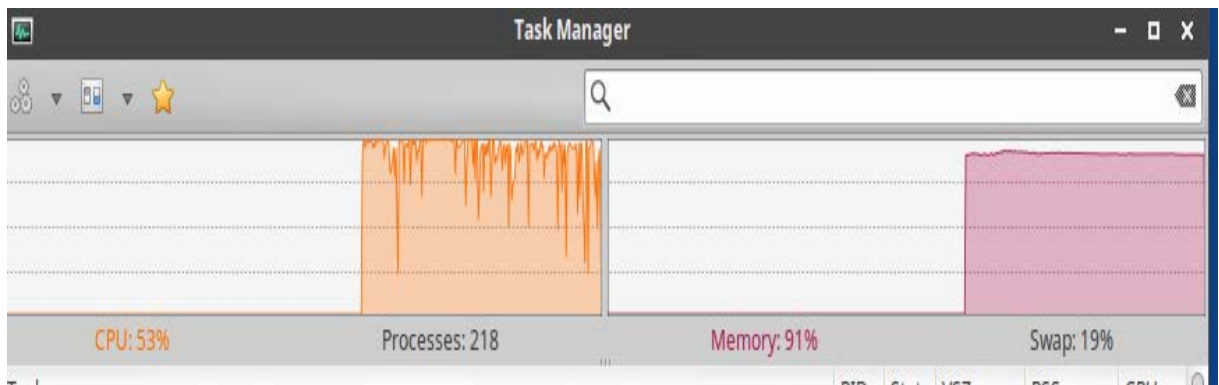


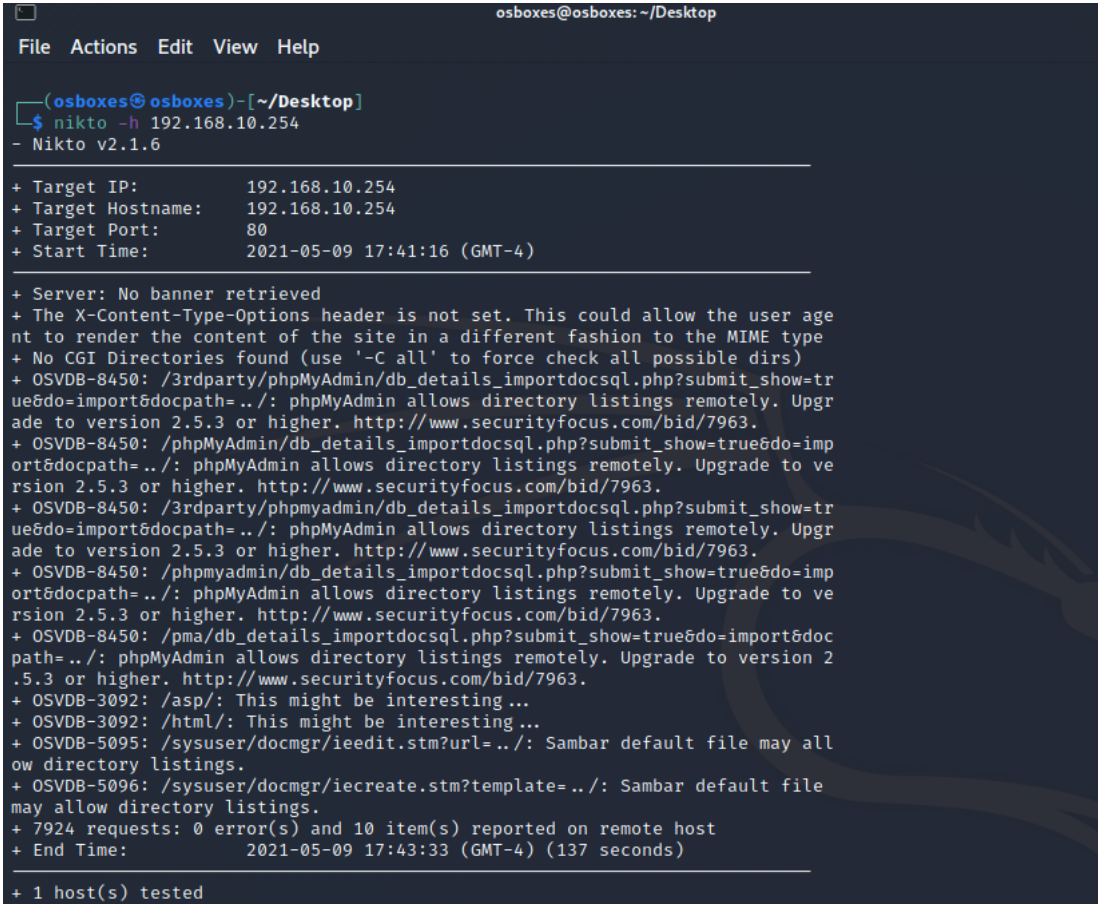
Figure 52 – State of the machine after DDOS attack

6.3 Traditional Networks

For this testing a Mikrotik router was used that was connected to the Internet through the Internet Service provider. The IP address of the Mikrotik router was 192.168.10.254 and were connected four hosts. Furthermore, Kali Linux was used to perform the following tests.

6.3.1 Nikto

Nikto was used as a first test. Command `nikto -h 192.168.10.254` run and in figure 53, we can see the results.



```
osboxes@osboxes: ~/Desktop
File Actions Edit View Help
(osboxes@osboxes)-[~/Desktop]
$ nikto -h 192.168.10.254
- Nikto v2.1.6

+ Target IP: 192.168.10.254
+ Target Hostname: 192.168.10.254
+ Target Port: 80
+ Start Time: 2021-05-09 17:41:16 (GMT-4)

+ Server: No banner retrieved
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-8450: /3rdparty/phpMyAdmin/db_details_importdocsql.php?submit_show=true&do=import&docpath=../: phpMyAdmin allows directory listings remotely. Upgrade to version 2.5.3 or higher. http://www.securityfocus.com/bid/7963.
+ OSVDB-8450: /phpMyAdmin/db_details_importdocsql.php?submit_show=true&do=import&docpath=../: phpMyAdmin allows directory listings remotely. Upgrade to version 2.5.3 or higher. http://www.securityfocus.com/bid/7963.
+ OSVDB-8450: /3rdparty/phpmyadmin/db_details_importdocsql.php?submit_show=true&do=import&docpath=../: phpMyAdmin allows directory listings remotely. Upgrade to version 2.5.3 or higher. http://www.securityfocus.com/bid/7963.
+ OSVDB-8450: /phpmyadmin/db_details_importdocsql.php?submit_show=true&do=import&docpath=../: phpMyAdmin allows directory listings remotely. Upgrade to version 2.5.3 or higher. http://www.securityfocus.com/bid/7963.
+ OSVDB-8450: /pma/db_details_importdocsql.php?submit_show=true&do=import&docpath=../: phpMyAdmin allows directory listings remotely. Upgrade to version 2.5.3 or higher. http://www.securityfocus.com/bid/7963.
+ OSVDB-3092: /asp/: This might be interesting...
+ OSVDB-3092: /html/: This might be interesting...
+ OSVDB-5095: /sysuser/docmgr/ieedit.stm?url=../: Sambar default file may allow directory listings.
+ OSVDB-5096: /sysuser/docmgr/iecreate.stm?template=../: Sambar default file may allow directory listings.
+ 7924 requests: 0 error(s) and 10 item(s) reported on remote host
+ End Time: 2021-05-09 17:43:33 (GMT-4) (137 seconds)

+ 1 host(s) tested
```

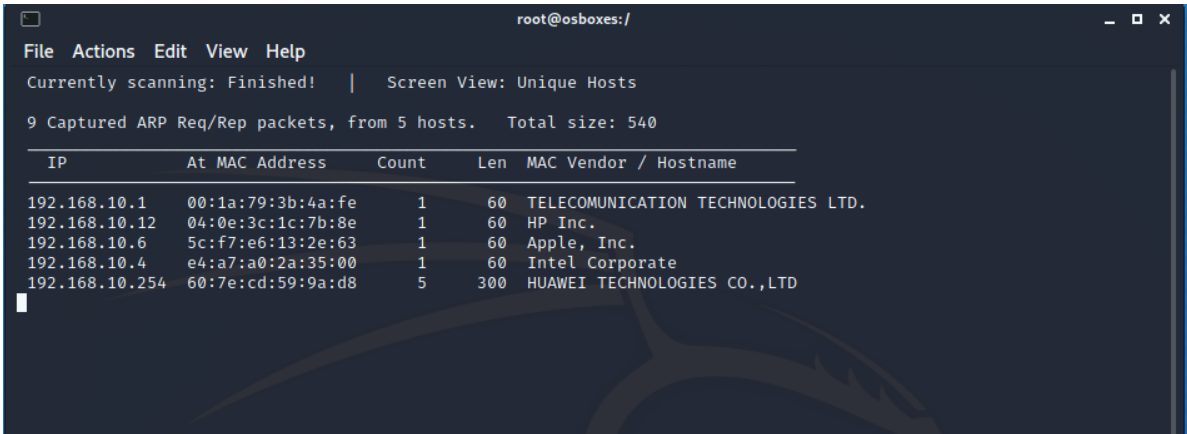
Figure 53 - Results of Nikto in traditional network

In contrary with the test in the SDN, here we can see many vulnerabilities. For example X-Content-Type header. In addition, we can see a lot OSVDB vulnerability that SDN controller never had. Furthermore, there also expired software that needs update like OSVDB-8450.

SDN controller is safer in this section as it does not use a variety of software to work as expected. The software that is using is only for the APIs that are used to connect data plane with the devices.

6.3.2 Network Discovery

The same test performed as in the SDN controller. Netdiscover was used and using the command `netdiscover 192.168.10.254` having the below results in figure 54.



```
root@osboxes:/
File Actions Edit View Help
Currently scanning: Finished! | Screen View: Unique Hosts
9 Captured ARP Req/Rep packets, from 5 hosts. Total size: 540
-----
IP           At MAC Address  Count  Len  MAC Vendor / Hostname
-----
192.168.10.1 00:1a:79:3b:4a:fe 1      60  TELECOMUNICATION TECHNOLOGIES LTD.
192.168.10.12 04:0e:3c:1c:7b:8e 1      60  HP Inc.
192.168.10.6 5c:f7:e6:13:2e:63 1      60  Apple, Inc.
192.168.10.4 e4:a7:a0:2a:35:00 1      60  Intel Corporate
192.168.10.254 60:7e:cd:59:9a:d8 5      300 HUAWEI TECHNOLOGIES CO.,LTD
```

Figure 54 - Results of Netdiscover in traditional network

The difference with the SDN controller is that in traditional networks if you run this commands it will find all the devices that are connected. In the case of the SDN controller, the netdiscover could not find the hosts but only the cluster of the controllers. From this, we can understand that there is a better information security in SDN because the ARP table looks to be more secure. Netdiscover is using the IP addresses from the ARP to find the hosts and as we can see, traditional network is more vulnerable in this test.

6.3.3 NMAP

Nmap is used once again, performing the command `nmap 192.168.10.254`. As we see the results in figure 55, we can understand that traditional networks need more ports open that SDN does. SDN controller will use only the port to configure the IP address of the device that is using. Usually 8181. Traditional networks, because are using more ports open will have a greater chance for an attack than the SDN.


```
root@osboxes:/home/osboxes/Desktop
File Actions Edit View Help
(osboxes@osboxes)-[~/Desktop]
$ sudo su
[sudo] password for osboxes:
(root@osboxes)-[~/home/osboxes/Desktop]
# nmap 192.168.10.254
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-09 18:23 EDT
Nmap scan report for 192.168.10.254
Host is up (0.00055s latency).
Not shown: 995 closed ports
PORT      STATE SERVICE
21/tcp    filtered ftp
22/tcp    filtered ssh
23/tcp    filtered telnet
53/tcp    open  domain
80/tcp    open  http
MAC Address: 60:7E:CD:59:9A:D8 (Huawei Technologies)

Nmap done: 1 IP address (1 host up) scanned in 1.41 seconds
```

Figure 55 - Results of NMAP in traditional network

6.3.4 DDOS Attack

In this assessment, hping3 is used as the test in SDN. Using the command from Kali Linux, **hping3 192.168.10.254**, figure 56 we tried to perform a DDOS attack to the Mikrotik router. Equally, with SDN, traditional network performed very well in this test without losing its availability and responding very well to the attack.

```
(osboxes@osboxes)-[~]
$ sudo su
[sudo] password for osboxes:
(root@osboxes)-[~/home/osboxes]
# hping3 192.168.10.254
HPING 192.168.10.254 (eth0 192.168.10.254): NO FLAGS are set, 40 headers + 0 data bytes
```

Figure 56 - Performing DDOS attack

Examining the results, we can see in figure 57 the state of the machine before the attack and in figure 58 the state after the attack. Before the attack, the packets were at a normal state with a couple of spikes from time to time. After the attack started, network spikes expanded to a very high level together with the TCP connections but there was a very good balance of the flood that prevent router to flood.

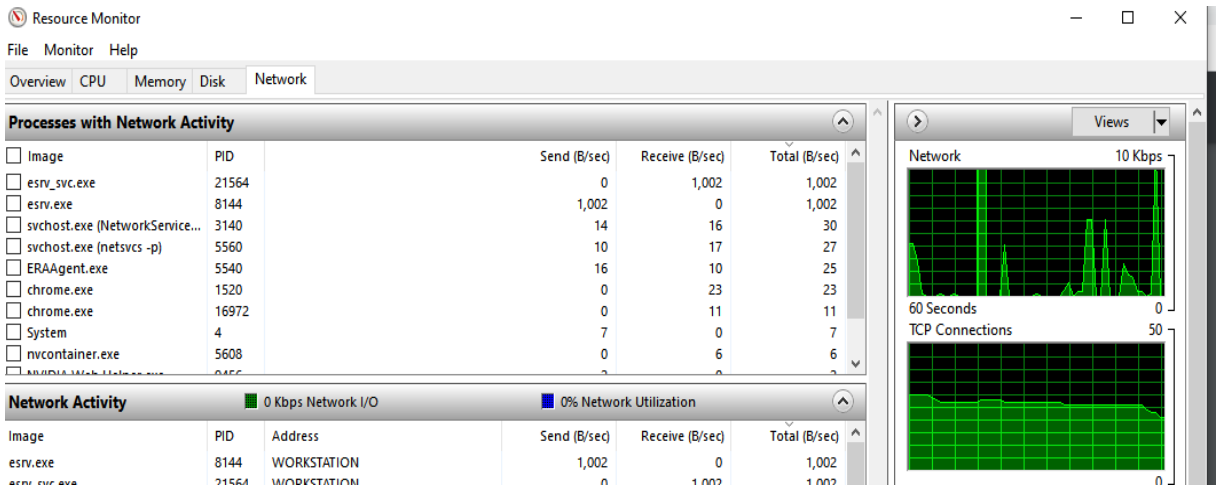


Figure 57 - State of the machine before DDOS attack traditional network

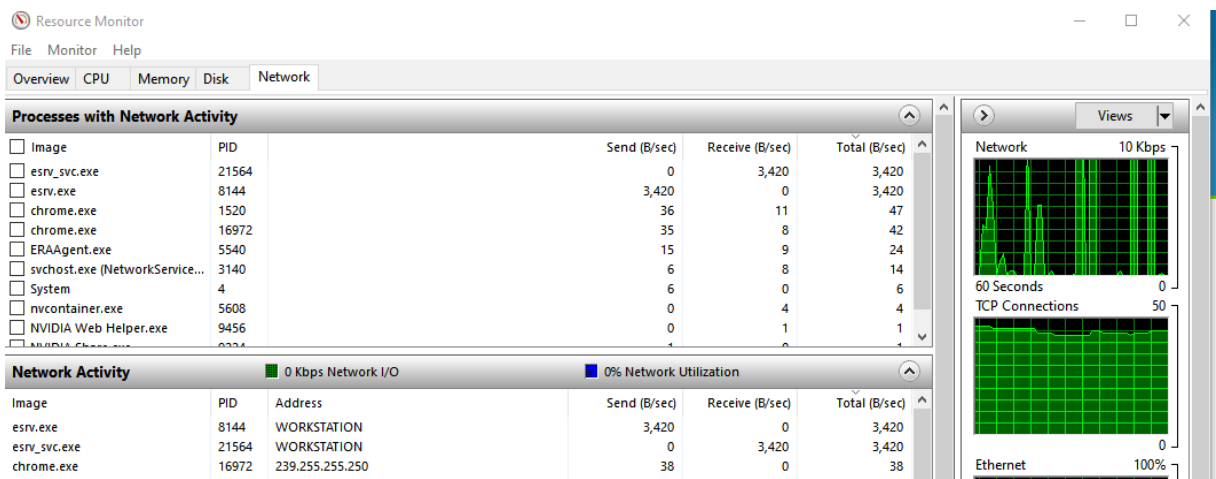


Figure 58 - State of the machine after DDOS attack traditional network

6.3.5 Dictionary Attack

Because Kali Linux is used, this test is performed by Hydra. Hydra is a tool that is used for brute force username and passwords. Firstly, hydra-gtk was used (figure 59) but there were not any results of any username or password. A list of passwords and usernames was uploaded to the tool but still it could not break the password.

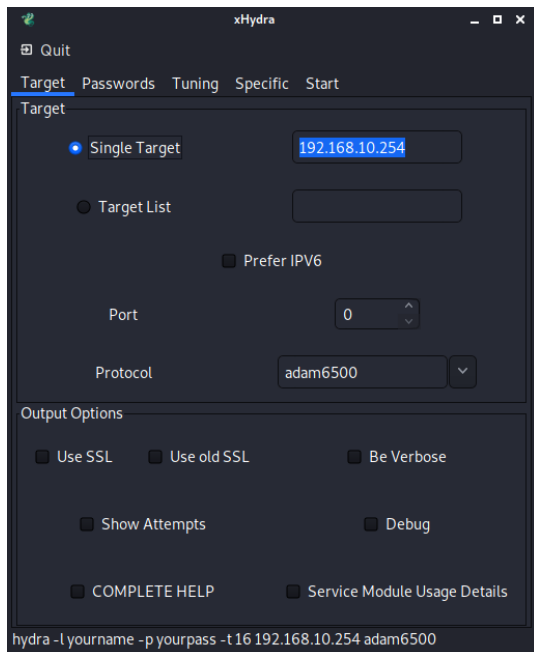


Figure 59 - Hydra-gtk

Then Hydra is used in command line (figure 60) using a list of passwords and usernames. In many cases, we used the real username with case sensitive passwords so we can examine the behavior. Again, Hydra did not manage to attack the password.

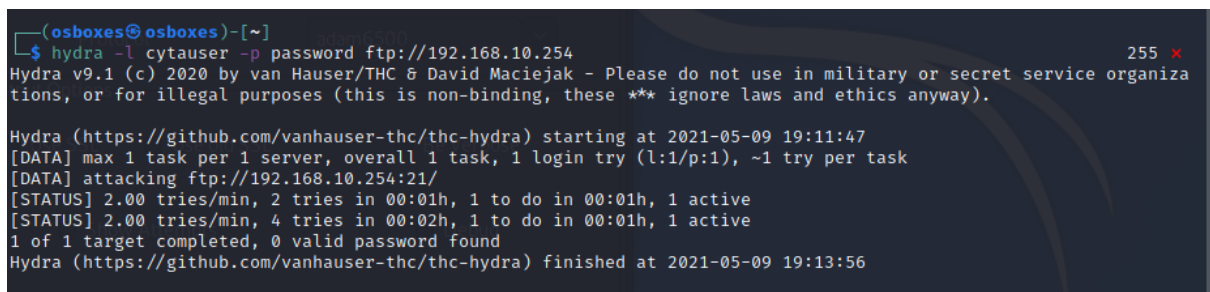


Figure 60 - Hydra command line

The difference that we can see concerning SDN and traditional network, is that in SDN network the combination was found with the username and the password. We strongly believe that the reason that the SDN was less secure than traditional in this test was because all the credentials of ONOS were the default and never changed.

7. Discussion

Security is a major concept in our days that must take seriously. The main aspects of the security is confidentiality, integrity and availability. Resiliency of any network type SDN or traditional, systems high availability and every other technology are depending on these aspects. There are countless factors that a system or a network to lose its resilience and availability, like misconfiguration, human error, physical disaster and much more. The key reason that systems lose their resiliency and availability is the lack of security and their vulnerabilities to cyber-attacks.

Nowadays, plenty of tools and techniques exist that a person can use for hacking systems and networks. Nevertheless, there are quite a lot of ways to protect from them. For this project, the tools and techniques that used were mostly Linux based, as Linux is famous for its penetration techniques. In both tests, SDN and traditional, the tools that have been used if the same and in some cases very similar. Starting with the crawling test that the results were very decent. The two high priority CVE's that found in SDN can be fix very easily with the use of TLS and SSL. Traditional network router did not even allowed crawling tools to gather information.

Furthermore, network discovery performed for both types of networks. Although the Linux tool netdiscover found almost everything for both types of networks, it is not something to worry. This traffic can stopped by using firewalls and configuration files so the packets that the tool is sending to drop. Something that the SDN was better in this test, was the fact that netdiscover tool discovered only the controller and not the hosts. In contrary, netdiscover tool discovered all the hosts that were connecter to the router in the traditional network.

In addition, NMAP tool performed a scan for the both types of network. The results in this testing were expected because ONOS uses its local address and a router from a traditional network uses much more. The advantage of the SDN in this test is that uses only port 8181 instead of traditional router that was using 21/23/53/80. The less ports are open the less chances they are for vulnerabilities. Also, a good practice is to use only the ports that are required.

Moreover, both networks respond almost excellent to a DDOS attack. An attack of type DDOS performed to both networks using hping3 Linux tool. We tried to flood both machines with packets so to make them unavailable but both of them return intact. The issue that we noticed in this testing was that during the attack, SDN web login turned out to respond very slowly but it was still available.

Both networks were vulnerable to the dictionary attack. In SDN, the credentials were exploit very easily and quick, using the Burp suite tool. Furthermore, Linux tool hydra performed the respective test for the traditional network and exploit the username after two hours. We believe that if we were waiting for a couple of hours more, maybe password would be exploited also. The reasons that the credentials exploited was that we had not change the defaults. Very good ways to prevent these attacks are to require strong passwords, proper protection of credentials, two factor authentication, and configuring password control. In the same way, all default passwords and default usernames must change during the initial setup.

An intelligent SDN network can provide network resilience because of the abovementioned. SDN respond better in the tests in contrast with the traditional network although this technology is much “younger” that the traditional. Even though these entire tests performed in a lab environment, it is possible to partly validate that a controller and generally SDN can avoid breaches and keep its availability high. As per the results, SDN controller can protect the web applications that are connected through the API's and be able to provide a safe environment.

8. Conclusion

This thesis focused in identifying whether SDN networks is a viable solution and offers the potential for more efficient and secure networks. The main purpose of the thesis is to determine whether SDNs can offer the requested security in cyber range environments. After running several tests and analyzing SDN technology, software-defined network can offer network security as efficient as the traditional networks. In some assessments, SDN and traditional had the same results but there is a room for improvement for both of the technologies.

SDN technology is capable of improving network security instead of the traditional hardware-based network. Furthermore, there are plenty of solutions and applications provide enhanced security through SDNs. In this thesis, ONOS controller used as a solution to test the security of an SDN technology and it did well. Considering everything, SDN can enhance security because its architecture can help in detecting attacks earlier than in a traditional network. Because SDN allows dynamic, programmatically capable network configuration it allows many professionals to innovate by designing new protocols or application. In this way, the network security can have a positive impact and provide new features for the information security.

SDN network is a continuous development concept that is improving and we are certain that it will provide many capabilities and make networks more resilient in comparison with the traditional network. Now, technology is driven to cloud solutions and is unquestionable that the technology of the SDN is more suitable than the traditional one.

8.2 Limitations & Future work

An amount of difficulties encountered during this research, mostly in installing ONOS controller. Installing and configuring ONOS with mininet was a complex task as we have never done it before and this was the first time. In addition, we noticed that there were not a many prior research studies and this limited the literature review.

Regarding future work, a more comprehensive security penetration can be done, using all the tools of Kali Linux suite. Also, we could discover more applications to enhance security of the controller and the SDN in general. On the subject of vulnerabilities, we would like to perform DDOS and dictionary attacks in several other SDN controllers and compare the results.

References

- [01] IBM Technology consulting and services, "SDN Versus Traditional Networking Explained", August 2019
- [02] Linda Rosencrance, Jennifer English and John Burke "Software-defined Networking", August 2018
- [03] Michael Cooney, "What is SDN and where software-defined networking is going", April 2019
- [04] Hyojoon Kim, Nick Feamster "Improving network management with software defined networking", IEEE Communications Magazine , Volume: 51 , Issue: 2 , February 2013
- [05] Wenfeng Xia, Yonggang Wen, Chuan Heng Foh, Dusit Niyato, Haiyong Xie, "A Survey on Software-Defined Networking" IEEE Communications Surveys & Tutorials , Volume: 17 , Issue: 1 , Firstquarter 2015
- [06] Diego Kreutz, Fernando M. V. Ramos, Paulo Esteves Veríssimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig, "Software-Defined Networking: A Comprehensive Survey", Proceedings of the IEEE , Volume: 103 , Issue: 1 , Jan. 2015
- [07] Scott Fulton, "How software-defined networking changed everything", May 2018
- [08] Sahrish Khan Tayyaba, Munam Ali Shah, Omair Ahmad Khan, Abdul Wahab Ahmed, "Software Defined Network (SDN) Based Internet of Things (IoT): A Road Ahead", ICFNDS '17: Proceedings of the International Conference on Future Networks and Distributed Systems July 2017 Article No.: 15
- [09] "SDN architecture" Issue 1 June, 2014 ONF TR-502 information available at https://www.opennetworking.org/wp-content/uploads/2013/02/TR_SDN_ARCH_1.0_06062014.pdf
- [10] SEAN MICHAEL KERNER APRIL 29, 2013 OpenFlow SDN Inventor Martin Casado on SDN, VMware, and Software Defined Networking Hype information available at <https://www.enterprisenetworkingplanet.com>
- [11] TEN – Università di Genova Talk @ IEIIT – Consiglio Nazionale delle Ricerche (CNR) Genova 28 Marzo 2014 <https://cseweb.ucsd.edu/classes/fa16/cse291-g/applications/ln/SDN.pdf>
- [12] From Wikipedia, the free encyclopedia <https://en.wikipedia.org/wiki/OpenFlow>
- [13] From Wikipedia, the free encyclopedia information available at https://en.wikipedia.org/wiki/Open_Networking_Foundation

- [14] From Wikipedia, the free encyclopedia information available at https://en.wikipedia.org/wiki/Software-defined_networking#History
- [15] 2021 Juniper Networks, Inc. <https://www.juniper.net/us/en/solutions/sdn/what-is-sdn/>
- [16] 2020 Cisco Solutions, “Software-Defined Networks” information available at <https://www.cisco.com/c/en/us/solutions/software-defined-networking/overview.html>
- [17] Michael Cooney, Network World APR 16, 2019 “What is SDN and where software-defined networking is going”, information available at <https://www.networkworld.com/article/3209131/what-sdn-is-and-where-its-going.html>
- [18] Rob Tomkins, 2018 “What is SDN” <https://www.ciena.com/insights/what-is/What-Is-SDN.html>
- [19] ONF, 2017 “Software-Defined Network (SDN) Definition” <https://opennetworking.org/sdn-definition/>
- [20] August 26, 2013 Connor Craven “What Is OpenFlow and How it Relates to SDN” <https://www.sdxcentral.com/networking/sdn/definitions/what-is-openflow/>
- [21] Issue 1 June, 2014 ONF TR-502 “SDN architecture” https://opennetworking.org/wp-content/uploads/2013/02/TR_SDN_ARCH_1.0_06062014.pdf
- [22] Wendell Odom, Feb 12, 2020, “Introduction to Controller-Based Networking” <https://www.ciscopress.com>
- [23] O’Reilly Media, Inc. 2021, “Components of an SDN”, information available at <https://www.oreilly.com>
- [24] Yanbiao Li, Dafang Zhang, Javid Taheri, Keqin Li, March 2018, “SDN components and OpenFlow” https://digital-library.theiet.org/content/books/10.1049/pbpc015e_ch3
- [25] IP Cisco, 2018, “SDN Architecture Components” <https://ipcisco.com/lesson/sdn-architecture-components/>
- [26] Connor Craven 2020, “SDN Control Plane & SDN Data Plane” <https://www.sdxcentral.com/networking/sdn/definitions/inside-sdn-architecture/>
- [27] From Wikipedia, the free encyclopedia https://en.wikipedia.org/wiki/Software-defined_networking

- [28] Andrew Froehlich, 2016, "Understanding SDN components and where they're going" <https://searchnetworking.techtarget.com/tip/Understanding-SDN-components-and-where-theyre-going>
- [29] Priyanka Kumari, June 2019, "What is software-defined networking (SDN) and the architecture"
- [30] Arash Shaghghi, Mohamed Ali Kaafar, Rajkumar Buyya, Sanjay Jha "Software-Defined Network (SDN) Data Plane Security: Issues, Solutions and Future Directions" [1804.00262.pdf \(arxiv.org\)](https://arxiv.org/abs/1804.00262)
- [31] Mingwei Yang, Jan 2018, "SDN based in-band adaptive coding by distributed pseudonoise preamble detection in optical networks" https://www.researchgate.net/figure/Architecture-of-the-proposed-SDN-control-plane_fig1_322814030
- [32] Karthik Madhava, April 2013, "[What is Software-Defined Networking \(SDN\) Application](http://lavelle.com/what-is-software-defined-networking-sdn-application/)" (lavelle.com)
- [33] Sandra Scott-Hayward, Christopher Kane and Sakir Sezer, "Operation Checkpoint: SDN Application Control," in IEEE 22nd International Conference on Network Protocols, Washington, DC, USA, 2014
- [34] Kevin Benton, L. Jean Camp ,Chris Small, "OpenFlow Vulnerability Assessment," in Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, New York, 2013.
- [35] Jérôme François, Issam Aib, Raouf Boutaba. FireCol: A Collaborative Protection Network for the Detection of Flooding DDoS Attacks. IEEE/ACM Transactions on Networking, IEEE/ACM, 2012, 20 (6), pp.1828-1841. [ff10.1109/TNET.2012.2194508ff](https://doi.org/10.1109/TNET.2012.2194508)
- [36] Lisa Schehlmann, Sebastian Abt and Harald Baier, "Blessing or Curse? Revisiting Security Aspects of Software-Defined Networking," in 10th International Conference on Network and Service Management (CNSM) and Workshop, Rio de Janeiro, 2014.
- [37] Rowan Kloti, Vasileios Kotronis, Paul Smith, "OpenFlow: A Security Analysis," in International Conference on Network Protocols (ICNP), 2013
- [38] IST-153 Workshop on CYBER RESILIENCE Resilience and Security in Software Defined Networking Camen Mas-Machuca, Senior Member, IEEE, Petra Vizarreta, Raphael Durner, and Jacek Rak, Member, IEEE
- [39] A. Aydeger, K. Akkaya, M. H. Cintuglu, A. S. Uluagac and O. Mohammed, "Software defined networking for resilient communications in Smart Grid active distribution networks," 2016

- IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 2016, pp. 1-6, doi: 10.1109/ICC.2016.7511049.
- [40] S. Sezer et al., "Are we ready for SDN? Implementation challenges for software-defined networks," in IEEE Communications Magazine, vol. 51, no. 7, pp. 36-43, July 2013, doi: 10.1109/MCOM.2013.6553676.
- [41] R. Amin, M. Reisslein and N. Shah, "Hybrid SDN Networks: A Survey of Existing Approaches," in IEEE Communications Surveys & Tutorials, vol. 20, no. 4, pp. 3259-3306, Fourthquarter 2018, doi: 10.1109/COMST.2018.2837161.
- [42] P. Vizarreta, C. M. Machuca and W. Kellerer, "Controller placement strategies for a resilient SDN control plane," 2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM), Halmstad, Sweden, 2016, pp. 253-259, doi: 10.1109/RNDM.2016.7608295.
- [43] <https://wiki.onosproject.org/display/ONOS/ONOS>
- [44] SAYAN KUMAR PAL, Feb 2021, "Software Engineering, Spiral Model"
- [45] 2021 Mininet Project Contributors <http://mininet.org/>
- [46] 2021 Shay Chen, Netsparker <https://www.netsparker.com/>
- [47] CWE-1004: Sensitive Cookie Without 'HttpOnly' Flag
- [48] 2000 Jianxin Jeff Yan Computer Laboratory, Cambridge University, UK. "DENIAL OF SERVICE: ANOTHER EXAMPLE"

