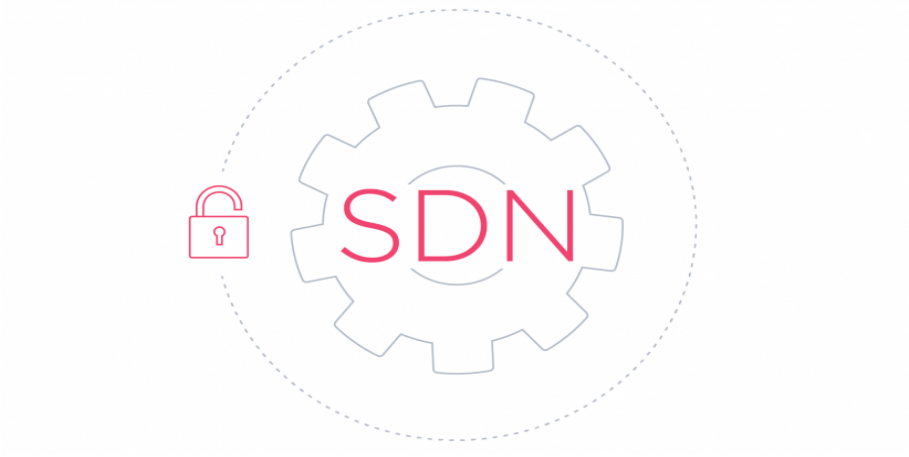
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores



Penetration Testing of Software Defined Networks through Web Applications

RICARDO MIGUEL NUNES ANDRADE

Licenciado em Engenharia Eletrónica, Telecomunicações e Computadores

Dissertação para obtenção do Grau de Mestre
em Engenharia Eletrónica e Telecomunicações

Orientador : [Doutor] Pedro Renato Tavares de Pinho

Júri:

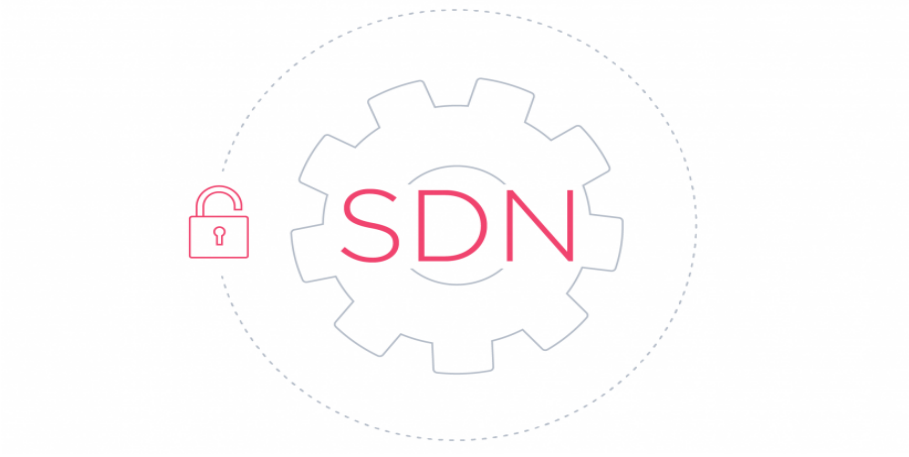
Presidente: Professor Doutor Nuno Cruz

Vogais: Professor Doutor Fernando Ramos
Professor Doutor Pedro Pinho

Dezembro, 2019

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores



Penetration Testing of Software Defined Networks through Web Applications

RICARDO MIGUEL NUNES ANDRADE

Licenciado em Engenharia Eletrónica, Telecomunicações e Computadores

Dissertação para obtenção do Grau de Mestre
em Engenharia Eletrónica e Telecomunicações

Orientador : [Doutor] Pedro Renato Tavares de Pinho

Júri:

Presidente: Professor Doutor Nuno Cruz

Vogais: Professor Doutor Fernando Ramos
Professor Doutor Pedro Pinho

Dezembro, 2019

Acknowledgements

Antes de mais quero agradecer ao meu orientador Pedro Renato Tavares de Pinho por todo o apoio durante a realização desta dissertação, por toda a disponibilidade e por todo o trabalho.

Agradecer à Infinera, em especial a toda a equipa do SDN, por toda a ajuda e acima de tudo por possibilitar e permitir a realização de um trabalho final de mestrado com base num produto em grande expansão.

Quero também agradecer ao Instituto Superior de Engenharia de Lisboa por me ter dado a possibilidade de realizar este trabalho e ao Instituto de Telecomunicações pelo suporte à realização desta dissertação.

Agradecer aos meus pais, ao meu irmão e à minha família que me acompanhou durante todo o meu percurso académico, que me aconselharam e de certa forma me orientaram quando as coisas não estavam a correr da melhor forma.

Por último, agradecer aos meus amigos e pedir desculpa por todos os convites rejeitados pois caso contrário não teria sido possível finalizar este trabalho.

Um obrigado a todas as pessoas que não foram mencionadas mas que tiveram um grande impacto na possibilidade da realização deste trabalho.

Abstract

Networks have been evolving along the years, but nowadays, the network elements need great capabilities to process big networks, for this case Software Defined Network (SDN) appeared to centralize all the control of the network in one place. There are two types of SDN solutions; the transport and the packet, and both have the same theoretical purpose, which is to abstract the network elements such as routers and switches, from the complexity of the management and control of the network itself. The main difference between both solutions states that; the transport solution has the primary purpose of establishing and provision services within the network while the packet solution is more packet-oriented, which implies the use in traditional networks.
This work will focus on the security of a real and deployed solution pack containing web applications that are connected to SDN. The solution belongs to Infinera, and this work purpose is to investigate how does SDN responds to various attacks deployed against a web application that directly connects to it. It was used a particular Linux distribution named Kali to execute the tests; were used different tools, and some scripts developed in order to automate the test procedures.
A first impression of the results of the tests was that even though security is a very un-explored paradigm when regarding SDN, this solution handled very well the deployed attacks and, therefore, a very secure solution regarding these tests. It was also essential to notice that SDN as independent software, concerning the web application, it was not very affected by thus attacks. Some vulnerabilities were discovered, and for thus, suggested a remediation solution.

Keywords: Software Defined Network, Transport Controller, Security, Penetration Testing, Web Applications ...

Resumo

As redes têm sofrido alterações ao longo dos anos, contudo nos dias de hoje, os elementos de rede necessitam de uma grande capacidade para processar redes de grande dimensão, para tal apareceu o SDN para centralizar todo o controlo da rede num único lugar. Existem dois tipos de soluções SDN; de transporte e de pacotes, e ambas têm o mesmo propósito teórico, que consiste na abstração dos elementos de rede tais como routers e switches, da complexidade de gestão e controlo da própria rede. A principal diferença entre ambas as soluções começa nos nomes; a solução de transporte tem o principal propósito de estabelecer e provisionar serviços na rede enquanto que a solução de pacotes é mais orientada aos pacotes o que implica o uso em rede tradicionais. Este trabalho vai se focar na segurança de uma solução real e implementada que contem aplicações web que se encontram ligadas ao SDN. A solução pertence à Infinera, e este trabalho propõe-se a investigar a resposta do SDN a vários ataques lançados contra a aplicação web que está diretamente ligada. Foi usado uma distribuição particular de Linux denominada de Kali para executar os testes; foram usadas diferentes ferramentas, e foram desenvolvidos alguns scripts com o objetivo de automatizar os procedimentos de teste. A primeira impressão dos resultados dos testes foi que apesar de a segurança ser um paradigma pouco explorado a respeito do SDN, esta solução tem respostas muito positivas a todos os ataques que foram executados e, portanto, é uma solução segura a respeito destes testes. Foi também essencial verificar que o SDN como software independente, relativamente à aplicação web, não foi muito afetado pelos ataques. Algumas vulnerabilidades foram descobertas, e para tais, foram sugeridas uma solução.

Palavras-chave: Redes definidas por software, Controlador de transporte, Segurança, Testes de penetração, Aplicações web...

Contents

List of Figures

Acronyms

AES	Advanced Encryption Standard.
API	Application Programming Interface.
ARP	Address Resolution Protocol.
BGP	Border Gateway Protocol.
CSRF	Cross-site Request Forgery.
DAST	Dynamic Application Security Testing.
DDoS	Distributed Denial Of Service.
DoS	Denial Of Service.
ECDH	Elliptic-curve Diffie–Hellman.
ECDSA	Elliptic Curve Digital Signature Algorithm.
FMS	Fluhrer, Mantin and Shamir attack.
GCM	Galois Counter Mode.
GPL	General Public License.
GUI	Graphical User Interface.
HTML	HyperText Markup Language.
HTTP	Hypertext Transfer Protocol.
HTTPS	HyperText Transfer Protocol Secure.
ICMP	Internet Control Message Protocol.
ICPC	Internet Protocol Control Protocol.
IDE	Integrated Development Environment.
IDS	Intrusion Detection System.
IEEE	Institute of Electrical and Electronics Engineers.
IP	Internet Protocol.
IPS	Intrusion Prevention System.
ISO	International Organization for Standardization.

ITU	International Telecommunications Union.
JSON	JavaScript Object Notation.
LCP	Link Control Protocol.
MAC	Media Access Control.
MEF	Metro Ethernet Forum.
MPLS	Multi Protocol Label Switching.
NEs	Network Elements.
NETCONF	Network Configuration Protocol.
NMAP	Network Mapper.
OSI	Open Systems Interconnection.
OTDR	Optical time-domain reflectometer.
OWASP	Open Web Application Security Project.
POP3	Post Office Protocol v.3.
PRBS	Pseudorandom Binary Sequence.
PTW	Pychkine-Tews-Weinmann.
QoS	Quality Of Service.
REST	Representational State Transfer.
RESTCONF	Representational State Transfer Configuration Protocol.
SAST	Static Application Security Testing.
SDN	Software Defined Network.
SHA	Secure Hash Algorithms.
SMTP	Simple Message Transfer Protocol.
SNMP	Simple Network Management Protocol.
SP	Service Provider.
SPF	Single Point of Failure.
SSH	Secure SHel.

TL1	Transaction Language 1.
TWAMP	Two-Way Active Measurement Protocol.
URL	Uniform Resource Locator.
WAN	Wide Area Network.
WEP	Wired Equivalent Privacy.
WPA-PSK	Wi-Fi Protected Access-Pre Shared Key.
XML	Extensible Markup Language.
XSS	Cross Site Scripting.
ZAP	Zed Attack Proxy.

1

Introduction

Telecommunications represents one of the backbones of the economy that is continuously evolving during the years. Before the emergence of the Internet and other data networks, telecommunications had a clear meaning for people to communicate with each other at a distance. The primers invented a system with fire and drums, which allows the evolutions of humans to explore other ways that lead to the cable use communications that allow the communication between countries and continents. The progress leads to a new era, wireless communications, that nowadays are the most common way for people and companies to communicate and do business. This evolution leads to new services and concepts that intend to improve the lifestyle and business of people dynamically because technologies as they evolve also need more bandwidth, more Quality Of Service (QoS), and more automation in order to accompany the evolution of technologies in people lives.

1.1 Thesis background

Along the years technologies like xDSL (Digital Subscriber Line), POTS (Plain Old Telephony System), PDH/SDH (Plesiochronous/Synchronous Digital Hierarchy), Ethernet, GSM (Global System for Mobile), WDM (Wavelength Division Multiplex), OTN (Optical Transport Network) and LTE (Long Term Evolution) constitute the revolutions in telecommunications. These advances mean a lot for

the society because it allows more people to communicate faster and more efficient with higher QoS(Quality of Service). However, for the service provider, it represents high costs, because it is essential to maintain the interoperability between types of equipment from different generations and keep everything running without LoS (Loss of Service).

The reason why there are different types of equipment from different generations is a synonym of permanent expenses for the service provider because there is much maintenance of older equipment. The interoperability also requires much know-how, and the need for service providers companies to extend their internal contributors, which represents higher costs. At this level of technologies there are some solutions that companies use in order to manage and control their networks; protocols like Simple Network Management Protocol (SNMP) or Internet Control Message Protocol (ICMP) for managing and protocols like Internet Protocol Control Protocol (ICPC) or Link Control Protocol (LCP) for control and to keep everything running the Network Elements (NEs) like routers and switches must have higher processing capabilities that represent more expenses to the company.

These protocols relate to the packet solution, and over the years, with constant evolutions, the fiber appeared and led the communications to a new paradigm. This paradigm brought various questions like how will be made the management and the control of the networks or what impacts it would have on the networks? SDN is a new uprising technology that intends to centralize the control of the network in one place, in the SDN controller. SDN brings new ideas and new concepts that need to be understood, but the primary purpose is:

- Centralize the control;
- Allow the interoperability between physical equipment;
- Dynamically take actions, according to network events;
- Provisioned services and scheduled network actions.

There are two types of SDN solutions, the packet solution, and the transport solution. The main difference between each solution is that in the packet solution, the packets may or may not flow through the controller in order to know which is next-hop and keep the traffic moving. While the transport solution does not rely on packets, the primary purpose of this solution is to orchestrate, establish, and provisioning services in the network. In other words, the packet solution works in the electrical domain while the transport solution works in the optical domain. What concerns to this work, the transport solution is the one to be discussed.

1.2 Thesis motivation

The appearance of SDN is recent, and the security is really important topic because if the product is not secure the client network and information where SDN is going to be deployed might be compromised. Therefore it is essential to study the impacts and vulnerabilities that might be presented in a real solution. SDN has an all is just the controller, but there is an external tool that communicates with the controller to perform actions in the network. This tool is connected to the Internet and a point of the entrance from outside attacks that might harm the controller. Which means that is very important to understand which impacts thus attacks have on the SDN controller and understand how it is the reponse to thus attacks from the SDN controller. Infinera is a company that has an SDN solution. The following figure 1.1 shows the solution.

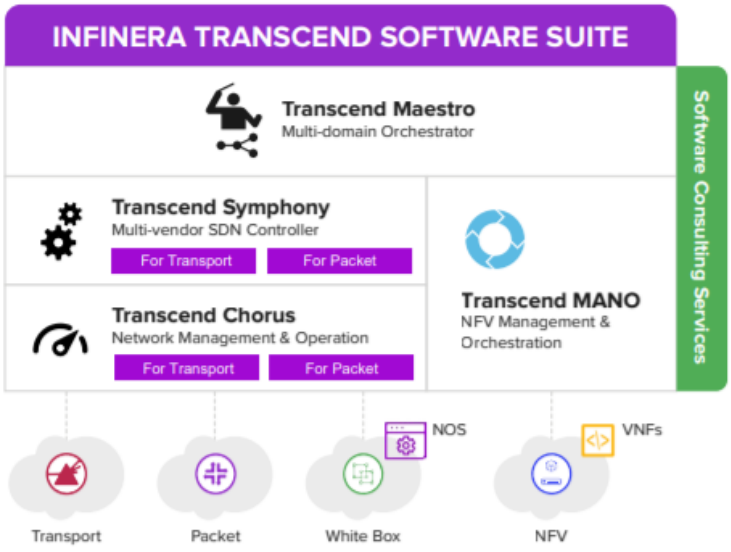


Figure 1.1: Infinera soluiton pack[1]

In this solution pack, there is a tool named Transcend Maestro, used by clients to orchestrate actions in the network, and this is the tool that is on the internet and connects to the SDN controller. It is also possible to observe that there are different controllers for the transport and the packet solutions.

1.3 Objectives

The present dissertation consists of the development of in-depth knowledge about the concept of SDN networks, their architecture, and their elements to analyze the vulnerabilities of a real solution between the SDN controller and web applications directly connected. In order to achieve the objectives is proposed that first of all, knowledge about SDN, web applications, and security regarding both matters be acquired. Secondly, the intention is to exploit and understand which exhibited vulnerabilities are in a real solution with SDN and a web application, create scripts to automate the tests and analyze the results in order to report the errors so in the future thus vulnerabilities can be fixed and that Infinera solution pack, is secured rearding external threats.

1.4 Thesis structure

The present document is organized into five chapters.

- Chapter 1 - The first chapter of the presented document is where the objectives are discussed, and the state of the art exhibited. Also, the contributions that this work will have.
- Chapter 2 - The second chapter has presented the evolutions of the new networks and their limitations — the development of the SDN controller concept and its architecture alongside with communication interfaces. This chapter is also dedicated to security matters regarding ethical hacking, penetration testing, web applications, tools, security testing (Black box and White box), and vulnerabilities.
- Chapter 3 - The third chapter centers on the development of the test environment, where is presented the architecture of the network used to perform the tests.
- Chapter 4 - The fourth chapter is concerned about performing the attacks of black box and white box to explore vulnerabilities in the tested web application. Analyse the results and propose solutions accordingly.
- Chapter 5 - This chapter presents a summary of all developed work identifying the overcome problems and proposals of future work regarding these topics.

1.5 Contributions

This work will contribute to improving the security of Service Provider (SP) networks on the process of migration to SDN. This is achieved by designing and exploiting mechanisms to use against the web application that is directly connected to the SDN controller. These tests will help SP to protect their optical networks and make them more resilient to external threats. The mechanisms used in this thesis will be included in the Infinera OEM solutions and documentation, helping our customers in securing their infrastructure.

2

SDN Networks

SDN came to revolutionize the traditional networks through the creation of new perspectives to solve some major issues occurring like the overload of the NEs as routers and switches or the difficulty to manage big networks. SDN is a new concept that brings centralized, programable, and flexible purposes for the new era of communications. It all started in 2006 with the project SANE [8]. This project brought the idea of centralization, however it was meant for security purposes. SANE defines a single protection layer that governs all connectivity within the enterprise. All routing and access control decisions are made by a logically-centralized server that grants access to services by handing out capabilities (encrypted source routes) according to declarative access control policies. In 2007, a new project comes to light, the ETHANE [9]. ETHANE was quite close to actual implementation of SDN. A major drawback of ETHANE was that its network policies were compromised by application traffic. The actual development of SDN inspired by SANE and ETHANE was started in 2007 at Standford University by Martin Casado and a group of researchers.

2.1 Limitations of modern telecommunications networks

Over the years, as the telecommunications networks evolve there were always the concern of the retrocompatibility with older equipments. Companies like Coriant, Infinera, Verizon, Nokia, Siemens develop poprietary hardware and software that are compliant with the standards from International Telecommunications Union (ITU) , Institute of Electrical and Electronics Engineers (IEEE) and Metro Ethernet Forum (MEF). These entities are resposible to normalize and define processes for the world market to develop standards for hardware and software.

Nowadays there are very different solutions from different vendors operating together, which means lots of complexity, high costs and lots of know how when maintenance, in different equipments, are needed because different equipments use different languages and configuration settings. In order to operate with them, different communication protocols are used, like telnet or Secure SHel (SSH). This is only possible because all equipments are standard.

In order to manage the networks are used protocols like SNMP, Network Configuration Protocol (NETCONF) and Transaction Language 1 (TL1). These protocols are very used, and there are some new extensions like Representational State Transfer Configuration Protocol (RESTCONF) that is not intended to replace NETCONF, but rather to provide an HTTP interface that follows Representational State Transfer (REST) principles and is compatible with the NETCONF datastore model. These protocols are fundamental in the management of the network because they not only monitor the network but also distribute access lists, configure different types of equipment simultaneously, among other important things. It also allows web applications to connect to the northbound interface of the SDN controller in order to execute actions.

The increasing of the networks complexity means that simple tasks like maintencance and control take a lot of time and effort from the companies which is translated to loss of money. For this reasons companies are seeking for solutions that allows them to save that kind of money, by having an automated solution that solves the main issues of maintenace, versatability and control.

The tradicional networks possess:

- High complexity configuration;

- Lack of generic configuration mechanisms;
- Very complex configurations;
- High maintenance costs and effort.

2.2 Development of SDN concept

Since the late 70's [10] the Standardization of the protocols has been handled by the International Organization for Standardization (ISO). This entity conduct a program to develop general standards and methods of networking. The communications protocols have been followed the same conduct of other protocols, however the communications protocols have their own model to follow, the Open Systems Interconnection (OSI) like the figure 2.1

Figure 2.1: OSI Model [2]

Over the years the architecture of traditional networks has evolved and became more complex, because of the interoperability of the technologies. With the development of the SDN concept the mistake of designing an architecture too simple that could in the future became something really complex, like the architecture of traditional networks. The figure 2.2, is an architecture for SDN and is based in the centralization of the control plane, that is the core of this architecture.

This separation intends to divide the complexity and create eficient responses according to what happens in the network. Taking in account the architecture of traditional networks, where the same information were duplicate over the nots in the network cause a heaviest load in the network that result in an increase of the latency and an inscrease of the complexity of the network.

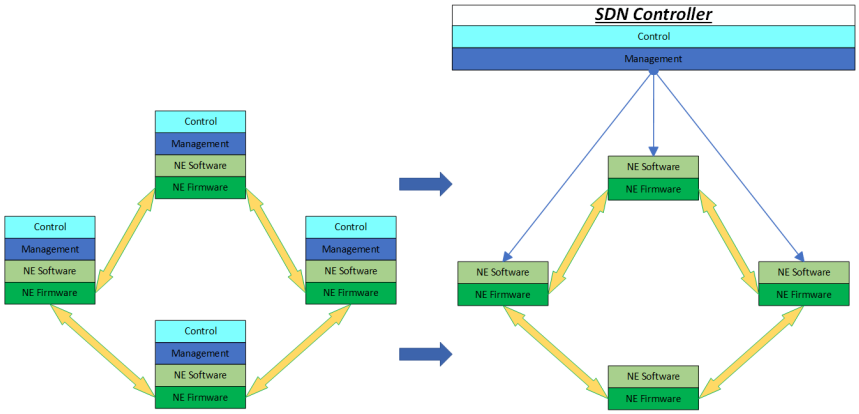


Figure 2.2: a) Traditional Networks b) Centralized networks [3]

2.3 Communication interfaces

SDN architeture is based on one thing, centralization of the network control. This ideia allows the partial unification of the control and management plane. Although the data and the control plane are disjoint they need to communicate, as the control and management plane. The figure 2.3 ilustrates the different layers and how they comunicate.

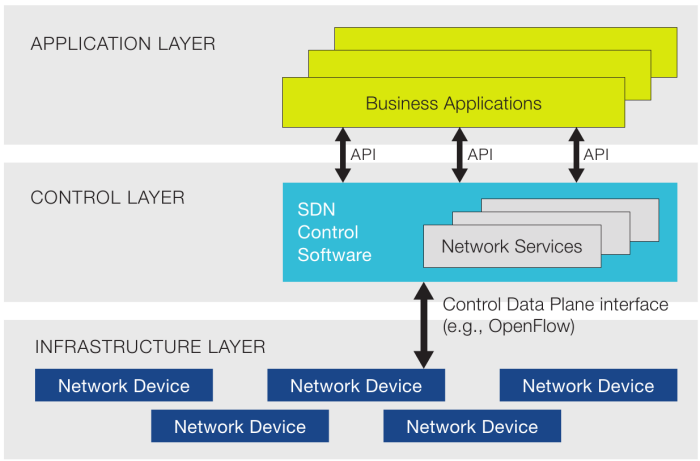


Figure 2.3: SDN Interface Architecture [4]

In order for the management plane to communicate with the control plane there is a Northbound interface and the Southbound interface for the communication between the control and data plane. There is also two more interfaces, thus are the westbound and eastbound interfaces that allow the communication with other SDN controllers.

2.3.1 Northbound Interface

Is through this interface that the programming of all the network using a reserved, unique and secure channel, occur. This interfaces establish the link between the applications running in the management plane and the control plane. Accordingly to Sarwar Raza (HP), David Lenrow (Plexxi) from Open Networking Foundation **[11]**, the NorthBound has three important goals, the first is to provide extensible, stable, and portable NBI Application Programming Interface (API) to controller, network services, and application developers. The second goal is to increase portability of software designed to interact with SDN controllers. This will be done by defining multiple APIs at differing levels of abstraction and breadth of domain to allow network behavior to be more programmable, and automated. The last is to ensure that controller vendors are free to innovate, using API extensions, within their designs. Depending on the type of controller used, the way how applications and controllers communicate can differ. Usually the communication is done trough common and very used programming languages like, Hypertext Transfer Protocol (HTTP), JavaScript Object Notation (JSON), Extensible Markup Language (XML). At Infinera SDN controller the communication is via HTTP using REST through the use of GET, POST, PUT and DELETE methods that follow the API structure.

2.3.2 Southbound Interface

The southbound interface can be based in several protocol such as: OpenFlow, OVSDB, ForCES **[12]** and others like SNMP, NETCONF, BGP. This feature allows the compatibility between different vendors controllers. The instruction set of the forwarding devices is defined by the southbound API, which is part of the southbound interface. Furthermore, the southbound interface also defines the communication protocol between forwarding devices and control plane elements. This protocol formalizes the way the control and data plane elements interact.

2.3.3 Eastbound / Westbound Interfaces

SDN is a single centralized controller and for that reason it has a single point of failure, the controller himself. Accordingly, to **[13]**, there are two main propositions to tackle the main issue above with multiple controllers is the logically centralized control planes, like is presented in figure 2.4.

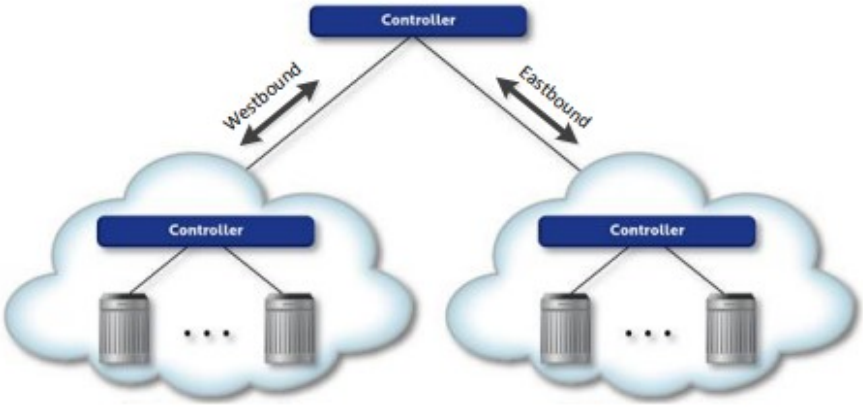


Figure 2.4: SDN distributed system (Eastbound and Westbound Interface) [5]

The logically centralized control plane balances charges between controllers and uses a shared database to unify decisions. It was proposed to extend SDN for large centralized networks, where each controller manages its domain and centralizes the necessary data to other controllers. The primary use of this category is in large data centers and Wide Area Network (WAN) networks that suffer from high costs and latency, due to the complexity of the infrastructure and protocol (e.g., Border Gateway Protocol (BGP) and Multi Protocol Label Switching (MPLS)) that handles the traffic.

2.4 Penetration testing in SDN

Security is a concept that is present in our every day today, and because as a client, security is something implied when we start trusting the operator, and because of this trust created between client and operator, is the responsibility of the operator to guarantee the security of data and make sure that nothing gets modified.

For the operator to guarantee the security, it needs to perform very regular tests over their hardware and software. This regular testing, like everything, needs a plan to be executed in order to keep things organized and it is an easy way to understand the expected work-flow to be executed.

In the security world, these types of tests have a specific name, called, penetration tests. Penetrating testing [14] is the process to identify security vulnerabilities in an application by scanning, analyze and exploiting the system or network, in order to find weaknesses and week points, through an authorized simulated

attack. It is crucial to understand that the attacks must be authorized. Otherwise, it is a crime.

The purpose of penetration testing is to guarantee that essential data flows safely avoiding hackers that might gain unauthorized access to the system. They achieve this by digging through the software in order to expose a loophole or a defect that grants them full access to the computer.

As SDN represents a Single Point of Failure (SPF) and with such massive, dangerous and evolving cyber-attacks happening, it has become unavoidable to perform penetration testing on a regular basis to protect the information systems against security breaches. Penetration testing is primarily needed for:

- Business or significant data that must be secured while shifting it between different systems or across the network;
- Several buyers that are requesting pen testing as a section of the software release period;
- Securing user data;
- Discover security vulnerabilities in applications;
- Detecting loopholes in the systemto to avoid network downtime;
- The evaluation of the company impact concerning successful attacks allows them to secure and fix their breaches;
- Satisfy the information security agreement in the organization;
- An effective security strategy in the organization.

The enumerated penetration tests are a way of categorizing different types of pen tests, exists a second way to organize the types of penetration tests into three categories, black box, white box and grey box penetrating testing.

- **Black box penetrating test:** Is an approach, where the tester assesses the target system, network or process without the knowledge of its details. They have a very high level of inputs like URL or company name which they penetrate the target environment. No code examined here.
- **White box penetrating test:** Is an approach, where is provided to the tester the complete details about the target environment – Systems, networks, OS,

IP addresses or the source code. It also examines the code and finds out design and development errors. This test is a simulation of an internal security attack.

- **Grey box penetrating test:** Is an approach, where the tester does not know the full details about the target environment. This test is a simulation of an external security attack.

2.5 Ethical hacking

Ethical hacking [15] is what distinguishes a good hacker from a bad one. Ethical hacking apart from testing duties, are associated with other responsibilities. The main idea is to think like a malicious hacker and proceed the same way in order to replicate a malicious hacker at work instead of exploiting the vulnerabilities for malicious purposes.

- An ethical hacker is responsible for some things, and one of them is to seek countermeasures to thicken and reinforce the system's defenses. An ethical hacker might also have to employ all or some of these strategies to penetrate a system;
- Uses port scanning tools to scan their systems and find open ports;
- Test application installations and make sure that they have no vulnerabilities;
- Apply different social engineering techniques like shoulder surfing to obtain entrance to significant data or play the sympathy card to deceive employees to provide their passwords;
- Strive to evade Intrusion Detection System (IDS), Intrusion Prevention System (IPS), honeypots, and firewalls.
- Sniffing networks, bypassing and breaking wireless encryption, and capturing web servers and web applications sessions;
- Ethical hackers may also handle issues associated with laptop theft and employee deception.

Detecting how well the organization reacts to these and other tactics help test the strength of the security policy and security infrastructure. An ethical hacker attempts the same types of attacks as a malicious hacker would try — and then help organizations strengthen their defenses. There are four types of hackers [16] the grey hat hacker, white hat hacker, black hat hacker, and red hat hacker. All of them have different ways to contribute to the hacker community. While a penetration tester is an ethical hacker, also known as a white hat hacker, the black hat hacker is the complete opposite because it explores the vulnerabilities and access without authorization to do so. This type of hacker is malicious, and the only purpose of hacking is to steal information and use it for their gain, breaking the rules and commit crimes while doing so. The grey hat hackers, on the other hand, are a mix of both worlds between the white hat hacker and the black hat hacker, even though they break laws and committed crimes they do it with the only purpose of learning and finding problems, where often they disclose the problems found to the responsible entities. The red hat hacker is a unique mix from the other three types of hackers because they are often black hat hackers, but they are employed by the government which became a particular type of white hacker. Usually, they are working for a company. However, their intentions and motivations did not change, and for that reason, they decided to create a new category to insert this exception, they called them red hat hackers. The hacking world is tremendous and by living in a world where everything has computerized a company must have the ability and the concern of protecting their assets against the problems of the outside where the only purpose of the malicious hackers is to tear down companies, steal their money and their personal information.

2.6 Penetration Testing Software

Hacking tools allow the hackers to identify problems, vulnerabilities and breaches in a system or network, however, there is no tool like a Swiss army knife, there are different types of tools with specific applications and is the hacker job to use them to their full potential and use them to the right purpose. This sub-chapter have the purpose to present some of the most used tools by the hackers during their activities and understand the purpose of the tool itself and understand the use of those tools by a white hat hacker. A penetration tester has as their primal job to perform security testing, and they complete this task by keeping the network and the system up to regular scans. Regarding scanning there different types of scanning tools, some tools scan ports, other tools that scan, applications,

code or web applications, and so on.

2.6.1 Free and open source tools

Network Mapper (NMAP) [17] is used to perform port scanning and create maps of the network. It is an available and open source tool and used by a lot of professionals around the world for network inventory, checks for open ports, managing service, upgrade schedules and monitor host or service uptime. This tool [18] essentially is used for network identification and conducting security audits. It applies raw IP packets in original forms to learn what hosts are reachable on the network, what services (application signature and version) those hosts are storing information about, what operating system (fingerprinting) and what type of packet filter/firewalls are being used. NMAP is a console-based tool available in different Linux flavors that also comes with its Graphical User Interface (GUI) version Zenmap (official NMAP Security Scanner GUI) for the ease of use. Metasploit [19] is a vulnerability exploitation tool that can be used to execute various tasks. It is best-known for their open source framework, a tool for developing and executing exploiting code against remote target machines. Metasploit is primarily a security framework that grants the user with valuable information about known security vulnerabilities and benefits to formulate penetration testing and IDS testing plans strategies and methodologies for exploitation. Still, in the same matter of exploitation, the next tool is a wireless hacking tool and is very popular because of their effectiveness in password cracking. The tool is named,Aircrack-ng [20]. It is an 802.11 Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access-Pre Shared Key (WPA-PSK) keys cracking, a hacking tool that can recover keys when sufficient data packets have been captured. The fact that Aircrack-ng implements standard Fluhrer, Mantin and Shamir attack (FMS)([21] - The attack allows an attacker to recover the key in an RC4 encrypted stream from a large number of messages in that stream) attacks along with some optimizations like KoreK (This attack [22], when successful, can decrypt a WEP data packet without knowing the key) attacks, as well as the Pychkine-Tews-Weinmann (PTW) attacks make their attacks more potent. John, the Ripper [23], is a popular password cracking pen-testing tool that is most commonly used to perform dictionary attacks. This tool performs a dictionary attack, where takes text strings samples, from famous and complex words found in a list of most commonly used passwords. By encrypting each one of the passwords on the list, using the same algorithm that was used to encrypt the victim password it is possible to compare the output and

compare to the victim password and validate if the there is a match. Another similar tool to John the Ripper is the THC Hydra [24], where the main difference between them is that the John the Ripper tool is an "offline" password cracker, while THC Hydra is an "online" password cracker. The main difference between the "offline" and "online" method is the way the cracking works. In one hand John the Ripper uses dictionary attacks when the THC Hydra uses brute force attacks to crack the passwords. A brute-force attack [25] consists of an intruder trying multiple passwords or passphrases with the purpose of ultimately guessing correctly. The attacker systematically checks all possible passwords and passphrases until the correct one is found. Alternatively, the attacker can attempt to guess the key which is typically created from the password using an essential derivation function. This is known as an exhaustive key search. The Open Web Application Security Project (OWASP) Zed Attack Proxy (ZAP) [26] is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications. It is designed to be used by people with a wide range of security experience and as such is ideal for developers and functional testers who are new to penetration testing as well as being a useful addition to an experienced pen tester's toolbox. ZAP gives automated scanners as well as several tools that allow cyber specialists to discover security vulnerabilities manually. Recognizing and being able to master this tool would also be advantageous to a career as a penetration tester. Nikto [27] is another classic hacking tool that many pen testers like to use. It is an Open Source General Public License (GPL) web server scanner which offers extensive tests against web servers for various items, including over 6700 potentially hazardous files, checks for out-of-date variants of over 1250 servers, and variant-specific problems on over 270 servers. It additionally checks for server configuration details such as the presence of various index files, HTTP server options, and will strive to catalog installed web servers and software. Scan items and plugins are regularly updated and can be automatically updated. Nikto will get pulled up by any decent IDS tool, so it is valuable when administering a white-hat/ white-box pen test. Another tool, Wireshark [28], is a standard tool in networking. It is the network protocol analyzer tool which let us examine the traffic across the network. It has features like live capture packets and analyzes packets to discover different things associated with the network by monitoring the data at the micro-level. Wireshark has been highly developed, and it includes filters, color-coding and other features that let the user dig deep inside the network traffic and examine single packets. Another, very important tool for a penetration tester is the Maltego [29] tool. This is a digital forensic tool that is used

to deliver an overall cyber threat picture to the enterprise or local environment in which an organization operates. Matlego is a great tool for a black box or grey box penetration test because it can be applied to collect data about the network because the main focus of Maltego is examining real-world associations linking information that is openly accessible on the internet. Includes foot-printing internet infrastructure as well as collecting data about the people and organization. Maltego also produces results in an extensive range of graphical layouts that support clustering information, which makes comprehending relationships instant and accurate. It is also possible to see hidden connections. This all disaggregated tools are handy, and they are the most common tools used by penetration testers in their daily job. They believe that in order to do a good job they need to have the right tools and in order to do that, having this all desegregated tools is not the right way to do so. For that reason, the Offensive Security [30] team release the Kali Linux that came through a rebuild of the old Backtrack Linux distribution and adhering completely to Debian development standards. Kali Linux [31] is a Debian-based Linux distribution aimed at advanced Penetration Testing and Security Auditing. Kali contains several hundred tools which are geared towards various information security tasks, such as Penetration Testing, Security research, Computer Forensics, and Reverse Engineering. This is a specially tailored tool designed to satisfy the needs of penetration testing professionals. What concerns to this document the Kali Linux, and the tools available, will be used the ones focusing on web application penetration testing.

2.6.2 Web Applications

Web applications offer accessibility to companies and customers alike. Their ubiquity makes them a favorite offensive target for cybercriminals. As a result, web application security testing, or scanning and testing web applications for risk, is crucial. Like it was presented previously, Infinera SDN product has a web interface available to the clients for them to communicate with the system. It is crucial to test that web interface and validate if has any implications that may be carried over the SDN controller, and understand how thus vulnerabilities open a window of exposure to the controller and the network itself. In order to understand a problem we must first identify them. After identifying and scanning the system, how can that problem be tested in order to reproduce it and make sure that the problem no longer exists? For last it will be discussed, and it will be validated if that fixed leaves others exposers and others vulnerabilities that may

harm the system.

2.6.2.1 Security and Scanning

Scanning [32] the web applications for vulnerabilities it represents some security measures that in today's threat scenarios are not optional. A Web application to work it needs to freely allow traffic through a variety of ports that may or may not require authentication. For that matter, hackers often attack the most commonly used ports, like the port 80 (HTTP) or Port 443 (HyperText Transfer Protocol Secure (HTTPS)), even e-mail ports like the port 25 (Simple Message Transfer Protocol (SMTP)) and port 110 (Post Office Protocol v.3 (POP3)). The fact is that Web application are the preferred ways of exploiting breaches by the hackers, proofed by Verizon Data Breach Investigations Report 2018 [33], it implies that the security requirements must be more prominent, and in order to do that the regular and continuously monitoring and scanning of Web applications decrease the odds of being hack through a Web application breach exponentially. This continuously scans and monitoring of the Web Applications, lead to proactively identify vulnerabilities and remediate them before a breach occurs because this leaves the company security one step ahead of attackers. When performing this types of preventions like scanning and monitoring it is important to keep in mind to always choose the best tools available and not choosing the ones that are exclusively free, because despite the fact of a tool being free might be good for the company financially it may lead to inconclusive reports, the one calls false negative. When a report says that everything is all right and the security of the network is on top when actually in that precise moment they are being hacked. There are some tools like AppSpider, BurpSuite, NetSparker, Arachni, W3af, and Vega that are one of the tops in scanning Web applications and are very useful to all penetration testers because they not only search for vulnerabilities but also they mapped the entire network in order to identify and remediate the problem more quickly and efficiently. An essential requirement is that reports are easy to read and the output information returned from the tools have the information exposed in a dynamic form. Beyond the critical report's information to make use of the data that the scanner finds is not enough, is also very important to convert that reports information into a specific remediation plan. This type of plan can provide prioritization of tasks, including what needs to be fixed, why and by when. To distinguish a great vulnerability scanner form a not so great one is that the great one has the ability and allow the track and measure of the data within the scanner software itself. The threats are here, and the threads landscape

is continuously growing and developing. Given the number of web applications that people interact with daily, whether for business or personal use, it is critical that these apps are protected. By scanning the applications regularly, it will be possible to identify and remediate vulnerabilities before a breach occurs in order always to stay one step ahead of attackers.

2.6.2.2 Security Testing

The use of Web application is in continuous growth because they offer convenience to businesses and customers' worldwide, makes them a popular attack target for cybercriminals. As a countermeasure for thus attacks, web application security testing [34], or scanning and testing a web application for risk, is essential for the protection and the integrity of the customer's data. Like it was presented, accordingly, to the Verizon Data Breach Investigations Report 2018 [33], the web application is one of the targets of the preferred attacks in confirmed data breaches. They report that the majority of the attacks using web applications consist on getting personal information and payment, but these results presented in different sectors, case in proof the personal information, is a target of the educational sector while the payment target is more common in the retail sector. In the case of the educational sector, the way of obtaining personal information is due to denial of service attacks through web application flaws. The same happens in the retail sector where the denial of service is the most responsible for obtaining the high number as a preferred attack. It is also an interesting fact that 68% of breaches took months or longer to discover, and in many cases, it is not even the organization itself that spots the breach. A very concerning fact is that, even if a company follows the best practices to protect itself against common web application attacks. Like the OWASP top 10 common web applications attacks [35], the truth is that it may not be sufficient to stop this epidemic. Web applications can be so complex that might confuse the systems design to detect an attacker's intrusion automatically, thus are the IDS [36] and IPS [37], and when working alone might not be sufficient, that is why the web application security testing is so necessary, because it can fill some of the gaps that thus systems bring and leave wide open. In terms, of web application security testing, there are three different approaches the Dynamic Application Security Testing (DAST), the Static Application Security Testing (SAST) and the Application Penetration Testing.

- **DAST** – This approach involves looking for vulnerabilities in a web application that an attacker might try to exploit. This method is used to find

which vulnerabilities an attacker can discover in order to exploit them and understand how they could break into the system from the outside. The advantage of this method is that is not required to access the application's source code, which makes this type of tests the ones to perform more regularly and more quickly.

- **SAST** – This is a more inside-out approach, which means, and when comparing with the DAST it scans for vulnerabilities in the web application's source code. Because this approach needs the source code, it allows a more precise and uncover more hidden loose ends and allow the possibility of taking a snapshot in real time of the web application's security.
- **Application Penetration Testing** – This approach, like the name, suggests, it involves a security professional and some required tools. This security professional will try to think like an attacker and proceed the same way an attacker might break into a web application using both their security know-how. The tools have an essential job to find exploitable flaws.

There are a few basic things when it comes to testing the security of web applications. If a system is considered critical and the business is directly dependent on the proper functioning of that system than it should be tested often in order to find security vulnerabilities. Another primary suggestion is to test the security as early as possible, if better during the software design because it is not good policy to leave security testing for the last thing on the stack because any of the vulnerabilities found might compromise the already completed development. One last suggestion has to do with the fact of keeping development teams on track by prioritizing remediation and bug fixing instead of keeping build over the unstable base structure. Web application security is more critical than always. By executing a web application security scanner and developing some basic best practices for both testing and remediation, companies can significantly decrease their risk and help preserve their systems safe from attackers.

2.6.2.3 Vulnerabilities

Like it was presented, web applications are one of the preferred ways, for hackers, to find breaches to compromise the systems and networks, but it is the job of the penetration tester to avoid and prevent thus breaches and an essential thing for a penetration tester or any security employee, for that matter, in order to understand that a breach exists it needs to understand how to exploit the breaches.

The use of specific tools helps in this job but is vital to know what tool to choose. It is clear that nothing is perfect and if there are problems they need to be fixed, but it is essential to understand why the web applications are such a desirable way for hackers. Over the years, professionals were always learning from their mistakes and keep evolving along as the hackers, but one thing that is clear is that over the years the type of attacks, in general, have been the same but directed into different applications. This repetition of attacks allows the OWASP to get together a list of the top 10 application security risks. Thus risks are [35], Injection, Broken Authentication, Sensitive Data Exposure, XML External Entities, Broken Access Control, Security Misconfiguration, Cross-Site Scripting, Insecure Deserialization, Using Components with Known Vulnerabilities and Insufficient Logging&Monitoring. "According to Quocirca and Veracode's State of Software Security Report, Enterprise Testing of Software Supply Chain 65 percent of a typical enterprise application portfolio comes from third parties, yet 90 percent of third-party code does not comply with enterprise security standards such as OWASP Top 10"[38]. The third-party software must allow tests as much as the internal software where everything is connected, and if something is infected, then all the system might be too. The responsibility to test the security of that software should recall to the company who develop that software. If they are releasing it should at least respect the OWASP standards.

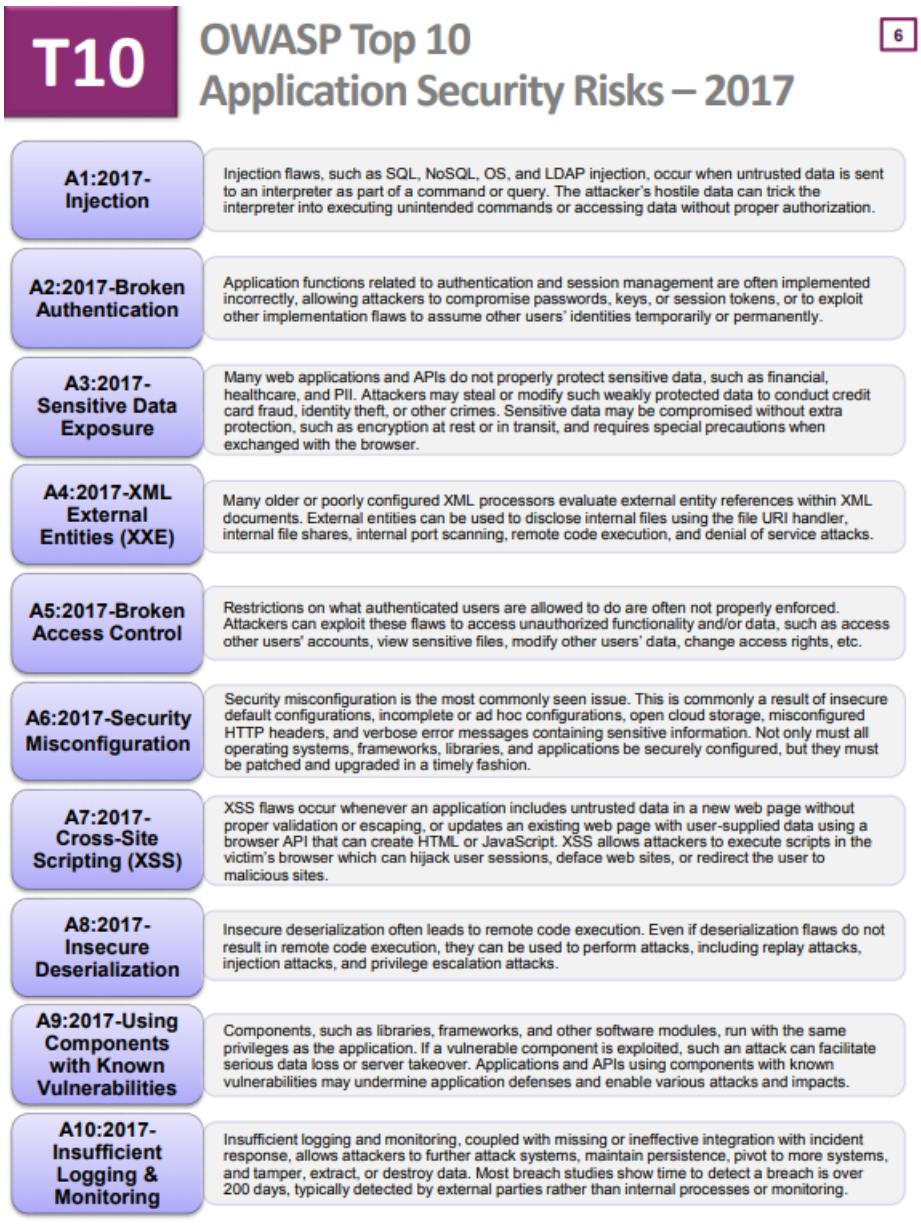


Figure 2.5: OWASP Top 10 Application Security Risks [6]

2.7 State of the art

Nowadays, and with the growth of the SDN concept, the issues that may concern the product allow the evolution of the SDN concept. Therefore, security issues are attended in a variety of different ways because every issue might have different solutions. Previous work, including [39], has detailed security challenges and threats that arise from SDN. They fundamentally highlight the importance of reaching several security goals such as confidentiality, authenticity, integrity, and availability of various entities within the SDN architecture. Though they do not propose any reliable security mechanisms needed to achieve these goals. To approach some of these security goals, the authors of [40] have proposed PermOF, a permission system for OpenFlow implementations. However, the author's focus is to refer to as OpenFlow apps, which represent applications that reside directly within the control plane. This approach is only valid while the applications remain in the controller and fail to address the security challenges that arise while being deployed external applications to the controller. In a similar approach, [41] proposes the OperationCheckpoint system, which enables the restriction of applications using the API of the open-source controller Floodlight. This approach, however, is controller-dependent and also lacks a mechanism to verify the authenticity of external applications. Additionally, SE-floodlight, which is an extension of Fort-NOX [42], also provides a controller-dependent solution and is more focused on addressing the security issues surrounding application residing within the controller. The previous approaches intend to protect the SDN controller, but what if the controller does not have any of the protection, and it is in a safe location, and the only exploitable way is through the web application. This brings a new paradigm about securing the web application instead of the controller. Nowadays, web applications like banking are relying on real-time scanning to evaluate if there are some dangerous content that might be harmful. However, the tested web application (Transcend Maestro) may be a little different because it is not available to everyone at the start and therefore does not need continuous scanning, although it is a plus. However, in case it is accessed, it makes the SDN controller vulnerable if no security is implemented. This is the purpose of this work, to understand how a real web application that is directly connected to SDN can impact the controller.

There are companies like Radware with DefenseFlow product, Fortinet, or Akamai that have solutions regarding web application security. BurpSuite or OWASP Zap have scanners to test web applications. It is quite an interest to understand

that somehow, the world of web application and SDN are connected, but when security comes abroad, both worlds appear not to be related.

3

Methodology of test environment

The last chapter ended with comments about the state of the art, and the fact that the security of web applications and SDN are somehow separated. This and the next chapter intends to bring a real, deployed SDN solution and trying to find and understand how the attacks can be deployed (discussing some tools to each attack) and how the attacks towards the web application can affect the SDN controller.

3.1 Tests Environment

The tests environment is build exclusively over real environment using five physical machines with Windows 10 and CentOS 7.6 as operating systems, with 16GB of RAM and 256GB of hard disk. There is a deployment of the oracle DB, Transcend Controller, Transcend Chorus, Transcend Maestro, Transcend MANO, and another machine with another linux distribuition named Kali where all the tests where executed. The following figure 3.1, represents the test environment where all the tests were executed.

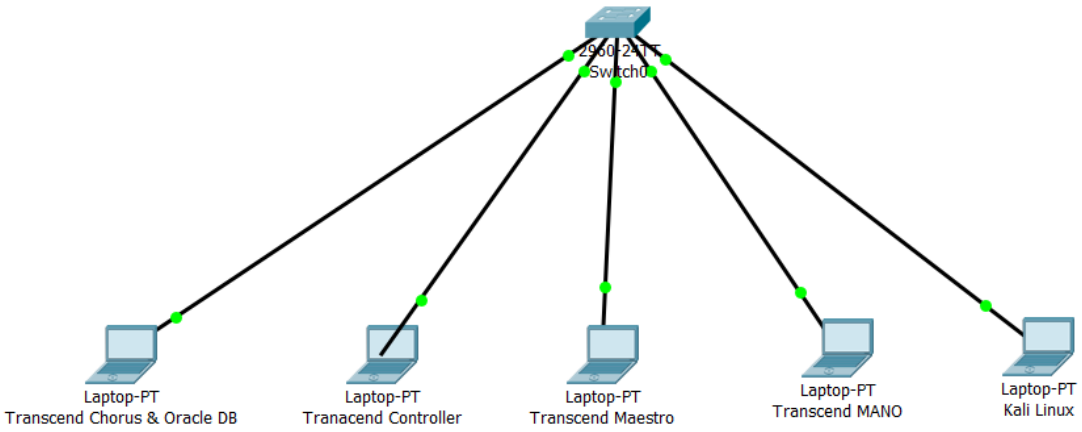


Figure 3.1: Topology of test environment

The machines possess the following dedicated hardware:

Transcend Chorus and Oracle DB

- *Memory:* 16GB;
- *Processors:* 4;
- *Operating System:* CentOS 7.6.

Transcend Controller

- *Memory:* 16GB;
- *Processors:* 4;
- *Operating System:* CentOS 7.6.

Transcend Maestro

- *Memory:* 16GB;
- *Processors:* 4;
- *Operating System:* Windows 10.

Transcend MANO

- *Memory:* 4GB;
- *Processors:* 4.

- *Operating System:* Windows 10;

Kali Linux

- *Memory:* 16GB;
- *Processors:* 4;
- *Operating System:* CentOS 7.6.

In the *Transcend Chorus and Oracle DB* machine is where the management of the network happens, and it is this machine that manages the NEs and all the network information. It is also responsible for informing the Transcend Controller of all reserved resources. In this machine was installed a local database from Oracle with the version 18c, this was the latest version when were performed the tests.

In the *Transcend Controller* machine is where the control of the network happens and where services created in the network, this machine communicates with Transcend Chorus and with Transcend Maestro.

The *Transcend Maestro* machine is where the orchestrator can execute the order to create services. It is also possible to also have some control over the controller, see the topology of the network, endpoints, notifications, or logs. Therefore it is connected to the northbound interface of the Transcend Controller.

Inside the *Transcend MANO*, is the client that connects to the Transcend Chorus server. It also allows the orchestrator to manage the network by choosing which NEs it is going to connect, manage thus NEs, and decide how is the network configured.

The last but not least, the *Kali Linux* is where all the tests where executed and obtained all security test results performed on this setup.

For reasons of development with python, it was installed the Pycharm: The Python Integrated Development Environment (IDE), where it was developed some automatic tests, that will be described in the next sub-chapter.

3.2 OEM controller components, functions and links

The OEM controller, named by Transcend Controller, is composed of several components that need to interwork with each other to make the communications between the SP controller and the NEs possible. The communication established to

the controller is through the use of an HTTP API. This API adapts to the type of operations that executes in the optical networks. In SDN, there are several standard APIs that can be used to make the communication between the controller, in the control plane layer, and the NEs in the data plane layer. The most commonly used is the OpenFlow [43]. This standard API enables the controller to perform a subset of network management operations to the entries at the flow tables (as such, it is adapted to use with network switches). Optical networks support other types of network configurations, for instance, the creation and deletion of cross-connections that are not supported by OpenFlow. For this purpose, is used another protocol, Netconf [44], which is a complement to Yang [45], a data modeling language applied to form configuration and state data manipulated by NETCONF. On top of this protocol, the RESTCONF [46] protocol was developed to provide an abstraction layer over HTTP for accessing data defined in YANG, using the datastores defined in NETCONF. This protocol can be used to make configurations in the optical networks.

The Infinera OEM controller is composed internally by a Connection Provisioning Manager (CPM), Service Level Agreement Manager (SLAM), Path Computation Element (PCE), Network Domain Manager(NDM), Architecture Facility Manager (AFM), Left-Wing and TOPO+ (Topology reader). NDM is one of the essential components of our project. The reason is that it will be accessed by the Service Provider Administration Portal (SPAP) to view and make configurations in the network domain, using the Open Interface Forum (OIF) rest API.

The previous components have the following functions: the CPM is responsible for creating, deleting, and rerouting connections requests sent by SLAM. It's responsible for performing all necessary configurations (e.g. PortMode, FEC, Frequency, ResourceUsage, SupervisionMode, Cross-connection creation, etc.); the SLAM is responsible for service connection management; the PCE is responsible for computing a route using the paths available; the NDM implements the northbound API standard to expose the network-related data, where the Transcend Maestro (Multi-Domain Orchestrator) is connected; The main focus of AFM is to report network topology and multiplexing capabilities. Emphasis is on capabilities; Left-Wing is responsible for connecting to the Transcend Chorus (Network Management and Operation) as works as a proxy. Finally, the TOPO+ gives the topology of the optical network that will be used by the PCE to compute new routes. All these components are shown in figure 3.2.

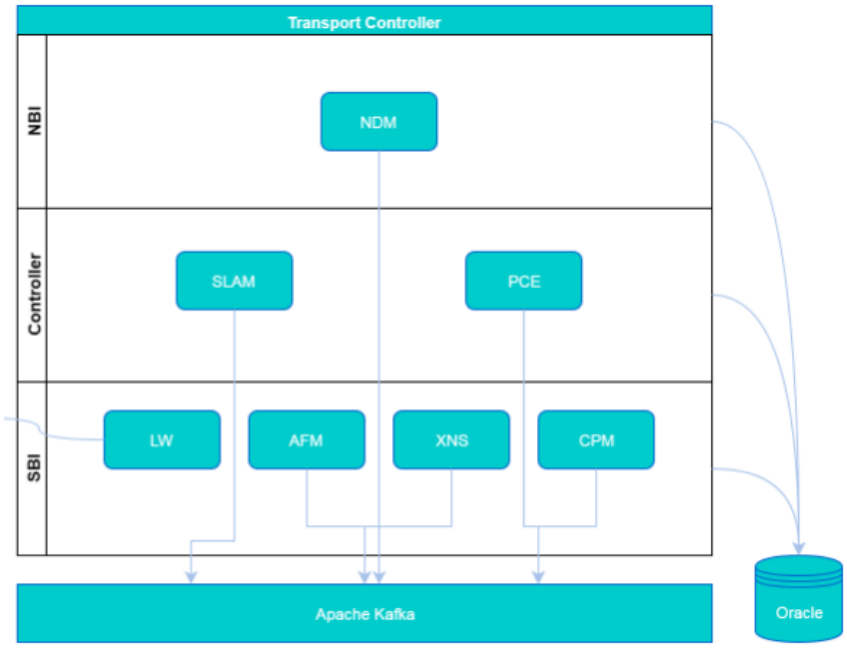


Figure 3.2: Infinera SDN Architecture

Typically, the workflow executed is the following:

- Modeling - defines the network model that is created directly from the network to the OIF to be used by the SP;
- Partitioning - the partitioning associates the resources with a specific network domain;
- Virtualization - the network model, created by the system administrator is used to associate the network to the Network Domain user (in this case the system administrator control what users see).

SDN is connected through Left-Wing to Transcend Chorus [1], this is designed to automate end-to-end network actions, including service activation testing leveraging network element features such as Pseudorandom Binary Sequence (PRBS) test and loopback that emulates white noise to perfection, RFC 2544 - Benchmarking Methodology for Network Interconnect Devices, Y.1564 - Ethernet service activation test methodology, and Two-Way Active Measurement Protocol (TWAMP). In extension to discovering network inventory, links, and physical topology, Chorus offers both service and network element error management,

including alarm management, event correlation, and root cause analysis. In addition to real-time and historical performance monitoring with visualization capabilities, including network heat maps, it offers high-level performance monitoring abilities such as Optical time-domain reflectometer (OTDR) and optical power monitoring. Chorus also automates platform management functions, including commissioning, software upgrades, configuration backup, and user/security management.

The northbound interface of SDN is where the Transcend Maestro is connected [1]. Transcend Maestro is designed to allow automation and end-to-end services to spread technology layers from Layer 0 through Layer 3 and to span various domains with support for third-party SDN controllers via open standards-based interfaces. Maestro can minimize cost and latency by offloading traffic to lower layers where feasible with multilayer visualization, significantly supporting users' ability to recognize traffic flows across multiple layers. Maestro cost-effectively maximizes service availability with disjoint routes, efficient multilayer protection, and restoration, and by assuring end-to-end SLA compliance.

In this software suite, there is also the Transcend MANO, responsible for the NFV Management and Orchestration.

An essential mechanism to the SP is the division of the network in separate domains. For SP, it helps in maintaining the system secure. When the network is set up by the administrators, they can use one domain with no restrictions, and then after the network provisioning, they can create other domains to different operators with specific restrictions. In this case, the operators will have access only to the resources that they need to manage their optical services.

3.3 Selection of penetration tests methodologies

As it was discussed in the last chapter about the state of the art, we realize that everything created and developed was dependent and deployed inside the controller. However, instead of challenging the controller directly, challenge the applications that are connected to it via the northbound interface using an HTTP connection, like the Transcend Maestro uses to connect the Transcend Controller. Another point that distinguishes this from any other work stands in the fact that these tests were performed in a real and deployed solution that real clients rely on and depend on. Therefore, and because security issues regarding web application with the SDN controller might be left aside, it is important to validate

how vulnerable this solution is and understand what remediation methods can be applied to fix the vulnerabilities that may exist. To perform thus tests, this work intends to understand which vulnerabilities can be found by performing tests from the inside and performing tests from the outside. The last one is the one that enables an attacker to break in the application and harm the controller. There are two types of test methodologies used to fulfill these requirements of inside and outside testing and thus are white-box and black-box penetration testing, respectively.

3.4 White box

White-box penetration test determines the vulnerabilities internally the system that might be exploitable. White Box Testing is defined as the testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers.

White box testing involves the testing of the software code for the following.

- Internal security holes;
- Broken or poorly structured paths in the coding processes;
- The flow of specific inputs through the code;
- Expected output;
- The functionality of conditional loops;
- Testing of each statement, object, and function on an individual basis;
- The testing can be done at system, integration and unit levels of software development.

One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

In order to perform white-box tests two premisses that must be understood:

1) Understand the source code:

- The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

2) Create test cases and execute them:

- The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include Manual Testing, trial, and error testing and the use of testing tools as we will explain further on in this article.

3.4.1 White Box Testing Techniques

A major White box testing technique is Code Coverage analysis. Code Coverage analysis eliminates gaps in a Test Case suite. It identifies areas of a program that are not exercised by a set of test cases. Once gaps are identified, you create test cases to verify untested parts of the code, thereby increasing the quality of the software product

There are automated tools available to perform Code coverage analysis. Below are a few coverage analysis techniques

- **Statement Coverage -** This technique requires every possible statement in the code to be tested at least once during the testing process of software engineering.
- **Branch Coverage -** This technique checks every possible path (if-else and other conditional loops) of a software application.

3.4.2 White Box Testing Tools

There is a various number of white-box testing tool, starting with Veracode, OWASP Zed Attack Proxy (ZAP), Burp Suite and more. Thus softwares allow the scanning of the web application, use their tools to exploit the vulnerabilities and also have the results in a report format. They are applications that serve as proxy between the browser and the web app and for that reason all packets are still unincrypted which allows to use tem and manipulate them in order to exploit a vulnerability. In case of Veracode and Burp Suite are both paid softwares although only Burp Suite has a free software that allows the use of all manual tools while the Veracode is full paid. The results of the sacns between them are very similiar acoordingly to IT Central Station [47] and TrustRadius [48]. On the other hand, OWASP Zed Attack Proxy (ZAP) is one of the world's most popular free security tools and is actively maintained by hundreds of international volunteers. It can automatically find security vulnerabilities in web applications while the developing and testing of the applications are on going. After analysing this tools the selection went to Burp Suite, because it has a free software, a variety of tools and it was relativelly easy to get a trial licence for one of the paid versions. Burps Suite works as a proxy, located between the web application and the browser, and all the data passes through this software where is analyze, shown the results and after that proxy the packets to the destination. Burp Suite is software from PortSwigger [49] and is the leading software for web security testing, because of thousands of organizations like Google, Amazon, AT&T, Verizon, Facebook, Microsoft, Oracle, and others, use Burp Suite to find security exposures before it is too late. By using cutting-edge scanning technology, it identifies the very latest vulnerabilities. Burp Suite has three editions, the Community, the Professional, and the Enterprise. The differences between them are that the Community edition is free, while the other two are paid. It was possible to evaluate how Burp Suite can be useful because this version otherwise of the community version has a web vulnerability scanner that allows a live crawler and live packet analyzer.

Burp Suite has tools to use in particular tasks such as:

- Scanner - This is used to scan websites content and security vulnerabilities automatically;
- Intruder - This allows to perform customized automated attacks, to carry out all sorts of testing assignments;

- Repeater - This is used to modify and reissue individual HTTP requests over and over manually;
- Collaborator client - This is used to create Burp Collaborator payloads and control for resulting out-of-band interplays;
- Clickbandit - This is used to create clickjacking exploits toward vulnerable applications;
- Sequencer - This is used to analyze the quality of randomness in an application's session tokens;
- Decoder - This allows to modify bits of application data using standard encoding and decoding schemes;
- Comparer - This is used to achieve a visual comparison of bits of application data to discover intriguing variations.

The following figure 3.3 shows Burp Suite dashboard.

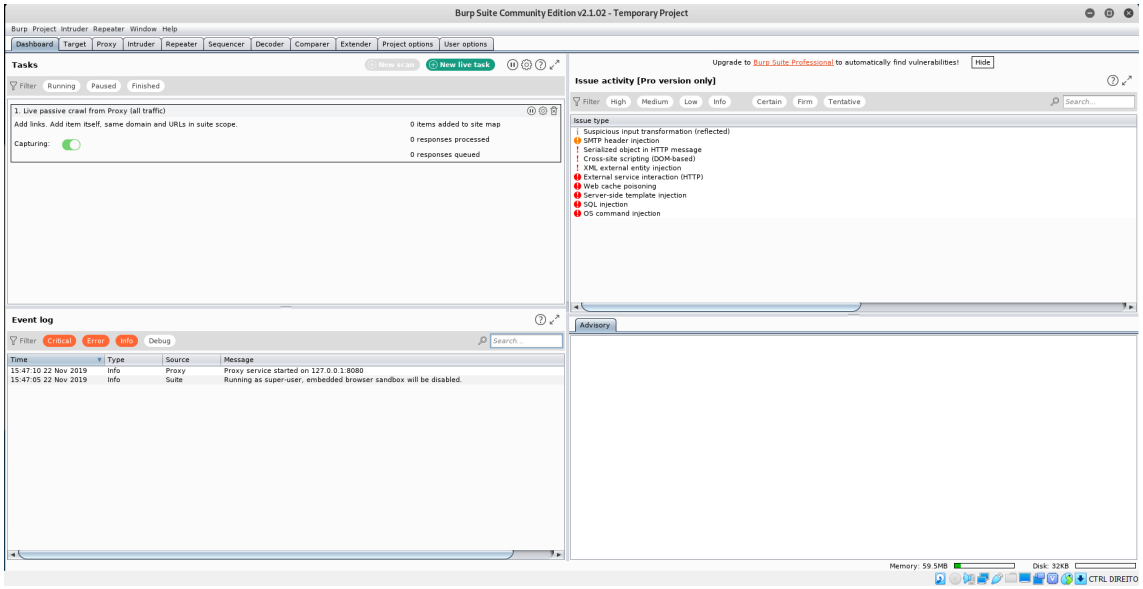


Figure 3.3: Burp Suite dashboard

Burp Suite scanner tool was used to scan web sites. Burp Suite scans can be launched in different ways, through the scan from specific URLs, scan only specific items, or live scanning. The tests regarding this tool the live scanning were the performed scan because it had the crawl and audit option in an automated way that allows the mapping of the website and the check for vulnerabilities. This

tool also allows the generation of reports of the issues found via Burp Scanner in HTML format or XML format.

3.5 Black box

A black-box penetration test determines the vulnerabilities in a system that are exploitable from outside the network. Black-box penetration testing relies on dynamic analysis of currently running programs and policies within the target network. A black-box penetration tester must be familiar with automated scanning tools and methodologies for manual penetration testing. Black-box penetration testers also need to be capable of creating their map of a target network based on their observations since it is not provided such a diagram. In terms of the working path, there will be used manual tools to try and map the site of the web application; after that, it will be employed some techniques such as SQL Injection, and Cross-Site Scripting (XSS) to find vulnerabilities in the web application.

3.5.1 Gathering information tools

Gathering information is the first and most crucial step of a black-box penetration test because it is at this point that the attacker learns about the network itself, which machines are connected, and what is their Internet Protocol (IP) and Media Access Control (MAC) address. There are a variety of tools that allow this gathering of this information like netdiscover, Nmap, Zenmap, hping3. However, to start the process of information gathering was used as a tool named netdiscover.

3.5.1.1 Network

Netdiscover is a simple Address Resolution Protocol (ARP) scanner that can be used to scan for live hosts in a network. It can scan for multiple subnets also. It merely produces the output in a live display(ncurse). This can be used in the first phases of a pentest where you have access to a network. Netdiscover is an initial-recon and straightforward tool which can be very handy. It has features like:

- Simple Arp Scanner;
- Works in both Active and Passive modes;

- Produces a live display of identified hosts;
- Able to scan multiple subnets;
- Timing Options.

The first step was to understand which machines were connected to the network and for that it was used the feature simple ARP scanner, where it sends to the network ARP requests with one IP and the receiver that own that IP answers to the request with a message of ARP response containing the information of his MAC, afterward if created a matrix with IPs and MAC. Since this tool uses ARP reconnaissance to get information about one machine, it is necessary to understand how the arguments can be used to take the most information possible. This tools as some options that can be used, and are listed in figure 3.4.

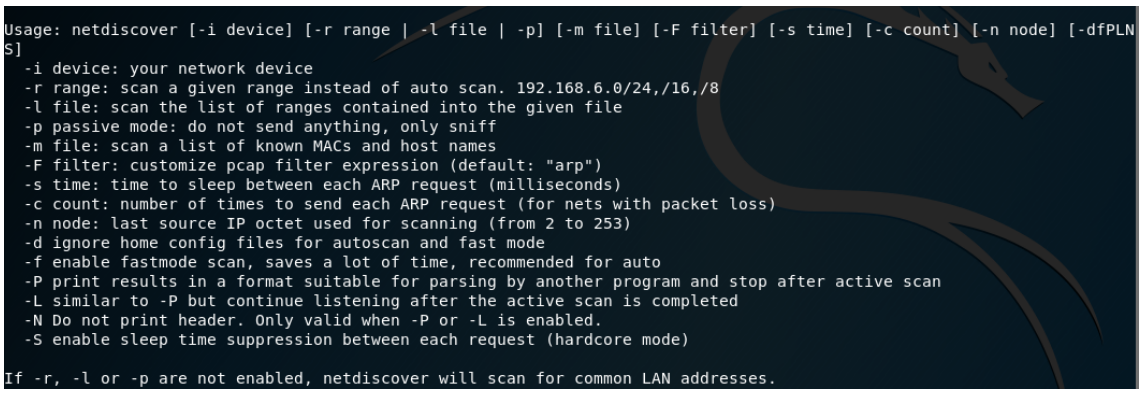


Figure 3.4: Netdiscover options

It is possible to see that one of the available options is the range, which means that passing as argument the range of IPs that we want to check. The tool sends to each IP an ARP request and gets as response an ARP response, to use it, the attacker must be connected to the same network, which means that the attacker has an IP that matches the network. An IP is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol communication. In every network, there are two addresses used for network and broadcast addressing. These addresses are not accessible for link addressing. So if there are four addresses in the network, there can be two links defined. The only exception is the network with a 32-bit mask (network with only one address), which is used for determining the link's local IP address in the routing table. It signifies that masks with 31 bits and a network with two addresses can not be used.

A network address is an original address in the network, and it is used for the association network division. All the IP addresses, using the identical network

address segment, are in the same network segment. Because the network address is the first address in the network, it can not be a random IP address, but it must match with the network mask in a binary representation, for lower bits in the network address must be zeros, as long as the mask has zeros.

A broadcast address is the highest address in the network, and it is handled for addressing all the links in the network at the same time. It means that the IP packet, where the destination address is broadcast address, is sent to all links of the IP network. It is essential for remote announcements in the network section. In some instances, it is used for attacking targets by hackers or can cause problems in more extended network segments. As using this tool, the attacker must be connected to the same network; therefore, the IP in its machine is just one more that is encapsulated in the mask. This means that the attacker knows all the range of IPs of that network; however, he does not know which ones are already given.

3.5.1.2 Web applications

For gathering information about web applications is essential to understand that a web application consists of application code written to perform some action(s) on a web server and display the result to a web browser client. A web server is a software operating on a computer, to which web browsers connect using HTTP/HTTPS to run web applications. There are different deployments of web applications. There are Static Web Applications and include professional portfolios or digital resumes; in other words, static content. Dynamic Web Applications are much more involved on a technological level. They use databases for data storing, and its contents are renewed each time the user accesses them. E-Commerce is the type of application development process that is more complex because it must support electronic payments that can be executed from credit cards, PayPal, or other payment methods. Portal Web App is referring to a type of application that accesses the various sections or categories through a home page. Animated Web Applications are inevitably associated with Flash technology. This programming approach allows displaying content with animated effects. The last one, Web Application with a content management system, is widespread among the content pages: personal blogs, corporate blogs, professional blogs, news pages, articles, or media. After analyzing the number of different types of web applications is possible to distinguish which parameters are familiar with each type of web application, and this is security because the connections between the web application and the webserver are the ones that are more exposed

to threads. Therefore it is crucial to test, analyze, and fix the problems that might create vulnerabilities.

There are many tools to perform security tests on web applications, but first is essential to understand which type of web application is and, more importantly, understand if there is one in the network. There are tools like NMAP and Zenmap that have the utility to network identification and security auditing. Multiple systems and network administrators additionally find it beneficial for tasks such as network inventory, managing service, upgrade schedules, and monitoring host or service uptime. Nmap uses raw IP packets in various ways to determine what hosts are accessible on the network, what services (application name and version) those hosts are allowing, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and multiple of other things. It was designed to scan large networks fast, although it works fine upon single hosts. Nmap runs on all main computer operating systems, and official binary packages are accessible for Linux, Windows, and Mac OS X. In extension to the traditional command-line, Nmap incorporates an advanced GUI and results in a viewer (Zenmap). The following figure 3.5 shows the Zenmap GUI.

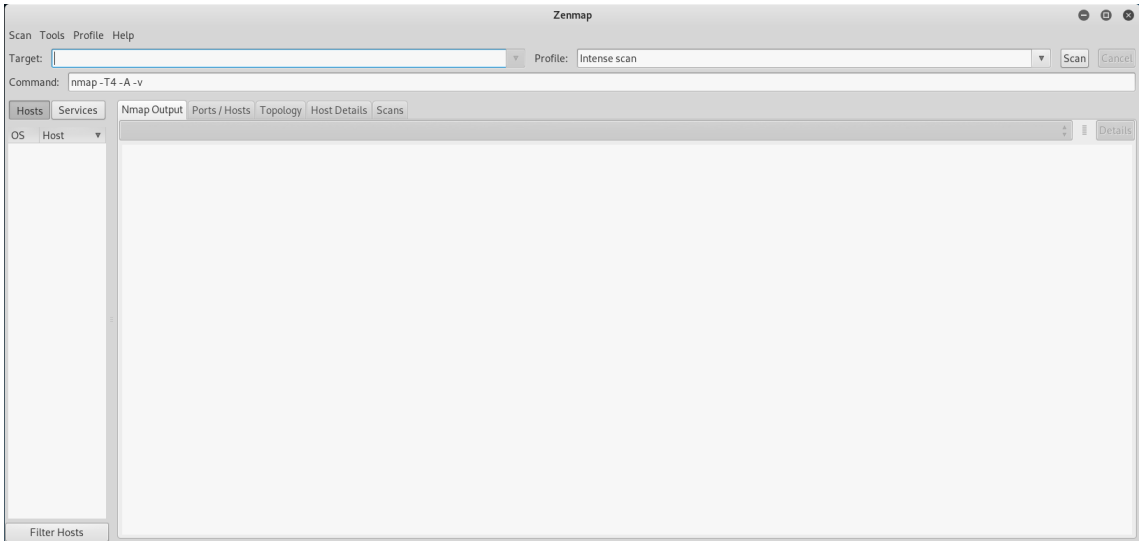


Figure 3.5: Zenmap GUI

It also as a resilient data transfer, redirection, and debugging tool (Ncat), a utility for analyzing scans results (Ndiff), and a packet generation and response analysis tool (Nping).

Now that the attacker knows which machines are connected in the network is

essential to understand which ports are open and vulnerable to attacks. To implement automation to the information gathering was created a phyton script that consists of using Nmap to get the information about all open ports of all the targets previously discovered and understand which open ports have HyperText Markup Language (HTML) information.

Another purpose of this script is to understand which opened ports have web servers and for that reason another tool was used, named Nikto. Nikto is an Open Source (GPL) web server scanner which performs comprehensive tests against web servers for multiple items. It has multiple options, like shown in the figure 3.6.

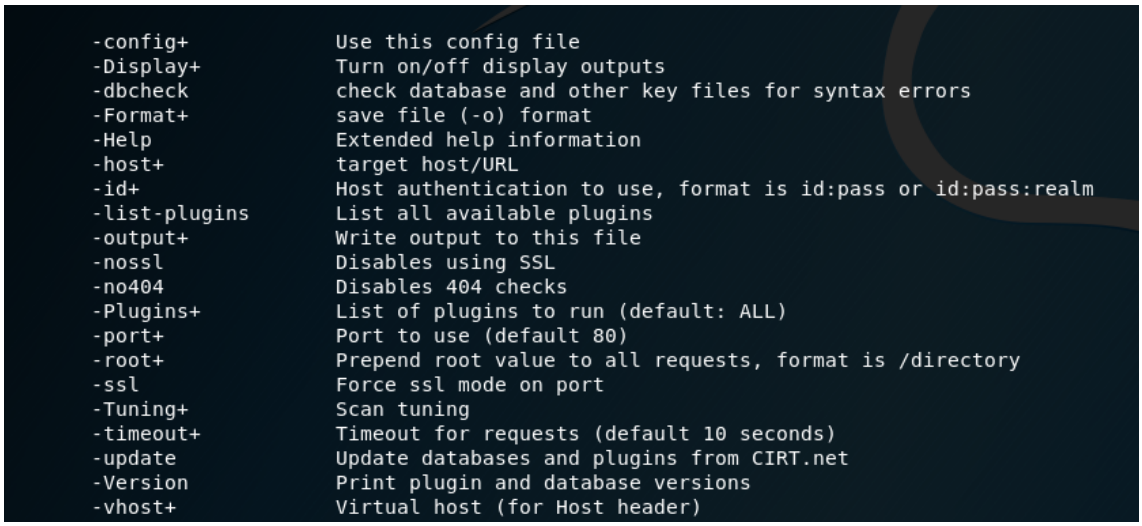


Figure 3.6: Nikto options

It also checks for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software. Scan items and plugins are frequently updated and can be automatically updated.

3.5.2 Dictionary attack

After doing the information gathering and understanding, who is the target, it is possible to start to map the web application to understand what type of information can be useful to exploit the controller. For that information gathering, there are several tools like Recon-ng or Skipfish. Recon-ng is a fully modular framework and makes it simple for even the latest Python developers to participate. Each module is a subclass of the “module” class. The “module” type is a

customized "cmd" speaker equipped with built-in functionality that implements simple interfaces to daily tasks such as normalizing output, communicating with the database, creating web requests, and managing API keys. Consequently, all the hard work has been done. Building modules is simple and necessitates little more than a few minutes. Recon-ng is a full-featured Web Reconnaissance framework written in Python. Finished with sovereign modules, database synergy, interactive help, and command completion – Recon-ng provides a robust context in which open-source web-based reconnaissance can be handled promptly and thoroughly. Recon-ng has a look and feels even has a similar flow to the Metasploit Framework, decreasing the knowledge curve for leveraging the framework. It is, of course, considerably different though, recon-ng is not designed to face with existing frameworks, as it is meant exclusively for web-based open source reconnaissance. This tool works as a passive web application reconnaissance tool. Using Skipfish, it was possible to get information about the full URL to access the page of the Transcend Maestro. Skipfish [50] is a web application reconnaissance tool developed by Google. It provides an interactive sitemap for the targeted site by taking out a recursive crawl and dictionary-based probes. The resulting map is then annotated with the output from various active (but hopefully non-disruptive) security checks. The final statement produced by the tool is meant to serve as a foundation for professional web application security assessments. Skipfish carries some key features like:

- **High speed:** pure C code, highly optimized HTTP approach, minimal CPU footprint – easily reaching 2000 requests per second with responsive targets.
- **High speed:** heuristics to maintain a variety of quirky web frameworks and mixed-technology sites, with automatic knowledge abilities, on-the-fly wordlist making, and form autocompletion.
- **Cutting-edge security logic:** great quality, low false positive, differential security checks, proficient of detecting a spectrum of detailed flaws, including blind injection vectors.

The resulting map is then annotated with the output from some active (but hopefully non-disruptive) security checks. The last report produced by the tool is intended to serve as a foundation for expert web application security assessments. Skipfish works as an active web application reconnaissance tool, and this is what differentiates Skipfish from Recon-ng. For this use case, an active reconnaissance tool was more suitable because active reconnaissance is the process of examining

a computer system to scope technical weaknesses that can be used to access it. System information is used to gain unauthorized access to protected materials, infiltrating any firewalls or routers. The hacker then actively drafts the network infrastructure, using tools such as NSLookup to recognize hosts. Once they have been discovered, a port scan is handled to reveal any possible vulnerabilities. Security restrictions have therefore been employed, with information resistance found sent back to the hacker, while, on the other hand, performing passive reconnaissance doesn't require contact with any infrastructure, allowing hackers to bypass potential obstacles. The reconnaissance defines the target company, partner and employee details, technology in use, IP information, and so on, then retreats with information collated. For all these reasons, the skipfish tool was the one selected to use.

3.5.2.1 Automated script

Via script, the implementation consisted of using a web driver named selenium [51]; this is a web driver and is compatible with a different browser; in this case, it was used Chrome browser. The following figure 3.7 shows de webdriver configuration.

```
login_url = "https://10.0.0.4:9443/transcend-portal/#/login"

new_options = webdriver.ChromeOptions()
new_options.add_argument('--igonre-certificate-errors')

browser = webdriver.Chrome("/root/Downloads/chromedriver", options=new_options)

browser.maximize_window()

browser.get(login_url)
```

Figure 3.7: Dictionay python script attack (Selenium webdriver configuration excerpt implementation)

To configure the web driver for correct use is firstly necessary to indicate which type of browser it is going to be used and particularly for the Chrome browsers; the location of the chrome driver must be indicated. In this case, the default options were overwritten because it was necessary to ignore SSL certificate errors because Google does not know the certification authority of the certificate used by Maestro because it was created internally by Infinera.

Selenium also allows to use the id of the text areas to replace thous empty spaces with the user and password, and the last step is to use selenium to click the login button to validate the login request in the system. As it is shown in the figure 3.8. These IDs names were obtained after inspecting the elements of the Maestro login page.

```
user = browser.find_element_by_xpath("//input[@id = 'mat-input-0']")
user.send_keys(username)

key = browser.find_element_by_xpath("//input[@id = 'mat-input-1']")
key.send_keys(password)

browser.find_element_by_xpath("//button[@name = 'login-submit']").click()
```

Figure 3.8: Dictionary python script attack(IDs of login form excerpt implementation)

After selenium finds the elements, the app must send the desired values to fill thus areas, and that is why the function *send_keys* is used. At last, the credentials used are cleared from the text areas to use the next username and password combination.

3.5.2.2 Burp Suite

The community edition was used to perform, exclusively, manual tests. Like it was presented, Burp Suite works a little bit different because Burp Suite works as a proxy; therefore, it must be configured the proxy with IP 127.0.0.1 and port 8080. The figure 3.9 shows that configuration.

Authentication is the center of an application's security upon unauthorized entrance. If an attacker can break an application's authentication function, then they may be able to own the entire application. In the browser login page, it was entered some arbitrary details into the login page and submit the request. After Burp Suite receives the request, it then sent to the "Intruder" tool of Burp Suite. Burp Intruder works by using an HTTP request (named the "base request"), altering the request in several systematic methods, assigning each modified version of the request, and investigating the application's responses to identify unknown features.

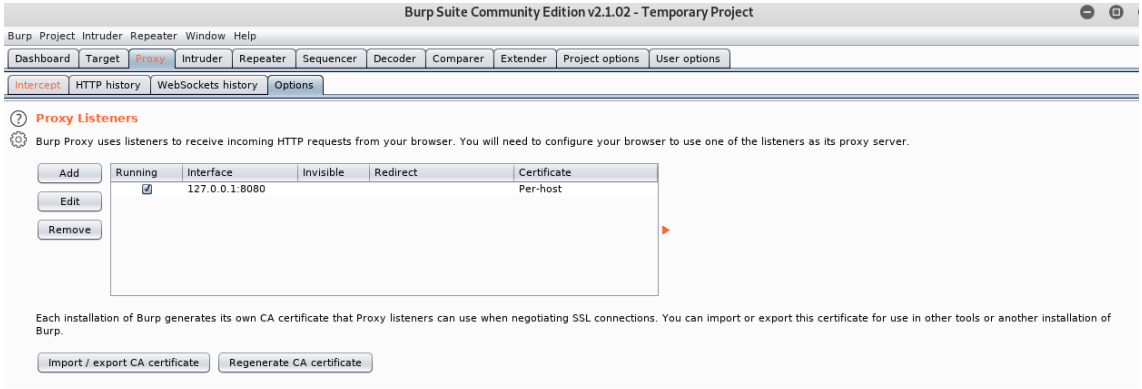


Figure 3.9: BurpSuite proxy configuration

Buro Suite pre-set some payload positions, however thus positions must be selected only for the login credentials, username, and password. Add the "username" and "password" parameter values as positions by highlighting them. There are four different types of attacks, and for each attack, it must be specified one or more positions of payloads and the sets in the base request anywhere the payloads are to be set. The following figure 3.10 shows the the Burp Suite intruder acttacks page.

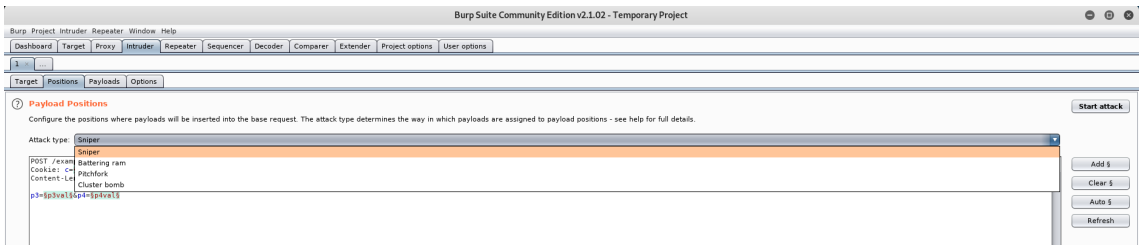


Figure 3.10: BurpSuite Intruder attacks

Various techniques of producing payloads are available (including simple lists of strings, dates, numbers, bit flipping, brute force, and many others). Payloads can be put into set positions using distinctive algorithms. Various tools are available to help analyze the results and identify unusual items for further investigation. Thus attacks are:

- Sniper - This uses a unique set of payloads. It targets a specific payload position and points each payload into that position. This attack is beneficial for fuzzing an amount of requests parameters separately for general vulnerabilities.

- Battering ram - This uses a unique set of payloads. It recursively goes through the payloads and distributes the same payload into all of the defined payload positions at once. This attack type is beneficial when an attack needs the same input to be entered in various positions within the request (e.g., a username within a Cookie and a body parameter).

- Pitchfork - This uses various payload sets. There is a distinct payload set for each defined position (up to a maximum of 20). The attack recursively goes through all payloads concurrently and place one payload into each defined position. In other words, the first request will place the first payload from payload set 1 into position 1, and the first payload from payload set 2 into position 2; the second request will place the second payload from payload set 1 into position 1 and the second payload from payload set 2 into position 2, and move on. This attack type is useful where an attack requires different but related input to be inserted in various places within the request (e.g., a username in one parameter, and a known ID number corresponding to that username in another parameter).

- Cluster bomb - This uses various payload sets. There is a different payload set for each set position (up to a maximum of 20). The attack iterates through each payload set in turn so that all permutations of payload combinations are tested. I.e., if there are two payload positions, the attack will place the first payload from payload set 2 into position 2, and iterate through all the payloads in payload set 1 in position 1; it will then place the second payload from payload set 2 into position 2, and iterate through all the payloads in payload set 1 in position 1. This attack type is useful where an attack requires different and unrelated or unknown input to be inserted in multiple places within the request (e.g., when guessing credentials, a username in one parameter, and a password in another parameter).

In the "Payload options," enter some possible passwords, this can be done manually or using a custom pre-set list. At last, perform the attacks pressing the "Start attack" button.

3.5.3 SQL Injection

SQL injection [52] arises when user-controllable data is incorporated into database SQL queries in an unsafe manner. An attacker can supply crafted input to break

out of the data context in which appears and interfere with the structure of the surrounding query. A wide range of damaging attacks can often be delivered via SQL injection, including reading or modifying critical application data, interfering with application logic, escalating privileges within the database, and taking control of the database server. There are two ways to explore vulnerabilities through web applications, by attacking the database directly or exploiting URL requests that use queries in their URL in order to query the database and trick it into giving sensitive information about the users and their passwords.

3.5.3.1 Burp Suite

Burp Suite is a software with various tools that can be used for different tests; in this case, it can be used to test a web application against SQL Injection attacks. In these tests, it was also used the Burp Suite community edition only to use the manual tools available to test this vulnerability. For that, it was also used Burp Intruder and will be explained in more detail in the next chapter when the results of using this tool be discussed. However, this vulnerability can also be tested with the Scanner tool available in Burp Suite Professional version. Burp Scanner is a tool for implementing automated vulnerability scans of web applications. Burp Scanner can be used alongside the methodology of the manual tools to promptly recognize various types of well-known vulnerabilities, keeping the focus on issues that require human intelligence and ingenuity to discover. With most web scanners, a URL is provided for the application, and then follow a progression bar update until the scan is terminated and a report is provided. This scanning model has meaningful restrictions, which lead to incomplete coverage, missed vulnerabilities, and misdirected effort. (The most critical problems are: crawler limitations due to changing client-side technologies, inability to interoperate with the complex stateful nature of today's applications, failure to supply suitable input to complete multi-stage processes, problems working with authentication and session handling mechanisms, and many others.). Burp's favored method for scanning applies a different, user-driven model; this provides fine-grained authority over an individual request that gets scanned and immediate feedback about the results. This method helps to circumvent many of the technical difficulties encountered by conventional scanners. The scanner can be conducted using the browser to ensure that no critical areas of functionality are avoided. With full control over what gets scanned, dangerous functionality can be avoided, acknowledge replicated functionality, and pass through any input validation conditions that a fully automated scanner might struggle. Moreover, because it is given

direct feedback about the scanner's activity, it is guaranteed that difficulties with authentication and session handling are bypassed and that issues originated by multi-stage processes and stateful functions are properly handled. By using a scanner in this way, it is possible to cover an essential range of vulnerabilities whose detection can be automated.

- **Passive Scanning Mode:** Burp Scanner can operate in a purely passive mode. Here, the Scanner doesn't send any new requests of its own. It merely analyzes the contents of existing requests and responses, and deduces vulnerabilities from those. Many types of security vulnerabilities can be detected using only passive techniques. By default, Burp carries out passive scanning of all traffic passing through Burp Proxy. After you have configured your target scope, you might want to reconfigure the live passive scanning settings, so that only in-scope items are passively scanned. This will prevent the Results tab from accumulating passive scan issues for targets you are not interested in.
- **Active Scanning Mode:** In the active scanning mode, Burp sends various crafted requests to the application, and analyzes the resulting responses looking for evidence of vulnerabilities. Active Scanning is capable of identifying a much wider range of vulnerabilities, and is essential when performing a comprehensive test of an application.

3.5.3.2 SQLMap

Another tool, name SQLMap [53], is used to test SQL injection vulnerabilities. SQLMap is an open-source penetration testing tool that automates the method of identifying and exploiting SQL injection defects and carrying over of database servers. It comes with a detection engine, several features for the penetration tester, and a wide variety of switches serving from database, over data fetching from the database to entering the underlying file system, and performing commands on the operating system via out-of-band connections. The main options of this tool are presented in figure 3.11.

It has full support for MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, Informix, HSQLDB and H2 database management systems. What concerns this work the support of the Oracle database is imperative, and this tool checks the requirement. It also supports other features like:

Figure 3.11: SQLmap options

- Full support for six SQL injection techniques: boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries, and out-of-band.

- support to quickly connect to the database without crossing a SQL injection by giving DBMS credentials, IP address, port, and database name.

- It supports the enumeration of users, password hashes, privileges, roles,

databases, tables, and columns.

- Automatic recognition of password hash formats and support for cracking them using a dictionary-based attack.
- It enables to dump database tables completely, a range of entries, or specific columns as per user's choice.
- support to search for specific database names, specific tables across all databases, or specific columns across all databases tables.

3.5.4 Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) attacks are a sort of injection, in which malicious scripts are introduced into differently harmless and trusted websites. XSS attacks occur while an attacker uses a web application to send harmful code, usually in the form of a browser script, to another end-user. Defects that concede these attacks to work are quite famous and happen anywhere a web application accepts input from a user within the output it produces without verifying or encoding it. There are three types of XSS attacks [54], the Stored XSS (AKA Persistent or Type I, generally occurs when user input is stored on the target server, such as in a database, in a message forum, visitor log, comment field, etc. And then, a victim can retrieve the stored data from the web application without that data being made safe to render in the browser. With the advent of HTML5 and other browser technologies, we can envision the attack payload being permanently stored in the victim's browser, such as an HTML5 database, and never being sent to the server at all. Reflected XSS (AKA Non-Persistent or Type II) occurs when user input is immediately returned by a web application in an error message, search result, or any other response that includes some or all of the feedback contributed by the user as part of the request, without that data being delivered safely to render in the browser, and without enduringly storing the user-provided data. In some events, the user-provided data may nevermore leave the browser. DOM Based XSS (AKA Type-0) is a form of XSS where the entire tainted data flow from source to sink takes place in the browser, i.e., the source of the data is in the DOM, the sink is also in the DOM, and the data flow never leaves the browser. For example, the source (where is read malicious data) could be the URL of the page (e.g., document.location.href), or it could be an element of the HTML, and the sink is a sensitive method call that causes the execution of the malicious data (e.g., document.write).

3.5.4.1 Cross Site "Scripter"

The tests of this vulnerability were made using the tool Cross Site "Scripter" [7]. Cross Site "Scripter" (XSSer) is an automatic framework to detect, exploit and report XSS vulnerabilities in web-based applications and uses a URL/Hash Generation Schema, like the one in figure 3.12

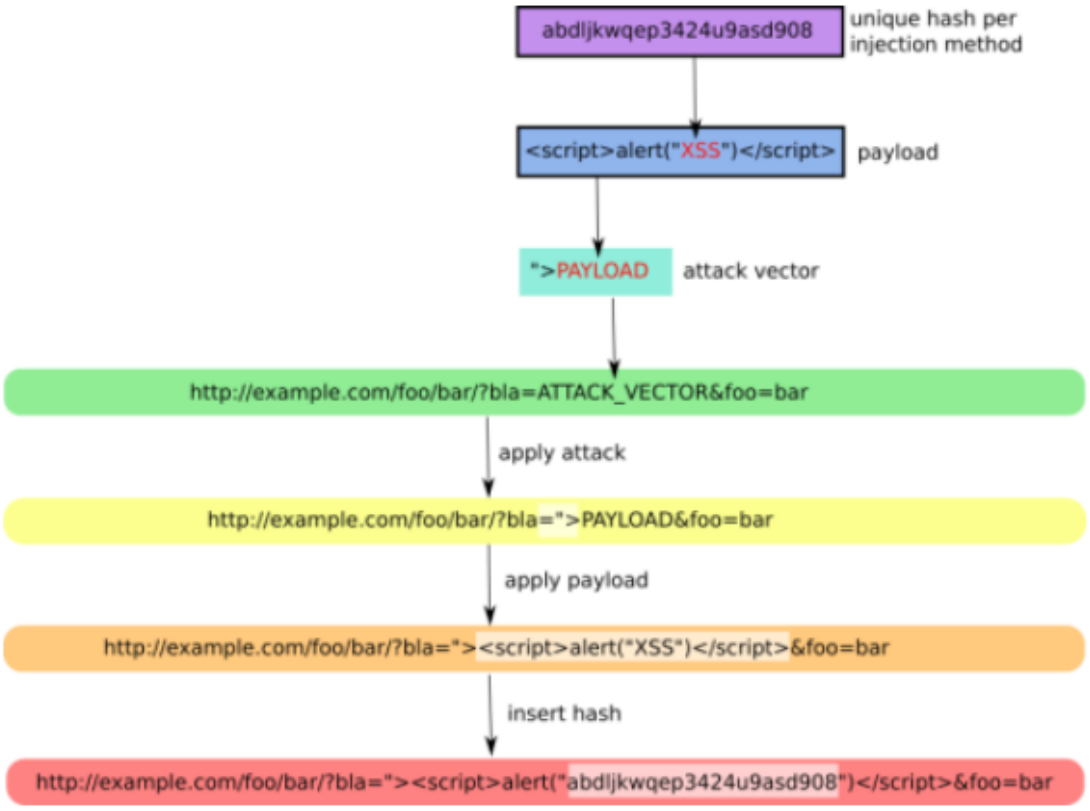


Figure 3.12: Working schema of the tool Cross Site "Scripter" [7]

This tool provides several options to try to bypass certain filters and various special techniques for code injection as it is shown in the figure 3.13

```
Usage:

xsser [OPTIONS] [--all <url> |-u <url> |-i <file> |-d <dork> (options)|-l ] [-g <get> |-p <post> |-c <crawl> (options)]
[Request(s)] [Checker(s)] [Vector(s)] [Anti-antiXSS/IDS] [Bypasser(s)] [Technique(s)] [Final Injection(s)] [Reporting]
iscellaneous}

Cross Site "Scripter" is an automatic -framework- to detect, exploit and
report XSS vulnerabilities in web-based applications.

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -s, --statistics      show advanced statistics output results
  -v, --verbose         active verbose mode output results
  --gtk                 launch XSSer GTK Interface
  --wizard              start Wizard Helper!

  *Special Features*:
    You can set Vector(s) and Bypasser(s) to build complex scripts for XSS
    code embedded. XST allows you to discover if target is vulnerable to
    'Cross Site Tracing' [CAPEC-107]:

    --imx=IMX           IMX - Create an image with XSS (--imx image.png)
    --fla=FLASH         FLA - Create a flash movie with XSS (--fla movie.swf)
    --xst=XST           XST - Cross Site Tracing (--xst http(s)://host.com)

  *Select Target(s)*:
    At least one of these options must to be specified to set the source
    to get target(s) urls from:

    --all=TARGET        Automatically audit an entire target
    -u URL, --url=URL   Enter target to audit
    -i READFILE         Read target(s) urls from file
    -d DORK             Search target(s) using a query (ex: 'news.php?id=')
    -l                  Search from a list of 'dorks'
    --De=DORK_ENGINE    Use this search engine (default: yahoo)
    --Da                Search massively using all search engines
```

Figure 3.13: XSSer options

This option will be explained with more detailed in the next chapter, where it will be discussed the results.

4

Evaluation

The previous chapter explain which tools were going to be used, what was their purpose and how they work. This chapter intends to provide the results of tests and regarding this work understand if after the attacks the SDN controller was aftected in any way that have could disrupt the system and the network control and management. To establish a concept and exploit the actual vulnerabilities of a web application that connects to an SDN controller, was used a real and proprietary SDN solution.

4.1 White Box - Results

Burp Scanner is a tool for conducting automated scans of web sites to find content and audit for vulnerabilities. The task involved in executing a scan includes critical phases:

- Crawling for content - This entails navigating throughout the application, trailing links, submitting forms, and logging in where necessary, to record the content of the application and the navigational paths within it;
- Auditing for vulnerabilities - This involves examining the application's traffic and behavior to recognize security vulnerabilities and additional issues. Depending on the scan configuration, it may require sending a considerable number of requests to the application.

4.1.1 Crawling

Burp Suite crawling [55] prepares a phase of scanning involving navigating around the application, following links, submitting forms, and logging in where necessary, to catalog the content of the application and the navigational paths within it. This ostensibly simple task presents a variety of challenges that Burp's crawler can meet to create a precise map of the application. The crawling allows detecting some significant vulnerabilities such as core approach or session handling that be discussed furthermore.

4.1.1.1 Core approach

Burp's crawler travels throughout a target application in the same way a user with a browser, by clicking links and submitting input. It assembles a map of the application's content and functionality in the report of a directed graph, describing the different sections in the application and the links between those sections.

The crawler executes no presumptions about the URL structure used by the application. Locations are recognized (and re-identified later) based on their contents, not the URL that was used to reach them. It enables the crawler to certainly handle modern applications that place evanescent data such as Cross-site Request Forgery (CSRF) tokens or cache-busters into URLs. The strategy also concedes the crawler to manage applications that use the same URL to reach various locations, based on the nature of the application or the user's interplay with it. As the crawler operates around and increases coverage of the target application, it tracks the peaks in the graph that have not been achieved. These represent the links that have been recognized within the application but not yet attended. Nevertheless, the crawler never "jumps" to a indeterminate link and visits it out of context. Alternatively, it either drives straight from its current location or returns to the start location and navigates from there. Crawling in a way, makes no presumptions about URL structure and is extremely effective in dealing with modern web applications, but can likely lead to difficulties in seeing "too much" content. Recent web sites often hold a mass of unnecessary navigational paths, indicating that everything links everything else. Burp's crawler applies a diversity of methods to approach this subject: it builds up fingerprints of links to previously visited locations to avoid visiting them redundantly; it crawls in a breadth-first order that prioritizes the discovery of new content, and it has configurable cutoffs that restrain the extent of the crawl.

Accordingly to the results of Burp Suite about the portal, were identified with medium severity requests where the token appeared within the query string. However, the problem was in the token where the flag was not set to be covered, like <input type="hidden"> in the request header or any security policy what so ever in order to protect the token itself. The following figure 4.1 demonstrates the Burp Suite report regarding the token problem.

Figure 4.1: Report of token problem

This type of vulnerabilities can lead to sensitive information within URLs that may be logged in various locations, including the user's browser, the web server, and any forward or reverse proxy servers linking the two endpoints. URLs may also be revealed on-screen, bookmarked, or sent around users. They may be exposed to third parties via the Referer header if any off-site links are followed. Placing session tokens into the URL increases the risk of being captured by an attacker. It allows the attacker to perform actions over the controller by assuming the session and impersonates himself as the "real" owner of the session. The SDN controller nothing can do to prevent this type of attack because it already trusted the connection and any action executed via MAESTRO is going to be accepted, and operations performed in the network are also going to be accepted. These actions may be harmful to the controller in a way that might create entropy, deleting services already created, or even change passwords and lock everything for the users, executing a ransomware attack. The applications with these vulnerabilities should use an alternative mechanism for transmitting session tokens, such

as HTTP cookies or hidden fields in forms that are submitted using the POST method.

4.1.1.2 Session handling

Because Burp's crawler drives throughout a target application in the same way as a user with a browser, it can automatically operate with almost any session-handling mechanism that browsers can deal.

The crawler employs multiple crawler "agents" to parallelize its work. Each agent represents a distinct user of the application navigating around with their browser. Each agent has its cookie jar, which is renewed when the application issues it with a cookie. When an operator returns to the start position to begin crawling from there, its cookie jar is removed, to simulate an utterly new session.

The requests that the crawler makes as it navigates around are constructed dynamically based on the preceding response, so CSRF tokens in URLs or form fields are handled automatically. This allows the crawler to correctly navigate functions that use complex session-handling, with zero configuration by the user.

The following figure 4.2 demonstrates the result of the Burp Suite session-handling mechanism.

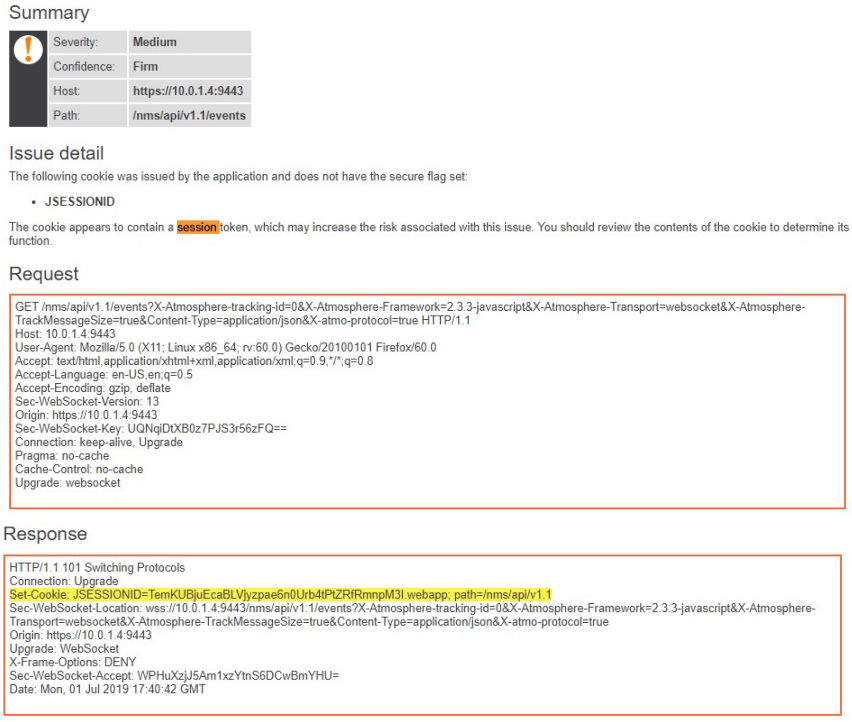


Figure 4.2: Report of session handling

This vulnerability has some common problems concerning the SDN controller as the core approach because this problem shows that the cookie jar contains the session token. The session token is represented with the parameter JSESSIONID, and if the secure flag is set on a cookie, then browsers will not submit the cookie in any requests that use an unencrypted HTTP connection, thereby restricting the cookie from being easily intercepted by an attacker that is monitoring the network traffic. If the secure flag is not set, then the cookie will be forwarded in clear-text if the user visits any HTTP URLs within the cookie's scope. An attacker may be able to provoke this event by supplying proper user links, either directly or via another web site. An attacker, to exploit this vulnerability, must be well-positioned to eavesdrop on the victim's network traffic. This scenario typically happens when a client communicates with the server over a vulnerable connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. An attacker situated in the user's network or the application's hosting infrastructure could also perform this attack. This vulnerability can be remediated, and for that, the secure flag should be set on all cookies that are used for sending sensitive data when reaching content over HTTPS. If cookies are used to send session tokens, then fields of the application that are accessed over HTTPS should apply their own session handling mechanism, and the session tokens used should never be dispatched over unencrypted communications.

4.1.2 Other reports

Burp Suite crawler also identifies issues regarding cookies without HttpOnly flag set, link manipulation (DOM-based), source code disclosure, unencrypted communications, private IP addresses disclosure, and some cacheable https responses. Each issue represents vulnerabilities to the system, some issues are more severe than the others, however, and according to the issues remediations given by Burp Suite report, they actively participate in the expose of the system to the outside threats and vulnerabilities. However, only the cookies without HttpOnly flag set and link manipulation (DOM-based) are more disruptive and critical to the system and are the ones to be discussed.

It is initiating with cookies without the HttpOnly flag set. This issue is very explored by the hackers to perform Cross Site Scripting (XSS), because if it is set the HttpOnly attribute on a cookie, then the cookie's value cannot be read or set by client-side JavaScript. This measure makes specific client-side attacks, such as cross-site scripting, slightly harder to exploit by preventing them from trivially

capturing the cookie's value via an injected script.

The figure 4.3 demonstrates the result obtained using Burp Suite.

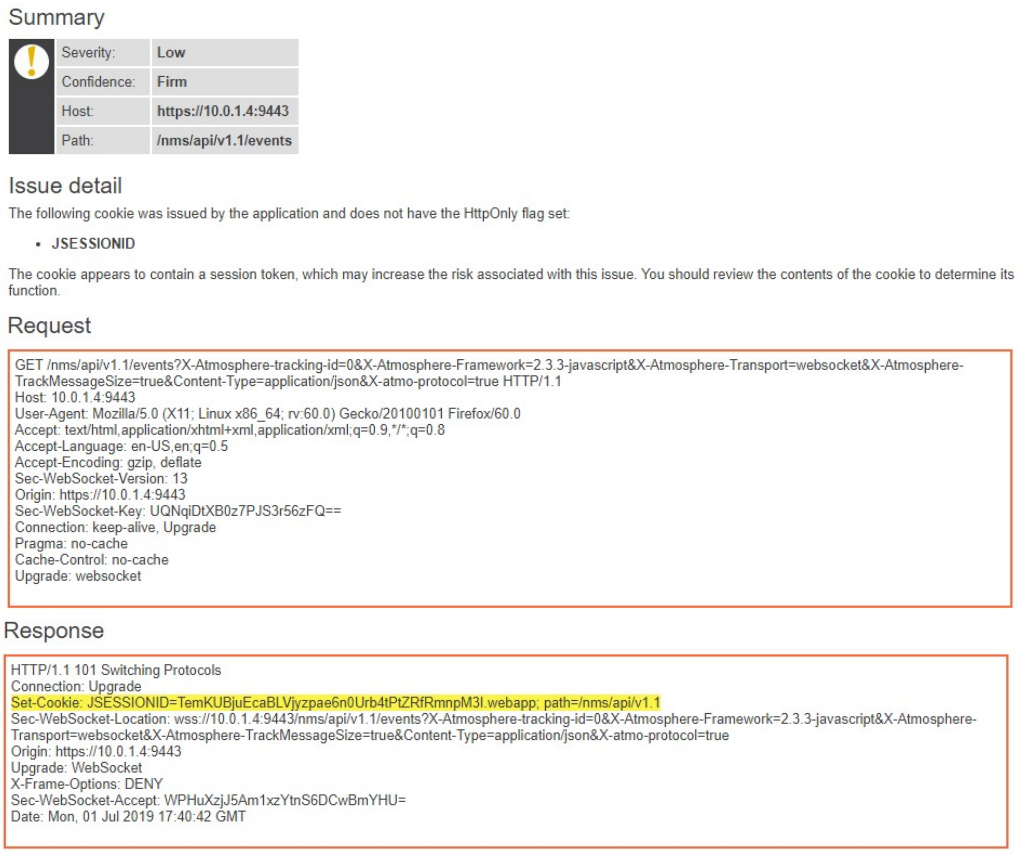


Figure 4.3: Report of cookie without HttpOnly flag set

DOM-based vulnerabilities appear when a client-side script gathers data from a controllable element of the DOM (for example, the URL) and prepares this data in an insecure way.

DOM-based link manipulation begins when a script letters controllable data to a navigation target in the current page, like a clickable link or the submission URL of a form. An attacker may be capable of using some vulnerability to assemble a URL that, if visited by another application user, will alter the victim of links inside the response. An attacker may be capable of leveraging this to produce several attacks, including:

- Making the user redirect to an uncertain external URL to promote a phishing attack.
- Making the user offer delicate form data to a server managed by the intruder.

- Making the user produce an unintended action inside the application by editing the file or query line connected with a link.
- Circumventing browser anti-XSS protection by injecting on-site links holding XSS exploits, as browser anti-XSS defenses typically do not run on on-site links.

The figure 4.4 demonstrates the result obtained using Burp Suite.

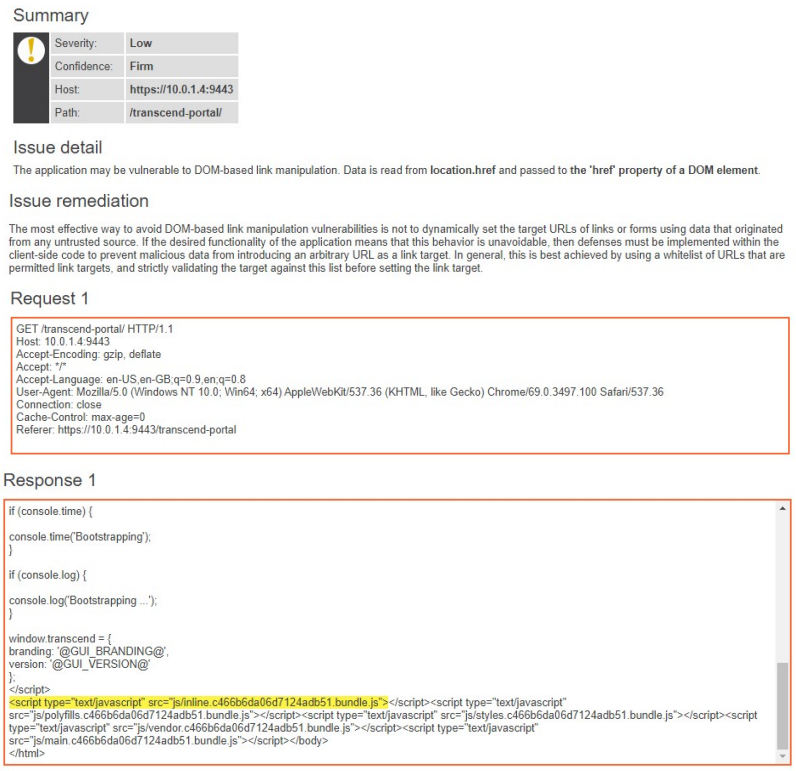


Figure 4.4: Report of link manipulation

Both vulnerabilities intend to exploit flaws using the JavaScript code, and in case of success, the controller is exposed because the session was stollen. This vulnerability might be exploitable, and the cookie might be retrieved via an injected script. If the HttpOnly value is set on a cookie, then the cookie's value cannot be read or set by client-side JavaScript, and as a result of not setting is the possibility to take over the SDN controller and the network itself. This action causes certain client-side attacks, such as cross-site scripting, slightly more laborious to exploit by preventing them from trivially taking the cookie's value via an injected script. However, this vulnerability can be remediated by simpling setting the HttpOnly flag; however, the restrictions imposed by this flag can potentially be circumvented in some circumstances, because other attacks can be deployed by client-side script injection.

4.2 Black Box - Results

This chapter has the purpose of using the tools that were explained in the previous chapter, use the functionality to find vulnerabilities in a web application that is connected to the SDN controller and therefore evaluate if any of this attacks has any impact that might disrupt the correct functioning of the SDN controller. This type of attack (black box) is performed without knowing the system from the inside, and this is why the first thing is to gather information about the network.

4.2.1 Network information gathering

Network information gathering allows the attacker to know which devices are connected to the network and for the attacker to be able to do that, it must be connected to the same network as the targeting devices. To obtain network information, the tool, netdiscover, was used, and make use of this software, the option -P and -N were used. The -r option is extra concerning because it scans a given range instead of auto-scanning. The result is presented in the following figure 4.5

```
root@kali:~# netdiscover -r 10.0.0.0 -P -N
 10.0.0.1        6c:3b:e5:f2:fb:4d      1      60  Hewlett Packard
 10.0.0.2        6c:3b:e5:f2:fb:4d      1      60  Hewlett Packard
 10.0.0.3        a0:1d:48:b6:f3:24      1      60  Hewlett Packard
 10.0.0.4        6c:3b:e5:f2:fb:23      1      60  Hewlett Packard
 10.0.0.5        e0:3f:49:36:bd:b6      1      60  ASUSTek COMPUTER INC.

-- Active scan completed, 5 Hosts found.
```

Figure 4.5: Netdiscover result

The result shows that there are five machines connected in the network, and the attacker represents five machines that might be exploitable. The main focus of this attack is to get information about a web application; however, the attacker with this information cannot get the full information about the content of each machine.

4.2.2 Web application information gathering

The attacker has now the primary purpose of finding which machine has a web server that might be exploitable. To be able to simulate that process in an automated manner was developed a python script. The script was developed using PyCharm Community 2019.2 from JetBrains and is divided into three major

phases, to being the first one, to gather information about all ports from the targets, which means understand which ports are open and which application is running. The script uses tools like NMAP to gather the information about the open ports of all ports, as it is shown in the following excerpt of code in figure 4.6 where is demonstrated how it was implemented.

```
def scan_targets(targets, ports_range):
    print("Start scanning...")
    nm = nmap.PortScanner()
    nm.scan('10.0.0.0/29', '1-65535')
    nm.scaninfo()
    nm.csv()
    sockets = []
    for host in nm.all_hosts():
        sockets.append(host)
        print('------------------')
        print('Host; %s (%s)' % (host, nm[host].hostname()))
        print('State : %s' % nm[host].state())
        for protocol in nm[host].all_protocols():
            print('------------------')
            print('Protocol; %s ' % protocol)
            lport = nm[host][protocol].keys()
            aux = []
            for port in lport:
                aux.append(port)
                print('port : %s\tstate : %s' % (port, nm[host][protocol][port]['state']))
            sockets.append(aux)
    return sockets
```

Figure 4.6: AnalyzePorts python script (Nmap excerpt implementation)

The primary purpose of this code is gathering information about open ports of all targets in the network. Therefore the arguments of the function are a list of the target IPs and the range of ports to analyze. The function core uses the Nmap tool and prints some information that might be useful to understand which part of the code is running when it is launched the script. The figure 4.7 shows an output example of this excerpt.

```
Host; 10.0.0.4 ()
State : up
------------------
Protocol; tcp
port : 135  state : open
port : 139  state : open
port : 445  state : open
port : 5040 state : open
port : 9443 state : open
port : 49664    state : open
port : 49665    state : open
port : 49666    state : open
port : 49667    state : open
port : 49668    state : open
port : 49669    state : open
port : 49670    state : open
```

Figure 4.7: AnalyzePorts python script (Nmap excerpt output example)

The function returns a list of sockets (IP and port) combinations to be evaluated in the second phase. The second phase consists of understanding if this open ports support HTTP, HTTPS, or both protocols and evaluating if there is a HTML body that might be used to exploit. The following figure 4.8 and figure 4.9 show the excerpts of code used in this phase.

```
def getHttpRequest(IP, port):
    URL = "http://" + IP + ":" + str(port)

    try:
        r = requests.get(URL, timeout=5)
        r.close()
        print(r)

        return 1
    except requests.exceptions.ConnectionError:
        print("This URL = %s does not support http" % (URL))
        return -1
    except requests.exceptions.ConnectTimeout:
        print("This URL = %s does not support http" % (URL))
        return -2
    except requests.exceptions.ReadTimeout:
        print("This URL = %s does not support http" % (URL))
        return -3

    return 0
```

Figure 4.8: AnalyzePorts python script (HTTP excerpt implementation)

The main purpose of this code is evaluate if a port supports, or not, HTTP or

```python
def getHttpsRequest(IP, port):
    URL = "https://" + IP + ":" + str(port)

    with no_ssl_verification():
        try:
            r = requests.get(URL, verify=False, timeout=5)
            r.close()
            print(r.content.title())

            return 1
        except requests.exceptions.SSLError:
            print("This URL = %s does not support https" % (URL))
            return -1
        except requests.exceptions.ConnectionError:
            print("This URL = %s does not support https" % (URL))
            return -2
        except requests.exceptions.ConnectTimeout:
            print("This URL = %s does not support https" % (URL))
            return -3
        except requests.exceptions.ReadTimeout:
            print("This URL = %s does not support https" % (URL))
            return -4

    return 0
```

Figure 4.9: AnalyzePorts python script (HTTPS excerpt implementation)

HTTPS and in case it does support is printed the HTML body. The figure 4.10 below shows an output example of this excerpt.

```
9000
This URL = http://10.0.0.2:9000 does not support http
b'<!Doctype Html>\n\n<Html Xmlns="Http://Www.W3.Org/1999/Xhtml">\n<Head>\n\t<Title>Tnms - Login</Title>\n
 <Meta Charset="Utf-8" />\n    <Meta Http-Equiv="X-Ua-Compatible" Content="Ie=Edge" />\n    <Meta Name="View
port" Content="Width=Device-Width, Initial-Scale=1" />\n\n    <Link Rel="Icon" Type="Image/X-Icon" Href="/Im
ages/Icon/Favicon.Ico" />\n\n    <!-- Style -->\n    <Link Href="/Webjars/Bootstrap/3.3.6/Css/Bootstrap.Min.
Css" Rel="Stylesheet" />\n    <Link Href="/Css/Style.Css" Rel="Stylesheet" />\n\t\n    \n\n    <!-- Html5 Sh
im And Respond.Js For Ie8 Support Of Html5 Elements And Media Queries -->\n    <!-- Warning: Respond.Js Does
n\'T Work If You View The Page Via File:// -->\n    <!--[If Lt Ie 9]>\n    <Script Src="Https://Oss.Maxcdn.C
om/Html5Shiv/3.7.2/Html5Shiv.Min.Js"></Script>\n    <Script Src="Https://Oss.Maxcdn.Com/Respond/1.4.2/Respon
d.Min.Js"></Script>\n    <![Endif]-->\n</Head>\n<Body Class="Login-Body">\n    <Div Class="Container-Fluid">
\n            <Div Class="Panel Panel-Default Panel-Login">\n                <Div Class="Panel-Body">\n
              <Div Class="Brand">\n                        <Img Src="/Images/Logo/Png/Company-Logo.Png" Cla
ss="Img-Responsive" Alt="Infinera" />\n                    </Div>\n\n                    <Div Class="Title">
\n                        <H5>Tnms</H5>\n                    </Div>\n                    <Div Class="Product
">\n                        <H3>Client Portal</H3>\n                </Div>\n\n                    \n
              \n\n                    <Form Method="Post" Action="/Login">\n                        <Div C
lass="Form-Group">\n                            <Input Type="Text" Name="Username" Class="Form-Control" Id="
Username" Placeholder="Username" />\n                        </Div>\n                        <Div Class="For
m-Group">\n                            <Input Type="Password" Name="Password" Class="Form-Control" Id="Passw
ord" Placeholder="Password" />\n                        </Div>\n                        <Button Id="Submit"
Type="Submit" Class="Btn Btn-Primary">Log In</Button>\n                    <Input Type="Hidden" Name="_Csrf"
 Value="2B723930-16Fe-4023-8D91-3Ba16F89E7B5" /></Form>\n\n                <Div Class="About">\n
              Version\n                        <Span>18.00.3.37.0</Span>\n                    </Div>\n
          </Div>\n            </Div>\n        </Div>\n\n    <Script Src="/Webjars/Angularjs/1.4.8/Angular.
Min.Js" Type="Text/Javascript"></Script>\n    <!-- Jquery (Necessary For Bootstrap\'S Javascript Plugins) --
>\n    <Script Src="/Webjars/Jquery/2.1.4/Jquery.Min.Js" Type="Text/Javascript"></Script>\n    <!-- Include
All Compiled Plugins (Below), Or Include Individual Files As Needed -->\n    <Script Src="/Webjars/Bootstrap
/3.3.6/Js/Bootstrap.Min.Js" Type="Text/Javascript"></Script>\n</Body>\n</Html>'
```

Figure 4.10: AnalyzePorts python script (HTTP and HTTPS excerpt output example)

To complete the results, it was essential to understand which one of thus open

ports that had HTML information; therefore, the third phase was implemented and consists of using a tool named Nikto. The following figure 4.11 is the excerpt that demonstrates how this tool was integrated into the previous script.

```
def getInfoAboutWebServer():
    for i in range(len(sockets_with_info)):
        if i % 2 == 0:
            print("-------------------------------------------------------")
            host = sockets[i]
        else:
            for j in sockets[i]:
                host_port = str(host) + ':' + str(j)
                result = subprocess.check_output(['nikto', '-host', host_port])
                if result.decode("utf-8").__contains__("No web server"):
                    print("The socket " + host_port + " is not a web server")
                else:
                    print("The socket " + host_port + " is a web server")
                    print(result)
                print("<<<<<<<<<<<<<<<<<<<<<<")
    print("-------------------------------------------------------")
```

Figure 4.11: AnalyzePorts python script (Nikto excerpt implementation)

Nikto was used to validate if a specific socket has a web server. Nikto was used to every port opened that had HTML body. The result is presented in the following figure 4.12.

```
<<<<<<<<<<<<<<<<<<<<<<
The socket 10.0.0.4:9443is a web server
b"- Nikto v2.1.6\n---------------------------------------------------------------------------\n+ Target IP:
   10.0.0.4\n+ Target Hostname:    10.0.0.4\n+ Target Port:        9443\n---------------------------------------
--------------------------------\n+ SSL Info:        Subject:  /O=OSSNMS/OU=TNMS/CN=TNMS Server\n
    Ciphers:  ECDHE-ECDSA-AES128-GCM-SHA256\n                Issuer:   /O=OSSNMS/OU=TNMS/CN=TNMS Server\n+ Star
t Time:         2019-11-01 13:36:12 (GMT-5)\n---------------------------------------------------------------------------
------\n+ Server: No banner retrieved\n+ The X-XSS-Protection header is not defined. This header can hint to the u
ser agent to protect against some forms of XSS\n+ The site uses SSL and the Strict-Transport-Security HTTP header
is not defined.\n+ The site uses SSL and Expect-CT header is not present.\n+ 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
\n+ No CGI Directories found (use '-C all' to force check all possible dirs)\n+ Hostname '10.0.0.4' does not match
 certificate's names: TNMS\n+ 7888 requests: 0 error(s) and 5 item(s) reported on remote host\n+ End Time:
   2019-11-01 13:40:03 (GMT-5) (231 seconds)\n---------------------------------------------------------------------------
-------\n+ 1 host(s) tested\n"
<<<<<<<<<<<<<<<<<<<<<<
```

Figure 4.12: AnalyzePorts python script (Nikto excerpt output example)

After analyzing the results, there is a web application in the machine with socket 10.0.0.4:9443. It has a certificate with cipher ECDHE-ECDSA-AES128-GCM-SHA256, which means that the key exchanger is the Elliptic-curve Diffie–Hellman (ECDH), authentication Elliptic Curve Digital Signature Algorithm (ECDSA), the cipher is the Advanced Encryption Standard (AES) with 128 bits and with Galois Counter Mode (GCM), and with hash algorithm Secure Hash Algorithms (SHA) with 256 bits. This information might be useful for the attacker, depending on the type of

attacks that are going to be executed. What concerns the SDN controller; at this moment, it was not yet be subjected to any attack that might cause instability to it. These results were based only on getting information from the machines and the network.

4.2.3 Dictionary attack

At this point, the attacker knows who is the target; however, some information is missing because of the HTML information of the socket 10.0.0.4:9443 returned the error code 404 - Not Found, but the webserver is running. To overcome this issue, the attacker must map the web application, and for that matter, the toll Skipfish was used and following figure 4.13 shows the output of the Skipfish scan that allows the gathering of the necessary information about the full Uniform Resource Locator (URL) of the Transcend Maestro page.

Figure 4.13: Skipfish scan result

After getting the full URL to the Maestro, the access was protected with user and password. For an attacker to perform the correct attacks is essential to identify which type of web application comes across the attacker. Classifying this is a Portal Web App type because it is an application that accesses the various sections or categories through a home page. The following figure 4.14 shows the login page of Maestro.

Now that the attacker gained access to the login page is important to inspect the page to try to obtain information that might be used to gain access and explore the content of the web application.

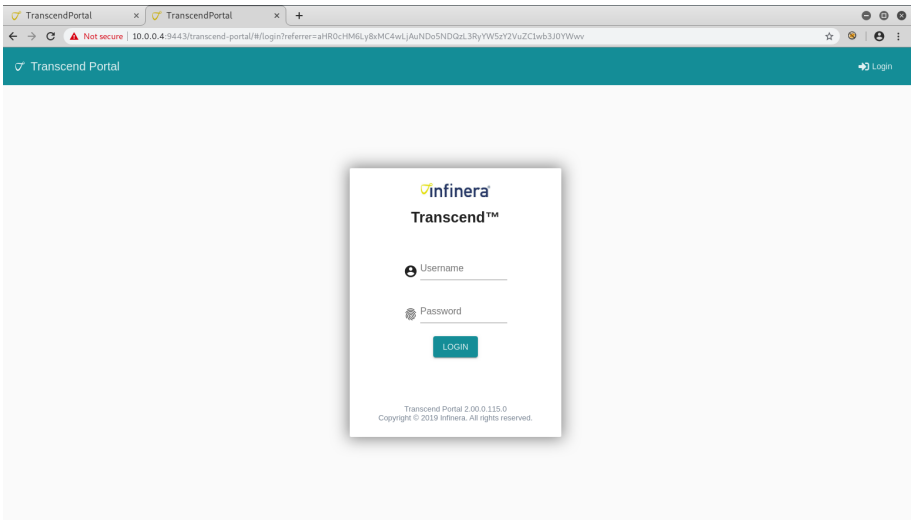


Figure 4.14: Transcend Maestro login page

Inspecting the login page leads to acquired curious information about the web application. After selecting inspecting the web page and in the tab network one of the requests with the name, info, had the information about all deployments of the product as it is shown in the figure 4.15

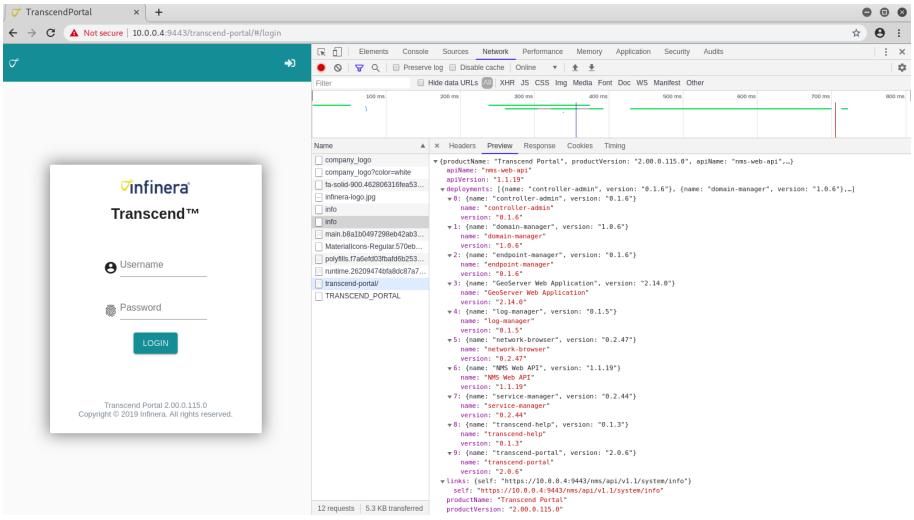


Figure 4.15: Transcend Maestro login page - Inspected

After trying to access every deployment, the actual result was that because no login was successful; therefore no token for the session created, the requests for each implementation end up redirected to the login page, which represents a correct behavior from the system. However, this might be useful to the attacker in case he gains access, and some of the deployments are not directly available in the application.

At this point, the attacker already explored the surroundings of the web application, and the main target is to gain access to the web application, and the first way to try to get it is through a dictionary attack. These attacks are time-consuming, and if the password is strong, it might take years to the attacker to gain access. However, to perform a dictionary attack, the only thing that is needed is a dictionary file with various user/passwords. Dictionary attacks are the last but not the least choice from the hackers because it is the method that takes the longest, but it might be the last option in the book. To bypass this barrier and since these tests are academic and only intend to simulate a scenario where an actual breakthrough happened, the used users and passwords files had the real credentials to log in successfully. This file was obtained here [56], and the content results of leaked and preferred users and passwords combinations. These tests were performed using two different ways using a python script designed and created from scratch and using Burp Suite manual tools.

4.2.3.1 Dictionary attack with automated script

The automated script was already explained in the previous chapter and the result of the script to a dictionary attack is shown in the following figure 4.16

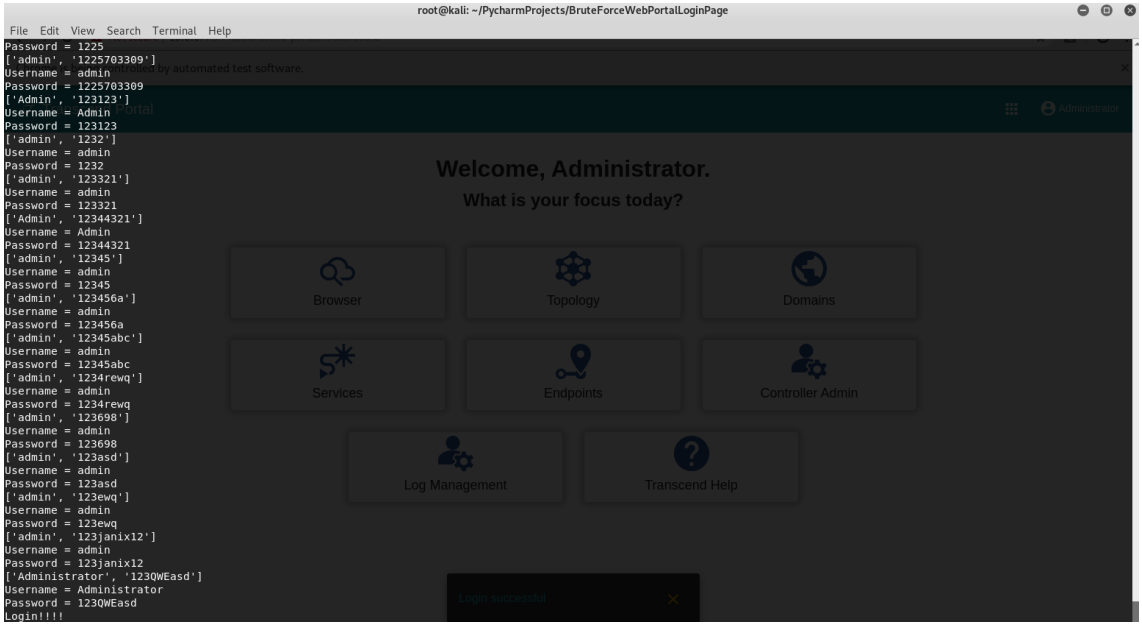


Figure 4.16: Dictionary python script attack - Result

Looking at the results was possible to confirm that the login was succeeded and that the URL of the web page is in the dashboard.

4.2.3.2 Dictionary attack with Burp Suite

After that, all requests firstly made are manually analyzed over the browser by Burp Suite. When the first request of the login attempt arrives at Burp Suite, the username and password are marked, is used a tool, incorporated in Burp Suite, named Intruder, and that payload is sent off to that tool. There are four types of attacks available, but only two of them are suitable for this task because we need to work with more than one payload set. Thus attacks are the Cluster Bomb that combines all the possibilities within each payload and the Pitchfork attack, and this is a more lightweight because it iterates each list independently. The following figure 4.17 and figure 4.18 demonstrates respectively thus payloads.

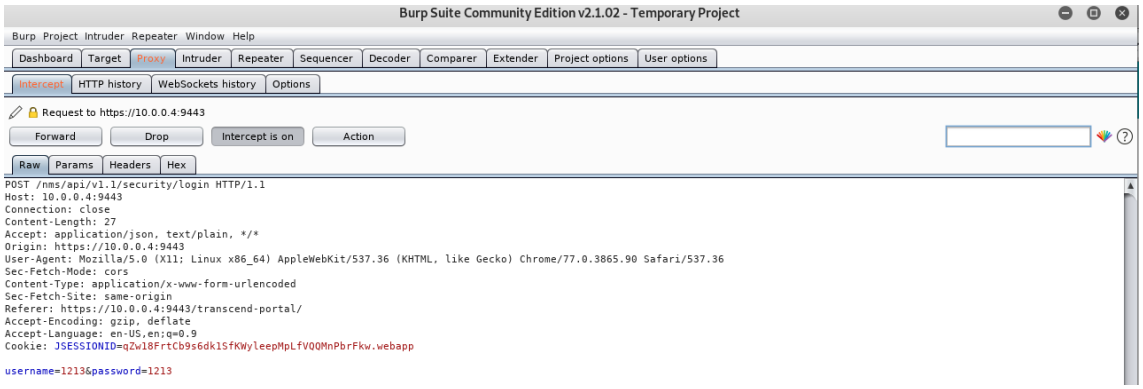


Figure 4.17: Burp Suite - Payload of login request

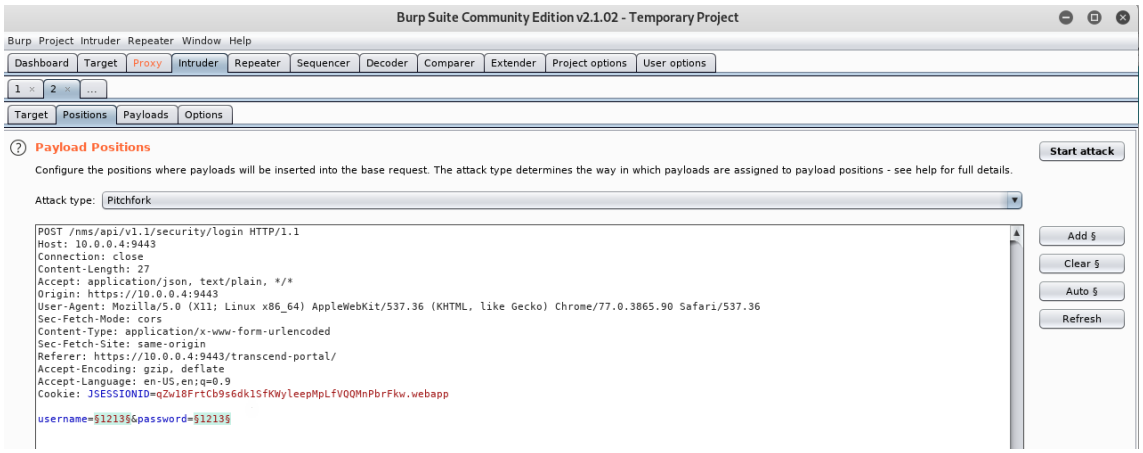


Figure 4.18: Burp Suite (Intruder) - Payload of login request

Intruder, accordingly to Burp Suite [57], is a tool for automating customized attacks against web applications. For intruder to work, firstly, it must be assigned

the payload fields that we want to modify and recursively test against the server, the same way, in this case, we want to iterate different options of users and password against the server, so this variable must be selected. The figure 4.19 and figure 4.20 demonstrates the set of both payloads for the intruder tool.

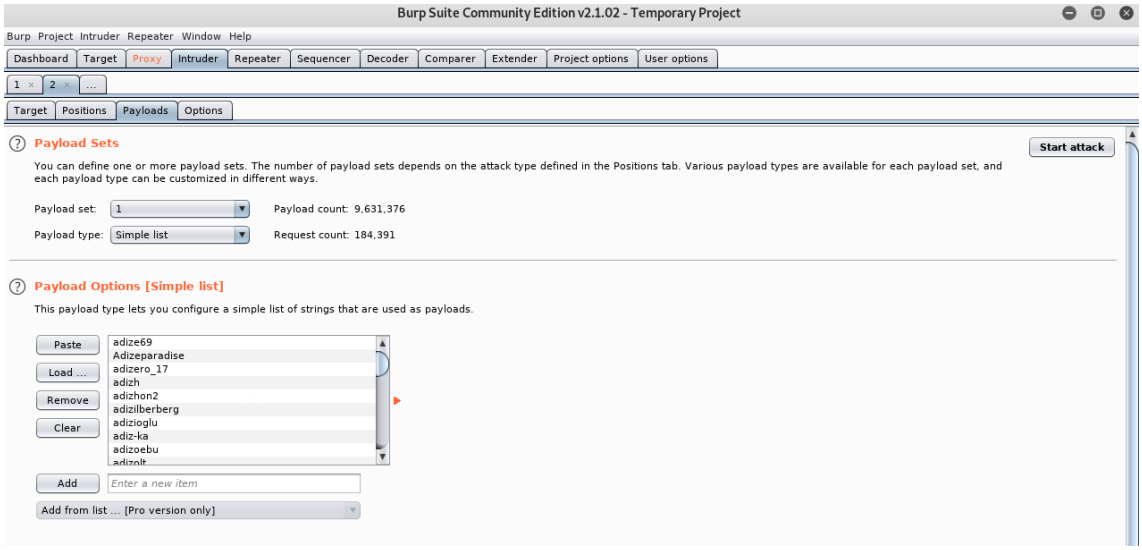


Figure 4.19: Burp Suite (Intruder) - Set payload 1

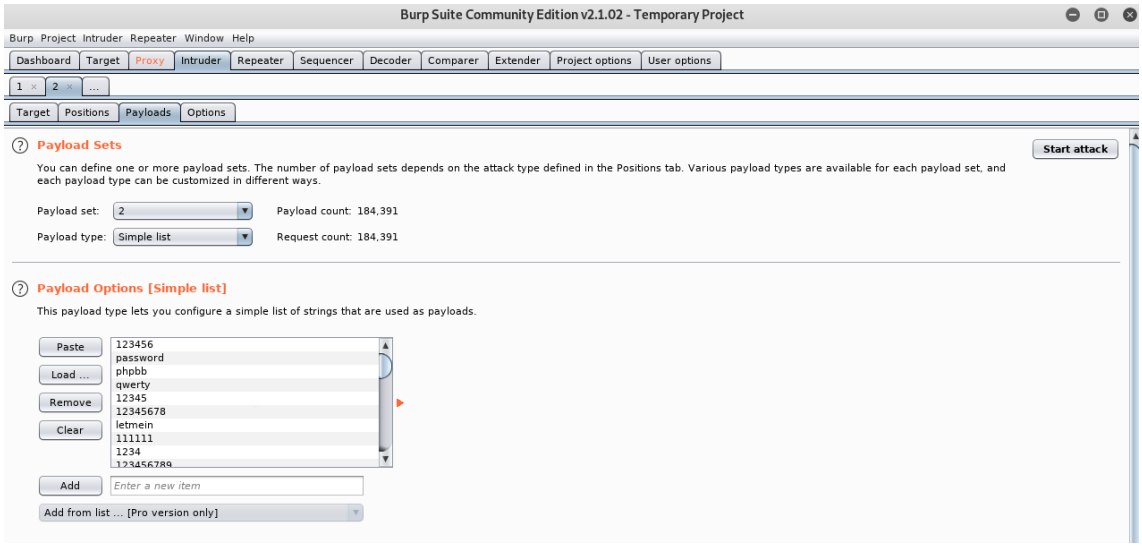


Figure 4.20: Burp Suite (Intruder) - Set payload 2

The result of Burp Suite to the dictionary attack is shown in the following figure 4.21.

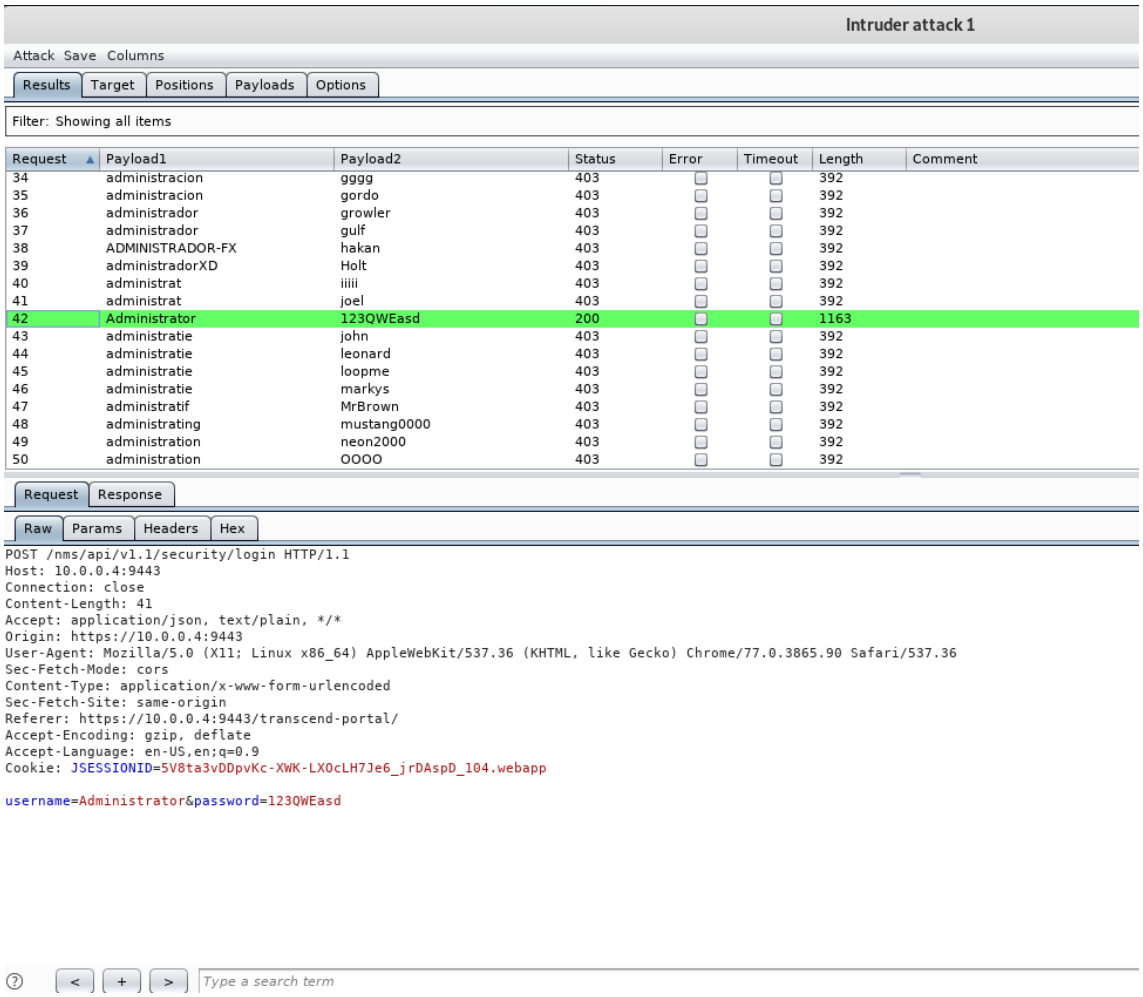


Figure 4.21: Burp Suite - Dictionary attack result

4.2.3.3 Analysis

Now that both automated script and the Burp Suite were executed is possible to conclude two things. The first one is concerned about the response to the failed attempts. In this case, for each failed attempt, there was only the information about the login failed, and no specification about why it has failed, in case the user and the password are wrong there is no specificity about why it has failed. The implementation could have been entirely different if, for example, a message with Account Lock Out appeared; however, in some instances, this could be due to a lockout policy based on a certain number of wrong login attempts. Although intended to protect the account, such policies can frequently give rise to additional vulnerabilities. A malicious user may be able to lockout multiple accounts, denying access to a system. Besides, a locked-out account may cause variances in the behavior of the application; this behavior should be explored and potentially

exploited, and also, the attacker would be aware that this specific user exists in the system and only has a password that needs to be discovered. The second thing is related to the dictionary attack testing itself because these types of tests are very stressful because it takes many resources out from the machine, and this might limit the operations and the correct functioning of the system. What concerns the SDN controller, it had to manage all the upcoming request from the Maestro, process them, and send them to the right entity; hence, this attack might be very critical to the machine resources the machine of the SDN controller did not crave. The figure 4.22 and 4.23 shows the state of the SDN machine before and during the dictionary attack via automated script.

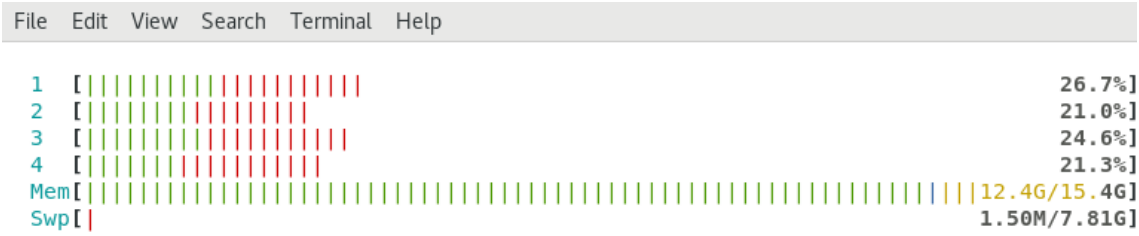


Figure 4.22: SDN machine state - Before dictionary attack with automated script

Figure 4.23: SDN machine state - During dictionary attack with automated script

While the figure 4.24 and 4.25 shows the state of the SDN machine before and during the dictionary attack via Burp Intruder.

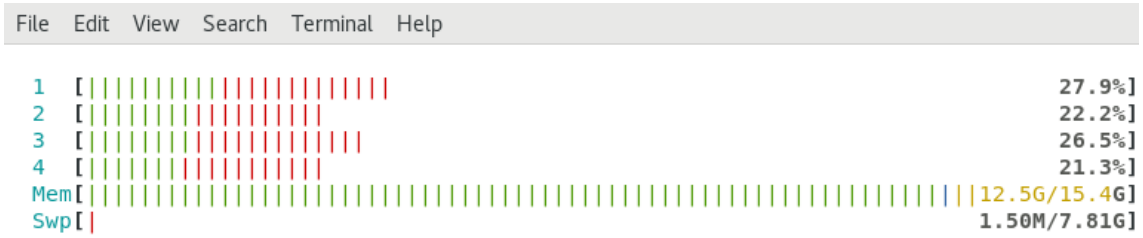


Figure 4.24: SDN machine state - Before dictionary attack with Burp Suite

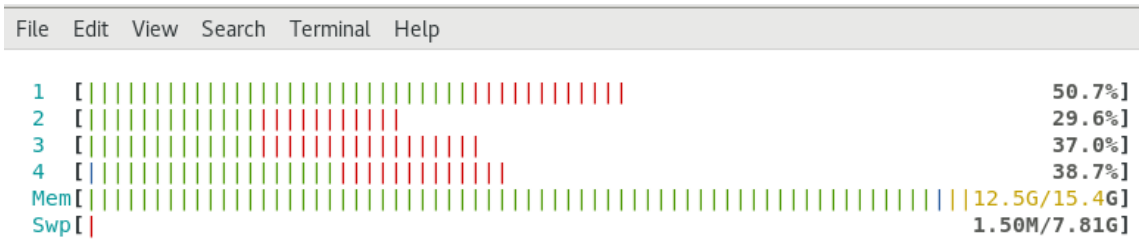


Figure 4.25: SDN machine state - During dictionary attack with Burp Suite

Analyzing the results, it is possible to understand that these types of attacks did not produce a Denial Of Service (DoS) on the controller, which represents a very good result concerning the availability of the software and the system as an all. This might have been possible because of a load balancing mechanism that is implemented in the Nginx component inside of the SDN controller.

4.2.4 SQL Injection

In the previous chapter, it was discussed the concerns about the SQL Injection attacks, what they might represent in terms of impacts for the solutions, and the network itself. It was also discussed some tools that might help to improve the detection and analyze if specific applications can be subjected to this type of attack. For this particular issue, it was used the Burp Suite Intruder tool and SQLMap tool.

4.2.4.1 SQL Injection with Burp Suite

The first tool, Burp Suite Intruder, was used to perform some SQL Injection tests. Thus tests consist of using one of the requests that contained in the URL a query to allow the modification of that value to perform the attack. Firstly it was checked

all the requests manually to during the load of the login page, and only one request had in the URL a query that might eventually be exploitable. The following figure 4.26 shows all requests from the web application during the loading process.

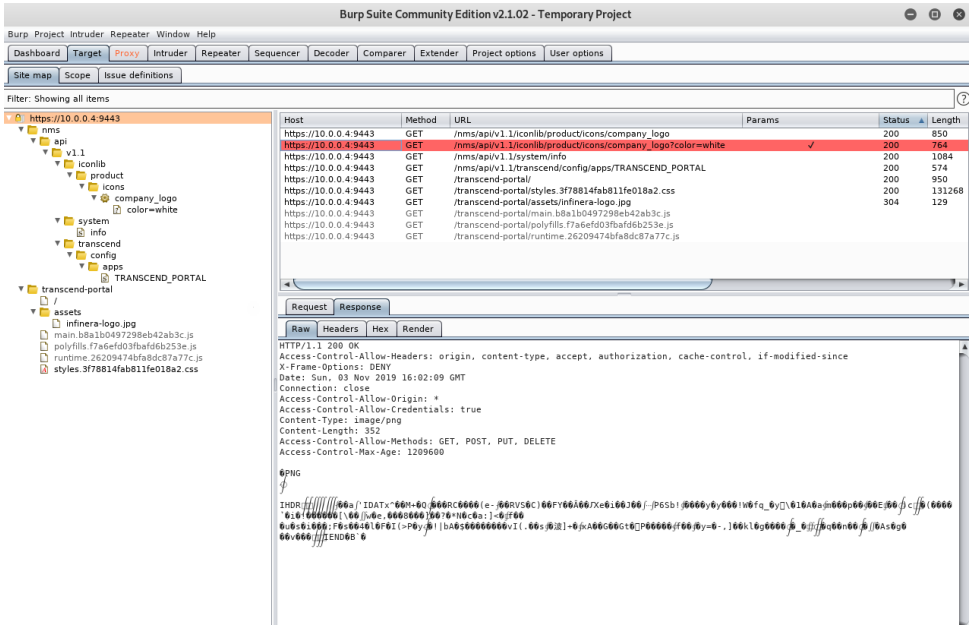


Figure 4.26: Burp Suite login page - Requests

After that, the request was sent to the intruder tool, and some payloads were tested to check if changing that attribute the response had any information that might be useful. The following figure 4.27 shows the used payloads. Because this type of vulnerabilities is related to databases, some of the standard nomenclatures of databases stand in the use of characters like *, ? or ”. That was it was intended to test, evaluate if using this payload in that field the application returned some information that could be used by the attacker.

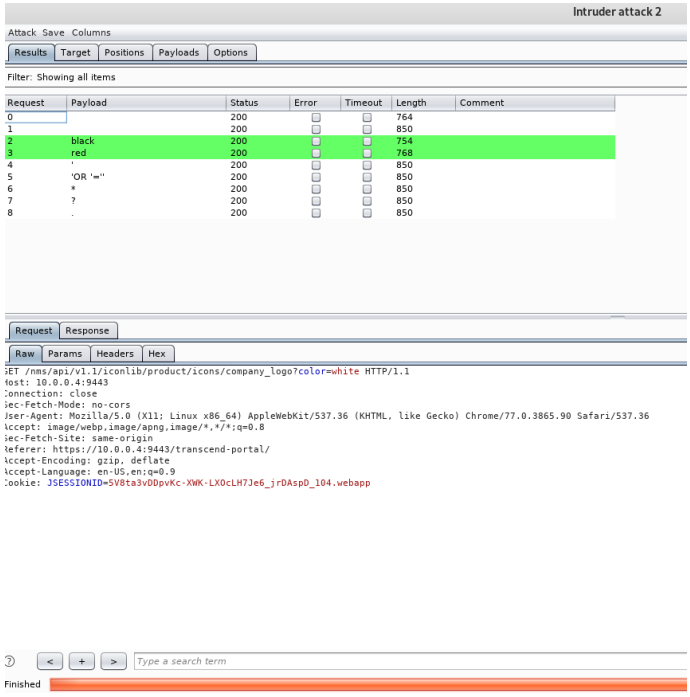


Figure 4.27: Burp Suite - Intruder requests

It is possible to analyze that all the replies return 200 OK, which means that all the requests were accepted and analyzing the responses there is no useful information, different from the original request.

4.2.4.2 SQL Injection with SQLMap

Another tool, named SQLmap, was used. It is an automated tool and works as a crawler and tries to find SQL injection vulnerabilities that might be exploitable in this matter. The command used to use this tool was **sqlmap –url https://10.0.0.4:9443/transcend-portal/login/ –user-agemt=SQLMAP –batch –level=5 –risk=3 –dbms=ORACLE**. Since this is an automated tool, the result is clear, so after executing the command and after the tool injected and execute various tests, the result was that no parameters seem to be injectable, as it is shown in the figure 4.28.

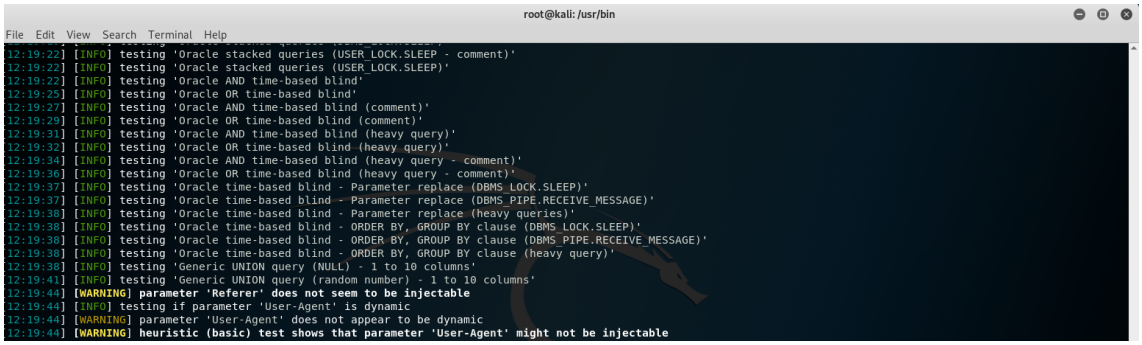


Figure 4.28: SQLmap result

4.2.4.3 Analysis

After analyzing these results, it is possible to affirm that this web application is resilient to SQL Injection attacks. The workflow of the requests consists that every request is sent and analyze by the SDN controller; therefore, the content inside each message is validated by the Northbound Interface component in the SDN controller. In this case if the content of the message is verified and if the there are some SQL Injection attack disguised in the request the SDN controller might be vulnerable because it can assume that this the content of the request is valid to return a correct response but with some critical information regarding the system, depending in how the attacker created the request. Accordingly to the results, the SDN controller, even though it receives some requests attempting to use SQL Injection attacks, the response of the controller was very satisfactory this because no response had critical information about the system and using the result of Burp Suite it is possible to validate and validate this point by presenting proof that even though all request were valid no response had sensitive information about the system, and is safe to assume that being the SDN controller the intermediate between the Maestro and the Network nothing was not compromised what concerns to this type of attacks. This is also an excellent example of good practices, and for that reason, the SDN controller is very resilient to these types of attacks and.

4.2.5 XSS

In the previous chapter, it was discussed the concerns about the cross site scripting attacks, what they might represent in terms of impacts for the solutions, and the network itself. It was also discussed some tools that might help to improve

the detection and analyze if specific applications can be subjected to this type of attack. For this particular issue, it was used the Cross Site "Scripter" tool.

4.2.5.1 XSS with Cross Site "Scripter"

This is an automated tool that has tests for different issues; the first test is a simple one and consists of using only the url, which means that the command used for this test was **"xsser -u "https://10.0.0.4:9443/transcend-portal/login/""**. The following figure 4.29 shows the result.

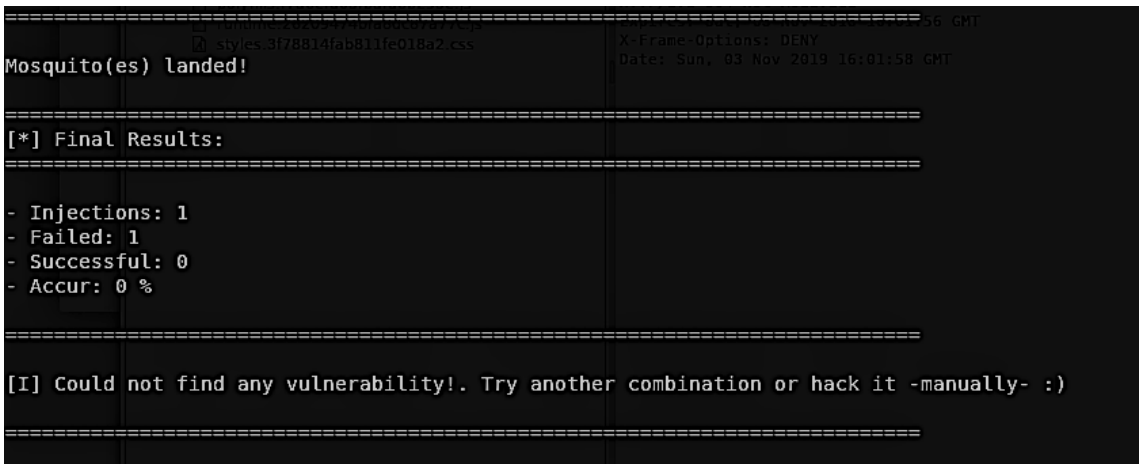


Figure 4.29: XSSer - simple command

The next test brings more parameters; the command used was **"xsser -u "https://10.0.0.4:9443/transcend-portal/login/" –auto –reverse-check"**. The difference with these parameters is that with –auto is injected a list of vectors provided by XSSer and with the parameter –reverse-check establish a reverse connection from target to XSSer to certify that is completely vulnerable. Therefore the following figure 4.30 shows the result.

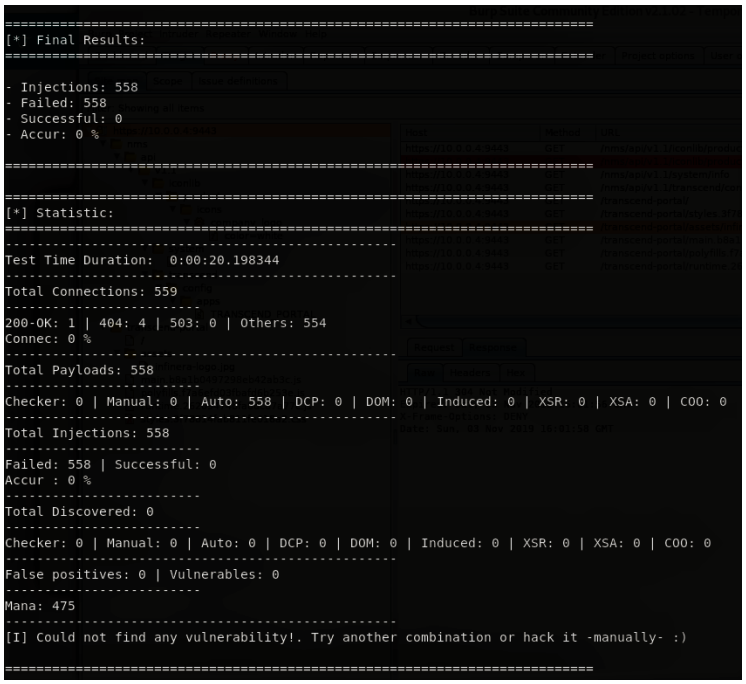


Figure 4.30: XSSer - Auto vectors list and reverse check command

The following test is a little bit different and the command used was **"xsser -u "https://10.0.0.4:9443/transcend-portal/login/" –Coo –Dom –Fp="<script> alert(123)</script>""**. This test consists of injecting cookies and using DOM shadow, which means exploiting user sessions that might be saved in the browser and use the DOM code of the website to explore vulnerabilities by validating the existence of configurable HTML fields. That is the reason that in this test, it was used the –Fp parameter that allows using exploitable code. The exploitable code used to intend to create a warning window and write 123, but this is only possible if a vulnerability is discovered. The following figure 4.31 shows the result.

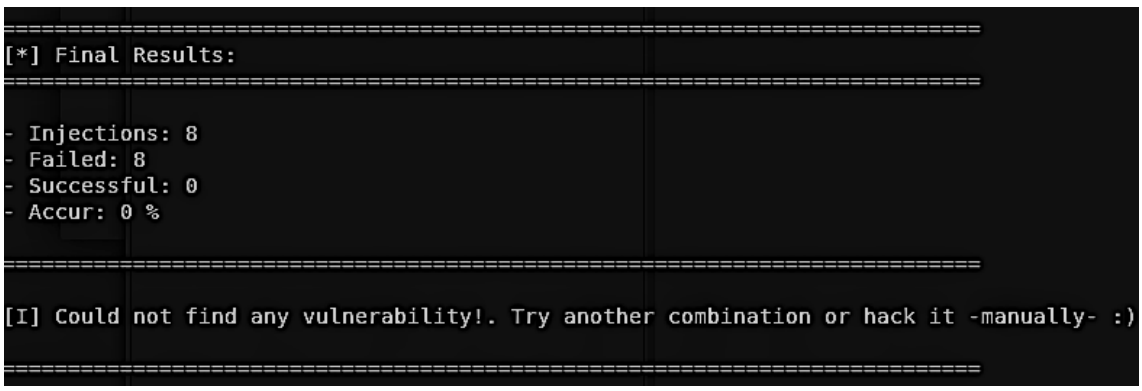


Figure 4.31: XSSer - Cookie and DOM command

This last test is based in heuristics, where the objective is to discover parameter filtered through the method of "guessing" or "intuition", which means using heuristics. This method starts to send some requests and depeneding on the web application response the next requests are adapted acoordingly, i a way of guessing and assumption regarding thus responses. The command used was **"xsser -u "https://10.0.0.4:9443/transcend-portal/login/" –heuristic"**. The following figure 4.32 shows the result

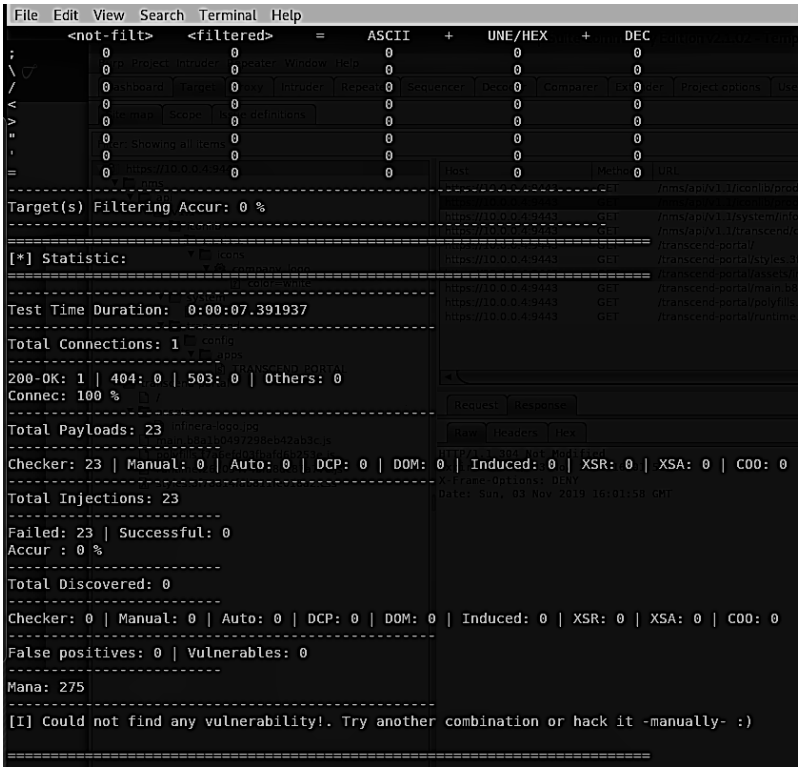


Figure 4.32: XSSer - Heuristic command

4.2.5.2 Analysis

By comparing all the results, it is possible to understand that none of the results returned any vulnerability; but it is required to analyze each performed test. The first one had the unique intend of understanding if a simple test could find any vulnerabilities regarding cross-site scripting in web applications. The second test has higher complexity, and the result was also very positive because not only it was able to identify any vulnerability but also showed that the SDN controller was not disrupted. The connection between the web application and the controller was stable. The third test aimed to inject cookies to exploit the data of the session and successfully deploy the attacks, but as the results show, no exploitable

code was found, and injecting cookies was not enough to break the security barrier between the web application and the controller. It also a very positive result to visualize that even though injecting various session cookies, the SDN controller handles it pretty well because the previous connection was not lost, and the stability maintained. The last but not least test intended to explore vulnerabilities in a more ad-hoc way which means that there was no vulnerability identified, and the main purpose was to find one by performing attacks based on the intuition to understand how not pre-available attacks vectors might be directly linked to specific issues and understand if, for some reasons, actual vulnerabilities in this web application might be hidden. Concluding all the reports, where nothing in the web application was exploitable regarding this type of attack, it is almost safe to assume that the web application is protected against XSS attacks.

5

Conclusions

The study presented in this thesis illustrates that SDN is something to take into account in the next few years, and it will be certainly be incorporated by various service providers. The case of AT&T is the most respectful because when SDN was very green, they took the risk and adopt SDN in their network because they need "desperately" to have automation in their network. So if AT&T took the step, the other would follow.

Having the possibility to test the security vulnerabilities of a real and deployed solution brings something different and new in the community of SDN, more precisely the community of security of SDN, this because all the work already made to these days is only based on emulated environments. Although these tests weren't performed directly to the controller itself, the primary purpose was to avoid breaches to the controller through web applications that connect to the controller via the northbound interface and also understand how thus attacks might create entropy to the SDN controller, how the web applications connected to the northbound interface can affect the controller when they are under attacks by a third entity.

As a conclusion to this work, it was possible to validate that the Infinera solution pack gives the clients a very safe and reliable environment regarding the performed attacks. Thus attacks were separated into two different testing mechanisms, white box, and the black box. The white box testing consists in assessing the solution knowing what it is inside the box, and in terms of security thus

attacks are conducted knowing the system, while in the black box testing the attacker can't see what it is inside the box and needs to investigate and collect data about the network and the solution before performing attacks such as SQL Injection or Cross-Site Scripting. After analyzing the results all of them had a favorable report, and the SDN controller was not destabilized and the communication between web application and controller was never broken and all the tests performed were directly targeted to the web application even though that thus attacks might had repercussions to the controller, such as gaining access to the controller and the network itself, deleting all the configurations and services creating a shutdown in the entire network.

SDN networks will bring a new breath to the traditional networks because the possibilities are tremendous at very different levels. The development of this thesis will bring more security information about the Infinera solution pack. This information will be given to the costumers enabling them to embrace this technology with more confidence now that SDN is becoming a more stable product and more reliable in terms of security.

5.1 Future Work

Regarding future work there are some possibilities, like improving the developed scripts by giving them more functionality and making them more resilient to errors, allowing Infinera and the security testers to rely more on proprietary tools than other tools that might be limited for the implementation and tests that might be needed.

There is also the possibility of exploiting new vulnerabilities like Distributed Denial Of Service (DDoS) or path traversal. Create new scripts and tools to explore these vulnerabilities. As future work the test of others web application deployed in the Infinera solution pack are also a target for testing.

It is also essential to automate the security tests regarding the web application to have a recurrent security test every time a build is deployed to ensure that the problems discovered are fixed in the time of the release date.

Bibliography

[1] Infinera. Infinera transcend software suite. (accessed: 03.09.2019). [Online]. Available: https://www.infinera.com/wp-content/uploads/Infinera-Transcend-Software-Suite-0162SN-RevA-0519.pdf (pp. xv, 3, 31, and 32)

[2] V. Almeida, "Redes de internet," Instituto Superior de Engenharia de Lisboa, Tech. Rep., 2011. (pp. xv and 9)

[3] D. Pernes, "Aplicações sobre redes definidas por software baseadas em openflow," Instituto Superior de Engenharia de Lisboa, Tech. Rep., 2016. (pp. xv and 10)

[4] sdxcentral. Understanding the sdn architecture – sdn control plane and sdn data plane. (accessed: 15.05.2019). [Online]. Available: https://www.sdxcentral.com/networking/sdn/definitions/inside-sdn-architecture/ (pp. xv and 10)

[5] D. Pernes, "Aplicações sobre redes definidas por software baseadas em openflow," Instituto Superior de Engenharia de Lisboa, Tech. Rep., 2016. (pp. xv and 12)

[6] OWASP, "Owasp top 10 application security risks – 2017," 2017, (accessed: 10.10.2018). [Online]. Available: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf (pp. xv and 23)

[7] T. Hive! Xsser. (accessed: 22.11.2019). [Online]. Available: https://xsser.03c8.net/ (pp. xvi and 51)

[8] A. A. M. J. F. D. B. N. M. Martin Casado, Tal Garfinkel and S. Shenker. Sane: A protection architecture for enterprise networks. (accessed: 06.03.2019). [Online]. Available: http://yuba.stanford.edu/~casado/sane.pdf (p. 7)

[9] J. P. J. L. N. M. Martìn Casado, Michael J. Freedman and S. Shenker. Ethane: Taking control of the enterprise. (accessed: 06.03.2019). [Online]. Available: http://cs.brown.edu/courses/csci2950-u/s14/papers/Casado07Ethane.pdf (p. 7)

[10] Wikipedia. Osi model. (accessed: 10.04.2019). [Online]. Available: https://en.wikipedia.org/wiki/OSI_model (p. 9)

[11] D. L. P. SarwarRaza (HP). Open networking foundation north bound interface working group (nbi-wg) charter. (accessed: 15.05.2019). [Online]. Available: https://www.opennetworking.org/images/stories/downloads/working-groups/charter-nbi.pdf (p. 11)

[12] D. K. Fernando Ramos. Software-defined networking: A comprehensive survey. (accessed: 15.05.2019). [Online]. Available: https://ieeexplore.ieee.org/document/6994333 (p. 11)

[13] M. B. m. Fouad Benamrane and R. Benaini. An east-west interface for distributed sdn controlplane: Implementation and evaluation. (accessed: 15.05.2019). [Online]. Available: https://www.researchgate.net/publication/308856869_An_East-West_interface_for_distributed_SDN_control_plane_Implementation_and_evaluation (p. 11)

[14] S. Testing. Penetration testing - complete guide with sample test cases. (accessed: 11.09.2018). [Online]. Available: https://www.softwaretestinghelp.com/penetration-testing-guide/ (p. 12)

[15] J. Manikandan. Who's an ethical hacker? (accessed: 11.09.2018). [Online]. Available: https://www.simplilearn.com/roles-of-ethical-hacker-article (p. 14)

[16] S. employee. What is the difference between black, white and grey hat hackers? (accessed: 11.09.2018). [Online]. Available: https://us.norton.com/internetsecurity-emerging-threats-what-is-the-difference-between-black-white-and-grey-hat-hackers.html (p. 15)

[17] NMAP. Nmap - introduction. (accessed: 11.09.2018). [Online]. Available: https://nmap.org/ (p. 16)

[18] I. Shakeel. Top ten hacking tools of 2018. (accessed: 11.09.2018). [Online]. Available: https://resources.infosecinstitute.com/top-ten-hacking-tools-of-2016/#gref (p. 16)

[19] Rapid7. Metasploit - getting started. (accessed: 11.09.2018). [Online]. Available: https://metasploit.help.rapid7.com/docs (p. 16)

[20] Aircrack-ng. Aircrack-ng - description. (accessed: 11.09.2018). [Online]. Available: https://www.aircrack-ng.org/ (p. 16)

[21] Bluefoxicy. Fluhrer, mantin and shamir attack. (accessed: 11.09.2018). [Online]. Available: https://en.wikipedia.org/wiki/Fluhrer,_Mantin_and_Shamir_attack (p. 16)

[22] Jeroenimo. Korek chopchop. (accessed: 11.09.2018). [Online]. Available: https://www.aircrack-ng.org/doku.php?id=korek_chopchop&do= (p. 16)

[23] Rapid7. John the ripper password cracker. (accessed: 11.09.2018). [Online]. Available: https://www.openwall.com/john/ (p. 16)

[24] Sectools. Thc hydra. (accessed: 11.09.2018). [Online]. Available: https://sectools.org/tool/hydra/ (p. 17)

[25] L. Bošnjak, J. Sreš, and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 5 2018, pp. 1161–1166. (p. 17)

[26] P. Coimbra. Owasp zed attack proxy project. (accessed: 11.09.2018). [Online]. Available: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project (p. 17)

[27] C. Sullo and D. Lodge. Nikto2. (accessed: 11.09.2018). [Online]. Available: https://cirt.net/Nikto2 (p. 17)

[28] About wireshark. (accessed: 11.09.2018). [Online]. Available: https://www.wireshark.org/#aboutWS (p. 17)

[29] What is maltego ce? (accessed: 11.09.2018). [Online]. Available: https://www.paterva.com/web7/buy/maltego-clients/maltego-ce.php (p. 17)

[30] Industry-defining penetration testing training, certifications and service. (accessed: 11.09.2018). [Online]. Available: https://www.offensive-security.com/ (p. 18)

[31] What is kali linux? (accessed: 11.09.2018). [Online]. Available: https://docs.kali.org/introduction/what-is-kali-linux (p. 18)

[32] Rapid7. Web application security and scanning. (accessed: 11.09.2018). [Online]. Available: https://www.rapid7.com/fundamentals/web-application-security/ (p. 19)

[33] V. Enterprise, "2018 data breach investigations report," Verizon Enterprise, Tech. Rep. 11, 2018. [Online]. Available: https://www.verizonenterprise.com/resources/reports/rp_DBIR_2018_Report_execsummary_en_xg.pdf (pp. 19 and 20)

[34] Rapid7. Web application security testing. (accessed: 11.09.2018). [Online]. Available: https://www.rapid7.com/fundamentals/web-application-security-testing/ (p. 20)

[35] N. Smithline. Top 10-2017 top 10. (accessed: 11.09.2018). [Online]. Available: https://www.owasp.org/index.php?title=Top_10-2017_Top_10&action=history (pp. 20 and 22)

[36] Ids. (accessed: 11.09.2018). [Online]. Available: https://pt.wikipedia.org/wiki/IDS (p. 20)

[37] Application security fallacies realities. (accessed: 11.09.2018). [Online]. Available: https://www.veracode.com/sites/default/files/Resources/Whitepapers/application-security-fallacies-and-realities-veracode.pdf (p. 20)

[38] Veracode, "Application security fallacies realities," Verizon Enterprise, Tech. Rep., 2015. [Online]. Available: https://www.veracode.com/sites/default/files/Resources/Whitepapers/application-security-fallacies-and-realities-veracode.pdf (p. 22)

[39] C. Banse and S. Rangarajan, "A secure northbound interface for sdn applications," 08 2015. (p. 24)

[40] R. Sahay, G. Blanc, Z. Zhang, and H. Debar, "Towards autonomic ddos mitigation using software defined networking," 02 2015. (p. 24)

[41] S. Scott-Hayward, C. Kane, and S. Sezer, "Operationcheckpoint: Sdn application control," in *2014 IEEE 22nd International Conference on Network Protocols*, Oct 2014, pp. 618–623. (p. 24)

[42] H. Hu, W. Han, G.-J. Ahn, and Z. Zhao, "Flowguard: Building robust firewalls for software-defined networks," 08 2014. (p. 24)

[43] I. E. T. Force. Software-defined networking (sdn): Layers and architecture terminology. (accessed: 02.09.2019). [Online]. Available: https://tools.ietf.org/html/rfc7426 (p. 30)

[44] ——. Netconf. (accessed: 02.09.2019). [Online]. Available: https://tools.ietf.org/html/rfc6241 (p. 30)

[45] ——. Yang - a data modeling language for the network configuration protocol (netconf). (accessed: 02.09.2019). [Online]. Available: https://tools.ietf.org/html/rfc6020 (p. 30)

[46] ——. Restconf protocol. (accessed: 02.09.2019). [Online]. Available: https://tools.ietf.org/html/rfc8040 (p. 30)

[47] I. C. Station. Compare portswigger burp vs. veracode. (accessed: 22.11.2019). [Online]. Available: https://www.itcentralstation.com/products/comparisons/portswigger-burp_vs_veracode (p. 35)

[48] TrustRadius. Burp suite vs veracode. (accessed: 22.11.2019). [Online]. Available: https://www.trustradius.com/compare-products/burp-suite-vs-veracode (p. 35)

[49] P. W. Security. Burp suite. (accessed: 02.09.2019). [Online]. Available: https://portswigger.net/ (p. 35)

[50] G. I. Z. H. Roschke. Skipfish package description. (accessed: 03.09.2019). [Online]. Available: https://tools.kali.org/web-applications/skipfish (p. 42)

[51] https://www.seleniumhq.org/about/contributors.jsp. Seleniumhq browser automation. (accessed: 03.09.2019). [Online]. Available: https://www.seleniumhq.org/ (p. 43)

[52] P. W. Security. Using burp to detect sql injection flaws. (accessed: 03.09.2019). [Online]. Available: https://support.portswigger.net/customer/portal/articles/1965677-using-burp-to-detect-sql-injection-flaws (p. 46)

[53] M. S. Bernardo Damele A. G. sqlmap. (accessed: 22.11.2019). [Online]. Available: http://sqlmap.org/ (p. 48)

[54] OWASP. Types of cross-site scripting. (accessed: 03.09.2019). [Online]. Available: https://www.owasp.org/index.php/Types_of_Cross-Site_Scripting (p. 50)

[55] P. W. Security. Burp suite - crawling. (accessed: 02.09.2019). [Online]. Available: https://portswigger.net/burp/documentation/scanner/crawling (p. 54)

[56] (accessed: 03.09.2019). [Online]. Available: http://download.g0tmi1k.com/wordlists/ (p. 67)

[57] P. W. Security. Burp suite - intruder. (accessed: 03.09.2019). [Online]. Available: https://portswigger.net/burp/documentation/desktop/tools/intruder/getting-started (p. 68)