Virtual environment for cybersecurity tests

Marco Edgar Fernandes Afonso
marco.afonso@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

December 2020

Abstract

Cybersecurity is currently one of the most critical problems in the Internet and for that reason requires deep attention. Alongside the advances of technology, vulnerabilities and attacks are discovered on a daily basis. This MSc Dissertation presents a virtual environment based on GNS3, a network emulation tool, that allows performing a large number of cybersecurity experiments, consisting of attacks that affect the Internet and network configurations that resist/mitigate these attacks. More specifically, this dissertation is organized in two main subjects: network attacks and network protection.

Regarding network attacks, this document describes experiments that demonstrate a set of layer-2 attacks and the corresponding countermeasures. We also address three different examples of attacks to the DNS protocol and the implementation of a botnet. Most of the attacks were performed using Kali Linux but we also developed a tool for performing layer-2 attacks, using the Scapy Python library.

Regarding network protection, we address AAA systems based on RADIUS and TACACS+, firewall solutions based on ASA and zone-based policy technologies, including high-availability provisioning, IDSs based on Snort, and IPSec VPNs based on DMVPN and GETVPN.

This MSc dissertation was supported by Instituto de Telecomunicaçôens.

Keywords: Cybersecurity; Layer-2; AAA; Firewall; ASA; Botnet; VPN; IDS; Kali Linux; GNS3; attack; nmap; dSniff; macof; ettercap;yersinia; hping3; dns2tcp; metasploit; scapy;

1. Introduction

Cybersecurity is one of the biggest concerns on the Internet these days because it is what avoids systems and networks from being breached and theft. There are many reasons for a network to get attacked. Security breaches, for example in a company’s network can, for instance, disrupt e-commerce, compromise private information about employees and data. This can result in intellectual property theft and even lead an enterprise to bankruptcy.

To cause a security breach two things are needed. The first thing is an attack vector, which is a way to gain access to a network or a device. The second required thing to cause a security breach is an agent to create and use this attack vector. The agents responsible for performing the attack are called hackers.

With all the tools and mechanisms developed since cybersecurity is a known issue, even an inexperienced attacker can perform an offense capable of taking an entire network down. This type of problems are not admissible in a world where reliance on computer systems, Internet, wireless network, and “smart” devices constantly increases. It is part of engineers’ job to ensure that the systems are as secure as possible against all the adversities.

1.1. Objectives

To configure and test an exhaustive set of countermeasures to prevent existing security threats, it is necessary to have some kind of infrastructure. Thus, this work has the objective to build a virtual environment lab for tests based in GNS3. GNS3 is a network emulation tool that allows creating diverse network topologies and emulating attacks, which precisely fulfills the intended goal. If cybersecurity tests are performed in physical environments, there is a high probability to jeopardize some hosts or even the entire network. For this matter, the virtual environment enables the possibility of exploring configurations and vulnerabilities without concerning about targeted networks.

The developed system covers two main subjects: network attacks and network protection. Regarding network attacks this document describes experiments that demonstrate a set of layer-2 attacks and the corresponding countermeasures. We also address three different examples of attacks to the DNS protocol and the implementation of a botnet. Most of the attacks were performed using Kali Linux but we also developed a tool for
performing layer-2 attacks, using the Scapy Python library.

Regarding network protection, we address AAA systems based on RADIUS and TACACS+, firewall solutions based on ASA and zone-based policy technologies, including high-availability provisioning, IDSs based on Snort, and IPSec VPNs based on DMVPN and GETVPN.

2. Security concepts and techniques
To use security techniques in the best possible way, first, it is crucial to understand them. In this chapter, the concepts and techniques related to the dissertation to be carried out will be explained. The concept set comprises two sections: network attacks and network protection. The section about network attacks describes layer-2 security, DNS attacks and botnets. The section about network protection describes Authentication Authorization Accounting (AAA), firewalls, Intrusion Detection Systems IDS, Intrusion Prevention Systems (IPS), Virtual Private Networks (VPNs) and Kali Linux.

2.1. Network attacks
2.1.1 Layer-2 Security
In the past few years, switching has been playing one of the most important parts in shifting data in a reliable, efficient and secure way across networks. The data-link layer, corresponding to the Layer-2 of the OSI model, provides the means to transfer data between network hosts. A frame is the data unit of bits used in Layer 2. Frames are sent and received from hosts on the same LAN. In terms of security, Layer-2 confers some challenges. It became mandatory to understand and configure the network for these possible threats, that we address in the next sections.

CAM overflow attacks
Switches store the source MAC address of the received packets and the IDs of the ports at which the packets were received in a table called CAM table. This is used to forward packets directly to the intend destinations, avoiding the flooding of packets to all switch ports. The existing problem is that the switch, depending on the available resources (memory space), can hold only a maximum number of MAC addresses in the CAM table. This problem can be exploited by performing a CAM overflow attack, that happens when an attacker connected to one or more ports of a switch performs a DoS attack that attempts to flood the CAM table by injecting thousands, or even millions, of random MAC addresses into the switch, which causes unavailability in the switch due to the memory limit. After that the switch will forward the frames out of all ports behaving as a hub. To prevent CAM this attack we need to limit the number of MAC addresses of the hosts that access the same port of the switch.

DHCP spoofing attacks
The DHCP [5] spoofing attack, a MitM attack, is an attack where a device can act as a fake DHCP server in the network. This is an attack where a device can act as a fake DHCP server in the network. This device can gain access to a victim’s traffic by spoofing responses that would be sent by the authentic DHCP server. To understand how the attack works, we first describe the operation of DHCP. It all starts with an available client host sending a DHCP Discover message to the network to discover a DHCP server. The DHCP server that receives this message sends to the network a DHCP offer message offering an available IP address. After that, the client sends a DHCP request message to lease the IP address. Finally, a DHCP server will answer with a DHCP Ack message to the client, with IP address, subnet mask, and default gateway. Since we have two DHCP servers, a fake one and a valid one, the fake DHCP server will try to win the race against the true one to be the first to assign an IP address, subnet mask and default gateway to the client request. If the fake DHCP server succeeds, the default gateway address of the client host will correspond to the attacker’s IP address. To prevent this attack, the traffic in switch ports needs to be controlled in order to block messages coming from rogue DHCP servers.

DHCP starvation attacks
A DHCP server contains a pool of IP addresses that can assign to clients in the network. These IP addresses are assigned by the server responding to the DHCP Request messages sent by the client. The DHCP starvation attack allows an attacker to continuously send forged DHCP Discover messages to the DHCP server. The server will respond to this messages with DHCP Offer messages and consequently, the attacker will send multiple DHCP Request messages to the server. As the server finishes the messages exchange with DHCP Ack messages, it can no longer provide IP addresses to the legitimate clients in the network, since it already assigned all the available IP addresses in the pool, which causes denial of service on the network. To prevent this attack, we have to limit the number of DHCP packets per second that an interface receives, shutting down the interface if the limit is exceeded.

ARP spoofing attacks
ARP [13] is the protocol that makes an appropriate mapping of a MAC address to an IP address. Network equipment keep an ARP lookup table where all the associations between MAC addresses and IP addresses are stored. When a net-
work device needs to communicate with another, for example by pinging a host, the system will first of all examine this table looking for an entry with the wanted MAC address assigned to an IP address. If the MAC address is already cached, there is no need to use ARP. If the IP address is not associated to a MAC address in the ARP table, the device will try to discover it by sending an ARP request. The machine with the requested IP address will answer with ARP reply that will also contain the MAC address. ARP spoofing is an effective way of sniffing traffic since it is possible to redirect packets from a target client for an attacker by forging ARP replies. This can be done because these forged ARP replies will change the MAC address of the default gateway in the target ARP table, which originates redirection of the target outgoing packets. The way engineered to prevent this attack is to ensure that a switch only relays valid ARP messages. ARP messages will be considered invalid if an intercepted packet has an invalid binding between IP and MAC address.

**STP manipulation attacks**

To increase the availability in Layer 2 we use redundancy to protect the network from having a single failure point. When this redundancy is assured in a network, STP [15] guarantees that loops and duplicated frames do not occur. This is assured by BPDU frames, that dynamically block/unblock redundant paths when there are changes in the network, for example, a failure in a switch. STP designates a single switch as the root bridge and uses it to do all the path calculations.

The arising vulnerability is the STP manipulation attack. Attackers can manipulate STP to perform an attack by spoofing the root bridge and modifying the network’s topology. To accomplish this, the attacker broadcasts STP configuration and topology change BPDU’s to force the spanning tree to recalculate the paths. The modified BPDU sent by the attacker carries a lower bridge priority, attempting to be selected as the root bridge. In case of success, the attacking host becomes the root bridge and get access to traffic that otherwise would not be possible. To mitigate this attack, it is necessary to enforce the placement of the root bridge in the network, preventing that an inappropriate switch becomes the root bridge. We also need to disable ports that receive BPDU trying to modify the spanning-tree.

**DTP attacks**

The DTP attack allows that traffic from one VLAN can be accessed by another VLAN without the need of a router. To perform this, an attacker takes advantage of a vulnerability in the automatic trunking port that some switches contain in their ports. Trunk ports are ports that carry the VLANs traffic accessible by a specific switch, marking frames with a unique tag, to identify to which VLAN the frame belongs. The attacker pretends to be a switch, establishing a trunk link with the victim switch. If it succeeds, the attacker will be able to access all the VLANs on the switch. To prevent DTP attacks from happening, it is necessary to disable the auto-trunking negotiation on ports and manually enable trunk links. Another good practice to mitigate VLAN hopping attacks is to disable unused ports in all the switches on the network.

**VLAN double-tagging attacks**

The VLAN double tagging attack takes advantage of the 802.1Q protocol. This protocol is responsible for marking each packet with a tag, that indicates to which VLAN it belongs. This attack intends to access to a VLAN that should not be accessed. In this attack, the attacker sends a packet with two tags attached to it to a switch. The first tag has to correspond to the native VLAN of the trunk port and the second tag corresponds to the VLAN that the attacker intends to access. When the packet arrives to the first switch, it sees its first tag removed and it is forwarded, because it is in the native VLAN. When it arrives to the second switch the initially hidden tag (by the first one) is now removed, providing access to the VLAN. To prevent this kind of attack, the native VLAN must be configured different from the default (VLAN 1), in order to reduce the attacker possibilities to get the native VLAN number right.

**PVLAN proxy attacks**

Sometimes, hosts connected to the same switch require to keep their traffic hidden from their neighbours. PVLANs assure this by providing isolation between ports in a switch. They restrict traffic at the layer-2. This is possible through the existence of 3 different types of ports in a switch: promiscuous ports, community ports and isolated ports. Promiscuous ports can communicate with all the other ports. Community ports can communicate with the other community ports and with promiscuous ports. Isolated ports can only communicate with promiscuous ports.

Although PVLAN supplements the VLAN technology, it comes with a vulnerability that can be explored. Since an isolated port is inaccessible by any other ports apart from the promiscuous port, an attacker has to use this port to access an isolated one. This is possible through a PVLAN proxy attack where the attacker tries to communicate with the host connected to the isolated port (victim) by sending a packet destined to the victim IP address and the MAC address of the router connected to the promiscuous port. Since the router is connected to the promiscuous port, the switch will forward all the incoming traffic regarding the destina-
tion. To perform this attack the attacker needs to add an entry in its own ARP table, where a match between the victim IP address and the router MAC address is added. To prevent this attack, the network administrator needs to configure an access list in the router that blocks the traffic incoming from the same subnet as the victim.

MAC address spoofing attacks
As already addressed in this document, spoofing attacks occur when an host impersonates another to get access to information that otherwise would not be able to get. Switches keep record of MAC addresses in their CAM tables, by associating each MAC address to a switch port. When a MAC address spoofing attack occurs, the attacker changes his MAC address to the MAC address of the victim. After that, the switch CAM table, changes the association of the victim MAC address from the port where the victim is connected, to the port where the attacker is connected. If the attack is successfully performed, the switch will now redirect the packets destined to the victim, to the attacker. For this attack to happen successfully, the attacker needs to know the MAC address of the victim intended to attack. In order to prevent this kind of attack there are solutions that store multiple MAC address and IP address bindings. If the incoming bindings do not match with the correct stored bindings the packets are discarded.

2.1.2 DNS attacks
In this section we will describe attacks to the DNS protocol, an application layer protocol responsible for the naming system of the resources connected to the Internet. The attacks to the DNS include the manipulation of DNS messages content to redirect the target into malicious domains. There are some works regarding DNS traffic analysis that intend to find out which domains are malicious, such as G. Zhao [17] and Exposure [2]. In the next subsections we approach some examples of DNS attacks.

DNS spoofing
The DNS spoofing attack, consists in the creation of a forged DNS record to redirect the victim into a fake web page. If well succeeded, the victim can insert, for example, login credentials, that can be used by the attacker to steal other accounts of the victim in the Internet. This attack can be deployed in 2 different ways: MitM and Kaminsky. In the MitM way, the attacker redirects the DNS requests sent by the victim to himself, for example, using an ARP spoofing attack and then, sends forged DNS responds, redirecting the victim to a forged web page. Using the Kaminsky method, the attacker introduces a forged DNS record into the DNS server. When the victim sends a request to the DNS server, it will be redirected to a forged web page. To prevent DNS spoofing attacks from happening, a protocol called DNSSEC [1] was developed, which was designed to DNS. DNSSEC adds data authentication and integrity to the DNS.

DMS tunneling
The DNS tunneling attack occurs when an attacker sends data over the DNS messages. This kind of attack takes advantage of security mechanisms, such as Firewalls, that will probably block some traffic, for example HTTP or TCP, but will allow the DNS protocol. These DNS messages, in this case DNS responses, are forged by the attacker, in order to contain the intended traffic that will be sent to the victim.

2.1.3 Botnets
Botnets are significant threats in the Internet, since attackers can infect and control a vast number of hosts, being able to use them in coordinated attacks that can cause serious damage (e.g. DDoS, SPAM). A botnet constitutes a network of bots, which depending on the number of bots, may achieve the power of a supercomputer. The bots contained in a botnet are remotely controlled by an attacker, usually named bot master. When a host is infected by malicious code, it becomes a bot. After that the bot master can manipulate it via C&C (Command and Control), which is a server used to control all the bots in the botnet [6].

2.1.4 Kali Linux
So far, we have already covered multiple defensive mechanisms regarding cybersecurity. To properly test the security in different equipment, we must have a system and suitable tools to deploy various attacks. Kali Linux is an open-source, Debian-based Linux distribution. It contains hundreds of tools capable of multiple tasks such as: sniffing and spoofing, information gathering, vulnerability analysis, wireless attacks, password attacks, web applications, exploitation tools, stress testing, maintaining access, reverse engineering, hardware hacking, reporting tools.

2.2. Network protection
2.2.1 AAA
Access control is very important because it can limit who can access specific resources. AAA is a network security system that permits setting up access control on a network device and tracking user activities. Authentication refers to a single identifying information for each user registered in a system. Authorization comes right after authentication, after a user is authenticated, the system must specify which resources the user can access.
These solutions are needed since Internet worms ingress at the entry and exit points of the network. It is imperative to have favorable detection and prevention systems. IPS solutions are in- 
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-
tion and prevention systems. IPS solutions are in-
creasing attacks it is imperative to have favorable detec-

There are two different AAA implementations: Local AAA and Server-based AAA. Local AAA im-
plementations are satisfactory in small networks. However, these solutions don’t scale well. That is why Server-based AAA implementations ap-
ppeared. They can be used to manage the user and administrative access needs for an entire network in a server. Subsequently, all the devices in the network can refer to this central server. To communicate with AAA servers two protocols may be used: TACACS+ and RADIUS. Each one of them has different strengths and functionalities.

2.2.2 Firewalls

Firewall refers to a mechanism that permits to separate protected areas from non-protected areas, preventing unauthorized users from accessing network resources that they are not allowed to access. Firewalls provide protection by using ACLs, that can be standard, extended, numbered and named. Another technology used for network security is provided by stateful firewalls that use tables to track the real-time state of the created sessions (e.g. TCP or UDP sessions). Stateful firewalls consider the session-oriented nature of network traffic. Nowadays the number of existing types of firewalls is wide, such as stateful, packet filtering, application gateway, proxy, address translation, host-based, transparent, and finally, hybrid firewalls. A proper network design must contain one or more firewalls correctly placed to protect resources, that unauthorized users should not have access, while available resources must be securely accessed by authorized users.

2.2.3 IDS and IPS

So far we already covered enough cybersecurity threats and techniques to understand that security issues cannot be managed by only one application. None of the technologies addressed before are capable to defend a network against Internet worms and viruses. A network must be able to recognize and mitigate worms and virus threats. Intrusion prevention is required throughout the entire network to detect and stop these kind of threats and other types as well, for instance, DoS and DDoS attacks. To defend against fast-moving and evolving attacks it is imperative to have favorable detection and prevention systems. IPS solutions are integrated at the entry and exit points of the network. These solutions are needed since Internet worms and viruses can spread across the world in a matter of minutes.

One of the approaches to prevent worms and viruses from crossing a network is for a network administrator to continuously monitor the network by analyzing log files generated in hosts. This solution is not doable in big networks since manual analysis of log files is a long task.

To improve this method, IDSs were created to monitor the traffic on a network by analyzing a copy of the traffic stream, comparing it to known malicious signatures. This operation is comparable to software that checks for viruses. IDSs operate off-line, which means that they act passively. Despite the traffic being monitored and reported, there are no actions taken on malicious packets. But there is an advantage of using IDS. Since IDSs operate with a copy of the traffic, the packet flows are not negatively affected.

The best solution to this problem is IPS. IPSs are devices that can immediately detect and stop an attack, monitoring traffic of Layer 3 and Layer 4. They analyze packet payloads looking for embedded attacks, which enables them to identify and stop the attacks. This technology use signatures to detect patterns in traffic. A signature is a set of rules defined to detect malicious activity or to gather information. IPSs differ from IDSs since they take immediate action to a malicious packet, while IDSs have a passive behavior. The disadvantage of IPS is that a badly configured IPS can negatively affect packet flows.

2.2.4 VPNs

VPNs are used to create an end-to-end private network connection over third-party networks. They use tunnels to enable remote users to access network resources as if their devices were directly connected to the private network itself. The first VPNs to appear were strictly IP tunnels that did not contemplate authentication or encryption of the data. That is why some modern cryptog- raphic methods are applied to VPNs to establish a secure end-to-end private connections with encrypted data. There are two basic types of VPNs: remote-access and site-to-site. Remote-access are created when VPN information is not initially set up. Site-to-site are created when devices on both sides are aware of the VPN configuration.

In a Site-to-site-VPN, hosts exchange TCP/IP traffic through a VPN gateway which can be a simple router or a firewall. This gateway is in charge of encapsulating and encrypting outbound traffic from a site and sending it through a VPN tunnel over the Internet to a peer VPN gateway located at the other site. When the other VPN gateway receives
the traffic, it strips the headers, decrypts the content, and directions the packet to the target host in the private network. The VPN topologies have not stopped evolving and feature more complex solutions such as DMVPN [3] and GETVPN [9].

3. Experiments
This chapter describes the experiments performed in this dissertation, based on the security mechanisms addressed in the previous chapter. Regarding network attacks, each experiment performed contains a description of the executed attack and how the concerned vulnerabilities were exploited, and if there is the possibility, the countermeasures deployed in order to mitigate these attacks. For network protection, each experiment contains a description of the deployed security infrastructures and where it is possible, the attacks that prove the effectiveness of these mechanisms.

3.1. Network attacks
3.1.1 DHCP spoofing

Description
To perform a DHCP spoofing attack we used the GNS3 network tool, where we set up (i) a Docker [8] container with Kali Linux, acting as the fake DHCP Server, (ii) two Cisco switches, (iii) a Cisco router acting as the real DHCP Server, and (iv) a PC playing the victim part. The network topology is portrayed in figure 1. Before performing the attack, we must configure the real DHCP server, where we assign an IP to the DHCP server interface connected to the switch SW2 (lines 2-4), and we configure the pool of available addresses (lines 6-9). Once this is done, we wait for all the machines to get IP addresses assigned. After that, we can proceed with launching the attack. To perform the attack, in the Kali Linux (fake DHCP Server), we use the ettercap tool [10], a tool that allows performing MitM attacks, which in this particular case is a DHCP Spoofing attack. This fake DHCP Server will try to win a race against the real DHCP Server, by attempting to respond first to the victim’s request message, as represented in figure 2. To send the messages to the victim with the fake addresses, we need to enter the following command: ettercap -T -i eth0 -M dhcp:172.168.1.10/255.255.255.0/4.4.4.4 where the -T flag means that we are using the tool with the text interface, the -i flag indicates the interface to send the fake DHCP response, the -M parameter represents the type of the attack MitM and the remaining parameter stands for the DHCP Spoofing attack with the used address pool, net mask and DNS. As we can see, the fake DHCP server with the IP address 192.168.1.2 outruns the true one, successfully generating the intended fake DHCP responses (figure 3), providing the generated IP address and gateway to the victim (figure 4).

Countermeasures
After performing the attack, we must implement security measures. Facing this problem, Cisco has a feature, called DHCP snooping that can prevent this kind of attack. The DHCP snooping is a technology that permits the switch ports configuration in one of two states: trusted and untrusted. In a trusted case, the switch port can let through DHCP Server will try to win a race against the real DHCP Server, by attempting to respond first to the victim’s request message, as represented in figure 2. To send the messages to the victim with the fake addresses, we need to enter the following command: ettercap -T -i eth0 -M dhcp:172.168.1.10/255.255.255.0/4.4.4.4 where the -T flag means that we are using the tool with the text interface, the -i flag indicates the interface to send the fake DHCP response, the -M parameter represents the type of the attack MitM and the remaining parameter stands for the DHCP Spoofing attack with the used address pool, net mask and DNS. As we can see, the fake DHCP server with the IP address 192.168.1.2 outruns the true one, successfully generating the intended fake DHCP responses (figure 3), providing the generated IP address and gateway to the victim (figure 4).

Figure 1: Topology used in the DHCP Spoofing attack

Figure 2: Ettercap tool performing the attack

Figure 3: DHCP exchanged messages during the attack

Figure 4: Victim with fake IP address, gateway and DNS provided by the attacker
responses; this case is reserved to a trustworthy DHCP Server. In an untrusted state, DHCP responses will not be allowed, to prevent fake DHCP responses from attackers. To facilitate the routers programming, the switch ports don’t need to be individually configured because if a port is not set to be trusted, it is untrusted by default. The commands needed to configure the DHCP Snooping in the switch SW1 are the following:

These configuration considers as trusted, the interface g0/0 (lines 4-5) which is the interface where the DHCP trustworthy responses come from as we can observe in the figure 5. This output is possible to obtain through the following command: show ip dhcp snooping

If we perform the attack again, we are able to see that the attacker can no longer provide the fake DHCP messages to the victim, instead, the legitimate DHCP server provides the IP address and the gateway to the PC (figure 6).

3.1.2 DNS spoofing attack

3.1.3 Description

To perform a DNS spoofing attack we used the GNS3 network tool, where we set up (i) a Docker [8] container with Kali Linux, acting as the attacker (fake DNS server), (ii) a GNS3 appliance with the Mozilla Firefox browser so that we can see the attack happening, playing the Victim role, (iii) a GNS3 appliance acting as a Web server that contains the web page to where the victim will be redirected, (iv) a NAT that allows connection to the Internet, needed to access a Internet web page, and finally (v) a switch that connects all the nodes (figure 7).

In this attack, we will spoof the domain of linkedin.com, redirecting the a request made to linkedin.com to a rogue web page hosted in the web server.

To perform this attack we use the ettercap tool in the Kali Linux, which we already used in the experiment 3.1.1. First we have to do some changes in two configuration files. The first file to modify is the /etc/ettercap/etter.conf [12]. This file determines how ettercap behaves. The variables ec_uid and ec_gid, which specify the privileges of the user and the group ID, both have to be changed to zero (figure 8) and the variables redir_command_on, redir_command_off, redir6_command_on and redir6_command_off have to be uncommented (figure 9). This is needed to allow the redirection of the victim to the wanted DNS server. Despite in this experiment we are only using IPv4, ettercap cannot deploy the attack without uncommenting the IPv6 options and that is the reason why we uncommented the lines regarding IPv6 as well.
Variables redir\_command\_on, redir\_command\_off, redir6\_command\_on and redir6\_command\_off

/etc/ettercap/etter.dns. Here we choose where the spoofed domain will be redirected (figure 10). In this experiment we chose to spoof the linkedin.com domain.

After that we have to set up our Web server. First, we edit the file /var/www/html/index.html in order to configure the look of our web page. Then we launch our web server, with nginx [7], using the following command: service nginx start

Before deploying the attack we can verify that the domain linkedin.com still leads the victim to the supposed IP address (figure 11)

Finally, the attack can be launched, with ettercap using the following command:

\texttt{ettercap -T -i eth0 -M \textasciitilde arp:remote -P dns\_spoof
//192.168.122.1///192.168.122.166/}

where -T means that we are using ettercap with the text interface, -i indicates the interface that sends the attack, -M refers to the type of the attack MitM, the \texttt{arp:remote} appears to make the traffic redirection to the attacker, the -P stands for the ettercap plugins [11] (in this case we will use the \texttt{dns\_spoof} plugin) the IP addresses indicated correspond respectively to the default-gateway of the network and to the IP address of the victim. The ettercap takes action, first performing an ARP poisoning attack, which implicates that the attacker and the victim have to be in the same subnet and then spoofing the domain linkedin.com (figure 12). If the victim tries to access the domain linkedin.com, it will result in a redirection to the created web page (figure 13), so we can conclude that the attack was successfully performed. To better understand the sequence of exchanged messages that make this attack possible, we can observe them in figure 14, the ARP messages that correspond to the ARP spoof that redirects all the traffic to the attacker and then the DNS messages that correspond to the DNS query sent from the victim and to the answers sent by the attacker, faking the location of the domain linkedin.com and redirecting it to the fake web page created.
3.2. Network protection

3.2.1 AAA

To present a solution regarding AAA, we conduct an experiment using the GNS3 network tool, where we set up an AAA Server, two Cisco router clients, one of them using the TACACS+ protocol and the other using RADIUS, and a Cisco router (R1) for test purposes. The topology used in the experiment was the one of figure 15. We configure two different users in the AAA server, that is implemented in a Linux machine that supports RADIUS (using freeradius [4] as the server) and TACACS+, one named admin and the other named guest. The equipment used in our clients support a wide variety of AAA options. For authentication, it is possible to configure a list of methods to use for logins, for PPP, or for ARAP. In our experiment we used login authentication, where our users have “gns3” as the configured password. Regarding authorization, it is possible to configure security policies for each user and to configure the type of commands each user can perform regarding their different privilege levels. In terms of accounting, we can get records of the network sessions established, records of the beginning and the end of a terminal session, or records of the commands performed by the user. In this case, we configured accounting to get the information that both clients send at the beginning of the established session and at the end of it.

Initially we need to configure the AAA server. For RADIUS, we add the intended users in the configuration file located at /etc/freeradius/3.0/users. Where the admin has all the privileges, and the guest has lower privileges, for example, for the guest it will not be possible to configure the router or to visualize the configuration of the router.

For TACACS+, we do the same, with the respective syntax, in the configuration file located at /etc/tacacs+/tac_plus.conf.

Now that we have configured the AAA server we need to launch the services of freeradius and tacacs_plus.

After that we need to configure the routers RADIUSCli and TACACSCli that contain, respectively, the RADIUS client and the TACACS+ client. For the router RADIUSCli we allow telnet sessions to test RADIUS. Regarding authentication we configure the list of logins, prioritising RADIUS authentication to access the console and to enable the router, but allowing local authentication in case of failure. In terms of authorization we also give priority to the security policies implemented in the RADIUS, but allowing local authorization in case of failure. For accounting, we chose to get the records in the beginning and at the end of the session.

For the configuration of the TACACSCli we allow telnet sessions to test TACACS+. Regarding authentication we configure the list of logins, prioritising TACACS+ authentication to access the console and to enable the router, but allowing local authentication in case of failure. In terms of authorization we also give priority to the security policies implemented in the TACACS+, but allowing local authorization in case of failure. For accounting, we chose to get the records in the beginning and at the end of the session.

To see the proper operation of the RADIUS protocol we establish two telnet sessions between R1 and RADIUSCli. In one of the sessions we logged in as the admin. The admin was configured as privilege 15, so it can perform all the possible commands in the router. In the other session we logged in as the guest (figure 17). The guest was configured as privilege 5, so it can only perform basic show commands and cannot change the router configurations. The messages exchanged during a session are shown in figure 18. In the admin session, it is possible to observe the message Access-Request that corresponds to authentication and authorization, that appear together due to the operation of RADIUS (figure 19).

To see the proper operation of the TACACS+ protocol we also establish two telnet sessions between R1 and TACACSCli. In one of the sessions we logged in as the admin (figure 20). As before, the admin was configured as privilege 15, so it can perform all the possible commands in the router. In
3.2 Snort

Regarding the matter of IDSs we perform some experiments with the tool snort [16]. IDSs allow more detailed analysis of the incoming packets than firewalls. The exploration of this tool is divided in two parts, where first we describe the configuration file (snort.conf), where we set up a series of variables that facilitate our network analysis. Then, we perform some attacks and configure some snort rules that work as countermeasures to this set of attacks. Snort analyses incoming packets looking for certain patterns that correspond to the defined rules, being able to detect an attack. This tool can detect several types of attacks such as DoS, buffer overflows, malware, etc. In this experiment we show how Snort is able to detect 3 different DoS attacks, such as TCP SYN flood, ICMP flood and UDP flood. In order to perform these experiments we used the GNS3 network tool, where we set up a Kali Linux docker container acting as the attacker, a Docker container with the snort tool installed and a PC to test the network that we want to defend (figure 23).

Configuration file

The snort configuration file (snort.conf) is usually located in the /etc/snort directory. We can edit this file through any text editor. In this experiment we used the following command: `nano /etc/snort/snort.conf` In this file we can define the network we intend to defend, the external networks...
that may generate traffic that we want to control and a set of different servers that our network can possibly have, such as an HTTP server, a DNS server, a SSH server, etc. (figure 24). We can also define the ports where we run on different servers like, for instance, HTTP or FTP (figure 25).

We also need to define the set of rules that we will include on our IDS, that can use rules previously created by Snort engineers or the rules created by ourselves (figure 26). The Snort rules are located in the directory /etc/snort/rules.

**Rule definition and attacks**

To test the operation of Snort we will define some rules that will detect/prevent a set of DoS attacks. Then we will perform 3 different DoS attacks using the tool hping3 [14], to prove the correct operation of the rules. The first attack that we will perform is the TCP SYN Flood attack, which consists in sending numerous TCP SYN packets to multiple ports of the victim host in order to cause a DoS. The second attack that we will perform is the UDP flood, where the attacker floods multiple ports of the victim with UDP packets to cause DoS. The last attack that we will perform is the ICMP flood where we cause DoS by sending numerous ICMP packets to the victim.

The defined rules must be included in a file located at the directory /etc/snort/rules. In this experiment the rules were included in the file local.rules, using the following command to access it: nano /etc/snort/rules/local.rules The defined Snort rules were the following:

```
alert tcp any any ->$HOME_NET any (msg:“TCP SYN Flood attack”;flow:stateless;flags:S; detection filter:track by dst, count 500, seconds 5; sid:1000005;)
```

```
alert udp any any ->$HOME_NET any (msg:“UDP Flood attack”;detection filter:track by dst, count 500, seconds 5; sid:1000006;)
```

```
alert icmp any any ->$HOME_NET any (msg:“ICMP Flood attack”; detection filter: track by dst, count 500, seconds 5; sid:1000007;)
```

where in the first rule we detect a TCP SYN flood attack (lines 1-3), by considering that the attack is happening if more than 500 TCP connections are established in 5 seconds; in the second rule we detect a UDP flood attack (lines 4-6), by considering that the attack is happening if more than 500 UDP connections are established in 5 seconds; in the third rule we detect an ICMP flood attack (lines 7-8), by considering that the attack is happening if more than 500 ICMP requests arrive at our network in 5 seconds. In this configuration we are only detecting the attacks. In order to prevent it, the only thing necessary is to change the “alert” keyword in the rules to the “drop” keyword. This will make Snort drop the packets instead of only displaying alert messages.

After defining the rules we perform each one of the attacks with the Snort running to observe the detection of the attacks. We can launch Snort with the following command: snort -A console -q -c /etc/snort/snort.conf -i eth1 where we indicate the location of Snort configuration file and the interface to defend.

To perform the TCP SYN flood attack we use hping3 like this: hping3 -S -flood 192.168.2.100 The Snort console displays the detected attack through alert messages (figure 27), detecting that the TCP packets are being sent with a frequency that triggers the defined rule for TCP traffic, which happens during a TCP flood attack.
4. Conclusion

Cybersecurity is a vital issue in today’s Internet, and if not addressed carefully, it can bring disastrous consequences to networks and equipment. Efficiently securing Layer-2 is relevant since the existing vulnerabilities affect some of the most used protocols at that level. Providing AAA is essential as well, because whoever enters in a network must be allowed to do so, have permissions defined and get their actions registered. Firewalls are responsible for inspecting the incoming/outcoming packets of our network, introducing innovative concepts such as DMZ, IDS and IPS work at inspecting the traffic in our network, based on rules defined to categorize the traffic. VPNs, namely those based on IPsec, ensure that the information traveling through networks stays secure. To test and explore possible vulnerabilities, the Kali Linux distribution displays a gathering of tools, to perform attacks of different types in order to accomplish the best security practices.

In this MSc dissertation, we organised the work in two different parts: network attacks and network protection. In network attacks, we performed several experiments regarding Layer-2, namely a CAM overflow attack, a DHCP spoofing, a DHCP starvation, an ARP spoofing attack, a STP manipulation, DTP attack, a VLAN double tagging, a PVLAN proxy and a MAC address spoofing, as well as the corresponding countermeasures. We also performed experiments addressing three DNS attacks, such as DNS spoofing, DNS Kaminsky and DNS tunneling. We created a SSH botnet implementation using Python. And we developed a Layer-2 tool, using Scapy, where we can build packets from scratch, which allows us to deploy some of the Layer-2 attacks early approached.

In network protection we implemented a solution covering the AAA matter. We also developed experiments with two types of firewalls and we compared the difference between these implementations. We developed an experiment with the technology Snort to illustrate the operation of an IDS system. We approached the matter of high availability in firewalls. And lastly, we elaborated a solution with two different Cisco approaches to IPsec based VPNs, namely DMVPN and GETVPN.

The virtual environment developed in this dissertation was shown to be a flexible tool for exploring network security attacks and countermeasures.

References


