

# 数据库系统表相关学习

## 如何利用数据库功能读写文件，需要什么样的条件(权限)

如果涉及到在服务器上写入文件，那么能否成功执行就要看 `secure_file_priv`

- 如果 `secure_file_priv` 为空时，那么对导入和到处就有限制
- 当这个为一个指定的目录时，那么就只能向这个指定目录导入或者导出
- 当这个值被设置成 NULL 时，那么就禁止导入导出

那么我这里的环境是在 ubuntu 下的 mysql，进入 `mysqlmysql -u root -p`

```
mysql> show variables like '%secure%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| require_secure_transport | OFF |
| secure_auth | ON |
| secure_file_priv | /var/lib/mysql-files/ |
+-----+-----+
3 rows in set (0.01 sec)

mysql>
```

看的到 `secure_file_priv` 是被设置了路径的，如果要修改就直接在 `mysql.ini` 中修改，然后重启。

## load data infile

```
load data local infile "/etc/passwd" into table tb;
```

```
mysql> load data local infile "/etc/passwd" into table tb;
Query OK, 44 rows affected, 132 warnings (0.03 sec)
Records: 44 Deleted: 0 Skipped: 0 Warnings: 132

mysql> select * from tb;
+-----+-----+
| id | name | age |
+-----+-----+
| 1 | Gsuhy | 20 |
| 0 | NULL | NULL |
| 0 | NULL | NULL |
| 0 | NULL | NULL |
+-----+-----+
```

这里导进去都是 Null，可能是因为创建表的时候数据类型不对，因为我 id 是 Int 类型，而导进去的 passwd 都是 char 活在 varchar，这里不纠结

## into outfile

outfile和dumpfile有些不同,outfile可以导出多行数据,同时outfile在将数据写到文件里时有特殊的格式转换,而outfile写文件有这么一些前提:

- 目标目录要有可写去拿先
- 当前数据库用户要有FILE权限
- 目标文件不能已经存在(否则写文件的话会报错)
- secure\_file\_priv的值为空
- 路径完整

```
mysql> select * from tb into outfile '/var/lib/mysql-files/1.txt'
-> fields terminated by ','
-> lines terminated by '\n';\
Query OK, 45 rows affected (0.00 sec)

mysql> ]
```

这里我将上面tb表的数据导出来

```
root@gsuhy-virtual-machine:/home/gsuhy# cat /var/lib/mysql-files/1.txt
1,Gsuhy,20
0,\N,\N
0,\N,\N
```

## load\_file

前提条件:

- 当前权限对该文件可读
- 文件在该服务器上
- 路径完整
- 文件大小小于`max\_allowed\_packet`
- 当前数据库用户有FILE权限
- secure\_file\_priv的值为空,如果值为某目录,那么就只能对该目录的文件进行操作

语法: `select load_file('路径')`

```
type help; or \n for help. type \c to clear the current input statement.

mysql> select load_file('/mysql> select load_file('/data/1.txt') as result;
+-----+
| result |
+-----+
| NULL   |
+-----+
1 row in set (0.00 sec)
```

我这里是load\_file一个不存在的txt,所以返回Null,还有一个原因就是我这儿的secure\_file\_priv不是空,而是有路径的,所以我直接load\_file这个路径下的一个txt

```
mysql> select load_file('/var/lib/mysql-files/1.txt') as result;
+-----+
| result |
+-----+
| 1,Gsuhy,20 |
+-----+
```

## dumpfile

dumpfile 只能导出一行数据，并且写文件时保持原数据的格式

```
select * from tb into dumpfile '路径'
```

## 注入查询步骤(库名，表，字段名，内容及用户基本信息)

### 获取库名

```
SELECT schema_name from information_schema.schemata;
```

```
mysql> select schema_name from information_schema.schemata;
+-----+
| schema_name |
+-----+
| information_schema |
| challenges   |
| db_messagebook |
| dvwa         |
| mysql       |
| performance_schema |
| security     |
| test        |
+-----+
8 rows in set (0.00 sec)
```

### 获取表名

```
SELECT table_name from information_schema.tables WHERE
table_schema='dvwa';
```

```
mysql> select table_name from information_schema.tables where table_schema='dvwa';
+-----+
| table_name |
+-----+
| guestbook  |
| users      |
+-----+
2 rows in set (0.00 sec)

mysql>
```

### 获取字段名

```
SELECT column_name from information_schema.columns WHERE
table_schema='dvwa' and table_name='users';
```

```
mysql> select column_name from information_schema.columns where table_schema='dvwa' and table_name='users';
```

column_name
user_id
first_name
last_name
user
password
avatar
last_login
failed_login

## 最后数据内容

```
select user_id,password from dvwa.users;
```

```
mysql> select user_id,password from dvwa.users;
```

user_id	password
1	5f4dcc3b5aa765d61d8327deb882cf99
2	e99a18c428cb38d5f260853678922e03
3	8d3533d75ae2c3966d7e0d4fcc69216b
4	0d107d09f5bbe40cade3de5c71e9e9b7
5	5f4dcc3b5aa765d61d8327deb882cf99

5 rows in set (0.01 sec)

```
mysql> _
```

## 还有一些其他操作

- 获取数据库版本: select version()

```
mysql> select version();
```

version()
5.5.53

1 row in set (0.00 sec)

- 获取当前操作系统: select @@version\_compile\_os

```
mysql> select @@version_compile_os;
```

@@version_compile_os
Win32

1 row in set (0.00 sec)

```
mysql> _
```

- 获取当前用户: select user()

```
mysql> select user();
```

user()
root@127.0.0.1

```
1 row in set (0.00 sec)
```

- 文件保存路径: `select @@datadir;`

```
@@datadir
C:\phpStudy\PHPTutorial\MySQL\data\
1 row in set (0.00 sec)
```

- 数据库开放端口: `select @@port;`

```
mysql> select @@port;
+-----+
| @@port |
+-----+
|    3306 |
+-----+
1 row in set (0.00 sec)
```

## 扩展(查询用户hash, 使用hashcat对获取的hash进行爆破)

先用hash-identifier检测一下hash类型

[illegible]

## 介绍一下hashcat常用选项

- **-a** : 指定要使用的模式
- **-m** : 指定要破解的hash类型所对应的id
- **-o** : 指定破解成功后的hash以及对应的明文密码的存放位置

- `--force`: 忽略破解过程中的警告，跑单条hash可用
- `--username`: 忽略hash中的用户名

因为这里我就只爆破一个hash，所以用`--force`避免报错，然后我自己随便写了一个字典进行爆破，

```
root@kali:~# cat dict.txt
pa
pass
passwd
password
```

`hashcat --force -a 0 -m 0 /root/hash.txt /root/dict.txt`但是，报错了

提示 `This device's constant buffer size is too small`

说虚拟内存太小,但是虚拟机内存已经8G的，google上找了一圈没办法只好重新在win10下搞一波hashcat

<https://hashcat.net/files/hashcat-3.6.0.7z>

解压过后直接cmd进到文件目录，然后 `hashcat64 --force -a 0 -m 0 5f4dcc3b5aa765d61d8327deb882cf99`

```
Session.....: hashcat
Status.....: Running
Hash.Type.....: MD5
Hash.Target.....: 5f4dcc3b5aa765d61d8327deb882cf99
Time.Started.....: Tue Aug 13 17:48:36 2019 (9 secs)
Time.Estimated.....: Tue Aug 13 17:48:45 2019 (0 secs)
Guess.Base.....: Pipe
Speed.Dev.#1.....: 0 H/s (0.00ms)
Speed.Dev.#2.....: 0 H/s (0.00ms)
Speed.Dev.#*.....: 0 H/s
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 0
Rejected.....: 0
Restore.Point.....: 0
Candidates.#1.....: [Copying]
Candidates.#2.....: [Copying]
HWMon.Dev.#1.....: Temp: 54c Util: 20% Core:1493MHz Mem:3504MHz Bus:8
HWMon.Dev.#2.....: N/A

Tracking performance lower than expected? Append -w 2 to the commandline.

Session.....: hashcat
Status.....: Running
Hash.Type.....: MD5
Hash.Target.....: 5f4dcc3b5aa765d61d8327deb882cf99
Time.Started.....: Tue Aug 13 17:48:36 2019 (19 secs)
Time.Estimated.....: Tue Aug 13 17:48:55 2019 (0 secs)
Guess.Base.....: Pipe
Speed.Dev.#1.....: 0 H/s (0.00ms)
Speed.Dev.#2.....: 0 H/s (0.00ms)
Speed.Dev.#*.....: 0 H/s
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 0
Rejected.....: 0
Restore.Point.....: 0
Candidates.#1.....: [Copying]
Candidates.#2.....: [Copying]
HWMon.Dev.#1.....: Temp: 53c Util: 14% Core:1493MHz Mem:3504MHz Bus:8
HWMon.Dev.#2.....: N/A

Session.....: hashcat
Status.....: Running
Hash.Type.....: MD5
Hash.Target.....: 5f4dcc3b5aa765d61d8327deb882cf99
Time.Started.....: Tue Aug 13 17:48:36 2019 (29 secs)
Time.Estimated.....: Tue Aug 13 17:49:05 2019 (0 secs)
Guess.Base.....: Pipe
Speed.Dev.#1.....: 0 H/s (0.00ms)
Speed.Dev.#2.....: 0 H/s (0.00ms)
Speed.Dev.#*.....: 0 H/s
Recovered.....: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 0
Rejected.....: 0
Restore.Point.....: 0
Candidates.#1.....: [Copying]
Candidates.#2.....: [Copying]
HWMon.Dev.#1.....: Temp: 53c Util: 22% Core:696MHz Mem:2504MHz Bus:8
HWMon.Dev.#2.....: N/A

Session.....: hashcat
```

`5f4dcc3b5aa765d61d8327deb882cf99:password`

还有一种凑巧的方式就是用kali里面的 `findmyhash`，是一个爆破的脚本。

## 参考

## hashcat

<http://www.onebug.org/testing/75.html>

<https://klionsec.github.io/2017/04/26/use-hashcat-crack-hash/>

## mysql

<https://www.jb51.net/article/139858.htm>

<https://www.jb51.net/article/158595.htm>