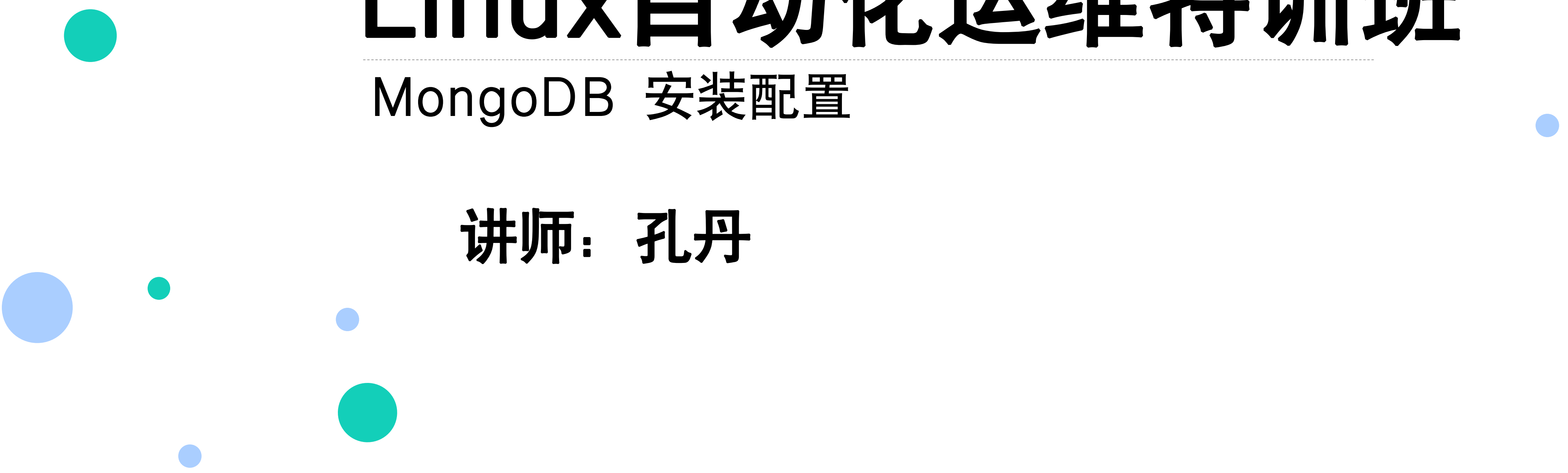




Linux自动化运维特训班

MongoDB 管理

讲师：孔丹



大纲

- 用户管理
- 数据导入导出
- 备份与恢复
- MongoDB监控

MongoDB中用户管理

MMongoDB数据库默认是没有用户名及密码的，即无权限访问限制。为了方便数据库的管理和安全，需创建数据库用户。
用户中权限的说明

权限	说明
Read	允许用户读取指定数据库
readWrite	允许用户读写指定数据库
dbAdmin	允许用户在指定数据库中执行管理函数，如索引创建、删除，查看统计或访问system.profile
userAdmin	允许用户向system.users集合写入，可以找指定数据库里创建、删除和管理用户
clusterAdmin	只在admin数据库中可用，赋予用户所有分片和复制集相关函数的管理权限。
readAnyDatabase	只在admin数据库中可用，赋予用户所有数据库的读权限
readWriteAnyDatabase	只在admin数据库中可用，赋予用户所有数据库的读写权限
userAdminAnyDatabase	只在admin数据库中可用，赋予用户所有数据库的userAdmin权限
dbAdminAnyDatabase	只在admin数据库中可用，赋予用户所有数据库的dbAdmin权限。
root	只在admin数据库中可用。超级账号，超级权限

MongoDB中用户管理

用户创建的语法:

```
{  
  user: "<name>",  
  pwd: "<cleartext password>",  
  customData: { <any information> },  
  roles: [  
    { role: "<role>",  
      db: "<database>" } | "<role>",  
    ...  
  ]  
}
```

语法说明

user字段: 用户的名字;

pwd字段: 用户的密码;

cusomData字段: 为任意内容, 例如可以为用户全名介绍;

roles字段: 指定用户的角色, 可以用一个空数组给新用户设定空角色;

roles 字段: 可以指定内置角色和用户定义的角色

MongoDB中用户管理

创建管理员用户:

进入管理数据库

```
> use admin
```

创建管理用户, root权限

```
db.createUser(  
  {  
    user: "root",  
    pwd: "root",  
    roles: [ { role: "root", db: "admin" } ]  
  }  
)
```

注意:

创建管理员角色用户的时候, 必须到admin下创建。

删除的时候也要到相应的库下操作。

查看创建完用户后的collections;

```
> show tables;
```

查看创建的管理员用户

```
> show users
```

MongoDB中用户管理

创建管理员用户:

验证用户是否能用

```
> db.auth("root","root")
```

用户创建完成后在配置文件中开启用户验证

```
cat >>/etc/mongod.conf<<-'EOF'
```

```
security:
```

```
  authorization: enabled
```

```
EOF
```

重启服务

```
systemctl restart mongod
```

MongoDB中用户管理

创建管理员用户：

登陆测试，注意登陆时选择admin数据库

注意：用户在哪个数据库下创建的，最后加上什么库。

方法一：命令行中进行登陆

```
[mongod@MongoDB ~]$ mongo -uroot -proot admin
```

方法二：在数据库中进行登陆验证：

```
> use admin
```

```
switched to db admin
```

```
> db.auth("root","root")
```


MongoDB中用户管理

按生产需求创建应用用户:

创建对某库的只读用户

在test库创建只读用户test

```
use test
db.createUser(
  {
    user: "test",
    pwd: "test",
    roles: [ { role: "read", db: "test" } ]
  }
)
```

测试用户是否创建成功

```
db.auth("test","test")
show users;
```

登录test用户，并测试是否只读

```
show collections;
db.createCollection('b')
```


MongoDB中用户管理

按生产需求创建应用用户:

创建某库的读写用户

创建test1用户, 权限为读写

```
db.createUser(  
  {  
    user: "test1",  
    pwd: "test1",  
    roles: [ { role: "readWrite", db: "test" } ]  
  }  
)
```

查看并测试用户

```
show users;  
db.auth("test1","test1")
```

MongoDB中用户管理

按生产需求创建应用用户:

创建对多库不同权限的用户

创建对app为读写权限，对test库为只读权限的用户

```
use app
db.createUser(
  {
    user: "app",
    pwd: "app",
    roles: [ { role: "readWrite", db: "app" },
              { role: "read", db: "test" }
            ]
  }
)
```

查看并测试用户

```
show users
db.auth("app","app")
```

MongoDB中用户管理

删除用户:

删除app用户: 先登录到admin数据库

```
mongo -uroot -p123456 127.0.0.1/admin  
use admin  
db.removeUser("app")
```

```
use app  
db.dropUser("app")
```

MongoDB中用户管理

自定义数据库

创建app数据库的管理员：先登录到admin数据库

```
use app
db.createUser(
{
user: "admin",
pwd: "admin",
roles: [ { role: "dbAdmin", db: "app" } ]
}
)
```

创建app数据库读写权限的用户并具有clusterAdmin权限：

```
use app
db.createUser(
{
user: "app04",
pwd: "app04",
roles: [ { role: "readWrite", db: "app" },
{ role: "clusterAdmin", db: "admin" }
]
}
)
```

导出工具mongoexport

Mongodb中的mongoexport工具可以把一个collection导出成JSON格式或CSV格式的文件。可以通过参数指定导出的数据项，也可以根据指定的条件导出数据。
该命令的参数如下：

参数	参数说明
-h	指明数据库宿主机的IP
-u	指明数据库的用户名
-p	指明数据库的密码
-d	指明数据库的名字
-c	指明collection的名字
-f	指明要导出那些列
-o	指明到要导出的文件名
-q	指明导出数据的过滤条件
--type	指定文件类型
--authenticationDatabase	验证数据的名称

mongoexport备份实践

准备备份素材

```
MongoDB Enterprise > use my_mongodb  
> db.createCollection("user")
```

插入数据

```
db.user.insertMany(  
  [{id:201,name:'张三',sex:'男',age:19},  
   {id:202,name:'李四',sex:'女',age:22},  
   {id:203,name:'王五',sex:'男',age:20},  
   {id:204,name:'赵六',sex:'女',age:21},  
   {id:205,name:'钱七',sex:'男',age:20}]  
)
```

备份my_mongodb库下的user集合

```
mongoexport -h 192.168.95.15:27017 --authenticationDatabase  
admin -d my_mongodb -c user -o /home/mongod/backup/user.dat
```

注：备份文件的名称可以自定义，默认导出了JSON格式的数据。

导出CSV格式的数据

```
mongoexport -h 192.168.95.15:27017 --authenticationDatabase  
admin -d my_mongodb -c user --type=csv -f id,name,age -o  
/home/mongod/backup/user_csv.dat
```


导入工具mongoimport

Mongodb中的mongoimport工具可以把一个特定格式文件中的内容导入到指定的collection中。该工具可以导入JSON格式数据，也可以导入CSV格式数据。该命令的参数如下：

参数	参数说明
-h	指明数据库宿主机的IP
-u	指明数据库的用户名
-p	指明数据库的密码
-d	指明数据库的名字
-c	指明collection的名字
-f	指明要导出那些列
-o	指明到要导出的文件名
-q	指明导出数据的过滤条件
--drop	插入之前先删除原有的
--headerline	指明第一行是列名，不需要导入。
-j	同时运行的插入操作数（默认为1），并行
--authenticationDatabase	验证数据的名称

mongoimport恢复实践

将之前恢复的数据导入

```
mongoimport -h 192.168.95.15:27017 --authenticationDatabase  
admin -d my_mongodb -c user --drop  
/home/mongod/backup/user.dat
```

将之前恢复的CSV格式数据导入

```
mongoimport -h 192.168.95.15:27017 --authenticationDatabase  
admin -d my_mongodb -c user --type=csv --headerline --file  
/home/mongod/backup/user_csv.dat
```

mysql数据迁移至mongodb数据库

将mysql（此处使用mysql 5.7.14）数据库中的mysql下的user表导出。

```
select user,host,authentication_string from mysql.user  
into outfile '/var/lib/mysql-files/user.csv'  
fields terminated by ','  
optionally enclosed by '"'  
escaped by '\\'  
lines terminated by '\\r\\n';
```

命令说明:

into outfile '/var/lib/mysql-files/user.csv'	-----导出文件位置
fields terminated by ','	-----字段间以,号分隔
optionally enclosed by '"'	-----字段用"号括起
escaped by '\\'	-----字段中使用的转义符为"
lines terminated by '\\r\\n';	-----行以\\r\\n结束

mysql数据迁移至mongodb数据库

查看导出内容

```
mysql> system cat /var/lib/mysql-files/user.csv
"root","localhost","*6BB4837EB74329105EE4568DDA7DC67ED2CA2A
D9"
"mysql.sys","localhost","*THISISNOTAVALIDPASSWORDTHATCANBEU
SEDHERE"
"tom","192.168.95.*","*12CDA53FDED2778A43788A07273AA206E9039
873"
"admin","%","*12CDA53FDED2778A43788A07273AA206E9039873"
"rep","192.168.95.%","*12CDA53FDED2778A43788A07273AA206E903
9873"
"mycat","192.168.95.%","*12CDA53FDED2778A43788A07273AA206E9
039873"
```

在mongodb中导入数据

```
mongoimport -h 192.168.95.15:27017 --authenticationDatabase
admin -d test -c user -f user,host,password --type=csv --file
/var/lib/mysql-files/user.csv
```

mysql数据迁移至mongodb数据库

在mongodb中查看导入数据

```
MongoDB Enterprise > use test
```

```
switched to db test
```

```
MongoDB Enterprise > db.user.find()
```

```
{ "_id" : ObjectId("5db664fda38d4c52857bb5ea"), "user" : "root",
```

```
"host" : "localhost", "password" :
```

```
"*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9" }
```

```
{ "_id" : ObjectId("5db664fda38d4c52857bb5eb"), "user" :
```

```
"mysql.sys", "host" : "localhost", "password" :
```

```
"*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE" }
```

```
{ "_id" : ObjectId("5db664fda38d4c52857bb5ec"), "user" : "tom",
```

```
"host" : "192.168.95.*", "password" :
```

```
"*12CDA53FDED2778A43788A07273AA206E9039873" }
```

```
{ "_id" : ObjectId("5db664fda38d4c52857bb5ed"), "user" : "admin",
```

```
"host" : "%", "password" :
```

```
"*12CDA53FDED2778A43788A07273AA206E9039873" }
```

```
{ "_id" : ObjectId("5db664fda38d4c52857bb5ee"), "user" : "rep",
```

```
"host" : "192.168.95.%", "password" :
```

```
"*12CDA53FDED2778A43788A07273AA206E9039873" }
```

```
{ "_id" : ObjectId("5db664fda38d4c52857bb5ef"), "user" : "mycat",
```

```
"host" : "192.168.95.%", "password" :
```

```
"*12CDA53FDED2778A43788A07273AA206E9039873" }
```

mongodump备份工具

mongodump的参数与mongoexport的参数基本一致
mongodump备份实践

全库备份

```
mongodump -h 192.168.95.15:27017 --authenticationDatabase admin -  
o /home/mongod/backup/full
```

备份test库

```
mongodump -h 192.168.95.15:27017 --authenticationDatabase admin -  
d test -o /home/mongod/backup/
```

备份test库下的user集合

```
mongodump -h 192.168.95.15:27017 --authenticationDatabase admin -  
d test -c user -o /home/mongod/backup/
```

压缩备份库

```
mongodump -h 192.168.95.15:27017 --authenticationDatabase admin -  
d test -o /home/mongod/backup/ --gzip
```

压缩备份单表

```
mongodump -h 192.168.95.15:27017 --authenticationDatabase admin -  
d test -c user -o /home/mongod/backup/ --gzip
```

mongorestore恢复实践

mongodump的参数与mongoimport的参数基本一致
mongodump备份实践

全库备份中恢复单库（基于之前的全库备份）

```
mongorestore -h 192.168.95.15:27017 --authenticationDatabase admin  
-d test --drop /home/mongod/backup/full/test
```

恢复test库

```
mongorestore -h 192.168.95.15:27017 --authenticationDatabase admin  
-d test /home/mongod/backup/test
```

恢复test库下的user集合

```
mongorestore -h 192.168.95.15:27017 --authenticationDatabase admin  
-d test -c user /home/mongod/backup/test/user.bson
```


mongorestore恢复实践

mongodump的参数与mongoimport的参数基本一致
mongodump备份实践

--drop参数实践恢复

恢复单库

```
mongorestore -h 192.168.95.15:27017 --authenticationDatabase admin -  
d test --drop /home/mongod/backup/test/
```

恢复单表

```
mongorestore -h 192.168.95.15:27017 --authenticationDatabase admin -  
d test -c user --drop /home/mongod/backup/test/user.bson
```


导入导出与备份恢复的对比

1. mongoexport/mongoimport导入/导出的是JSON格式，而 mongodump/mongorestore导入/导出的是BSON格式。
2. JSON可读性强但体积较大，BSON则是二进制文件，体积小但对人类几乎没有可读性。
3. 在一些mongodb版本之间，BSON格式可能会随版本不同而有所不同，所以不同版本之间用mongodump/mongorestore可能不会成功，具体要看版本之间的兼容性。当无法使用BSON进行跨版本的数据迁移的时候，使用JSON格式即mongoexport/mongoimport是一个可选项。跨版本的 mongodump/mongorestore并不推荐，实在要做请先检查文档看两个版本是否兼容（大部分时候是的）。
4. JSON虽然具有较好的跨版本通用性，但其只保留了数据部分，不保留索引，账户等其他基础信息。使用时应该注意。

MongoDB监控

为什么要监控？

> 监控及时获得应用的运行状态信息，在问题出现时及时发现。

监控什么？

> CPU、内存、磁盘I/O、应用程序（MongoDB）、进程监控（ps - aux）、错误日志监控

MongoDB监控

MongoDB集群监控方式

db.serverStatus()

查看实例运行状态（内存使用、锁、用户连接等信息）

通过比对前后快照进行性能分析

"connections"	# 当前连接到本机处于活动状态的连接数
"activeClients"	# 连接到当前实例处于活动状态的客户端数量
"locks"	# 锁相关参数
"opcounters"	# 启动之后的参数
"opcountersRepl"	# 复制想关
"storageEngine"	# 查看数据库的存储引擎
"mem"	# 内存相关

MongoDB监控

MongoDB集群监控方式

状态:

`db.stats()`

显示信息说明:

"db" : "test", 表示当前是针对"test"这个数据库的描述。想要查看其他数据库, 可以先运行 `$ use databasename(e.g $use admin)`.

"collections" : 3, 表示当前数据库有多少个collections. 可以通过运行 `show collections` 查看当前数据库具体有哪些collection.

"objects" : 13, 表示当前数据库所有collection总共有多少行数据。显示的数据是一个估计值, 并不是非常精确。

"avgObjSize" : 36, 表示每行数据是大小, 也是估计值, 单位是bytes

"dataSize" : 468, 表示当前数据库所有数据的总大小, 不是指占有磁盘大小。单位是bytes

"storageSize" : 13312, 表示当前数据库占有磁盘大小, 单位是bytes, 因为mongodb有预分配空间机制, 为了防止当有大量数据插入时对磁盘的压力, 因此会事先多分配磁盘空间。

"numExtents" : 3, 似乎没有什么真实意义。我弄明白之后再详细补充说明。

"indexes" : 1, 表示system.indexes表数据行数。

"indexSize" : 8192, 表示索引占有磁盘大小。单位是bytes

"fileSize" : 201326592, 表示当前数据库预分配的文件大小, 例如test.0, test.1, 不包括test.ns。

MongoDB监控

mongostat

实时数据库状态，读写、加锁、索引命中、缺页中断、读写等待队列等情况。

每秒刷新一次状态值，并能提供良好的可读性，通过这些参数可以观察到MongoDB系统整体性能情况。

```
[mongod@MongoDB oplog]$ mongostat -h 192.168.95.15 --port 28017
insert query update delete getmore command flushes mapped
vsize   res faults qr|qw ar|aw netIn netOut conn set repl
time
    *0    *0    *0    *0      0    1|0      0      303.0M
13.0M    0    0|0    0|0   143b    8k    1      RTR 2018-01-
08T17:28:42+08:00
```

MongoDB监控

mongostat
参数说明:

参数	参数说明
insert	每秒插入量
query	每秒查询量
update	每秒更新量
delete	每秒删除量
conn	当前连接数
qr qw	客户端查询排队长度（读 写）最好为0，如果有堆积，数据库处理慢。
ar aw	活跃客户端数量（读 写）
time	当前时间

MongoDB监控

mongotop命令说明:

```
[mongod@MongoDB oplog]$ mongotop -h 127.0.0.1:27017
```

mongotop重要指标

ns: 数据库命名空间, 后者结合了数据库名称和集合。

total: mongod在这个命令空间上花费的总时间。

read: 在这个命令空间上mongod执行读操作花费的时间。

write: 在这个命名空间上mongod进行写操作花费的时间。

MongoDB监控

db级别命令

db.currentOp()

查看数据库当前执行什么操作。

用于查看长时间运行进程。

通过（执行时长、操作、锁、等待锁时长)等条件过滤。

如果发现一个操作太长，把数据库卡死的话，可以用这个命令杀死他：

> db.killOp(608605)

db.setProfilingLevel()

设置server级别慢日志

打开profiling:

0:不保存

1:保存慢查询日志

2:保存所有查询日志

总结

- 用户管理
- 数据导入导出
- 备份与恢复
- MongoDB监控

作业

□ 1、创建管理员账号admin，密码为admin@163.com

□ 2、素材如下，完成下面备份任务

MongoDB Enterprise > use my_mongodb

> db.createCollection("user")

插入数据

db.user.insertMany(

[{id:201,name:'张三',sex:'男',age:19},

{id:202,name:'李四',sex:'女',age:22},

{id:203,name:'王五',sex:'男',age:20},

{id:204,name:'赵六',sex:'女',age:21},

{id:205,name:'钱七',sex:'男',age:20}]

1) mongoexport备份my_mongodb，格式为csv

2) 使用mongoimport恢复之前备份的my_mongodb

3) mongodump备份my_mongodb库下的user集合

4) 使用mongorestore恢复上一步的备份



谢谢观看

更多好课，请关注[万门大学APP](#)

