Hadoop集群部署权限总结

2014.11.25 | Comments

这是一篇总结的文章，主要介绍 Hadoop 集群快速部署权限的步骤以及一些注意事项。如果你想了解详细的过程，请参考本博客中其他的文章。

1. 开始之前

hadoop 集群一共有三个节点，每个节点的 ip、hostname、角色如下：

```
192.168.56.121 cdh1 NameNode、kerberos-server、ldap-server、sentry-store
192.168.56.122 cdh2 DataNode、yarn、hive、impala
192.168.56.123 cdh3 DataNode、yarn、hive、impala
```

一些注意事项：

- 操作系统为 CentOs6.2
- Hadoop 版本为 CDH5.2
- **hostname 请使用小写**，因为 kerberos 中区分大小写，而 hadoop 中会使用 hostname 的小写替换 `_HOST` ，impala 直接使用 hostname 替换 `_HOST` 。
- 开始之前，请确认 hadoop 集群部署安装成功，不管是否配置 HA，请规划好每个节点的角色。我这里为了简单，以三个节点的集群为例做说明，你可以参考本文并结合你的实际情况做调整。
- 请确认防火墙关闭，以及集群内和 kerberos 以及 ldap 服务器保持**时钟同步**。
- cdh1 为管理节点，故需要做好 cdh1 到集群所有节点的**无密码登陆**，包括其本身。

集群中每个节点的 hosts 如下：

```
$ cat /etc/hosts
127.0.0.1       localhost

192.168.56.121 cdh1
192.168.56.122 cdh2
192.168.56.123 cdh3
```

为了方便管理集群，使用 cdh1 作为管理节点，并在 /opt/shell 目录编写了几脚本，/opt/shell/cmd.sh 用于批量执行命令：

```
$ cat /opt/shell/cmd.sh

#!/bin/sh

for node in 121 122 123;do
        echo "==============="192.168.56.$node"==============="
        ssh 192.168.56.$node $1
done
```

/opt/shell/cmd.sh 用于批量执行命令：

```
$ cat /opt/shell/syn.sh

#!/bin/sh

for node in 121 122 123;do
        echo "==============="192.168.56.$node"==============="
        scp -r $1 192.168.56.$node:$2
done
```

/opt/shell/cluster.sh 用于批量维护集群各个服务：

```
$ cat /opt/shell/cluster.sh
#!/bin/sh
for node in 121 122 123;do
        echo "==============="192.168.56.$node"==============="
        ssh 192.168.56.$node 'for src in `ls /etc/init.d|grep '$1'`;do service $src '$2'; don
e'
done
```

2. 安装 kerberos

在 cdh1 节点修改 /etc/krb5.conf 如下：

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = JAVACHEN.COM
 dns_lookup_realm = false
 dns_lookup_kdc = false
 ticket_lifetime = 24h
 renew_lifetime = 7d
 forwardable = true
 default_tgs_enctypes = aes256-cts-hmac-sha1-96
 default_tkt_enctypes = aes256-cts-hmac-sha1-96
 permitted_enctypes = aes256-cts-hmac-sha1-96
 clockskew = 120
 udp_preference_limit = 1

[realms]
 JAVACHEN.COM = {
  kdc = cdh1
  admin_server = cdh1
 }

[domain_realm]
 .javachen.com = JAVACHEN.COM
 javachen.com = JAVACHEN.COM
```

修改 `/var/kerberos/krb5kdc/kdc.conf` 如下：

```
[kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 JAVACHEN.COM = {
  #master_key_type = aes256-cts
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  max_renewable_life = 7d
  max_life = 1d
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hma
c:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal
  default_principal_flags = +renewable, +forwardable
 }
```

修改 `/var/kerberos/krb5kdc/kadm5.acl` 如下：

```
*/admin@JAVACHEN.COM  *
```

将 `cdh1` 上的 `/etc/krb5.conf` 同步到集群各个节点上：

```
sh /opt/shell/syn.sh /etc/krb5.conf /etc/krb5.conf
```

写了一个脚本安装和初始化 kerberos，供大家参考（详细的脚本，请参考 install_kerberos.sh (https://github.com/javachen/hadoop-install/tree/master/shell/bin/install_kerberos.sh) 和 init_kerberos.sh (https://github.com/javachen/hadoop-install/tree/master/shell/bin/init_kerberos.sh) ）：

```
# install the kerberos components
yum install -y krb5-server
yum install -y openldap-clients
yum install -y krb5-workstation

rm -rf /var/kerberos/krb5kdc/*.keytab /var/kerberos/krb5kdc/prin*

kdb5_util create -r JAVACHEN.COM -s

chkconfig --level 35 krb5kdc on
chkconfig --level 35 kadmin on
service krb5kdc restart
service kadmin restart

echo -e "root\nroot" | kadmin.local -q "addprinc root/admin"

DNS=JAVACHEN.COM
HOSTNAME=`hostname -i`

#读取/etc/host文件中ip为 192.168.56 开头的机器名称并排除自己（kerberos 服务器）
for host in  `cat /etc/hosts|grep 192.168.56|grep -v $HOSTNAME|awk '{print $2}'` ;do
        for user in hdfs; do
                kadmin.local -q "addprinc -randkey $user/$host@$DNS"
                kadmin.local -q "xst -k /var/kerberos/krb5kdc/$user-un.keytab $user/$host@$DN
S"
        done
        for user in HTTP hive yarn mapred impala zookeeper zkcli hbase llama sentry solr hue;
 do
                kadmin.local -q "addprinc -randkey $user/$host@$DNS"
                kadmin.local -q "xst -k /var/kerberos/krb5kdc/$user.keytab $user/$host@$DNS"
        done
done

# 合并
cd /var/kerberos/krb5kdc/
echo -e "rkt hdfs-un.keytab\nrkt HTTP.keytab\nwkt hdfs.keytab" | ktutil

#kerberos 重新初始化之后，还需要添加下面代码用于集成 ldap

kadmin.local -q "addprinc ldapadmin@JAVACHEN.COM"
kadmin.local -q "addprinc -randkey ldap/cdh1@JAVACHEN.COM"
kadmin.local -q "ktadd -k /etc/openldap/ldap.keytab ldap/cdh1@JAVACHEN.COM"

/etc/init.d/slapd restart

#测试 ldap 是否可以正常使用
ldapsearch -x -b 'dc=javachen,dc=com'
```

运行上面的脚本，然后将上面生成的 keytab 同步到其他节点并设置权限：

```
sh /opt/shell/syn.sh /opt/keytab/hdfs.keytab /etc/hadoop/conf/
sh /opt/shell/syn.sh /opt/keytab/mapred.keytab /etc/hadoop/conf/
sh /opt/shell/syn.sh /opt/keytab/yarn.keytab /etc/hadoop/conf/
sh /opt/shell/syn.sh /opt/keytab/hive.keytab /etc/hive/conf/
sh /opt/shell/syn.sh /opt/keytab/impala.keytab /etc/impala/conf/
sh /opt/shell/syn.sh /opt/keytab/zookeeper.keytab /etc/zookeeper/conf/
sh /opt/shell/syn.sh /opt/keytab/zkcli.keytab /etc/zookeeper/conf/
sh /opt/shell/syn.sh /opt/keytab/sentry.keytab /etc/sentry/conf/

sh /opt/shell/cmd.sh "chown hdfs:hadoop /etc/hadoop/conf/hdfs.keytab ;chmod 400 /etc/hadoop/conf/*.keytab"
sh /opt/shell/cmd.sh "chown mapred:hadoop /etc/hadoop/conf/mapred.keytab ;chmod 400 /etc/hadoop/conf/*.keytab"
sh /opt/shell/cmd.sh "chown yarn:hadoop /etc/hadoop/conf/yarn.keytab ;chmod 400 /etc/hadoop/conf/*.keytab"
sh /opt/shell/cmd.sh "chown hive:hadoop /etc/hive/conf/hive.keytab ;chmod 400 /etc/hive/conf/*.keytab"
sh /opt/shell/cmd.sh "chown impala:hadoop /etc/impala/conf/impala.keytab ;chmod 400 /etc/impala/conf/*.keytab"
sh /opt/shell/cmd.sh "chown zookeeper:hadoop /etc/zookeeper/conf/*.keytab ;chmod 400 /etc/zookeeper/conf/*.keytab"

# sentry 只安装在 cdh1 节点
chown sentry:hadoop /etc/sentry/conf/*.keytab ;chmod 400 /etc/sentry/conf/*.keytab
```

在集群中每个节点安装 kerberos 客户端：

```
sh /opt/shell/cmd.sh "yum install krb5-workstation -y"
```

批量获取 root/admin 用户的 ticket

```
sh /opt/shell/cmd.sh "echo root|kinit root/admin"
```

3. hadoop 集成 kerberos

更新每个节点上的 JCE 文件并修改 /etc/default/hadoop-hdfs-datanode，并且修改 hdfs、yarn、mapred、hive 的配置文件。

如果配置了 HA，则先配置 zookeeper 集成 kerberos。

同步配置文件：

```
sh /opt/shell/syn.sh /etc/hadoop/conf /etc/hadoop
sh /opt/shell/syn.sh /etc/zookeeper/conf /etc/zookeeper

sh /opt/shell/cmd.sh "cd /etc/hadoop/conf/; chown root:yarn container-executor.cfg ; chmod 400 container-executor.cfg"

sh /opt/shell/syn.sh /etc/hive/conf /etc/hive
```

接下来就是依次获取每个服务对应的 ticket 并启动对应的服务，我创建了一个脚本 /opt/shell/manager_cluster.sh 来做这件事：

```
#!/bin/bash

role=$1
dir=$role
command=$2

if [ X"$role" == X"hdfs" ];then
        dir=hadoop
fi

if [ X"$role" == X"yarn" ];then
        dir=hadoop
fi

if [ X"$role" == X"mapred" ];then
        dir=hadoop
fi

echo $dir $role $command
for node in 121 122 123 ;do
        echo "========192.168.56.$node========"
        ssh 192.168.56.$node '
                host=`hostname -f| tr "[:upper:]" "[:lower:]"`
                path="'$role'/$host"
                #echo $path
                principal=`klist -k /etc/'$dir'/conf/'$role'.keytab | grep $path | head -n1 |
 cut -d " " -f5`
                echo $principal
                if [ X"$principal" == X ]; then
                        principal=`klist -k /etc/'$dir'/conf/'$role'.keytab | grep $path | he
ad -n1 | cut -d " " -f4`
                        echo $principal
                        if [ X"$principal" == X ]; then
                                        echo "Failed to get hdfs Kerberos principal"
                                        exit 1
                        fi
                fi
                kinit -r 24l -kt /etc/'$dir'/conf/'$role'.keytab $principal
                if [ $? -ne 0 ]; then
                                echo "Failed to login as hdfs by kinit command"
                                exit 1
                fi
                kinit -R
                for src in `ls /etc/init.d|grep '$role'`;do service $src '$command'; done

        '
done
```

启动命令：

```
# 启动 zookeeper
sh /opt/shell/manager_cluster.sh zookeeper restart

# 获取 hdfs 服务的 ticket
sh /opt/shell/manager_cluster.sh hdfs status

# 使用普通脚本依次启动 hadoop-hdfs-zkfc、hadoop-hdfs-journalnode、hadoop-hdfs-namenode、hadoop-hdfs-datanode
sh /opt/shell/cluster.sh hadoop-hdfs-zkfc restart
sh /opt/shell/cluster.sh hadoop-hdfs-journalnode restart
sh /opt/shell/cluster.sh hadoop-hdfs-namenode restart
sh /opt/shell/cluster.sh hadoop-hdfs-datanode restart

sh /opt/shell/manager_cluster.sh yarn restart
sh /opt/shell/manager_cluster.sh mapred restart

sh /opt/shell/manager_cluster.sh hive restart
```

修改 `impala` 配置文件并同步到其他节点，然后启动 `impala` 服务：

```
\cp /etc/hadoop/conf/core-site.xml /etc/impala/conf/
\cp /etc/hadoop/conf/hdfs-site.xml /etc/impala/conf/
\cp /etc/hive/conf/hive-site.xml /etc/impala/conf/

sh /opt/shell/syn.sh /etc/impala/conf /etc/impala/
sh /opt/shell/syn.sh /etc/default/impala /etc/default/impala
sh /opt/shell/manager_cluster.sh impala restart
```

到此，集群应该启动成功了。

3 使用 java 代码测试 kerberos

在 hdfs 中集成 kerberos 之前，可以先使用下面代码(Krb.java)进行测试：

```
import com.sun.security.auth.module.Krb5LoginModule;

import javax.security.auth.Subject;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStream;
import java.util.HashMap;
import java.util.Map;
import java.util.Properties;

public class Krb {
    private void loginImpl(final String propertiesFileName) throws Exception {
        System.out.println("NB: system property to specify the krb5 config: [java.security.kr
b5.conf]");
        //System.setProperty("java.security.krb5.conf", "/etc/krb5.conf");

        System.out.println(System.getProperty("java.version"));

        System.setProperty("sun.security.krb5.debug", "true");

        final Subject subject = new Subject();

        final Krb5LoginModule krb5LoginModule = new Krb5LoginModule();
        final Map<String,String> optionMap = new HashMap<String,String>();

        if (propertiesFileName == null) {
            //optionMap.put("ticketCache", "/tmp/krb5cc_1000");
            optionMap.put("keyTab", "/etc/krb5.keytab");
            optionMap.put("principal", "foo"); // default realm

            optionMap.put("doNotPrompt", "true");
            optionMap.put("refreshKrb5Config", "true");
            optionMap.put("useTicketCache", "true");
            optionMap.put("renewTGT", "true");
            optionMap.put("useKeyTab", "true");
            optionMap.put("storeKey", "true");
            optionMap.put("isInitiator", "true");
        } else {
            File f = new File(propertiesFileName);
            System.out.println("======= loading property file ["+f.getAbsolutePath()+"]");
            Properties p = new Properties();
            InputStream is = new FileInputStream(f);
            try {
                p.load(is);
            } finally {
                is.close();
            }
            optionMap.putAll((Map)p);
        }
        optionMap.put("debug", "true"); // switch on debug of the Java implementation

        krb5LoginModule.initialize(subject, null, new HashMap<String,String>(), optionMap);

        boolean loginOk = krb5LoginModule.login();
        System.out.println("======= login:  " + loginOk);

        boolean commitOk = krb5LoginModule.commit();
        System.out.println("======= commit: " + commitOk);
        System.out.println("======= Subject: " + subject);
    }

    public static void main(String[] args) throws Exception {
        System.out.println("A property file with the login context can be specified as the 1s
t and the only paramater.");
        final Krb krb = new Krb();
        krb.loginImpl(args.length == 0 ? null : args[0]);
    }
}
```

创建一个配置文件krb5.properties：

```
keyTab=/etc/hadoop/conf/hdfs.keytab
principal=hdfs/cdh1@JAVACHEN.COM

doNotPrompt=true
refreshKrb5Config=true
useTicketCache=true
renewTGT=true
useKeyTab=true
storeKey=true
isInitiator=true
```

编译 java 代码并运行：

```
# 先销毁当前 ticket

$ kdestroy

$ javac Krb.java

$ java -cp . Krb ./krb5.properties
```

4. 安装 ldap

使用下面命令在 cdh1 节点快速安装 ldap-server：

```
yum install db4 db4-utils db4-devel cyrus-sasl* krb5-server-ldap -y
yum install openldap openldap-servers openldap-clients openldap-devel compat-openldap -y

# 更新配置库：
rm -rf /var/lib/ldap/*
cp /usr/share/openldap-servers/DB_CONFIG.example /var/lib/ldap/DB_CONFIG
chown -R ldap.ldap /var/lib/ldap

# 备份原来的 slapd-conf
cp -rf /etc/openldap/slapd.d /etc/openldap/slapd.d.bak

cp /usr/share/doc/krb5-server-ldap-1.10.3/kerberos.schema /etc/openldap/schema/
touch /etc/openldap/slapd.conf

echo "include /etc/openldap/schema/corba.schema
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/duaconf.schema
include /etc/openldap/schema/dyngroup.schema
include /etc/openldap/schema/inetorgperson.schema
include /etc/openldap/schema/java.schema
include /etc/openldap/schema/misc.schema
include /etc/openldap/schema/nis.schema
include /etc/openldap/schema/openldap.schema
include /etc/openldap/schema/ppolicy.schema
include /etc/openldap/schema/collective.schema
include /etc/openldap/schema/kerberos.schema" > /etc/openldap/slapd.conf

echo -e "pidfile /var/run/openldap/slapd.pid\nargsfile /var/run/openldap/slapd.args" >> /etc/
openldap/slapd.conf
slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d
chown -R ldap:ldap /etc/openldap/slapd.d && chmod -R 700 /etc/openldap/slapd.d

#重启服务
chkconfig --add slapd
chkconfig --level 345 slapd on

/etc/init.d/slapd restart
```

集成 `kerberos`：

```
# 创建管理员用户
kadmin.local -q "addprinc ldapadmin@JAVACHEN.COM"
kadmin.local -q "addprinc -randkey ldap/cdh1@JAVACHEN.COM"

rm -rf /etc/openldap/ldap.keytab
kadmin.local -q "ktadd -k /etc/openldap/ldap.keytab ldap/cdh1@JAVACHEN.COM"

chown -R ldap:ldap /etc/openldap/ldap.keytab
/etc/init.d/slapd restart
```

创建 `modify.ldif` 文件用于更新数据库：

```
dn: olcDatabase={2}bdb,cn=config
changetype: modify
replace: olcSuffix
olcSuffix: dc=javachen,dc=com

dn: olcDatabase={2}bdb,cn=config
changetype: modify
replace: olcRootDN
# Temporary lines to allow initial setup
olcRootDN: uid=ldapadmin,ou=people,dc=javachen,dc=com

dn: olcDatabase={2}bdb,cn=config
changetype: modify
add: olcRootPW
olcRootPW: secret

dn: cn=config
changetype: modify
add: olcAuthzRegexp
olcAuthzRegexp: uid=([^,]*),cn=GSSAPI,cn=auth uid=$1,ou=people,dc=javachen,dc=com

dn: olcDatabase={2}bdb,cn=config
changetype: modify
add: olcAccess
# Everyone can read everything
olcAccess: {0}to dn.base="" by * read
# The ldapadm dn has full write access
olcAccess: {1}to * by dn="uid=ldapadmin,ou=people,dc=javachen,dc=com" write by * read
```

运行下面命令更新数据库：

```
ldapmodify -Y EXTERNAL -H ldapi:/// -f modify.ldif
```

添加用户和组，创建 `setup.ldif` 如下：

```
dn: dc=javachen,dc=com
objectClass: top
objectClass: dcObject
objectclass: organization
o: javachen com
dc: javachen

dn: ou=people,dc=javachen,dc=com
objectclass: organizationalUnit
ou: people
description: Users

dn: ou=group,dc=javachen,dc=com
objectClass: organizationalUnit
ou: group

dn: uid=ldapadmin,ou=people,dc=javachen,dc=com
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: LDAP admin account
uid: ldapadmin
sn: ldapadmin
uidNumber: 1001
gidNumber: 100
homeDirectory: /home/ldap
loginShell: /bin/bash
```

运行下面命令导入到数据库：

```
ldapadd -x -D "uid=ldapadmin,ou=people,dc=javachen,dc=com" -w secret -f setup.ldif
```

接下来，可以在 `ldap` 服务器上创建一些本地系统用户，然后将这些用户导入到 `ldap` 服务中。

先安装 `migrationtools` 然后修改 `/usr/share/migrationtools/migrate_common.ph` 文件中的 `defalut DNS domain` 和 `defalut base`。

```


# 创建 admin 组
groupadd admin

# 创建 test 和 hive 用户，用于后面测试 sentry
useradd test hive
usermod -G admin test
usermod -G admin hive

# 将关键用户导入到 ldap
grep -E "bi_|hive|test" /etc/passwd  >/opt/passwd.txt
/usr/share/migrationtools/migrate_passwd.pl /opt/passwd.txt /opt/passwd.ldif
ldapadd -x -D "uid=ldapadmin,ou=people,dc=javachen,dc=com" -w secret -f /opt/passwd.ldif

# 将 admin 组导入到 ldap
grep -E "admin" /etc/group  >/opt/group.txt
/usr/share/migrationtools/migrate_group.pl /opt/group.txt /opt/group.ldif
ldapadd -x -D "uid=ldapadmin,ou=people,dc=javachen,dc=com" -w secret -f /opt/group.ldif
```

然后，你可以依次为每个用户设置密码，使用下面命令：

```
ldappasswd -x -D 'uid=ldapadmin,ou=people,dc=javachen,dc=com' -w secret "uid=hive,ou=people,dc=javachen,dc=com" -S
```

另外，这些用户和组都是存在于 ldap 服务器上的，需要将其远程挂载到 hadoop 的每个节点上，否则，你需要在每个节点创建对应的用户和组（目前，测试是这样的）。

6. 集成 sentry

这部分建议使用数据库的方式存储规则，不建议生产环境使用文件保存方式。

详细的配置，请参考 Impala和Hive集成Sentry (/2014/11/14/config-impala-and-hive-with-sentry.html)

通过 beeline 使用 `hive/cdh1@JAVACHEN.COM` 连接 hive-server2 创建一些角色和组：

```
create role admin_role;
GRANT ALL ON SERVER server1 TO ROLE admin_role;
GRANT ROLE admin_role TO GROUP admin;
GRANT ROLE admin_role TO GROUP hive;

create role test_role;
GRANT ALL ON DATABASE testdb TO ROLE test_role;
GRANT ALL ON DATABASE default TO ROLE test_role;
GRANT ROLE test_role TO GROUP test;
```

上面 amdin 和 hive 组具有所有数据库的管理员权限，而 test 组只有 testdb 和 default 库的读写权限。

在 impala-shell 中通过 ldap 的方式传入不同的用户，可以测试读写权限。

7. 如何添加新用户并设置权限？

下面以 test2 账号为例，说明如何添加新的用户并设置访问权限。test2 需要具有以下权限

- dw_default 库：读权限
- dw_user 库 t1表：读权限
- dw_user 库 t2 表：读权限

在 LDAP 服务器上 上添加 LDAP 用户并设置密码，首先添加系统用户：

```
useradd test2
```

然后使用 LDAP 工具将该用户导入到 LDAP：

```
grep -E "test2" /etc/passwd  >/opt/passwd.txt
/usr/share/migrationtools/migrate_passwd.pl /opt/passwd.txt /opt/passwd.ldif
ldapadd -x -D "uid=ldapadmin,ou=people,dc=javachen,dc=com" -w secret -f /opt/passwd.ldif
```

给 test2 用户生成一个随机密码，然后修改 LDAP 中 test2 的密码：

```
ldappasswd -x -D 'uid=ldapadmin,ou=people,dc=javachen,dc=com' -w secret "uid=test2,ou=people,
dc=javachen,dc=com" -S
```

在每台datanode机器上创建 test2 用户和 secure_analyst 分组，test2 属于 secure_analyst 分组：

```
sh /opt/shell/cmd.sh "groupadd secure_analyst ; useradd test2; usermod -G secure_analyst,test
2 test2"
```

在 hive 中创建角色：

运行 `beeline -u "jdbc:hive2://cdh1:10000/default;principal=hive/cdh1@JAVACHEN.COM"`，然后输入下面语句在 sentry 中创建角色和授予权限给 secure_analyst 组：

```
create role dw_default_r;
GRANT SELECT ON DATABASE dw_default TO ROLE dw_default_r;

create role dw_user;
GRANT SELECT ON DATABASE dw_user TO ROLE dw_user_r;

use dw_user;
create role dw_user_secure_r;
GRANT SELECT ON table t1 TO ROLE dw_user_secure_r;
GRANT SELECT ON table t2 TO ROLE dw_user_secure_r;

GRANT ROLE dw_default_r TO GROUP secure_analyst;
GRANT ROLE dw_user_secure_r TO GROUP secure_analyst;
```

然后，需要 impala 刷新元数据，然后进行测试，可能会需要一些时间 impala-catalog 才能刷新过来。

最后进行测试，这部分略。