1. **分布式部署**
2. **环境准备**

操作系统环境以centos7.5为例，节点规划如下：

|  |  |
| --- | --- |
| 节点名 | 部署进程 |
| 192.168.10.1 | NameNode、DataNode、ResourceManager、NodeManager、QuorumPeerMain、statestored、catalogd、Kafka |
| 192.168.10.2 | NameNode、DataNode、ResourceManager、NodeManager、QuorumPeerMain、impalad(如使用rhinosql则进程为rhinosqld)、Kafka |
| 192.168.10.3 | DataNode、NodeManager、QuorumPeerMain、impalad、Kafka |
| 192.168.10.4 | DataNode、NodeManager |

* 1. **修改操作系统参数**
* 为**每台**机器关闭防火墙并设置seleniux策略，以rhino001为例；root用户操作

[root@rhino001]# systemctl stop firewalld.service

[root@rhino001]# systemctl disable firewalld.service

[root@rhino001]# setenforce 0

[root@rhino001]# vim /etc/selinux/config

|  |
| --- |
| # This file controls the state of SELinux on the system.  # SELINUX= can take one of these three values:  # enforcing - SELinux security policy is enforced.  # permissive - SELinux prints warnings instead of enforcing.  # disabled - No SELinux policy is loaded.  SELINUX=disabled  # SELINUXTYPE= can take one of three two values:  # targeted - Targeted processes are protected,  # minimum - Modification of targeted policy. Only selected processes are protected.  # mls - Multi Level Security protection.  SELINUXTYPE=targeted |

* 为**每台**机器修改打开的文件句柄数，以rhino001为例；

[root@rhino001]# vim /etc/security/limits.conf

在文件最下方添加如下内容：

|  |
| --- |
| \* soft nofile 65536  \* hard nofile 65536  \* soft nproc 65536  \* hard nproc 65536  \* soft memlock -1  \* hard memlock -1 |

[root@rhino001]# vim /etc/security/limits.d/20-nproc.conf

修改如下内容

|  |
| --- |
| \* soft nproc 65536 |

[root@rhino001]#vi /etc/pam.d/config-util

在文件最后加入一行

|  |
| --- |
| session required pam\_limits.so |

[root@rhino001]#vi /etc/profile

添加如下两行

ulimit -SHn 65536

ulimit -SHu 65536

[root@rhino001]#vi /etc/sysctl.conf (该配置生效需要重启)

|  |
| --- |
| vm.max\_map\_count=262144 |

也可以通过sysctl -w vm.max\_map\_count=262144使得该配置临时生效。

* 为系统中的一台机器配置时间同步服务，其他机器同步该台机器的时间即可。

|  |
| --- |
| 此处假设192.168.10.1为ntp服务端，192.168.10.2、192.168.10.3、192.168.10.4为ntp客户端需要同步服务端上面的时间。  说明：此项为配置时间同步使用，确保每台机器在同一个时区，本手册统一时区在Asia/Shanghai，通过date命令查看    出现"CST"字样说明时区是在“Asia/Shanghai”，如果显示不是的话，则需要设置“Asia/Shanghai”时区  之后开始做时间同步，X86服务器上配置NTP服务  X86机器（基于CentOS7.5系统）时间同步： root用户操作  一、安装NTP服务之前，首先确保每台机器关闭防火墙及selinux  二、在每台机器上安装ntp服务  1、查看是否已经安装好ntp服务  rpm -q ntp  如果显示已安装则直接进行第二步操作，否则先进行ntp服务的安装  执行yum安装即可：  yum install ntp  2、修改ntp服务端机器（rhino001）上的配置文件  vim /etc/ntp.conf  ###################################################  # permit the source to query or modify the service on this system.  restrict default nomodify notrap nopeer noquery  # Permit all access over the loopback interface. This could  # be tightened as well, but to do so would effect some of  # the administrative functions.  restrict 127.0.0.1  restrict ::1  # Hosts on local network are less restricted.  #restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap  # Use public servers from the pool.ntp.org project.  # Please consider joining the pool (http://www.pool.ntp.org/join.html).  如下部分配置为时钟同步源所在的服务器信息  server 192.168.10.1  #server 0.centos.pool.ntp.org iburst  #server 1.centos.pool.ntp.org iburst  #server 2.centos.pool.ntp.org iburst  #server 3.centos.pool.ntp.org iburst  #broadcast 192.168.1.255 autokey # broadcast server  #broadcastclient # broadcast client  #broadcast 224.0.1.1 autokey # multicast server  #multicastclient 224.0.1.1 # multicast client  #manycastserver 239.255.254.254 # manycast server  #manycastclient 239.255.254.254 autokey # manycast client  # Enable public key cryptography.  #crypto  includefile /etc/ntp/crypto/pw  # Key file containing the keys and key identifiers used when operating  # with symmetric key cryptography.  keys /etc/ntp/keys  # Specify the key identifiers which are trusted.  #trustedkey 4 8 42  # Specify the key identifier to use with the ntpdc utility.  #requestkey 8  # Specify the key identifier to use with the ntpq utility.  #controlkey 8  # Enable writing of statistics records.  #statistics clockstats cryptostats loopstats peerstats  # Disable the monitoring facility to prevent amplification attacks using ntpdc  # monlist command when default restrict does not include the noquery flag. See  # CVE-2013-5211 for more details.  # Note: Monitoring will not be disabled with the limited restriction flag.  disable monitor  ###############################################################  执行以下命令，添加开机ntpd服务自启动并重启ntpd服务  chkconfig --add ntpd //添加ntp服务  chkconfig ntpd on //添加开机自启ntp  service ntpd restart  3、修改客户端机器（192.168.10.2、192.168.10.3、192.168.10.4）上的配置文件  vim /etc/ntp.conf  (1) **注释**如下语句  #server 0.centos.pool.ntp.org iburst  #server 1.centos.pool.ntp.org iburst  #server 2.centos.pool.ntp.org iburst  #server 3.centos.pool.ntp.org iburst  (2) 在注释语句下方**添加**要同步的时间服务器IP地址  server 192.168.10.1  到此，配置文件修改完成  (3) 执行以下命令，添加开机ntpd服务自启动并重启ntpd服务  [root@rhino001 ~]# systemctl restart ntpd.service  [root@rhino001 ~]# systemctl enable ntpd.service |

注意事项客户端时间同步不会立刻生效，有时需等待十几分钟才能同步服务端的时间。

* 重启系统使各项配置生效

[root@rhino001]# reboot

* 1. **为每台机器配置rhino用户**
* 在分布式组网中的**每台**机器上，新建rhino用户并为它设定密码。

[root@rhino001 ~]# useradd -g root -d /home/rhino -m rhino

[root@rhino001 ~]# passwd rhino

* 以rhino用户将相关软件包拷贝到rhino001机器的rhino用户home目录下。
  1. **为每台机器设置hosts**

在分布式组网的所有节点，以root用户登录，配置主机名称，使主机之间可以互相识别。

[root@rhino001 ~]# vim /etc/hosts

|  |
| --- |
| 127.0.0.1 localhost  192.168.10.1 rhino001  192.168.10.2 rhino002  192.168.10.3 rhino003  192.168.10.4 rhino004 |

**注意**：首行必须配置为“127.0.0.1 localhost”，不能使用其他配置。

* 1. **创建主机之间rhino用户的无密码互访**
* 以**rhino用户**登录master节点，创建.ssh目录(如果目录存在则忽略此步，其他机器也需要检查是否存在~/.ssh目录，如果不存在，则创建)

[rhino@rhino001 ~]$ mkdir ~/.ssh

* 输入命令：ssh-keygen -t rsa生成public key(id\_rsa.pub)和private key(id\_rsa)文件。当出现”Enter Passphrase”的提示时直接按回车

[rhino@rhino001 ~]$ ssh-keygen -t rsa

* 将id\_dsa.pub的内容添加到~/.ssh/authorized\_keys中

[rhino@rhino001 ~]$ cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

* 将.ssh目录拷贝到其他机器上

[rhino@rhino001 ~]$ scp ~/.ssh/authorized\_keys rhino@192.168.10.2:~/.ssh/

[rhino@rhino001 ~]$ scp ~/.ssh/authorized\_keys rhino@192.168.10.3:~/.ssh/

[rhino@rhino001 ~]$ scp ~/.ssh/authorized\_keys rhino@192.168.10.4:~/.ssh/

* 在每台机器上修改目录文件权限

[rhino@rhino001 ~]$ chmod 700 ~/.ssh

[rhino@rhino001 ~]$ chmod 600 ~/.ssh/authorized\_keys

* 拷贝完成后在master机器上利用如下命令测试是否不需要密码即可登陆：

[rhino@rhino001 ~]$ ssh rhino002

[rhino@rhino001 ~]$ ssh rhino003

[rhino@rhino001 ~]$ ssh rhino004

* 创建rhino002到rhino001的无密码登陆

在rhino002中输入命令：ssh-keygen -t rsa生成public key(id\_rsa.pub)和private key(id\_rsa)文件。当出现”Enter Passphrase”的提示时直接按回车

[rhino@rhino002 ~]$ ssh-keygen -t rsa

将rhino002公钥拷贝到rhino001上

[rhino@rhino002 ~]$ scp ~/.ssh/id\_rsa.pub rhino001:~/.ssh/rhino002.pub

在rhino001上将rhino002.pub追加到authorized\_keys中

[rhino@rhino001 ~]$ cat ~/.ssh/rhino002.pub >> ~/.ssh/authorized\_keys

在rhino002上测试是否能无密码访问rhino001

[rhino@rhino002 ~]$ ssh rhino001

* 1. **为每台机器安装JDK**
* 在rhino001上以rhino用户安装JDK，安装包名为jdk-8u101-linux-x64.tar.gz

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar -zxvf jdk-8u101-linux-x64.tar.gz

* 配置JAVA\_HOME环境变量

[rhino@rhino001 ~]$ vim ~/.bashrc

|  |
| --- |
| export JAVA\_HOME=/home/rhino/jdk1.8.0\_101  export PATH=$JAVA\_HOME/bin:$PATH |

* 拷贝JDK及环境变量到slave机器

[rhino@rhino001 ~]$ scp -r ~/jdk1.8.0\_101 rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/jdk1.8.0\_101 rhino@rhino003:~

[rhino@rhino001 ~]$ scp -r ~/jdk1.8.0\_101 rhino@rhino004:~

[rhino@rhino001 ~]$ scp -r ~/.bashrc rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/.bashrc rhino@rhino003:~

[rhino@rhino001 ~]$ scp -r ~/.bashrc rhino@rhino004:~

* 使每台机器环境变量生效

[rhino@rhino001 ~]$ source ~/.bashrc

1. **部署元数据MYSQL双主备模式**

**如不需要双主备模式，可参考yarn单机安装手册。一般安装单台mysql即可。**

**（注意：hive\_rhino库依赖于字符集latin1，cloud库需要是utf8字符集。）**

**（**安装MySQL时，需要先将字符集改为latin1，**修改mysql库的字符集：alter database hive\_rhino character set latin1;**然后安装impala，随后启动impala进程会生成hive-rhino数据库（字符集为latin1，表数量为29）。impala进程启动后，需要将字符集改回utf-8，输入**alter database cloud character set utf8;）**

节点规划如下：

|  |  |
| --- | --- |
| 节点名 | 部署进程 |
| 192.168.10.1 | mysqld端口:3306 |
| 192.168.10.2 | mysqld端口:3306 |

* 1. **安装部署**
* 准备工作(很重要)
  + rhinosql依赖stclient中的动态链接库，故先为X86机器安装stclient，首先确认stclient在rhino001、rhino002、rhino003是否存在同名目录，如果存在，执行如下命令：

rm –rf stclient

* + 在rhino001上解压，然后分别拷贝到其他三台机器上。

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar -zxvf stclient-2.0.1-linux-x86-64.tar.gz

[rhino@rhino001 ~]$cd ~

[rhino@rhino001 ~]$ scp -r stclient rhino002:~

[rhino@rhino001 ~]$ scp -r stclient rhino003:~

* + 确认在rhino001、rhino002是否存在与mysql-5.6.25-linux-x86\_64同名目录，如果存在，执行如下命令：

rm –rf mysql-5.6.25-linux-x86\_64

* 在rhino001获取mysql安装包并将安装包拷贝到rhino002机器上

[rhino@rhino001 ~]$ tar -zxvf mysql-5.6.25-linux-x86\_64.tar.gz

[rhino@rhino001 ~]$scp -r ~/mysql-5.6.25-linux-x86\_64 rhino@rhino002:~

* 以rhino用户分别为两台机器执行如下命令

[rhino@rhino001 ~]$

mkdir -p /home/rhino/mysql-5.6.25-linux-x86\_64/mydata/lock/subsys

mkdir -p /home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp

* 分别为两台机器修改配置文件

[rhino@rhino001 ~]$

vim /home/rhino/mysql-5.6.25-linux-x86\_64/my.cnf

|  |
| --- |
| [client]  port=3306  default-character-set=utf8  [mysqld]  join\_buffer\_size=16M  sort\_buffer\_size=16M  read\_rnd\_buffer\_size=16M  port=3306  socket= /home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock  #character-set-server=latin1  character-set-server=utf8  basedir=/home/rhino/mysql-5.6.25-linux-x86\_64  datadir=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata  explicit\_defaults\_for\_timestamp=true  binlog\_format=MIXED  log-bin=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/master-bin  server\_id=131 #两台主备mysql id需要不一样,在一个网段内如果有其他mysql做主备，那么它们的id值都不能相同。  auto\_increment\_increment=1  auto\_increment\_offset=1  #binlog-do-db=test  #binlog-ignore-db=mysql  #replicate-do-db=test  #replicate-ignore-db=mysql  log-slave-updates  slave-skip-errors=all  sql\_mode=NO\_ENGINE\_SUBSTITUTION,STRICT\_TRANS\_TABLES |

[rhino@rhino001 ~]$

vim /home/rhino/mysql-5.6.25-linux-x86\_64/support-files/mysql.server

修改其中的basedir 、datadir变量、lockdir 变量。

在变量basedir 、datadir、lockdir**第一次出现**的位置修改：

|  |
| --- |
| basedir=/home/rhino/mysql-5.6.25-linux-x86\_64/  datadir=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata  lockdir='/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/lock/subsys' |

* 1. **初始化及启动mysql数据库**
* 为两台机器分别用mysql用户初始化数据库

[rhino@rhino001 ~]$ cd /home/rhino/mysql-5.6.25-linux-x86\_64

[rhino@rhino001 ~]$

scripts/mysql\_install\_db --defaults-file=/home/rhino/mysql-5.6.25-linux-x86\_64/my.cnf --datadir=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata --basedir=/home/rhino/mysql-5.6.25-linux-x86\_64

* 为两台机器上启动mysql

[rhino@rhino001 ~]$

nohup bin/mysqld\_safe --defaults-file=/home/rhino/mysql-5.6.25-linux-x86\_64/my.cnf --user=rhino &

* 为两台机器上设置root密码

[rhino@rhino001 ~]$ bin/mysqladmin --socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock -P3306 -uroot password

输入root用户密码并确认；

* 进入mysql shell

[rhino@rhino001 ~]$

bin/mysql --socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock -P3306 -uroot -p

输入上一步修改完成后的root用户的密码

* *停止mysql命令(需要停止时执行)*

*[rhino@rhino001~]$ bin/mysqladmin --socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock -P3306 -uroot -p shutdown*

* 1. **登陆mysql设置双主备模式**
* 登陆rhino001、rhino002 的mysql，假设root密码为root

[rhino@rhino001 ~] cd mysql-5.6.25-linux-x86\_64

[rhino@rhino001~]$ bin/mysql --socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock -P3306 -uroot -proot

注：如果用户密码不是root，请将上述语句中红色root改成2.2节中设置的实际的root用户密码

* 显示rhino001状态

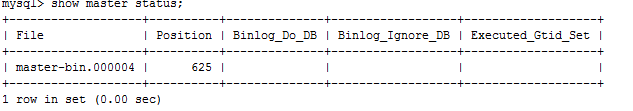
mysql>GRANT REPLICATION SLAVE ON \*.\* TO 'backup'@'192.168.10.2' IDENTIFIED BY 'backup';

mysql> GRANT REPLICATION SLAVE ON \*.\* TO 'root'@'\*' IDENTIFIED BY 'root';

mysql> FLUSH PRIVILEGES;

mysql> show master status;

下图为rhino001的状态:



* 显示rhino002状态

以rhino用户登录到rhino002服务器上

[rhino@rhino002 ~] cd mysql-5.6.25-linux-x86\_64

[rhino@rhino002 ~] bin/mysql --socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock -P3306 -uroot -proot

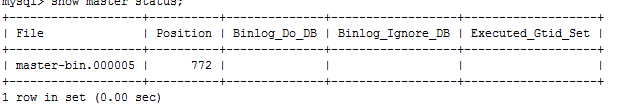
mysql>GRANT REPLICATION SLAVE ON \*.\* TO 'backup'@'192.168.10.1' IDENTIFIED BY 'backup';

mysql> GRANT REPLICATION SLAVE ON \*.\* TO 'root'@'\*' IDENTIFIED BY 'root';

mysql> FLUSH PRIVILEGES;

mysql> show master status;

下图为rhino002的状态:



* 为rhino001设置主备模式

mysql>

CHANGE MASTER TO MASTER\_HOST='192.168.10.2', MASTER\_USER='backup',MASTER\_PASSWORD='backup', MASTER\_LOG\_FILE='master-bin.000005',MASTER\_LOG\_POS=772;

|  |
| --- |
| 注：此处MASTER\_LOG\_FILE和MASTER\_LOG\_POS的值都是从rhino002的状态中获取。 |

mysql> start slave;

* 为rhino002设置主备模式

mysql>

CHANGE MASTER TO MASTER\_HOST='192.168.10.1', MASTER\_USER='backup',MASTER\_PASSWORD='backup', MASTER\_LOG\_FILE='master-bin.000004',MASTER\_LOG\_POS=625;

|  |
| --- |
| 注：此处MASTER\_LOG\_FILE和MASTER\_LOG\_POS的值都是从rhino001的状态中获取。 |

mysql> start slave;

* 1. **验证mysql双主备模式**
* 在rhino001上创建数据库及表

mysql> create database mysqltest;use mysqltest;

mysql> create table testtab(id int);insert into testtab values (1), (2);

* 在rhino002上查看数据库及表是否已经同步

mysql> show databases;use mysqltest;

mysql> show tables;select \* from testtab;

* 在rhino002上删除mysqltest数据库

mysql> drop database mysqltest;

* 查看rhino001上mysqltest数据库是否也已经删除成功

mysql> show databases;

* 1. **添加rhino用户**
* 登陆rhino001的mysql并为其创建rhino用户、登录密码及设置权限

注：此处的用户名及密码应与RHINOSQL 配置文件hive-site.xml中配置的相同。

mysql> GRANT ALL PRIVILEGES ON \*.\* TO 'rhino'@'%' IDENTIFIED BY 'rhino' WITH GRANT OPTION;

mysql>GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY 'root' WITH GRANT OPTION;

mysql> delete from mysql.user where user='';

mysql> flush privileges;

* 验证能否用新建的rhino用户登陆

[rhino@rhino001~]$/home/rhino/mysql-5.6.25-linux-x86\_64/bin/mysql --socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock -urhino -prhino

* 1. **添加mysql开机自启(两个都要)**

切换至**root用户**

[root@rhino001 ~]# vim /etc/rc.d/rc.local

添加

|  |
| --- |
| su - rhino -c "cd /home/rhino/mysql-5.6.25-linux-x86\_64; bin/mysqld\_safe --defaults-file=/home/rhino/mysql-5.6.25-linux-x86\_64/my.cnf --user=rhino &" |

1. **部署元数据Mysql双机切换**

元数据mysql需要做双机切换（根据局点具体要求而定），从而当主mysql机器宕机后可以立即切换到备份的mysql机器上。该章节以元数据mysql为例介绍了如何通过keepalived进程进行双机切换。最终mysql客户端连接的是虚拟出来的ip地址。

节点规划如下：

|  |  |  |
| --- | --- | --- |
| 节点名 | 部署进程 | 虚拟后的ip地址 |
| 192.168.10.1 | keepalived (master) | 192.168.10.49 |
| 192.168.10.2 | keepalived (slave) |

* 1. **安装部署**
* rhino001以root用户拷贝安装包

[root@rhino001 ~]#

cp /home/rhino/DSE-v2.1.1/keepalived-1.1.20.tar.gz /root

scp -r /home/rhino/DSE-v2.1.1/keepalived-1.1.20.tar.gz root@rhino002:~

分别在rhino001和rhino002上执行如下操作

[root@rhino001 ~]# cd /root; tar -zxvf keepalived-1.1.20.tar.gz

[root@rhino001 ~]#

cd keepalived-1.1.20;

chmod 700 sbin/keepalived;

chmod 755 etc/rc.d/init.d/keepalived;

chmod 755 etc/sysconfig/keepalived;

chmod 644 etc/keepalived/keepalived.conf;

chmod 644 etc/keepalived/samples/\*;

chmod 644 share/man/man5;

chmod 644 share/man/man8;

chmod 755 bin/genhash;

chmod 644 share/man/man1;

[root@rhino001 ~]# cp etc/rc.d/init.d/keepalived /etc/init.d/keepalived

[root@rhino001 ~]# cp sbin/keepalived /usr/sbin/

[root@rhino001 ~]# cp etc/sysconfig/keepalived /etc/sysconfig/

[root@rhino001 ~]# mkdir -p /etc/keepalived/

[root@rhino001 ~]# cp etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf

* 为rhino001和rhino002服务器keepalived进程加入开机自启

[root@rhino001 ~]# chkconfig --add keepalived

[root@rhino001 ~]# chkconfig keepalived on

* 1. **配置mysql双机切换**
* 配置主服务器rhino001

[root@rhino001 ~]# vim /etc/keepalived/keepalived.conf

|  |
| --- |
| #! Configuration File for keepalived  #check\_all puts all check in one process: including check jboss, vip, mysql, timeout, status  vrrp\_script chk\_all {  script "/etc/keepalived/check\_mysql.sh" # connects and exits  interval 2 # check every second  weight -2 # default prio: -2 if connect fails  }  global\_defs {  router\_id CM-HA  }  vrrp\_instance VI\_1 {  #state MASTER  state BACKUP  nopreempt  #通过ifconfig查看可以绑定的网卡名，是eth0还是eth1要根据实际来选择相应的网卡名称，下面同理  interface eth1  #同一网段内如果有另外一组keepalived进程，那么virtual\_router\_id值不能相同，同一组keepalived进程virtual\_router\_id相同  virtual\_router\_id 13  priority 100  advert\_int 1  authentication {  auth\_type PASS  auth\_pass 8888  }  virtual\_ipaddress {  192.168.10.49/24 dev eth1 brd 192.168.10.255 label eth1:1  }  track\_interface {  eth1  }  # notify\_master "/etc/keepalived/notify.sh master"  # notify\_backup "/etc/keepalived/notify.sh slave"  # notify\_fault "/etc/keepalived/notify.sh fault"  # notify\_stop "/etc/keepalived/notify.sh stop"  track\_script {  chk\_all  }  } |

* 配置备服务器rhino002

[root@rhino002 ~]# vim /etc/keepalived/keepalived.conf

|  |
| --- |
| #! Configuration File for keepalived  #check\_all puts all check in one process: including check jboss, vip, mysql, timeout, status  vrrp\_script chk\_all {  script "/etc/keepalived/check\_mysql.sh" # connects and exits  interval 2 # check every second  weight -2 # default prio: -2 if connect fails  }  global\_defs {  router\_id CM-HA  }  vrrp\_instance VI\_1 {  #state MASTER  state BACKUP  nopreempt  #通过ifconfig查看可以绑定的网卡名  interface eth1  #同一网段内如果有另外一组keepalived进程，那么virtual\_router\_id值不能相同，同一组keepalived进程virtual\_router\_id相同  virtual\_router\_id 13  priority 90  advert\_int 1  authentication {  auth\_type PASS  auth\_pass 8888  }  virtual\_ipaddress {  192.168.10.49/24 dev eth1 brd 192.168.10.255 label eth1:1  }  track\_interface {  eth1  }  #notify\_master "/etc/keepalived/notify.sh master"  #notify\_backup "/etc/keepalived/notify.sh slave"  #notify\_fault "/etc/keepalived/notify.sh fault"  #notify\_stop "/etc/keepalived/notify.sh stop"  track\_script {  chk\_all  }  } |

* 配置检测脚本

分别为主备服务器配置检测脚本

若需要检测haproxy，需要添加其中红色注释的部分，并按照注释修改

[root@rhino001 ~]# vim /etc/keepalived/check\_mysql.sh

|  |
| --- |
| #!/bin/bash  MYSQL="/home/rhino/mysql-5.6.25-linux-x86\_64/bin/mysql--socket=/home/rhino/mysql-5.6.25-linux-x86\_64/mydata/tmp/mysql.sock"  MYSQL\_HOST=localhost  MYSQL\_USER=root # mysql用户名  MYSQL\_PASSWORD=root # mysql密码  CHECK\_TIME=2  #mysql is working MYSQL\_OK is 1 , mysql down MYSQL\_OK is 0  MYSQL\_OK=1  #检测haproxy时定义haproxy状态  #HAPROXY\_OK=1  function check\_mysql\_helth (){  $MYSQL -h $MYSQL\_HOST -u $MYSQL\_USER -p${MYSQL\_PASSWORD} -e "show status;">/dev/null 2>&1  if [ $? -eq 0 ] ;then  MYSQL\_OK=1  else  MYSQL\_OK=0  fi  return $MYSQL\_OK  }  #检测haproxy状态的函数  #function check\_haproxy\_helth (){  # haproxyprocessnum=`ps -ef | grep haproxy | grep -v grep | awk '{print $2}' | wc -l`  # if [ $haproxyprocessnum -eq 0 ] ;then  # HAPROXY\_OK=0  # else  # HAPROXY\_OK=1  # fi  # return $HAPROXY\_OK  #}  while [ $CHECK\_TIME -ne 0 ]  do  let "CHECK\_TIME -= 1"  check\_mysql\_helth  if [ $MYSQL\_OK -eq 1 ] ; then  CHECK\_TIME=0  exit 0  fi  if [ $MYSQL\_OK -eq 0 ] && [ $CHECK\_TIME -eq 0 ]  then  /etc/init.d/keepalived stop  sleep 1  #此处可以添加需要拉起的进程，根据实际需求来定  /etc/init.d/keepalived start  exit 1  fi  sleep 1  done  #检测haproxy时上面的while…done部分替换为以下注释内容  #while [ $CHECK\_TIME -ne 0 ]  #do  # let "CHECK\_TIME -= 1"  # check\_mysql\_helth  # check\_haproxy\_helth  # if [ $MYSQL\_OK -eq 1 ] && [ $HAPROXY\_OK -eq 1 ]; then  # CHECK\_TIME=0  # exit 0  # fi  # if [ $MYSQL\_OK -eq 0 ] || [ $HAPROXY\_OK -eq 0 ] && [ $CHECK\_TIME -eq 0 ]  # then  # /etc/init.d/keepalived stop  # sleep 1  # /etc/init.d/keepalived start  # exit 1  # fi  # sleep 1  #done |

* 为主备服务器检测脚本添加执行权限

[root@rhino001 ~]# chmod u+x /etc/keepalived/check\_mysql.sh

* 1. **启停keepalived进程**
* 主备服务器启动keepalived

[root@rhino001 ~]# service keepalived start

* 主备服务器停止keepalived(需要停止时执行)

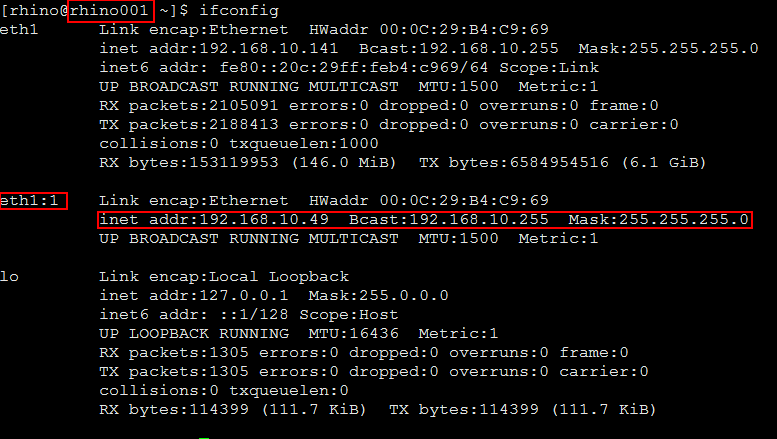
[root@rhino001 ~]# service keepalived stop

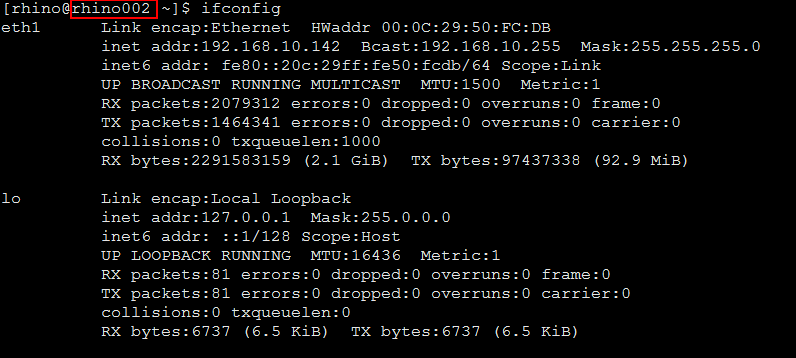
注：集群正常运行时不能关闭keepalived进程，若要切换虚拟IP，应该重启mysql

* 1. **验证双机切换**



主备服务器启动后，可以通过ifconfig命令来查看网卡的状态，其中有一台服务器包含一个虚拟网卡，另外一台机器没有虚拟网卡，如下图所示。





当将rhino001上的mysql停止时，虚拟网卡应该切换到rhino002上。

1. **部署HDFS集群**
   1. **安装Zookeeper**

* 以**rhino用户**在rhino001节点解压zookeeper-3.4.5-cdh5.7.0.tar.gz

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar -zxvf zookeeper-3.4.5-cdh5.7.0.tar.gz

* 修改zookeeper配置文件

[rhino@rhino001 ~]$ cd ~/zookeeper-3.4.5-cdh5.7.0/conf/

[rhino@rhino001 ~]$ mv zoo\_sample.cfg zoo.cfg

[rhino@rhino001 ~]$ vim zoo.cfg

|  |
| --- |
| tickTime=2000  initLimit=10  syncLimit=5  dataDir=/home/rhino/zookeeper-3.4.5-cdh5.7.0/data  clientPort=2181  server.1=rhino001:2888:3888  server.2=rhino002:2888:3888  server.3=rhino003:2888:3888 |

注：如果机器不够只部署一台zookeeper，则不需要最后三行的配置，则将此三行前面加#，注释掉即可。配置完成后直接启动该台zookeeper即可。通过zkServer.sh status命令显示处于standalone状态。

* 创建数据目录并创建myid文件

[rhino@rhino001 ~]$ mkdir /home/rhino/zookeeper-3.4.5-cdh5.7.0/data

[rhino@rhino001 ~]$ cd /home/rhino/zookeeper-3.4.5-cdh5.7.0/data

[rhino@rhino001 ~]$ touch myid; echo 1 > myid;

* 将zookeeper目录拷贝到其他两台机器

[rhino@rhino001 ~]$ scp -r ~/zookeeper-3.4.5-cdh5.7.0 rhino002:~

[rhino@rhino001 ~]$ scp -r ~/zookeeper-3.4.5-cdh5.7.0 rhino003:~

* 修改rhino002中的myid值

[rhino@rhino002 ~]$ echo 2 > /home/rhino/zookeeper-3.4.5-cdh5.7.0/data/myid

* 修改rhino003中的myid值

[rhino@rhino003 ~]$ echo 3 > /home/rhino/zookeeper-3.4.5-cdh5.7.0/data/myid

* 用此命令之前，在.bashrc中添加zookeeper环境变量

[rhino@rhino001 ~] vim .bashrc

export ZOOKEEPER\_HOME=/home/rhino/zookeeper-3.4.5-cdh5.7.0

export PATH =$JAVA\_HOME/bin:$ZOOKEEPER\_HOME/bin:$PATH，然后source .bashrc使环境变量生效，再使用该命令或切换到zookeeper对应路径下，./zkServer.sh 直接执行

* 分别在三台机器上启动zookeeper

[rhino@rhino001 ~]$ zkServer.sh start

* 分别在三台机器上查看zookeeper状态

[rhino@rhino001 ~]$ zkServer.sh status

如果有一台显示leader，两台显示follower则表示zookeeper集群启动成功。

* zookeeper停止命令（需要时停止）

[rhino@rhino001 ~]$ zkServer.sh stop

* 1. **安装HDFS**
* 以rhino用户在rhino001节点解压Hadoop安装包(hadoop-2.6.0-cdh5.7.0.tar.gz)

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar zxvf hadoop-2.6.0-cdh5.7.0.tar.gz

* + 1. **配置slaves节点**

[rhino@rhino001 ~]$ cd /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/

[rhino@rhino001 hadoop]$ vim slaves

|  |
| --- |
| rhino001 #注：slave的主机名  rhino002 #注：slave的主机名  rhino003 #注：slave的主机名  rhino004 #注：slave的主机名 |

注：slaves的主机配置与规划的DataNode主机名一致。

* + 1. **配置core-site.xml**

[rhino@rhino001 ~]$ cd /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/

[rhino@rhino001 hadoop]$ vim core-site.xml

|  |
| --- |
| <configuration>  <property>  <name>fs.defaultFS</name>  <value>hdfs://sinovatiocluster</value>  </property>  <property>  <name>ha.zookeeper.quorum</name>  <value>rhino001:2181,rhino002:2181,rhino003:2181</value>  </property>  <property>>  <name>hadoop.tmp.dir</name>  <value>file:/home/rhino/tmp</value>  </property>  <property>  <name>hadoop.proxyuser.hduser.hosts</name>  <value>\*</value>  </property>  <property>  <name>hadoop.proxyuser.hduser.groups</name>  <value>\*</value>  </property>  <property>  <name>fs.trash.interval</name>  <value>60</value>  </property>  <property>  <name>io.file.buffer.size</name>  <value>131072</value>  </property>  </configuration> |

* + 1. **配置hdfs-site.xml**

[rhino@rhino001 ~]$ cd /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/

[rhino@rhino001 hadoop]$ vim hdfs-site.xml

|  |
| --- |
| <configuration>  <property>  <name>dfs.client.read.shortcircuit</name>  <value>true</value>  </property>  <property>  <name>dfs.domain.socket.path</name>  <value>/home/rhino/hdfs-sockets/dn</value>  </property>  <property>  <name>dfs.datanode.hdfs-blocks-metadata.enabled</name>  <value>true</value>  </property>  <property>  <name>fs.hdfs.impl.disable.cache</name>  <value>true</value>  </property>  <property>  <name>dfs.client.file-block-storage-locations.timeout.millis</name>  <value>10000</value>  </property>  <property>  <name>dfs.nameservices</name>  <value>sinovatiocluster</value>  </property>  <property>  <name>dfs.ha.namenodes.sinovatiocluster</name>  <value>nn1,nn2</value>  </property>  <property>  <name>dfs.namenode.rpc-address.sinovatiocluster.nn1</name>  <value>rhino001:9020</value>  </property>  <property>  <name>dfs.namenode.rpc-address.sinovatiocluster.nn2</name>  <value>rhino002:9020</value>  </property>  <property>  <name>dfs.namenode.http-address.sinovatiocluster.nn1</name>  <value>rhino001:50070</value>  </property>  <property>  <name>dfs.namenode.http-address.sinovatiocluster.nn2</name>  <value>rhino002:50070</value>  </property>  <property>  <name>dfs.namenode.shared.edits.dir</name>  <value>qjournal://rhino001:8485;rhino002:8485;rhino003:8485/sinovatiocluster</value>  </property>  <property>  <name>dfs.client.failover.proxy.provider.sinovatiocluster</name>  <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>  </property>  <property>  <name>dfs.ha.fencing.methods</name>  <value>sshfence</value>  </property>  <property>  <name>dfs.ha.fencing.ssh.private-key-files</name>  <value>/home/rhino/.ssh/id\_rsa</value>  </property>  <property>  <name>dfs.journalnode.edits.dir</name>  <value>/home/rhino/journalnode/data</value>  </property>  <property>  <name>dfs.ha.automatic-failover.enabled</name>  <value>true</value>  </property>  <property>  <name>dfs.namenode.name.dir</name>  <value>file:///home/rhino/namenode</value>  </property>  <property>  <name>dfs.datanode.data.dir</name>  <value>/**data01**/datanode,/**data02**/datanode,/data03/datanode,/data04/datanode,/data05/datanode,/data06/datanode</value>  </property>  <property>  <name>dfs.replication</name>  <value>2</value>  </property>  <property>  <name>dfs.webhdfs.enabled</name>  <value>true</value>  </property>  <property>  <name>dfs.journalnode.http-address</name>  <value>0.0.0.0:8480</value>  </property>  <property>  <name>dfs.journalnode.rpc-address</name>  <value>0.0.0.0:8485</value>  </property>  <property>  <name>dfs.permissions.enabled</name>  <value>false</value>  </property>  <property>  <name>dfs.datanode.max.transfer.threads</name>  <value>4096</value>  </property>  <property>  <name>dfs.namenode.handler.count</name>  <value>10</value>  </property>  <property>  <name>dfs.datanode.handler.count</name>  <value>10</value>  </property>  <property>  <name>dfs.blocksize</name>  <value>268435456</value>  </property>  <property>  <name>fs.hdfs.impl.disable.cache</name>  <value>true</value>  </property>  </configuration> |

* + 1. **安装及初始化**
* 配置HADOOP\_HOME环境变量

[rhino@rhino001 ~]$ vim ~/.bashrc

|  |
| --- |
| export JAVA\_HOME=/home/rhino/jdk1.8.0\_101  export ZOOKEEPER\_HOME=/home/rhino/zookeeper-3.4.5-cdh5.7.0  export HADOOP\_HOME=/home/rhino/hadoop-2.6.0-cdh5.7.0  export HADOOP\_PID\_DIR=/home/rhino/pids  export YARN\_PID\_DIR=/home/rhino/pids  export PATH=$JAVA\_HOME/bin:$PATH:$ZOOKEEPER\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin |

* 将相关目录拷贝到其他机器

[rhino@rhino001 ~]$ scp -r ~/hadoop-2.6.0-cdh5.7.0 rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/hadoop-2.6.0-cdh5.7.0 rhino@rhino003:~

[rhino@rhino001 ~]$ scp -r ~/hadoop-2.6.0-cdh5.7.0 rhino@rhino004:~

[rhino@rhino001 ~]$ scp ~/.bashrc rhino@rhino002:~

[rhino@rhino001 ~]$ scp ~/.bashrc rhino@rhino003:~

[rhino@rhino001 ~]$ scp ~/.bashrc rhino@rhino004:~

* 使每台机器环境变量生效

[rhino@rhino001 ~]$ source ~/.bashrc

* 为每台机器创建hdfs-sockets目录

[rhino@rhino001 ~]$ mkdir/home/rhino/hdfs-sockets

* 为每台机器修改hdfs-sockets组权限

切换到root用下执行

[root@rhino001 ~]# chownrhino:root /home/rhino/hdfs-sockets

* 以rhino用户在rhino001、rhino002、rhino003上启动journalnode进程

[rhino@rhino001 ~]$ hadoop-daemon.sh start journalnode

[rhino@rhino002~]$ hadoop-daemon.sh start journalnode

[rhino@rhino003 ~]$ hadoop-daemon.sh start journalnode

* 在rhino001上格式化zkfc

[rhino@rhino001 ~]$ hdfs zkfc -formatZK

* 在rhino001上以rhino用户进行namenode格式化，并将格式化的数据拷贝到rhino002上。

[rhino@rhino001 ~]$ hdfs namenode-format

[rhino@rhino001 ~]$ scp -r /home/rhino/namenode rhino@rhino002:~/

* 在master上启动HDFS

[rhino@rhino001 ~]$ start-dfs.sh

* 检查HDFS是否正常启动

登录到hdfs监控页面192.168.10.1:50070/dfshealth.html查看Live Nodes是否为4(即配置的slaves节点个数)，如果为4即说明hdfs已正常启动。

* 停止HDFS(需要停止时执行)

在master节点运行如下命令：

[rhino@rhino001 ~]$ stop-dfs.sh

1. **部署impala集群**
   1. **安装impala**

* 以rhino用户在rhino001机器下解压impala安装包

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar -zxvf impala-2.12.0.tar.gz

[rhino@rhino001 ~]$ ln -s impala-2.12.0/ impala

* 拷贝**hadoop配置文件**到impala配置中

[rhino@rhino001 ~]$ cd /home/rhino/hadoop/etc/hadoop

[rhino@rhino001 hadoop]$cpcore-site.xmlhdfs-site.xml /home/rhino/impala/lib/impala/conf

* 配置**hive-site.xml**配置文件

[rhino@rhino001 ~]$ vim /home/rhino/impala/lib/impala/conf/hive-site.xml

|  |
| --- |
| <configuration>  <property>  <name>hive.metastore.client.socket.timeout</name>  <value>3600</value>  <description>MetaStore Client socket timeout in seconds</description>  </property>  <property>  <name>javax.jdo.option.ConnectionURL</name><value>jdbc:mysql://192.168.10.101:3306/hive\_rhino?createDatabaseIfNotExist=true</value>  </property>  <property>  <name>javax.jdo.option.ConnectionDriverName</name>  <value>com.mysql.jdbc.Driver</value>  </property>  <property>  <name>javax.jdo.option.ConnectionUserName</name>  <value>rhino</value>  </property>  <property>  <name>javax.jdo.option.ConnectionPassword</name>  <value>rhino</value>  </property>  <property>  <name>hive.querylog.location</name>  <value>${user.home}/hive-logs/querylog</value>  </property>  </configuration> |

* 配置**impala-env.sh**配置文件

[rhino@rhino001 ~]$ vim /home/rhino/impala/sbin/impala-env.sh

|  |
| --- |
| export IMPALA\_HOME=~/impala/lib/impala  export IMPALA\_BIN=$IMPALA\_HOME/sbin  export IMPALA\_CONF\_DIR=$IMPALA\_HOME/conf  export LIBHDFS\_OPTS=$IMPALA\_HOME/lib  export MYSQL\_CONNECTOR\_JAR=${MYSQL\_CONNECTOR\_JAR:$IMPALA\_HOME/lib/mysql-connector-java-5.1.31.jar}  export IMPALA\_SHELL\_HOME=$IMPALA\_HOME/../impala-shell  #**java tool options,具体-Xmx内存大小需要根据实际情况确定**  export IMPALAD\_JAVA\_TOOL\_OPTIONS=""  export CATALOGD\_JAVA\_TOOL\_OPTIONS="-Xmx**20G** -XX:+UseG1GC -XX:+UnlockExperimentalVMOptions -XX:MaxGCPauseMillis=200 -XX:G1NewSizePercent=1 -XX:InitiatingHeapOccupancyPercent=65 -XX:+ParallelRefProcEnabled -XX:ConcGCThreads=4 -XX:ParallelGCThreads=16 -XX:MaxTenuringThreshold=1 -XX:G1HeapRegionSize=32m -XX:G1MixedGCCountTarget=32 -XX:G1OldCSetRegionThresholdPercent=2 -XX:SurvivorRatio=4 -XX:+UnlockDiagnosticVMOptions -XX:+G1SummarizeConcMark"  #**impalad args,需要在对应路径下，新建下面的目录**  export IMPALA\_ARGS="--scratch\_dirs**=/data01/impala\_tmp,/data02/impala\_tmp,/data03/impala\_tmp,/data04/impala\_tmp,/data05/impala\_tmp,/data06/impala\_tmp**"  #catalogd args  export CATALOGD\_ARGS="--num\_metadata\_loading\_threads=12 "  #statestore args  export STATESTORE\_ARGS=""  export STATE\_STORE\_HOST=**"192.168.10.1"**  export CATALOG\_SERVICE\_HOST**="192.168.10.1"**  #**所有impalad的地址，如果需要在catalogd服务器上启动impala，ip请放在最后一个**  export impalad\_ips=( **"rhino002" "rhino003"**); |

* 配置start-impalad.sh配置文件

cd /home/impala/impala/sbin

vim start-impalad.sh

|  |
| --- |
| 找到  nohup $IMPALA\_HOME/../../bin/impalad ${IMPALA\_ARGS} -log\_dir=$IMPALA\_HOME/logs > $IMPALA\_HOME/impalad.out 2>&1 < /dev/null &  改成  nohup $IMPALA\_HOME/../../bin/impalad ${IMPALA\_ARGS} -use\_local\_tz\_for\_unix\_timestamp\_conversions=true -convert\_legacy\_hive\_parquet\_utc\_timestamps=true -log\_dir=$IMPALA\_HOME/logs > $IMPALA\_HOME/impalad.out 2>&1 < /dev/null & |

* 将rhino001上impala文件夹拷贝到其他机器

[rhino@rhino001 ~]$ scp -r ~/impala-2.12.0 rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/impala-2.12.0 rhino@rhino003:~

* 分别在rhino002，rhino003上建立软连接

[rhino@rhino002 ~]$ ln -s impala-2.12.0/ impala

[rhino@rhino003 ~]$ ln -s impala-2.12.0/ impala

* 1. **启动impala各进程**
* 在**rhino001**机器上启动**statestored**，**catalogd**进程

[rhino@rhino001 ~]$ ~/impala/sbin/start-statestored.sh

[rhino@rhino001 ~]$ ~/impala/sbin/start-catalogd.sh

* 在**rhino002**，**rhino003**机器上启动**impalad**进程

[rhino@rhino002 ~]$

~/impala/sbin/start-impalad.sh -state\_store\_host=192.168.10.1 -catalog\_service\_host=192.168.10.1

[rhino@rhino003 ~]$

~/impala/sbin/start-impalad.sh -state\_store\_host=192.168.10.1 -catalog\_service\_host=192.168.10.1

* 验证impala是否正常启动

在部署statestored和catalogd的节点使用如下命令查看进程是否存在：

ps -ef | grep statestored | grep -v grep; ps -ef | grep catalogd | grep -v grep

在部署impalad 的节点使用如下命令查看进程是否存在：

ps -ef | grep impalad| grep -v grep

* 1. **利用impala-shell建表并查询数据**
* 在rhino002上建立两个测试样本文件并上传hdfs

[rhino@rhino002 ~]$ vim ~/tab1.csv

|  |
| --- |
| 1,true,123.123,2012-10-24 08:55:00  2,false,1243.5,2012-10-25 13:40:00  3,false,24453.325,2008-08-22 09:33:21.123  4,false,243423.325,2007-05-12 22:32:21.33454  5,true,243.325,1953-04-22 09:11:33s |

[rhino@rhino002 ~]$ vim ~/tab2.csv

|  |
| --- |
| 1,true,12789.123  2,false,1243.5  3,false,24453.325  4,false,2423.3254  5,true,243.325  60,false,243565423.325  70,true,243.325  80,false,243423.325  90,true,243.325 |

[rhino@rhino002 ~]$ hdfs dfs -mkdir -p /user/rhino/sample\_data/tab1

[rhino@rhino002 ~]$ hdfs dfs -mkdir -p /user/rhino/sample\_data/tab2

[rhino@rhino002 ~]$ hdfs dfs -put ~/tab1.csv /user/rhino/sample\_data/tab1/

[rhino@rhino002 ~]$ hdfs dfs -put ~/tab2.csv /user/rhino/sample\_data/tab2/

* 建立两个测试表

[rhino@rhino002 ~]$ ~/impala/sbin/impala-shell.sh

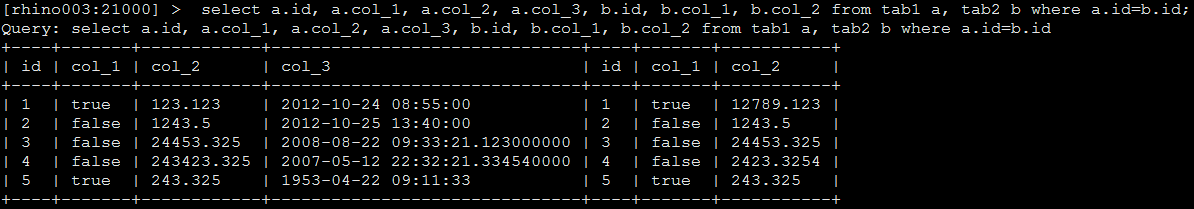
[rhino002:21000]> CREATE EXTERNAL TABLE tab1(id INT, col\_1 BOOLEAN, col\_2 DOUBLE, col\_3 TIMESTAMP) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/user/rhino/sample\_data/tab1';

[rhino002:21000]> CREATE EXTERNAL TABLE tab2(id INT, col\_1 BOOLEAN, col\_2 DOUBLE) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/user/rhino/sample\_data/tab2';

* 查询测试

[rhino002:21000]> select a.id, a.col\_1, a.col\_2, a.col\_3, b.id, b.col\_1, b.col\_2 from tab1 a, tab2 b where a.id=b.id;

查询结果如下图所示：



* 1. **停止impala各进程**
* 在rhino001机器上停止statestored，catalogd进程

[rhino@rhino001 ~]$ ~/impala/sbin/stop-statestored.sh; ~/impala/sbin/stop-catalogd.sh

* 在rhino002，rhino003机器上停止impalad进程

[rhino@rhino002 ~]$ ~/impala/sbin/stop-impalad.sh

1. **安装spark**
   1. **安装scala**

* 以rhino用户在rhino001节点解压scala安装包(scala-2.11.7.tgz)

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar zxvfscala-2.11.7.tgz

* 配置SCALA\_HOME环境变量

[rhino@rhino001 ~]$ vim ~/.bashrc

|  |
| --- |
| export JAVA\_HOME=/home/rhino/jdk1.8.0\_101  export HADOOP\_HOME=/home/rhino/hadoop-2.6.0-cdh5.7.0  export SCALA\_HOME=/home/rhino/scala-2.11.7  export SENTRY\_HOME=/home/rhino/sentry-1.4.0-cdh5.4.2  export PATH=$JAVA\_HOME/bin:$PATH:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin:$SCALA\_HOME/bin:$ SENTRY\_HOME/bin |

* 拷贝scala及环境变量到slave机器

[rhino@rhino001 ~]$ scp -r ~/scala-2.11.7 rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/scala-2.11.7 rhino@rhino003:~

[rhino@rhino001 ~]$ scp -r ~/scala-2.11.7 rhino@rhino004:~

[rhino@rhino001 ~]$ scp -r ~/.bashrc rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/.bashrc rhino@rhino003:~

[rhino@rhino001 ~]$ scp -r ~/.bashrc rhino@rhino004:~

* 使每台机器环境变量生效

[rhino@rhino001 ~]$ source ~/.bashrc

* 验证scala

[rhino@rhino001 ~]$ scala -version

Scala code runner version 2.11.7 -- Copyright 2002-2013, LAMP/EPFL

* 1. **配置hive-site.xml**

[rhino@rhino001 ~]$ vim/home/rhino/spark/conf/hive-site.xml

|  |
| --- |
| <configuration>  <property>  <name>hive.metastore.client.socket.timeout</name>  <value>3600</value>  <description>MetaStore Client socket timeout in seconds</description>  </property>  <property>  <name>javax.jdo.option.ConnectionURL</name>  <value>jdbc:mysql://192.168.10.49:3306/hive\_rhino?createDatabaseIfNotExist=true</value>  </property>  <property>  <name>javax.jdo.option.ConnectionDriverName</name>  <value>com.mysql.jdbc.Driver</value>  </property>  <property>  <name>javax.jdo.option.ConnectionUserName</name>  <value>rhino</value>  </property>  <property>  <name>javax.jdo.option.ConnectionPassword</name>  <value>rhino</value>  </property>  </configuration> |

* 1. **安装spark**
* 以rhino用户在rhino001节点解压spark安装包(spark-2.2.0-bin-hadoop2.6.tgz)

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001~]$tar -zxvfspark-2.2.0-bin-hadoop2.6.tgz

* 配置SPARK\_HOME环境变量

[rhino@rhino001 ~]$ vim ~/.bashrc

|  |
| --- |
| export JAVA\_HOME=/home/rhino/jdk1.8.0\_101  export HADOOP\_HOME=/home/rhino/hadoop-2.6.0-cdh5.7.0  export SCALA\_HOME=/home/rhino/scala-2.11.7  export SPARK\_HOME=/home/rhino/spark-2.2.0-bin-hadoop2.6  export SENTRY\_HOME=/home/rhino/sentry-1.4.0-cdh5.4.2  export SPARK\_PID\_DIR=/home/rhino/pids  export PATH=$JAVA\_HOME/bin:$PATH:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin:$SCALA\_HOME/bin:$SPARK\_HOME/bin:$SENTRY\_HOME/bin |

* 配置slaves节点

[rhino@rhino001 ~]$ cd /home/rhino/spark-2.2.0-bin-hadoop2.6/conf

[rhino@rhino001 ~]$ vim slaves

|  |
| --- |
| rhino003 #注：slave的主机名  rhino004 #注：slave的主机名 |

* 配置spark-en.sh文件

[rhino@rhino001 ~]$ cd /home/rhino/spark-2.2.0-bin-hadoop2.6/conf

[rhino@rhino001 ~]$ cp spark-env.sh.template spark-env.sh

在spark-env.sh文件末添加所需的环境变量配置：

[rhino@rhino001 ~]$ vimspark-env.sh

|  |
| --- |
| export JAVA\_HOME=/home/rhino/jdk1.8.0\_101  export SCALA\_HOME=/home/rhino/scala-2.11.7  export SPARK\_MASTER\_IP=192.168.10.1  export SPARK\_WORKER\_MEMORY=2g  export HADOOP\_CONF\_DIR=/home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop  exportSPARK\_CLASSPATH=$SPARK\_CLASSPATH:/home/rhino/spark/jars/mysql-connector-java-5.1.35.jar |

* 配置spark-defaults.conf文件

[rhino@rhino001 ~]$ cd /home/rhino/spark-2.2.0-bin-hadoop2.6/conf

[rhino@rhino001 ~]$ cp spark-defaults.conf.template spark-defaults.conf

在spark-defaults.conf文件末添加所需的配置：

[rhino@rhino001 ~]$ vim spark-defaults.conf

|  |
| --- |
| #spark临时目录  spark.local.dir /home/rhino/tmp  spark.sql.warehouse.dir hdfs://sinovatiocluster/user/hive/warehouse |

* 将相关目录拷贝到其他机器

[rhino@rhino001 ~]$ scp -r ~/spark-2.2.0-bin-hadoop2.6 rhino@rhino002:~

[rhino@rhino001 ~]$ scp -r ~/spark-2.2.0-bin-hadoop2.6 rhino@rhino003:~

[rhino@rhino001 ~]$ scp -r ~/spark-2.2.0-bin-hadoop2.6 rhino@rhino004:~

[rhino@rhino001 ~]$ scp ~/.bashrc rhino@rhino002:~

[rhino@rhino001 ~]$ scp ~/.bashrc rhino@rhino003:~

[rhino@rhino001 ~]$ scp ~/.bashrc rhino@rhino004:~

* 将每台机器环境变量生效

[rhino@rhino001 ~]$ source ~/.bashrc

* 启动spark分布式集群并查看信息（注：YARN方式不需要启动spark）

[rhino@rhino001 ~]$ cd ~/spark-2.2.0-bin-hadoop2.6/sbin

[rhino@rhino001 ~] ./start-all.sh

[rhino@rhino001 ~]jps

* 停止spark分布式集群

[rhino@rhino001 ~] ./stop-all.sh

* 1. **上传jar包**
* 将spark环境中的jar包放到HDFS

[rhino@rhino001 ~]$hdfsdfs -mkdir hdfs://sinovatiocluster/sharkjars-2.0/

[rhino@rhino001 ~]$hdfs dfs -put ~/spark-2.2.0-bin-hadoop2.6/jars/\* hdfs://sinovatiocluster/sharkjars-2.0/

* 1. **修改yarn文件**
* 修改yarn文件（如果缺少的内容已存在，则无需修改）

[rhino@rhino001 ~]$vim~/hadoop-2.6.0-cdh5.7.0/bin/yarn

|  |
| --- |
| elif [ "$COMMAND" = "nodemanager" ] ; then  CLASSPATH=${CLASSPATH}:$YARN\_CONF\_DIR/nm-config/log4j.properties  **for f in ${SPARK\_HOME}/yarn/\*.jar; do**  **CLASSPATH=${CLASSPATH}:${f}**  **done**  **#在下面这句前面增加上述内容**  CLASS='org.apache.hadoop.yarn.server.nodemanager.NodeManager' |

* 将yarn文件拷贝到所有Nodemanager节点

[rhino@rhino001 ~]$ scp~/hadoop-2.6.0-cdh5.7.0/bin/yarnrhino@rhino002:~/hadoop-2.6.0-cdh5.7.0/bin

[rhino@rhino001 ~]$ scp~/hadoop-2.6.0-cdh5.7.0/bin/yarn rhino@rhino003~/hadoop-2.6.0-cdh5.7.0/bin

[rhino@rhino001 ~]$ scp~/hadoop-2.6.0-cdh5.7.0/bin/yarnrhino@rhino004~/hadoop-2.6.0-cdh5.7.0/bin

1. **部署YARN**
   * 1. **配置yarn**

* 配置yarn-site.xml

[rhino@rhino001 ~]$ cd /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/

[rhino@rhino001 hadoop]$ vim yarn-site.xml

|  |
| --- |
| <?xml version="1.0"?>  <configuration>  <property>  <name>yarn.resourcemanager.connect.retry-interval.ms</name>  <value>2000</value>  </property>  <property>  <name>yarn.resourcemanager.ha.enabled</name>  <value>true</value>  <description>此版本需要测试ha模式下的yarn</description>  </property>  <property>  <name>yarn.resourcemanager.ha.automatic-failover.enabled</name>  <value>true</value>  </property>  <property>  <name>yarn.resourcemanager.ha.rm-ids</name>  <value>rm1,rm2</value>  </property>  <property>  <name>yarn.resourcemanager.zk.state-store.address</name>  <value>rhino001:2181,rhino002:2181,rhino003:2181</value>  </property>  <property>  <name>ha.zookeeper.quorum</name>  <value> rhino001:2181,rhino002:2181,rhino003:2181</value>  </property>  <property>  <name>yarn.resourcemanager.recovery.enabled</name>  <value>true</value>  </property>  <property>  <name>yarn.app.mapreduce.am.scheduler.connection.wait.interval-ms</name>  <value>5000</value>  </property>  <property>  <name>yarn.resourcemanager.address.rm1</name>  <value>rhino001:8032</value>  </property>  <property>  <name>yarn.resourcemanager.scheduler.address.rm1</name>  <value> rhino001:8030</value>  </property>  <property>  <name>yarn.resourcemanager.webapp.address.rm1</name>  <value> rhino001:8088</value>  </property>  <property>  <name>yarn.resourcemanager.resource-tracker.address.rm1</name>  <value> rhino001:8031</value>  </property>  <property>  <name>yarn.resourcemanager.admin.address.rm1</name>  <value> rhino001:8033</value>  </property>  <property>  <name>yarn.resourcemanager.hostname.rm1</name>  <value> rhino001</value>  </property>  <property>  <name>yarn.resourcemanager.hostname.rm2</name>  <value> rhino002</value>  </property>  <property>  <name>yarn.resourcemanager.address.rm2</name>  <value> rhino002:8032</value>  </property>  <property>  <name>yarn.resourcemanager.scheduler.address.rm2</name>  <value> rhino002:8030</value>  </property>  <property>  <name>yarn.resourcemanager.webapp.address.rm2</name>  <value> rhino002:8088</value>  </property>  <property>  <name>yarn.resourcemanager.resource-tracker.address.rm2</name>  <value> rhino002:8031</value>  </property>  <property>  <name>yarn.resourcemanager.admin.address.rm2</name>  <value> rhino002:8033</value>  </property>  <property>  <name>yarn.resourcemanager.zk-address</name>  <value>rhino001:2181,rhino002:2181,rhino003:2181</value>  </property>  <property>  <name>yarn.resourcemanager.cluster-id</name>  <value>sinovatiocluster</value>  </property>  <property>  <name>yarn.resourcemanager.scheduler.class</name><value>org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler</value>  <description>必填项</description>  </property>  <property>  <name>yarn.scheduler.fair.allocation.file</name>  <value>/home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/fair-scheduler.xml</value>  </property>  <property>  <name>yarn.scheduler.fair.preemption</name>  <value>true</value>  </property>  <property>  <name>yarn.scheduler.fair.assignmultiple</name>  <value>true</value>  </property>  <property>  <name>yarn.scheduler.fair.user-as-default-queue</name>  <value>false</value>  <description>default is True</description>  </property>  <property>  <name>yarn.scheduler.fair.allow-undeclared-pools</name>  <value>false</value>  <description>default is True</description>  </property>  <property>  <name>yarn.acl.enable</name>  <value>false</value>  </property>  <property>  <name>yarn.admin.acl</name>  <value>rhino</value>  </property>  <property>  <name>yarn.nodemanager.resource.memory-mb</name>  <value>113160</value>  <discription>每个nodemanager节点可用内存,单位MB</discription>  </property>  <property>  <name>yarn.nodemanager.resource.cpu-vcores</name>  <value>20</value>  </property>  <property>  <name>yarn.scheduler.minimum-allocation-mb</name>  <value>2048</value>  <discription>单个任务可申请最小内存，默认1024MB</discription>  </property>  <property>  <name>yarn.scheduler.maximum-allocation-mb</name>  <value>64000</value>  <discription>单个任务可申请最大内存，默认8192MB</discription>  </property>  <property>  <name>yarn.scheduler.maximum-allocation-vcores</name>  <value>60</value>  </property>  <property>  <name>yarn.nodemanager.vmem-check-enabled</name>  <value>false</value>  <description>Whether virtual memory limits will be enforced for containers</description>  </property>  <property>  <name>yarn.nodemanager.local-dirs</name>  <value>/home/rhino/spark/yarn\_spark/</value>  </property>  <property>  <name>yarn.nodemanager.aux-services</name>  <value>spark\_shuffle,mapreduce\_shuffle</value>  <description>开启动态分配时需要</description>  </property>  <property>  <name>yarn.nodemanager.aux-services.spark\_shuffle.class</name>  <value>org.apache.spark.network.yarn.YarnShuffleService</value>  <description>开启动态分配时需要</description>  </property>  <property>  <name>yarn.nodemanager.aux-services.mapreduce\_shuffle.class</name>  <value>org.apache.hadoop.mapred.ShuffleHandler</value>  </property>  <property>  <name>yarn.nodemanager.localizer.address</name>  <value>0.0.0.0:23344</value>  </property>  <property>  <name>yarn.nodemanager.webapp.address</name>  <value>0.0.0.0:23999</value>  </property>  <property>  <name>yarn.nodemanager.log-dirs</name>  <value>/home/rhino/hadoop-2.6.0-cdh5.7.0/logs</value>  </property>  </configuration> |

* 新建并配置fair-scheduler.xml（部署在集群内每台resourcemanager机器上）。

[rhino@rhino001 ~]$ cd /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/

[rhino@rhino001hadoop]$ vimfair-scheduler.xml

**(root是总资源，为其余队列资源之和。)**

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <allocations>  <queue name="root">  <maxResources>120000 mb,24 vcores</maxResources>  <aclSubmitApps></aclSubmitApps>  <schedulingMode>fair</schedulingMode>  <aclAdministerApps></aclAdministerApps>  <minResources>1000 mb,2 vcores</minResources>  <maxRunningApps>10</maxRunningApps>  <queue name="spark-max">  <maxResources>80000 mb,14 vcores</maxResources>  <aclSubmitApps>rhino,admin</aclSubmitApps>  <weight>2.0</weight>  <minResources>1000 mb,2 vcores</minResources>  <maxRunningApps>10</maxRunningApps>  <schedulingPolicy>fair</schedulingPolicy>  </queue>  <queue name="spark-mid">  <maxResources>30000 mb,10 vcores</maxResources>  <aclSubmitApps>rhino,admin</aclSubmitApps>  <weight>1.0</weight>  <minResources>1000 mb,2 vcores</minResources>  <maxRunningApps>10</maxRunningApps>  <schedulingPolicy>fair</schedulingPolicy>  </queue>  <queue name="spark-min">  <maxResources>10000 mb,10 vcores</maxResources>  <aclSubmitApps>rhino,admin</aclSubmitApps>  <weight>1.0</weight>  <minResources>1000 mb,2 vcores</minResources>  <maxRunningApps>10</maxRunningApps>  <schedulingPolicy>fair</schedulingPolicy>  </queue>  </queue>  </allocations> |

**说明：**

1. **fair-scheduler.xml配置是yarn资源分配采用fair模式下配置文件；**
2. **请根据实际情况配置队列个数，以及队列中的配置值；**
3. **aclSubmitApps和aclAdministerApps标签中如果没有用户，请添加空格。**
   * 1. **启动YARN**

* 以rhino用户启动yarn

在rm1节点运行如下命令：

[rhino@rhino001 ~]$ start-yarn.sh

在rm2节点运行如下命令：

[rhino@rhino002 ~]$yarn-daemon.sh start resourcemanager

* 检查YARN是否正常启动

登录到yarn监控页面192.168.10.1:8088/cluster/nodes，查看Active Nodes是否为4(即配置的slaves节点个数)，如果为4即说明yarn已正常启动。

* 停止YARN(需要停止时执行)

在rm1节点运行如下命令：

[rhino@rhino001 ~]$ stop-yarn.sh

在rm2节点运行如下命令：

[rhino@rhino002 ~]$yarn-daemon.sh stop resourcemanager

* + 1. **启动thriftserver**

注意：如果配置impala则不需要启动thriftserver，如果配置连接方式为sparksql则需要按照如下操作：

启动Thriftserver组件作为spark jdbc的服务端：

[rhino@rhino001 ~]$~/spark-2.2.0-bin-hadoop2.6/sbin/start-thriftserver.sh --master yarn-client --num-executors1 --executor-cores2 --executor-memory 4g--queue root.spark-mid

注：root.spark-mid为fair-scheduler.xml中配置的队列名。按照以上启动参数，spark-mid队列至少需要3核、6G内存的资源。

1. **部署Kafka集群**

Kafka集群的安装三台机器基本一致，主要就是配置server.properties中broker.id=0保证不一致，他们通过连接相同的zookeeper.connect进入集群。

* 1. **安装Kafka**
* 以rhino用户在rhino001机器下解压安装包

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar -zxvfkafka\_2.11-1.1.0.tgz

* 将kafka拷贝到其他机器并分别解压

[rhino@rhino001 ~]$scpkafka\_2.11-1.1.0.tgz rhino002:~

[rhino@rhino001 ~]$scpkafka\_2.11-1.1.0.tgz rhino003:~

在rhino002和rhino003上执行

[rhino@rhino001 ~]$ tar -zxvf kafka\_2.11-1.1.0.tgz

* 1. **配置kafka**
* 每台机器的配置类似，以rhino001上配置为例

[rhino@rhino001 ~]$ cd/home/rhino/kafka\_2.11-1.1.0

[rhino@rhino001 kafka\_2.11-1.1.0]$ vimconfig/server.properties

|  |
| --- |
| # The id of the broker. This must be set to a unique integer for each broker.  broker.id=0（每台机器确保不一样）  # Hostname and port the broker will advertise to producers and consumers. If not set,  # it uses the value for "listeners" if configured. Otherwise, it will use the value  # returned from java.net.InetAddress.getCanonicalHostName().  advertised.listeners=PLAINTEXT://rhino001:9092（rhino002填rhino002:9092，其他同理）  # A comma seperated list of directories under which to store log files  log.dirs=/data01/kafka-logs,/data02/kafka-logs,/data03/kafka-logs（如果有多块磁盘，并对kafka的性能有较高要求，建议在每个磁盘下都配置一个目录）  # Zookeeper connection string (see zookeeper docs for details).  zookeeper.connect=rhino001:2181,rhino002:2181,rhino003:2181 |

* 1. **启动kafka**
* 每台机器启动kafka服务

[rhino@rhino001 ~]$ cd/home/rhino/kafka\_2.11-1.1.0

[rhino@rhino001 kafka\_2.11-1.1.0]$ bin/kafka-server-start.sh -daemon config/server.properties

* 验证进程是否正常启动

使用 “ps -ef | grep kafka| grep -v grep” 命令观察进程是否存在。

* 验证消息发送和接收

发送消息：

[rhino@rhino001 kafka\_2.11-1.1.0]$ bin/kafka-console-producer.sh --broker-list rhino001:9092, rhino002:9092, rhino003:9092 --topic test

在控制台输入一些消息，按ctrl+c退出发送。

读取消息：

[rhino@rhino001 kafka\_2.11-1.1.0]$

控制台中看到之前发送的消息则证明kafka配置成功，按ctrl+c退出控制台。

* 1. **停止kafka**

分别在对应的机器上停止相应进程（需要停止时执行）

* 停止kafka进程

[rhino@rhino001 kafka\_2.11-1.1.0]$bin/kafka-server-stop.sh

* 1. **安装kafka manager**

Kafka manager可用于监控和管理kafka集群。

* + 1. **修改kafka**
* 修改kafka启动文件

修改每个kafka节点上的kafka-server-start.sh文件：

[rhino@rhino001 ~]$ vim/home/rhino/kafka\_2.11-1.1.0/bin/kafka-server-start.sh

|  |
| --- |
| export JMX\_PORT="8999"  #在下面这句前面增加上述内容  EXTRA\_ARGS=${EXTRA\_ARGS-'-name kafkaServer -loggc'} |

* 重启kafka

重新启动所有的kafka进程：

[rhino@rhino001 ~]$ /home/rhino/kafka\_2.11-1.1.0/bin/kafka-server-start.sh -daemon config/server.properties

* + 1. **安装和启动**
* 以rhino用户在rhino001机器下解压安装包

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ unzip kafka-manager-1.3.3.4.zip

* 修改配置文件

[rhino@rhino001 ~]$ vim/home/rhino/kafka-manager-1.3.3.4/conf/application.conf

|  |
| --- |
| kafka-manager.zkhosts="rhino001:2181,rhino002:2181,rhino003:2181" |

* 启动kafka-manager

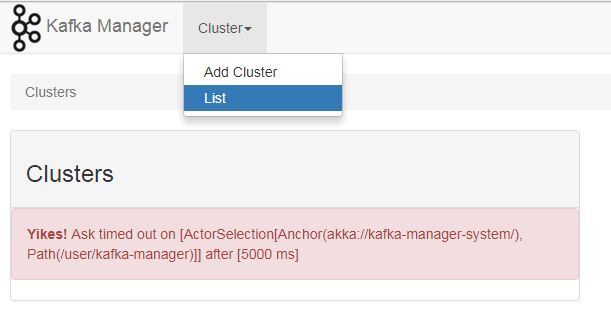
[rhino@rhino001 ~]$ cd/home/rhino/kafka-manager-1.3.3.4

[rhino@rhino001 ~]$ nohup bin/kafka-manager -Dconfig.file=conf/application.conf -Dhttp.port=9000 &

* + 1. **配置kafka manager**

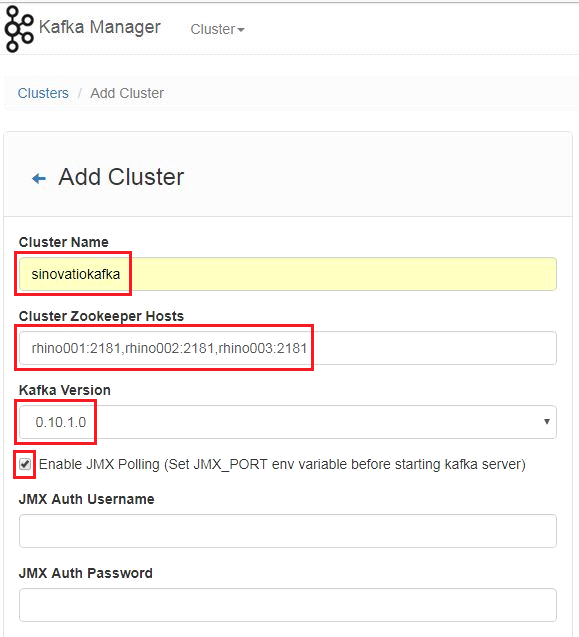
登录管理界面http://192.168.10.1:9000，对kafka manager进行配置。

* 增加集群



* 配置集群信息

修改图中红圈几处即可。



1. **安装Redis**

当前后台通讯选择redis，或使用**流式告警**且配置告警阈值，需要安装Redis。

* 1. **安装**

|  |
| --- |
| * 以rhino 用户解压redis安装文件   [rhino@rhino001 ~]$cd ~  [rhino@rhino001 ~]tar -zxvf redis-4.0.10.tar.gz   * make编译程序:   [rhino @rhino061 ~]$ cd redis-4.0.10/  [rhino @rhino061 redis-4.0.10]$ make  编译完成后页面提示：     * make安装程序:   切换到root用户下执行make install    安装完成后文件都生成在/usr/local/bin/下。   * 新建redis目录   rhino用户下执行：  mkdir -p /home/rhino/redis/bin  mkdir -p /home/rhino/redis/conf  root用户下执行：  mv /usr/local/bin/\* mkdir -p /home/rhino/redis/bin  chown rhino:rhino /home/rhino/redis/bin/\*  rhino用户下执行：  mv /home/rhino/redis-4.0.10/redis.conf /home/rhino/redis/conf |

* 1. **配置**

1. vim /home/rhino/redis/redis.conf

|  |
| --- |
| # By default Redis does not run as a daemon. Use 'yes' if you need it.  # Note that Redis will write a pid file in /var/run/redis.pid when daemonized.  daemonize yes  # IF YOU ARE SURE YOU WANT YOUR INSTANCE TO LISTEN TO ALL THE INTERFACES  # JUST COMMENT THE FOLLOWING LINE.  # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  bind 162.168.2.61  # requirepass foobared，指定密码，也可以不指定，如果指定，登录客户端时候需要密码认证  requirepass 123456 |

2. 配置环境变量

vim ~/.bashrc,加上如下内容：

|  |
| --- |
| export REDIS\_HOME=/home/rhino/redis  export PATH里加上$REDIS\_HOME/bin |

保存环境变量：

source ~/.bashrc

* 1. **启动和停止**
* 启动redis服务端

|  |
| --- |
| [rhino@rhino061 ~]$ redis-server /home/rhino/redis/conf/redis.conf |

然后通过ps -ef|grep redis命令或者安装目录下的logs目录查看进程是否启动成功。

* 启动redis客户端

|  |
| --- |
| 输入redis-cli命令启动客户端  [rhino@rhino061 ~]$ redis-cli -h 162.168.2.61  162.168.2.61:6379> auth 123456 #密码认证，如果没有密码，此步可以省略  OK  162.168.2.61:6379>info  #启动后输入info查看redis相关信息 |

* 停止redis服务

|  |
| --- |
| [rhino@rhino061 ~]$ redis-cli shutdown |

1. **安装tomcat**
   1. **前台tomcat**
      1. **安装**

注：若无特殊需要可直接用OceanMind系统的前台tomcat；若需要另外安装个tomcat供OceanEye使用，参考如下步骤。

1. 解压tomcat安装文件

[rhino@rhino181 ~]$cd /home/rhino

[rhino@rhino181 ~]$unzip apache-tomcat-8.0.35.zip

[rhino@rhino181 ~]mv /home/rhino/apache-tomcat-8.0.35 /home/rhino/apache-tomcat-8.0.35-oceaneye

**注：jdk必须使用1.8及以上版本**

1. 若环境中存在其他tomcat，请修改server.xml文件中的端口号，以免端口冲突无法启动tomcat。

[rhino@rhino002 conf]$ vi /home/rhino/apache-tomcat-8.0.35-oceaneye/conf/server.xml

<?xml version='1.0' encoding='utf-8'?>

<Server port="8005" shutdown="SHUTDOWN">

<Listener className="org.apache.catalina.startup.VersionLoggerListener" />

<Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />

<Listener className="org.apache.catalina.core.JasperListener" />

<Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />

<Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />

<Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />

<GlobalNamingResources>

<Resource name="UserDatabase" auth="Container"

type="org.apache.catalina.UserDatabase"

description="User database that can be updated and saved"

factory="org.apache.catalina.users.MemoryUserDatabaseFactory"

pathname="conf/tomcat-users.xml" />

</GlobalNamingResources>

<Service name="Catalina">

<Connector port="9999" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

<Connector port="8009" protocol="AJP/1.3"redirectPort="8443" />

<Engine name="Catalina" defaultHost="localhost">

<Realm className="org.apache.catalina.realm.LockOutRealm">

<Realm className="org.apache.catalina.realm.UserDatabaseRealm"

resourceName="UserDatabase"/>

</Realm>

<Host name="localhost" appBase="webapps"

unpackWARs="true" autoDeploy="true">

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

prefix="localhost\_access\_log." suffix=".txt"

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

</Host>

</Engine>

</Service>

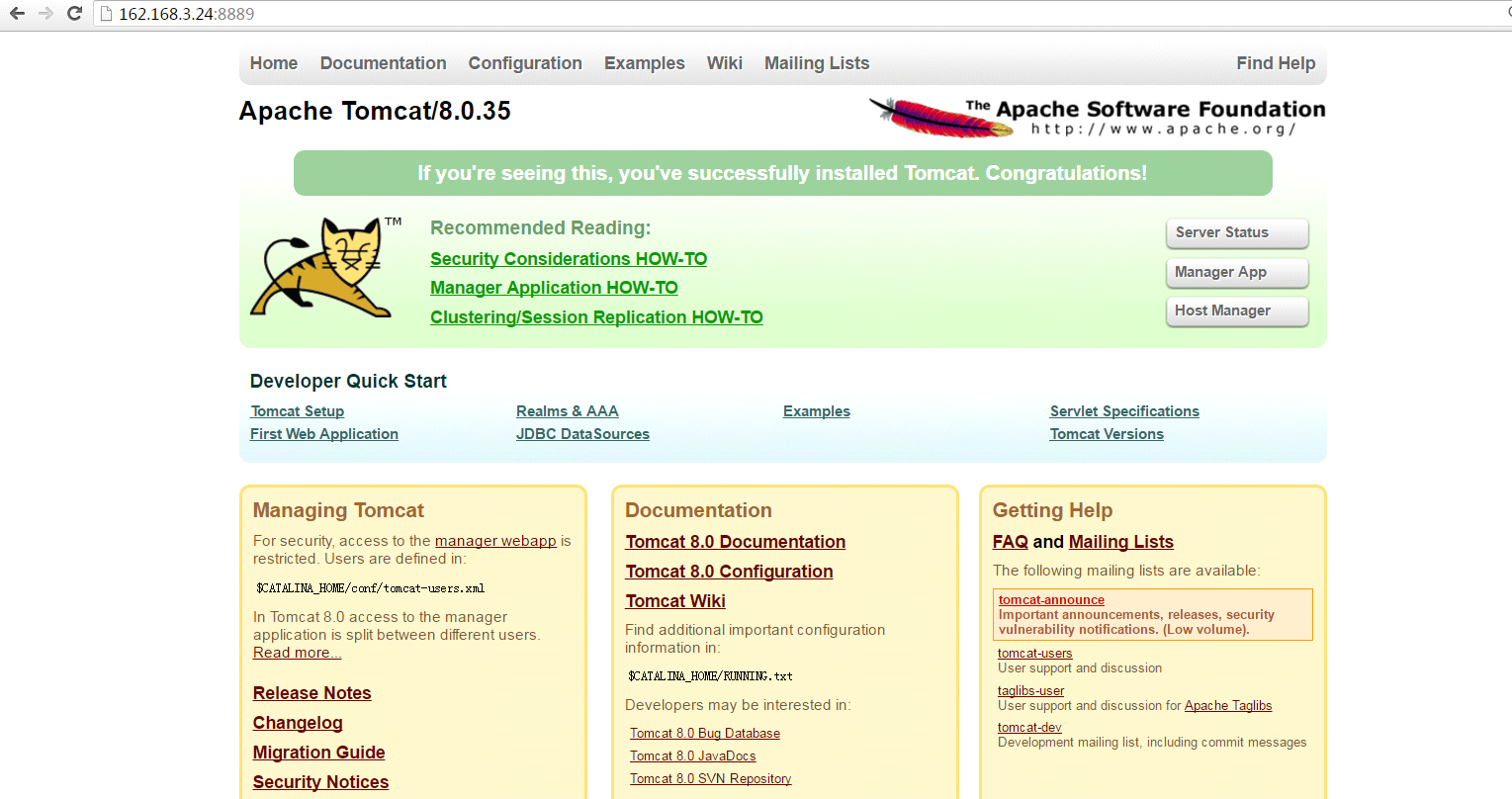
</Server>

1. 检验是否能正常启动tomcat

[rhino@rhino181 ~]$cd /home/rhino/apache-tomcat-8.0.35-oceaneye/bin/

[rhino@rhino181 ~]$./startup.sh

进入浏览器，输入ip:port可查看tomcat是否正常启动，启动成功的页面如下：



* + 1. **配置tomcat**

1. 将xxx.war文件拷贝至tomcat的webapp目录下

[rhino@rhino181 ~]$cp xxx.war /home/rhino/apache-tomcat-8.0.35-oceaneye/webapps

拷贝结束后将会自动解压成oceansource文件夹(若未生成该文件夹，请重启tomcat)

1. 修改oceaneye中相应的配置文件

[rhino@rhino181 ~]$

cd /home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/conf

修改settings.properties配置文件

(**仅显示部分需要修改的基础配置，若需要使用其他工具如：地图等，需修改配置文件其他项**)

|  |
| --- |
| jdbc.driver=com.mysql.jdbc.Driver  jdbc.url=jdbc:mysql://162.168.3.24:3306/cloud?useUnicode=true&amp;characterEncoding=utf8&amp;rewriteBatchedStatements=true&amp;allowMultiQueries=true  jdbc.username=rhino  jdbc.password=rhino  # \*\*\*\*\*\*\*\*\* oceanmind REST API settings \*\*\*\*\*\*\*  # when using agent,configure the following two configuration items separately  # when not using agent, configure them with the same value(actural address and port ).  oceanmind.ip.port.virtual=http://162.168.3.24:9999/oceanmind  oceanmind.ip.port.actural=http://162.168.3.24:9999/oceanmind  # \*\*\*\*\*\*\*\*\* 控件数据访问地址配置 \*\*\*\*\*\*\*  #component.url.frontend.host=http://10.45.134.176:18081/oceaneye  component.url.frontend.host=http://162.168.3.24:8889/oceaneye  component.url.backend.host=http://162.168.3.24:9999/oceanmind  component.url.shark.host=http://162.168.3.24:8888/shark  # \*\*\*\*\*\*\*\*\* GraphDb Connection Info \*\*\*\*\*\*\*  graphDb.jdbc.ip=162.168.3.24  graphDb.jdbc.port=7474  graphDb.jdbc.isAnonymous=false  graphDb.jdbc.username=neo4j  graphDb.jdbc.password=123456 |

1. 将hdfs安装目录下（$HADOOP\_HOME/etc/hadoop）的core-site.xml拷贝到/home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes下覆盖原文件：

[rhino@rhino001~]$ cp /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/core-site.xml /home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes

1. 将hdfs安装目录下（$HADOOP\_HOME/etc/hadoop）的hdfs-site.xml拷贝到/home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes下覆盖原文件：

[rhino@rhino001~]$ cp /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/hdfs-site.xml /home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes

1. 将hdfs安装目录下（$HADOOP\_HOME/etc/hadoop）的yarn-site.xml拷贝到/home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes下覆盖原文件：

[rhino@rhino001~]$ cp /home/rhino/hadoop-2.6.0-cdh5.7.0/etc/hadoop/yarn-site.xml /home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes

1. 将hdfs安装目录下（$SPARK\_HOME/conf）的hive-site.xml拷贝到/home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes下覆盖原文件：

[rhino@rhino001~]$ cp /home/rhino/spark-2.2.0-bin-hadoop2.6/conf/hive-site.xml

/home/rhino/apache-tomcat-8.0.35-oceaneye/webapps/oceaneye/WEB-INF/classes

以上操作完成后重启tomcat

[rhino@rhino181 ~]$cd /home/rhino/apache-tomcat-8.0.35-oceaneye/bin/

[rhino@rhino181 ~]$./shutdown.sh

[rhino@rhino181 ~]$./startup.sh

1. **安装ftp服务器**
   1. **安装**

* 安装sftp和ftp

[rhino@rhino001 ~]$ rpm -ivh vsftpd-3.0.2-10.el7.x86\_64 rpm

[rhino@rhino001 ~]$ rpm -ivh ftp-0.17-66.el7.x86\_64.rpm

* 1. **配置连接数**
* 以root用户登录，配置ftp最大连接数

[root@rhino001 ~]# vim /etc/vsftpd/vsftpd.conf

|  |
| --- |
| max\_clients=300#指明服务器总的客户并发连接数为300  max\_per\_ip=50#指明每个客户机的最大连接数为50 |

* 配置sftp最大连接数

[root@rhino001 ~]# vim/etc/ssh/sshd\_config

|  |
| --- |
| #MaxStartups 10:30:100  MaxStartups 100 |

* 重启sshd服务

[root@rhino001 ~]]#service sshd restart

* 1. **启动ftp服务**
* 启动服务

[root@rhino001 ~]]#service vsftpd restart

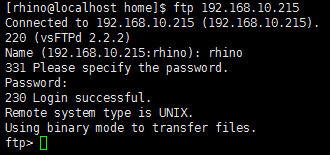
* 从其他ftp服务器连接到该服务器验证是否正常（假设rhino002机器下已经安装ftp服务器）

分别以ftp和sftp建立连接：

[rhino@rhino002 ~]$ftp rhino001 （验证ftp）

[rhino@rhino002 ~]$ sftp rhino001 （验证sftp）

输入用户名rhino，密码rhino，出现以下信息即为成功，输入quit、exit或bye断开连接。



1. **部署ElasticSearch5.1.1**

安装在所有X86机器上，下面以rhino001为例。

* 1. **安装ElasticSearch5.1.1**

以rhino用户解压ElasticSearch安装文件：Elasticsearch-5.1.1.tar.gz解压到/home/rhino/目录下。

[rhino@rhino001 ~]$ cd /home/rhino

[rhino@rhino001 ~]$ tar-zxvf elasticsearch-5.1.1.tar.gz

* 1. **修改配置**

ES服务使用9200、9300端口提供服务，安装前需要保证端口未被占用。

静态配置

根据实际服务器配置情况设置ES服务可以使用的内存，ES要求至少2g，实际生产环境应该配置30g。

[rhino@rhino001 ~]$vi /home/rhino/elasticsearch-5.1.1/config/jvm.options

|  |
| --- |
| -Xms30G  -Xmx30G |

[rhino@rhino001 ~]$vi /home/rhino/elasticsearch-5.1.1/config/elasticsearch.yml

|  |
| --- |
| #锁内存配置  bootstrap.memory\_lock: true  #配置服务器ip  network.host：192.168.10.1  #集群初始化节点列表  discovery.zen.ping.unicast.hosts: ["rhino001", "rhino002","rhino003"]  #elasticsearch数据存储路径，默认为安装目录下的data目录  path.data: /data01/esdata,/data02/esdata,/data03/esdata  #启用基于磁盘的分片分配，true为启用，false为禁止  cluster.routing.allocation.disk.threshold\_enabled: true  #节点中磁盘空间超过设定值，节点不再分配新的分片  cluster.routing.allocation.disk.watermark.low: 95%  #节点中磁盘空间超过设定值，节点开始尝试重新分配分片  cluster.routing.allocation.disk.watermark.high: 95%  #更新节点上磁盘使用信息的时间间隔  cluster.info.update.interval: 30s |

注：修改上面这个文件时，每个分号（:）后面需要有个空格。

**动态配置**

动态配置要在索引建好之后动态修改，命令如下：

|  |
| --- |
| curl -XPUT 'localhost:9200/index1/\_settings' -d '{"index.refresh\_interval":"10s"}' |

可以修改的配置项如下：

|  |
| --- |
| #索引刷新时间  index.refresh\_interval: 10s  #修改索引备份数，1表示集群中有2份数据  index.number\_of\_replicas: 1 |

* 1. **启动**

[rhino@rhino001 ~]$ /home/rhino/elasticsearch-5.1.1/bin/start-es.sh

然后通过ps -ef|grep elasticsearch命令或者安装目录下的logs目录查看进程是否启动成功。

* 1. **常用REST接口**

在shell命令行下，通过如下命令查看集群和索引

查看整个集群状态：

|  |
| --- |
| curl -XGET 'http://localhost:9200/\_cluster/stats?human&pretty' |

查看集群线程数，相当于集群负载：

|  |
| --- |
| curl -XGET http://localhost:9200/\_cat/thread\_pool?v |

查看索引分片数：

|  |
| --- |
| curl -XGET http://localhost:9200/\_cat/shards/index1?v |

查看文档结构：

|  |
| --- |
| curl -XGET 'localhost:9200/index1/\_mapping/type1' |

查看索引段：

|  |
| --- |
| curl -XGET 'http://localhost:9200/index1/\_segments?pretty' |

合并索引段：

|  |
| --- |
| curl -XPOST 'http://localhost:9200/index1/\_forcemerge?max\_num\_segments=5' |

* 1. **停止**

[rhino@rhino001 ~]$ /home/rhino/elasticsearch-5.1.1/bin/stop-es.sh

1. **附录一. 安装本地yum源**

[root@ rhino001 ~]# mkdir /mnt/cdrom/ ------》创建光盘挂载目录

假设/dev/cdrom为光盘所在目录

[root@ rhino001 ~]# mount /dev/cdrom /mnt/cdrom/ --------》挂载光盘

若挂载root目录下的镜像文件，则执行

mount -o loop /root/CentOS-6.5-x86\_64-bin-DVD1.iso /mnt/cdrom/

[root@ rhino001 ~]# cd /etc/yum.repos.d/ --------》切换到YUM配置目录

[root@ rhino001 yum.repos.d]# tar czf repo.tar.gz ./\* --------》将原来所有的文件进行打包备份

[root@ rhino001 yum.repos.d]# rm -f CentOS-Base.repo CentOS-Debuginfo.repo CentOS-Vault.repo --------》删除文件，只保留CentOS-Media.repo

[root@ rhino001 yum.repos.d]# vim CentOS-Media.repo

|  |
| --- |
| [c6-media]  name=CentOS-$releasever - Media  baseurl=file:///mnt/cdrom/ ##这里为本地源路径  gpgcheck=1  enabled=1##开启本地源  gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6 |

[root@ rhino001 yum.repos.d]#yum clean all --------》清除原先yum缓存