# 高创集群&项目部署指导书

# 装机要求

**操作系统：** Centos 7.5.1804

**磁盘划分：** 1. 除/，/boot，swap以外，剩余磁盘均挂载到/opt下

1. /boot 和swap规划完成后，剩余磁盘直接挂载至/目录下
2. 规划系统必要分区如/boot，swap后，剩余磁盘先不做分区和挂载，待装机成功后自由划分（不推荐）

**文件系统格式：必须统一使用ext4文件系统！！**

# 显卡驱动安装（标准平台执行）

### 下载驱动

根据nvidia显卡的具体型号，从官网上下载驱动，由于大数据现阶段申请的显卡统一为：

NVIDIA GXT 1080TI这款型号官网地址为：<http://www.geforce.cn/drivers>

### 安装编译环境（需要连接外网）

安装gcc、kernel-devel、kernel-headers

yum -y install gcc kernel-devel "kernel-devel-uname-r == $(uname -r)"

### 阻止nouveau模块的加载

修改/etc/modprobe.d/blacklist.conf文件

方法：在文件中添加blacklist nouveau，注释掉blacklist nvidiafb

blacklist.conf不存在时，执行以下命令：

echo -e "blacklist nouveau\noptions nouveau modeset=0" > /etc/modprobe.d/blacklist.conf

### 重新建立initramfs image文件

mv /boot/initramfs-$(uname -r).img /boot/initramfs-$(uname -r).img.bak

dracut /boot/initramfs-$(uname -r).img $(uname -r)

### 检查nouveau模块状态

重启节点：reboot now

在root用户下执行，查看并确保nouveau没有被加载

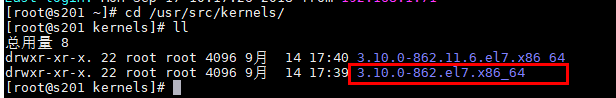
执行lsmod | grep nouveau无输出即可

### 安装驱动

执行 init 3 进入命令行模式（必须执行，否则安装驱动时会出错）

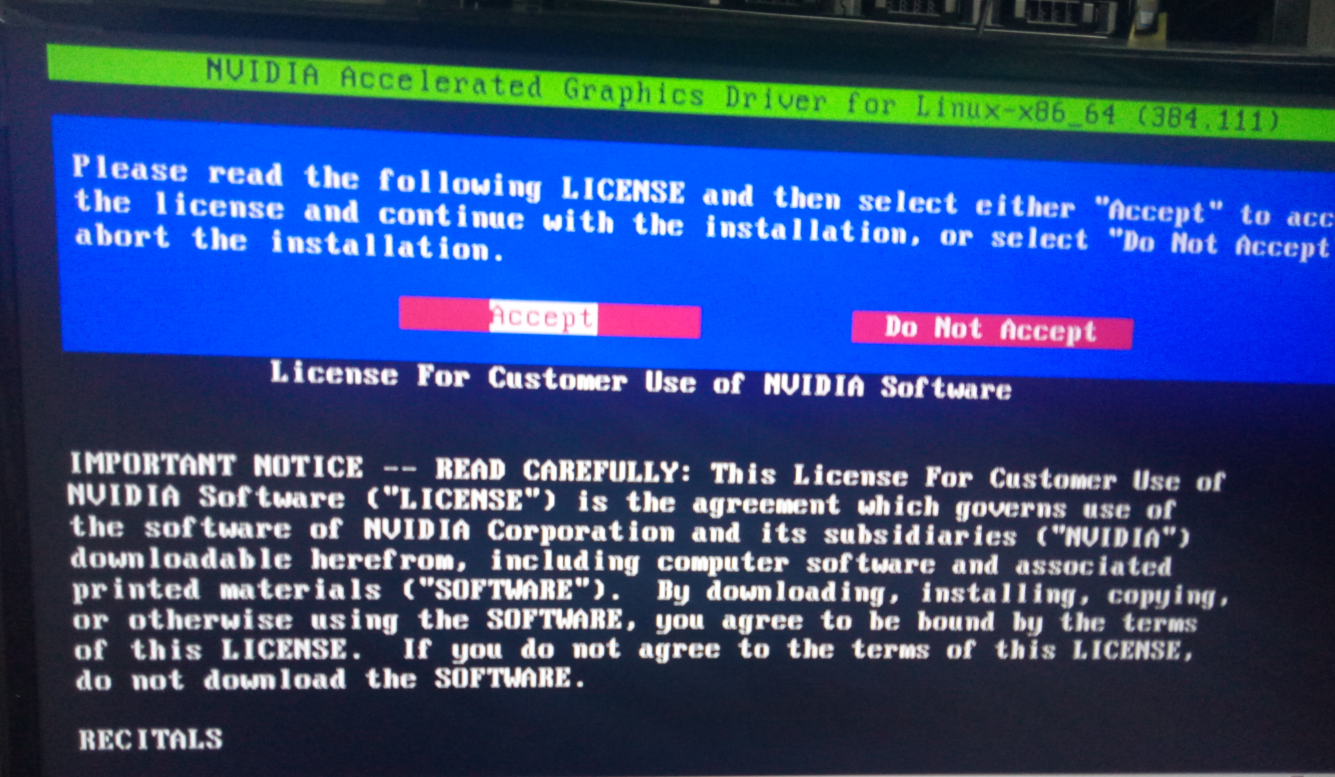
进入驱动所在目录执行命令

其中的xxxxx请根据驱动文件和/usr/src/kernels/下的文件来写



./NVIDIA-Linux-xxxxx.run --kernel-source-path=/usr/src/kernels/3.xxxxx-xxx.el7.x86\_64 -k $(uname -r)

点击accept后安装即可



# 系统环境准备

### 平台类型及对应节点数

|  |  |
| --- | --- |
| 平台类型 | 节点数 |
| 标准平台 | 4个节点及以上  部署过程参照[标准平台部署](#_标准平台搭建) |
| 虚拟机小型化平台 | 4个节点及以上  部署过程参照[虚拟机小型化平台部署](#_虚拟机小型化平台搭建) |
| 服务器小型化平台 | 1个节点  部署过程参照[服务器小型化平台部署](#_服务器小型化平台搭建) |

### NTP服务器搭建（服务器小型化平台无需执行此步骤）

服务器上默认已安装ntp服务，选定一台服务器作为server，其余几台为client，实现client机器从server端同步时间。

编辑/etc/ntp.conf文件

**Server端：**

新增行

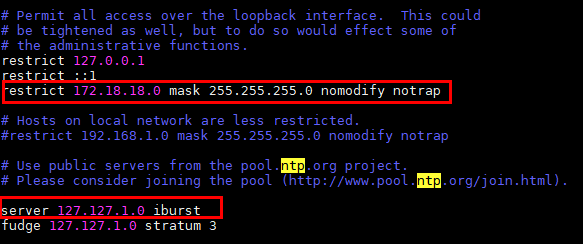
restrict 网段 mask 255.255.255.0 nomodify notrap

其中网段根据服务器实际网段设置

设置ntp服务器为自身（127.127.1.0）

server 127.127.1.0 iburst

fudge 127.127.1.0 stratum 3

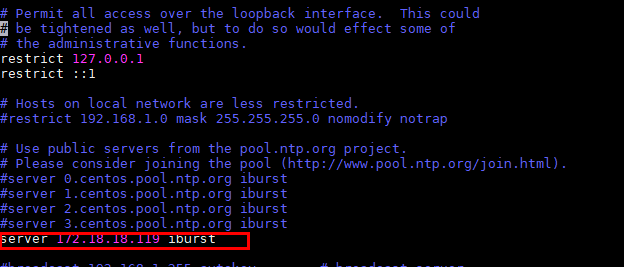


**Client端：**

新增行

设置ntp服务器为自身

server <server端IP> iburst



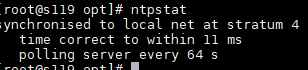
在所有节点上启动ntp服务

systemctl start ntpd 启动ntp服务

systemctl enable ntpd 把ntp服务设为开机启动

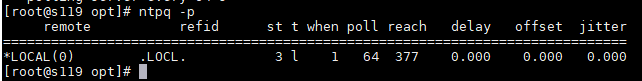
查看ntp服务器状态

ntpstat

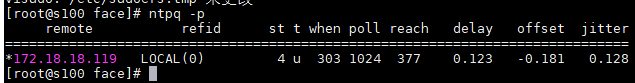


ntpq -p

**Server端输出：**



**Client端输出：**



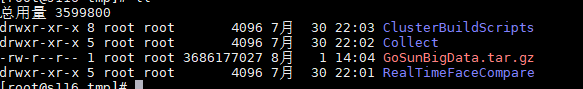
### 部署环境准备

#### 准备部署脚本及安装包

解压tar包，执行解压命令

解压命令：tar -zxvf GoSunBigDataDeploy.tar.gz

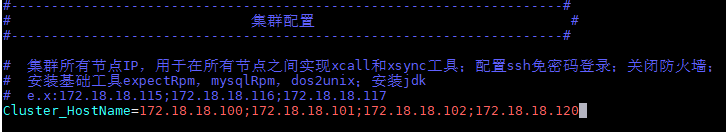
解压后目录结构如下：



#### 修改配置文件

进入/opt/GoSunBigDataDeploy/conf目录，修改cluster\_conf.properties

编辑配置文件，写上所有节点ip

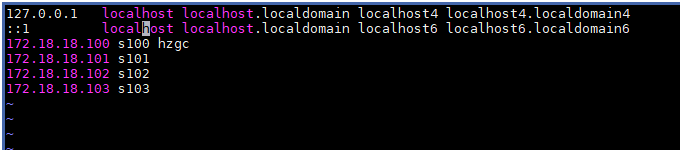


#### 修改hosts文件

配置主机名和主机ip映射

编辑/etc/hosts文件

以4节点为例，在文件结尾添加以下内容，并保存退出(hzgc可配在任意节点下)



#### 执行installEnv

进入install目录

cd /opt/GoSunBigDataDeploy/install

执行环境部署脚本

sh installEnv.sh

installEnv.sh包含：

1. 在当前节点上安装sshpass，dos2unix工具
2. 所有节点上安装expect工具
3. 配置所有节点（集群节点和ftp节点）间的免密登录
4. 分发/etc/hosts文件
5. 删除原有环境变量
6. 关闭所有节点的防火墙
7. 检查selinux状态，若为enable，则设置成disabled，需要手动重启

**服务器小型化平台在此步骤执行成功后重启节点**

**注意：**若是脚本安装出错，根据报错情况解决后重新执行脚本即可

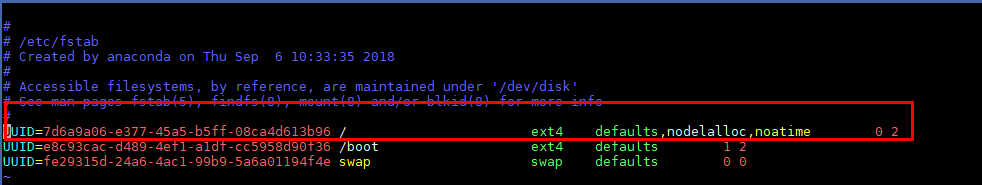
### 磁盘参数修改（服务器小型化平台无需执行此步骤）

编辑 /etc/fstab 文件，添加 nodelalloc 和noatime挂载参数，修改最后两个参数为0和2

**（必须添加，否则在安装TiDB时检测无法通过）**

**例：**

UUID=c51eb23b-195c-4061-92a9-3fad812cc12f /data1 ext4 defaults,nodelalloc,noatime 0 2



执行以下命令，如果文件系统为 ext4，并且挂载参数中包含 nodelalloc 表示生效：

# mount -t ext4

/dev/nvme0n1 on /data1 type ext4 (rw,noatime,nodelalloc,data=ordered)



**参数修改完成后需要重启节点**

# 标准平台搭建

## 标准分布式集群部署

### 集群安装准备

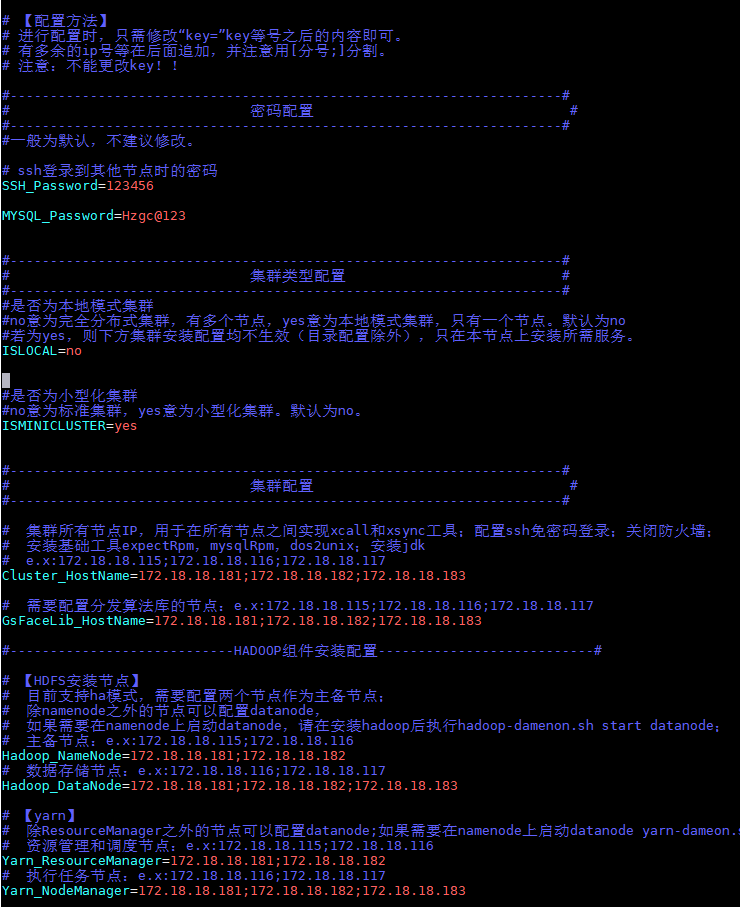
#### 修改集群安装配置文件

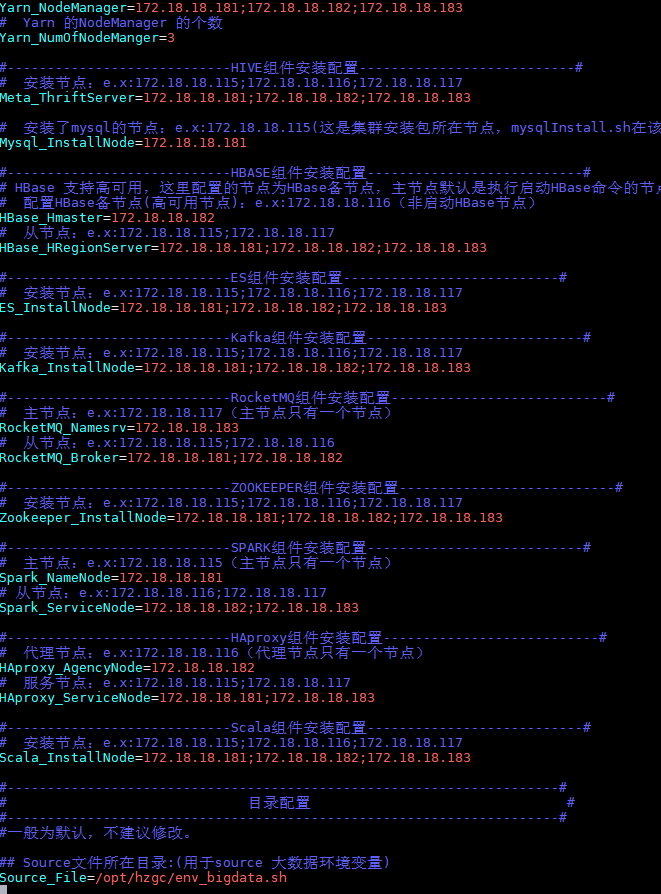
进入/opt/GoSunBigDataDeploy/conf目录，修改cluster\_conf.properties

**注意：ISLOCAL选项为no，ISMINICLUSTER选项为no**

vim cluster\_conf.properties

部分配置如下：





### 集群组件安装

进入install目录

cd /opt/GoSunBigDataDeploy/install/normal

执行组件安装脚本

sh installCompent.sh

installCompent.sh包含：

1. 检查selinux状态，若为enabled，会提示用户需改为disabled，并中断脚本执行

重启机器后再执行installCompent.sh即可

1. 安装并配置mysql
2. 安装并配置jdk
3. 安装并配置zookeeper，hadoop，hbase，hive，scala，kafka，spark，rocketmq，haproxy，elastic，azkaban，kibana，phoenix
4. 配置环境变量
5. 配置组件的日志目录
6. 根据集群类型和机器配置调整yarn的相关配置项

**注意：**若是安装出错会中断脚本执行，根据报错解决后重新执行命令

### 启动停止集群服务

进入服务脚本目录，执行对应服务的启动停止脚本

#### 启动集群服务

##### 启动zookeeper

cd /opt/GoSunBigDataDeploy/service

sh zookeeperStart.sh

脚本输出如下：



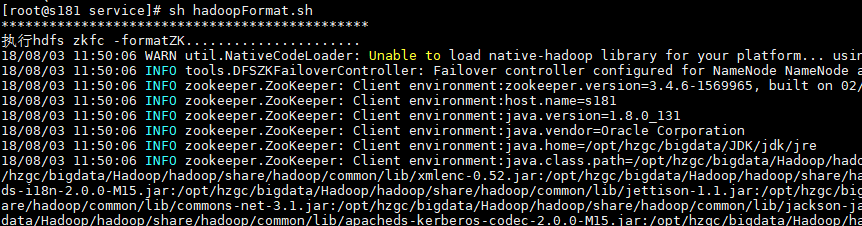
##### 启动hadoop

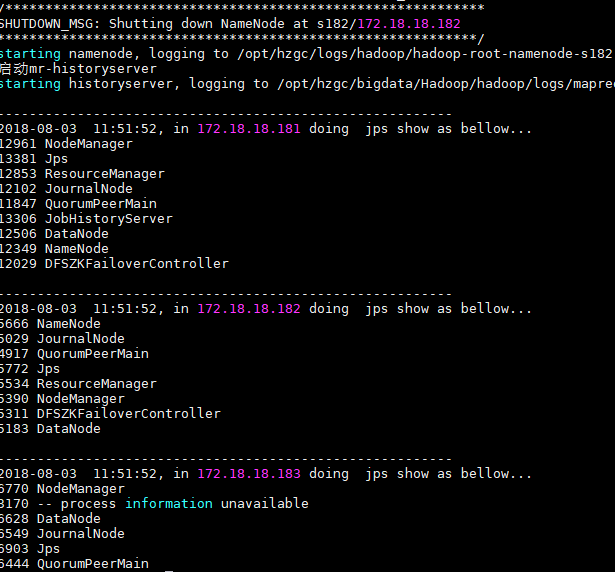
cd /opt/GoSunBigDataDeploy/service

第一次启动时执行hadoopFormat.sh，停止后启动执行hadoopStart.sh

sh hadoopFormat.sh

脚本输出如下：



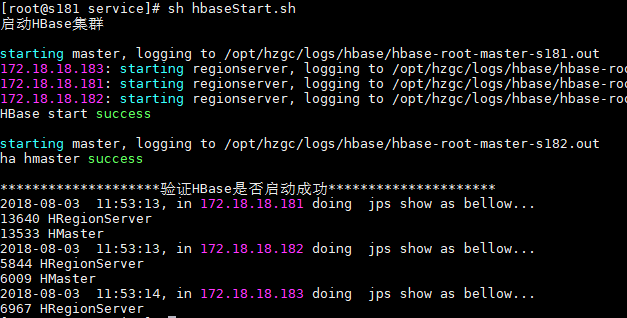


##### 启动hbase

cd /opt/GoSunBigDataDeploy/service

sh hbaseStart.sh

脚本输出如下：

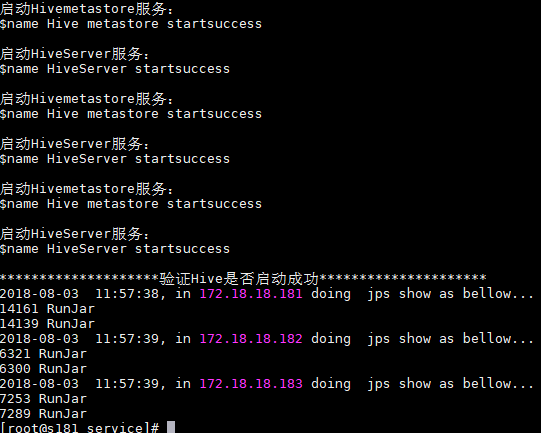


##### 启动hive

cd /opt/GoSunBigDataDeploy/service

sh hiveStart.sh

脚本输出如下：

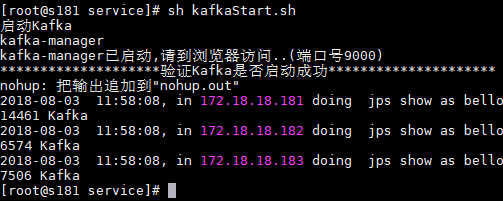


##### 启动kafka

cd /opt/GoSunBigDataDeploy/service

sh kafkaStart.sh

脚本输出如下：



##### 使用spark jdbc

第一次使用spark请先在hdfs创建sparkJobHistory

/opt/hzgc/bigdata/Hadoop/hadoop/bin/hdfs dfs -mkdir /sparkJobHistory

命令输出如下：



##### 启动thriftserver服务

进入到/opt/GoSunBigDataDeploy/service目录下执行sh thriftServerStart.sh

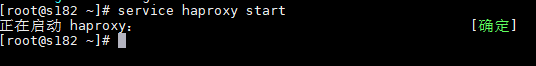
使用方法：sh thriftServerStart.sh1G 1G 1 1 1（请根据实际情况配置），这五个参数分别表示：driver\_memory；executor\_memory；driver\_cores；executor\_cores；num\_executors

##### 启动Haproxy

切换至Haproxy代理节点（一般为第二个节点，如s182），（在任意目录下均可）

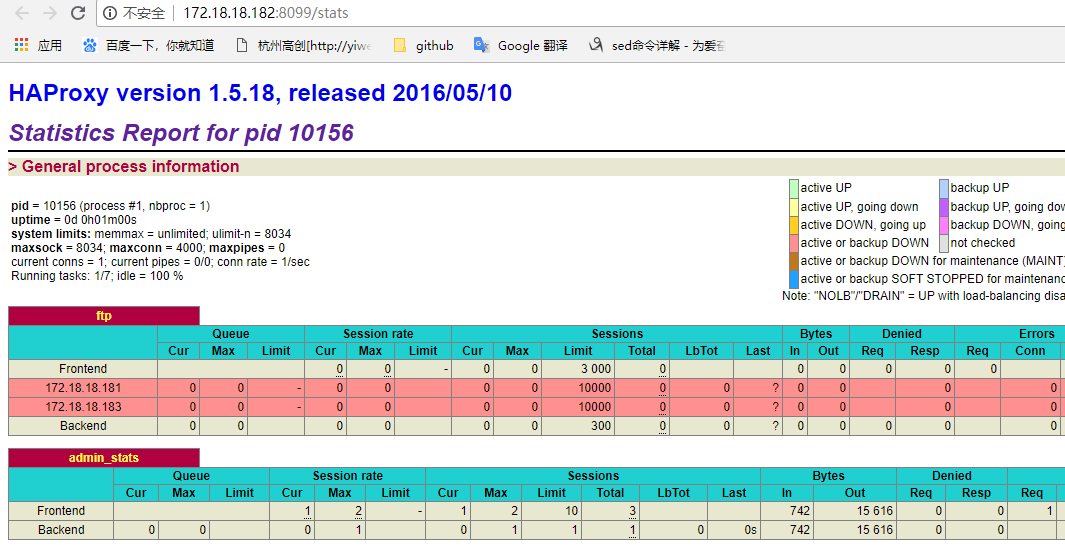
service haproxy start

输出如下：



在浏览器上验证是否启动：

172.18.18.106:8099/stats

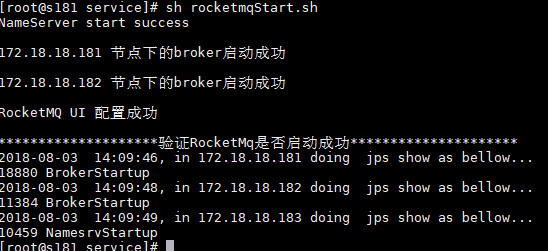


##### 启动rocketmq

cd /opt/GoSunBigDataDeploy/service

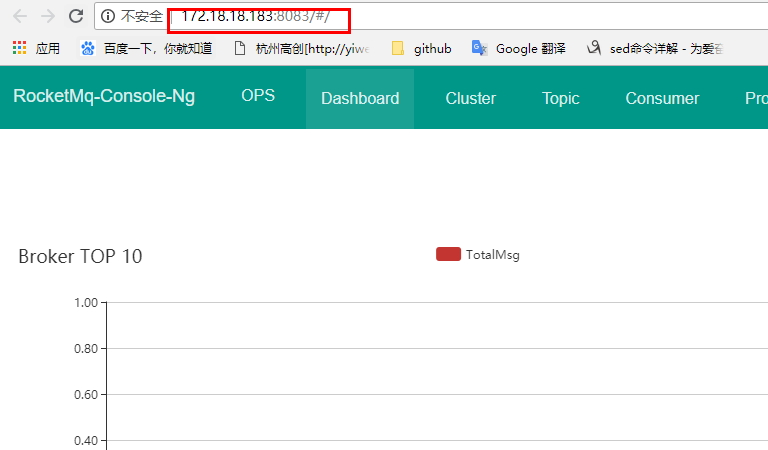
sh rocketmqStart.sh

脚本输出如下：



在浏览器上验证是否启动：

172.18.18.183:8083

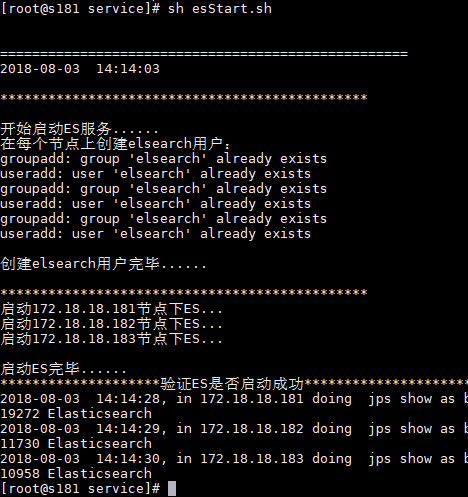


##### 启动elasticsearch

cd /opt/GoSunBigDataDeploy/service

sh esStart.sh

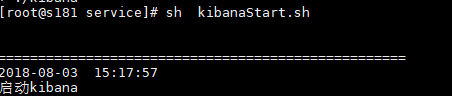
脚本输出如下：



##### 启动kibana

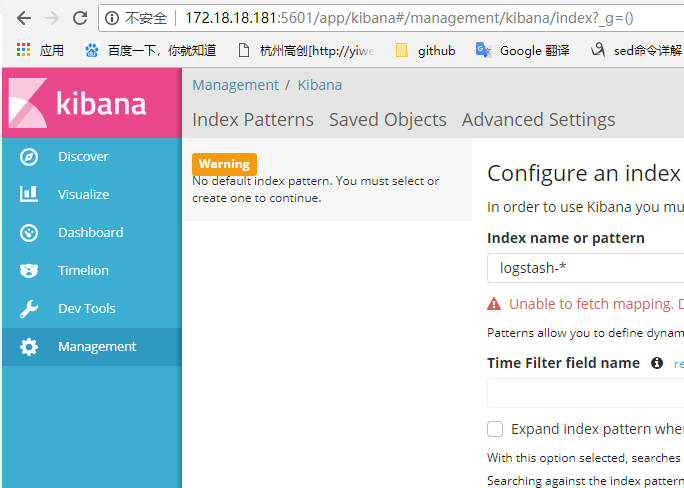
cd /opt/GoSunBigDataDeploy/service

sh kibanaStart.sh



在浏览器中验证是否启动成功：

<http://172.18.18.181:5601/>

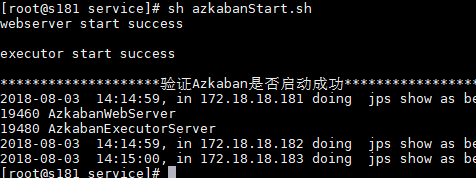


##### 启动Azkaban(在安装mysql所在节点启动)

cd /opt/GoSunBigDataDeploy/service

sh azkabanStart.sh

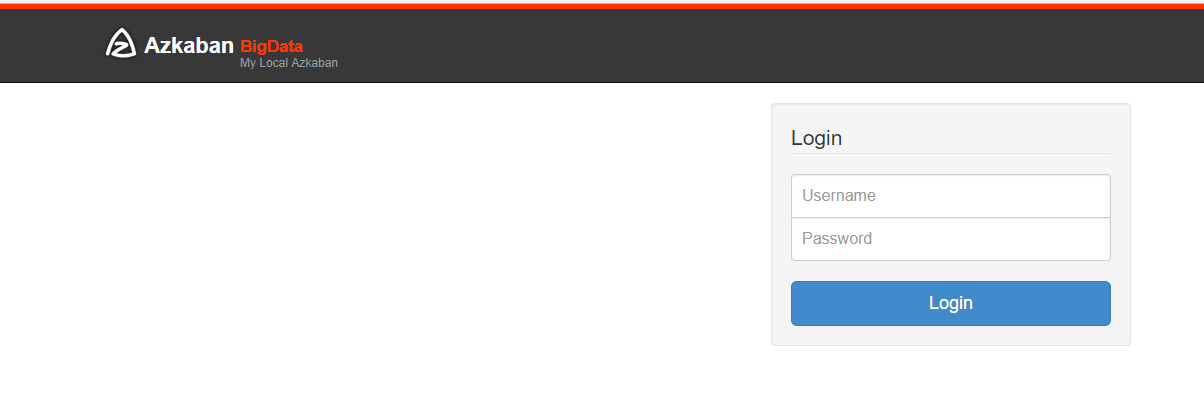
脚本输出如下：



在浏览器中验证是否启动成功：

[https://172.18.18.181:8443/stats](https://172.18.18.105:8443/stats)

用户名：admin 密码：admin

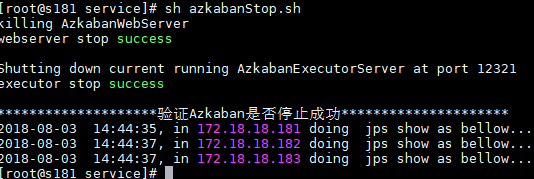


#### 停止集群服务

##### 停止azkaban

cd /opt/GoSunBigDataDeploy/service

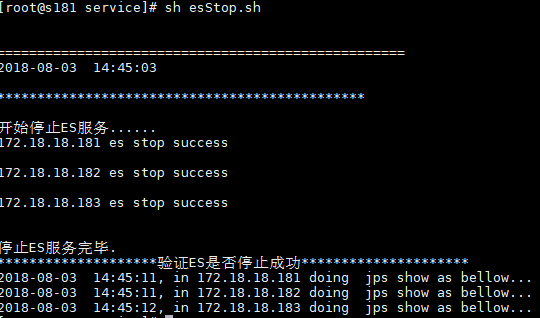
sh azkabanStop.sh



##### 停止elasticsearch

cd /opt/GoSunBigDataDeploy/service

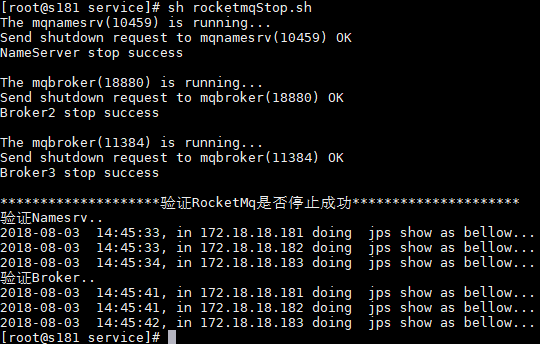
sh esStop.sh



##### 停止rocketmq

cd /opt/GoSunBigDataDeploy/service

sh rocketmqStop.sh



##### 停止kafka

cd /opt/GoSunBigDataDeploy/service

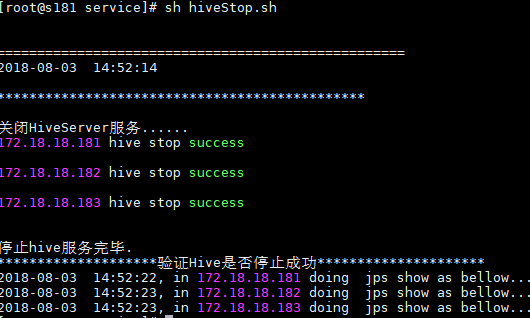
sh kafkaStop.sh



##### 停止hive

cd /opt/GoSunBigDataDeploy/service

sh hiveStop.sh



##### 停止hbase

cd /opt/GoSunBigDataDeploy/service

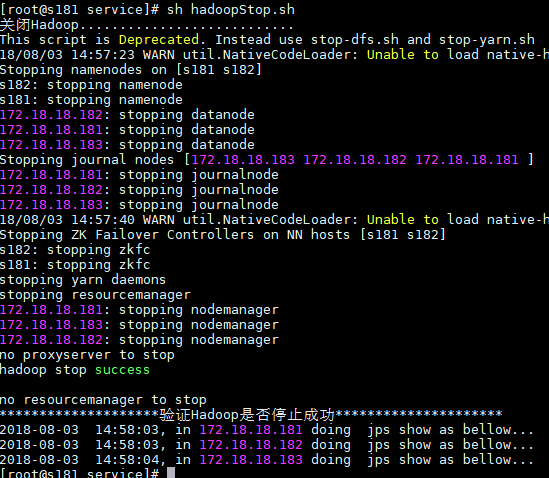
sh hbaseStop.sh



##### 停止hadoop

cd /opt/GoSunBigDataDeploy/service

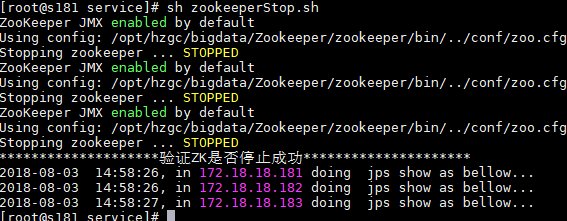
sh hadoopStop.sh



##### 停止zookeeper

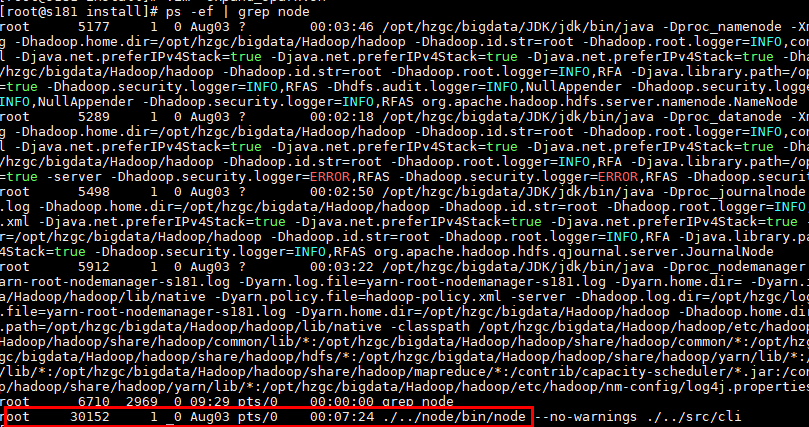
cd /opt/GoSunBigDataDeploy/service

sh zookeeperStop.sh



##### 停止kibana

ps -ef | grep node



Kill -9 pid(进程号)

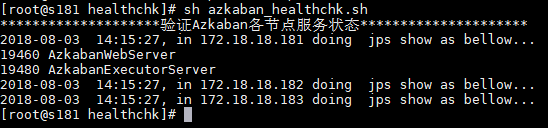
### 集群服务检查

进入服务检查目录，执行对应服务的进程检查脚本

#### Azkaban服务检查

cd /opt/GoSunBigDataDeploy/healthchk

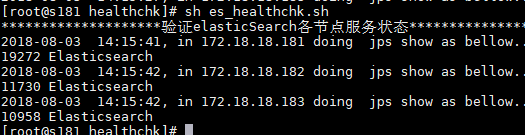
sh azkaban\_healthchk.sh



#### Elastic服务检查

cd /opt/GoSunBigDataDeploy/healthchk

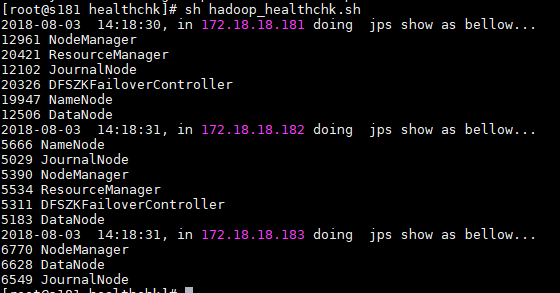
sh es\_healthchk.sh



#### Hadoop服务检查

cd /opt/GoSunBigDataDeploy/healthchk

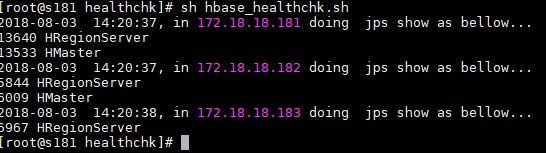
sh hadoop\_healthchk.sh



#### HBase服务检查

cd /opt/GoSunBigDataDeploy/healthchk

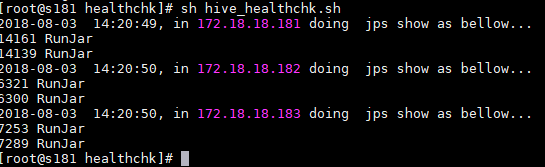
sh hbase\_healthchk.sh



#### Hive服务检查

cd /opt/GoSunBigDataDeploy/healthchk

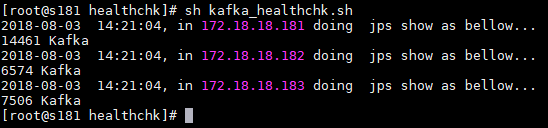
sh hive\_healthchk.sh



#### Kafka服务检查

cd /opt/GoSunBigDataDeploy/healthchk

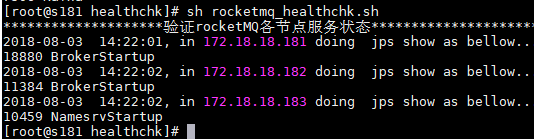
sh kafka\_healthchk.sh



#### Rocketmq服务检查

cd /opt/GoSunBigDataDeploy/healthchk

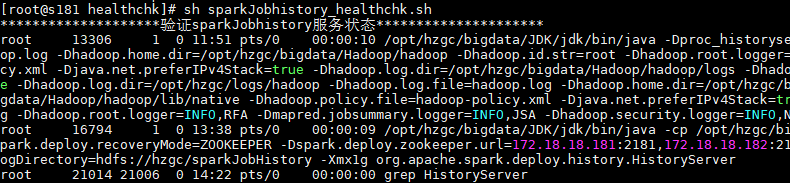
sh rocketmq\_healthchk.sh



#### Sparkjobhistory服务检查

cd /opt/GoSunBigDataDeploy/healthchk

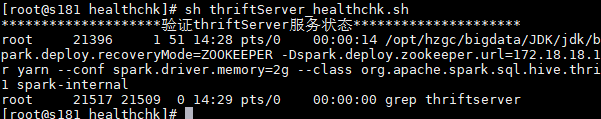
sh sparkJobhistory\_healthchk.sh



#### Thriftserver服务检查

cd /opt/GoSunBigDataDeploy/healthchk

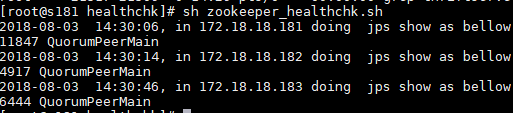
sh thriftServer\_healthchk.sh



#### Zookeeper服务检查

cd /opt/GoSunBigDataDeploy/healthchk

sh zookeeper\_healthchk.sh



### 集群扩展

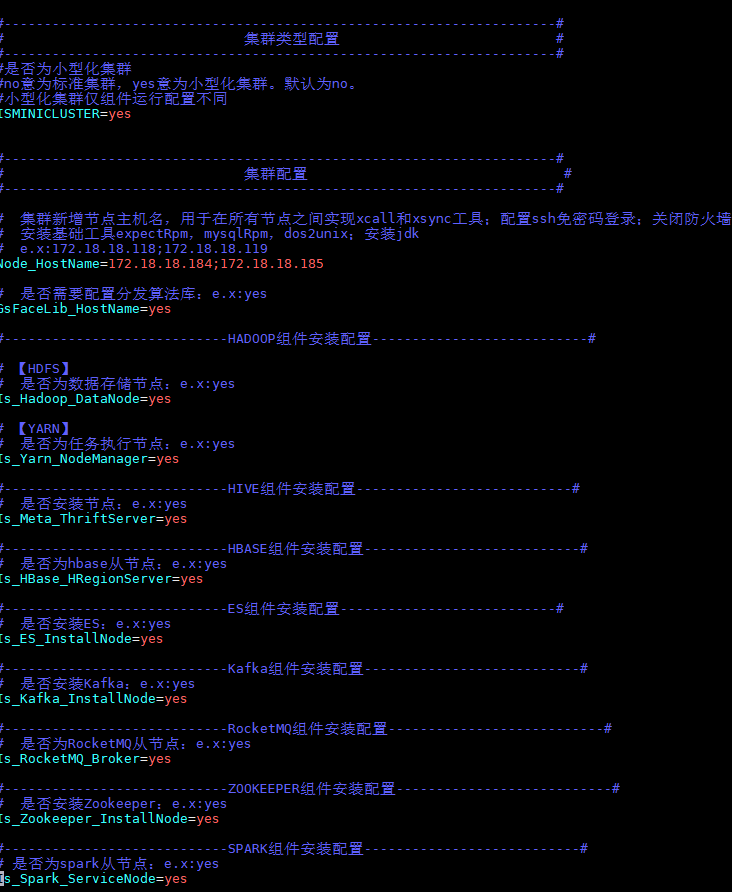
进入集群扩展模块目录，修改配置文件，依次执行扩展脚本，服务启动脚本（部分服务不支持动态扩展，需重启整个集群后才生效）。

#### 修改扩展配置文件

cd /opt/GoSunBigDataDeploy/expand/conf

vim expand\_conf.properties

将需要安装的组件的key改为yes即可，不需要的改为no，详细说明见配置文件注释



#### 配置主机名和主机ip映射

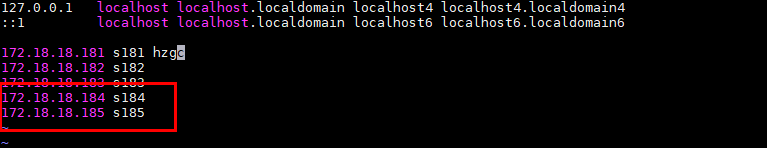
修改/etc/hosts文件

执行命令：vim /etc/hosts

在文件结尾添加以下内容，并保存退出(hzgc可配在任意节点下)

172.18.18.184 s184

172.18.18.185 s185



#### 执行扩展节点环境部署脚本

cd /opt/GoSunBigDataDeploy/expand/install

执行expandEnv.sh

expandEnv.sh包含：

1. 扩展节点安装dos2unix，expect
2. 配置原有节点和新增节点的免密登录
3. 分发新的host文件
4. 删除扩展节点上的原有环境变量
5. 关闭扩展节点防火墙
6. 在扩展节点上安装jdk

#### 执行扩展节点服务部署脚本

cd /opt/GoSunBigDataDeploy/expand/install

执行expandCompent.sh

expendCompent.sh包含：

1. 安装hadoop中的datanode，resourcemanager服务
2. 安装zookeeper服务
3. 安装hbase中的regionserver服务
4. 安装hive，kafka，scala，spark，rocketmq，es服务
5. 配置环境变量和服务启动目录
6. 将节点上服务安装情况同步到主配置文件cluster\_conf.properties中

#### 启动扩展节点服务

进入扩展节点服务启动目录，执行服务启动脚本（若在目录下没有找到对应服务的启动脚本，则说明该服务需要集群重启生效）

cd /opt/GoSunBigDataDeploy/expand/service

##### 启动datanode

sh start\_datanode.sh

##### 启动elasticSearch

sh start\_es.sh

##### 启动hive

sh start\_hive.sh

##### 启动nodemanager

sh start\_nodemanager.sh

##### 启动rocketmq

sh start\_rocketmq.sh

### 部署TiDB

#### 安装依赖的rpm包

执行命令

sh /opt/GoSunBigDataDeploy/tidb-ansible/ansible-denpendency/install\_ansible\_system\_rpms.sh

#### 创建tidb用户及互信

useradd -m -d /home/tidb tidb

设置密码

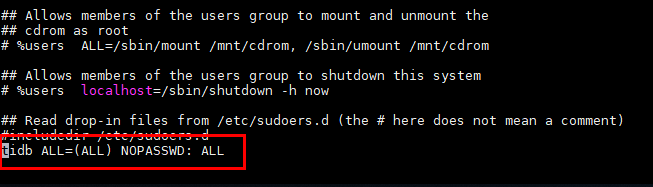
passwd tidb

配置 tidb 用户 sudo 免密码

执行命令 visudo

在文件末尾加上

tidb ALL=(ALL) NOPASSWD: ALL



生成tidb用户的 ssh key:

切换用户后生成秘钥文件

su - tidb

ssh-keygen -t -rsa

成功后,私钥文件为 /home/tidb/.ssh/id\_rsa

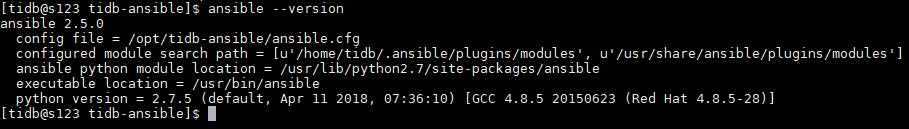
公钥文件为 /home/tidb/.ssh/id\_rsa.pub

#### 安装Ansible

进入目录 tidb-ansible/ansible-install

执行install\_ansible.sh

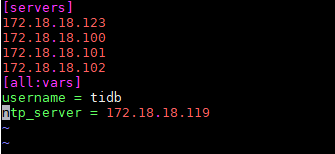
安装完成后可以通过ansible --verson查看版本



#### 配置tidb用户的免密登录

修改hosts.ini文件

servers下写上所有需要部署TiDB的节点ip

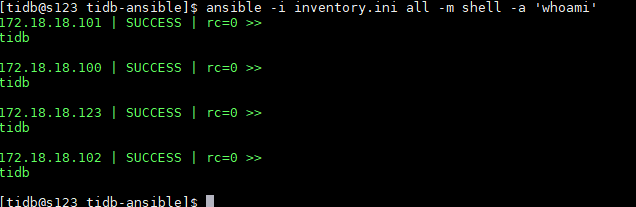


执行命令，会根据配置文件去目标机器上创建用户并配置tidb用户的免密登录

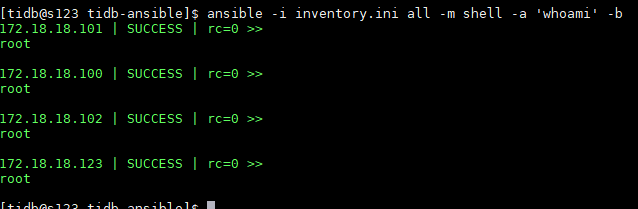
ansible-playbook -i hosts.ini create\_users.yml -u root -k

验证互信是否成功：

ansible -i inventory.ini all -m shell -a 'whoami'



验证是否可以从tidb用户免密切换到root用户



#### 配置CPRfreq调节器模式

查看系统支持的调节器模式

cpupower frequency-info --governors



如果返回为Not available

则跳过调节器配置



所有节点需要统一调节器模式为**performance**

设置本节点的调节器模式

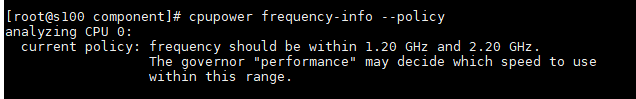
cpupower frequency-set --governor performance

根据配置文件设置所有节点的调节器模式

ansible -i hosts.ini all -m shell -a "cpupower frequency-set --governor performance" -u tidb -b

查看系统当前调节器模式

cpupower frequency-info --policy



#### 配置inventory.ini文件

**配置原则：**

PD节点必须为奇数个（1个或3个）

TiKV节点为3个

TiDB节点至多2个，或1个

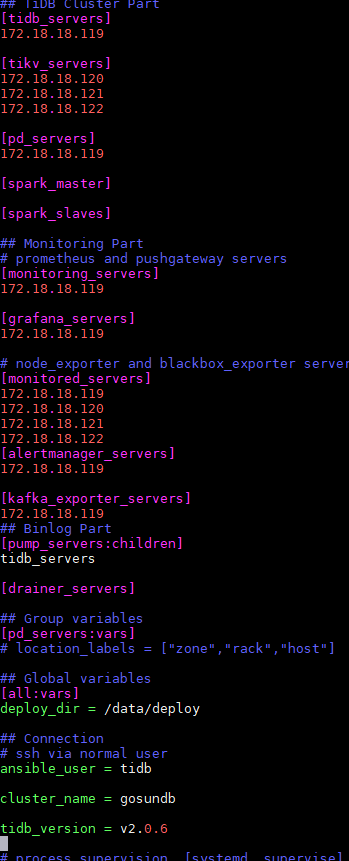
TiDB，PD节点不能与TiKV节点为同一个

TiDB节点和PD节点可以是同一个

**参数说明：**

|  |  |
| --- | --- |
| 参数名 | 参数值 |
| [tidb\_servers] | 部署tidb服务的节点ip |
| [tikv\_servers] | 部署tikc服务的节点ip |
| [pd\_servers] | 部署pd服务的节点ip |
| [monitoring\_servers] | 配置为tidb节点 |
| [grafana\_servers] | 配置为tidb节点 |
| [monitored\_servers] | 配置tidb，tikv，pd节点ip |
| [alertmanager\_servers] | 配置tidb节点ip |
| [kafka\_exporter\_servers] | 配置tidb节点ip |
| deploy\_dir | 根据磁盘挂载及参数修改情况配置，参数修改见[磁盘参数修改](#_磁盘参数修改（服务器小型化平台无需执行此步骤）)  例：修改参数的磁盘挂载在/opt下，配置为deploy\_dir=/opt/deploy |
| ansible\_user | tidb |
| cluster\_name | gosundb |

例如：



#### 启动TiDB

启动命令：

ansible-playbook start.yml

#### 测试TiDB是否可用

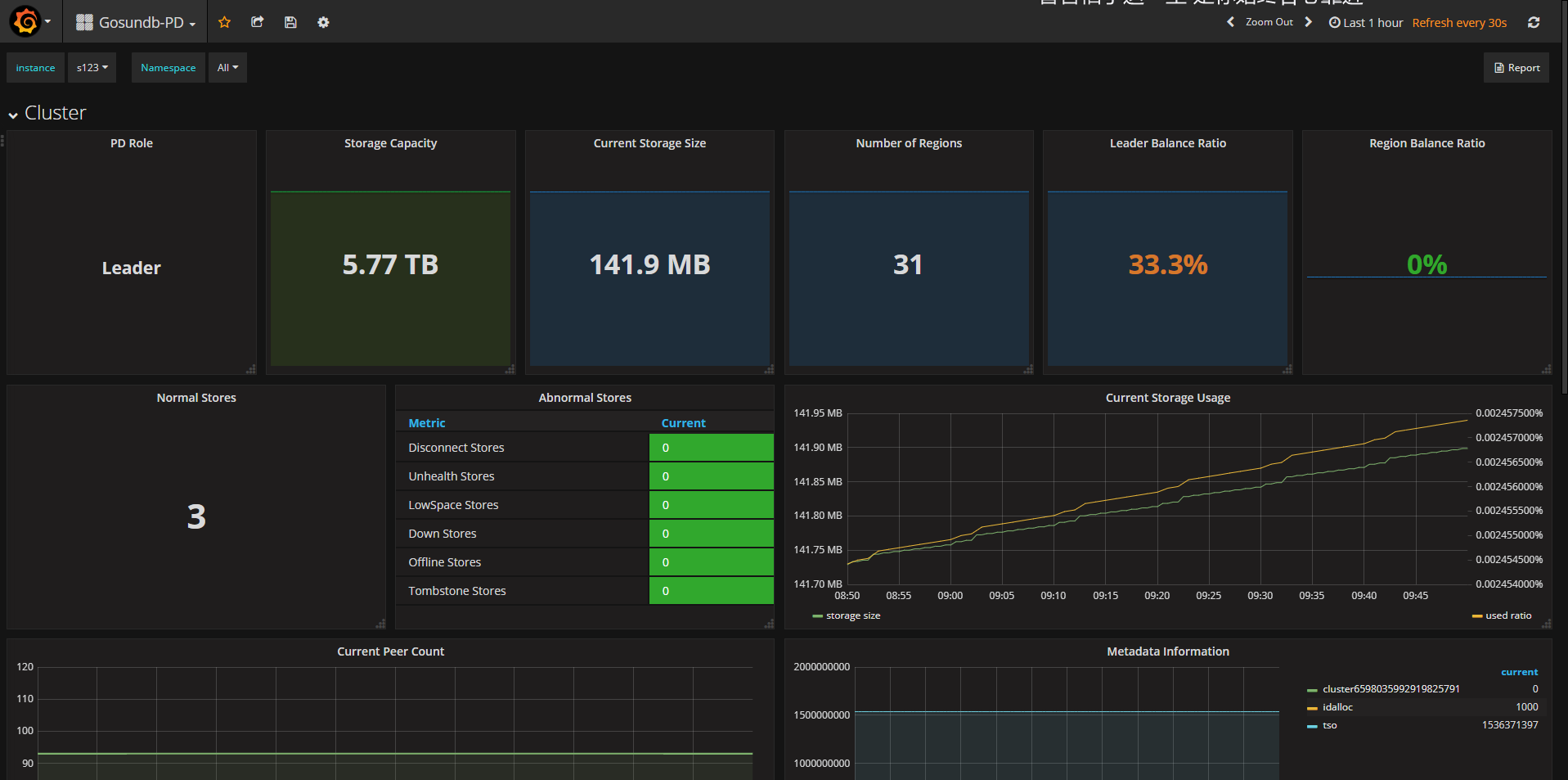
mysql -u root -h IP -P 4000

使用方法和mysql一样

监控页面地址：

http://<TiDB节点IP>:3000

默认帐号密码是：admin/admin



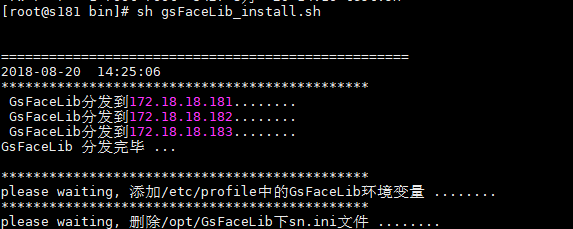
## 项目部署

### 部署算法库

#### 分发配置

cd /opt/GoSunBigDataDeploy/project/bin/

sh gsFaceLib\_install.sh

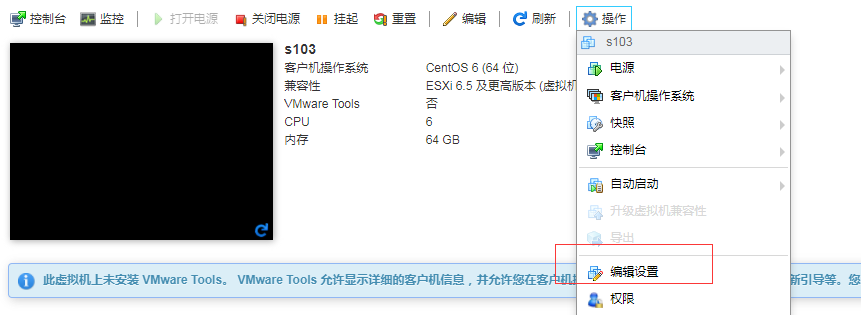


#### 添加加密狗

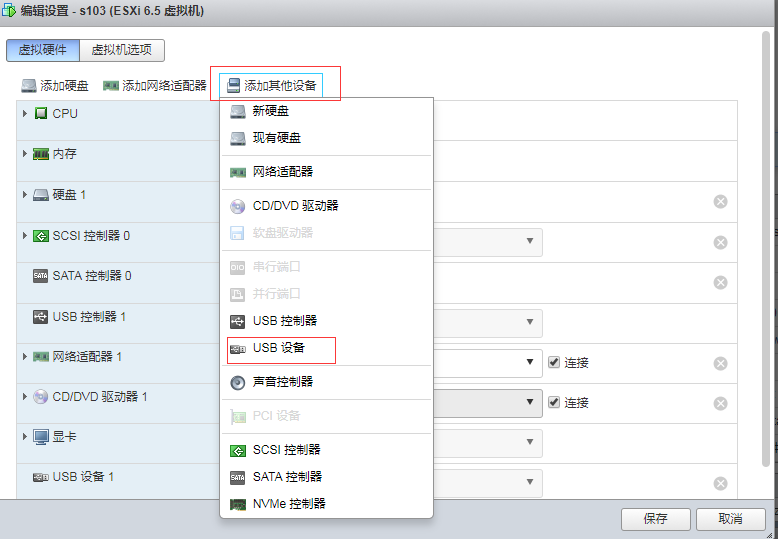
##### 添加加密狗驱动（vmware情况下执行）

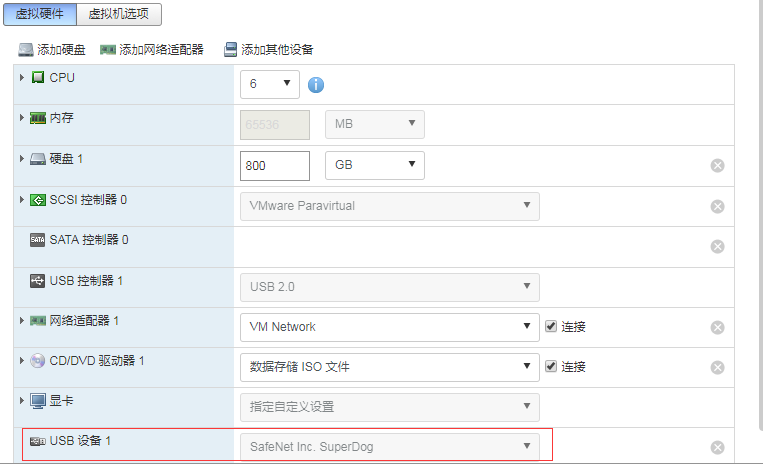
在对应的服务器中添加加密狗

登录对应服务器的vmware，编辑对应虚拟机设置：



添加USB控制器，后选择添加加密狗：





##### 安装加密狗驱动（每台需要配置算法的服务器下执行）

cd /opt/GsFaceLib/driver/

rpm -ivh superdog-1.0-1.i386.rpm



#### 配置算法库

##### 修改图片解析方式

cd /opt/GsFaceLib/

vim facelib.config

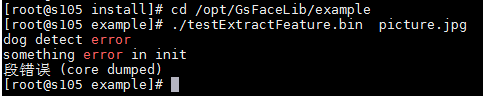
修改为device=CPU



##### 进入/opt/GsFaceLib/example目录下

cd /opt/GsFaceLib/example

./testExtractFeature.bin picture.jpg



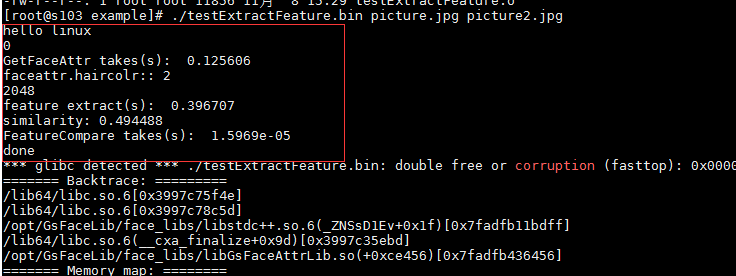
##### 生成新的sn.ini文件

将新产生的sn.ini文件内容发给**算法组 周琛辉** 转成相应的密钥后保存到sn.ini

##### 继续进入/opt/GsFaceLib/example执行

cd /opt/GsFaceLib/example

./testExtractFeature.bin picture.jpg



### 部署Collect

#### 修改FTP配置文件

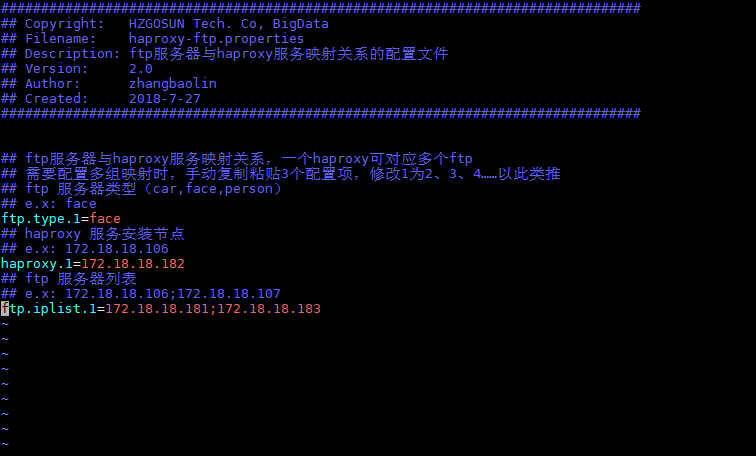
vim /opt/GoSunBigDataDeploy/project/conf/haproxy-ftp.properties

参数说明：

ftp.type 服务器存储数据类型：face（人脸），person（行人），car（汽车）

haproxy 代理节点ip

ftp.iplist ftp服务器列表



#### 启动FTP

cd /opt/Collect

sh bin/start-ftpserver.sh

查看Ftp是否启动成功

tailf logs/ftpserver.log

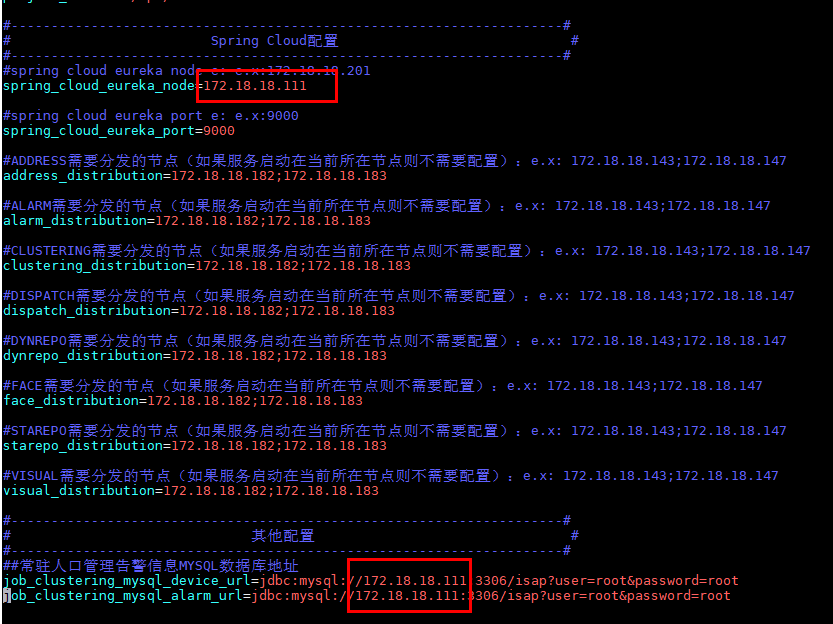


### 部署GoSunBigData

#### 修改配置文件

vim /opt/GoSunBigDataDeploy/project/conf/haproxy-ftp.properties

Spring\_cloud的ip和端口、mysql的ip向平台组同事获取



#### 执行部署脚本

cd /opt/GoSunBigDataDeploy/project/bin/

sh project\_distribute.sh

#### CLuster模块（建表，验证，启动job）

执行建表脚本

cd /opt/GoSunBigDataDeploy/project/bin/

sh create-all-table.sh

##### 验证Es索引

查看es索引是否创建成功

在浏览器中输入http://172.18.18.181:9200/dynamic?pretty

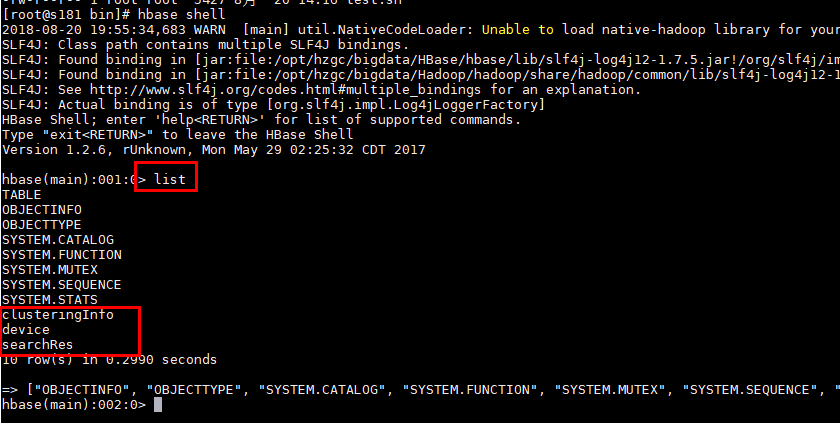


##### 验证Hbase表

执行下面命令查看hbase中表数据

source /opt/hzgc/env\_bigdata.sh

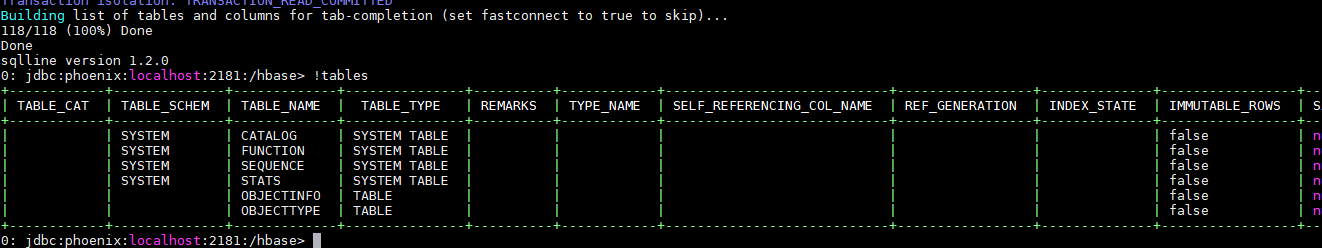
hbase shell



##### 验证Phoenix表

查看phoenix下的表

sqlline.py



##### 验证Spark

查看spark-udf

输入hive

show functions



##### 启动start-face-alarm-job.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-face-alarm-job.sh

##### 启动start-kafka-to-parquet.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-kafka-to-parquet.sh

##### 启动start-kmeans-clustering-job.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-kmeans-clustering-job.sh

##### 启动azkaban任务

cd /opt/GoSunBigDataDeploy/Cluster/azkaban/bin

sh create-schedule-job-to-zip.sh

cd zip 进入压缩包目录

ll 查看压缩包

TIM图片20180621100436

把压缩包拷贝到本地

进入Azkaban网页 <https://172.18.18.103:8443启动定时任务>

点击Create Project



填写Name和Description然后点击Create Project



点击Upload上传对应的三个zip包，上传成功后点击Upload





点击Execute Flow然后点击Schedule





填写对应的Time(可以往后填一点)和Date(选择当天就ok)，Recurrence中Days根据ZIP包的名称进行选择，然后点击Schedule，然后点击Continue就ok了



#### 微服务模块

common目录：address，face，visual

face目录：alarm，peoplemanager，dispatch，dynRepo，staRepo，dynperson，dyncar

服务启动方法以address为例：

启动address服务

cd /opt/GoSunBigDataDeploy/Service/common

cd address

sh bin/start-address.sh

tailf log/2018-06-22.address.0.log 查看服务是否启动成功



alarm，peoplemanager，dispatch，dynRepo，face，staRepo，visual，dynperson，dyncar服务启动与上面步骤类似(服务启动完成后jps查看下对应的服务进程是否启动成功)

# 虚拟机小型化平台搭建

## 虚拟机分布式集群部署

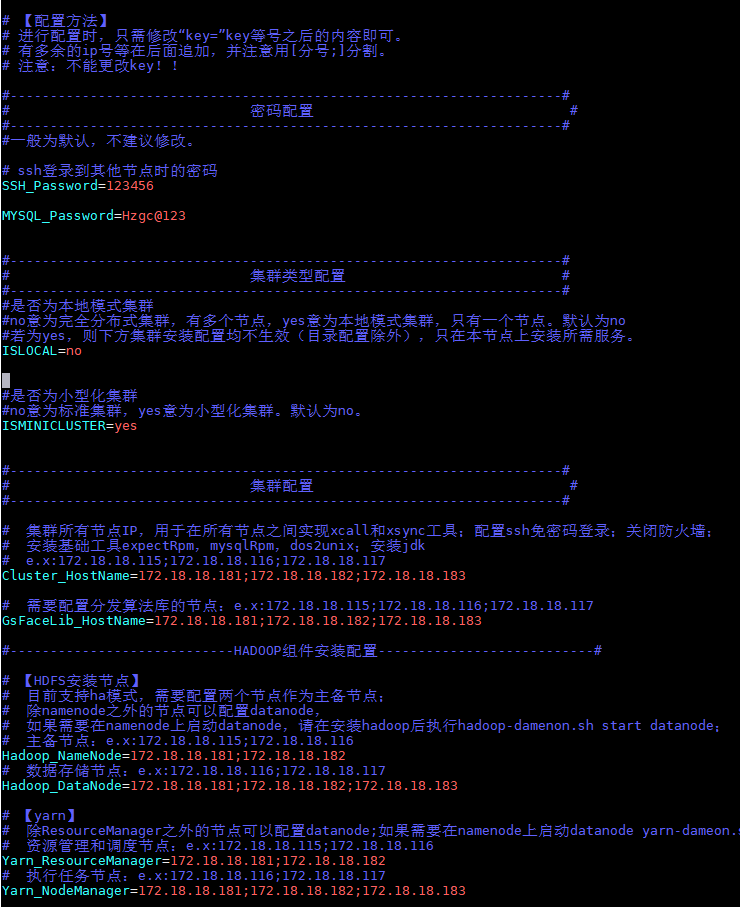
### 集群安装准备

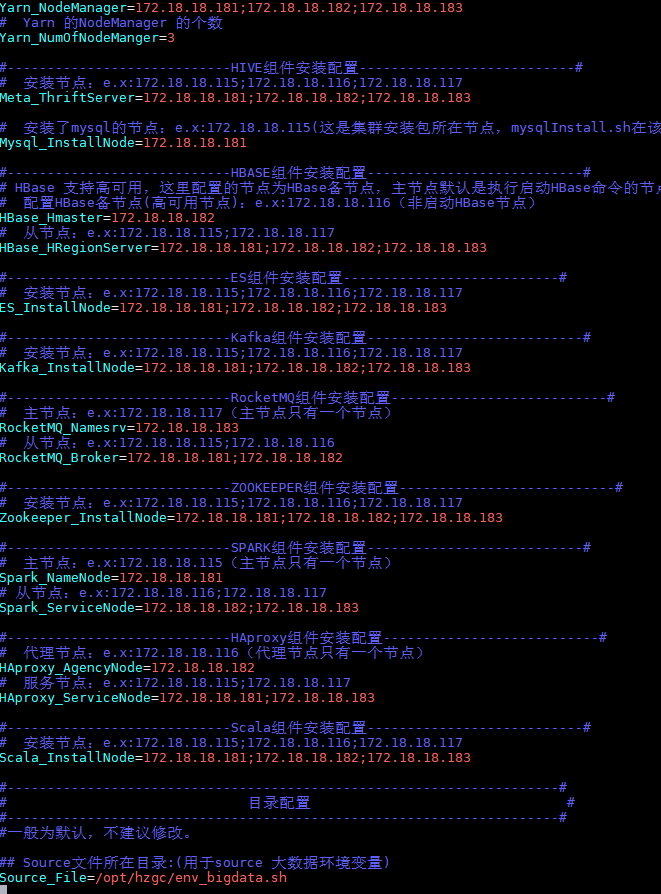
#### 修改集群安装配置文件

进入/opt/GoSunBigDataDeploy/conf目录，修改cluster\_conf.properties

**注意：ISLOCAL选项为no，ISMINICLUSTER选项为yes**

vim cluster\_conf.properties

部分配置如下：



### 集群组件安装

进入install目录

cd /opt/GoSunBigDataDeploy/install/normal

执行组件安装脚本

sh installCompent.sh

installCompent.sh包含：

1. 检查selinux状态，若为enabled，会提示用户需改为disabled，并中断脚本执行

重启机器后再执行installCompent.sh即可

1. 安装并配置mysql
2. 安装并配置jdk
3. 安装并配置zookeeper，hadoop，hbase，hive，scala，kafka，spark，rocketmq，haproxy，elastic，azkaban，kibana，phoenix
4. 配置环境变量
5. 配置组件的日志目录
6. 根据集群类型和机器配置调整yarn的相关配置项

**注意：**若是安装出错会中断脚本执行，根据报错解决后重新执行命令

### 启动停止集群服务

进入服务脚本目录，执行对应服务的启动停止脚本

#### 启动集群服务

##### 启动zookeeper

cd /opt/GoSunBigDataDeploy/service

sh zookeeperStart.sh

脚本输出如下：



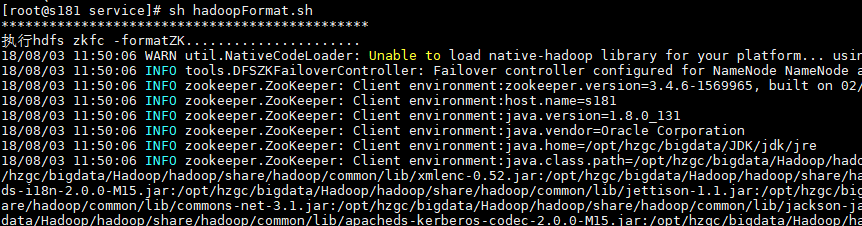
##### 启动hadoop

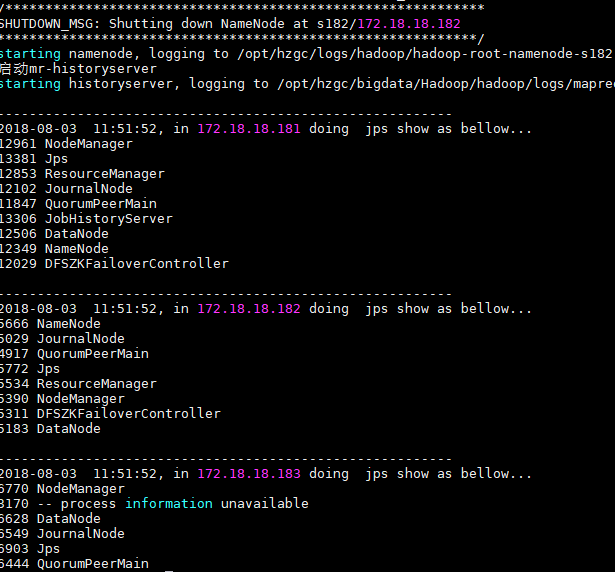
cd /opt/GoSunBigDataDeploy/service

第一次启动时执行hadoopFormat.sh，停止后启动执行hadoopStart.sh

sh hadoopFormat.sh

脚本输出如下：



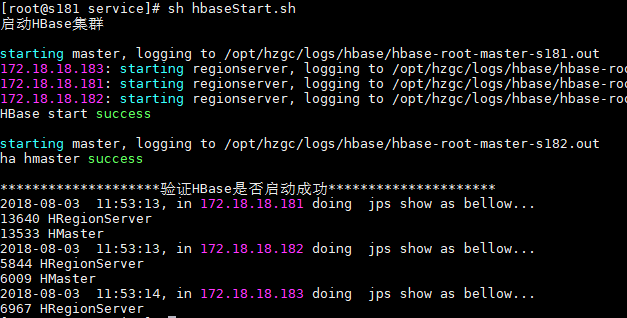


##### 启动hbase

cd /opt/GoSunBigDataDeploy/service

sh hbaseStart.sh

脚本输出如下：

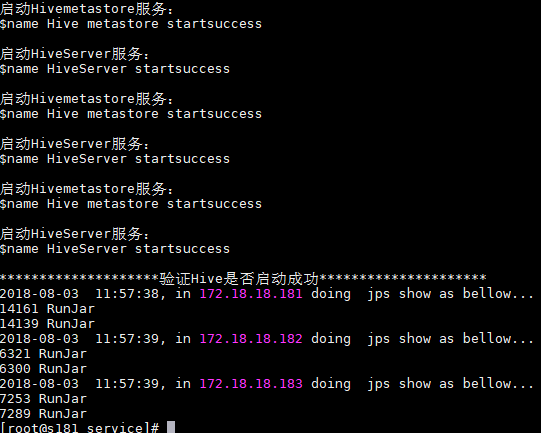


##### 启动hive

cd /opt/GoSunBigDataDeploy/service

sh hiveStart.sh

脚本输出如下：

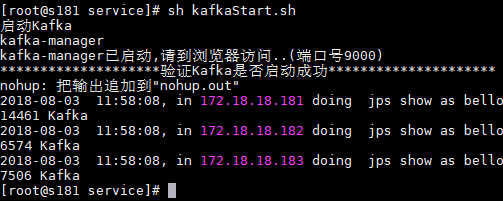


##### 启动kafka

cd /opt/GoSunBigDataDeploy/service

sh kafkaStart.sh

脚本输出如下：



##### 使用spark jdbc

第一次使用spark请先在hdfs创建sparkJobHistory

/opt/hzgc/bigdata/Hadoop/hadoop/bin/hdfs dfs -mkdir /sparkJobHistory

命令输出如下：



##### 启动thriftserver服务

进入到/opt/GoSunBigDataDeploy/service目录下执行sh thriftServerStart.sh

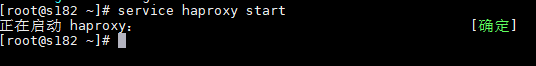
使用方法：sh thriftServerStart.sh1G 1G 1 1 1（请根据实际情况配置），这五个参数分别表示：driver\_memory；executor\_memory；driver\_cores；executor\_cores；num\_executors

##### 启动Haproxy

切换至Haproxy代理节点（一般为第二个节点，如s182），（在任意目录下均可）

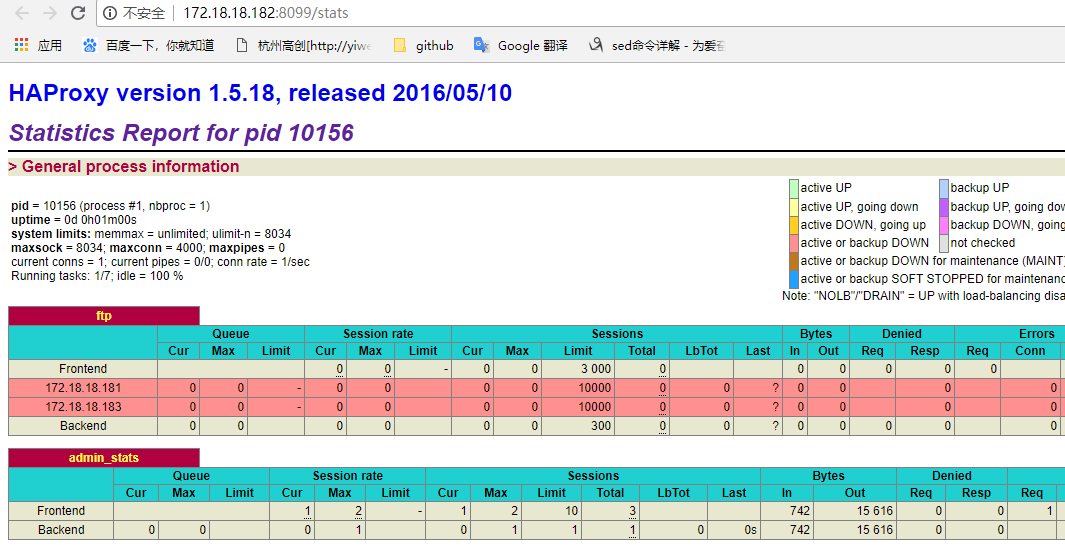
service haproxy start

输出如下：



在浏览器上验证是否启动：

172.18.18.106:8099/stats

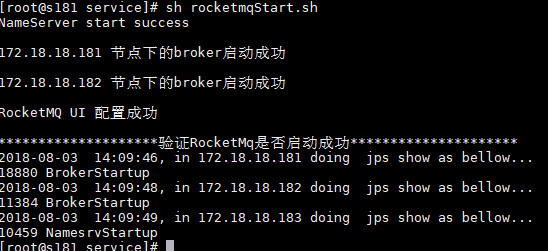


##### 启动rocketmq

cd /opt/GoSunBigDataDeploy/service

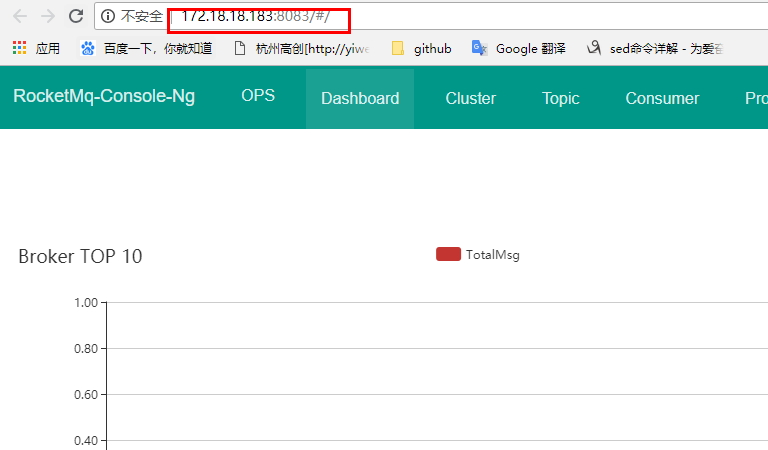
sh rocketmqStart.sh

脚本输出如下：



在浏览器上验证是否启动：

172.18.18.183:8083

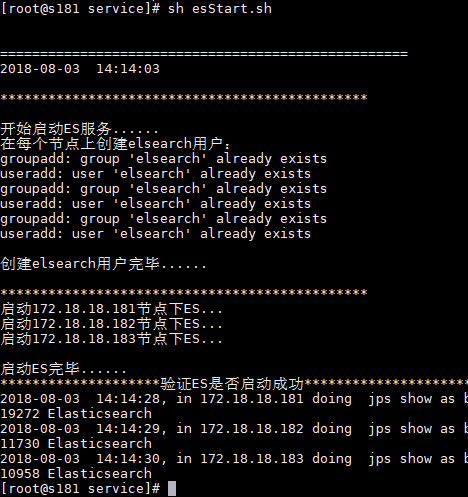


##### 启动elasticsearch

cd /opt/GoSunBigDataDeploy/service

sh esStart.sh

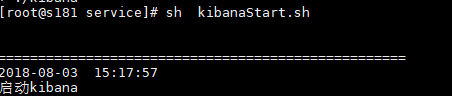
脚本输出如下：



##### 启动kibana

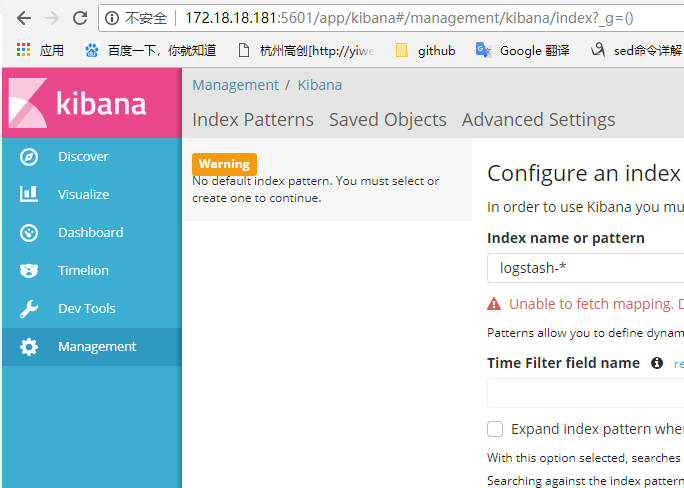
cd /opt/GoSunBigDataDeploy/service

sh kibanaStart.sh



在浏览器中验证是否启动成功：

<http://172.18.18.181:5601/>

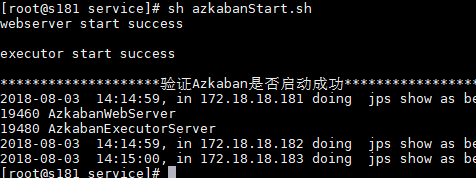


##### 启动Azkaban(在安装mysql所在节点启动)

cd /opt/GoSunBigDataDeploy/service

sh azkabanStart.sh

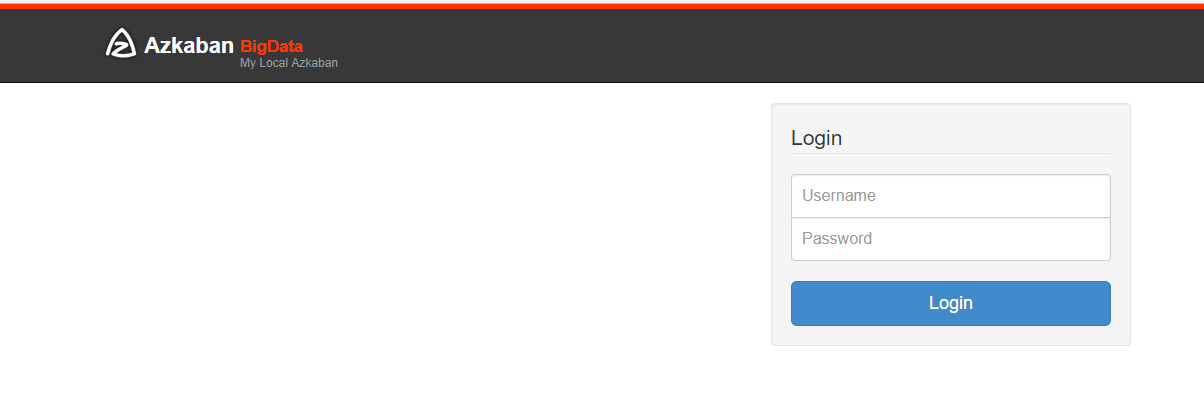
脚本输出如下：



在浏览器中验证是否启动成功：

[https://172.18.18.181:8443/stats](https://172.18.18.105:8443/stats)

用户名：admin 密码：admin

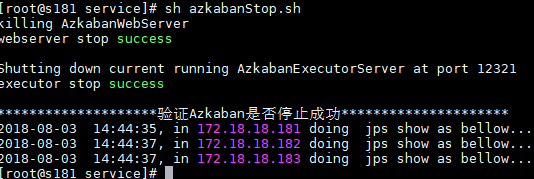


#### 停止集群服务

##### 停止azkaban

cd /opt/GoSunBigDataDeploy/service

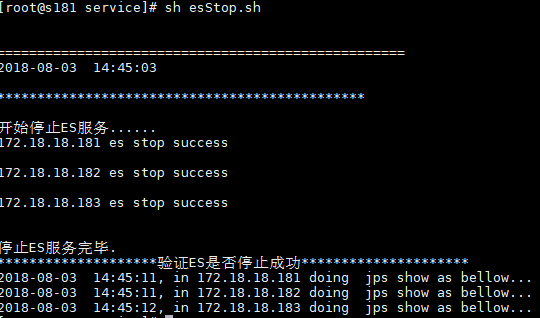
sh azkabanStop.sh



##### 停止elasticsearch

cd /opt/GoSunBigDataDeploy/service

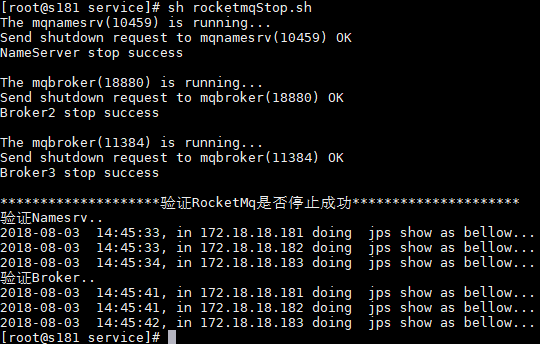
sh esStop.sh



##### 停止rocketmq

cd /opt/GoSunBigDataDeploy/service

sh rocketmqStop.sh



##### 停止kafka

cd /opt/GoSunBigDataDeploy/service

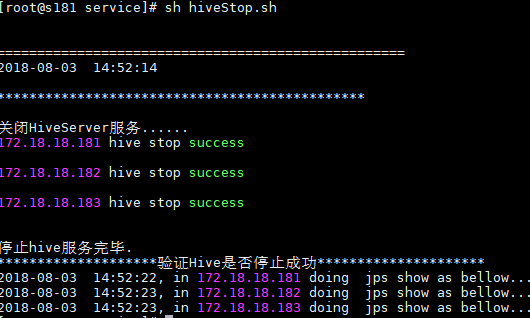
sh kafkaStop.sh



##### 停止hive

cd /opt/GoSunBigDataDeploy/service

sh hiveStop.sh



##### 停止hbase

cd /opt/GoSunBigDataDeploy/service

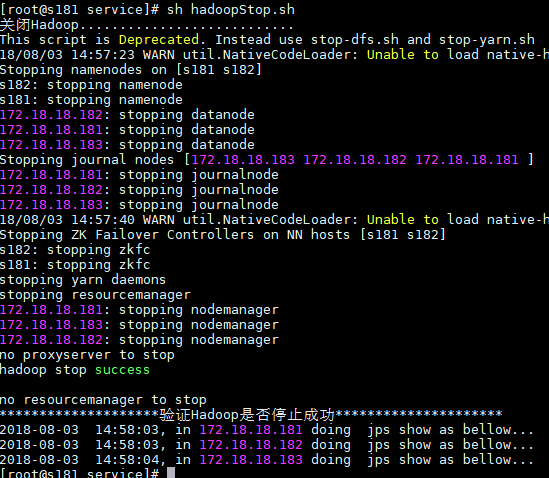
sh hbaseStop.sh



##### 停止hadoop

cd /opt/GoSunBigDataDeploy/service

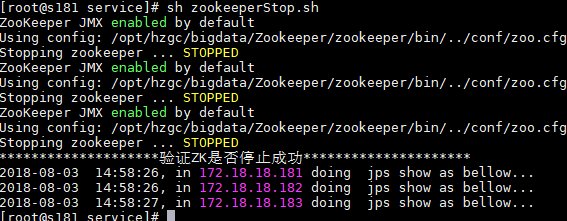
sh hadoopStop.sh



##### 停止zookeeper

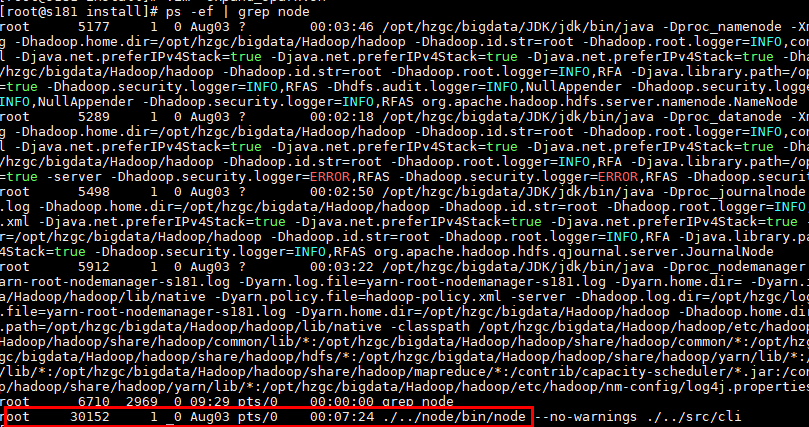
cd /opt/GoSunBigDataDeploy/service

sh zookeeperStop.sh



##### 停止kibana

ps -ef | grep node



Kill -9 pid(进程号)

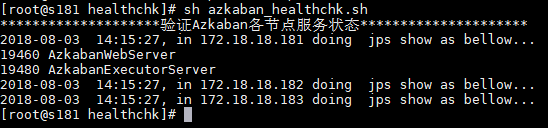
### 集群服务检查

进入服务检查目录，执行对应服务的进程检查脚本

#### Azkaban服务检查

cd /opt/GoSunBigDataDeploy/healthchk

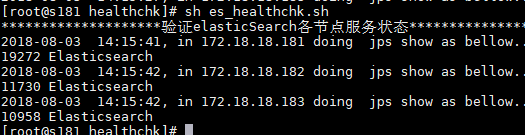
sh azkaban\_healthchk.sh



#### Elastic服务检查

cd /opt/GoSunBigDataDeploy/healthchk

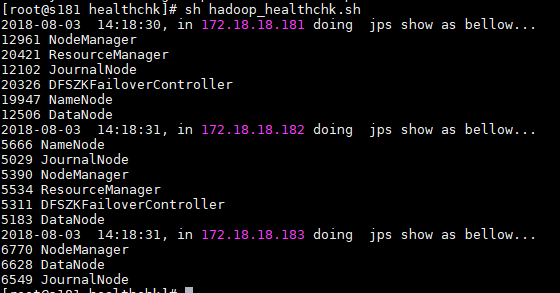
sh es\_healthchk.sh



#### Hadoop服务检查

cd /opt/GoSunBigDataDeploy/healthchk

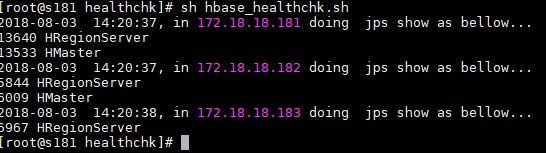
sh hadoop\_healthchk.sh



#### HBase服务检查

cd /opt/GoSunBigDataDeploy/healthchk

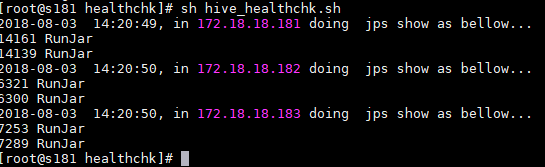
sh hbase\_healthchk.sh



#### Hive服务检查

cd /opt/GoSunBigDataDeploy/healthchk

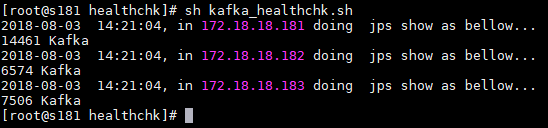
sh hive\_healthchk.sh



#### Kafka服务检查

cd /opt/GoSunBigDataDeploy/healthchk

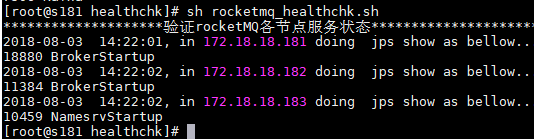
sh kafka\_healthchk.sh



#### Rocketmq服务检查

cd /opt/GoSunBigDataDeploy/healthchk

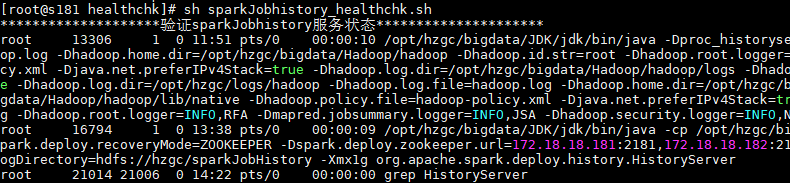
sh rocketmq\_healthchk.sh



#### Sparkjobhistory服务检查

cd /opt/GoSunBigDataDeploy/healthchk

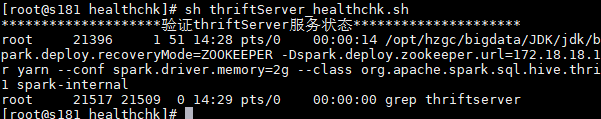
sh sparkJobhistory\_healthchk.sh



#### Thriftserver服务检查

cd /opt/GoSunBigDataDeploy/healthchk

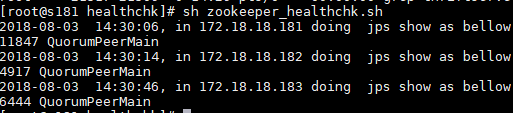
sh thriftServer\_healthchk.sh



#### Zookeeper服务检查

cd /opt/GoSunBigDataDeploy/healthchk

sh zookeeper\_healthchk.sh



### 集群扩展

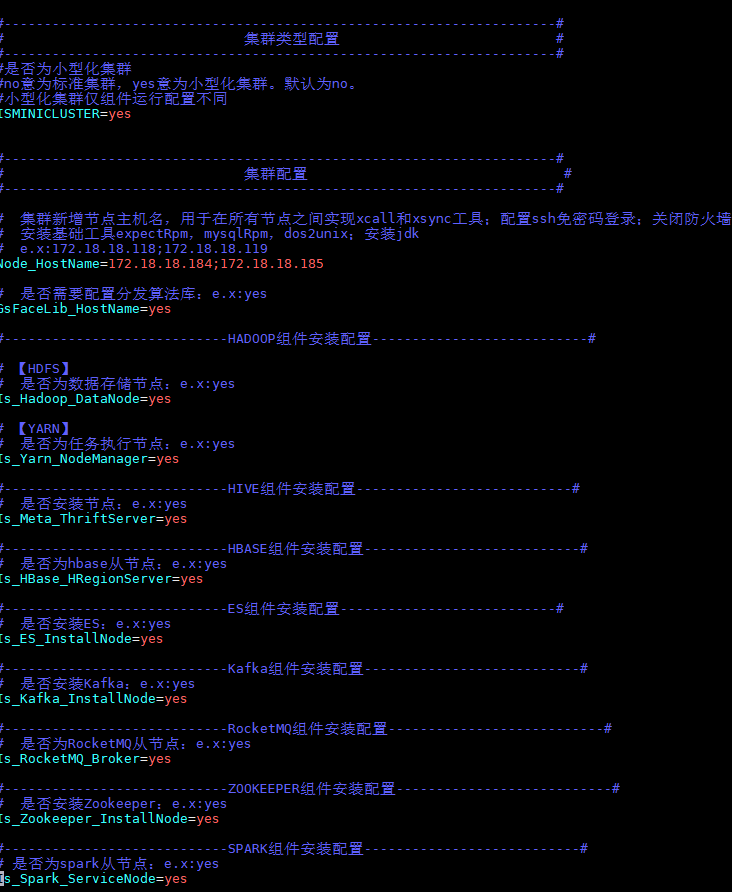
进入集群扩展模块目录，修改配置文件，依次执行扩展脚本，服务启动脚本（部分服务不支持动态扩展，需重启整个集群后才生效）。

#### 修改扩展配置文件

cd /opt/GoSunBigDataDeploy/expand/conf

vim expand\_conf.properties

将需要安装的组件的key改为yes即可，不需要的改为no，详细说明见配置文件注释



#### 配置主机名和主机ip映射

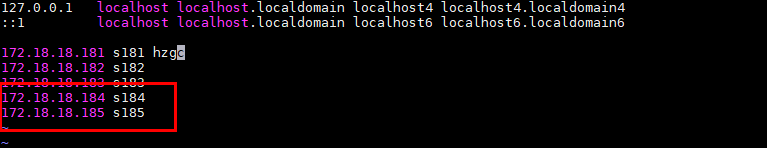
修改/etc/hosts文件

执行命令：vim /etc/hosts

在文件结尾添加以下内容，并保存退出(hzgc可配在任意节点下)

172.18.18.184 s184

172.18.18.185 s185



#### 执行扩展节点环境部署脚本

cd /opt/GoSunBigDataDeploy/expand/install

执行expandEnv.sh

expandEnv.sh包含：

1. 扩展节点安装dos2unix，expect
2. 配置原有节点和新增节点的免密登录
3. 分发新的host文件
4. 删除扩展节点上的原有环境变量
5. 关闭扩展节点防火墙
6. 在扩展节点上安装jdk

#### 执行扩展节点服务部署脚本

cd /opt/GoSunBigDataDeploy/expand/install

执行expandCompent.sh

expendCompent.sh包含：

1. 安装hadoop中的datanode，resourcemanager服务
2. 安装zookeeper服务
3. 安装hbase中的regionserver服务
4. 安装hive，kafka，scala，spark，rocketmq，es服务
5. 配置环境变量和服务启动目录
6. 将节点上服务安装情况同步到主配置文件cluster\_conf.properties中

#### 启动扩展节点服务

进入扩展节点服务启动目录，执行服务启动脚本（若在目录下没有找到对应服务的启动脚本，则说明该服务需要集群重启生效）

cd /opt/GoSunBigDataDeploy/expand/service

##### 启动datanode

sh start\_datanode.sh

##### 启动elasticSearch

sh start\_es.sh

##### 启动hive

sh start\_hive.sh

##### 启动nodemanager

sh start\_nodemanager.sh

##### 启动rocketmq

sh start\_rocketmq.sh

### 部署TiDB

#### 安装依赖的rpm包

执行命令

sh /opt/GoSunBigDataDeploy/tidb-ansible/ansible-denpendency/install\_ansible\_system\_rpms.sh

#### 创建tidb用户及互信

useradd -m -d /home/tidb tidb

设置密码

passwd tidb

配置 tidb 用户 sudo 免密码

执行命令 visudo

在文件末尾加上

tidb ALL=(ALL) NOPASSWD: ALL

生成tidb用户的 ssh key:

切换用户后生成秘钥文件

su - tidb

ssh-keygen -t -rsa

成功后,私钥文件为 /home/tidb/.ssh/id\_rsa

公钥文件为 /home/tidb/.ssh/id\_rsa.pub

#### 安装Ansible

进入目录 tidb-ansible/ansible-install

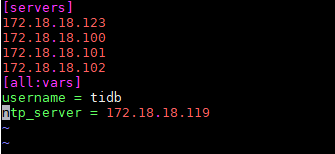
执行install\_ansible.sh

安装完成后可以通过ansible --verson查看版本

#### 配置tidb用户的免密登录

修改hosts.ini文件

servers下写上所有需要部署TiDB的节点ip

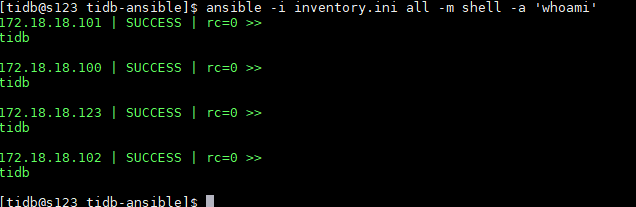


执行命令，会根据配置文件去目标机器上创建用户并配置tidb用户的免密登录

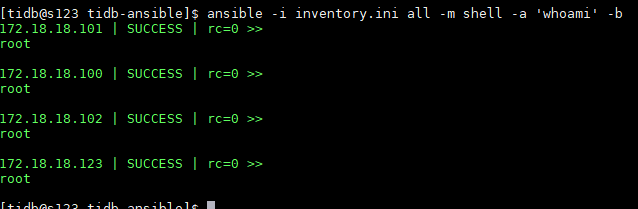
ansible-playbook -i hosts.ini create\_users.yml -u root -k

验证互信是否成功：

ansible -i inventory.ini all -m shell -a 'whoami'



验证是否可以从tidb用户免密切换到root用户



#### 配置inventory.ini文件

配置原则：

PD节点必须为奇数个（1个或3个）

TiKV节点为3个

TiDB节点至多2个，或1个

TiDB，PD节点不能与TiKV节点为同一个

参数说明：

[tidb\_servers]

部署tidb服务的节点ip

[tikv\_servers]

部署tikc服务的节点ip

[pd\_servers]

部署pd服务的节点ip

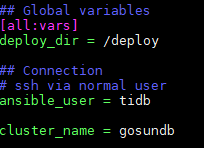


deploy\_dir=

根据磁盘挂载情况配置 例：挂盘在/opt 下，配置为deploy\_dir=/opt/deploy

必须配置为tidb用户

ansible\_user= tidb



#### 启动TiDB

启动命令：

ansible-playbook start.yml

#### 测试TiDB是否可用

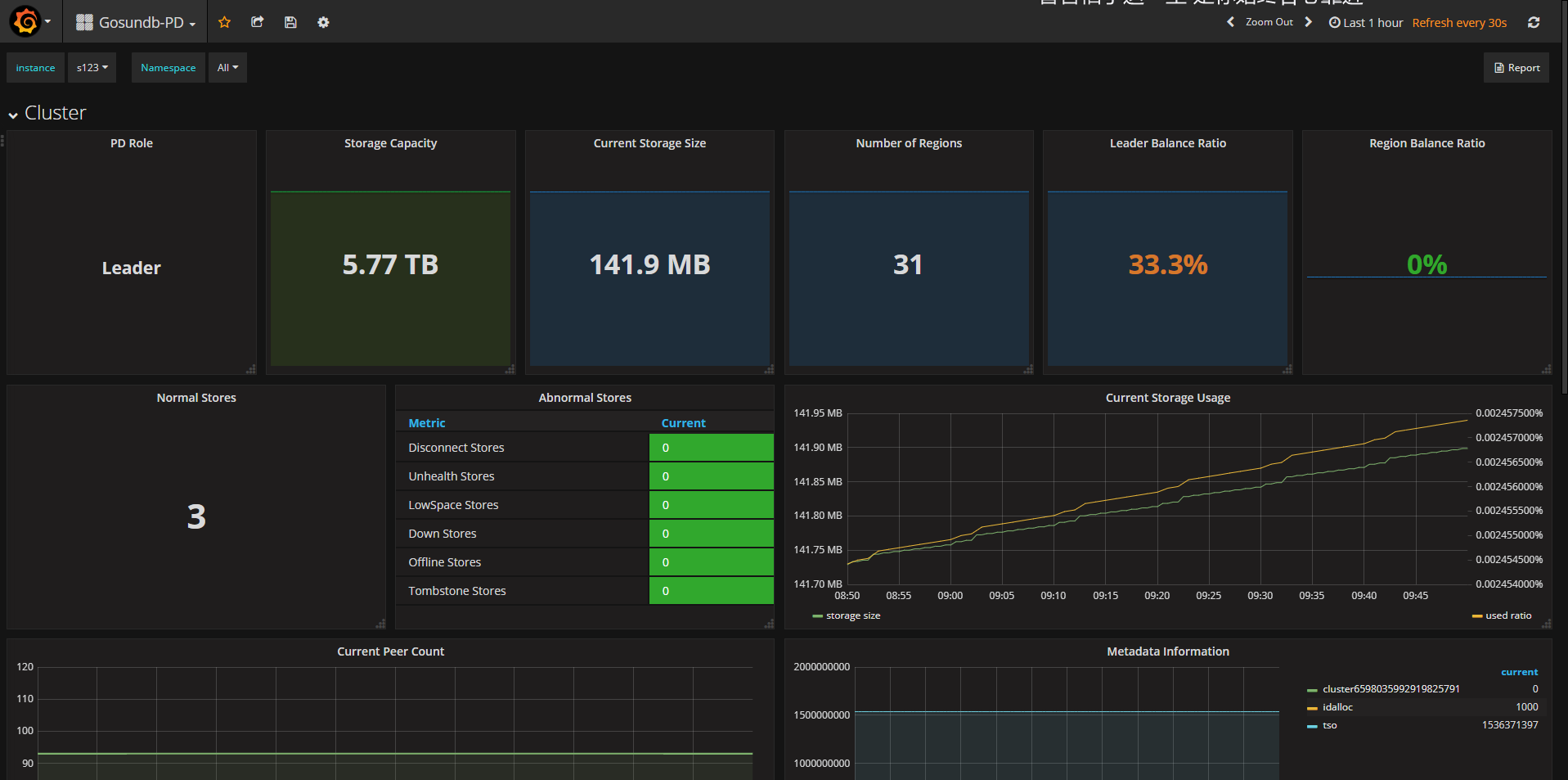
mysql -u root -h IP -P 4000

使用方法和mysql一样

浏览器地址：

http://<TiDB节点IP>:3000

默认帐号密码是：admin/admin



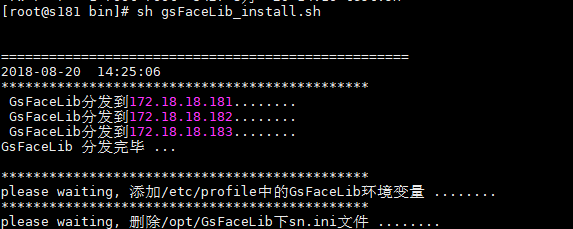
## 项目部署

### 部署算法库

#### 分发配置

cd /opt/GoSunBigDataDeploy/project/bin/

sh gsFaceLib\_install.sh

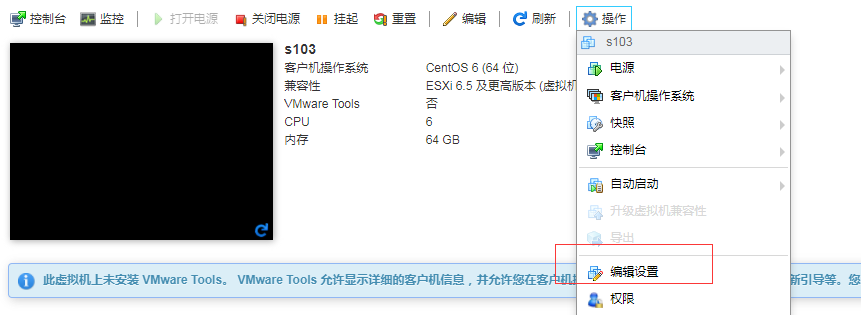


#### 添加加密狗

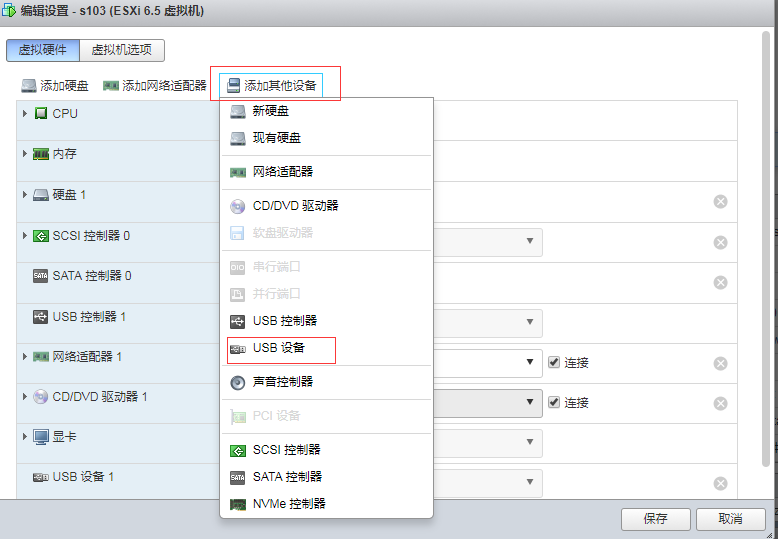
#### 添加加密狗驱动（vmware情况下执行）

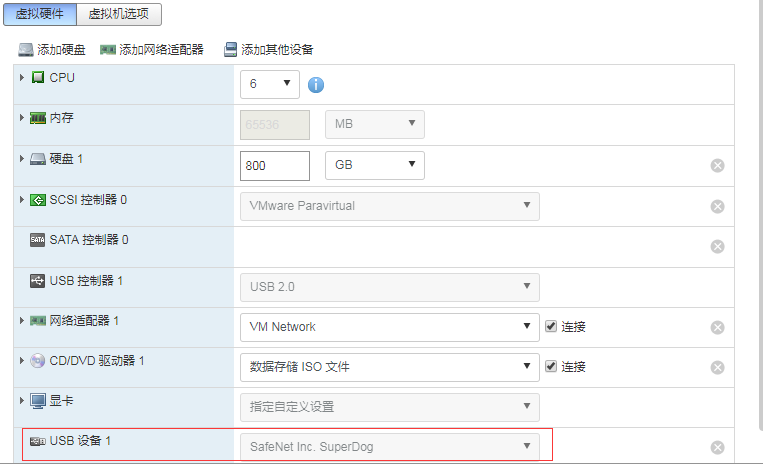
在对应的服务器中添加加密狗

登录对应服务器的vmware，编辑对应虚拟机设置：



添加USB控制器，后选择添加加密狗：





#### 添加加密狗驱动（每台需要配置算法的服务器下执行）

cd /opt/GsFaceLib/driver/

rpm -ivh superdog-1.0-1.i386.rpm



#### 配置算法库

##### 修改图片解析方式

cd /opt/GsFaceLib/

vim facelib.config

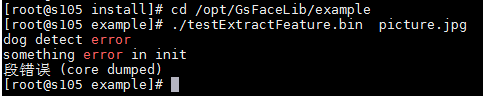
修改为device=CPU



##### 进入/opt/GsFaceLib/example目录下

cd /opt/GsFaceLib/example

./testExtractFeature.bin picture.jpg



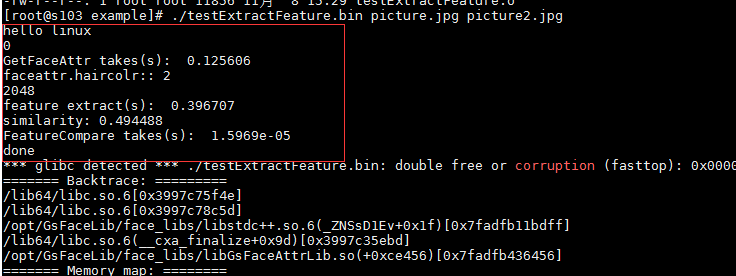
##### 生成新的sn.ini文件

将新产生的sn.ini文件内容发给**算法组 周琛辉** 转成相应的密钥后保存到sn.ini

##### 继续进入/opt/GsFaceLib/example执行

cd /opt/GsFaceLib/example

./testExtractFeature.bin picture.jpg



### 部署Collect

#### 修改FTP配置文件

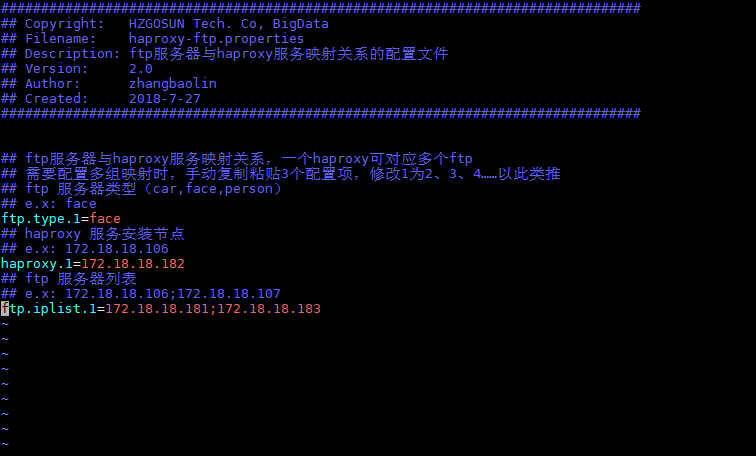
vim /opt/GoSunBigDataDeploy/project/conf/haproxy-ftp.properties

参数说明：

ftp.type 服务器存储数据类型：face（人脸），person（行人），car（汽车）

haproxy 代理节点ip

ftp.iplist ftp服务器列表



#### 启动FTP

cd /opt/Collect

sh bin/start-ftpserver.sh

查看Ftp是否启动成功

tailf logs/ftpserver.log

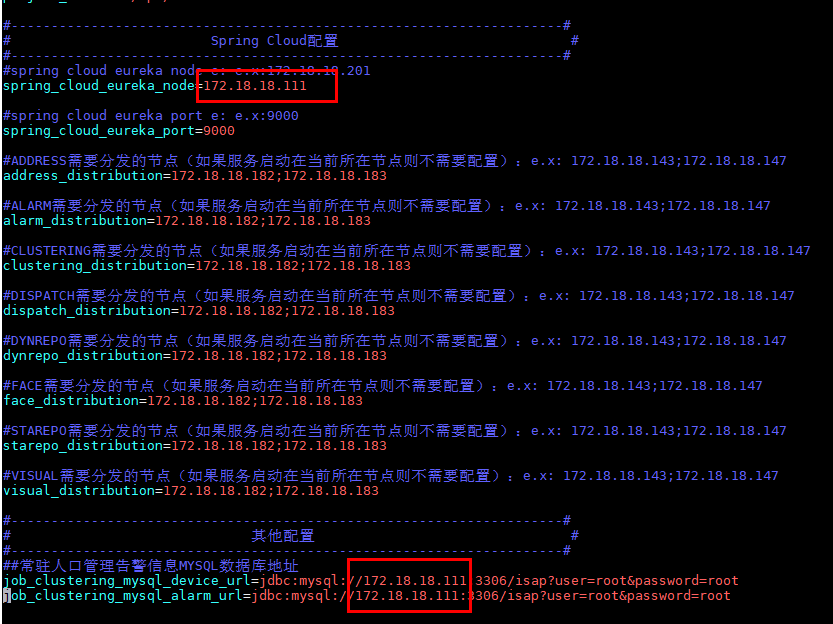


### 部署GoSunBigData

#### 修改配置文件

vim /opt/GoSunBigDataDeploy/project/conf/haproxy-ftp.properties

Spring\_cloud的ip和端口、mysql的ip向平台组同事获取



#### 执行部署脚本

cd /opt/GoSunBigDataDeploy/project/bin/

sh project\_distribute.sh

#### CLuster模块（建表，验证，启动job）

执行建表脚本

cd /opt/GoSunBigDataDeploy/project/bin/

sh create-all-table.sh

##### 验证Es索引

查看es索引是否创建成功

在浏览器中输入http://172.18.18.181:9200/dynamic?pretty

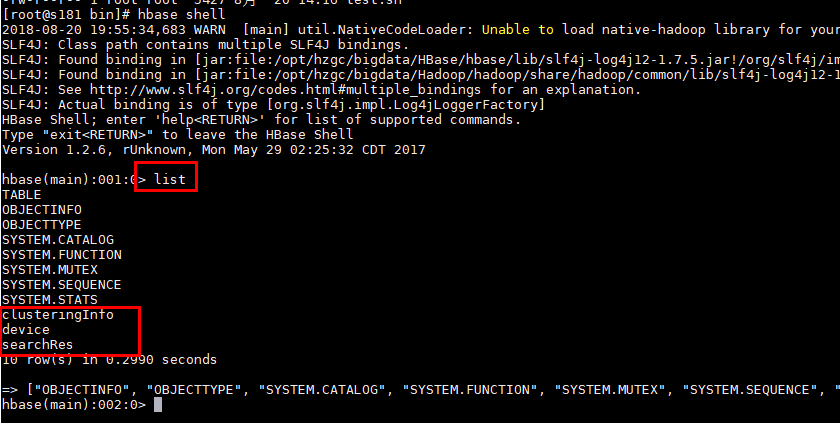


##### 验证Hbase表

执行下面命令查看hbase中表数据

source /opt/hzgc/env\_bigdata.sh

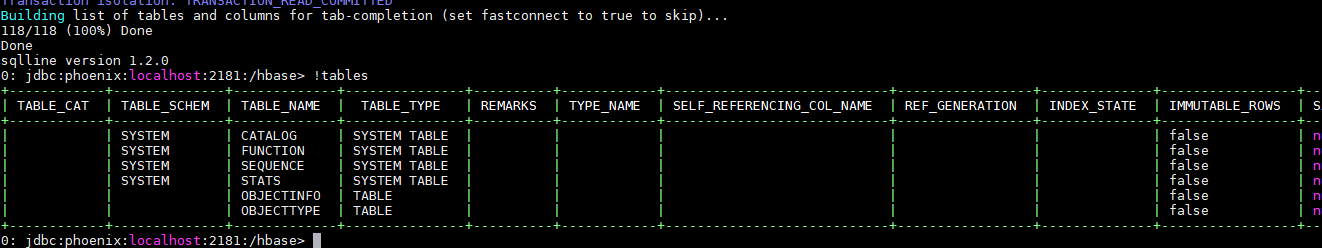
hbase shell



##### 验证Phoenix表

查看phoenix下的表

sqlline.py



##### 验证Spark

查看spark-udf

输入hive

show functions



##### 启动start-face-alarm-job.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-face-alarm-job.sh

##### 启动start-kafka-to-parquet.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-kafka-to-parquet.sh

##### 启动start-kmeans-clustering-job.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-kmeans-clustering-job.sh

##### 启动azkaban任务

cd /opt/GoSunBigDataDeploy/Cluster/azkaban/bin

sh create-schedule-job-to-zip.sh

cd zip 进入压缩包目录

ll 查看压缩包

TIM图片20180621100436

把压缩包拷贝到本地

进入Azkaban网页 <https://172.18.18.103:8443启动定时任务>

点击Create Project



填写Name和Description然后点击Create Project



点击Upload上传对应的三个zip包，上传成功后点击Upload





点击Execute Flow然后点击Schedule





填写对应的Time(可以往后填一点)和Date(选择当天就ok)，Recurrence中Days根据ZIP包的名称进行选择，然后点击Schedule，然后点击Continue就ok了



#### 微服务模块

common目录：address，face，visual

face目录：alarm，peoplemanager，dispatch，dynRepo，staRepo，dynperson，dyncar

服务启动方法以address为例：

启动address服务

cd /opt/GoSunBigDataDeploy/Service/common

cd address

sh bin/start-address.sh

tailf log/2018-06-22.address.0.log 查看服务是否启动成功



alarm，peoplemanager，dispatch，dynRepo，face，staRepo，visual，dynperson，dyncar服务启动与上面步骤类似(服务启动完成后jps查看下对应的服务进程是否启动成功)

# 服务器小型化平台搭建

## 服务器小型化集群搭建

### 集群安装准备

进入bin目录，执行环境部署脚本

cd /opt/GoSunBigDataDeploy/install/local/bin

执行命令：

sh envInstall\_local.sh

envInstall\_local.sh包含：

1. 本节点上安装sshpass，dos2unix，expect
2. 配置免密登录
3. 删除环境变量
4. 关闭防火墙
5. 创建日志目录

### 集群组件安装

进入bin目录，执行集群部署脚本

cd /opt/GoSunBigDataDeploy/install/local/bin

执行命令：

sh compentInstall\_local.sh

compentInstall\_local.sh包含：

1. 安装JDK，scala，mysql
2. 安装Hadoop，HBase，Zookeeper，Spark，Phoenix
3. 安装Rocketmq，azkaban，ElasticSearch，Kinaba

### 启动停止集群服务

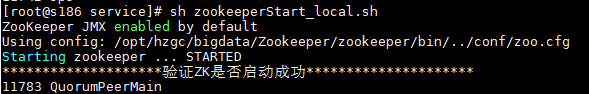
进入单节点组件启停目录

#### 启动集群服务

##### 启动zookeeper

cd /opt/GoSunBigDataDeploy/service\_local

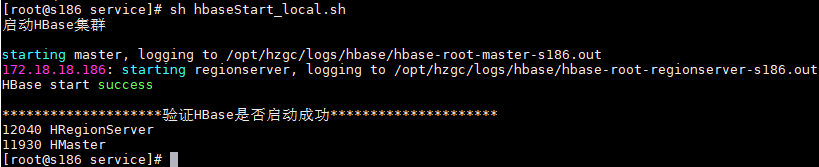
sh zookeeperStart\_local.sh



##### 启动hbase

cd /opt/GoSunBigDataDeploy/service\_local

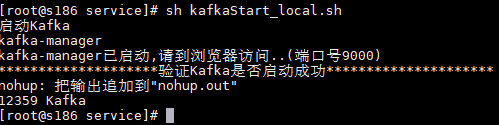
sh hbaseStart\_local.sh



##### 启动kafka

cd /opt/GoSunBigDataDeploy/service\_local

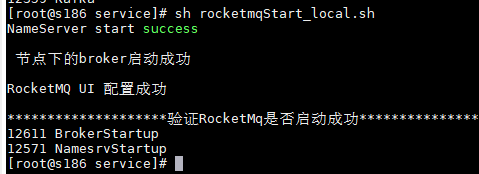
sh kafkaStart\_local.sh



##### 启动rocketmq

cd /opt/GoSunBigDataDeploy/service\_local

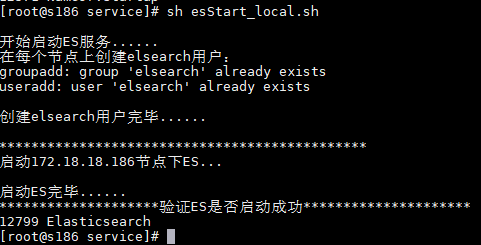
sh rocketmqStart\_local.sh



##### 启动elasticserach

cd /opt/GoSunBigDataDeploy/service\_local

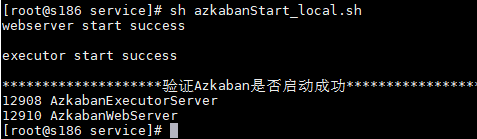
sh esStart\_local.sh



##### 启动azkaban

cd /opt/GoSunBigDataDeploy/service\_local

sh azkabanStart\_local.sh



#### 停止集群服务

##### 停止zookeeper

cd /opt/GoSunBigDataDeploy/service\_local

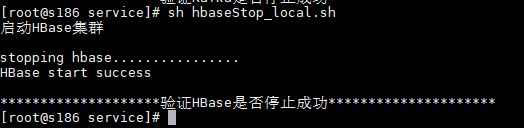
sh zookeeperStop\_local.sh



##### 停止hbase

cd /opt/GoSunBigDataDeploy/service\_local

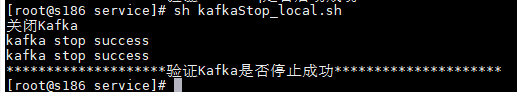
sh hbaseStop\_local.sh



##### 停止kafka

cd /opt/GoSunBigDataDeploy/service\_local

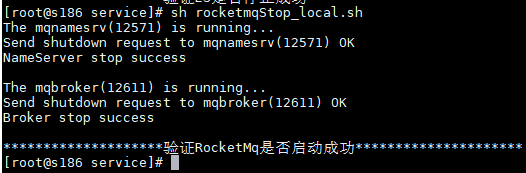
sh kafkaStop\_local.sh



##### 停止rocketmq

cd /opt/GoSunBigDataDeploy/service\_local

sh rocketmqStop\_local.sh



##### 停止elasticserach

cd /opt/GoSunBigDataDeploy/service\_local

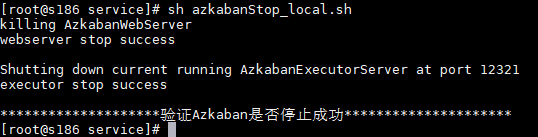
sh esStop\_local.sh



##### 停止azkaban

cd /opt/GoSunBigDataDeploy/service\_local

sh azkabanStop\_local.sh



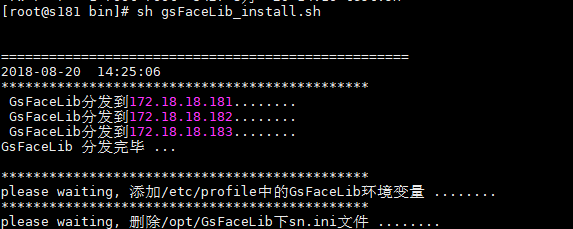
## 项目部署

### 部署算法库

#### 分发配置

cd /opt/GoSunBigDataDeploy/project/bin/

sh gsFaceLib\_install.sh

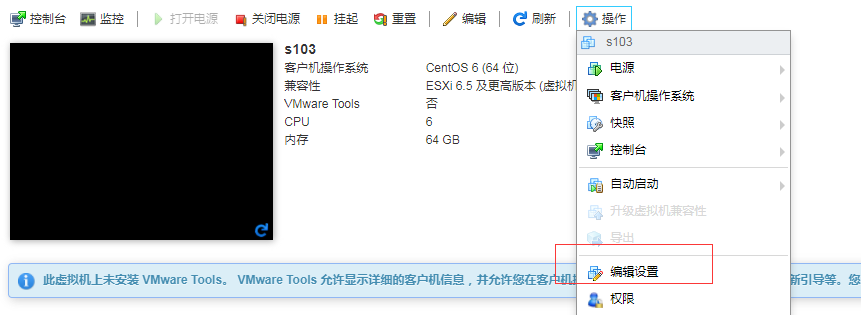


#### 添加加密狗

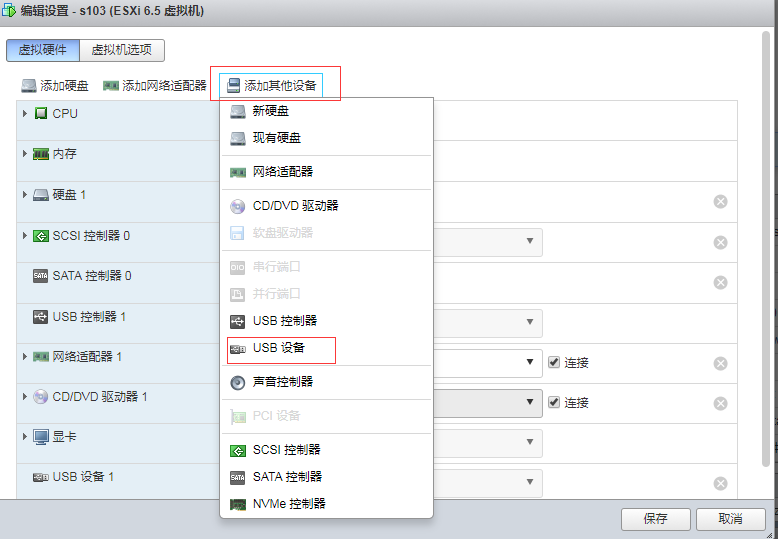
#### 添加加密狗驱动（vmware情况下执行）

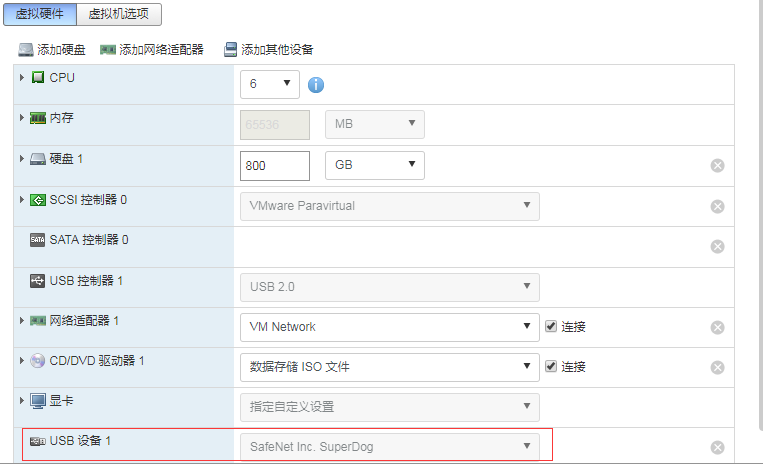
在对应的服务器中添加加密狗

登录对应服务器的vmware，编辑对应虚拟机设置：



添加USB控制器，后选择添加加密狗：





#### 添加加密狗驱动（每台需要配置算法的服务器下执行）

cd /opt/GsFaceLib/driver/

rpm -ivh superdog-1.0-1.i386.rpm



#### 配置算法库

##### 修改图片解析方式

cd /opt/GsFaceLib/

vim facelib.config

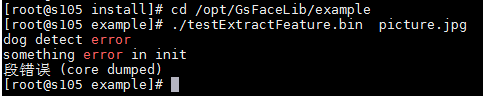
修改为device=CPU



##### 进入/opt/GsFaceLib/example目录下

cd /opt/GsFaceLib/example

./testExtractFeature.bin picture.jpg



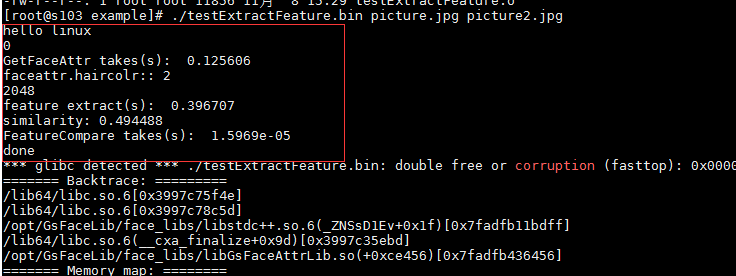
##### 生成新的sn.ini文件

将新产生的sn.ini文件内容发给**算法组 周琛辉** 转成相应的密钥后保存到sn.ini

##### 继续进入/opt/GsFaceLib/example执行

cd /opt/GsFaceLib/example

./testExtractFeature.bin picture.jpg



### 部署Collect

#### 修改FTP配置文件

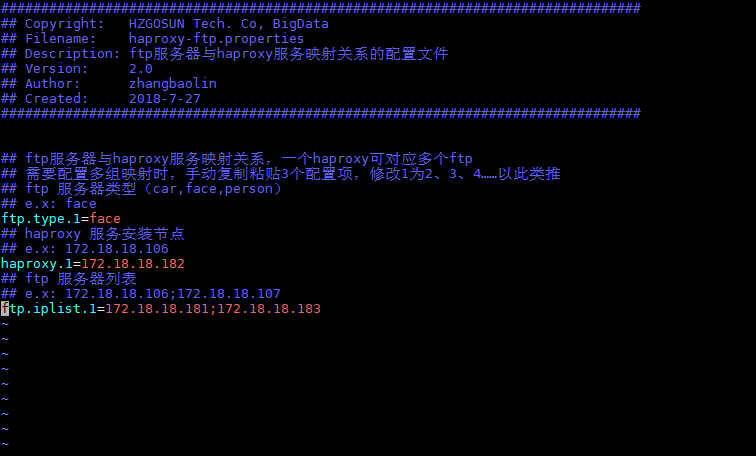
vim /opt/GoSunBigDataDeploy/project/conf/haproxy-ftp.properties

参数说明：

ftp.type 服务器存储数据类型：face（人脸），person（行人），car（汽车）

haproxy 代理节点ip

ftp.iplist ftp服务器列表



#### 启动FTP

cd /opt/Collect

sh bin/start-ftpserver.sh

查看Ftp是否启动成功

tailf logs/ftpserver.log

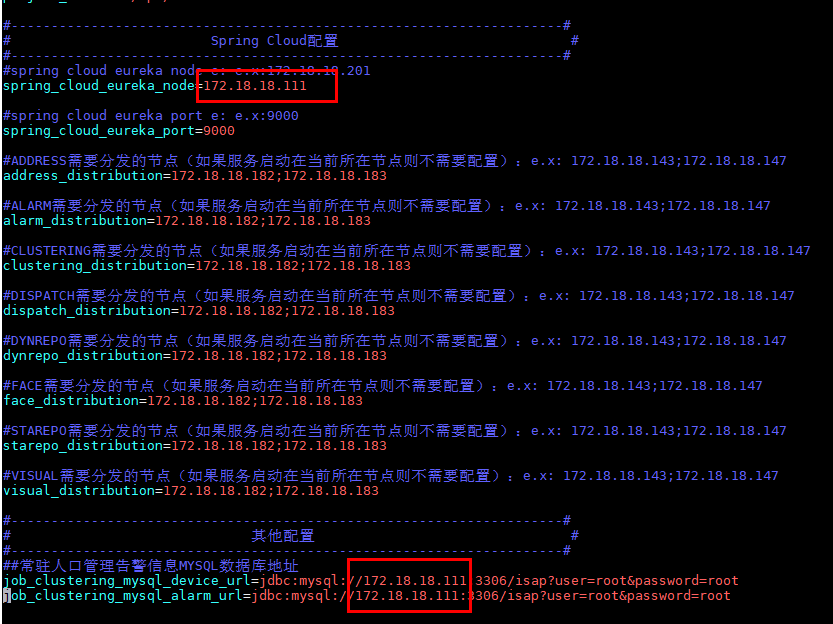


### 部署GoSunBigData

#### 修改配置文件

vim /opt/GoSunBigDataDeploy/project/conf/haproxy-ftp.properties

Spring\_cloud的ip和端口、mysql的ip向平台组同事获取



#### 执行部署脚本

cd /opt/GoSunBigDataDeploy/project/bin/

sh project\_distribute.sh

#### CLuster模块（建表，验证，启动job）

执行建表脚本

cd /opt/GoSunBigDataDeploy/project/bin/

sh create-all-table.sh

##### 验证Es索引

查看es索引是否创建成功

在浏览器中输入http://172.18.18.181:9200/dynamic?pretty

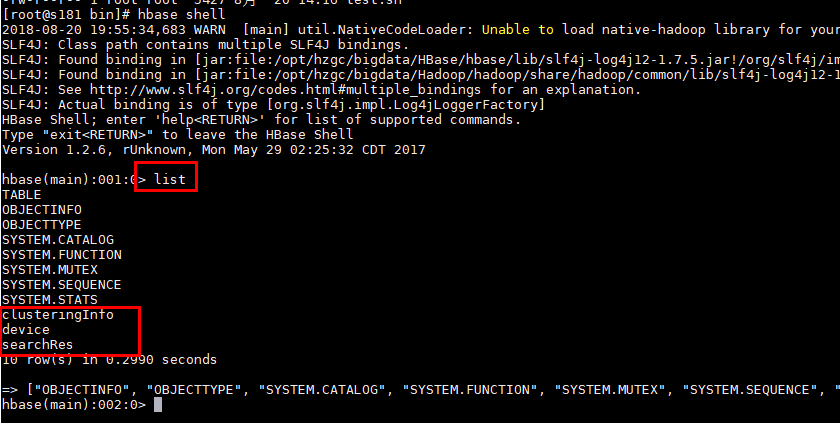


##### 验证Hbase表

执行下面命令查看hbase中表数据

source /opt/hzgc/env\_bigdata.sh

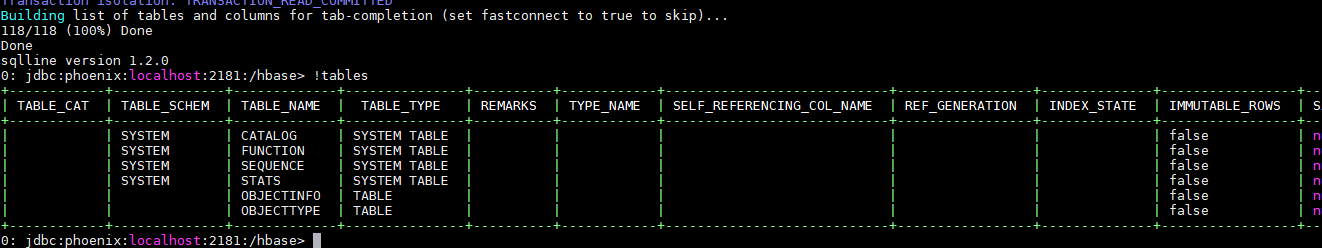
hbase shell



##### 验证Phoenix表

查看phoenix下的表

sqlline.py



##### 验证Spark

查看spark-udf

输入hive

show functions



##### 启动start-face-alarm-job.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-face-alarm-job.sh

##### 启动start-kafka-to-parquet.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-kafka-to-parquet.sh

##### 启动start-kmeans-clustering-job.sh

cd /opt/GoSunBigDataDeploy/Cluster/spark/bin

执行命令 sh start-kmeans-clustering-job.sh

##### 启动azkaban任务

cd /opt/GoSunBigDataDeploy/Cluster/azkaban/bin

sh create-schedule-job-to-zip.sh

cd zip 进入压缩包目录

ll 查看压缩包

TIM图片20180621100436

把压缩包拷贝到本地

进入Azkaban网页 <https://172.18.18.103:8443启动定时任务>

点击Create Project



填写Name和Description然后点击Create Project



点击Upload上传对应的三个zip包，上传成功后点击Upload





点击Execute Flow然后点击Schedule





填写对应的Time(可以往后填一点)和Date(选择当天就ok)，Recurrence中Days根据ZIP包的名称进行选择，然后点击Schedule，然后点击Continue就ok了



#### 微服务模块

common目录：address，face，visual

face目录：alarm，peoplemanager，dispatch，dynRepo，staRepo，dynperson，dyncar

服务启动方法以address为例：

启动address服务

cd /opt/GoSunBigDataDeploy/Service/common

cd address

sh bin/start-address.sh

tailf log/2018-06-22.address.0.log 查看服务是否启动成功



alarm，peoplemanager，dispatch，dynRepo，face，staRepo，visual，dynperson，dyncar服务启动与上面步骤类似(服务启动完成后jps查看下对应的服务进程是否启动成功)