XXXXXXXX

**安装部署手册**

北京易华录信息技术股份有限公司

2018年5月

目 录

[1. 文档说明 1](#_Toc515877408)

[1.1. 编写目的 1](#_Toc515877409)

[1.2. 适用范围 1](#_Toc515877410)

[1.3. 术语与缩写解释 1](#_Toc515877411)

[1.4. 其他说明 2](#_Toc515877412)

[2. 软件清单 3](#_Toc515877413)

[2.1. 基础软件清单 3](#_Toc515877414)

[2.2. 自有软件清单 4](#_Toc515877415)

[3. 部署顺序 4](#_Toc515877416)

[4. 服务器规划 4](#_Toc515877417)

[5. 基础软件部署 5](#_Toc515877418)

[5.1. Linux系统配置(所有节点) 5](#_Toc515877419)

[5.2. JDK安装配置(所有节点) 8](#_Toc515877420)

[5.3. Zookeeper安装配置 8](#_Toc515877421)

[5.4. Hadoop安装配置 10](#_Toc515877422)

[5.5. Spark安装配置 25](#_Toc515877423)

[5.6. ElasticSearch安装配置 28](#_Toc515877424)

[5.7. Kafka安装配置 44](#_Toc515877425)

[5.8. Kafka Manager安装 46](#_Toc515877426)

[5.9. Redis安装配置 47](#_Toc515877427)

[5.10. RabbitMq安装配置 48](#_Toc515877428)

[5.11. Tomcat安装 50](#_Toc515877429)

[6. 业务软件部署 51](#_Toc515877430)

[6.1. 资源管理平台web-dap 51](#_Toc515877431)

[6.2. 可视化平台web-overview 53](#_Toc515877432)

[6.3. 舆情消息接收服务travel-rabbitmq-service 55](#_Toc515877433)

[6.4. 可视化服务overview-service 57](#_Toc515877434)

[6.5. 系统管理中心gcits-data-platform、ehl-sys-user 58](#_Toc515877435)

[6.6. 昌平停车管理系统EHL\_PARK 62](#_Toc515877436)

[6.7. 旅游数据分析服务 td-service 64](#_Toc515877437)

[6.8. 大数据查询统计服务ehl-trip-query-service 67](#_Toc515877438)

# 

# 文档说明

## 编写目的

使软件实施人员快速、准确的部署昌平旅游大数据平台的基础环境及配套应用软件。

## 适用范围

本文档用于安装部署阶段使用，用于指导相关的软件实施人员进行部署。

由于部署阶段使用的操作系统为Linux，需要相关人员对Linux命令有一定的基本了解。

## 术语与缩写解释

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
| **Hadoop** | 开源分布式文件存储系统和分布式运算系统 |
| **Zookeeper** | 分布式对象管理器，负责分布式程序的协调管理 |
| **Kafka** | 分布式消息队列，基于硬盘顺序读写 |
| **HDFS** | hadoop的文件系统 |
| **Tomcat** | Web容器 |
| **Spark** | 开源分布式全栈计算平台，立足于内存计算 |
| **ElasticSearch** | 基于Lucene的分布式全文搜索引擎 |
| **Fastdfs** | 开源轻量级分布式文件系统，适合图片视频文件存储 |
| **Redis** | 开源高性能的key-value数据库，支持内存存储 |
| **RabbitMQ** | 开源分布式消息队列系统 |

## 其他说明

### 通用说明

1. 本文档所有术语定义为四号加粗字体，如**HDFS**。
2. 所有linux命令以斜体表示，如 *source /etc/profile*。
3. 所有命令、路径以及文件名区分大小写。
4. 有些配置一样的软件可以直接在服务器之间scp复制，包括环境变量的配置。(哪些软件可以用于复制安装，软件安装环节有说明)。
5. 配置文件中以#开头的行为注释行，配置时可忽略。
6. 软件部署、使用均使用root用户登录，文档后续不再重复说明。

### 软件目录约定

1. 软件目录

本系统所有软件均统一安装目录为/app

/app/jdk

/app/hadoop

/app/tomcat

/app/kafka

/app/zookeeper

/app/elasticsearch

1. 数据目录

数据目录统一为/data1，如果有多个则顺序编号，如/data2,/data3

# 软件清单

## 基础软件清单

基础软件是指昌平旅游大数据平台部署所需的大数据分布式存储及计算环境。基础软件的安装部署依赖于大数据平台产品基础环境，各组件版本如下，不可随意更换版本。安装部署参见章节7-基础软件部署。

|  |  |
| --- | --- |
| **软件名称** | **相关软件** |
| **CentOS(Linux操作系统)** | **CentOS-7-x86\_64-DVD-1503-01.iso** |
| **JDK** | **jdk-7u75-linux-x64.tar.gz** |
| **Zookeeper** | **zookeeper-3.4.6.tar.gz** |
| **Hadoop** | **hadoop-2.6.0.tar.gz** |
| **Kafka** | **kafka\_2.11-0.8.2.0.tgz** |
| **Redis** | **redis-3.0.1.tar.gz** |
| **RabbitMq** | **erlang-17.4-1.el6.x86\_64.rpm** |
| **rabbitmq-server-3.5.0-1.noarch.rpm** |
| **Spark** | **scala-2.10.5.tgz** |
| **spark-1.5.2-bin-hadoop2.6.tgz** |
| **ElasticSearch** | **elasticsearch-2.2.0.tar.gz** |
| **elasticsearch-head-master.zip** |
| **Fastdfs** | **libfastcommon-master.zip** |
| **FastDFS\_v5.08.tar.gz** |
| **jdk** | **jdk-7u75-linux-x64.tar.gz** |
| **Tomcat** | **apache-tomcat-7.0.57.tar.gz** |
| **Oracle** | **Oracle 11g （11.2.0.3）** |
| **其他** | **autogen-libopts-5.18-5.el7.x86\_64.rpm**  **ntpdate-4.2.6p5-19.el7.centos.x86\_64.rpm**  **ntp-4.2.6p5-19.el7.centos.x86\_64.rpm**  **mpfr-3.1.1-4.el7.x86\_64.rpm**  **libmpc-1.0.1-3.el7.x86\_64.rpm**  **cpp-4.8.3-9.el7.x86\_64.rpm**  **kernel-headers-3.10.0-229.el7.x86\_64.rpm**  **glibc-headers-2.17-78.el7.x86\_64.rpm**  **glibc-devel-2.17-78.el7.x86\_64.rpm**  **gcc-4.8.3-9.el7.x86\_64.rpm** |

## 自有软件清单

|  |  |  |
| --- | --- | --- |
| **软件名称** | **所属产品** | **备注** |
| **web-dap** |  |  |
| **web-overview** |  |  |
| **travel-rabbitmq-service** |  |  |
| **overview-service** |  |  |
| **gcits-data-platform** |  |  |
| **ehl-sys-user** |  |  |
| **EHL\_PARK** |  |  |
| **td-service** |  |  |
| **ehl-trip-query-service** |  |  |

# 部署顺序

软件安装顺序请按照本文档指定顺序安装，并测试通过。

1. 基础环境部署

包含Linux系统、JDK、Zookeeper等，具体参见第7章。

1. 业务软件部署

各个系统或服务建议按照相应章节顺序进行部署。

# 服务器规划

基本原则：

1、NameNode,分为主备节点，各独占一台服务器

2、DataNode,多节点，每个DataNode独占一台服务器

3、Zookeeper多节点，必须为奇数个，比如3个。每个独占一台服务器

4、Kafka多节点，每个独占一台服务器

5、Spark 双master，worker3个以上

6、fastdfs同一分组中至少为2台服务器

具体服务器规划以具体交付项目的深化设计方案部署相关章节为准。

# 基础软件部署

## Linux系统配置(所有节点)

### 操作系统安装注意事项

1. 字符集为UTF-8
2. 服务器时区为Asia/Shanghai
3. 安装操作系统阶段rpm包只选择Basic Server即可
4. root用户密码统一为ehl1234

### 设置主机名称

1. 使用root用户登录CentOS系统
2. 执行hostnamectl set-hostname **hostname**命令(加粗字体为需要设定的主机名)
3. 执行reboot重新启动系统，生效主机名

### 设置ip地址

1. 执行vi /etc/sysconfig/network-scripts/ifcfg-ens160命令，修改文件内容如下：

BOOTPROTO=dhcp 修改为 BOOTPROTO=static

ONBOOT=no 修改为 ONBOOT=yes

追加IPADDR=**10.150.27.246**(加粗字体为需要设定的IP地址)

追加NETMASK=**255.255.255.0**(加粗字体为需要设定的IP地址掩码)

追加GATEWAY=**10.150.27.254**(加粗字体为需要设定的IP地址网关)

1. 执行service network restart生效网卡
2. 关闭selinux
3. 执行vi /etc/sysconfig/selinux命令，修改文件内容如下：

SELINUX=enforcing修改为 SELINUX=disabled

1. 执行setenforce 0，关闭selinux

### 关闭防火墙

关闭防火墙

systemctl stop firewalld.service

禁止防火墙自动启用

systemctl disable firewalld.service

### 安装ntp时间同步服务

从CentOS-7-x86\_64-DVD-1503-01.iso安装镜像或本版本安装包中获取如下rpm包，并执行以下命令

1. rpm -ivh autogen-libopts-5.18-5.el7.x86\_64.rpm
2. rpm -ivh ntpdate-4.2.6p5-19.el7.centos.x86\_64.rpm
3. rpm -ivh ntp-4.2.6p5-19.el7.centos.x86\_64.rpm

进行NTP服务配置与启动

NTP服务端执行vi /etc/ntp.conf 修改文件内容如下:

1. restrict default nomodify nopeer noquery
2. 注释掉以下几行：
   * + 1. #server 0.centos.pool.ntp.org iburst
       2. #server 1.centos.pool.ntp.org iburst
       3. #server 2.centos.pool.ntp.org iburst
       4. #server 3.centos.pool.ntp.org iburst
3. 追加以下两行：
   * + 1. server 127.127.1.0
       2. fudge 127.127.1.0 stratum 10
4. 执行service ntpd start启动NTP服务
5. 执行chkconfig ntpd on设置NTP自动启动

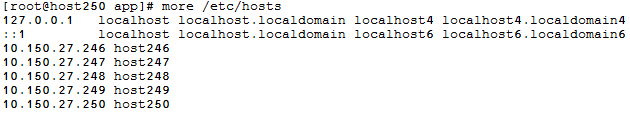
NTP客户端执行vi /etc/ntp.conf 修改文件内容如下:

1. 追加server **10.150.27.246**(加粗部分为NTP服务端IP地址)
2. 注释掉以下几行：
   * + 1. #server 0.centos.pool.ntp.org iburst
       2. #server 1.centos.pool.ntp.org iburst
       3. #server 2.centos.pool.ntp.org iburst
       4. #server 3.centos.pool.ntp.org iburst
3. 执行ntpdate 10.150.27.246(红字部分为NTP服务端IP地址)同步时间
4. 执行service ntpd start启动NTP服务
5. 执行chkconfig ntpd on设置NTP自动启动

### 配置系统参数

1. 执行vi /etc/hosts修改文件内容：

追加所有规划内的主机IP和主机名，例：



1. 执行vi /etc/profile命令修改配置文件

追加ulimit –n 65536

1. 执行. /etc/profile生效配置文件

### 配置ssh免校验登录

1. 为各个节点创建rsa密码文件：

执行ssh-keygen -t rsa

(注：输入上面指令后，需连续三次输入回车，弹出密码图后，执行下一条指令)

1. 创建密码文件，选定一台服务器节点如节点1，作为密码文件信息收集节点，执行以下命令：

cd /root/.ssh

touch authorized\_keys #创建密码文件

cat id\_rsa.pub >> authorized\_keys #将节点1的rsa密令写入密码文件

1. 合并密码文件
2. 执行ssh **host247** cat /root/.ssh/id\_rsa.pub >> authorized\_keys命令将剩余所有节点的id\_rsa.pub文件写入密码文件中(加粗部分为其它节点主机名)；
3. 执行scp命令将节点1的authorized\_keys文件推送给其他所有节点，命令如下(红字部分为其它节点主机名)：

scp authorized\_keys [root@**host247**:/root/.ssh/](mailto:root@host247:/root/.ssh/)

1. 在各个节点生效ssh免密码登录设置

在各个节点执行ssh **host247** date(加粗部分为其它节点主机名)

## JDK安装配置(所有节点)

1. 从软件包中获取jdk-7u75-linux-x64.tar.gz包，并上传至服务器/app目录下
2. 执行tar xvf jdk-7u75-linux-x64.tar.gz进行解压缩
3. 执行mv jdk1.7.0\_75 jdk更改目录名称
4. 使用vi修改文件etc/profile，追加以下内容：

export JAVA\_HOME=/app/jdk

export JRE\_HOME=/app/jdk/jre

export PATH=$PATH:$JAVA\_HOME/bin

export CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

1. 执行. /etc/profile生效文件，并执行java -version验证安装生效。



## Zookeeper安装配置

获取zookeeper安装包 zookeeper-3.4.6.tar.gz，上传到/app目录下，然后执行以下操作：

1. 进入/app目录 ***cd /app***
2. 解压zookeeper安装包 ***tar -zxvf zookeeper-3.4.6.tar.gz***
3. 重命名zookeeper安装目录 ***mv zookeeper-3.4.6 zookeeper***
4. 修改文件执行权限 ***chmod -R 755 /app/zookeeper/bin***

### 修改zoo.cfg配置文件

1. 进入conf目录 ***cd /app/zookeeper/conf/***
2. 复制一份zoo.cfg配置文件 ***cp zoo\_sample.cfg zoo.cfg***
3. 打开zoo.cfg配置文件 ***vi zoo.cfg***
4. 在文件末尾追加如下内容，以3个zookeeper服务器为例：

server.1=192.168.4.1:2888:3888

server.2=192.168.4.2:2888:3888

server.3=192.168.4.3:2888:3888

autopurge.purgeInterval=1

修改红色字体为实际zookeeper服务器的IP即可，多个zookeeper服务器就配置多行，server.后面的序号依次增加。

1. 修改数据存储地址

dataDir=/data1/zookeeper/data

如果/data1/zookeeper/data目录不存在，请先创建

1. 保存并退出 ***:wq***

### 修改myid值

1. 进入/data1/zookeeper/data目录 ***cd /data1/zookeeper/data***
2. 修改myid值 ***echo id> myid***

根据当前服务器的IP，修改红色字体id为步骤2中第4)步对应的id值，在执行命令。以server.1=192.168.4.1:2888:3888为例：如服务器IP是192.168.4.1，则修改上述命名中id为1。以此类推。

### 启动zookeeper

1. 进入bin目录  ***cd /app/zookeeper/bin***
2. 启动服务 ***./zkServer.sh start &***
3. 执行jps命令查看服务是否启动成功 ***jps***

如果出现QuorumPeerMain 这样的进程名称，则表示zookeeper服务启动成功

1. 使用scp命令将zookeeper目录依次完整复制至其他服务器

***scp -r /app/zookeeper root@host1:/app*** (红色字体修改为目标zookeeper主机名或IP)

1. 按4.2.2章节描述，依次修改各目标服务器的myid值，并启动zookeeper服务。

## Hadoop安装配置

使用root用户登录CentOS系统，在需要安装hadoop的服务器上执行以下命令创建数据存储目录：

***mkdir -p /data1/hadoop/hdfs/dfs***

***mkdir -p /data1/hadoop/hdfs/name***

***mkdir -p /data1/hadoop/hdfs/journal***

***mkdir -p /data1/hadoop/tmp***

从软件包中获取hadoop-2.6.0.tar.gz包，并上传至服务器/app目录下

执行***tar zxvf hadoop-2.6.0.tar.gz***进行解压缩

执行***mv hadoop-2.6.0 hadoop***更改目录名称

复制hadoop-lzo扩展包

1. 从软件包中获取***hadoop-lzo-compiled.tar.gz***,复制至/app/目录下；
2. 执行 ***tar zxvf hadoop-lzo-compiled.tar.gz***解压缩；
3. 所有***hadoop***节点均执行以上操作；

### 配置Hadoop环境变量

1. 打开配置文件 ***vi /etc/profile***
2. 在文件末尾追加如下内容：

export HADOOP\_HOME=/app/hadoop

export PATH=$PATH:$HADOOP\_HOME/bin

export PATH=$PATH:$HADOOP\_HOME/sbin

**export HADOOP\_HOME\_WARN\_SUPPRESS=1**

1. 保存并退出 ***:wq***
2. 使hadoop环境变量生效 ***source /etc/profile***
3. 确认hadoop环境变量是否生效 ***echo $HADOOP\_HOME***

如出现 /app/hadoop则表示hadoop环境变量配置正确

### 修改hadoop-env.sh配置文件

1. 打开配置文件 ***vi /app/hadoop/etc/hadoop/hadoop-env.sh***
2. 修改java环境变量

export JAVA\_HOME=/app/jdk

1. 修改hadoop pid文件目录

export HADOOP\_PID\_DIR=/app/hadoop/pids

1. 添加lzo环境变量

export LD\_LIBRARY\_PATH=/app/hadoop/lzo/lib

1. 保存并退出 ***:wq***

### 修改yarn-env.sh配置文件

1. 打开配置文件 ***vi /app/hadoop/etc/hadoop/yarn-env.sh***
2. 修改java环境变量

export JAVA\_HOME=/app/jdk

1. 修改yarn log文件路径

export YARN\_LOG\_DIR=/app/hadoop/logs

1. 保存并退出 ***:wq***

### 修改core-site.xml配置文件

1. 打开配置文件 vi /app/hadoop/etc/hadoop/core-site.xml

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://cluster1:8020</value>

</property>

<property>

<name>ha.zookeeper.quorum</name>

<value>host108:2181,host109:2181,host110:2181</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/data1/hadoop/tmp</value>

</property>

<property>

<name>io.compression.codecs</name>

<value>org.apache.hadoop.io.compress.GzipCodec,org.apache.hadoop.io.compress.DefaultCodec,com.hadoop.compression.lzo.LzoCodec,com.hadoop.compression.lzo.LzopCodec,org.apache.hadoop.io.compress.BZip2Codec</value>

</property>

<property>

<name>io.compression.codec.lzo.class</name>

<value>com.hadoop.compression.lzo.LzoCodec</value>

</property>

1. 在文件中追加如下内容（如存在下述内容则忽略此步）：
2. 修改ha.zookeeper.quorum属性的value值，如上图红色标示的内容，改为实际zookeeper服务器的主机名，端口号保持不变，可增加多个，用英文逗号隔开；
3. 修改hadoop文件系统压缩属性

io.compression.codecs及io.compression.codec.lzo.class；

1. 其他保持默认，执行:wq退出。

### 修改hdfs-site.xml

1. 打开配置文件vi /app/hadoop/etc/hadoop/hdfs-site.xml；
2. 在文本中追加以下内容，如存在则忽略；

<configuration>

<!--副本数-->

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

<!--定义集群名称-->

<property>

<name>dfs.nameservices</name>

<value>cluster1</value>

</property>

<!-- cluster1配置-->

<!-- namenode别名-->

<property>

<name>dfs.ha.namenodes.cluster1</name>

<value>nn1,nn2</value>

</property>

<!-- RPC地址配置-->

<property>

<name>dfs.namenode.rpc-address.cluster1.nn1</name>

<value>host210:8020</value>

</property>

<property>

<name>dfs.namenode.rpc-address.cluster1.nn2</name>

<value>host211:8020</value>

</property>

<!-- http地址配置-->

<property>

<name>dfs.namenode.http-address.cluster1.nn1</name>

<value>host210:50070</value>

</property>

<property>

<name>dfs.namenode.http-address.cluster1.nn2</name>

<value>host211:50070</value>

</property>

<!--cluster1的两个NameNode共享edits文件目录时，使用的JournalNode集群信息-->

<property>

<name>dfs.namenode.shared.edits.dir</name>

<value>qjournal://host210:8485;host211:8485;host212:8485/cluster1</value>

</property>

<!--cluster1是否启动自动故障恢复，即当NameNode出故障时，是否自动切换到另一台NameNode,默认为false-->

<property>

<name>dfs.ha.automatic-failover.enabled.cluster1</name>

<value>true</value>

</property>

<!--cluster1出故障时，哪个实现类负责执行故障切换-->

<property>

<name>dfs.client.failover.proxy.provider.cluster1</name>

<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>

</property>

<!--cluster1配置 end-->

<!--namenode信息保存地址，可以配置多个，以,分隔-->

<property>

<name>dfs.namenode.name.dir</name>

<value>/data1/hadoop/hdfs/name</value>

</property>

<!--datanode信息保存地址，可以配置多个，以,分隔-->

<property>

<name>dfs.datanode.data.dir</name>

<value>/data1/hadoop/hdfs/dfs</value>

</property>

<!--JournalNode集群在对NameNode的目录进行共享时，自己存储数据的磁盘路径-->

<property>

<name>dfs.journalnode.edits.dir</name>

<value>/data1/hadoop/hdfs/journal</value>

</property>

<!--一旦需要NameNode切换，使用ssh方式进行操作-->

<property>

<name>dfs.ha.fencing.methods</name>

<value>sshfence</value>

</property>

<!--如果使用ssh进行故障切换，使用ssh通信时用的密钥存储的位置-->

<property>

<name>dfs.ha.fencing.ssh.private-key-files</name>

<value>/root/.ssh/id\_rsa</value>

</property>

### mapred-site.xml配置

1. 切换至/app/hadoop/etc/hadoop/目录下；
2. 执行***cp mapred-site.xml.template mapred-site.xml***命令创建mapred-site.xml配置文件；
3. 执行***vi mapred-site.xml***命令，在文件中追加如下内容（如存在下述内容则忽略此步）。

<configuration>

<!-- MR YARN Application properties -->

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

<description>

The runtime framework for executing MapReduce jobs.

Can be one of local, classic or yarn.

</description>

</property>

<!-- jobhistory properties -->

<property>

<name>mapreduce.jobhistory.address</name>

<value>host106:10020</value>

<description>MapReduce JobHistory Server IPC host:port</description>

</property>

<property>

<name>mapreduce.jobhistory.webapp.address</name>

<value>host106:19888</value>

<description>MapReduce JobHistory Server Web UI host:port</description>

</property>

<property>

<name>mapred.compress.map.output</name>

<value>true</value>

</property>

<property>

<name>mapred.map.output.compression.codec</name>

<value>com.hadoop.compression.lzo.LzoCodec</value>

</property>

<property>

<name>mapred.child.env</name>

<value>LD\_LIBRARY\_PATH=/app/hadoop/lzo/lib</value>

</property>

1. 修改mapreduce.jobhistory.address属性的value值，如蓝色表示内容，改为NameNode 节点或者 SecondaryNameNode节点的主机名，二者取其一。
2. 修改mapreduce.jobhistory.webapp.address属性的value值，如蓝色表示内容，改为NameNode 节点或者 SecondaryNameNode节点的主机名，二者取其一，与4）步骤中属性值一致即可。
3. 增加lzo相关配置，mapred.compress.map.output、mapred.map.output.compression.codec及mapred.child.env；
4. 其他保存默认，保存并退出 ***:wq***

### 修改yarn-site.xml配置文件

1. 打开配置文件 vi /app/hadoop/etc/hadoop/yarn-site.xml
2. 在<configuration></configuration>之间添加如下内容：

<property>

<name>yarn.resourcemanager.ha.enabled</name>

<value>true</value>

</property>

<property>

<name>yarn.resourcemanager.ha.rm-ids</name>

<value>rm1,rm2</value>

</property>

<property>

<name>yarn.resourcemanager.hostname.rm1</name>

<value>host210</value>

</property>

<property>

<name>yarn.resourcemanager.hostname.rm2</name>

<value>host211</value>

</property>

<property>

<name>yarn.resourcemanager.recovery.enabled</name>

<value>true</value>

</property>

<property>

<name>yarn.resourcemanager.store.class</name>

<value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</value>

</property>

<property>

<name>yarn.resourcemanager.zk-address</name>

<value>host215:2181,host216:2181,host217:2181</value>

<description>For multiple zk services, separate them with comma</description>

</property>

<property>

<name>yarn.resourcemanager.cluster-id</name>

<value>yarn-ha</value>

</property>

<property>

<name>yarn.resourcemanager.webapp.address.rm1</name>

<value>host210:9026</value>

</property>

<property>

<name>yarn.resourcemanager.webapp.address.rm2</name>

<value>host211:9026</value>

</property>

<property>

<description>The class to use as the resource scheduler.</description>

<name>yarn.resourcemanager.scheduler.class</name>

<value>org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler</value>

</property>

<property>

<description>fair-scheduler conf location</description>

<name>yarn.scheduler.fair.allocation.file</name>

<value>/app/hadoop/etc/hadoop/fairscheduler.xml</value>

</property>

<property>

<description>

List of directories to store localized files in. An

application's localized file directory will be found in:

${yarn.nodemanager.local-dirs}/usercache/${user}/appcache/application\_${appid}.

Individual containers' work directories, called container\_${contid}, will

be subdirectories of this.

</description>

<name>yarn.nodemanager.local-dirs</name>

<value>/app/hadoop/local</value>

</property>

<property>

<name>yarn.nodemanager.log-dirs</name>

<value>/app/hadoop/logs</value>

</property>

<property>

<description>Whether to enable log aggregation</description>

<name>yarn.log-aggregation-enable</name>

<value>true</value>

</property>

<property>

<description>Where to aggregate logs to.</description>

<name>yarn.nodemanager.remote-app-log-dir</name>

<value>/app/hadoop/logs</value>

</property>

<property>

<description>

Amount of physical memory, in MB, that can be allocated

for containers.

</description>

<name>yarn.nodemanager.resource.memory-mb</name>

<value>4096</value>

</property>

<property>

<description>

Number of CPU cores that can be allocated

for containers.

</description>

<name>yarn.nodemanager.resource.cpu-vcores</name>

<value>4</value>

</property>

<property>

<description>the valid service name should only contain a-zA-Z0-9\_ and can not start with numbers</description>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</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>

### 修改slaves文件

1. 打开文件 ***vi /app/hadoop/etc/hadoop/slaves***
2. 添加DataNode节点主机名，每行一个，例如
3. host212
4. host213
5. host214
6. 保存并退出，:wq

### 修改fairscheduler.xml文件

1. 打开配置文件  ***vi /app/hadoop/etc/hadoop/fairscheduler.xml***
2. 在文件中追加如下内容（如存在下述内容则忽略此步）：

|  |
| --- |
| <?xmlversion="1.0"?>  <allocations>  <queuename="infrastructure">  <minResources>1024 mb, 2 vcores </minResources>  <maxResources>1536 mb, 4 vcores </maxResources>  <maxRunningApps>200</maxRunningApps>  <minSharePreemptionTimeout>300</minSharePreemptionTimeout>  <weight>1.0</weight>  <aclSubmitApps>root,yarn,search,hdfs</aclSubmitApps>  </queue>  <queuename="tool">  <minResources>1024 mb, 3 vcores</minResources>  <maxResources>2048 mb, 4 vcores</maxResources>  </queue>  </allocations> |

1. 保存并退出 ***:wq。***

### 文件分发

上述所有步骤完成后，依次将配置好的hadoop目录完整复制到其他服务器，包括所有的NameNode和DataNode节点的app目录下。

***scp -r /app/hadoop root@host212:/app*** (红色字体修改为目标主机名或IP)

### 集群启动

#### 首次启动

**第一次启动Hadoop集群时，按以下步骤执行：**

如zookeeper没有启动，则各个zookeeper服务器节点通过执行***cd /app/zookeeper/bin***命令移动到zookeeper的bin目录下，执行.***/zkServer.sh start &*** 启动zookeeper服务

1. 执行hdfs zkfc -formatZK命令，格式化ZooKeeper集群，在ZooKeeper集群上建立HA的相应节点(master)
2. 执行以下命令启动JournalNode集群(红字部分为JournalNode集群服务器主机名，与hdfs-site.xml文件内dfs.namenode.shared.edits.dir参数信息配置一致)：

hadoop-daemons.sh --hostnames 'host210 host211 host212' start journalnode

1. 在NameNode Master主节点服务器host210上执行以下命令，启动NameNode

***hdfs namenode -format -clusterId cluster1***

***hadoop-daemon.sh start namenode***

1. 在NameNode Master备节点服务器host211上执行以下命令，启动NameNode

***hdfs namenode -bootstrapStandby***

***hadoop-daemon.sh start namenode***

1. 在NameNode Master主节点服务器上执行以下命令，启动所有DataNode(红字部分为DataNode集群服务器主机名)

hadoop-daemons.sh --hostnames 'host212 host213 host214' start datanode

1. 在NameNode Master主节点服务器上执行以下命令，启动故障切换(红字部分为NameNode集群服务器主机名)

hadoop-daemons.sh --hostnames 'host210 host211' start zkfc

1. 在NameNode Master主节点服务器上执行以下命令，启动Resource Manager(红字部分为NameNode集群服务器主机名)

yarn-daemons.sh --hostnames 'host210 host211' start resourcemanager

1. 在NameNode Master主节点服务器上执行以下命令，启动所有节点管理器(红字部分为所有Hadoop集群服务器主机名)

yarn-daemons.sh --hostnames 'host210 host211 host212 host213 host214' start nodemanager

#### 停机后启动

**日常停机状态启动Hadoop集群时，按以下步骤执行：**

如zookeeper没有启动，则各个zookeeper服务器节点通过执行cd /app/zookeeper/bin命令移动到zookeeper的bin目录下，执行./zkServer.sh start & 启动zookeeper服务

1. 执行以下命令启动JournalNode集群(红字部分为JournalNode集群服务器主机名，与hdfs-site.xml文件内dfs.namenode.shared.edits.dir参数信息配置一致)：

hadoop-daemons.sh --hostnames 'host210 host211 host212' start journalnode

1. 执行以下命令启动所有NameNode节点(红字部分为NameNode集群服务器主机名)：

hadoop-daemons.sh --hostnames 'host210 host211' start namenode

1. 在NameNode Master主节点服务器上执行以下命令，启动故障切换(红字部分为NameNode集群服务器主机名)

hadoop-daemons.sh --hostnames 'host210 host211' start zkfc

1. 在NameNode Master主节点服务器上执行以下命令，启动所有DataNode(红字部分为DataNode集群服务器主机名)

hadoop-daemons.sh --hostnames 'host212 host213 host214' start datanode

1. 在NameNode Master主节点服务器上执行以下命令，启动Resource Manager(红字部分为NameNode集群服务器主机名)

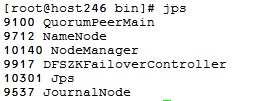
yarn-daemons.sh --hostnames 'host210 host211' start resourcemanager

1. 在NameNode Master主节点服务器上执行以下命令，启动所有节点管理器(红字部分为所有Hadoop集群服务器主机名)

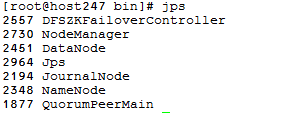
yarn-daemons.sh --hostnames 'host210 host211 host212 host213 host214' start nodemanager

#### 验证服务状态

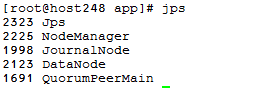
1. NameNode主节点：



1. NameNode备节点



1. 数据节点

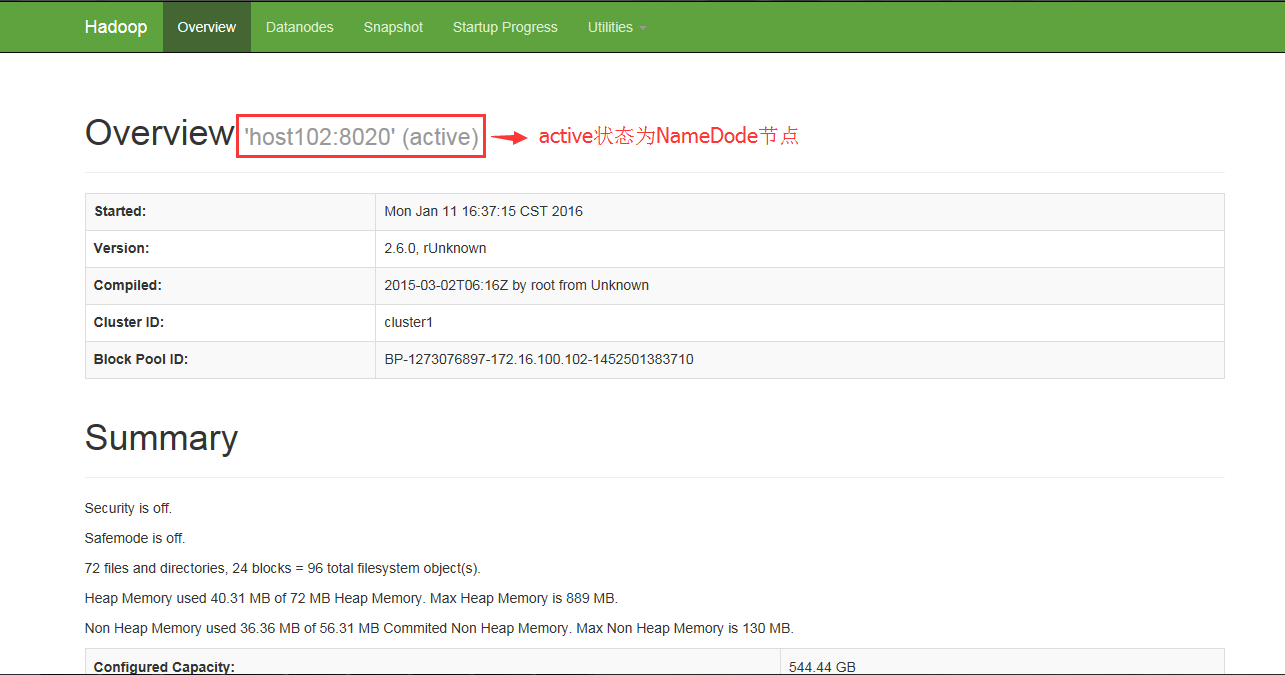


查看DFS信息 ***hdfs dfsadmin -report***  会显示如下类似的信息：

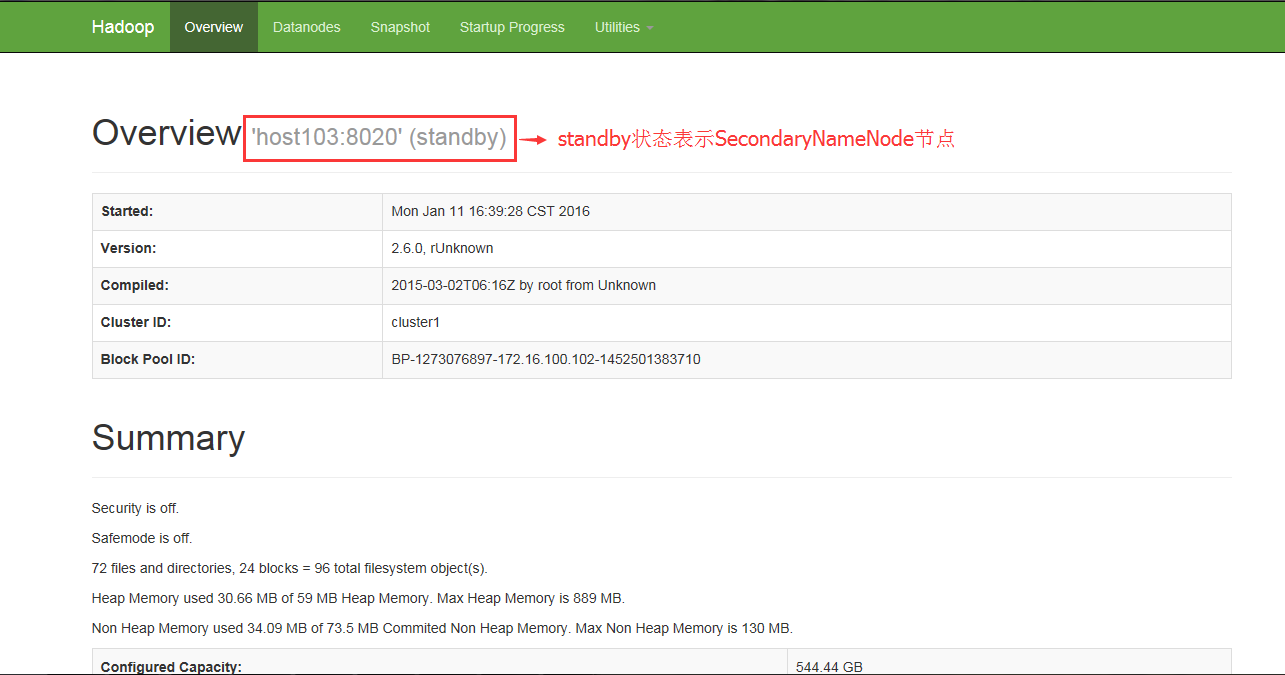
|  |
| --- |
| Configured Capacity: 584591757312 (544.44 GB)  Present Capacity: 443525192310 (413.07 GB)  DFS Remaining: 443498729472 (413.04 GB)  DFS Used: 26462838 (25.24 MB)  DFS Used%: 0.01%  Under replicated blocks: 0  Blocks with corrupt replicas: 0  Missing blocks: 0  -------------------------------------------------  Live datanodes (3):  Name: 172.16.100.105:50010 (host105)  Hostname: host105  Decommission Status : Normal  Configured Capacity: 194863919104 (181.48 GB)  DFS Used: 7718203 (7.36 MB)  Non DFS Used: 47020341957 (43.79 GB)  DFS Remaining: 147835858944 (137.68 GB)  DFS Used%: 0.00%  DFS Remaining%: 75.87%  Configured Cache Capacity: 0 (0 B)  Cache Used: 0 (0 B)  Cache Remaining: 0 (0 B)  Cache Used%: 100.00%  Cache Remaining%: 0.00%  Xceivers: 9  Last contact: Fri Jan 15 16:06:39 CST 2016 |

#### Hadoop管理网站确认

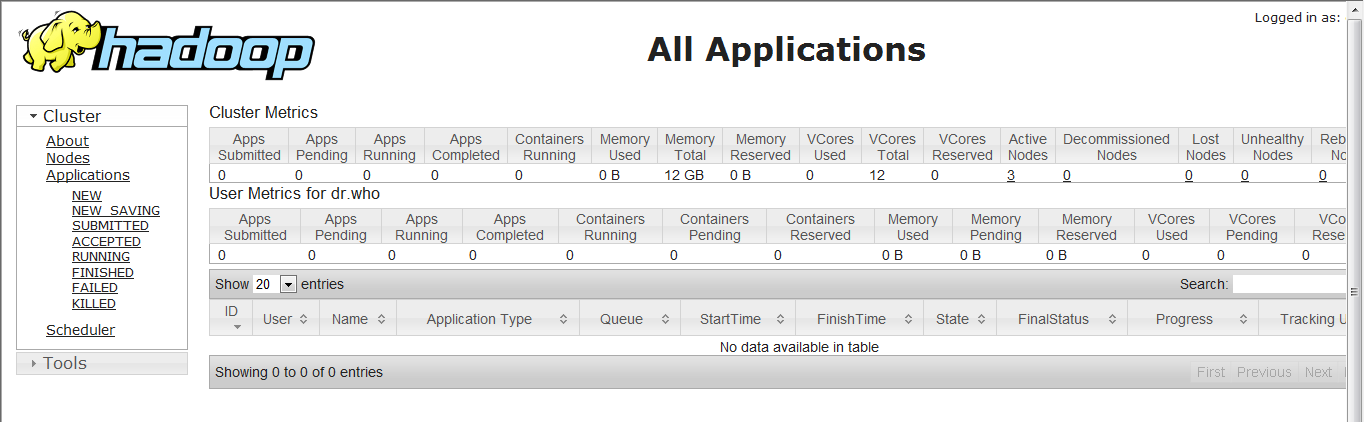
浏览器地址栏中输入：[**http://NameNodeIP:50070/**](http://namenodeip:50070/) （NameNodeIP改为NameNode节点服务器的IP即可），会出现如下界面：



浏览器地址栏中输入：[**http://SecondaryNameNodeIP:50070/**](http://namenodeip:50070/) （SecondaryNameNodeIP改为SecondaryNameNode节点服务器的IP即可），会出现如下界面：



浏览器地址栏中输入：[**http://NameNodeIP:9026/**](http://namenodeip:50070/)**cluster** ，会出现如下界面：



## Spark安装配置

如无特别说明，以下节点仅限spark相关节点，具体见服务器规划一节。

### scala安装

1. 从软件包中获取scala-2.10.5.tgz包，并上传至spark master节点host210 解压至/app目录下
2. 执行以下命令配置环境变量：

***echo 'export SCALA\_HOME=/app/scala-2.10.5'>> /etc/profile***

***echo 'export PATH=$SCALA\_HOME/bin:$PATH'>> /etc/profile***

1. 执行scp命令将文件复制到worker节点：

***scp –r scala-2.10.5 host117:/app/***

***scp –r scala-2.10.5 host118:/app/***

***scp /etc/profile host117:/etc/profile***

***scp /etc/profile host118:/etc/profile***

1. 在各节点激活profile文件：

***source /etc/profile***

1. 输入scala –version，如果输出以下信息，则说明安装成功

Scala code runner version 2.10.5 -- Copyright 2002-2013, LAMP/EPFL

### Spark安装

1. 切换至master节点host210,从软件包中获取spark-1.5.2-bin-hadoop2.6.tgz，并上传至app目录；
2. 在app目录解压缩，并更改目录名称；

***tar -zxvf spark-1.5.2-bin-hadoop2.6.tgz***

***mv spark-1.5.2-bin-hadoop2.6 spark1.5.2***

1. 在/app/spark1.5.2目录中配置slave文件；

***cp ./conf/slaves.template ./conf/slaves***

在slaves文件中追加worker节点配置

***echo host212 >> ./conf/slaves***

***echo host213 >> ./conf/slaves***

***echo host214 >> ./conf/slaves***

1. 在/app/spark1.5.2目录中，配置spark-env.sh;

***cp ./conf/spark-env.sh.template ./conf/spark-env.sh***

1. 使用vi 编辑spark-env文件，其中标红字段根据work节点cpu核数一致、内存情况调整为work节点内存减去5g；标蓝部分根据部署zookeeper服务地址调整。

*SPARK\_LOCAL\_DIRS=/app/spark1.5.2*

*export HADOOP\_HOME=/app/hadoop*

*export HADOOP\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop*

*export SPARK\_JAR=/app/spark1.5.2/lib/spark-assembly-1.5.2-hadoop2.6.0.jar*

*SPARK\_WORKER\_CORES=24*

*SPARK\_WORKER\_MEMORY=12g*

*SPARK\_DAEMON\_JAVA\_OPTS=' -Dspark.local.dir=$SPARK\_LOCAL\_DIR -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -Xloggc:$SPARK\_HOME/logs/gc.log -XX:+UseConcMarkSweepGC -XX:+UseCMSCompactAtFullCollection -XX:CMSInitiatingOccupancyFraction=60 -Dspark.deploy.recoveryMode=ZOOKEEPER -Dspark.deploy.zookeeper.url=host215:2181,host216:2181,host217:2181 -Dspark.deploy.zookeeper.dir=/spark'*

*SPARK\_PID\_DIR=/app/spark1.5.2/pid*

蓝色字体请根据zookeeper实际部署情况进行配置*。*

1. 在/app/spark1.5.2目录中，配置log4j.properties;

***vi ./conf/log4j.properties***

将下面配置内容粘贴进去，然后保存退出

log4j.rootLogger=INFO,LogFile

log4j.appender.LogFile=org.apache.log4j.RollingFileAppender

log4j.appender.LogFile.Threshold=INFO

log4j.appender.LogFile.File=${user.dir}/logs/infoFile.log

log4j.appender.LogFile.MaxFileSize=500MB

log4j.appender.LogFile.MaxBackupIndex=10

log4j.appender.LogFile.layout=org.apache.log4j.PatternLayout

log4j.appender.LogFile.layout.ConversionPattern=%d{yyyy/MM/dd HH\:mm\:ss} %5p %c{1}\:%L - %m%n

1. 向备master及worker节点，分发spark安装文件及配置；

***scp –r spark1.5.2 host211:/app/***

***scp –r spark1.5.2 host212:/app/***

***scp –r spark1.5.2 host213:/app/***

***scp –r spark1.5.2 host214:/app/***

1. 在master及worker节点设置并生效环境变量；

***echo 'export SPARK\_HOME=/app/spark1.5.2'>> /etc/profile***

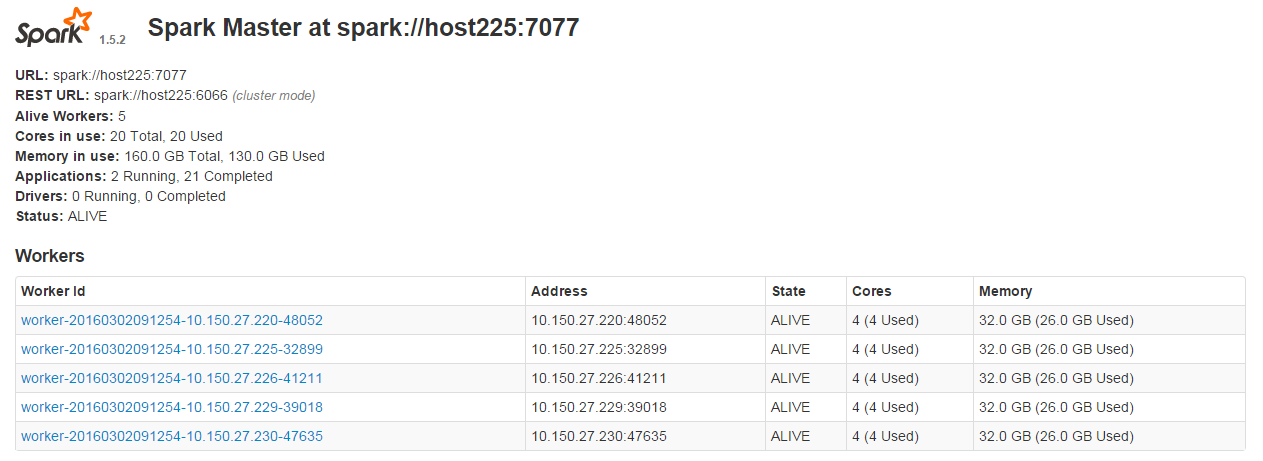
***echo 'export PATH=$SPARK\_HOME/bin:$PATH'>> /etc/profile***

***source /etc/profile***

1. 在master1节点的/app/spark1.5.2目录下，执行***./sbin/start-all.sh***
2. 登陆master2服务节点，在/app/spark1.5.2目录中，执行***./sbin/start-master.sh***
3. 服务验证：

如master启动成功，访问http://host210:8080，弹出以下界面：

红色为master节点ip



## ElasticSearch安装配置

### 软件安装及集群配置

在ElasticSearch各节点执行以下安装操作

1. 在服务器上执行以下命令创建目录；

***mkdir -p /data1/elasticsearch/indexdata***

***mkdir -p /data1/elasticsearch/work***

***mkdir -p /data1/elasticsearch/logs***

1. 配置环境变量，设置系统环境变量使es可以锁内存，红色部分为es使用内存（设置为机器物理内存的1/3）

***echo "ulimit -l unlimited">> /etc/profile***

***echo 'export ES\_HEAP\_SIZE=16g' >> /etc/profile./***

***. /etc/profile***

1. 安装软件及配置；
2. 从软件包中获取elasticsearch-2.2.0.tar.gz包和elasticsearch-head-master.zip包，并上传至服务器/app目录下
3. 在/app目录执行解压缩
4. tar -xzvfelasticsearch-2.2.0.tar.gz
5. 更改目录名称
6. mv elasticsearch-2.2.0 elasticsearch
7. 追加以下内容到配置文件 /app/elasticsearch/config/elasticsearch.yml 中

*cluster.name: ehl\_elasticsearch*

*path.data: /data1/elasticsearch/indexdata*

*path.work: /data1/elasticsearch/work*

*path.logs: /data1/elasticsearch/logs*

*bootstrap.mlockall: true*

*discovery.zen.fd.ping\_interval: 10s*

*discovery.zen.fd.ping\_retries: 10*

*index.mapper.dynamic: false*

*node.name: ${HOSTNAME}*

*network.host: \_eth0\_*

*discovery.zen.ping.unicast.hosts: ["host215", "host216", "host217"]*

各配置属性‘:’符号后需添加空格占位符。

cluster.name 为集群名称，各节点一致

Network.host 为主机名对应网卡，执行ifconfig可查看，默认ens160

discovery.zen.ping.unicast.hosts为节点列表

index.max\_result\_window为分页查询支持的最大数据量，通常取默认值1000000即可。（查控分析过车记录查询功能中如单次查询数据集超过该值，不能直接跳转至结果最后一页）。

1. 使用scp命令将修改后的elasticsearch目录拷贝至其他节点
2. 安装ElasticSearch插件

(1)安装head插件

***/app/elasticsearch/bin/plugin install file:/app/elasticsearch-head-master.zip***

(2)安装ik插件

下载ik插件ik.tar.gz

******

拷贝***ik.tar.gz***到 ***/app/elasticsearch/bin/plugin***

解压***ik***插件 ***tar zxvf ik.tar.gz***

删除压缩包 ***rm -f ik.tar.gz***

修改 ***ik/plugin-descriptor.properties***文件中的***elasticsearch.version=2.2.0***（和现在安装elasticsearch版本一致）

1. 启动ElasticSearch服务

在/app/elasticsearch/bin下执行以下命令

***./elasticsearch -d -Des.insecure.allow.root=true***

### 创建旅游大数据表及别名

在ElasticSearch中任意一节点如host215，执行建库、表操作

1. 创建pass\_car表及数据库；

这里标黄色的部分根据实际情况修改，host215改成elasticsearch集群任意一个节点地址，number\_of\_shards的值改为n\*10(n为work数).

curl -XPUT http://10.2.25.236:9200/ehl\_trip

curl -XPUT http://http://10.2.25.236:9200/ehl\_trip/\_settings -d '{ "index" : { "max\_result\_window" : 100000000}}'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/jqtdinfo/\_mapping?pertty -d '{

"jqtdinfo": {

"properties": {

"tdid": {

"type": "string",

"index": "not\_analyzed"

},

"jqid": {

"type": "string",

"index": "not\_analyzed"

},

"jqname": {

"type": "string",

"index": "not\_analyzed"

},

"starttime": {

"type": "long",

"index": "not\_analyzed"

},

"endtime": {

"type": "long",

"index": "not\_analyzed"

},

"city": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/humaninfo/\_mapping?pertty -d '{

"humaninfo": {

"properties": {

"tdid": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/deviceinfo/\_mapping?pertty -d '{

"deviceinfo": {

"properties": {

"tdid": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/weekfrequencyinfo/\_mapping?pertty -d '{

"weekfrequencyinfo": {

"properties": {

"tdid": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/\_mapping?pertty -d '{

"deviceinfo": {

"properties": {

"deviceType": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/humaninfo/\_mapping?pertty -d '{

"humaninfo": {

"properties": {

"tdid": {

"type": "string",

"index": "not\_analyzed"

}

"gendar": {

"type": "string",

"index": "not\_analyzed"

},

"gendarWeight": {

"type": "integer",

"index": "not\_analyzed"

},

"age": {

"type": "string",

"index": "not\_analyzed"

},

"ageWeight": {

"type": "float",

"index": "not\_analyzed"

},

"profession": {

"type": "string",

"index": "not\_analyzed"

},

"professionWeight": {

"type": "float",

"index": "not\_analyzed"

},

"marrage": {

"type": "string",

"index": "not\_analyzed"

},

"marrageWeight": {

"type": "float",

"index": "not\_analyzed"

},

"car": {

"type": "string",

"index": "not\_analyzed"

},

"carWeight": {

"type": "float",

"index": "not\_analyzed"

},

"beingParent": {

"type": "string",

"index": "not\_analyzed"

},

"beingParentWeight": {

"type": "float",

"index": "not\_analyzed"

},

"hasBaby": {

"type": "string",

"index": "not\_analyzed"

},

"hasBabyWeight": {

"type": "float",

"index": "not\_analyzed"

},

"hasoneBaby": {

"type": "string",

"index": "not\_analyzed"

},

"hasoneBabyWeight": {

"type": "float",

"index": "not\_analyzed"

},

"hasTwoBaby": {

"type": "string",

"index": "not\_analyzed"

},

"hasTwoBabyWeight": {

"type": "float",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/deviceinfo/\_mapping?pertty -d '{

"deviceinfo": {

"properties": {

"standardModel": {

"type": "string",

"index": "not\_analyzed"

},

"screen": {

"type": "string",

"index": "not\_analyzed"

},

"price": {

"type": "string",

"index": "not\_analyzed"

},

"functiontype": {

"type": "string",

"index": "not\_analyzed"

},

"standardBrand": {

"type": "string",

"index": "not\_analyzed"

},

"timeToMarket": {

"type": "string",

"index": "not\_analyzed"

},

"hardwareType": {

"type": "string",

"index": "not\_analyzed"

},

"deviceType": {

"type": "string",

"index": "not\_analyzed"

},

"cnBrand": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/weekfrequencyinfo/\_mapping?pertty -d '{

"weekfrequencyinfo": {

"properties": {

"week": {

"type": "string",

"index": "not\_analyzed"

},

"latlng": {

"type": "string",

"index": "not\_analyzed"

},

"frequency": {

"type": "integer",

"index": "not\_analyzed"

},

"hour": {

"type": "string",

"index": "not\_analyzed"

},

"weekday": {

"type": "integer",

"index": "not\_analyzed"

},

"locationname": {

"type": "string",

"index": "not\_analyzed"

},

"jqid": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/jqregularcity/\_mapping?pertty -d '{

"jqregularcity": {

"properties": {

"tdid": {

"type": "long",

"index": "not\_analyzed"

},

"starttime": {

"type": "long",

"index": "not\_analyzed"

},

"count": {

"type": "integer",

"index": "not\_analyzed"

},

"citycode": {

"type": "string",

"index": "not\_analyzed"

},

"cityname": {

"type": "string",

"index": "not\_analyzed"

},

"jqid": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

curl -XPOST http://10.2.25.236:9200/ehl\_trip/regularshowcity/\_mapping?pertty -d '{

"regularshowcity": {

"properties": {

"tdid": {

"type": "string",

"index": "not\_analyzed"

},

"month": {

"type": "long",

"index": "not\_analyzed"

},

"city": {

"type": "string",

"index": "not\_analyzed"

},

"citycode": {

"type": "string",

"index": "not\_analyzed"

},

"provice": {

"type": "string",

"index": "not\_analyzed"

},

"provicecode": {

"type": "string",

"index": "not\_analyzed"

},

"city2": {

"type": "string",

"index": "not\_analyzed"

},

"citycode2": {

"type": "string",

"index": "not\_analyzed"

},

"provice2": {

"type": "string",

"index": "not\_analyzed"

},

"provicecode2": {

"type": "string",

"index": "not\_analyzed"

},

"city3": {

"type": "string",

"index": "not\_analyzed"

},

"citycode3": {

"type": "string",

"index": "not\_analyzed"

},

"provice3": {

"type": "string",

"index": "not\_analyzed"

},

"provicecode3": {

"type": "string",

"index": "not\_analyzed"

}

}

}

}

'

### 查询窗口设置

在任一elasticsearch节点，在shell中输入以下命令，修改默认查询窗口大小。

*curl -XPUT "http://10.150.27.215:9200/ehl\_trip/\_settings" -d '{ "index" : { "max\_result\_window" : 1000000 } }'*

其中红色部分为elasticsearch集群节点ip，根据实际情况修改。如执行成功，返回

{"acknowledged":true}

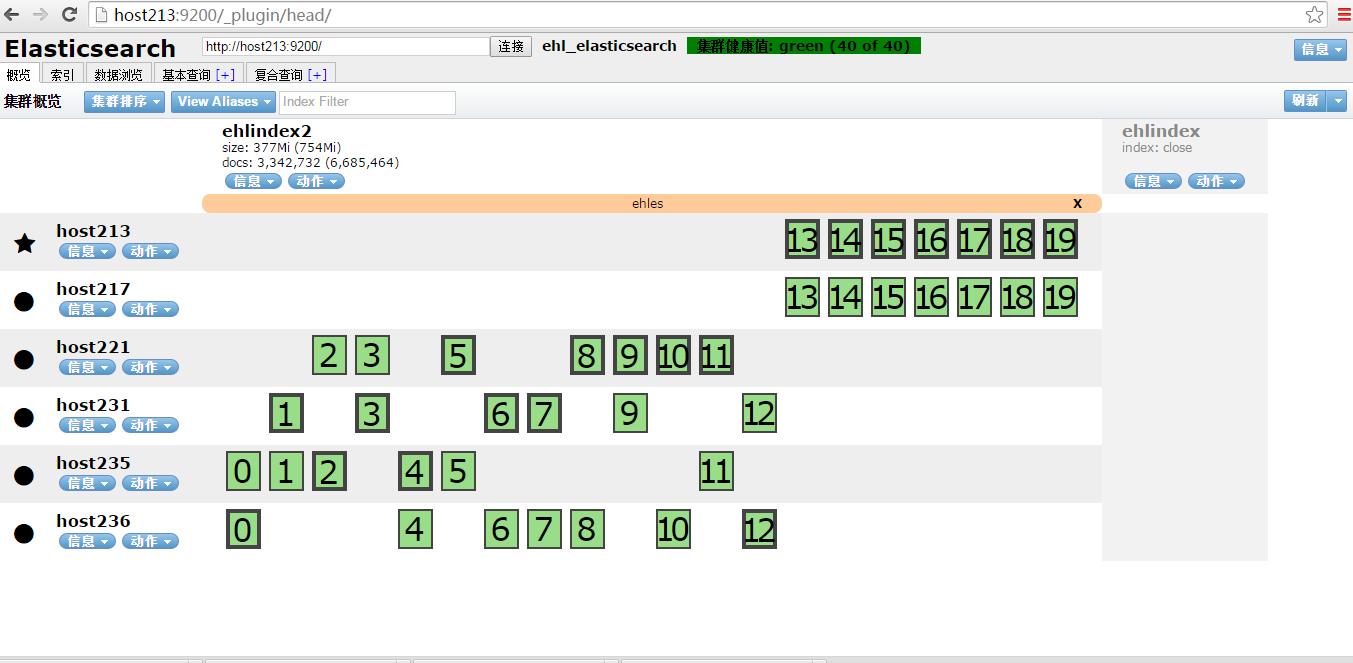
### 验证集群状态

使用浏览器访问

[http://host215:9200/\_plugin/head/](http://host213:9200/_plugin/head/)

红色字体代表，安装插件的ElasticSearch集群节点。

正常情况显示如下：



## Kafka安装配置

### 软件包部署

从软件包中获取kafka\_2.11-0.8.2.0.tgz包，并上传至服务器/app目录下，然后执行以下操作：

1. 执行***tar xvf kafka\_2.11-0.8.2.0.tgz***进行解压缩
2. 执行***mv kafka\_2.11-0.8.2.0 kafka***更改目录名称

### 修改server.properties配置

1. 打开配置***vi /app/kafka/config/server.properties***
2. 修改配置文件如下：

*# Kafka集群顺序编号，从0开始，所有Kafka服务器此编号不可重复*

*broker.id=0*

*#设置该节点IP地址*

*host.name=10.150.27.250*

*log.dirs=/data1/kafka-logs*

*#分区数，根据实际情况（磁盘数）设定*

*num.partitions=7*

*#更改为zookeeper服务器地址 zookeeper.connect=10.150.27.215:2181,10.150.27.216:2181,10.150.27.217:2181*

注意：num.partitions为集群的总分区数，这里设置n\*7(n为kafka机器总数)

### 修改kafka-run-class.sh配置文件

1. 打开配置文件 ***vi /app/kafka/bin/kafka-run-class.sh***
2. 修改配置文件中的如下内容，其他保持默认即可：

*# Memory options*

*if [ -z "$KAFKA\_HEAP\_OPTS" ]; then*

*KAFKA\_HEAP\_OPTS="-Xmx1024M"*

*fi*

按照注释描述，结合服务器内存资源配置kafka内存。

1. 保存并退出 ***:wq***

### 启动kafka

1. 进入bin目录 ***cd /app/kafka/bin***
2. 启动kafka ***nohup ./kafka-server-start.sh ../config/server.properties &***
3. 查看服务是否启动成功 ***jps***

如果看到类似 37400 Kafka 则表示服务已正常启动（进程ID号可变）

1. 使用scp命令复制完整的kafka目录到所有其他服务器

***scp -r /app/kafka root@host1:/app***(红色字体修改为目标kafka主机名或IP)

1. 按4.5.1章节描述，依次在各目标服务器上修改server.properties文件中broker.id和host.name两项的值。

重复上述1、2、3步骤，依次启动各目标服务器的kafka服务

## Kafka Manager安装

### 安装kafka-manager

获取安装包kafka-manager-1.0-SNAPSHOT.zip，上传至/app目录下，然后执行如下操作：

1. 进入/app目录 cd /app

解压kafka-manager 安装包 ***unzip kafka-manager-1.0-SNAPSHOT.zip***

1. 重命名kafka-manager目录 ***mv kafka-manager-1.0-SNAPSHOT kafka-manager***

### 修改配置文件application.conf

1. 打开配置文件 ***vi /app/kafka-manager/conf/application.conf***
2. 修改如下的内容，其他保持默认即可：

*kafka-manager.zkhosts="kafka-manager-zookeeper:2181"*

如黄色标示的内容，改为任意一个可用的zookeeper节点主机名或IP，该地址用于kafka-manager数据存储

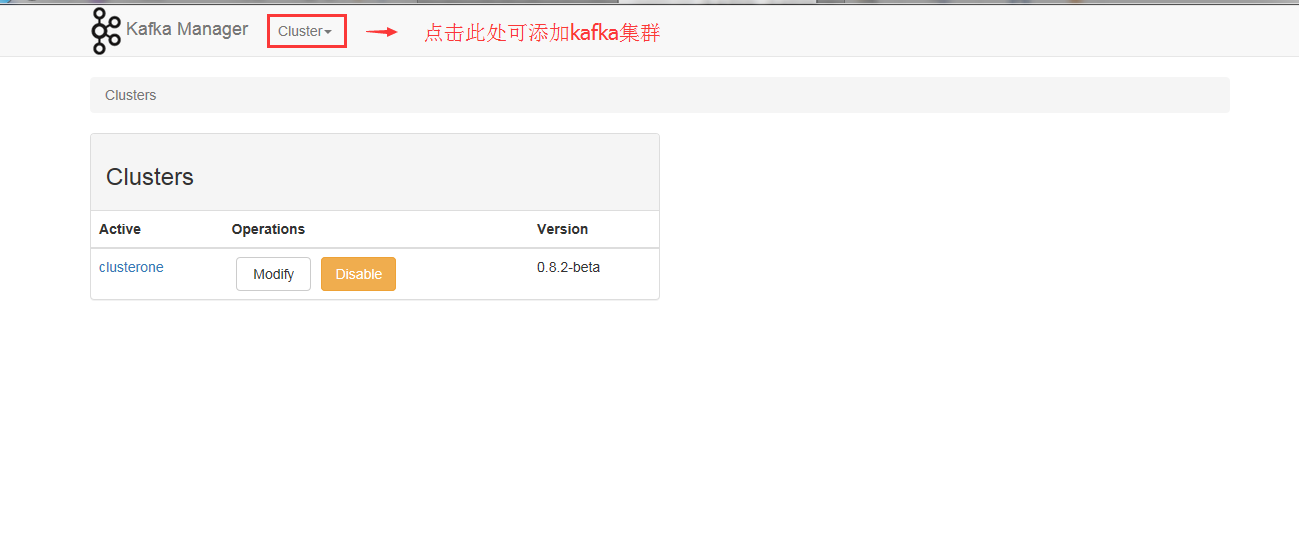
1. 保存并退出 ***:wq***

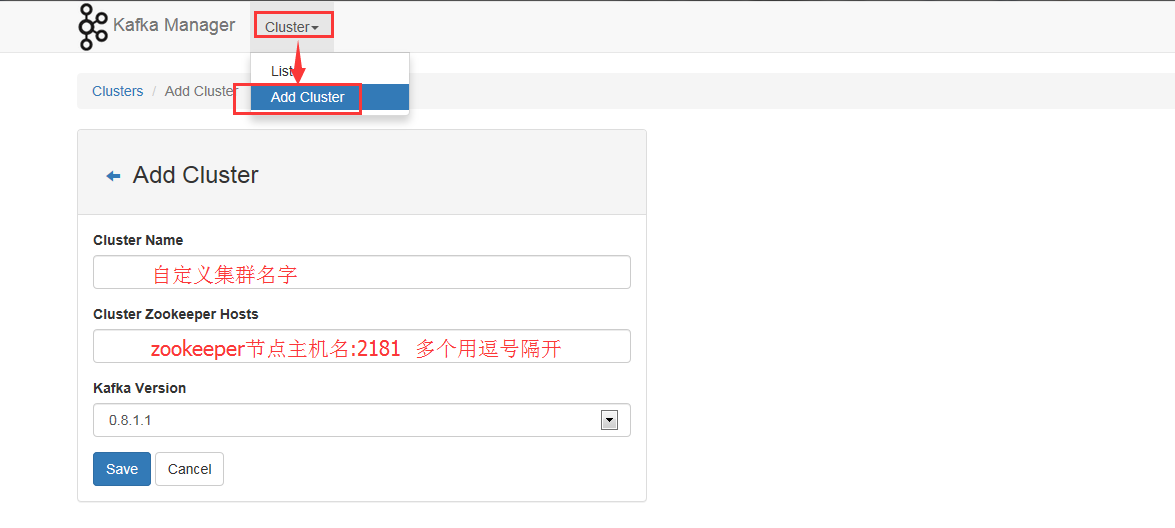
### 启动kafka-manager

1. 进入bin目录 ***cd /app/kafka-manager/bin***
2. 启动kafka-manager **./kafka-manager -Dconfig.file=../conf/application.conf &**查看服务是否启动成功 ***jps***
3. 如果看到类似 10551 NettyServer 则表示服务已正常启动

### Kafka-manager管理网站确认

浏览器地址栏中输入：[**http://IP:9**](http://namenodeip:50070/)**000** （IP改为Kafka-manager节点服务器的IP即可），会出现如下界面：





## Redis安装配置

### 软件依赖包安装

可从CentOS-7-x86\_64-DVD-1503-01.iso安装镜像或或本版本安装包中获取，并以下面顺序安装依赖的rpm包。

***rpm -ivh mpfr-3.1.1-4.el7.x86\_64.rpm***

***rpm -ivh libmpc-1.0.1-3.el7.x86\_64.rpm***

***rpm -ivh cpp-4.8.3-9.el7.x86\_64.rpm***

***rpm -ivh kernel-headers-3.10.0-229.el7.x86\_64.rpm***

***rpm -ivh glibc-headers-2.17-78.el7.x86\_64.rpm***

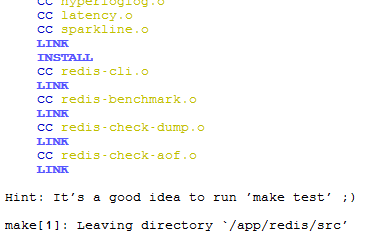
***rpm -ivh glibc-devel-2.17-78.el7.x86\_64.rpm***

***rpm -ivh gcc-4.8.3-9.el7.x86\_64.rpm***

### Redis编译安装

从软件包中获取redis-3.0.1.tar.gz包，并上传至服务器/app目录下

1. 执行***tar xvf redis-3.0.1.tar.gz***进行解压缩
2. 执行***mv redis-3.0.1 redis***更改目录名称
3. 执行***cd /app/redis***移动到redis目录下
4. 执行***make MALLOC=libc***命令进行编译



### 修改redis.conf配置

执行***vi redis.conf***命令，修改配置文件如下：

daemonize no

修改为

daemonize yes

### 启动resis服务

执行***cd /app/redis/src***进入目录

执行***./redis-server &*** 启动redis服务

验证服务状态：

在/app/redis/src目录下执行***./redis-cli***命令，显示：



## RabbitMq安装配置

从软件版本中获取***erlang-17.4-1.el6.x86\_64.rpm***和***rabbitmq-server-3.5.0-1.noarch.rpm***

### 安装erlang

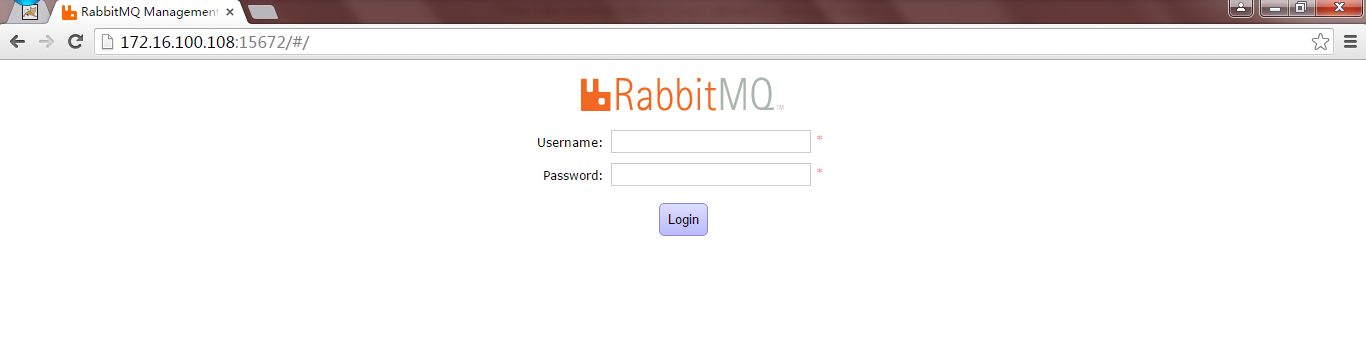
1. 进入/app目录 ***cd /app***
2. 安装erlang ***rpm -ivh --nodeps --force erlang-17.4-1.el6.x86\_64.rpm***

### 安装rabbitmq

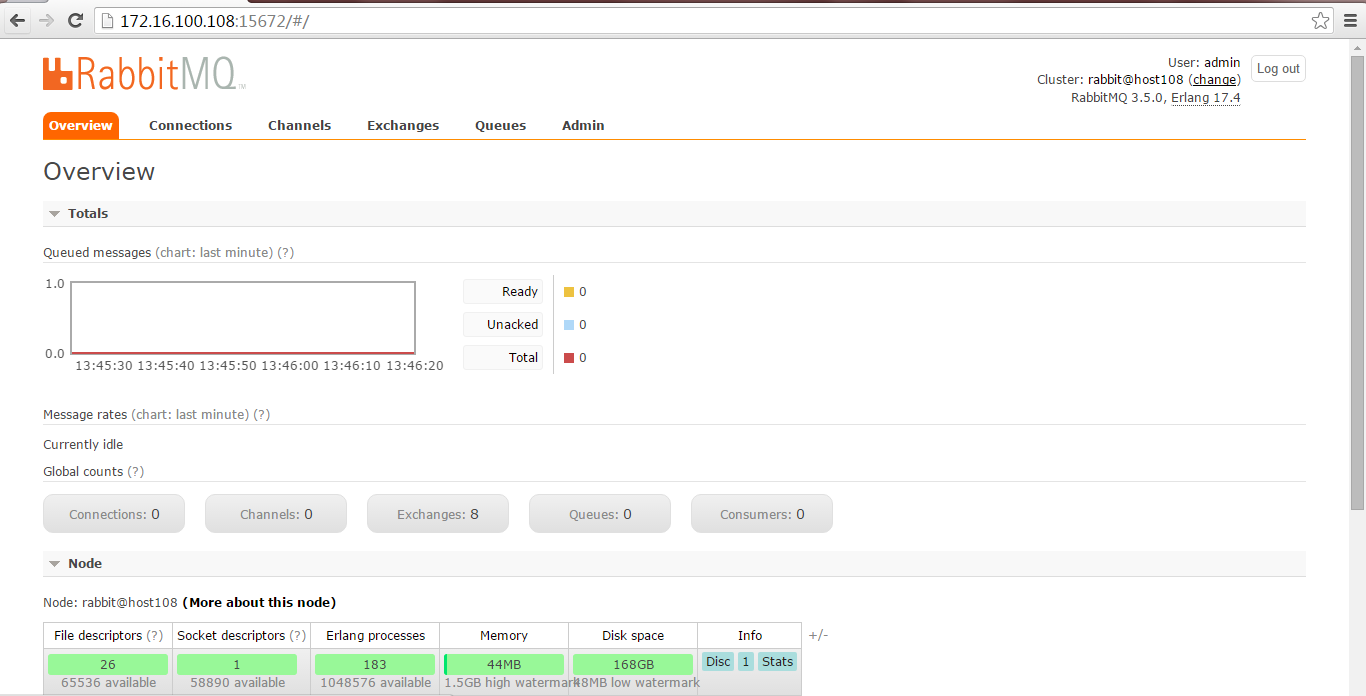
1. 进入/app目录 ***cd /app***
2. 安装rabbitmq***rpm -ivh --nodeps –force rabbitmq-server-3.5.0-1.noarch.rpm***
3. 启用管理插件 ***rabbitmq-plugins enable rabbitmq\_management***
4. 创建admin用户并赋权限
5. ***rabbitmqctl add\_user admin admin***
6. ***rabbitmqctl set\_user\_tags admin administrator***
7. ***rabbitmqctl set\_permissions -p / admin ".\*"".\*"".\*"***
8. 启动rabbitmq服务 ***service rabbitmq-server start***

### Rabbitmq网站确认

浏览器地址栏中输入：[**http://RabbitmqIP:15672/**](http://namenodeip:50070/)（RabbitmqIP改为主节点rabbitmq服务器的IP即可），会出现如下界面



admin用户登录以后，如下图所示：



## Tomcat安装

1. 获取***apache-tomcat-7.0.57.tar.gz***，把apache-tomcat-7.0.57.tar.gz解压至/app/tomcat
2. 修改tomcat端口为80,修改编码为UTF-8

tomcat/conf/server.xml

<Connector port="80" maxHttpHeaderSize="8192"

maxThreads="150" minSpareThreads="25" maxSpareThreads="75"

enableLookups="false" redirectPort="8443" acceptCount="100"

connectionTimeout="20000" disableUploadTimeout="true"

URIEncoding="UTF-8"/>

1. 把tomcat注册为服务

***cp tomcat/bin/catalina.sh /etc/init.d/tomcat***

1. 启动tomcat

***service tomcat start***

安装好之后可以复制到其他机器

# 业务软件部署

## 资源管理平台web-dap

1. 拷贝发布版web-dap-1.0-SNAPSHOT.war包到包到TOMCAT的webapps目录下
2. 执行mv web-dap-1.0-SNAPSHOT.war web-dap.war
3. 启动tomcat服务，解压war包.
4. 修改conf.properties文件

路径tomcat/webapps/web-dap/WEB-INF/classes/config/conf.properties，其中红色标示为必须更改或者需确认的配置（下同）

#rabbitmq配置

rabbitmqip=59.110.63.115  
rabbitmq.user=admin  
rabbitmq.pwd=admin  
httpsessiontimeout=86400

1. 修改dbglzx.properties

路径tomcat/webapps\web-dap\WEB-INF\classes\config\dbglzx.properties

# 系统管理中心数据源配置

glzx.jdbc.url=jdbc:mysql://10.2.25.175:3306/ehl-manager?characterEncoding=utf8&autoReconnect=true&failOverReadOnly=false

glzx.jdbc.driverClassName=com.mysql.jdbc.Driver

glzx.jdbc.username=root

glzx.jdbc.password=mysql\_123

1. 修改dbitgs.properties

路径tomcat/webapps\web-dap\WEB-INF\classes\config\dbitgs.properties

# 平台数据源配置

jdbc.url=jdbc:mysql://10.2.25.175:3306/cply?characterEncoding=utf8&autoReconnect=true&failOverReadOnly=false

jdbc.driverClassName=com.mysql.jdbc.Driver

jdbc.username=root

jdbc.password=mysql\_123

1. 修改gis.properties

路径tomcat/webapps\web-dap\WEB-INF\classes\config\gis.properties

# 栅格地图服务地址

map\_client\_url=http://10.2.25.244:8081/EHL\_MapClient/maps?charset\=UTF-8

1. 修改video.properties

路径tomcat/webapps\web-dap\WEB-INF\classes\config\video.properties

#视频配置平台数据源配置

video.jdbc.url=jdbc:mysql://101.201.236.218:3306/cvrdb?characterEncoding=utf8&autoReconnect=true&failOverReadOnly=false

video.jdbc.driverClassName=com.mysql.jdbc.Driver

video.jdbc.username=root

video.jdbc.password=123456

#默认的用户名- 密码-端口号

video.username=root

video.password=123456

video.serviceport=5061

1. 视频服务器配置路径修改

vedioServiceConfig.jsp文件，位于webpage\common目录下

修改视频服务器的IP和端口

<%@ **page language**="**java**" **contentType**="**text/html; charset=UTF-8**" **pageEncoding**="**UTF-8**"%>  
<%@ **taglib prefix**="**c**" **uri**="**http://java.sun.com/jstl/core\_rt**" %>  
**<%** String path = request.getContextPath();  
 String vedioServiceIp=**"101.201.236.218"**;  
 String vedioServicePort=**"5060"**;  
 String basePath = request.getScheme()+**"://"**+request.getServerName()+**":"**+request.getServerPort()+**"/"**;  
**%>**<**script type="text/javascript"**>  
 **var *vedioServiceIp*** = **'<%=**vedioServiceIp**%>'**;  
 **var *vedioServicePort*** = **'<%=**vedioServicePort**%>'**;  
</**script**>