# zookeeper单机版安装

## 获取zookeeper安装包

官网：https://zookeeper.apache.org/releases.html#download

上传zookeeper安装包到linux

## 解压安装包

### 在/usr/local/下创建文件夹zookeeper

[root@localhost local]# mkdir zookeeper

### 解压安装包到此目录下

[root@localhost usr]# tar -zxvf zookeeper-3.4.13.tar.gz -C /usr/local/zookeeper

## 进入到zookeeper安装文件夹下

[root@localhost usr]# cd /usr/local/zookeeper/zookeeper-3.4.13

### 创建文件夹data

[root@localhost zookeeper-3.4.13]# mkdir data

## 进入conf文件夹，并修改zoo\_sample.cfg为zoo.cfg

[root@localhost zookeeper-3.4.13]# cd conf

[root@localhost conf]# mv zoo\_sample.cfg zoo.cfg

### 修改zoo.cfg配置文件

修改的地方有：

dataDir=/tmp/zookeeper ----》 修改为自己创建的那个data文件夹路径 dataDir=/usr/local/zookeeper/zookeeper-3.4.13/data

clientPort=2181 ----》 端口根据需求修改，默认2181

## 启动zookeeper

[root@localhost bin]# ./zkServer.sh start

ZooKeeper JMX enabled by default

Using config: /usr/local/zookeeper/zookeeper-3.4.13/bin/../conf/zoo.cfg

Starting zookeeper ... STARTED

### 状态查询

[root@localhost bin]# ./zkServer.sh status

ZooKeeper JMX enabled by default

Using config: /usr/local/zookeeper/zookeeper-3.4.13/bin/../conf/zoo.cfg

Mode: standalone

## 关闭zookeeper

[root@localhost bin]# ./zkServer.sh stop

ZooKeeper JMX enabled by default

Using config: /usr/local/zookeeper/zookeeper-3.4.13/bin/../conf/zoo.cfg

Stopping zookeeper ... STOPPED

# zookeeper集群搭建

## 创建/usr/local/zookeepercluster 文件夹

[root@localhost local]# mkdir zookeepercluster

## 进入改文件夹创建三个文件夹 zookeeper01 zookeeper02 zookeeper03

[root@localhost ~]# cd /usr/local/zookeepercluster/

[root@localhost zookeepercluster]# mkdir zookeeper01

[root@localhost zookeepercluster]# mkdir zookeeper02

[root@localhost zookeepercluster]# mkdir zookeeper03

[root@localhost zookeepercluster]# ll

total 0

drwxr-xr-x 2 root root 6 Jan 19 16:48 zookeeper01

drwxr-xr-x 2 root root 6 Jan 19 16:48 zookeeper02

drwxr-xr-x 2 root root 6 Jan 19 16:48 zookeeper03

## 解压压缩包到zookeeper01 中

[root@localhost usr]# tar -zxvf zookeeper-3.4.13.tar.gz -C /usr/local/zookeepercluster/zookeeper01

## 修改在安装目录中创建data目录

[root@localhost zookeeper-3.4.13]# mkdir data

## 在data目录下创建myid文件，并输入内容 1

[root@localhost data]# echo 1 >> myid

## 进入conf文件夹，修改zoo\_sample.conf 为zoo.conf

[root@localhost zookeeper-3.4.13]# cd conf

## 修改zoo.conf 文件

修改的地方

dataDir=/tmp/zookeeper ---》 修改为自己创建的那个data文件夹路径 dataDir=/usr/local/zookeepercluster/zookeeper01/zookeeper-3.4.13/data

clientPort=2181 ----》 端口根据需求修改，默认2181 ,我修改为2181 clientPort=2182

添加的东西(每个zookeeper服务器都需要添加，内容一样)

server.1=192.168.153.136:2881:3881

server.2=192.168.153.136:2882:3882

server.3=192.168.153.136:2883:3883

参数介绍

server.x=ip:port1:port2

x 就是myid中的数值，我建立三个实例，每个实例的myid内容分别为1 2 3

ip 就是服务器所在机器的ip地址

port1 是集群通讯的端口

port2 是集群投票的端口

若在同一个机器上，则要求六个端口不同，且不被占用。

## 复制zookeeper01中修改好的安装文件夹到另外两个文件夹

[root@localhost zookeeper01]# cp zookeeper-3.4.13/ /usr/local/zookeepercluster/zookeeper02 -r

[root@localhost zookeeper01]# cp zookeeper-3.4.13/ /usr/local/zookeepercluster/zookeeper03 -r

## 在逐一修改 myid zoo.conf 的内容

### zookeeper02 中

myid内容为 2

zoo.conf中

dataDir=/usr/local/zookeepercluster/zookeeper02/zookeeper-3.4.13/data

clientPort=2183

### zookeeper03 中

myid内容为3

zoo.conf中

dataDir=/usr/local/zookeepercluster/zookeeper03/zookeeper-3.4.13/data

clientPort=2184

## 创建批处理文件（非必须）

### 批启动文件

#### 在zookeepercluster 文件夹下创建文件：zookeeper\_start\_all.sh

[root@localhost zookeepercluster]# touch zookeeper\_start\_all.sh

#### 文件内容：

cd /usr/local/zookeepercluster/zookeeper01/zookeeper-3.4.13/bin/

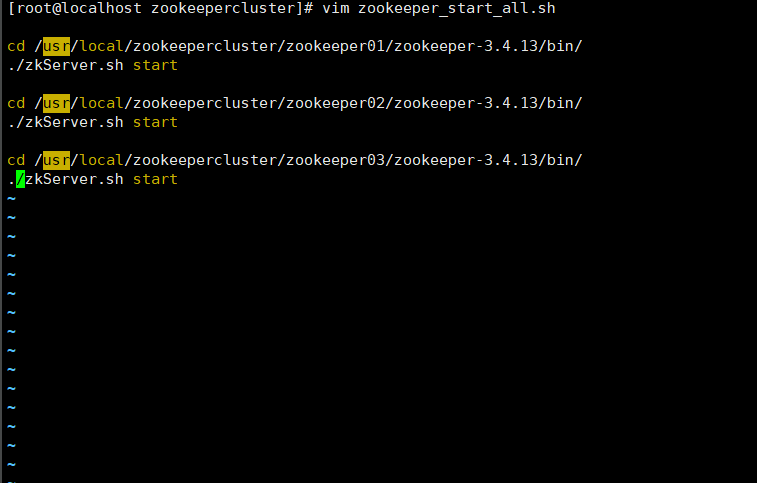
./zkServer.sh start

cd /usr/local/zookeepercluster/zookeeper02/zookeeper-3.4.13/bin/

./zkServer.sh start

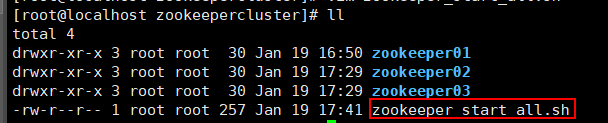
cd /usr/local/zookeepercluster/zookeeper03/zookeeper-3.4.13/bin/

./zkServer.sh start



#### 给改文件添加可执行权限

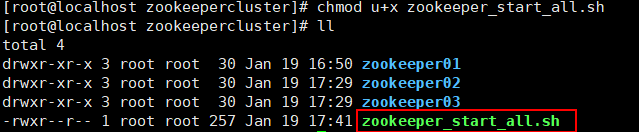
原文件灰色不可执行



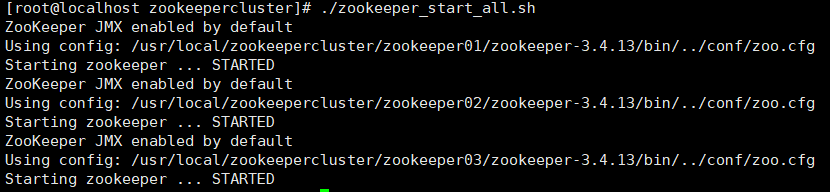
添加可执行权限

[root@localhost zookeepercluster]# chmod u+x zookeeper\_start\_all.sh （chmod 修改权限命令 u 当前用户 +添加 x 执行权限 后加文件名）

变成绿色可执行



#### 执行该命令：



#### 检查各个节点启动状态

[root@localhost zookeepercluster]# zookeeper01/zookeeper-3.4.13/bin/zkServer.sh status

ZooKeeper JMX enabled by default

Using config: /usr/local/zookeepercluster/zookeeper01/zookeeper-3.4.13/bin/../conf/zoo.cfg

Mode: follower (跟随者：从节点)

[root@localhost zookeepercluster]# zookeeper02/zookeeper-3.4.13/bin/zkServer.sh status

ZooKeeper JMX enabled by default

Using config: /usr/local/zookeepercluster/zookeeper02/zookeeper-3.4.13/bin/../conf/zoo.cfg

Mode: leader (主节点)

[root@localhost zookeepercluster]# zookeeper03/zookeeper-3.4.13/bin/zkServer.sh status

ZooKeeper JMX enabled by default

Using config: /usr/local/zookeepercluster/zookeeper03/zookeeper-3.4.13/bin/../conf/zoo.cfg

Mode: follower (跟随者：从节点)

### 批停止文件

与批处理启动文件相同：创建 zookeeper\_stop\_all.sh

内容：

cd /usr/local/zookeepercluster/zookeeper01/zookeeper-3.4.13/bin/

./zkServer.sh stop

cd /usr/local/zookeepercluster/zookeeper02/zookeeper-3.4.13/bin/

./zkServer.sh stop

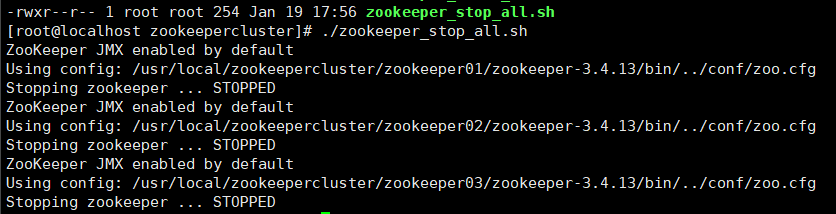
cd /usr/local/zookeepercluster/zookeeper03/zookeeper-3.4.13/bin/

./zkServer.sh stop

添加执行权限：

[root@localhost zookeepercluster]# chmod u+x zookeeper\_stop\_all.sh （chmod 修改权限命令 u 当前用户 +添加 x 执行权限 后加文件名）

执行：



产看

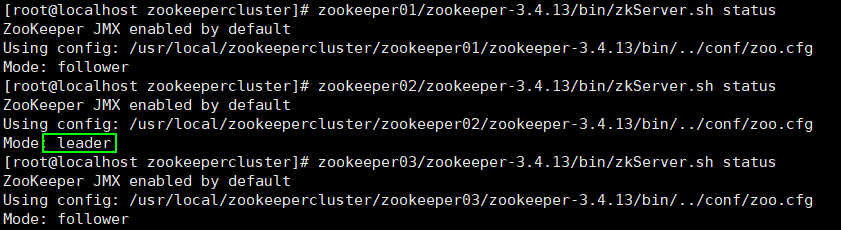


## 集群测试

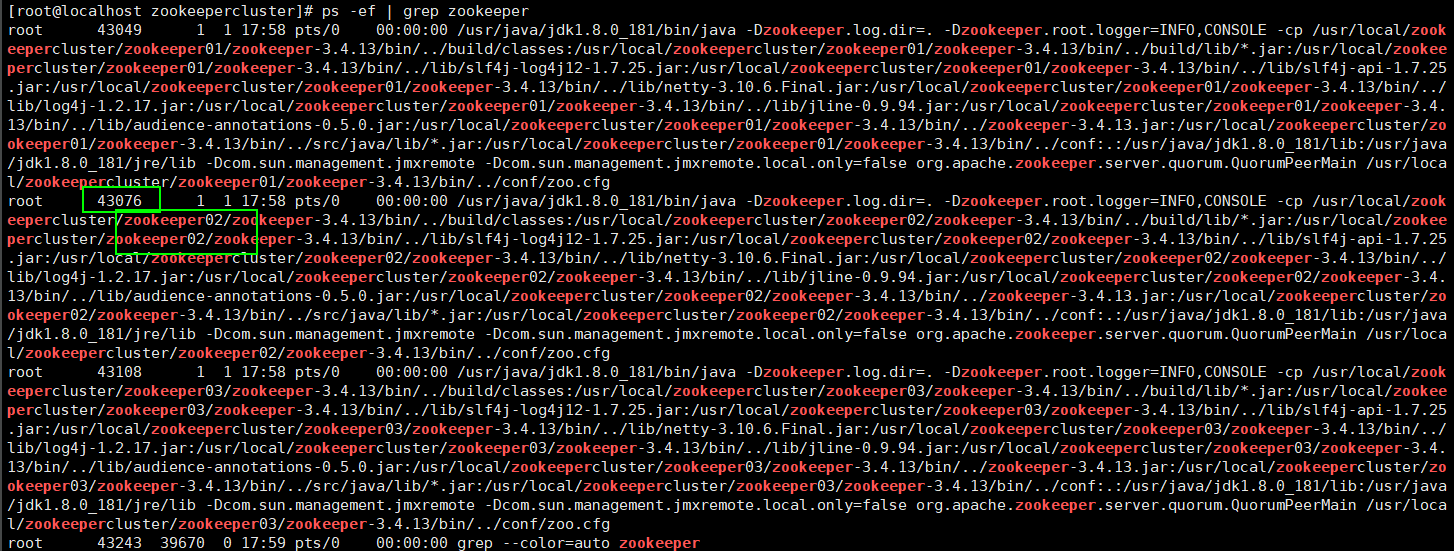
### 启动所有节点

[root@localhost zookeepercluster]# ./zookeeper\_start\_all.sh

### 查看个节点状态



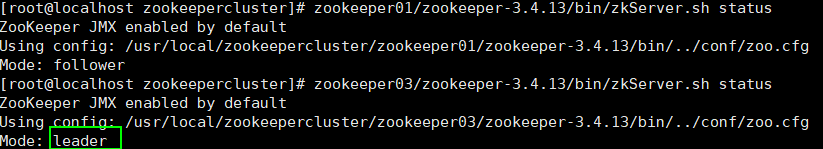
### 查看进程



### 杀死进程zookeeper02

[root@localhost zookeepercluster]# kill 9 43076

### 再次查看各存活节点状态



zookeeper03 被选举为leader

# 查看zookeeper上的配置文件：

使用zookeeper目录下的bin/zkCli.sh命令查看zookeeper上的配置文件：

注意点(如果不是默认的端口2181，要连接使用命令：

./zkCli.sh -server 192.168.153.136:2182

解释：

-server :指定ip地址

2182 ：指定连接的端口。

[zk: localhost:2182(CONNECTED) 4] ls /

[dubbo, zookeeper]

[zk: localhost:2182(CONNECTED) 5]

# zookeeper开启启动（未测试：单机版zookeeper的开启启动设置）

**编写脚本设置zookeeper开机启动**

在/etc/init.d/ 下创建zookeeper文件

[root@localhost zookeepercluster]# vi /etc/init.d/zookeeper

#!/bin/bash

#chkconfig:2345 20 90

#description:zookeeper

#processname:zookeeper

export JAVA\_HOME=/opt/module/jdk1.8

export ZOO\_LOG\_DIR=/opt/module/zookeeper-3.4.12/logs

case $1 in

start) su root /opt/module/zookeeper-3.4.12/bin/zkServer.sh start;; （集群可能是修改这个地方）

stop) su root /opt/module/zookeeper-3.4.12/bin/zkServer.sh stop;;

status) su root /opt/module/zookeeper-3.4.12/bin/zkServer.sh status;;

restart) su root /opt/module/zookeeper-3.4.12/bin/zkServer.sh restart;;

\*) echo "require start|stop|status|restart" ;;

esac

其中**chkconfig:2345 20 90**非常重要

2345:为主机运行的级别，表示主机运行在2、3、4、5个级别时都会启动zookeeper，而0 、1、6级别时停止zookeeper

20:开机启动的优先级，要比solr的50大

90:关机停止的顺序

一般设置时，先启动，则后停止，注意不要把启动值设置得太小，否则可能一些系统核心服务还没有启动起来，导致你的应用无法启动

JAVA\_HOME是必须的

ZOO\_LOG\_DIR是可选的，用于保存zookeeper启动时的日志文件，我把它指定到了自定义目录，否则你得使用root用户启动zookeeper或者给root用户根目录的写权限授予给zookeeper的启动用户

赋权限给脚本，启动zookeeper，设置为开机启动

[root@node21 software]# chmod +x /etc/init.d/zookeeper

[root@node21 software]# service zookeeper start

[root@node21 software]# chkconfig --add zookeeper

然后去/etc/rc3.d中查看zookeeper的启动顺序，查看zookeeper的停止顺序

[root@node21 software]# ls /etc/rc3.d/\*zoo\*

/etc/rc3.d/S20zookeeper

[root@node21 software]# ls /etc/rc0.d/\*zoo\*

/etc/rc0.d/K90zookeeper

# 外部连接

如果连接不上，别忘了设置防火墙，开放响应的端口

# java代码测试

依赖：

<dependency>

<groupId>org.apache.zookeeper</groupId>

<artifactId>zookeeper</artifactId>

<version>3.4.13</version>

</dependency>

<dependency>

<groupId>com.101tec</groupId>

<artifactId>zkclient</artifactId>

<version>0.11</version>

</dependency>

package com.taotao.search;

import java.io.IOException;

import java.util.List;

import org.apache.zookeeper.ZooKeeper;

public class ZooUtils{

static ZooKeeper zk;

//初始化链接客户端

static{

try {

zk= new ZooKeeper("192.168.153.136:2182", 10000, null);

} catch (IOException e) {

e.printStackTrace();

}

}

//删除zookeeper中的文件夹

public static void delPath(String path) throws Exception{

//获取文件夹的子类

List<String> paths=zk.getChildren(path, false);

//判断如果该文件夹为空，则直接删除

if(paths.size() == 0){

zk.delete(path, -1);

System.out.println("删除成功: "+path);

}else{

//如果不为空，则递归处理

for (String p:paths){

delPath(path+"/"+p);

}

//如果不为空，则在删除子类后再删除该文件夹

zk.delete(path, -1);

System.out.println("删除成功: "+path);

}

}

public static void main(String[] args) throws Exception {

//call function

delPath("/configs/collection1.AUTOCREATED");

System.out.println("删除完毕。。。");

}

}