**[Zookeeper 安装和配置](http://coolxing.iteye.com/blog/1871009)**

**博客分类：**

* [zookeeper](http://coolxing.iteye.com/category/278800)

[zookeeper](http://www.iteye.com/blogs/tag/zookeeper)[java](http://www.iteye.com/blogs/tag/java)[分布式](http://www.iteye.com/blogs/tag/%E5%88%86%E5%B8%83%E5%BC%8F)

[转载请注明作者和[原文链接](http://coolxing.iteye.com/blog/1871009),  如有谬误, 欢迎在评论中指正. ]

Zookeeper的安装和配置十分简单, 既可以配置成单机模式, 也可以配置成集群模式. 下面将分别进行介绍.

**单机模式**

[点击这里](http://zookeeper.apache.org/releases.html)下载zookeeper的安装包之后, 解压到合适目录. 进入zookeeper目录下的conf子目录, 创建zoo.cfg:

**Bash代码  [收藏代码](javascript:void())**

1. tickTime=2000
2. dataDir=/Users/apple/zookeeper/data
3. dataLogDir=/Users/apple/zookeeper/logs
4. clientPort=4180

参数说明:

* tickTime: zookeeper中使用的基本时间单位, 毫秒值.
* dataDir: 数据目录. 可以是任意目录.
* dataLogDir: log目录, 同样可以是任意目录. 如果没有设置该参数, 将使用和dataDir相同的设置.
* clientPort: 监听client连接的端口号.

至此, zookeeper的单机模式已经配置好了. 启动server只需运行脚本:

**Bash代码  [收藏代码](javascript:void())**

1. bin/zkServer.sh start

 Server启动之后, 就可以启动client连接server了, 执行脚本:

**Bash代码  [收藏代码](javascript:void())**

1. bin/zkCli.sh -server localhost:4180

**伪集群模式**

所谓伪集群, 是指在单台机器中启动多个zookeeper进程, 并组成一个集群. 以启动3个zookeeper进程为例.

将zookeeper的目录拷贝2份:

**Bash代码  [收藏代码](javascript:void())**

1. |--zookeeper0
2. |--zookeeper1
3. |--zookeeper2

 更改zookeeper0/conf/zoo.cfg文件为:

**Bash代码  [收藏代码](javascript:void())**

1. tickTime=2000
2. initLimit=5
3. syncLimit=2
4. dataDir=/Users/apple/zookeeper0/data
5. dataLogDir=/Users/apple/zookeeper0/logs
6. clientPort=4180
7. server.0=127.0.0.1:8880:7770
8. server.1=127.0.0.1:8881:7771
9. server.2=127.0.0.1:8882:7772

新增了几个参数, 其含义如下:

* initLimit: zookeeper集群中的包含多台server, 其中一台为leader, 集群中其余的server为follower. initLimit参数配置初始化连接时, follower和leader之间的最长心跳时间. 此时该参数设置为5, 说明时间限制为5倍tickTime, 即5\*2000=10000ms=10s.
* syncLimit: 该参数配置leader和follower之间发送消息, 请求和应答的最大时间长度. 此时该参数设置为2, 说明时间限制为2倍tickTime, 即4000ms.
* server.X=A:B:C 其中X是一个数字, 表示这是第几号server. A是该server所在的IP地址. B配置该server和集群中的leader交换消息所使用的端口. C配置选举leader时所使用的端口. 由于配置的是伪集群模式, 所以各个server的B, C参数必须不同.

参照zookeeper0/conf/zoo.cfg, 配置zookeeper1/conf/zoo.cfg, 和zookeeper2/conf/zoo.cfg文件. 只需更改dataDir, dataLogDir, clientPort参数即可.

在之前设置的dataDir中新建myid文件, 写入一个数字, 该数字表示这是第几号server. 该数字必须和zoo.cfg文件中的server.X中的X一一对应.  
/Users/apple/zookeeper0/data/myid文件中写入0, /Users/apple/zookeeper1/data/myid文件中写入1, /Users/apple/zookeeper2/data/myid文件中写入2.

分别进入/Users/apple/zookeeper0/bin, /Users/apple/zookeeper1/bin, /Users/apple/zookeeper2/bin三个目录, 启动server.  
任意选择一个server目录, 启动客户端:

**Bash代码  [收藏代码](javascript:void())**

1. bin/zkCli.sh -server localhost:4180

**集群模式**

集群模式的配置和伪集群基本一致.  
由于集群模式下, 各server部署在不同的机器上, 因此各server的conf/zoo.cfg文件可以完全一样.  
下面是一个示例:

**Bash代码  [收藏代码](javascript:void())**

1. tickTime=2000
2. initLimit=5
3. syncLimit=2
4. dataDir=/home/zookeeper/data
5. dataLogDir=/home/zookeeper/logs
6. clientPort=4180
7. server.43=10.1.39.43:2888:3888
8. server.47=10.1.39.47:2888:3888
9. server.48=10.1.39.48:2888:3888

示例中部署了3台zookeeper server, 分别部署在10.1.39.43, 10.1.39.47, 10.1.39.48上. 需要注意的是, 各server的dataDir目录下的myid文件中的数字必须不同.

10.1.39.43 server的myid为43, 10.1.39.47 server的myid为47, 10.1.39.48 server的myid为48.

# 使用DAEMONTOOLS监控ZOOKEEPER服务

daemontools(http://cr.yp.to/daemontools.html)是一个管理UNIX下服务的工具集。supervise监视一个服务。它启动一个服务、当服务停掉时它会重起服务。

Zookeeper在运行过程中，如果出现一些无法处理的异常，会直接退出进程，也就是所谓的快速失败（fail fast）模式。daemontools 能够帮助你监控ZK进程，一旦进程退出后，能够自动重启进程，从而使down掉的机器能够重新加入到集群中去。

## 安装daemontools

mkdir /package

chmod 755 /package

cd /package

wget http://cr.yp.to/daemontools/daemontools-0.76.tar.gz

tar zxf daemontools-0.76.tar.gz

cd admin/daemontools-0.76

vim src/error.h 找到：extern int errno; 改成：#include <errno.h>

package/install

## 监控Zookeeper

在/service下，新建文件夹zookeeper，新建run文件：

cd /service

mkdir zookeeper

cd zookeeper

vim run

chmod 755 run

run内容：

#!/bin/bash

exec 2>&1

exec /zk/zookeeper-3.4.4/bin/zkServer.sh start

运行：

supervise /service/zookeeper

//或者可以用nohup以后台方式运行。如下：

nohup supervise /service/zookeeper &

如果出现

supervise: fatal: unable to acquire /service/zookeeper/supervise/lock: temporary failure

快速rm /service/zookeeper/supervise 重新执行 supervise /service/zookeeper 即可

验证监控zookeeper是否成功：

kill zookeeper进程，查看zookeeper的进程是否自动重启

ps -aux|grep Dzookeeper //查看zookeeper的进程

参考文献： http://cr.yp.to/daemontools/faq/create.html

最後一步，啟動Storm的所有後臺進程。和Zookeeper一樣，Storm也是快速失敗（fail-fast)的系統，這樣Storm才能在任意時刻被停止，並且當進程重啟後被正確地恢復執行。這也是為什麼Storm不在進程內保存狀態的原因，即使Nimbus或Supervisors被重啟，運行中的Topologies不會受到影響。

以下是啟動Storm各個後臺進程的方式：

1. **Nimbus**: 在Storm主控節點上運行**”bin/storm nimbus >/dev/null 2>&1 &**”啟動Nimbus幕後程式，並放到後臺執行；
2. **Supervisor**: 在Storm各個工作節點上運行”bin/storm supervisor >/dev/null 2>&1 &”啟動Supervisor幕後程式，並放到後臺執行；
3. **UI**: 在Storm主控節點上運行”bin/storm ui >/dev/null 2>&1 &”啟動UI背景程式，並放到後臺執行，啟動後可以通過http://{nimbus host}:8080觀察集群的worker資源使用情況、Topologies的運行狀態等資訊。

<http://www.michael-noll.com/tutorials/running-multi-node-storm-cluster/>

./storm jar ~/storm-starter-0.9.3.jar storm.starter.WordCountTopology wordCount

http://www.michael-noll.com/blog/2013/03/13/running-a-multi-broker-apache-kafka-cluster-on-a-single-node/

Kafka installation

1. wget http://mirror.sdunix.com/apache/kafka/0.8.1.1/kafka\_2.9.2-0.8.1.1.tgz - See more at: <http://vulab.com/blog/?p=576#sthash.v03xZf8D.dpuf>
2. tar -xzf kafka\_2.9.2-0.8.1.1.tgz

http://my.oschina.net/cloudcoder/blog/299215

http://www.cnblogs.com/fxjwind/p/3794255.html