# zookeeper安装

1 用 SecureCRT 或 XShell 等 Linux 客户端工具连接至 CentOS7 服务器；
2 进入到 /usr/local/tools 目录中:
cd /usr/local/
如没有此目录则创建：
mkdir -p tools
3 下载 zookeeper-3.4.8.tar.gz：
wget http://apache.fayea.com/zookeeper/zookeeper-3.4.8/zookeeper-3.4.8.tar.gz
4 返回到上一级目录并创建 zookeeper 文件夹：
cd ..
mkdir -p zookeeper
5 将 zookeeper-3.4.8 文件从 /usr/local/tools 文件夹中移动到 /usr/local/zookeeper 文件夹中：
mv ./tools/zookeeper-3.4.8 ./zookeeper/
6 进入到 zookeeper/zookeeper-3.4.8/conf 目录中：
cd zookeeper/zookeeper-3.4.8/conf
7 复制 zoo_sample.cfg 文件并将新文件命名为 zoo.cfg：
cp zoo_sample.cfg zoo.cfg
8 修改 zoo.cfg 文件：
vi zoo.cfg
进入到 zoo.cfg 文件中；
修改数据文件夹 dataDir=/tmp/zookeeper 为 dataDir=/usr/local/zookeeper/zookeeper-3.4.8/data
配置日志文件夹 dataLogDir=/usr/local/zookeeper/zookeeper-3.4.8/logs
保存并退出；

9 将 zookeeper 的根目录设置到系统环境变量 PATH 中：
sudo vi /etc/profile

在打开的 profile 文件末尾追加如下配置：

```
export ZOOKEEPER_HOME=/usr/local/zookeeper/zookeeper-3.4.8
export PATH=$ZOOKEEPER_HOME/bin:$PATH
export PATH
```

保存并退出 vi；

刷新 profile 文件使之立即生效：

```
source /etc/profile
```

10 执行如下命令启动 zookeeper 服务：

```
zkServer.sh start
```

如打印如下信息则表明启动 zookeeper 成功：

```
ZooKeeper JMX enabled by default
Using config: /usr/local/zookeeper/zookeeper-
3.4.8/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
```

由于我们将 zookeeper 配置到了系统环境变量 PATH 中，故此可以在任何地方执行 zookeeper 命令；

如果我们没有做第 9 步，那么我们需要先进入到 zookeeper 的命令工具文件夹中：

```
cd /usr/local/zookeeper/zookeeper-3.4.8/bin
```

然后在执行启动 zookeeper 服务的操作；

11 查看 zookeeper 当前的状态：

```
zkServer.sh status
```

出现如下信息：

```
Using config: /usr/local/zookeeper/zookeeper-
3.4.8/bin/../conf/zoo.cfg
Mode: standalone
```

其中 standalone 表示处于单机模式；

12 关闭 zookeeper 服务：

```
zkServer.sh stop
```

如打印如下信息，则表明关闭 zookeeper 成功：

Using config: /usr/local/zookeeper/zookeeper-3.4.8/bin/../conf/zoo.cfg
Stopping zookeeper ... STOPPED
13 重启 zookeeper 服务：
zkServer.sh restart
如打印如下信息，则表明重启 zookeeper 服务成功：
ZooKeeper JMX enabled by default
Using config: /usr/local/zookeeper/zookeeper-3.4.8/bin/../conf/zoo.cfg
ZooKeeper JMX enabled by default
Using config: /usr/local/zookeeper/zookeeper-3.4.8/bin/../conf/zoo.cfg
Stopping zookeeper ... STOPPED
ZooKeeper JMX enabled by default
Using config: /usr/local/zookeeper/zookeeper-3.4.8/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
14 通过客户端程序连接指定 IP 的 zookeeper 服务器，也可以连接任意 IP 的 zookeeper 服务器：
zkCli.sh -server localhost:2181
如何是本机也可以直接使用如下命令：
zkCli.sh
15 查看 zookeeper 的启动状态：
echo ruok | nc localhost 2181
16 查看 zookeeper 的进程：
jps
打印信息中的 QuorumPeerMain 之前的数字表示 zookeeper 的PID；
17 在 /usr/local/zookeeper/zookeeper-3.4.8/data/ 目录下创建 myid 文件，其内容为服务的编号；

18 zooKeeper 服务之间不存在刻意的主从关系（master/slave 关系），各个节点都是服务器，如果服务器 leader 挂了，会立马从从服务器 follower 中选举一个出来作为新的 leader 服务器。 各 zookeeper 服务器之间通过 TCP 协议交流信息。

附一：

zookeeper 下载列表：

http://apache.fayea.com/zookeeper/

可以选择

http://apache.fayea.com/zookeeper/zookeeper-3.4.8/

附二：

zoo.cfg 配置文件：

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
# 同步阶段一个放松和接受请求之间，不允许超过多少个tick的时间
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
# dataDir=/tmp/zookeeper
# 配置数据目录
dataDir=/usr/local/zookeeper/zookeeper-3.4.8/data
# 配置日志目录
dataLogDir=/usr/local/zookeeper/zookeeper-3.4.8/logs
# the port at which the clients will connect
# 客户端访问 zookeeper 的端口号：
```

clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
#
http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_m
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1

# 2888,3888 are election port
# server.1、2、3 表示 zookeeper 集群中的 1、2、3 号服务器；
# zookeeper 表示当前的 zookeeper 服务器的IP，由于 zookeeper 以在 hosts 文件中映射了本地的 IP，故此这里直接写 zookeeper 即可；
# 2888 表示 1、2、3 号 zookeeper 服务器与 zookeeper 集群中的 leader 服务器交流信息的端口；
# 3888 表示当 zookeeper 集群中的 leader 服务器挂掉之后，用来选举新的 leader 服务器时交流信息的端口；
server.1=192.168.1.100:2888:3888
server.2=192.168.1.101:2888:3888
server.3=192.168.1.102:2888:3888
附三：
zookeeper的配置文件说明：
clientPort # 客户端连接server的端口，即对外服务端口，一般设置为2181。

dataDir # 存储快照文件snapshot的目录。默认情况下，事务日志也会存储在这里。建议同时配置参数dataLogDir, 事务日志的写性能直接影响zk性能。
tickTime # ZK中的一个时间单元。ZK中所有时间都是以这个时间单元为基础，进行整数倍配置的。例如，session的最小超时时间是2*tickTime。
dataLogDir # 事务日志输出目录。尽量给事务日志的输出配置单独的磁盘或是挂载点，这将极大的提升ZK性能。
globalOutstandingLimit # 最大请求堆积数。默认是1000。ZK运行的时候，尽管server已经没有空闲来处理更多的客户端请求了，但是还是允许客户端将请求提交到服务器上来，提高吞吐性能。当然，为了防止Server内存溢出，这个请求堆积数还是需要限制下的。 Java system property:zookeeper.globalOutstandingLimit.
preAllocSize # 预先开辟磁盘空间，用于后续写入事务日志。默认是64M，每个事务日志大小就是64M。如果ZK的快照频率较大的话，建议适当减小这个参数。
snapCount # 每进行snapCount次事务日志输出后，触发一次快照(snapshot), 此时，ZK会生成一个snapshot.*文件，同时创建一个新的事务日志文件log.*。默认是100000.（真正的代码实现中，会进行一定的随机数处理，以避免所有服务器在同一时间进行快照而影响性能）。
traceFile # 用于记录所有请求的log，一般调试过程中可以使用，但是生产环境不建议使用，会严重影响性能
maxClientCnxns # 单个客户端与单台服务器之间的连接数的限制，是ip级别的，默认是60，如果设置为0，那么表明不作任何限制。请注意这个限制的使用范围，仅仅是单台客户端机器与单台ZK服务器之间的连接数限制，不是针对指定客户端IP，也不是ZK集群的连接数限制，也不是单台ZK对所有客户端的连接数限制。
clientPortAddress # 对于多网卡的机器，可以为每个IP指定不同的监听端口。默认情况是所有IP都监听 clientPort 指定的端口。
minSessionTimeoutmaxSessionTimeout # Session超时时间限制，如果客户端设置的超时时间不在这个范围，那么会被强制设置为最大或最小时间。默认的Session超时时间是在2 * tickTime ~ 20 * tickTime 这个范围 。

fsync.warningthresholdms # 事务日志输出时，如果调用fsync方法超过指定的超时时间，那么会在日志中输出警告信息。默认是1000ms。
autopurge.purgeInterval # 3.4.0及之后版本，ZK提供了自动清理事务日志和快照文件的功能，这个参数指定了清理频率，单位是小时，需要配置一个1或更大的整数，默认是0，表不开启自动清理功能
autopurge.snapRetainCount # 这个参数和上面的参数搭配使用，这个参数指定了需要保留的文件数目。默认是保留3个。
electionAlg #在之前的版本中， 这个参数配置是允许我们选择leader选举算法，但是由于在以后的版本中，只会留下一种“TCP-based version of fast leader election”算法，所以这个参数目前看来没有用了。
initLimit # Follower在启动过程中，会从Leader同步所有最新数据，然后确定自己能够对外服务的起始状态。Leader允许F在 initLimit 时间内完成这个工作。通常情况下，我们不用太在意这个参数的设置。如果ZK集群的数据量确实很大了，F在启动的时候，从Leader上同步数据的时间也会相应变长，因此在这种情况下，有必要适当调大这个参数了。
syncLimit # 在运行过程中，Leader负责与ZK集群中所有机器进行通信，例如通过一些心跳检测机制，来检测机器的存活状态。如果L发出心跳包在syncLimit之后，还没有从F那收到响应，那么就认为这个F已经不在线了。注意：不要把这个参数设置得过大，否则可能会掩盖一些问题。
leaderServes # 默认情况下，Leader是会接受客户端连接，并提供正常的读写服务。但是，如果你想让Leader专注于集群中机器的协调，那么可以将这个参数设置为no，这样一来，会大大提高写操作的性能。
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. 这里的x是一个数字，与myid文件中的id是一致的。右边可以配置两个端口，第一个端口用于F和L之间的数据同步和其它通信，第二个端口用于Leader选举过程中投票通信。
group.x=nnnnn[:nnnnn]weight.x=nnnnn # 对机器分组和权重设置，
cnxTimeout # Leader选举过程中，打开一次连接的超时时间，默认是5s

zookeeper.DigestAuthenticationProvider.superDigest # ZK权限设置相关
skipACL # 对所有客户端请求都不作ACL检查。如果之前节点上设置有权限限制，一旦服务器上打开这个开头，那么也将失效
forceSync # 这个参数确定了是否需要在事务日志提交的时候调用FileChannel .force来保证数据完全同步到磁盘
jute.maxbuffer # 每个节点最大数据量，是默认是1M。这个限制必须在server和client端都进行设置才会生效。