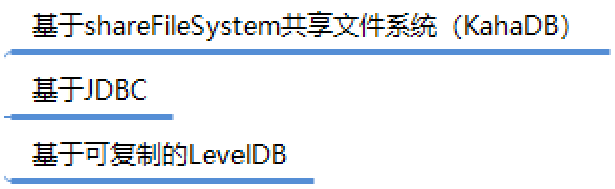
一、是什么

基于zookeeper和LevelDB搭建ActiveMQ集群。集群仅提供主备方式的高可用集群功能，避免单点故障。

二、zookeeper+replicated-leveldb-store的主从集群

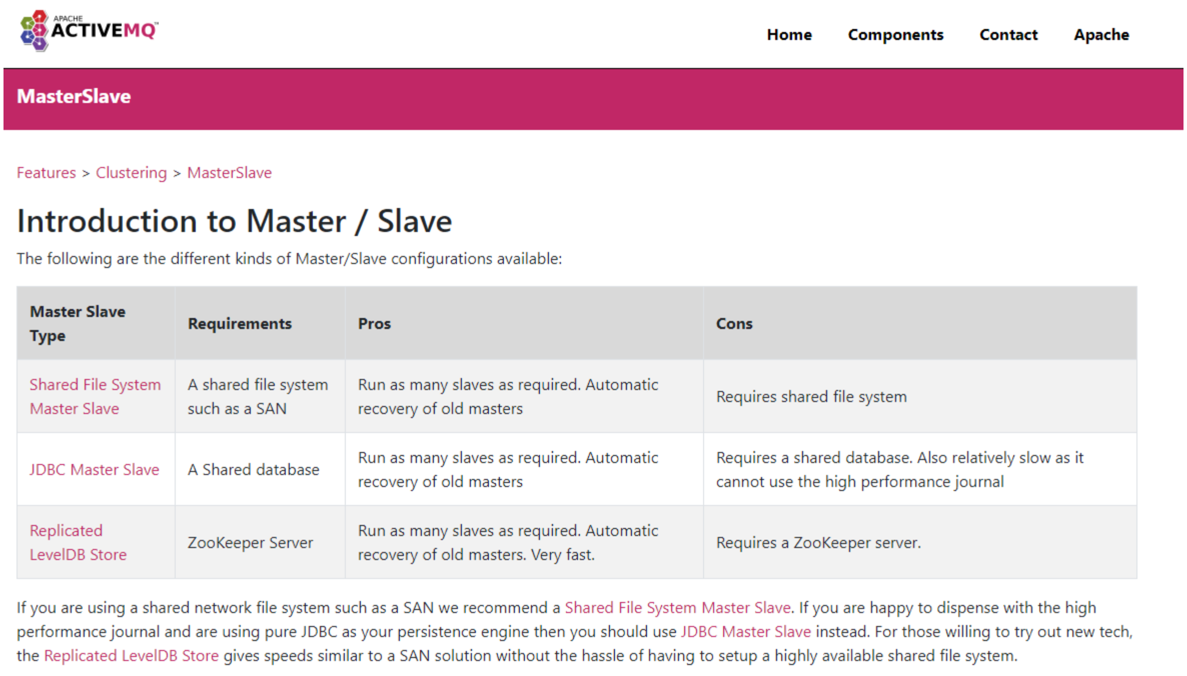
1、三种集群方式对比



基于shareFileSystem共享文件系统（KahaDB）

基于JDBC

基于可复制的LevelDB



Introduction to Master / Slave

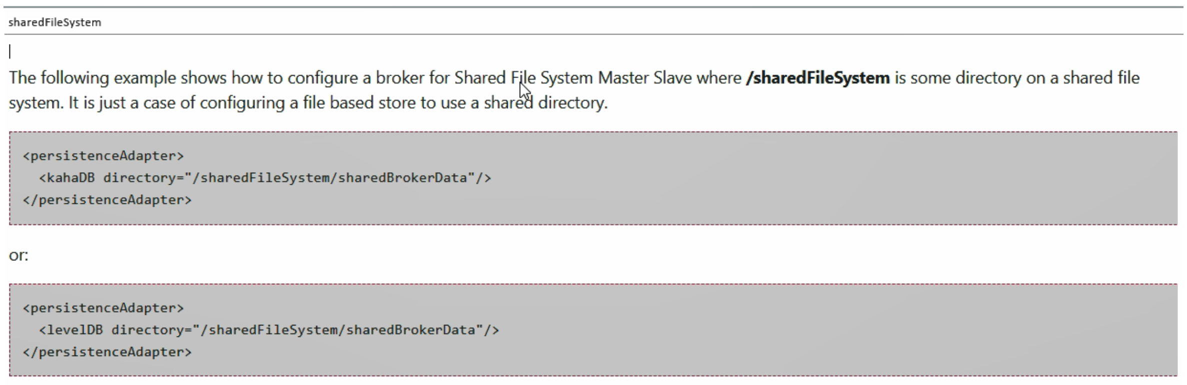
The following are the different kinds of Master/Slave configurations available:

Master Slave Type	Requirements	Pros	Cons
Shared File System Master Slave	A shared file system such as a SAN	Run as many slaves as required. Automatic recovery of old masters	Requires shared file system
JDBC Master Slave	A Shared database	Run as many slaves as required. Automatic recovery of old masters	Requires a shared database. Also relatively slow as it cannot use the high performance journal
Replicated LevelDB Store	ZooKeeper Server	Run as many slaves as required. Automatic recovery of old masters. Very fast.	Requires a ZooKeeper server.

If you are using a shared network file system such as a SAN we recommend a Shared File System Master Slave. If you are happy to dispense with the high performance journal and are using pure JDBC as your persistence engine then you should use JDBC Master Slave instead. For those willing to try out new tech, the Replicated LevelDB Store gives speeds similar to a SAN solution without the hassle of having to setup a highly available shared file system.

2、ZK+Replicated LevelDB Store

（1）ShareFileSystem



The following example shows how to configure a broker for Shared File System Master Slave where **/sharedFileSystem** is some directory on a shared file system. It is just a case of configuring a file based store to use a shared directory.

```
<persistenceAdapter>
  <kahaDB directory="/sharedFileSystem/sharedBrokerData"/>
</persistenceAdapter>
```

or:

```
<persistenceAdapter>
  <levelDB directory="/sharedFileSystem/sharedBrokerData"/>
</persistenceAdapter>
```

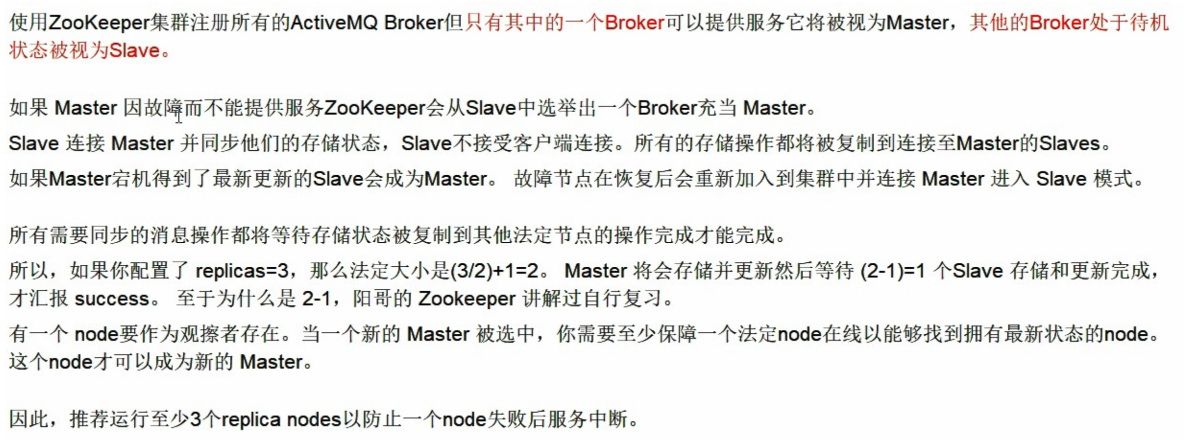
（2）是什么

官网：http://activemq.apache.org/replicated-leveldb-store

从ActiveMQ5.9开始，ActiveMQ的集群实现方式取消了传统的Masster-Slave方式。增加了基于Zookeeper+LevelDB的Master-Slave实现方式，从5.9版本后也是官网的推荐。

基于Zookeeper和LevelDB搭建ActiveMQ集群，集群仅提供主备方式的高可用集群功能，避免单点故障。

（3）官网集群原理

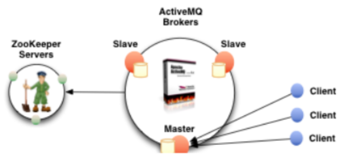


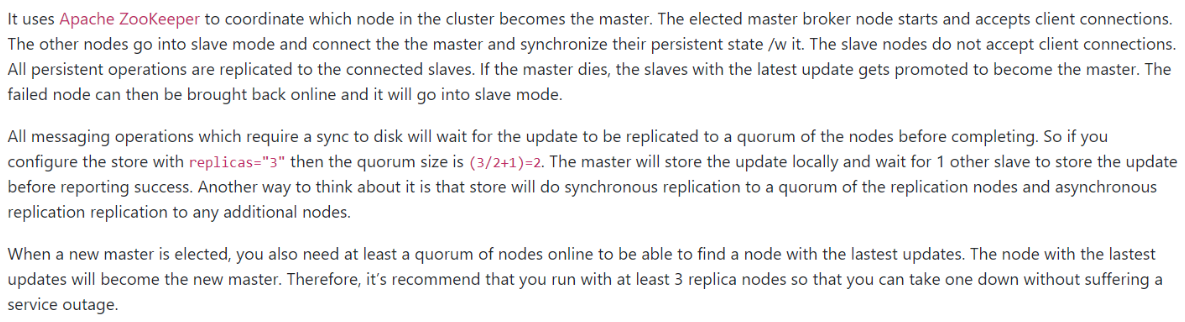
使用ZooKeeper集群注册所有的ActiveMQ Broker但只有其中的一个Broker可以提供服务它将被视为Master，其他的Broker处于待机状态被视为Slave。

如果 Master 因故障而不能提供服务ZooKeeper会从Slave中选举出一个Broker充当 Master。
Slave 连接 Master 并同步他们的存储状态，Slave不接受客户端连接。所有的存储操作都将被复制到连接至Master的Slaves。
如果Master宕机得到了最新更新的Slave会成为Master。 故障节点在恢复后会重新加入到集群中并连接 Master 进入 Slave 模式。

所有需要同步的消息操作都将等待存储状态被复制到其他法定节点的操作完成才能完成。
所以，如果你配置了 replicas=3，那么法定大小是(3/2)+1=2。 Master 将会存储并更新然后等待 (2-1)=1 个Slave 存储和更新完成，才汇报 success。 至于为什么是 2-1，阳哥的 Zookeeper 讲解过自行复习。
有一个 node要作为观擦者存在。当一个新的 Master 被选中，你需要至少保障一个法定node在线以能够找到拥有最新状态的node。这个node才可以成为新的 Master。

因此，推荐运行至少3个replica nodes以防止一个node失败后服务中断。





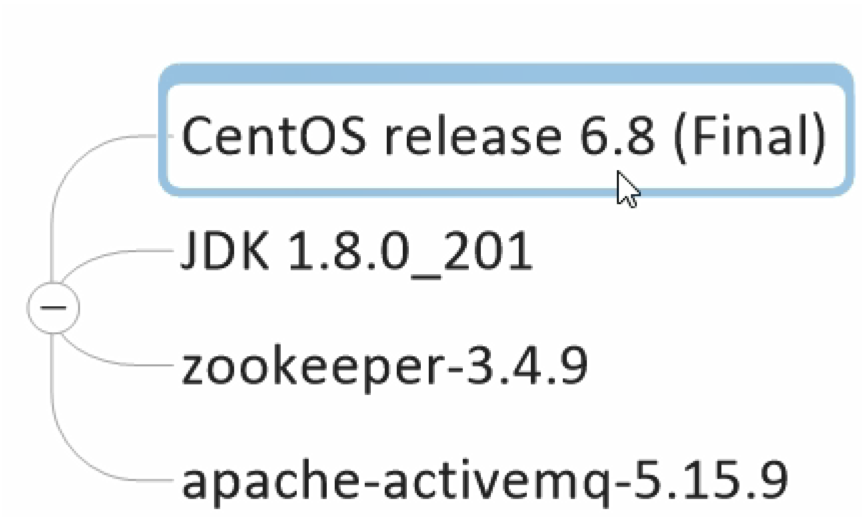
It uses Apache ZooKeeper to coordinate which node in the cluster becomes the master. The elected master broker node starts and accepts client connections. The other nodes go into slave mode and connect the the master and synchronize their persistent state /w it. The slave nodes do not accept client connections. All persistent operations are replicated to the connected slaves. If the master dies, the slaves with the latest update gets promoted to become the master. The failed node can then be brought back online and it will go into slave mode.

All messaging operations which require a sync to disk will wait for the update to be replicated to a quorum of the nodes before completing. So if you configure the store with `replicas="3"` then the quorum size is `(3/2+1)=2`. The master will store the update locally and wait for 1 other slave to store the update before reporting success. Another way to think about it is that store will do synchronous replication to a quorum of the replication nodes and asynchronous replication replication to any additional nodes.

When a new master is elected, you also need at least a quorum of nodes online to be able to find a node with the lastest updates. The node with the lastest updates will become the new master. Therefore, it's recommend that you run with at least 3 replica nodes so that you can take one down without suffering a service outage.

（4）部署规划和步骤

1.环境和版本



2.关闭防火墙并保证各个服务器能够ping通

3.要求具备zk集群并可以成功启动

配置集群教程:

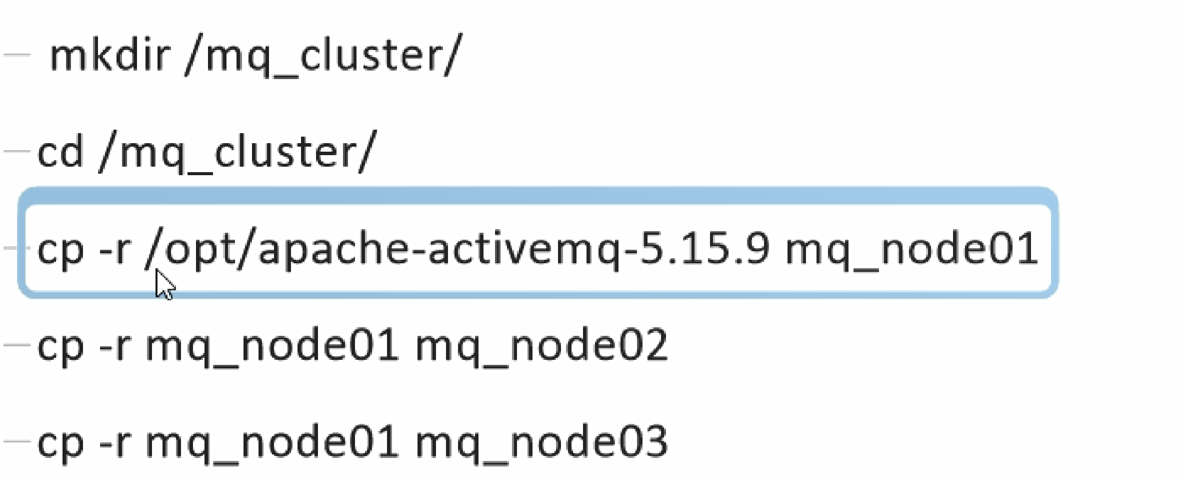
```
1,创建我们自己的文件夹存放Zookeeper
mkdir /my_zookeeper

2,下载Zookeeper
wget -P /my_zookeeper/
https://mirrors.tuna.tsinghua.edu.cn/apache/zookeeper/zookeeper-3.5.6/apache-
zookeeper-3.5.6-bin.tar.gz

3,解压
tar -zxvf apache-zookeeper-3.5.6-bin.tar.g

4,修改配置文件
文件名必须叫这个zoo.cfg
cp zoo_sample.cfg zoo.cfg
设置一下自定义的数据文件夹(注意要手动创建这个目录,后面会用到),,注意最后一定要有/结尾,没有/会报
错这是坑
dataDir=/my_zookeeper/apache-zookeeper-3.5.6-bin/data/
在zoo.cfg件最后面追加集群服务器
server.1=192.168.10.130:2888:3888
server.2=192.168.10.132:2888:3888
server.3=192.168.10.133:2888:3888

server.1=leantaot-zk-01:2888:3888
1是一个数字，标识这个是第几号服务器
leantaot-zk-01是这个服务器的IP地址（或者是与IP地址做了映射的主机名）
2888第一个端口用来集群成员的信息交换，标识这个服务器与集群中的leader服务器交换信息的端口
3888是在leader挂掉时专门用来进行选举leader所用的端口

6.把刚刚配置好的Zookeeper整个文件夹远程推送到其他服务器的/my_zookeeper文件夹内
scp -r /my_zookeeper/ root@192.168.10.132:/
scp -r /my_zookeeper/ root@192.168.10.133:/

7.创建myid文件，  id 与 zoo.cfg 中的序号对应
在192.168.10.130机器上执行
echo 1 > /my_zookeeper/apache-zookeeper-3.5.6-bin/data/zookeeper_server.pid
echo 1 > /my_zookeeper/apache-zookeeper-3.5.6-bin/data/myid
在192.168.10.132机器上执行
echo 2 > /my_zookeeper/apache-zookeeper-3.5.6-bin/data/zookeeper_server.pid
echo 2 > /my_zookeeper/apache-zookeeper-3.5.6-bin/data/myid
在192.168.10.133机器上执行
echo 3 > /my_zookeeper/apache-zookeeper-3.5.6-bin/data/zookeeper_server.pid
echo 3 > /my_zookeeper/apache-zookeeper-3.5.6-bin/data/myid

8.bin目录下启动Zookeeper做测试
 ./zkServer.sh start
输出
ZooKeeper JMX enabled by default
Using config: /my_zookeeper/apache-zookeeper-3.5.6-bin/bin/../conf/zoo.cfg
Starting zookeeper ... already running as process 1.
```

4.集群部署规划列表

主机	Zookeeper 集群端口	AMQ集群bind端口	AMQ消息tcp端口	管理控制台端口	AMQ节点安装目录
192.168.111.136	2191	bind="tcp://0.0.0.0:63631"	61616	8161	/mq_cluster/mq_node01
192.168.111.136	2192	bind="tcp://0.0.0.0:63632"	61617	8162	/mq_cluster/mq_node02
192.168.111.136	2193	bind="tcp://0.0.0.0:63633"	61618	8163	/mq_cluster/mq_node03

5.创建3台集群目录

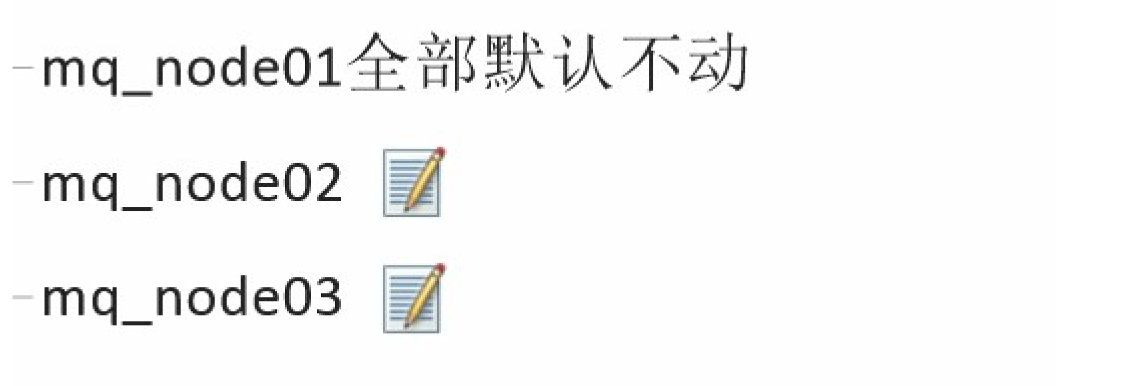
就是一台电脑复制三份ActiveMQ。



```
mkdir /mq_cluster/
cd /mq_cluster/
cp -r /opt/apache-activemq-5.15.9 mq_node01
cp -r mq_node01 mq_node02
cp -r mq_node01 mq_node03
```

6.修改管理控制台端口

就是ActiveMQ后台管理页面的访问端口。

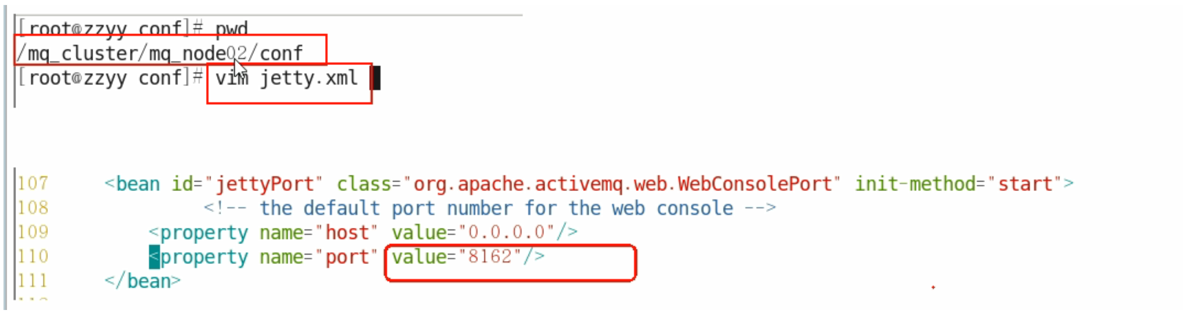


mq_node01全部默认不动

mq_node02

mq_node03

按照集群规划修改node02和node03：

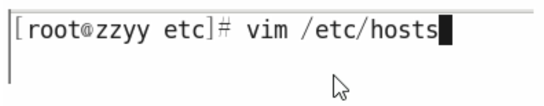


```
[root@zzyy conf]# pwd
/mq_cluster/mq_node02/conf
[root@zzyy conf]# vim jetty.xml
```

```
    <bean id="jettyPort" class="org.apache.activemq.web.WebConsolePort" init-method="start">
            <!-- the default port number for the web console -->
        <property name="host" value="0.0.0.0"/>
        <property name="port" value="8162"/>
    </bean>
```

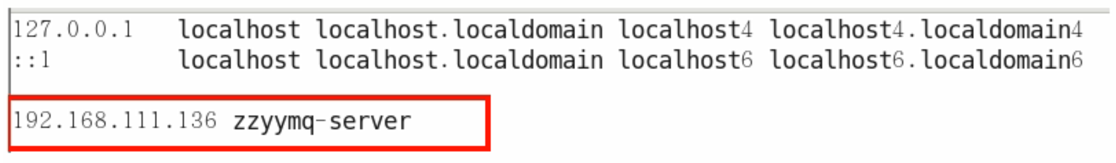
7.hostname名字映射

如果不映射只需要把mq配置文件的hostname改成当前主机ip。



```
[root@zzyy etc]# vim /etc/hosts
```

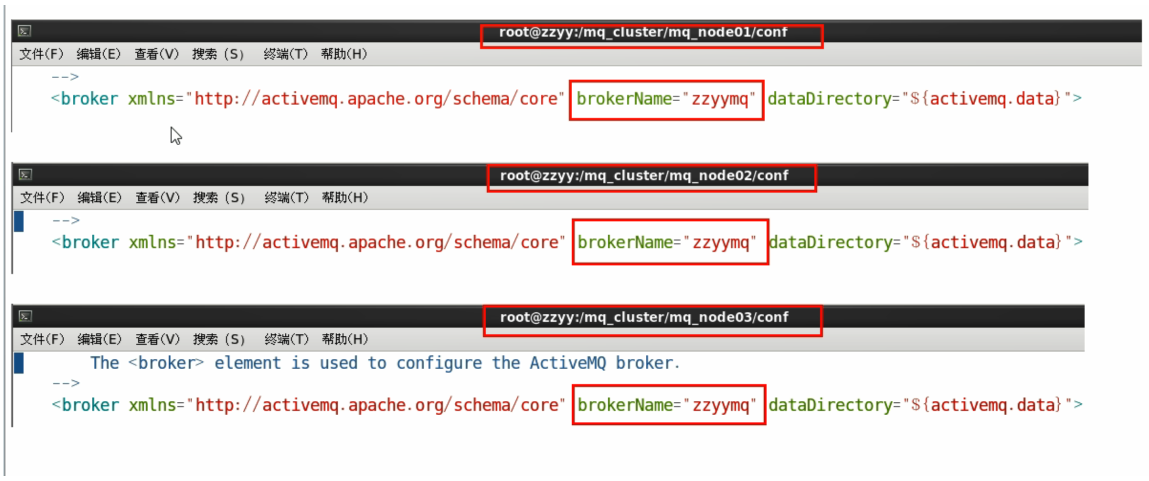
红框部分修改为自己的IP和地址映射



```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.111.136 zzyymq-server
```

8.ActiveMQ集群配置

配置文件里面的BrokerName要全部一致。

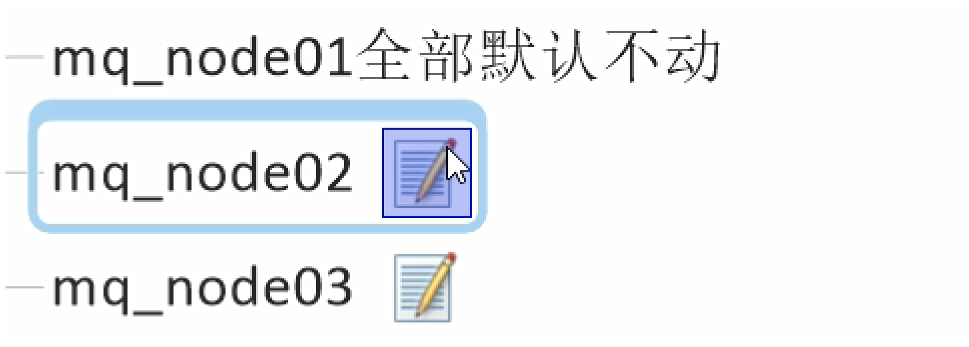


```
root@zzyy:/mq_cluster/mq_node01/conf
    -->
    <broker xmlns="http://activemq.apache.org/schema/core" brokerName="zzyymq" dataDirectory="${activemq.data}">

root@zzyy:/mq_cluster/mq_node02/conf
    -->
    <broker xmlns="http://activemq.apache.org/schema/core" brokerName="zzyymq" dataDirectory="${activemq.data}">

root@zzyy:/mq_cluster/mq_node03/conf
        The <broker> element is used to configure the ActiveMQ broker.
    -->
    <broker xmlns="http://activemq.apache.org/schema/core" brokerName="zzyymq" dataDirectory="${activemq.data}">
```

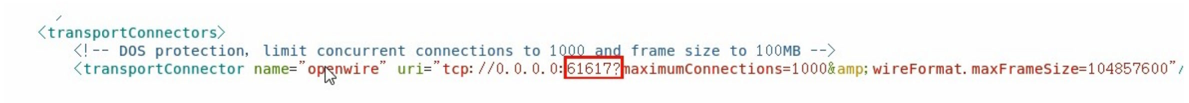
持久化配置(必须)，依次修改三个节点下的配置文件activemq.xml，注意bind的端口不一样：

```
<persistenceAdapter>
  <replicatedLevelDB
    directory="${activemq.data}/leveldb"
    replicas="3"
    bind="tcp://0.0.0.0:63633"
    zkAddress="localhost:2191,localhost:2192,localhost:2193"
    hostname="zzyymq-server"
    sync="local_disk"
    zkPath="/activemq/leveldb-stores"/>
</persistenceAdapter>
```

9.修改各个节点的消息端口



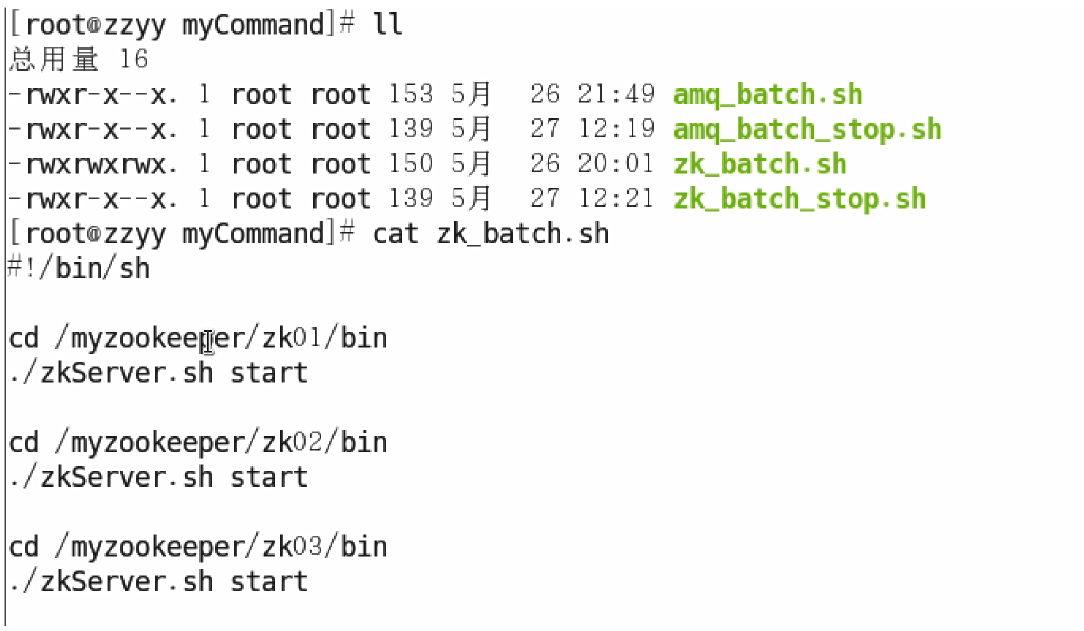
按照集群规划修改node02和node03：



```
<transportConnectors>
    <!-- DOS protection, limit concurrent connections to 1000 and frame size to 100MB -->
    <transportConnector name="openwire" uri="tcp://0.0.0.0:61617?maximumConnections=1000&amp;wireFormat.maxFrameSize=104857600"/
```

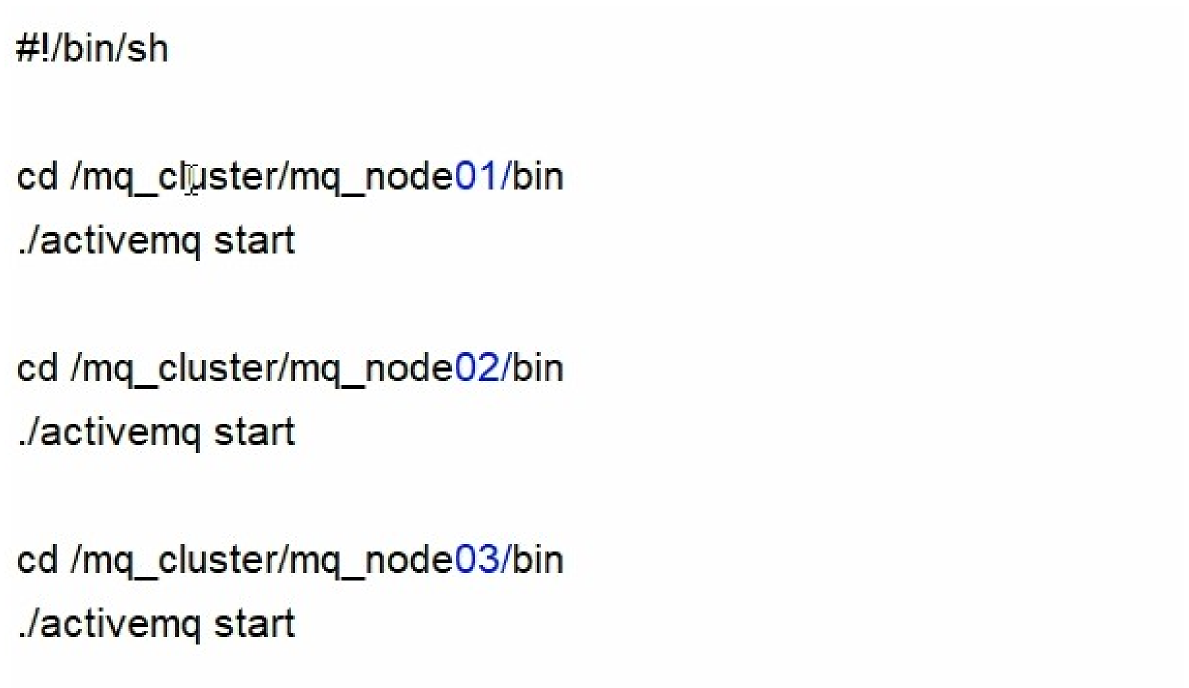
10.按顺序启动3个ActiveMQ节点

到这步前提是zk集群已经成功启动运行。**先启动Zk再启动ActiveMQ。**



```
[root@zzyy myCommand]# ll
总用量 16
-rwxr-x--x. 1 root root 153 5月  26 21:49 amq_batch.sh
-rwxr-x--x. 1 root root 139 5月  27 12:19 amq_batch_stop.sh
-rwxrwxrwx. 1 root root 150 5月  26 20:01 zk_batch.sh
-rwxr-x--x. 1 root root 139 5月  27 12:21 zk_batch_stop.sh
[root@zzyy myCommand]# cat zk_batch.sh
#!/bin/sh

cd /myzookeeper/zk01/bin
./zkServer.sh start

cd /myzookeeper/zk02/bin
./zkServer.sh start

cd /myzookeeper/zk03/bin
./zkServer.sh start
```

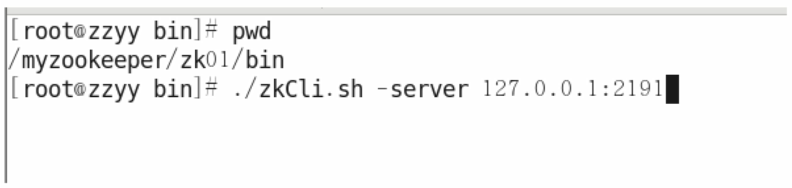
amq_batch.sh内容如下:



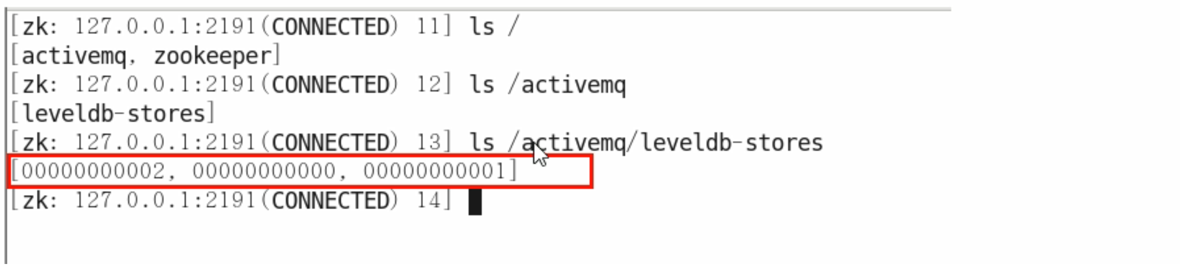
```
#!/bin/sh

cd /mq_cluster/mq_node01/bin
./activemq start

cd /mq_cluster/mq_node02/bin
./activemq start

cd /mq_cluster/mq_node03/bin
./activemq start
```

11.zk集群节点状态说明

3台Zk连接任意一台验证三台ActiveMQ是否注册上了Zookeeper。使用zkCli.sh连接一台Zookeeper：

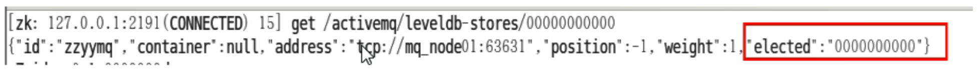


```
[root@zzyy bin]# pwd
/myzookeeper/zk01/bin
[root@zzyy bin]# ./zkCli.sh -server 127.0.0.1:2191
```

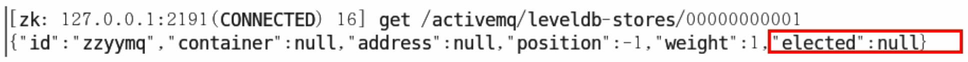


```
[zk: 127.0.0.1:2191(CONNECTED) 11] ls /
[activemq, zookeeper]
[zk: 127.0.0.1:2191(CONNECTED) 12] ls /activemq
[leveldb-stores]
[zk: 127.0.0.1:2191(CONNECTED) 13] ls /activemq/leveldb-stores
[00000000002, 00000000000, 00000000001]
[zk: 127.0.0.1:2191(CONNECTED) 14]
```

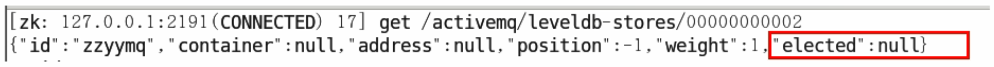
查看Master：



```
[zk: 127.0.0.1:2191(CONNECTED) 15] get /activemq/leveldb-stores/00000000000
{"id":"zzyymq","container":null,"address":"tcp://mq_node01:63631","position":-1,"weight":1,"elected":"0000000000"}
```



```
[zk: 127.0.0.1:2191(CONNECTED) 16] get /activemq/leveldb-stores/00000000001
{"id":"zzyymq","container":null,"address":null,"position":-1,"weight":1,"elected":null}
```



```
[zk: 127.0.0.1:2191(CONNECTED) 17] get /activemq/leveldb-stores/00000000002
{"id":"zzyymq","container":null,"address":null,"position":-1,"weight":1,"elected":null}
```

集群启动后对 ZooKeeper 数据的抓图，可以看到 ActiveMQ 的有 3 个节点，

分别是 00000000000，00000000001， 00000000002，

第一张图00000000000的值可以看到 elected 的值是不为空，说明这个节点是 Master，其他两个节点是 Slave。

（5）集群可用性测试

集群需要使用(failover:(tcp://192.168.10.130:61616,tcp://192.168.10.132:61616,tcp://192.168.10.133:61616))配置多个ActiveMQ

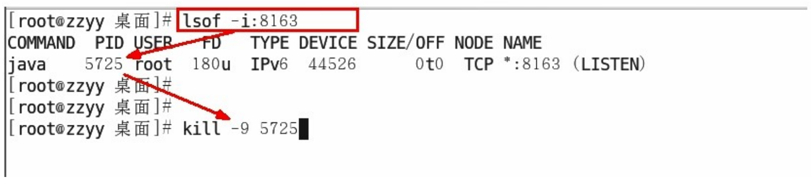
生产和消费者都修改

```
public static
final String ACTIVEMQ_URL = "failover:(tcp://192.168.111.136:61616,tcp://192.168.111.136:61617,tcp://192.168.111.136:61618)?randomize=f
alse";

public static final String QUEUE_NAME = "queue-cluster";
```

测试：3台机器中的ActiveMQ只会有一个MQ可以被客户端连接使用，在测试时可以把Master关掉，然后在重试客户端消息发送和消费还可以正常使用，则说明集群搭建正常。

假如现在：http://192.168.111.136:8163/

可以正常访问，说明这台是master



```
[root@zzyy 桌面]# lsof -i:8163
COMMAND  PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
java    5725 root  180u  IPv6  44526      0t0  TCP *:8163 (LISTEN)
[root@zzyy 桌面]#
[root@zzyy 桌面]#
[root@zzyy 桌面]# kill -9 5725
```