# ActiveMQ 集群搭建

集群搭建方案有两种：

##### Master-Slave部署方式

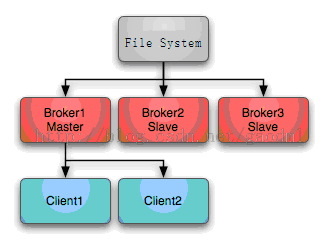
##### Broker-Cluster部署方式

方案1：Mastrer-Slave 主从方案

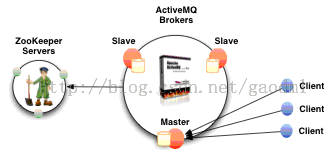
该方案提供三种方式

1.shared filesystem Master-Slave部署方式

主要是通过共享存储目录来实现master和slave的热备，所有的ActiveMQ应用都在不断地获取共享目录的控制权，哪个应用抢到了控制权，它就成为master。         多个共享存储目录的应用，谁先启动，谁就可以最早取得共享目录的控制权成为master，其他的应用就只能作为slave



2）shared database Master-Slave方式  
  
         与shared filesystem方式类似，只是共享的存储介质由文件系统改成了数据库而已。  
3）Replicated LevelDB Store方式  
  
         这种主备方式是ActiveMQ5.9以后才新增的特性，使用ZooKeeper协调选择一个node作为master。被选择的master broker node开启并接受客户端连接。  
  
其他node转入slave模式，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死了，得到了最新更新的slave被允许成为master。fialed node能够重新加入到网络中并连接master进入slave mode。所有需要同步的disk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，熟悉Zookeeper的应该知道，有一个node要作为观擦者存在。  
  
单一个新的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将会成为新的master。因此，推荐运行至少3个replica nodes，以防止一个node失败了，服务中断。



方案2：Broker-Cluster 方案

上面的方案只是主从的解决方法只能保证 可靠性 无法做负载均衡和分布式。下面提供负载均衡解决方案

Broker-Cluster部署方式中，各个broker通过网络互相连接，并共享queue。当broker-A上面指定的queue-A中接收到一个message处于pending状态，而此时没有consumer连接broker-A时。如果cluster中的broker-B上面由一个consumer在消费queue-A的消息，那么broker-B会先通过内部网络获取到broker-A上面的message，并通知自己的consumer来消费。

静态注册（知道broker的具体信息的时候才用）

主不需要做任何配置上的修改，只需要修改从上面conf的配置文件。

broker节点内面配置，主的信息

|  |  |
| --- | --- |
| 1  2  3  4  5 | <!-- static discovery config-->           <networkConnectors>               <networkConnector uri= "static:(<tcp://10.10.65.228:61617>)" duplex= "false" />           </networkConnectors>           <!-- static discovery config end--> |

动态注册

在activemq.xml文件中不直接指定Broker需要建立桥连接的其他Broker，由activemq在启动后动态查找：

1、  首先在Broker-A节点中添加networkConnector节点：

<networkConnectors>

                <networkConnectoruri="<multicast://default>"

           dynamicOnly="true"

           networkTTL="3"

           prefetchSize="1"

           decreaseNetworkConsumerPriority="true" />

</networkConnectors>

2、修改Broker-A节点中的服务提供端口为61616：

<transportConnectors>

         <transportConnectorname="openwire"uri="<tcp://0.0.0.0:61616>? " discoveryUri="multicast://default"/>

</transportConnectors>

3、在Broker-B节点中添加networkConnector节点：

<networkConnectors>

                <networkConnectoruri="<multicast://default>"

           dynamicOnly="true"

           networkTTL="3"

           prefetchSize="1"

           decreaseNetworkConsumerPriority="true" />

</networkConnectors>

4、修改Broker-B节点中的服务提供端口为61617：

<transportConnectors>

         <transportConnectorname="openwire"uri="<tcp://0.0.0.0:61617>" discoveryUri="multicast://default"/>

</transportConnectors>

5、启动Broker-A和Broker-B

在networkConnectors节点中配置multicast协议，用于寻找其他支持multicast的服务器。在transportConnector节点配置discoveryUri，声明服务器本身支持multicast协议。

广播自动发现服务器，适合于经常动态增减服务器的情况。优点是增减服务器，不需要为每个其他服务器节点修改配置。缺点是服务自动发现，缺少配置文件，对调试有影响。另外需要注意的是，由于广播功能，经常产生大量的消息传输，所以很多情况下运维是关掉这个服务的。使用Multicast Connector前要确保这个服务是开启的。

**客户端的Discover Protocol：**和Failover Protocol差不多，只不过是动态发现服务。配置如下：

|  |  |
| --- | --- |
| 1 | discovery:(multicast: //default) |

这样客户端会自动连接广播的url在multicast://default的服务器。