# ActiveMQ分享8--Broker Cluster

# 1 网状集群介绍

## 1.1 特征

    一群ActiveMQ Broker可以组成Broker 网络，即Broker Cluster，称之为连接网络集群、节点集群或者网状集群。网状集群有两种，分别是Static Discovery集群 和 Dynamic Discovery集群。本文后续也会讲到两种集群的配置方式。

网状集群具有以下特征：

 客户端连接的Broker节点失效，可以自动连接到其它Broker节点上。

 Broker可以发现集群中的其它Broker节点。

 消息可以在多个Broker之间路由。

 某个Broker上的消费者挂掉，尚未消费的消息会被路由到其它Broker（有存活的消费者）。

 Broker支持分布式的队列和主题。

这些特征，分别涉及了failover、 Discovery of Brokers和 Networks of Brokers。

## 1.2 failover

Client连接上若干的Broker中的一个，如果被连接的Broker失效，那么Client会通过failover机制自动连接到其它的Broker。

前面的学习笔记中也曾提到，Client使用failover机制，只需要Client编程时设置failover URL即可，示例如下。

|  |  |
| --- | --- |
| 1  2 | String url =  "failover:([tcp://localhost:61616,tcp://localhost:61626](tcp://localhost:61616,tcp:))?randomize=false" ;  ConnectionFactory contectionFactory =  new  ActiveMQConnectionFactory(user, password, url); |

更多关于ActiveMQ的failover机制，参见

<http://activemq.apache.org/failover-transport-reference.html>

## 1.3 Discovery of brokers

     Discovery of Brokers分为Static Discovery和Dynamic Discovery两种，其目的是让一个Broker节点知道网络中还有哪些Broker节点。

    Static Discovery，静态发现，就是采用一个写死的URL列表发现其它可以连接的Broker节点。采用这种发现机制的连接，将会尝试所有的URL，直到成功。（这是官网上鬼话的翻译<http://activemq.apache.org/static-transport-reference.html> ）

    通俗地说，静态发现就把所有的Broker节点URL都写死在activemq.xml中，URL示例如下。

    static:([tcp://localhost:61616,tcp://remotehost:61617?trace=false](tcp://localhost:61616,tcp:))

    Dynamic Discovery，动态发现，在配置Broker实例时，不需要知道所有其它实例的URL地址，而是采用多播(multicast)的方式，动态发现其它节点。其URL示例如下。

    discoveryUri="<multicast://default>"

动态发现的更多知识参见<http://activemq.apache.org/discovery-transport-reference.html> 。

## 1.4  Networks of Brokers

    如果网络中有多个Broker和多个Client，那么很有可能一个Broker节点只有producers（生产者），而没有consumers（消费者）。独立节点并不了解其它节点上的消费者，也就是说如果某个节点上没有消费者，那么这个节点上的消息可能会因得不到处理而积压起来。

    目前的解决方案是采用Networks of Brokers（节点网络）。它支持分布式的队列和主题。一个节点会相同地对待所有的订阅（subscription）：不管他们是来自本地的客户端连接，还是来自远程节点，它都会递送有关的消息拷贝到每个订阅。远程节点得到这个消息拷贝后，会依次把它递送到其内部的本地连接上。

# 2 Static Discovery集群

## 2.1 部署AMQ

假设D:\MQ\apache-activemq\nob\_static目录下有两个amq实例

D:\MQ\apache-activemq\nob\_static\amq1

D:\MQ\apache-activemq\nob\_static\amq2

## 2.2 配置activemq.xml

对于amq1

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | <!—修改brokerName-->  < broker  xmlns = "<http://activemq.apache.org/schema/core>"  brokerName = "static-broker1"                  dataDirectory = "${activemq.data}" >    < transportConnectors >     <!-通道只留下openwire，默认端口61616 -->     < transportConnector  name = "openwire"  uri = "<tcp://0.0.0.0:61616>" />  </ transportConnectors > |

对于amq2

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <!—修改brokerName-->  < broker  xmlns = "<http://activemq.apache.org/schema/core>"  brokerName = "static-broker2"                  dataDirectory = "${activemq.data}" >    <!—配置网络连接，采用静态发现。duplex=true，意味双向连接，amq1无须配置-->  < networkConnectors >     < networkConnector  uri = "static:(<tcp://localhost:61616>)"  duplex = "true" />  </ networkConnectors >    < transportConnectors >     <!-通道只留下openwire，端口61618 -->     < transportConnector  name = "openwire"  uri = "<tcp://0.0.0.0:61618>" />  </ transportConnectors > |

同一台机器上部署两个节点，为了防止端口冲突，把jetty.xml中的8161端口改成8162。

## 2.3 启动

先启动amq1，再启动amq2，观察控制台。

amq2输出信息

|  |  |
| --- | --- |
| 1  2 | Establishing network connection from vm: //static-broker2 ?async= false &network= true  to tcp: //localhost :61616 |

amq1输出信息

|  |  |
| --- | --- |
| 1  2 | Network connection between vm: //static-broker1 #4 and  tcp: ///127 .0.0.1:51578@61616 (static-broker2) has been established. |

# 3 Dynamic Discovery集群

## 3.1 部署AMQ

假设D:\MQ\apache-activemq\nob\_dynamic目录下有两个amq实例

D:\MQ\apache-activemq\nob\_dynamic\amq1

D:\MQ\apache-activemq\nob\_dynamic\amq2

## 3.2 配置activemq.xml

对于amq1

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | <!—修改brokerName-->  < broker  xmlns = "<http://activemq.apache.org/schema/core>"  brokerName = "dynamic-broker1"                  dataDirectory = "${activemq.data}" >    <!—broker节点下增加配置网络连接，采用动态发现-->  < networkConnectors >     < networkConnector  uri = "<multicast://default>" />  </ networkConnectors >    < transportConnectors >     <!-通道只留下openwire，默认端口61616 ，增加 discoveryUri -->     < transportConnector  name = "openwire"  uri = "<tcp://0.0.0.0:61616>"                         discoveryUri = "<multicast://default>"  />  </ transportConnectors > |

对于amq2

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <!—修改brokerName-->  < broker  xmlns = "<http://activemq.apache.org/schema/core>"  brokerName = "dynamic-broker2"                  dataDirectory = "${activemq.data}" >  <!—broker节点下增加配置网络连接，采用动态发现-->  < networkConnectors >     < networkConnector  uri = "<multicast://default>" />  </ networkConnectors >    < transportConnectors >     <!-通道只留下openwire，端口61618，配置discoveryUri -->     < transportConnector  name = "openwire"  uri = "<tcp://0.0.0.0:61618>"                         discoveryUri = "<multicast://default>"  />  </ transportConnectors > |

    同一台机器上部署两个节点，为了防止端口冲突，把jetty.xml中的8161端口改成8162。

## 3.3 启动

    先启动amq1，再启动amq2，观察控制台。

    AMQ1输出信息

|  |  |
| --- | --- |
| 1  2 | Network connection between vm: //dynamic-broker1 #2 and  tcp: //guohj-pc/192 .168.56.1:61618@49858 (dynamic-broker2)has been established. |

    AMQ2输出信息

|  |  |
| --- | --- |
| 1  2 | Network connection between vm: //dynamic-broker2 #0 and  tcp: //guohj-pc/192 .168.56.1:61616@49899 (dynamic-broker1)has been established. |

# 4 程序验证

    写程序验证一下。Client的生产者向broker1（amq1）发送“hello, world!”，消费者连接broker2（amq2），接受到“hello, world!”消息。

    生产者连接URL（61616，amq1）和队列名

|  |  |
| --- | --- |
| 1  2 | String url =  "failover:tcp://localhost:61616" ;  String subject =  "TOOL.DEFAULT" ; //队列名要相同 |

    消费者连接URL（61618，amq2）和队列名

|  |  |
| --- | --- |
| 1  2 | String url =  "failover:tcp://localhost:61618" ;  String subject =  "TOOL.DEFAULT" ; //队列名要相同 |

# 5 Network Connector

## 5.1 属性介绍

|  |  |  |
| --- | --- | --- |
| 参数名 | 默认值 | 描述 |
| name | bridge | 网络连接器名称，如果相同的两个节点间有多个网络连接器， 则每个网络连接器要使用不同的名称。 |
| dynamicOnly | false | 仅动态转发，如果为true，只有连接到节点上的消费者处于活动时， 消息才转发。 |
| decreaseNetworkConsumerPriority | false | （在网络跃点时）是否降低分发到队列消费者的优先级，实际上是对消费 者优先级的控制。 |
| networkTTL | 1 | 在节点网络中，消息或者订阅者能够跨越的节点的数量。 |
| conduitSubscriptions | true | 是否开启导管订阅，在节点网络中，多个消费者订阅（同一节点上）同一 目标，是否被当作同一个消费者。 |
| excludedDestinations | 空 | 排除目标列表，在该列表中的目标，将不会进行消息转发。 |
| dynamicallyIncludedDestinations | 空 | 动态包含列表，在该列表中的目标，将可以进行消息转发，如果为空，则意味着所有目标都可以转发。 |
| staticallyIncludedDestinations | 空 | 静态包含列表，在该列表中的目标，将可以进行消息转发，即使目标上没 有消费者，消息也可以转发出去。 |
| duplex | false | 是否双工，如果为true，则一个网络连接在生产者和消费者间可以双向传 递消息。此参数对hub-spoke模式特别有用，尤其是hub受防火墙保护时。 |

## 5.2 管道订阅-conduitSubscriptions

      如果两个brokers，分别是brokerA和brokerB，它们之间用forwarding bridge连接。有三个Consumer，订阅同一个Queue，其中1个连接到brokerA，2个连接到brokerB。

      如果conduitSubscriptions=true，那么brokerA上的consumer会得到15条消息，另外15条消息会发送给brokerB。此时负载并不均衡，因为此时brokerA将brokerB上的两个consumers视为1个。

      如果conduitSubscriptions=false， 那每 个consumer上都会收到10条消息。

## 5.3 双工-duplex

        前文配置静态网络连接集群的时候，已经用到了duplex。NetworkConnector在需要转发消息时默认是单向连接的。当broker2中配置duplex=true时，就变成了双向连接；即相当于配置了broker2到broker1的网络连接，也配置了broker1到broker2的网络连接。

        两个broker间可以建立两个以上的双向网络连接，从而增加吞吐量，或者对主题\队列分区，只需要指定它们使用不同的name即可。

# 6 总结

## 6.1 示例配置

ActiveMQ的examples目录中有静态集群和动态集群的配置示例，可以直接启动两个Broke实例。在cmd窗口中进入目录D:\MQ\apache-activemq\apache-activemq-5.10.1\bin。

启动动态集群，执行以下命令。

|  |  |
| --- | --- |
| 1  2 | activemq-admin start xbean:.. /examples/conf/activemq-dynamic-network-broker1 .xml  activemq-admin start xbean:.. /examples/conf/activemq-dynamic-network-broker2 .xml |

启动静态集群，执行以下命令。

|  |  |
| --- | --- |
| 1  2 | activemq-admin start xbean:.. /examples/conf/activemq-static-network-broker1 .xml  activemq-admin start xbean:.. /examples/conf/activemq-static-network-broker2 .xml |

## 6.2 优缺点

优点：

支持了消息路由和负载均衡。

缺点：

存在单点故障，如果某个Broker失效，有可能造成数据丢失。

如果为了保证消息的可靠性和，同时又支持负载均衡，可以采用Broker Cluster + Master Slave 模式的组合。