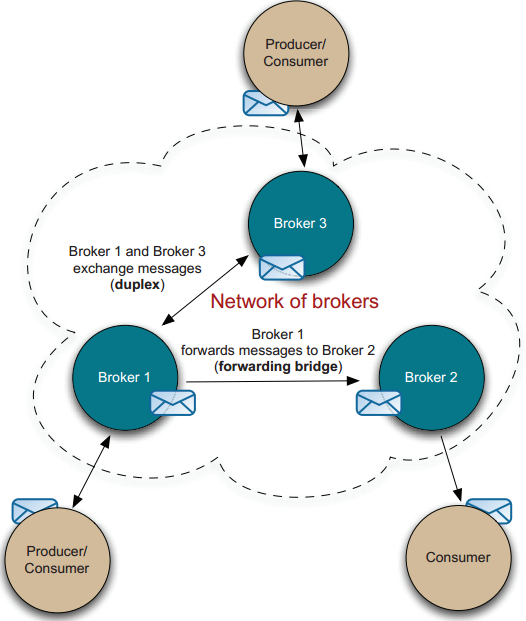
# ActiveMQ--2.多服务器间的通信

ActiveMQ支持多服务器(Broker)之间的网络连接，也就是集群。通过集群多个ActiveMQ Broker的实例，提供一个对外的统一服务，用来提高ActiveMQ的可用性和扩展性。

服务器之间的通信，按照通信方式可以分为两种， 桥接转发(Bridge Forwarding) 和双向传输(duplex)。顾名思义，桥接转发是将消息传递给另外一个ActiveMQ的Broker，双向传输是用一个通道既可以收消息，也可以发消息。



按照发现Broker的方式来区分，也可以分成另外两种，静态注册和动态发现。如果你确切的知道每一台Broker的地址和端口号，那么可以使用静态注册的方式；如果你并不知道每一台Broker的情况，比如，一个可动态扩展的生产环境，那么动态发现方式将非常合适。

每一种通信方式，同样是通过编辑%ACTIVEMQ\_HOME%conf\activemq.xml文件来完成配置。如下：

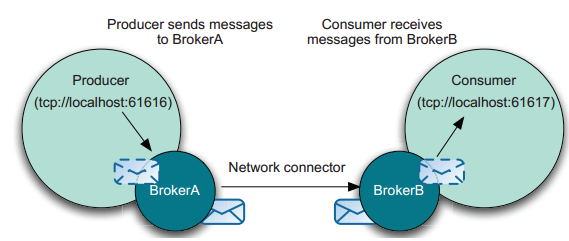
|  |  |
| --- | --- |
| 1  2  3 | < networkConnectors >       < networkConnector  name = "default-nc"  uri = "<multicast://default>" />  </ networkConnectors > |

在networkConnectors的节点下可以配置多个networkConnector，每个networkConnector是一种Broker-to-Broker的通信方式。下面具体介绍一下ActiveMQ支持的Broker-to-Broker的几种协议。

**静态注册：**

**服务器端的Static Connector：**被设置为Static Connector的两个Broker之间会转发消息，发给BrokerA的消息，会转发给BrokerB。但不是实时转发，而是在Consumer访问BrokerB的时候再转发。配置如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | < broker  xmlns = "<http://activemq.apache.org/schema/core>"  brokerName = "BrokerB"  dataDirectory = "${activemq.base}/data" >       < transportConnectors >           < transportConnector  name = "openwire"  uri = "<tcp://localhost:61616>"  />       </ transportConnectors >       < networkConnectors >           < networkConnector  uri = "static:(<tcp://remotehost:61616>)"  />       </ networkConnectors >  </ broker > |



Static Connector比较适合用在需要分散Consumer连接的场景。比如：有一个Broker有很多Consumer，而且每个Consumer在不同的地区。如果每个Consumer都连接同一个Broker，那么这个Broker将维护大量的连接，如果每个地区一个Broker，则每个地区的Consumer只需要连接本地的Broker。Broker之间通过Static自动转发消息，可以缓解单台Broker连接过多的问题。

**在实际测试的时候发现了《ActiveMQ in Action》书中有一点没写清楚。只有Queue才是Consumer消费的时候转发消息。持久化Topic是同步转发消息的，并且remotehost上面的消息都会被标记为已消费。**

**客户端的Failover Protocol：**顾名思义，这个协议是失效转移。客户端会默认使用随机算法选择事先配置好的几个Broker之一；也可以关闭随机选择。如果被选中的Broker发生状况，导致不能提供服务，客户端会选择另一个可用的Broker，如果某个Broker不可用，ActiveMQ会无限制的重试。(每次连接重试的延迟时间可配置)

默认配置，使用随机选择Broker算法：

|  |  |
| --- | --- |
| 1 | failover:(tcp: //localhost:61616，tcp://remotehost:61616) |

 关闭随机选择，优先选择第一个配置的Broker，只有之前的Broker不可用，再选择后面Broker：

|  |  |
| --- | --- |
| 1 | failover:(tcp: //localhost:61616,tcp://remotehost:61616)?randomize=false" |

由于自动重连机制，所以强烈推荐客户端使用faliover，即使是只有一个Broker，也可以通过自动重连功能，在Broker重新启动之后重新创建连接，而不用手动重启客户端。这样可以提升程序的健壮性。

 注意这个failover和static是不同的，static连接是用于服务器端自动转移消息，客户端并不知道服务器端是怎么转移消息的，也不知道到底服务器是几个，对客户端是完全透明的。failover是客户端失效转移，客户端连接服务器的的url需要明确的指出，所以客户端必须知道服务器端的网络拓扑结构。

**动态发现：**

**服务器端的Multicast Connector：**服务器之间通过广播功能自动发现其他服务器。每个服务器都会把自己的服务发布给224.0.0.0 - 239.255.255.255 IP地址段，也会从这个地址段找寻其他服务器。这样多个服务器之间可以主动的相互发现。配置如下：

[?](http://my.oschina.net/u/719192/blog/286454)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | < broker  xmlns = "<http://activemq.apache.org/schema/core>"  brokerName = "multicast"  dataDirectory = "${activemq.base}/data" >       < networkConnectors >           < networkConnector  name = "default-nc"  uri = "<multicast://default>" />       </ networkConnectors >       < transportConnectors >           < transportConnector  name = "openwire"  uri = "<tcp://localhost:61616>"  discoveryUri = "<multicast://default>" />       </ transportConnectors >  </ broker > |

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。

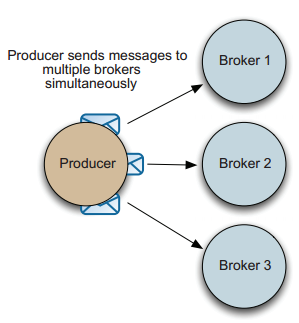
**客户端的Discover Protocol：**和Failover Protocol差不多，只不过是动态发现服务。配置如下：

|  |  |
| --- | --- |
| 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。

**客户端的Fanout Protocol：**扇出协议，用于一个客户端同时向多个服务器发同样的消息。

这样，发给remotehost的消息，会在Consumer从localhost中取消息的时候自动转发过去。



静态查找配置如下：

|  |  |
| --- | --- |
| 1 | fanout:( static :(tcp: //host1:61616,tcp://host2:61616,tcp://host3:61616)) |

当然也可以支持动态发现：

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

这个适用场景很少，毕竟客户端直接控制发冗余消息给服务器，实际上将变得更加不可控，不建议使用。

**客户端的Peer Protocol：**对等协议，用于前嵌入式服务器的连接，使用场景较少。