# RabbitMq集群

**內建集群的设计目标：**

客户端在节点崩溃的情况下可以运行，线性扩展来扩充消息的吞吐量

**可以保证消息的万无一失吗？**

当一个节点崩溃了以后，节点所有队列上的消息都会丢失。默认不会将队列的消息在集群中复制。

**集群中的队列**

在集群中不会复制，其他节点只会保存队列所处的节点和元数据，消息的传递给队列的所有者节点。

**集群中的交换器**

会进行复制。本质就是一个类似于hashmap的映射表。

**集群中的节点**

两种：内存节点，磁盘节点。单机情况下，一定是个磁盘节点。集群里面，要求每个集群必须有至少以一个磁盘节点，出于高可用考虑，建议配两个。

## 本机集群(不建议安装，有条件应在多个服务器上安装)：

RABBITMQ\_NODE\_PORT=5672 RABBITMQ\_NODENAME=rabbit rabbitmq-server -detached

RABBITMQ\_NODE\_PORT=5673 RABBITMQ\_NODENAME=rabbit\_1 rabbitmq-server -detached

RABBITMQ\_NODE\_PORT=5674 RABBITMQ\_NODENAME=rabbit\_2 rabbitmq-server -detached

rabbitmqctl -n rabbit\_1@centosvm stop\_app

rabbitmqctl -n rabbit\_1@centosvm reset

rabbitmqctl -n rabbit\_1@centosvm join\_cluster rabbit@centosvm

rabbitmqctl -n rabbit\_1@centosvm start\_app

rabbitmqctl cluster\_status

rabbitmqctl -n rabbit\_2@centosvm stop\_app

rabbitmqctl -n rabbit\_2@centosvm reset

rabbitmqctl -n rabbit\_2@centosvm join\_cluster rabbit@centosvm --ram

rabbitmqctl -n rabbit\_2@centosvm start\_app

rabbitmqctl cluster\_status

从外部要访问虚拟机中的mq记得在防火墙中打开端口

firewall-cmd --add-port=5673/tcp --permanent

firewall-cmd --add-port=5674/tcp --permanent

rabbitmqctl add\_user mq mq

rabbitmqctl set\_permissions mq ".\*" ".\*" ".\*"

rabbitmqctl set\_user\_tags mq administrator

rabbitmq-plugins -n rabbit\_1@centosvm enable rabbitmq\_management

## 多机下的集群

1. 修改 /etc/hosts

192.168.1.1 node1

192.168.1.2 node2

192.168.1.3 node3

2、Erlang Cookie 文件：/var/lib/rabbitmq/.erlang.cookie。将 node1 的该文件复制到 node2、node3，由于这个文件权限是 400，所以需要先修改 node2、node3 中的该文件权限为 777，然后将 node1 中的该文件拷贝到 node2、node3，最后将权限和所属用户/组修改回来。

3、运行各节点

4、在node2、node3上分别运行

[root@node2 ~]# rabbitmqctl stop\_app

[root@node2 ~]./rabbitmqctl reset

[root@node2 ~]# rabbitmqctl join\_cluster rabbit@node1

[root@node2 ~]# rabbitmqctl start\_app

内存节点则是rabbitmqctl join\_cluster rabbit@node1 –ram

## 移除集群中的节点

[root@node2 ~]# rabbitmqctl stop\_app

[root@node2 ~]./rabbitmqctl reset

[root@node2 ~]# rabbitmqctl start\_app

# 镜像队列

**什么是镜像队列**

如果RabbitMQ集群是由多个broker节点构成的，那么从服务的整体可用性上来讲，该集群对于单点失效是有弹性的，但是同时也需要注意：尽管exchange和binding能够在单点失效问题上幸免于难，但是queue和其上持有的message却不行，这是因为queue及其内容仅仅存储于单个节点之上，所以一个节点的失效表现为其对应的queue不可用。

引入RabbitMQ的镜像队列机制，将queue镜像到cluster中其他的节点之上。在该实现下，如果集群中的一个节点失效了，queue能自动地切换到镜像中的另一个节点以保证服务的可用性。在通常的用法中，针对每一个镜像队列都包含一个master和多个slave，分别对应于不同的节点。slave会准确地按照master执行命令的顺序进行命令执行，故slave与master上维护的状态应该是相同的。除了publish外所有动作都只会向master发送，然后由master将命令执行的结果广播给slave们，故看似从镜像队列中的消费操作实际上是在master上执行的。

RabbitMQ的镜像队列同时支持publisher confirm和事务两种机制。在事务机制中，只有当前事务在全部镜像queue中执行之后，客户端才会收到Tx.CommitOk的消息。同样的，在publisher confirm机制中，向publisher进行当前message确认的前提是该message被全部镜像所接受了。

**镜像队列的使用**

添加policy

Rabbitmqctl set\_policy Name Pattern Definition

Name:策略的名字

Pattern：队列匹配模式（正则表达式）

Definition：镜像的定义：ha-mode,ha-params,ha-sycn-mode

ha-mode: all/exactly/nodes

ha-params: n表示几个节点上复制/节点名称

ha-sycn-mode：automatic manual

对队列名称以“queue\_”队列进行镜像，只在两个节点上完成复制

Rabbitmqctl set\_policy ha\_queue\_two “^queue\_” ‘{“ha-mode”:”exactly”,”ha-params”:2,”ha-sycn-mode“：“atuomatic”}’

在代码中也要进行编写

# 使用HAProxy(以下步骤仅供参考)

作用：

安装配置：

1.下载最新haproxy安装包，官网：http://www.haproxy.org

2.上传到linux的haproxy用户根目录下，并解压：

tar -zxvf haproxy-1.5.8.tar.gz

创建目录/home/haproxy/haproxy

3.安装

cd haproxy-1.5.8

make TARGET=linux26 ARCH=x86\_64 PREFIX=/home/haproxy/haproxy #将haproxy安装到/home/haproxy/haproxy ,TARGET是指定内核版本

make install PREFIX=/home/haproxy/haproxy

进入/home/haproxy/haproxy 目录,创建/home/haproxy/haproxy/conf目录，复制配置examples

cp /home/haproxy/haproxy-1.5.8/examples/haproxy.cfg /home/haproxy/haproxy/conf/

4、配置修改(以haproxy rabbitmq 配置为关键字搜索)

# 互联网时代的消息中间件

**消息发送一致性**

Void busi{

//业务操作

//写库

//发送消息

}

业务成功，消息发送也要成功

业务失败，消息不应该发送

**消息的重复**

1. 让处理消息的服务具有幂等性

Update a set zz = 12; Update a set zz = zz+12(无幂等性);

1. db或者缓存保存消息的处理状况，消息id作为唯一性索引

**消息中间件与RPC的关系**

两者并不是水火不容的关系，两者可以很好的进行融合，结合起来使用。Rpc客户端调用rpc服务，或者rpc服务返回处理结果，就完全可以通过消息中间件进行。使用消息中间件做rpc有何好处：自动将消息路由到合适的地方，通过消息中间件可以在rpc服务集群中做到负载均衡，甚至当rpc服务中某台服务挂了，可以做到自动重发。

消息的数据量不能太大。