# **1 RabbitMQ集群与高可用**

RabbitMQ 的集群分两种模式，一种是默认集群模式，一种是镜像集群模式；

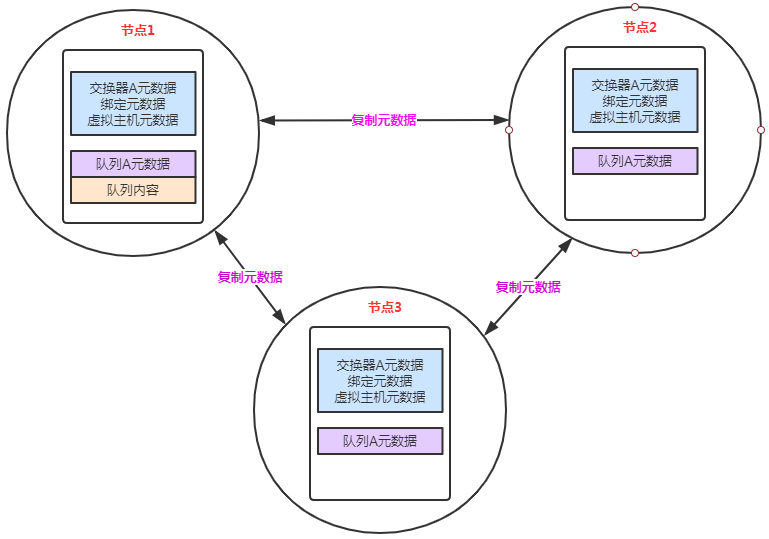
在RabbitMQ集群中所有的节点（一个节点就是一个RabbitMQ的broker服务器） 被归为两类：一类是磁盘节点，一类是内存节点；

磁盘节点会把集群的所有信息（比如交换机、绑定、队列等信息）持久化到磁盘中，而内存节点只会将这些信息保存到内存中，如果该节点宕机或重启，内存节点的数据会全部丢失，而磁盘节点的数据不会丢失；

## 1.1 默认集群模式

默认集群模式也叫 普通集群模式、或者 内置集群模式；

RabbitMQ默认集群模式，只会把交换机、队列、虚拟主机等元数据信息在各个节点同步，而具体队列中的消息内容不会在各个节点中同步；



**元数据**

队列元数据：队列名称和属性（是否可持久化，是否自动删除）

交换器元数据：交换器名称、类型和属性

绑定元数据：交换器和队列的绑定列表

vhost元数据：vhost内的相关属性，如安全属性等;

当用户访问其中任何一个RabbitMQ节点时，查询到的queue／user／exchange/vhost等信息都是相同的；

但集群中队列的具体信息数据只在队列的拥有者节点保存，其他节点只知道队列的元数据和指向该节点的指针，所以其他节点接收到不属于该节点队列的消息时会将该消息传递给该队列的拥有者节点上；

为什么集群不复制队列内容和状态到所有节点：

1. 存储空间；

2）性能；

如果消息需要复制到集群中每个节点，网络开销不可避免，持久化消息还需要写磁盘，占用磁盘空间。

**那么问题来了？**

如果有一个消息生产者或者消息消费者通过amqp-client的客户端连接到节点1进行消息的发送或接收，那么此时集群中的消息收发只与节点1相关，这个没有任何问题；

如果客户端相连的是节点2或者节点3 （队列1数据不在该节点上），那么情况又会是怎么样呢？

如果消息生产者所连接的是节点2或者节点3，此时队列1的完整数据不在该两个节点上，那么在发送消息过程中这两个节点主要起了一个路由转发作用，根据这两个节点上的元数据（也就是指向queue的owner node的指针）转发至节点1上，最终发送的消息还是会存储至节点1的队列1上；

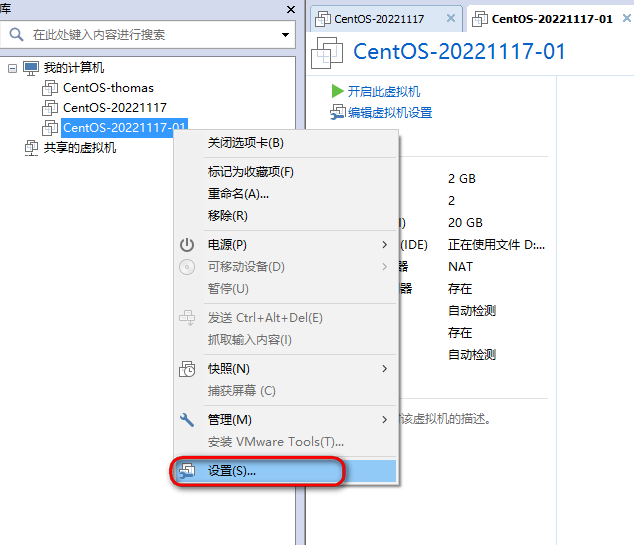
同样，如果消息消费者所连接的节点2或者节点3，那这两个节点也会作为路由节点起到转发作用，将会从节点1的队列1中获取消息进行消费；

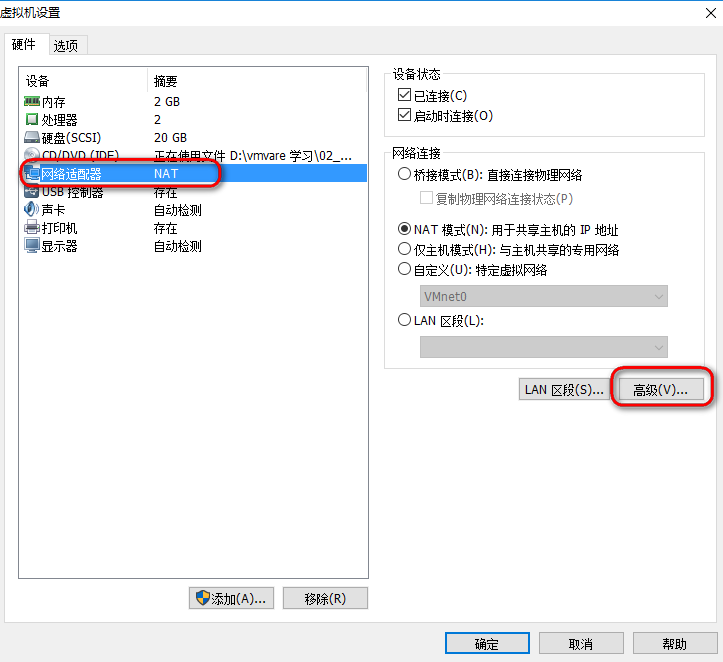
# 2 安装前准备

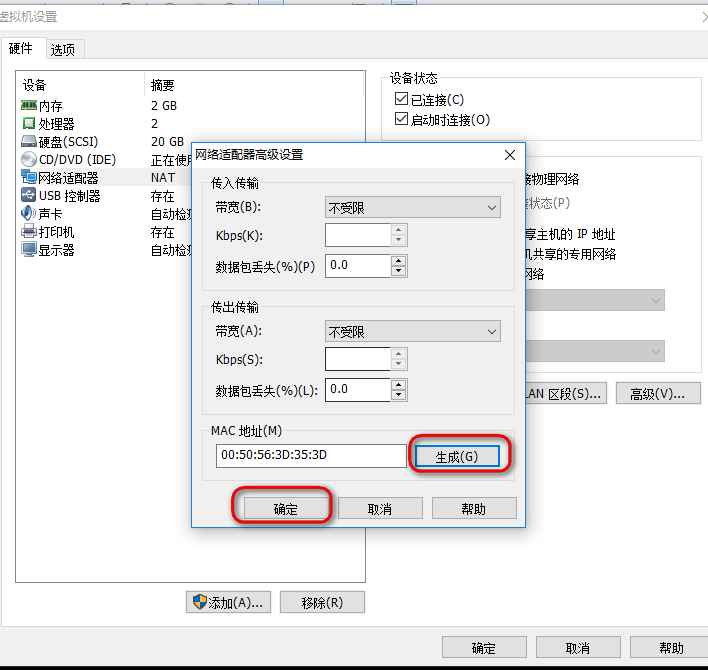
## 2.1 从已经安装好rabbitmq的机器 clone 三台机器

## 2.2 重新设置三台机器的mac地址

注意clone完，先不要启动三台机器，三台机器均要重新生成mac地址，防止clone出的机器ip地址重复







## 2.3 启动三台机器

启动并查看三台机器的ip地址：

|  |
| --- |
| ip a |

## 2.4 使用xshell 连接三台机器

## 2.5 修改三台机器的/etc/hosts 文件

首先需要配置一下hosts文件，因为RabbitMQ集群节点名称是读取hosts文件得到的；

vim /etc/hosts

|  |
| --- |
| 192.168.131.128 rabbit128  192.168.131.129 rabbit129  192.168.131.130 rabbit130 |

## 2.6 三台机器均重启网络，使节点名生效

|  |
| --- |
| systemctl restart network |

## 2.7 三台机器的xshell均退出，然后再重新连接

## 2.8 三台机器的防火墙处理

|  |
| --- |
| systemctl status firewalld  systemctl stop firewalld --关闭防火墙  systemctl disable firewalld --开机不启动防火墙 |

## 2.9 三台机器 .erlang.cookie文件保持一致

**由于是clone出的三台机器，所以肯定是一样的**

|  |
| --- |
| 如果我们使用解压缩方式安装的RabbitMQ，那么该文件会在${用户名}目录下，也就是${用户名}/.erlang.cookie；  如果我们使用rpm安装包方式进行安装，那么这个文件会在/var/lib/rabbitmq目录下； |

**注意 .erlang.cookie的权限为400，目前已经是400**

# **3 分别启动三台机器上的rabbitmq**

## 3.1 启动

|  |
| --- |
| **rabbitmq-server -detached** |

## 3.2 查看集群状态

使用以下命令查看集群状态

|  |
| --- |
| rabbitmqctl cluster\_status |

## 3.4 构建集群

在rabbitmq129机器上执行命令，让129的rabbitmq加入集群：

|  |
| --- |
| ./rabbitmqctl stop\_app  ./rabbitmqctl reset  ./rabbitmqctl join\_cluster rabbit@rabbit128 --ram  ./rabbitmqctl start\_app |

--ram 参数表示让rabbitmq129成为一个内存节点，如果不带参数默认为disk磁盘节点；

把rabbit129节点添加完之后；

在rabbit130节点上也执行同样的命令，使rabbit130节点也加入到集群中，

|  |
| --- |
| ./rabbitmqctl stop\_app  ./rabbitmqctl reset  ./rabbitmqctl join\_cluster rabbit@rabbit128 --ram  ./rabbitmqctl start\_app |

当然也可以让rabbit130作为一个磁盘节点

## 3.5 操作一个节点，添加用户和权限等

|  |
| --- |
| #列出用户  rabbitmqctl list\_users  # 添加用户  rabbitmqctl add\_user admin 123456  #查看权限  rabbitmqctl list\_permissions  #设置权限  rabbitmqctl set\_permissions admin ".\*" ".\*" ".\*"  #设置角色  rabbitmqctl set\_user\_tags admin administrator  #启动web控制台插件  ./rabbitmq-plugins enable rabbitmq\_management |

使用web浏览器添加一个虚拟主机 ：powernode

## 3.6 再次查看集群状态

当执行完操作以后我们在浏览器访问web管控台来看看效果；

随便在哪个节点打开web管控台都能看到集群环境各节点的信息；

也可以使用 ./rabbitmqctl cluster\_status 查看集群状态；

## 3.7 一些原理

|  |
| --- |
| RabbitMQ底层是通过Erlang架构来实现的，所以rabbitmqctl会启动Erlang节点，并基于Erlang节点来使用Erlang系统连接RabbitMQ节点，在连接过程中需要正确的Erlang Cookie和节点名称，Erlang节点通过交换Erlang Cookie以获得认证； |

以上就是RabbitMQ默认集群模式（普通集群模式）的搭建；

## 3.8 Springboot 连接集群

|  |
| --- |
| spring:  *#配置rabbitmq* rabbitmq:  *# 连接集群，使用逗号分隔* addresses: 192.168.150.150:5672,192.168.150.151:5672,192.168.150.152:5672  username: admin  password: 123456  virtual-host: powernode |

# **4 镜像集群模式**

镜像模式是基于默认集群模式加上一定的配置得来的；

在默认模式下的RabbitMQ集群，它会把所有节点的交换机、绑定、队列的元数据进行复制确保所有节点都有一份相同的元数据信息，但是队列数据分为两种：一种是队列的元数据信息（比如队列的最大容量，队列的名称等配置信息），另一种是队列里面的消息；

镜像模式，则是把所有的队列数据完全同步，包括元数据信息和消息数据信息，当然这对性能肯定会有一定影响，当对数据可靠性要求较高时，可以使用镜像模式；

实现镜像模式也非常简单，它是在普通集群模式基础之上搭建而成的；

镜像队列配置命令：

./rabbitmqctl set\_policy [-p Vhost] Name Pattern Definition [Priority]

-p Vhost： 可选参数，针对指定vhost下的queue进行设置；

Name: policy的名称；（可以自己取个名字就可以）

Pattern: queue的匹配模式(正则表达式)；

Definition：镜像定义，包括三个部分ha-mode, ha-params, ha-sync-mode；（json格式）

{“ha-mode”:”exactly”,”ha-params”:2}

ha-mode:指明镜像队列的模式，有效值为 all/exactly/nodes

all：表示在集群中所有的节点上进行镜像

exactly：表示在指定个数的节点上进行镜像，节点的个数由ha-params指定

nodes：表示在指定的节点上进行镜像，节点名称通过ha-params指定

ha-params：ha-mode模式需要用到的参数

ha-sync-mode：进行队列中消息的同步方式，有效值为automatic和manual

priority：可选参数，policy的优先级；

比如想配置所有名字开头为policy\_的队列进行镜像，镜像数量为2，那么命令如下（在任意节点执行如下命令）：

./rabbitmqctl set\_policy [-p Vhost] Name Pattern Definition [Priority]

|  |
| --- |
| ./rabbitmqctl set\_policy -p powernode **ha\_policy** "^policy\_" '{"ha-mode":"exactly","ha-params":2,"ha-sync-mode":"automatic"}' |

如果要在所有节点所有队列上进行镜像，则（在任意节点执行如下命令）：

所有节点、所有虚拟主机、所有队列 都进行镜像

./rabbitmqctl set\_policy [-p Vhost] Name Pattern Definition [Priority]

|  |
| --- |
| ./rabbitmqctl set\_policy **ha-all** "^" '{"ha-mode":"all"}' |

针对某个虚拟主机进行镜像

|  |
| --- |
| rabbitmqctl set\_policy -p powernode ha-all "^" '{"ha-mode":"all","ha-sync-mode":"automatic"}' |

（在默认集群模式的基础上，执行上面这个命令就可以把一个默认的集群模式变成镜像集群模式）