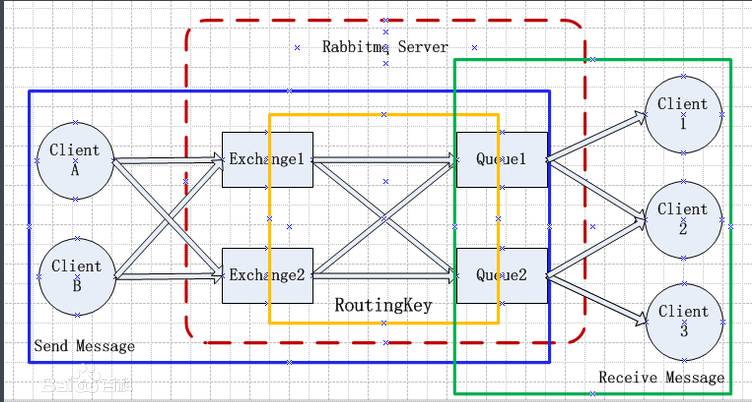
Rabbitmq

初识rabbitmq

RabbitMQ是流行的开源消息队列系统，用erlang语言开发。RabbitMQ是AMQP（高级消息队列协议）的标准实现。如果不熟悉AMQP，直接看RabbitMQ的文档会比较困难。不过它也只有几个关键概念，这里简单介绍



几个概念说明：

Broker：简单来说就是消息队列服务器实体。  
　　Exchange：消息交换机，它指定消息按什么规则，路由到哪个队列。  
　　Queue：消息队列载体，每个消息都会被投入到一个或多个队列。  
　　Binding：绑定，它的作用就是把exchange和queue按照路由规则绑定起来。  
　　Routing Key：路由关键字，exchange根据这个关键字进行消息投递。  
　　vhost：虚拟主机，一个broker里可以开设多个vhost，用作不同用户的权限分离。  
　　producer：消息生产者，就是投递消息的程序。  
　　consumer：消息消费者，就是接受消息的程序。  
　　channel：消息通道，在客户端的每个连接里，可建立多个channel，每个channel代表一个会话任务。

由Exchange，Queue，RoutingKey三个才能决定一个从Exchange到Queue的唯一的线路。

消息队列的使用过程大概如下：

（1）客户端连接到消息队列服务器，打开一个channel。  
　　（2）客户端声明一个exchange，并设置相关属性。  
　　（3）客户端声明一个queue，并设置相关属性。  
　　（4）客户端使用routing key，在exchange和queue之间建立好绑定关系。  
　　（5）客户端投递消息到exchange。

exchange接收到消息后，就根据消息的key和已经设置的binding，进行消息路由，将消息投递到一个或多个队列里。

exchange也有几个类型，完全根据key进行投递的叫做Direct交换机，例如，绑定时设置了routing key为”abc”，那么客户端提交的消息，只有设置了key为”abc”的才会投递到队列。对key进行模式匹配后进行投递的叫做Topic交换机，符号”#”匹配一个或多个词，符号”\*”匹配正好一个词。例如”abc.#”匹配”abc.def.ghi”，”abc.\*”只匹配”abc.def”。还有一种不需要key的，叫做Fanout交换机，它采取广播模式，一个消息进来时，投递到与该交换机绑定的所有队列。

RabbitMQ支持消息的持久化，也就是数据写在磁盘上，为了数据安全考虑，我想大多数用户都会选择持久化。消息队列持久化包括3个部分：  
　　（1）exchange持久化，在声明时指定durable => 1  
　　（2）queue持久化，在声明时指定durable => 1  
　　（3）消息持久化，在投递时指定delivery\_mode => 2（1是非持久化）

如果exchange和queue都是持久化的，那么它们之间的binding也是持久化的。如果exchange和queue两者之间有一个持久化，一个非持久化，就不允许建立绑定。

安装开发环境和库

1.将目录中的librabbitmq.so.1 放到目录 /usr/local/lib/librabbitmq.so.1

2.安装rabbitm需要的环境和库

yum install -y ncurses-devel

yum install gcc

yum install g++

yum install cmake

yum install make

yum install php

yum install mysql

yum install php-process

yum install php-devel

yum install mysql-server

#安装php的amq支持扩展

wget http://pecl.php.net/get/amqp-1.0.3.tgz

tar zxvf amqp-1.0.3.tgz

cd amqp-1.0.3

/usr/bin/phpize

./configure --with-php-config=/usr/bin/php-config --with-amqp

make && make install

#php.ini 添加

vi /etc/php.ini

extension="amqp.so"

#安装erlang支持

wget http://www.erlang.org/download/otp\_src\_R15B01.tar.gz

tar -zxvf otp\_src\_R15B01.tar.gz

cd otp\_src\_R15B01

./configure --prefix=/home/erlang --without-javac

make && make install

ln -s /home/erlang/bin/erl /usr/local/bin/erl

3. 安装rabbitma

解压rabbitmq-server-generic-unix-3.3.4.tar

进入sbin目录：

启动rabbitmq服务，执行 nohup ./rabbitmq-server start &

启动rabbitmq服务器以及命令

当第一次启动服务，检测数据库是否未初始化或者被删除，它会用下面的资源初始化一个新的数据库:

一个命名为 / 的虚拟宿主一个名为guest密码也为guest的用户，他拥有/虚拟宿主的所有权限如果你的中间件是公开访问的，最好修改guest用户的密码。管理概观rabbitmqctl 是RabbitMQ中间件的一个命令行管理工具。它通过连接一个中间件节点执行所有的动作。本地节点默认被命名为”rabbit”。可以通过这个命令前使 用”-n”标志明确的指定节点名称, 例如:# rabbitmqctl -n rabbit@shortstop add\_user tonyg changeit

这个命令指示RabbitMQ中间件在rabbit@shortstop 节点创建一个tonyg/changeit的用户。

在一个名为”server.example.com”的主机，RabbitMQ Erlang节点的名称通常是rabbit@server(除非RABBITMQ\_NODENAM在 中间件启动时候被设置)。hostnam -s 的输出通常是”@”符号正确的后缀。rabbitmqctl 默认产生详细输出。通过”-q”标示可选择安静模式。rabbitmqctl -q status应用和集群管理1.停止RabbitMQ应用，关闭节点

# rabbitmqctl stop

2.停止RabbitMQ应用

# rabbitmqctl stop\_app

3.启动RabbitMQ应用

# rabbitmqctl start\_app

4.显示RabbitMQ中间件各种信息

# rabbitmqctl status

5.重置RabbitMQ节点

# rabbitmqctl reset

# rabbitmqctl force\_reset

从它属于的任何集群中移除，从管理数据库中移除所有数据，例如配置过的用户和虚拟宿主, 删除所有持久化的消息。

force\_reset命令和reset的区别是无条件重置节点，不管当前管理数据库状态以及集群的配置。如果数据库或者集群配置发生错误才使用这个最后 的手段。

注意：只有在停止RabbitMQ应用后，reset和force\_reset才能成功。

6.循环日志文件

# rabbitmqctl rotate\_logs[suffix]

7.集群管理

# rabbitmqctl cluster clusternode…

用户管理

1.添加用户

# rabbitmqctl add\_user username password

2.删除用户

# rabbitmqctl delete\_user username

3.修改密码

# rabbitmqctl change\_password username newpassword

4.列出所有用户

# rabbitmqctl list\_users

权限控制1.创建虚拟主机

# rabbitmqctl add\_vhost vhostpath

2.删除虚拟主机

# rabbitmqctl delete\_vhost vhostpath

3.列出所有虚拟主机

# rabbitmqctl list\_vhosts

4.设置用户权限

# rabbitmqctl set\_permissions [-p vhostpath] username regexp regexp regexp

5.清除用户权限

# rabbitmqctl clear\_permissions [-p vhostpath] username

6.列出虚拟主机上的所有权限

# rabbitmqctl list\_permissions [-p vhostpath]

7.列出用户权限

# rabbitmqctl list\_user\_permissions username

例子：

添加 rabbitmqctl add\_vhost az

rabbitmqctl set\_permissions -p az guest ".\*" ".\*" ".\*"

接口描述

amqp\_connection\_state\_t amqp\_new\_connection(void);

    接口说明：声明一个新的amqp connection

int amqp\_open\_socket(char const \*hostname, int portnumber);

    接口说明：获取socket.

    参数说明：hostname         RabbitMQ server所在主机

                  portnumber      RabbitMQ server监听端口

void amqp\_set\_sockfd(amqp\_connection\_state\_t state,int sockfd);

     接口说明：将amqp connection和sockfd进行绑定

amqp\_rpc\_reply\_t amqp\_login(amqp\_connection\_state\_t state, char const \*vhost,int channel\_max,int frame\_max,int heartbeat,amqp\_sasl\_method\_enum sasl\_method, ...);

    接口说明：用于登录RabbitMQ server，主要目的为了进行权限管理；

    参数说明：state    amqp connection

                  vhost   rabbit-mq的虚机主机，是rabbit-mq进行权限管理的最小单位

                  channel\_max  最大链接数，此处设成0即可

                  frame\_max  和客户端通信时所允许的最大的frame size.默认值为131072，增大这个值有助于提高吞吐，降低这个值有利于降低时延

                  heartbeat 含义未知，默认值填0

                  sasl\_method  用于SSL鉴权，默认值参考后文demo

amqp\_channel\_open\_ok\_t \*amqp\_channel\_open(amqp\_connection\_state\_t state, amqp\_channel\_t channel);

    接口说明：用于关联conn和channel

amqp\_exchange\_declare\_ok\_t \*amqp\_exchange\_declare(amqp\_connection\_state\_t state, amqp\_channel\_t channel, amqp\_bytes\_t exchange, amqp\_bytes\_t type, amqp\_boolean\_t passive, amqp\_boolean\_t durable, amqp\_table\_t arguments);

    接口说明：声明declare

    参数说明：state

                  channel

                  exchange

                  type     "fanout"  "direct" "topic"三选一

                  passive

                  curable

                  arguments

amqp\_queue\_declare\_ok\_t \*amqp\_queue\_declare(amqp\_connection\_state\_t state, amqp\_channel\_t channel, amqp\_bytes\_t queue, amqp\_boolean\_t passive, amqp\_boolean\_t durable, amqp\_boolean\_t exclusive, amqp\_boolean\_t auto\_delete, amqp\_table\_t arguments);

    接口说明：声明queue

    参数说明：state   amqp connection

                  channel

                  queue  queue name

                  passive

                  durable  队列是否持久化

                  exclusive  当前连接不在时，队列是否自动删除

                  aoto\_delete 没有consumer时，队列是否自动删除

                  arguments 用于拓展参数，比如x-ha-policy用于mirrored queue

amqp\_queue\_bind\_ok\_t \*amqp\_queue\_bind(amqp\_connection\_state\_t state, amqp\_channel\_t channel, amqp\_bytes\_t queue, amqp\_bytes\_t exchange, amqp\_bytes\_t routing\_key, amqp\_tab le\_t arguments);

    接口说明：声明binding

amqp\_basic\_qos\_ok\_t \*amqp\_basic\_qos(amqp\_connection\_state\_t state, amqp\_channel\_t channel, uint32\_t prefetch\_size, uint16\_t prefetch\_count, amqp\_boolean\_t global);

     接口说明：qos是 quality of service，我们这里使用主要用于控制预取消息数，避免消息按条数均匀分配，需要和no\_ack配合使用

     参数说明：state

                   channel

                   prefetch\_size 以bytes为单位，0为unlimited

                   prefetch\_count 预取的消息条数

                   global

amqp\_basic\_consume\_ok\_t \*amqp\_basic\_consume(amqp\_connection\_state\_t state, amqp\_channel\_t channel, amqp\_bytes\_t queue, amqp\_bytes\_t consumer\_tag, amqp\_boolean\_t no\_local, amqp\_boolean\_t no\_ack, amqp\_boolean\_t exclusive, amqp\_table\_t arguments);

     接口说明：开始一个queue consumer

     参数说明：state

                   channel

                   queue

                   consumer\_tag

                   no\_local

                   no\_ack    是否需要确认消息后再从队列中删除消息

                   exclusive

                   arguments

int amqp\_basic\_ack(amqp\_connection\_state\_t state,amqp\_channel\_t channel,uint64\_t delivery\_tag,amqp\_boolean\_t multiple);

int amqp\_basic\_publish(amqp\_connection\_state\_t state,amqp\_channel\_t channel,amqp\_bytes\_t exchange,amqp\_bytes\_t routing\_key,amqp\_boolean\_t mandatory,amqp\_boolean\_t immediate,struct amqp\_basic\_properties\_t\_ const \*properties,amqp\_bytes\_t body);

    接口说明：发布消息

    参数说明：state

                  channel

                  exchange

                  routing\_key  当exchange为默认“”时，此处填写queue\_name，当exchange为direct，此处为binding\_key

                  mandatory 参见参考文献2

                  immediate 同上

                  properties 更多属性，如何设置消息持久化，参见文后demo

                  body 消息体

amqp\_rpc\_reply\_t amqp\_channel\_close(amqp\_connection\_state\_t state,amqp\_channel\_t channel,int code);

amqp\_rpc\_reply\_t amqp\_connection\_close(amqp\_connection\_state\_t state,int code);

int amqp\_destroy\_connection(amqp\_connection\_state\_t state);

[rabbitmq server的安装以及常用的命令](http://www.cnblogs.com/mingaixin/p/4134929.html)

Centos 源代码编译 安装 Erlang  
Erlang依赖哪些库？  
  
    A fully working GCC compiler environment  
         Ncurses development libraries  
         OpenSSL development libraries （如果使用mysql必须安装）  
  
    安装了这些库之后，必须要重新执行configure命令，configure之后会有提示哪些依赖的库没有安装，可以根据你的需要放弃安装一些库；上面的操作可以使用下面的命令实现：  
  
  
sudo yum -y install make gcc gcc-c++ kernel-devel m4 ncurses-devel openssl-devel  
wget http://www.erlang.org/download/otp\_src\_17.3.tar.gz  
tar -zxvf otp\_src\_17.3.tar.gz  
cd otp\_src\_17.3  
./configure --with-ssl  
make   
make install  
  
  
Centos 源代码编译 安装 RabbitMQ  
 yum install libxslt python zip unzip nc -y  
 wget http://www.rabbitmq.com/releases/rabbitmq-server/v3.4.1/rabbitmq-server-generic-unix-3.4.1.tar.gz  
 tar -zxvf rabbitmq-server-generic-unix-3.4.1.tar.gz  
 mv  解压出来的文件夹  /usr/loca/rabbitmq/  
 启动 rabbitmq server  
 /usr/local/rabbitmq/sbin/rabbitmq-server –detached  
  
关闭:rabbitmqctl stop  
  
另外如果你用同一个应用启动了不同实例（eg:RABBITMQ\_NODE\_PORT=5673 RABBITMQ\_NODENAME=hare rabbitmq-server -detached），  
在进行管理操作的时候需要指定节点名: rabbitmqctl -n node1  
  
  
  
 1.必需掌握的指令  
  
添加用户:  
  
rabbitmqctl add\_user rainbird password  
  
添加权限:  
  
rabbitmqctl set\_permissions -p "/" rainbird ".\*" ".\*" ".\*"  
  
删除测试用户:  
  
rabbitmqctl delete\_user guest  
  
   
  
所有指令列表(很简单的英文):  
  
add\_user        <UserName> <Password>  
  
delete\_user     <UserName>  
  
change\_password <UserName> <NewPassword>  
  
list\_users  
  
add\_vhost    <VHostPath>  
  
delete\_vhost <VHostPath>  
  
list\_vhosts  
  
set\_permissions   [-p <VHostPath>] <UserName> <Regexp> <Regexp> <Regexp>  
  
clear\_permissions [-p <VHostPath>] <UserName>  
  
list\_permissions  [-p <VHostPath>]  
  
list\_user\_permissions <UserName>  
  
list\_queues    [-p <VHostPath>] [<QueueInfoItem> ...]  
  
list\_exchanges [-p <VHostPath>] [<ExchangeInfoItem> ...]  
  
list\_bindings  [-p <VHostPath>]   
  
list\_connections [<ConnectionInfoItem> ...]  
  
       2.vhost  / 不能删除  
  
   
  
      删除/以后,新建立的vhost不能正常使用(即便不删除/,新建立的vhost也是不能正常使用).不知道为什么,有待研究.  
  
  
  
  
./rabbitmqctl  add\_user  admin  \*\*\*\*\*  
./rabbitmqctl  add\_user  push  \*\*\*\*\*\*  
./rabbitmqctl  set\_user\_tags  admin administrator  
./rabbitmqctl   set\_permissions  -p  /  admin  '.\*' '.\*' '.\*'   
./rabbitmqctl   set\_permissions  -p  /  push  '.\*' '.\*' '.\*'    
 ./rabbitmqctl  list\_permissions

http://www.rabbitmq.com/configure.html