# 两种简单Rabbitmq使用方案及其测试

## 方案一 简单负载均衡方案

### 问题：

rabbitmq可以为Consumers做负载均衡，但rabbimq自身并没有负载均衡。用户连接到rabbitmq集群的任意节点都可以访问集群中的任意消息队列，但一个消息队列只存储在一个物理节点上，其它节点只存储该队列的元数据，这使得当队列里只有一个队列时，系统性能受限于单个节点的网络带宽和主机性能。若使用多个队列来提升性能，也会有新的问题，即如何在队列之间做负载均衡，同时网络连接也会影响系统性能，比如当一个用户往某个消息队列发消息时，而该用户当前连接的节点不是该队列真实所在的物理节点，这必然会产生rabbitmq节点间通讯，从而消耗的一部分网络带宽。

### 方案：

为了解决以上问题，有以下方案（发送端做负载均衡,随机发送集群中任意节点），

1.建立多个消息队列，每个物理节点上消息队列数相同。

2.exchange的类型设置为direct，建立多个binding，每个队列对应一个key。

3.每个publisher建立到每个物理节点的连接。

4.每个worker订阅所有消息队列,。

5.发送消息时随机选择一个key,并使用该key对应的队列所有在节点的连接发送该消息。

6.当某个mq节点挂掉后，发送者将消息随机发送到其余节点，并一直监控该挂掉的节点是否重起，重启后，即可向该节点发消息。

示意图如下



### 测试：

#### 1.测试环境

硬件环境：

发送者（my031090, rds064071,rds064072）

rabbit节点（rds064073, rds064075,rds064074）

接收者（rds064076,rds064077,my031091）

软件环境：

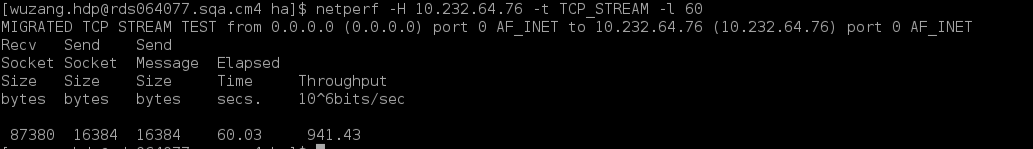
内核2.6.32-220.el6.x86\_64

rabbitmq: 2.8.1

erlang:R16B

rabbit-client: rabbit-erlang-client

网络环境：

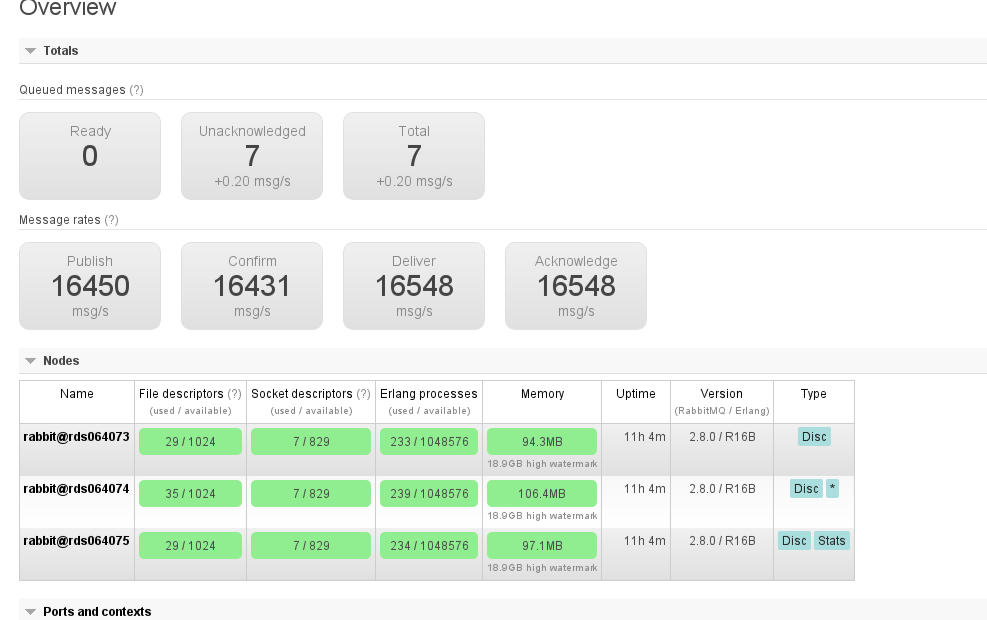


≈117MB/s

#### 2.测试结果

**（1）包大小：1byte**

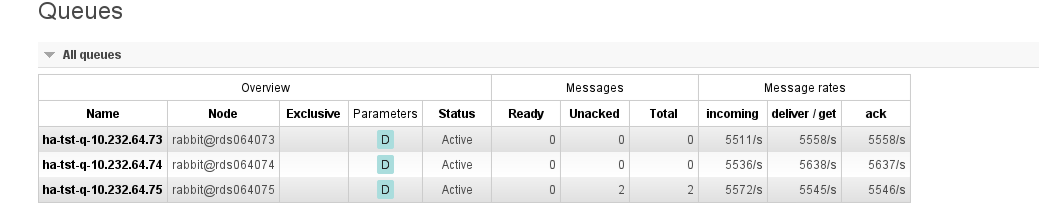
集群整体每秒传输包个数：



每个连接传输速率

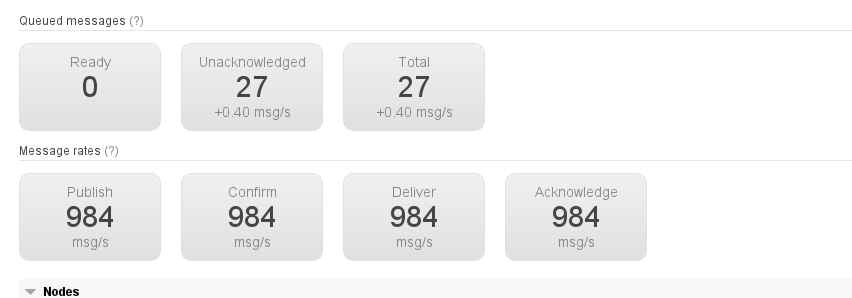


各队列数据包收发速率：

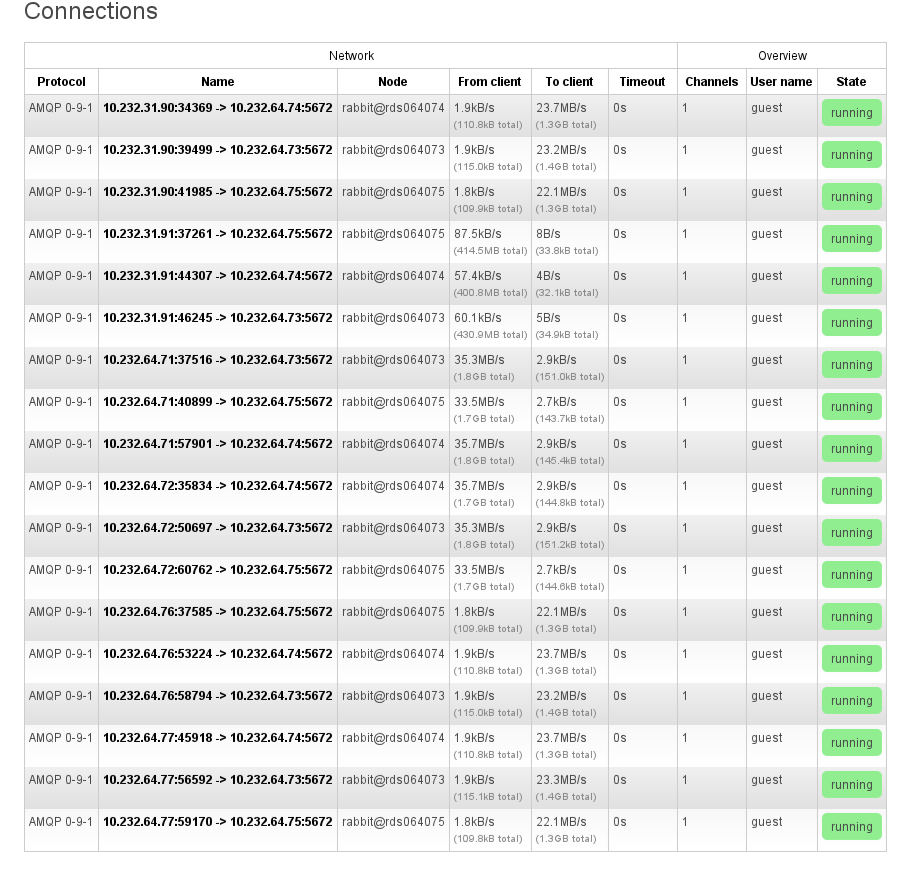


**（2）包大小：256k**

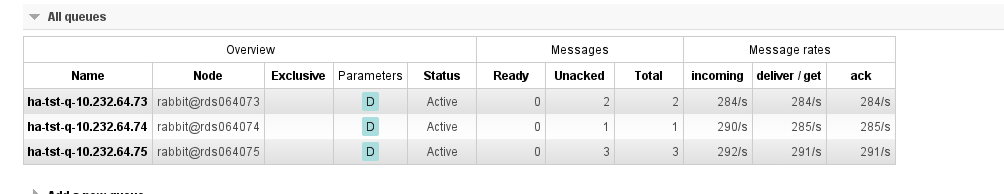
集群整体每秒传输包个数：



各连接传输速率：



队列收发速率：



#### 3.结论

从上述测试结果可以看出，该方案基本实习了Rabbitmq的负载均衡，在数据包大小为256k时网络吞吐量（250MB/s）也比较理想。

## 方案二 高可用方案

### 问题：

Rabbitmq现提供队列mirror功能，通过这一功能可以提高Rabbitmq的可靠性，当某个Rabbitmq节点故障时，只要其它节点里存在该故障节点的队列镜像，该队列就能继续正常工作不会丢失数据。但使用该功能也会有些副作用，它这种通过冗余数据保障可靠性的方式会降低系统的性能，因为往一个队列发数据也就会往这个队列的所有镜像队列发数据，这必然产生大量Rabbitmq节点间数据的交互，降低吞吐率，镜像越多性能必然下降越多。与此同时，为充分利用集群的的资源，需要创建多个队列，若在所有节点上都有每个队列的镜像来实现可靠性，则队列镜像数会太多，过多的RabbitMq集群内部网络通讯会吃掉大量网络带宽。

### 方案：

为解决上述问题，我们**实现一个允许挂一个节点的方案**，该方案在方案一的基础上加上以下2条：

1. 每个队列只有一个镜像，镜像的位置为“下一个节点”，节点的分布如下图



1. 消费者端监控所有链接，当发现某个节点挂掉时，自动连接到镜像节点，而当故障节点恢复时自动连接回来。

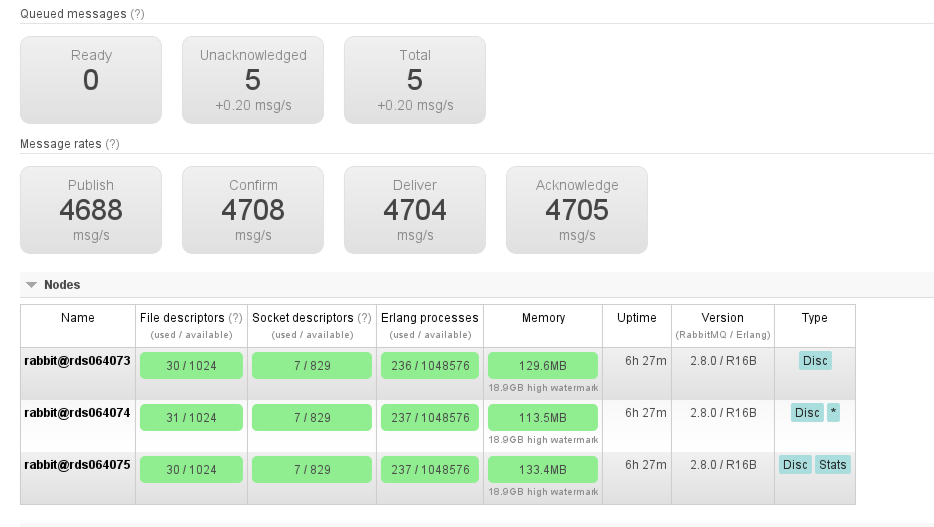
测试：

测试环境：与测试一相同

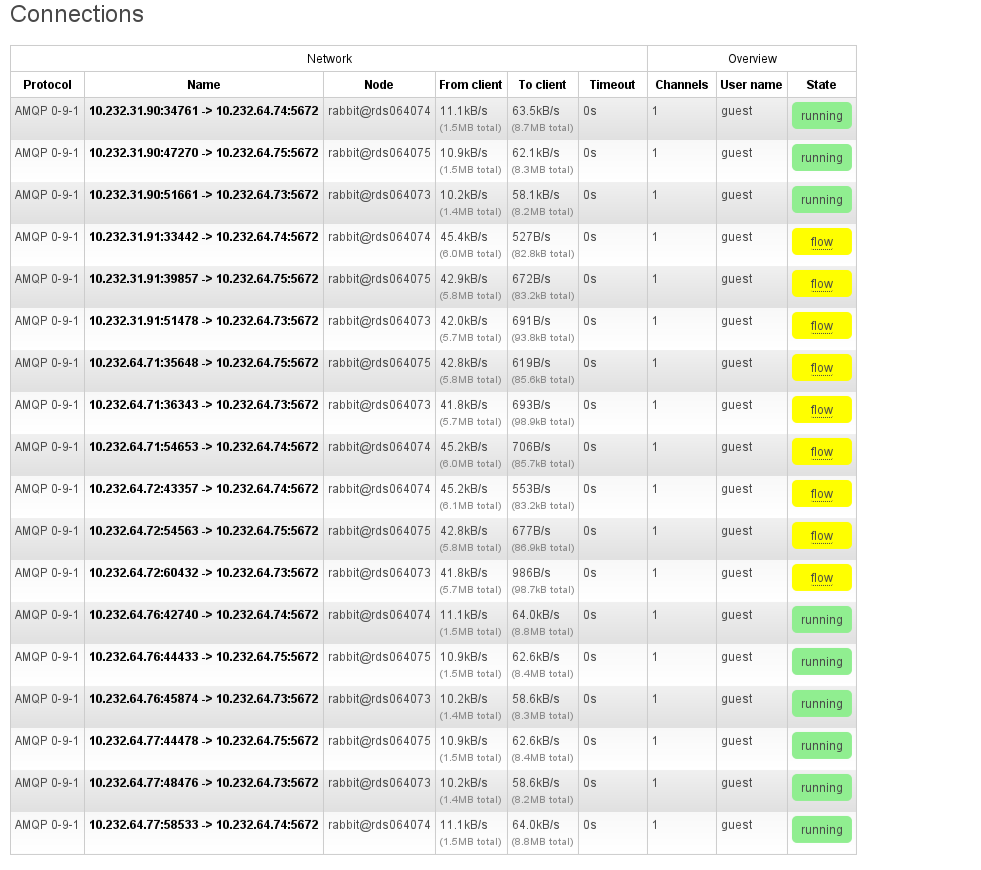
测试结果：

1. **包大小1byte：**

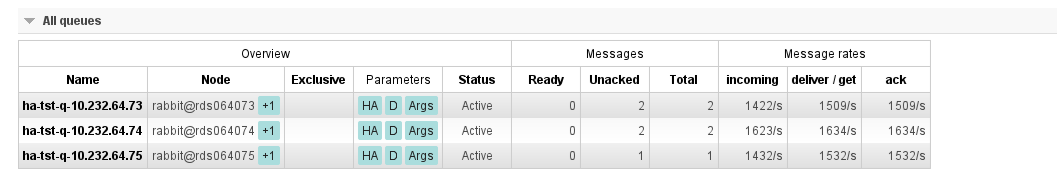
集群每秒处理包数：



各连接传输速率：

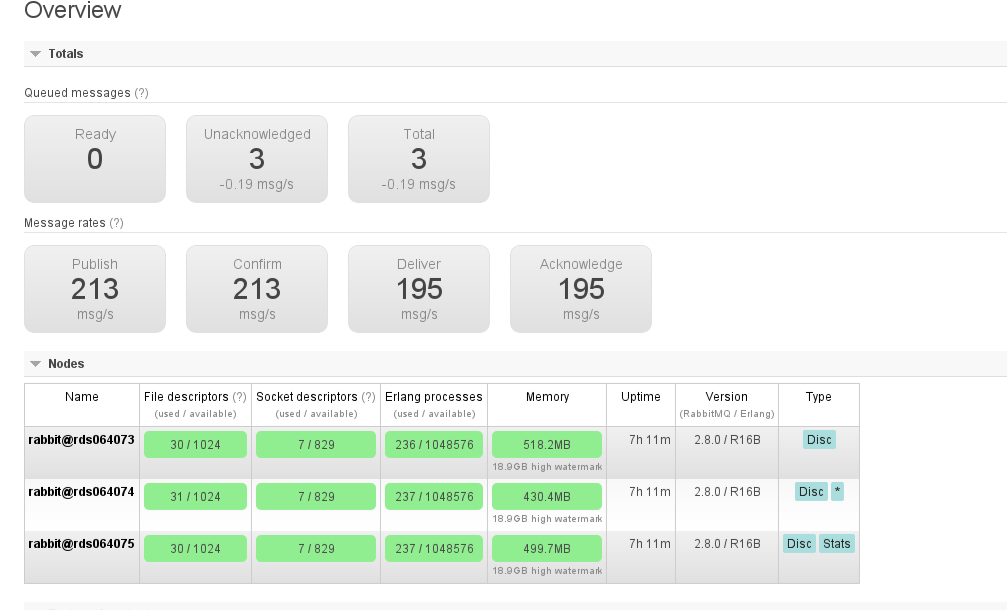


各队列数据包收发速率：

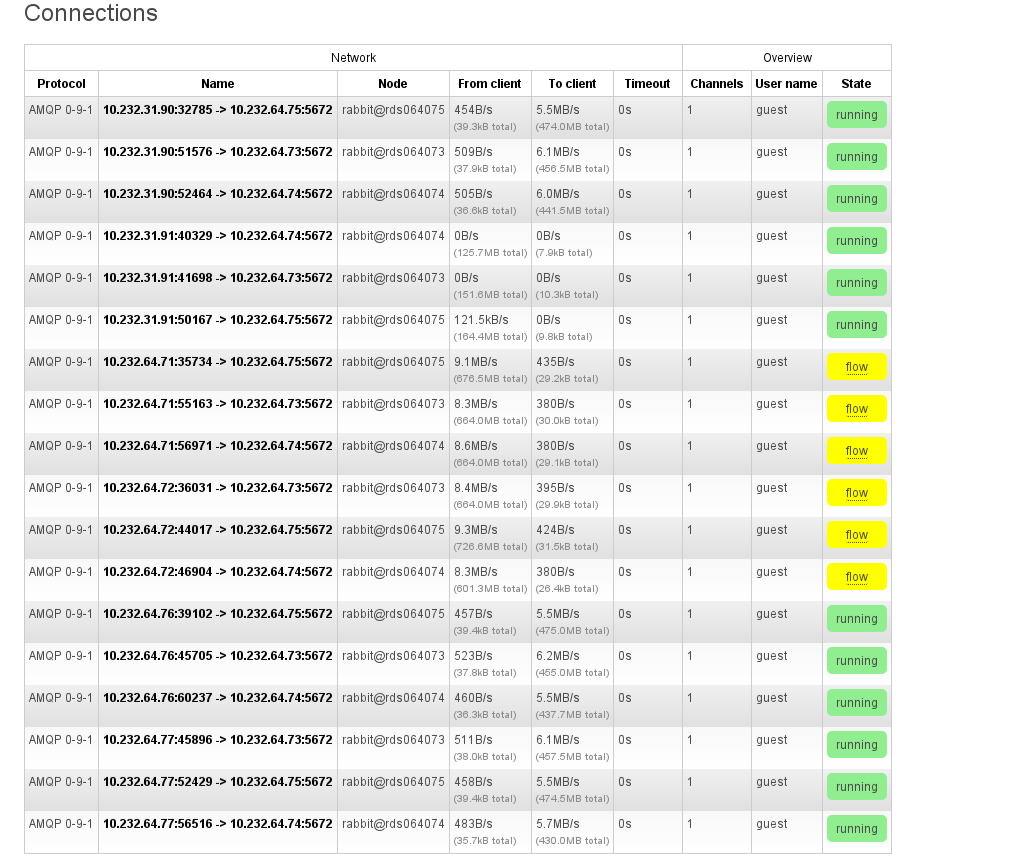


1. **包大小（256K）**

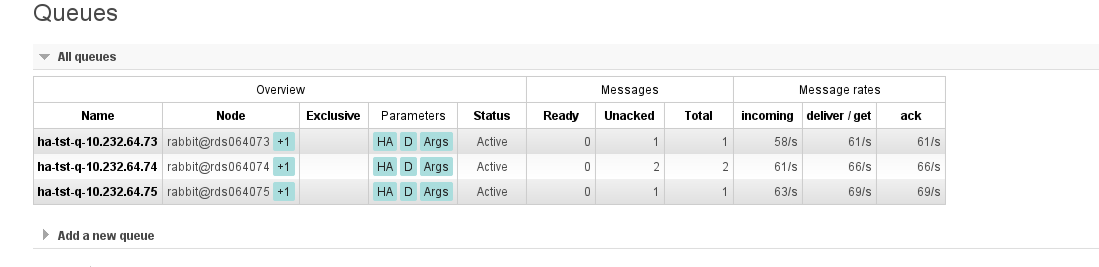
集群每秒处理包数：



各连接传输速率：



各队列数据包收发速率：



结论：

与方案一的测试结果做比较可以看出，开启mirror后吞吐量大大降低，只有不到原来的1/4，