GMQ消息堆积启动时长测试

针对版本V1.0.0

**©成都基础平台架构部**

2017/3/2

修订记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 修订内容 | 作者 | 审核 | 修订日期 |
| V1.0.0 | 初始版本 | 郜焱磊 | 成都基础平台架构部 | 2017/3/02 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[1 背景 5](#_Toc476297598)

[2 环境配置 5](#_Toc476297599)

[2.1 原集群节点 5](#_Toc476297600)

[2.2 升级集群节点 5](#_Toc476297601)

[3 测试目标 6](#_Toc476297602)

[3.1 集群初始状态 6](#_Toc476297603)

[4 升级测试 7](#_Toc476297604)

[4.1 测试一 7](#_Toc476297605)

[4.1.1 测试场景 7](#_Toc476297606)

[4.1.2 测试流程 7](#_Toc476297607)

[4.1.3 升级前 8](#_Toc476297608)

[4.1.4 开始升级 8](#_Toc476297609)

[4.1.5 升级后 9](#_Toc476297610)

[4.1.6 测试完毕 11](#_Toc476297611)

[4.1.7 本次测试结论 13](#_Toc476297612)

[4.2 测试二 13](#_Toc476297613)

[4.2.1 测试场景 13](#_Toc476297614)

[4.2.2 测试流程 13](#_Toc476297615)

[4.2.3 升级前 14](#_Toc476297616)

[4.2.4 开始升级 15](#_Toc476297617)

[4.2.5 升级后 17](#_Toc476297618)

[4.2.6 测试完毕 19](#_Toc476297619)

[4.2.7 本次测试结论 21](#_Toc476297620)

[5 整体结论 22](#_Toc476297621)

# 背景

由于系统升级会对服务进行关闭，需要掌握服务从关闭，启动到最终完全正常过程中不可用的时长。

# 环境配置

## 原集群节点

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 节点名称 | 主机地址 | 端口 | 备注 | 测试场景 |
| Broker- master1 | 10.128.31.104 | 10911 | 主 | 亿万级测试 |
| Broker- master1 | 10.128.31.106 | 10911 | 从 | 亿万级测试 |
| Broker- master3 | 10.128.31.223 | 10911 | 主 | 百，千万级测试 |
| Broker- master3 | 10.128.31.224 | 10911 | 从 | 百，千万级测试 |

# 测试目标

本次测试针对broker-master1，broker-master3两个节点进行测试，broker-master1消息堆积为亿万级。broker-master3消息堆积为千万级。 服务器与本地时间相差不超过5秒钟。

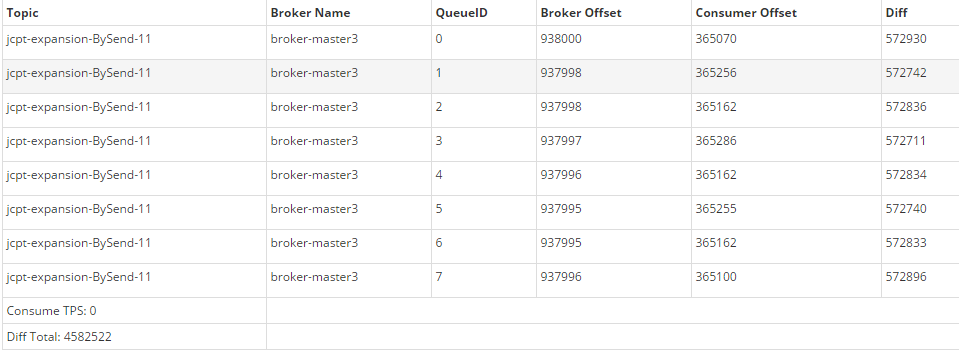
* 在堆积百万级消息情况集群，关闭重启多久恢复正常
* 在堆积千万级消息情况集群，关闭重启多久恢复正常
* 在堆积亿万级消息情况集群，关闭重启多久恢复正常

# 测试

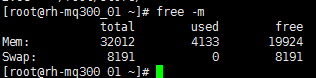
## 堆积百万级条数据启动时长

### 测试场景

堆积4582522条数据。



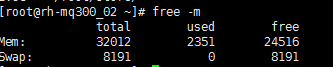
主节点内存



主节点硬盘



从节点内存



从节点硬盘

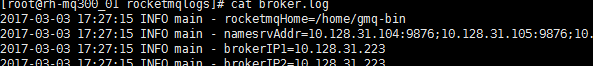


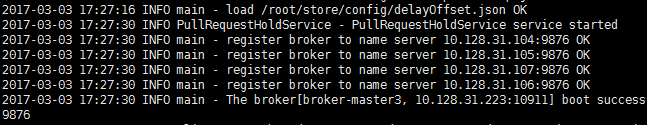
### 测试流程

1. 关闭节点写权限1分钟。
2. 关闭主从俩个节点。
3. 关闭进程
4. 备份Broker.log文件 为Broker\_backup.log。
5. 主从节点同时启动。
6. 发送程序，看程序是否有异常。
7. 查看新的Broker.log文件，计算启动时长

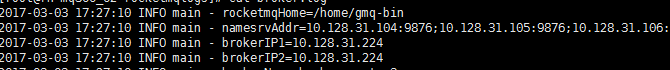
### 启动节点

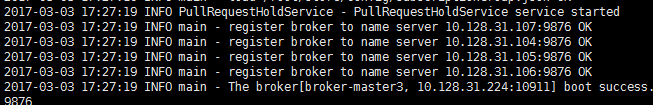
主节点：启动消耗时长为5秒



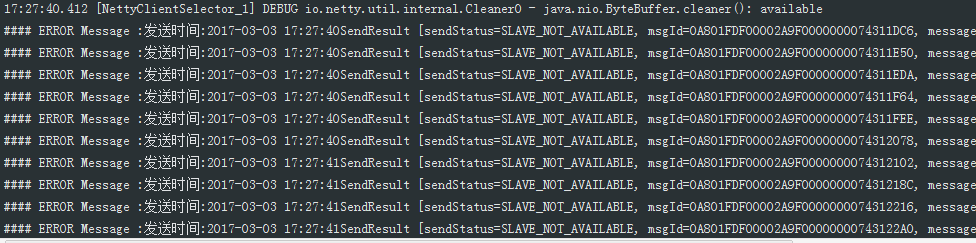


从节点：启动消耗时长9秒

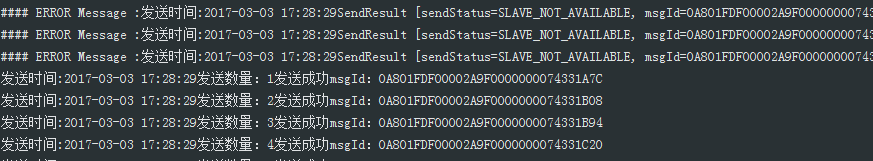




开始发送消息



在过1分钟过后正常



### 结论

* 在堆积4百万+条数据的情况下。数据大小为2个g，主节点和从节点启动时长分别为5秒和9秒。加上注册刷新路由30秒。如果主从节点同时启动保守估计2分钟内可加入集群。
* 但是启动主从后，立马发送消息，会出现SLAVE\_NOT\_AVAILABLE发送错误

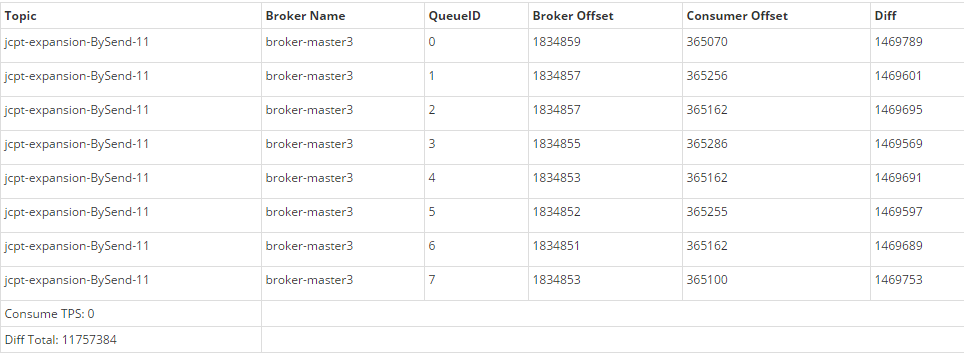
该错误官方解释为：消息发送成功,但是此时 slave 不可用,消息已经进入服务器队列,只有此时服务器宕机,消息才会丢失。并不会影响使用。

大约1分钟过后从节点可用。发送消息一切正常。

## 堆积千万级条数据启动时长

### 测试场景

堆积11757384条数据。



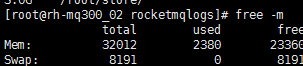
主节点内存



主节点硬盘



从节点内存



从节点硬盘

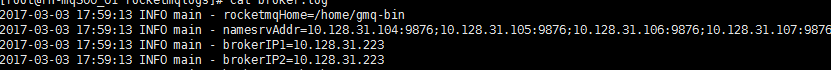


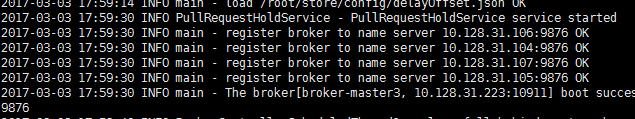
### 测试流程

1. 关闭节点写权限1分钟。
2. 关闭主从俩个节点。
3. 关闭进程
4. 备份Broker.log文件 为Broker\_backup.log。
5. 主从节点同时启动。
6. 发送程序，看程序是否有异常。
7. 查看新的Broker.log文件，计算启动时长

### 启动节点

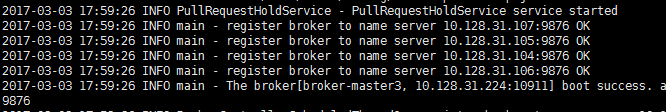
主节点：启动消耗时长为17秒



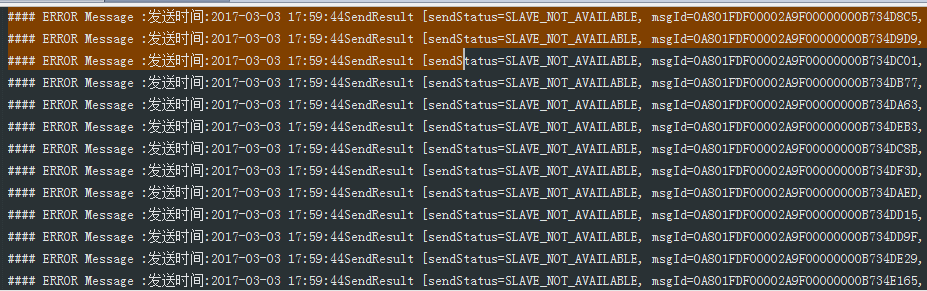


从节点：启动消耗时长10秒

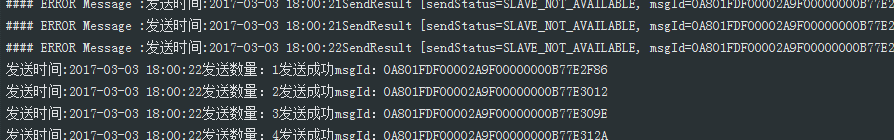




开始发送消息



在过40秒过后正常



### 结论

* 在堆积1100万+条数据的情况下。数据大小为3个g，主节点和从节点启动时长分别为17秒和10秒。加上注册刷新路由30秒。如果主从节点同时启动保守估计2分钟内可加入集群。
* 但是启动主从后，立马发送消息，会出现SLAVE\_NOT\_AVAILABLE发送错误

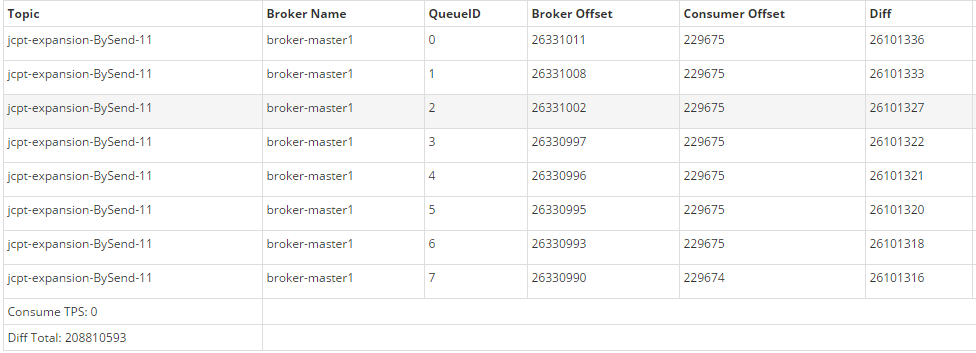
该错误官方解释为：消息发送成功,但是此时 slave 不可用,消息已经进入服务器队列,只有此时服务器宕机,消息才会丢失。并不会影响使用。

大约1分钟过后从节点可用。发送消息一切正常。

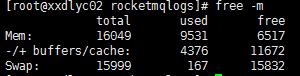
## 堆积两亿条数据启动时长一

### 测试场景

堆积208810593条数据。



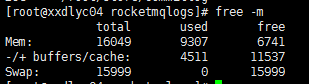
主节点内存



主节点硬盘



从节点内存



从节点硬盘

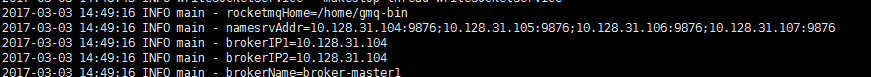


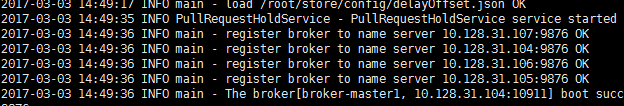
### 测试流程

1. 关闭节点写权限1分钟。
2. 关闭主从俩个节点。
3. 关闭进程
4. 备份Broker.log文件 为Broker\_backup.log。
5. 主从节点同时启动。
6. 发送程序，看程序是否有异常。
7. 查看新的Broker.log文件，计算启动时长

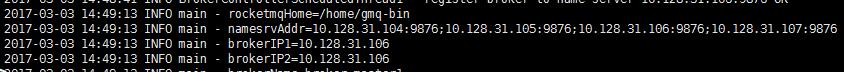
### 启动节点

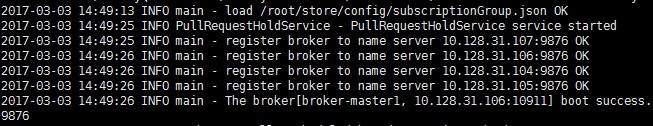
主节点：启动消耗时长为30秒



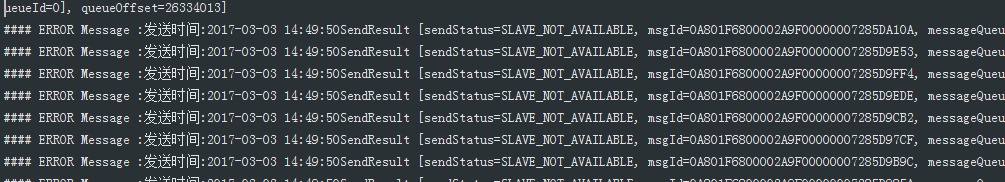


从节点：启动消耗时长13秒





立即开始发送消息



在过7分钟过后正常



### 结论

* 在堆积2亿+条数据的情况下。数据大小为29个g，主节点和从节点启动时长分别为13秒和30秒。加上注册刷新路由30秒。如果主从节点同时启动保守估计2分钟内可加入集群。
* 但是主从同时加入集群时，立马发送消息，会出现SLAVE\_NOT\_AVAILABLE发送错误

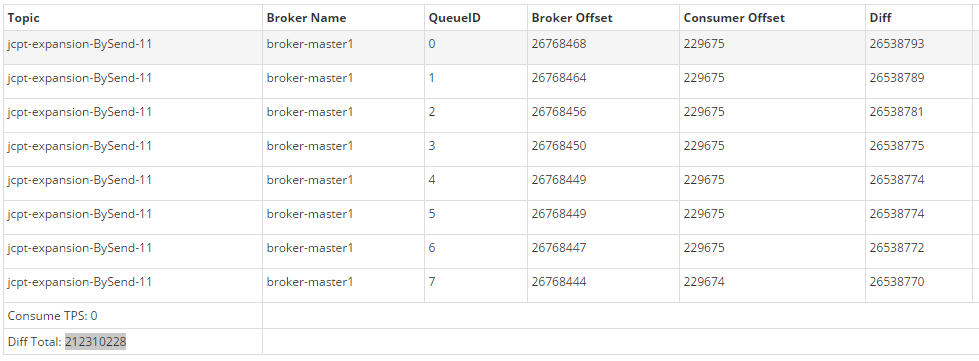
该错误官方解释为：消息发送成功,但是此时 slave 不可用,消息已经进入服务器队列,只有此时服务器宕机,消息才会丢失。并不会影响使用。

大约7分钟过后从节点可用。发送消息一切正常

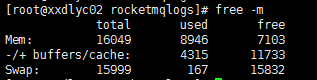
## 堆积两亿条数据启动时长二

### 测试场景

堆积212310228条数据。



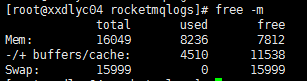
主节点内存



主节点硬盘



从节点内存



从节点硬盘



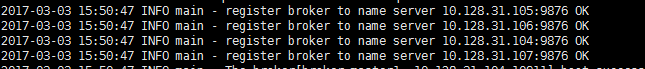
### 测试流程

1. 关闭节点写权限1分钟。
2. 关闭主从俩个节点。
3. 关闭进程
4. 备份Broker.log文件 为Broker\_backup.log。
5. 启动主节点，15分钟后启动从节点。
6. 发送程序，看程序是否有异常。
7. 查看新的Broker.log文件，计算启动时长

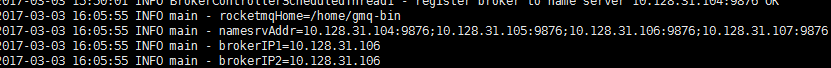
### 启动节点

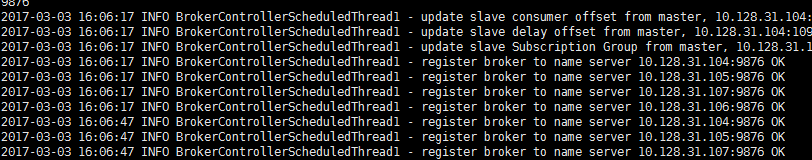
主节点：启动消耗时长为4秒



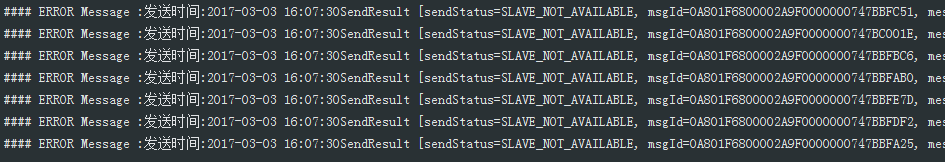


15分钟从节点：启动消耗时长20秒

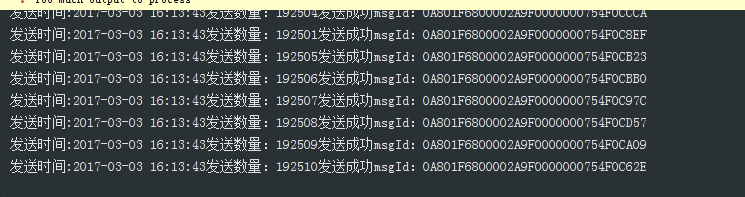




开始发送消息



在过6分钟过后正常



### 结论

* 在堆积2亿+条数据的情况下。数据大小为30个g，主节点和从节点启动时长分别为13秒和30秒。加上注册刷新路由30秒。如果主从节点同时启动保守估计2分钟内可加入集群。
* 但是启动主，15分钟后启动从同时加入集群时，立马发送消息，会出现SLAVE\_NOT\_AVAILABLE发送错误

该错误官方解释为：消息发送成功,但是此时 slave 不可用,消息已经进入服务器队列,只有此时服务器宕机,消息才会丢失。并不会影响使用。

大约6分钟过后从节点可用。发送消息一切正常。

# 整体结论

进过两轮的测试，可以看出两次除了发送消息数量的大小意外，扩容并没有对集群有很大影响。结论如下：

* 集群升级后新的节点可以自动加入集群，整个集群为可用状态状态。
* 在400w+条消息堆积的情况下，同时启动升级的主从节点，消息可以立即发送，这时消息不会同步到从节点上，从启动到完全正常，耗时约2分钟。保守估计需要5分钟的时间
* 在1100w+条消息堆积的情况下，同时启动升级的主从节点，消息可以立即发送，这时消息不会同步到从节点上，从启动到完全正常，耗时约2分钟。保守估计需要5分钟的时间
* 在2亿条消息堆积的情况下，同时启动升级的主从节点，消息可以立即发送，这时消息不会同步到从节点上，从启动到完全正常，耗时约7分钟。保守估计需要15分钟的时间
* 在2亿条消息堆积的情况下，先启动主节点，15分钟后启动从节点，消息可以立即发送，这时消息不会同步到从节点上，从启动到完全正常，耗时23分钟。保守估计需要25分钟的时间。

反之，先启动从节点，15分钟过后启动主节点。效果相同。

* 消息堆积越多，从启动到运行完全正常耗时越长。
* 不管是先起主节点还是从节点，随着堆积数目的增大，从节点不可用时间变长。
* 小数量的消息堆积对启动时间影响不大