1. 简介

此分布式消息队列支持消息的多topic，并且利用zookeeper实现了分布式集群。

1. 设计思路

系统架构图如下：



设计思路如下:

1. Zookeeper：负责维护以下信息：

* mq节点信息：结点数据【sid】:【 start\_time】,子节点:sid(该mq同步到的binlog-sid)。
* 消息topic：topic列表。
* source信息: source位于某个group的某个topic下，source存储的sid表示订阅该group，消息类型为topic同步的消息sid。

1. mq集群：

* 分为单机版和多机版。单机版mq见上版设计文档。多机版设计如下：
* mq按照组进行划分，每个mq实例属于某一个组。
* 每个组由一台master，多台salve组成。Master负责消息的接收与同步，slave只负责消息的同步。
* 当master宕机以后,sid最大的slave会自动升级为master来接收消息。Slave采用级联的方式进行消息同步。

1. Proxy:

* 连接zookeeper监控各个topic,以及各group的master节点的状态变化信息。
* Proxy分product\_proxy和subscribe\_proxy。其中product\_proxy负责消息发送，subscribe\_proxy负责消息订阅。
* Product\_proxy针对每个group-mq分别对应一个发送队列，消息将依次分法给对应mq。
* Recv\_proxy:从各group的master-mq中接收数据。由于mq需要负责更新各个source的同步sid信息，所以采用全部从master-mq中同步数据。

1. 系统架构

1. Zookeeper集群:
2. Zookeeper目录结构：

\---flatq # 根目录地址

|---group1 # mq组group1目录地址

| |---master:保存master-id信息(代理使用)

| |---mq：保存master-mq信息【master-id】:【新master最小sid】:【最新sid值】:【最新时间戳】

| | |---id-session\_id-seq ：格式【id】-【zk\_session\_id】-【zk\_seq】 存放数据：【配置信息】:【max\_sid】:【timestamp】 # mq实例（临时节点）

| | |---id-session\_id-seq ：格式【id】-【zk\_session\_id】-【zk\_seq】存放数据：【配置信息】:【max\_sid】:【timestamp】 # mq实例（临时节点）

| |---topic: topic目录节点【topic|max\_sid|timestamp|state:xxx】

| | |---topic1 ：保存【状态】topic状态0:正常 1:冻结

| | | |---source1：保存【sid】:【剩余消息数】:【更新时间】

| | |---topic2 ：保存【状态信息】

| | | |----souce1：保存【sid】:【剩余消息数】:【更新时间】

| |---deleted: 已经被删除的topic

| | |---topic3：被删除的topic

1. flatq/group1/mq/id-session\_id-seq：节点保存的【配置信息】:【sid】:【timestamp】,其中配置信息为本mq实例的recv/dispatch的监听端口。

【recv】

Ip=xxxx

Port=xxx

【inner\_dispatch】

Ip=xxxx

Port=xxxx

【outer\_dispatch】

Ip=xxx

Port=xxxx

1. Cwx\_mq/group1/topic/topic1:节点保存【状态】。格式为：

State=0/1

1. 单机版Mq流程
2. 架构图：



1. 业务流程：
2. 单机版mq分master/slave两种类型。
3. Master型mq启动时打开recv端口，负责消息的接收服务。
4. Slave型mq启动时与master-mq建立连接，从master中进行数据同步。
5. Masert/slave两种类型mq均提供数据同步功能。
6. 配置文件：

[cmn]

home=/usr/home/project/mq

server\_type=master/slave

sock\_buf\_kbyte=8192

max\_chunk\_kbyte=256

monitor=172.16.42.65:9900

sync\_conn\_num=10

[binlog]

path=/data2/mq\_data/data/binlog

file\_prefix=binlog

file\_max\_mbyte=1024

max\_file\_num=24

del\_out\_file=yes

cache=no

flush\_log\_num=1000

flush**\_**log\_second=1

[dispatch]

listen=\*:9903

source\_path=/data2/mq\_data/data/binlog/

source\_flush\_num=1

source\_flush\_second =10

[recv]

listen=\*:9901

[master]

listen=172.16.42.65:9903

zip=yes

1. 多机版Mq流程:
2. 架构图：



1. 业务流程
2. 启动时连接zookeeper注册组，节点信息。并监控/flat/【group】路径的变化。
3. 如为为master开始接收数据。如已存在master,则与master建立连接，开始数据的同步。
4. 监控到master宕机之后，选择sid最大的mq为master,新master mq开始接收数据，slave关闭旧Master连接开始从新Master同步数据。
5. 只有master可以接受数据。所有client端的source信息均保存到zookeeper的/cwx\_mq/group1/topic1/source1路径节点上。
6. 当监控到某个topic状态更改为冻结时，则master-mq开始拒绝该topic的接收。此时该topic数据同步仍可进行。
7. 线程细化架构图:



1. 配置文件：

[cmn]

home=/usr/home/project/mq

server\_type= zookeeper

sock\_buf\_kbyte=8192

max\_chunk\_kbyte=256

monitor=172.16.42.65:9900

sync\_conn\_num=10

id=mq1

[binlog]

path=/data2/mq\_data/data/binlog

file\_prefix=binlog

[dispatch]

listen=\*:9903

source\_flush\_num=1

source\_flush\_second =10

[recv]

listen=\*:9901

[zk]

server=ip:port,ip:port

auth=

root=/cwx\_mq

1. product\_Proxy:
2. 架构图：



1. 业务流程
2. 启动时与zk建立连接，获取各个group的master-mq的信息。形成group/topic映射关系表。
3. 根据各个group的master-mq信息，针对每个group分别启动一个线程并与各group的master-mq建立连接。
4. 接收用户发送数据，根据group/topic映射表将数据发送到对应连接上。
5. 如在指定的时间内mq没有返回结果信息则返回超时。
6. 如用户未指定group,并且需要发送的topic在多个group中均存在，那么将轮询的将消息发送个各个group中。
7. 配置文件：

[cmn]

home=/user/home/cwx\_proxy/

sock\_buf\_kbyte=8192

listen=host:port

conn\_num=3

send\_timeout=2

、

[zk]

Server=ip:port,ip:port

auth=

root=/cwx\_mq

1. subscribe\_proxy
2. 架构图：



1. 业务流程
2. 启动时与zk建立连接，获取并监控各个group的mq信息。
3. 接收用户订阅请求，启动新线程并根据zk中的信息与对应的mq建立连接。
4. 从mq中获取数据直接推送给用户。
5. 配置文件

[cmn]

home=/user/home/cwx\_proxy/

sock\_buf\_kbyte=8192

listen=host:port

sysn\_master=1/0

[zk]

Server=ip:port,ip:port

auth=

root=/cwx\_mq

1. Topic操作：
2. 新增topic：topic只有在zk中注册才可能进行消息的接收和发送。

#./zkclient.sh

#create /flatq/{group}/topic/{topic} 0

1. 更改topic状态: 连上zk修改topic信息

#./zkClient.sh

#delete /flatq/{group}/topic/{topic} 删除topic

#set /flatq/{group}/topic/{topoic} 1 冻结topic消息