BlotMQ-Store技术文档说明

针对版本V1.0.0

2017/11/21

BoltMQ-Store 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	周非	基础平台架构组	2017/11/21

目 录

1 存储

1.1 概述

存储模块主要包含存储Producer生产的消息、ConsumeQueue、索引等数据以及主从同步、刷盘、清理服务等。

1.2 零拷贝技术

零拷贝是通过将文件映射到内存上，直接操作文件，相比于传统的io(首先要调用系统IO，然后要将数据从内核空间传输到用户空间)，避免了很多不必要的数据拷贝，提高存储性能。

存储消息，使用了零拷贝，零拷贝包含以下两种方式：

方式	优点	缺点
mmap + write	即使频繁调用，使用小块文件传输，效率也很高	不能很好的利用 DMA 方式，会比 sendfile 多消耗 CPU，内存安全性控制复杂
sendfile	可以利用 DMA 方式，消耗 CPU 较少，大块文件传输效率高，无内存安全新问题	小块文件效率低于 mmap 方式

BlotMQ采用mmap+write方式，因为有小块数据传输的需求，效果会比sendfile更好。

1.3 CommitLog

CommitLog用于存储真实消息数据。CommitLog路径默认为用户工作目录/store/commitlog。

CommitLog存储目录结构：

commitlog

- 00000000000000000000

- 00000000001073741824

commitlog文件名生成的规则：

文件名的长度为 20 位，左边补零，剩余的为文件起始偏移量（第一个文件起始偏移量为 0）；

文件名字根据指定 commitlog 文件大小（默认文件大小为 1G，可以通过 MessageStoreConfig 的 mapedFileSizieCommitLog 进行配置）递增，文件大小单位为字节。

例如：

默认 commitlog 文件大小为 1G=1073741824b

第一文件的起始偏移量为 0，不足 20 位进行补零，故文件名 00000000000000000000，当第 一文件写满 ,第二文件的起始偏移量为 1073741824 ,不足 20 位进行补零 ,故文件名为 00000000001073741824，后面的文件名以此类推。

文件 n 起始偏移量 = size * (n- 1)

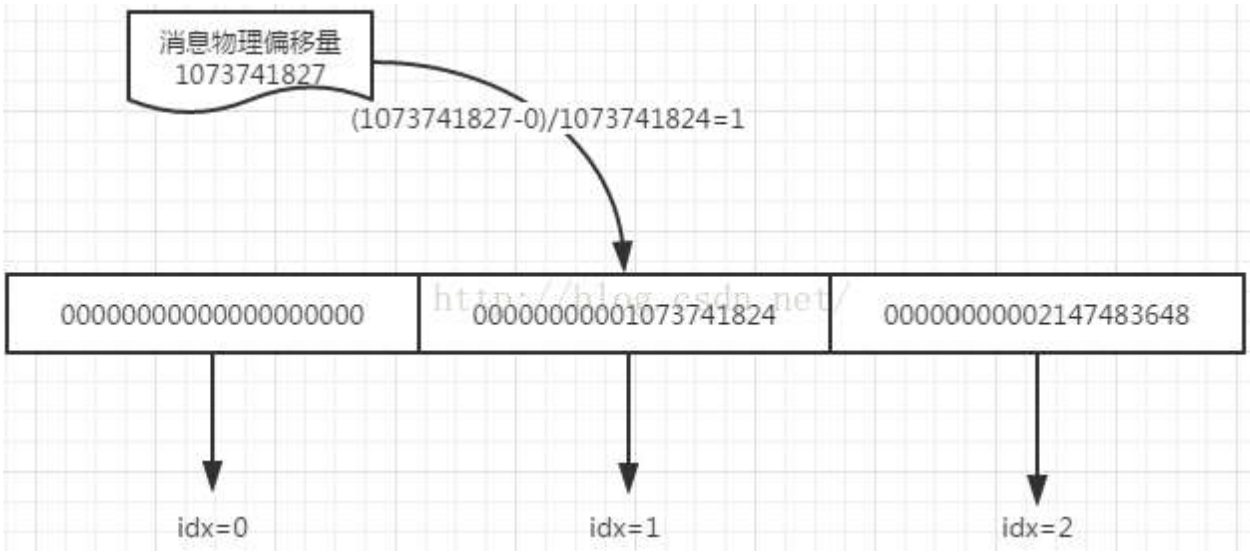
文件 1 起始偏移量 = 1073741824 * (1 - 1) = 0

文件 2 起始偏移量 = 1073741824 * (2 - 1) = 1073741824

通过 commitlog 文件名能够方便快速定位信息所在的文件。

文件 Index = (消息的起始物理偏移量-最早的文件的起始偏移量)/文件大小，

即 (1073741827-0)/1073741824=1，可得知该消息在队列中的第二个文件中：

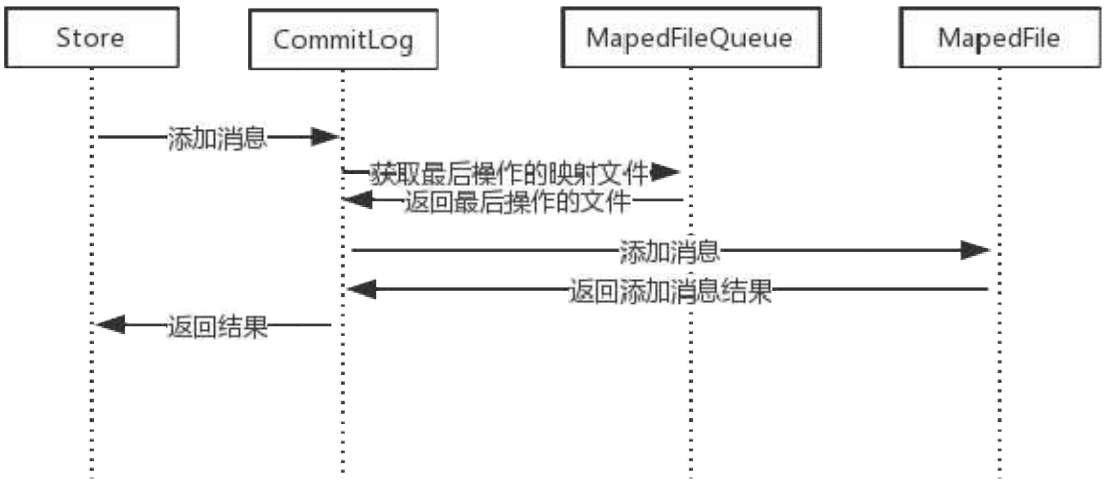


commitlog文件的消息结构：

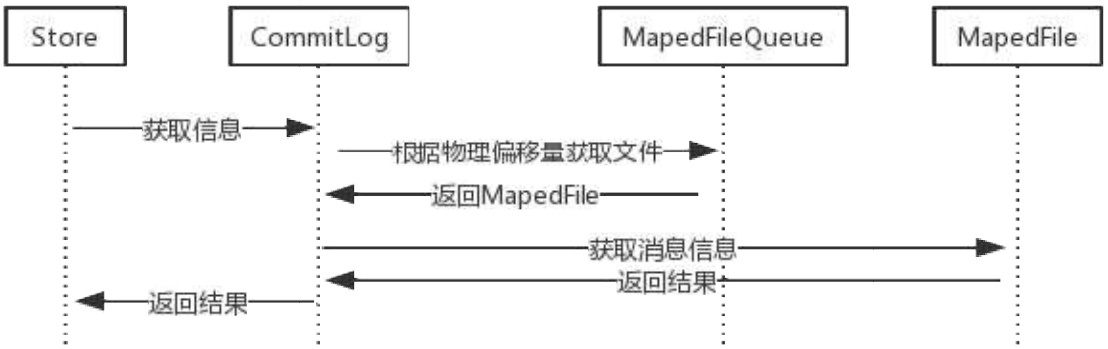
序号	字段	说明	字节数	备注

1	TotalSize	消息总长度	4	
2	MagicCode	MagicCode	4	MagicCode 分为：MessageMagicCode、BlankMagicCode。MessageMagicCode 表示正确的消息内容；BlankMagicCode 表示 CommitLog 文件空间不足，采用空字节占位写满文件。
3	BodyCRC	消息内容 CRC	4	BodyCRC 的值是对消息内容（body）进行 CRC32 生成的 32bit 冗余校验码，用于确保消息的正确性。
4	QueueId	消息队列编号	4	
5	Flag	消息标志	4	
6	QueueOffset	消息队列位置	8	自增值，消息队列逻辑位置，通过该值才能查找到 consume queue 中的数据；QueueOffset * 20 才是消息队列的物理偏移量。
7	PhysicalOffset	物理位置	8	
8	SysFlag	MessageSysFlag	4	
9	BornTimestamp	生产消息时间戳	8	
10	BornHost	生产消息的地址+端口	8	
11	StoreTimestamp	存储消息时间戳	8	
12	StoreHost	存储消息的地址+端口	8	
13	ReconsumeTimes	重新消费消息次数	4	
14	PreparedTransationOffset		8	
15	BodyLength	消息内容长度	4	
16	Body	消息内容	bodyLength	
17	TopicLength	Topic 长度	1	
18	Topic	topic	topicLength	
19	PropertiesLength	附加属性长度	2	
20	Properties	附加属性	propertiesLength	

添加 CommitLog 数据，将数据写入到 MapedFile，每个 MapedFile 对应着一个储存消息的二进制文件，MapedFile 在创建时会映射到内存上，添加消息时将需要保存的数据写入内存，后续有刷盘服务会将内存中数据持久化到二进制物理文件中，下图是添加 CommitLog 数据的主要业务流程：



查询 CommitLog 数据，直接从映射的内存中根据物理偏移量以及数据大小，获取数据，下图是查询 CommitLog 数据的主要业务流程：



1.4 **ConsumeQueue**

消费者逻辑队列，对应/store/consumequeue 文件夹，每个消费队列文件目录机构如下：

consumequeue

-- topic

-- queue id

-- 00000000000000000000

-- 00000000000000001040

-- 00000000000000002080

consumequeue文件名生成规则：

commitlog 文件名生成规则一致，需要注意的是：maped 文件大小为=向上取整（指定 size/消息位置信息 size）* 消息位置信息 size

例如：

指定消费队列文件大小=1024

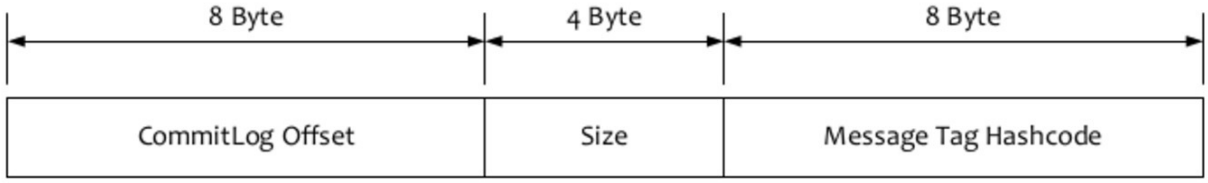
消息位置信息 size = 20

mapedFileSize = 向上取整（1024 / 20) * 20

mapedFileSize = 1040

consumequeue文件结构：

ConsumeQueue 中并不需要存储消息的内容，而存储的是消息在 CommitLog 中的 offset。也就是说，ConsumeQuue 其实是 CommitLog 的一个索引文件。

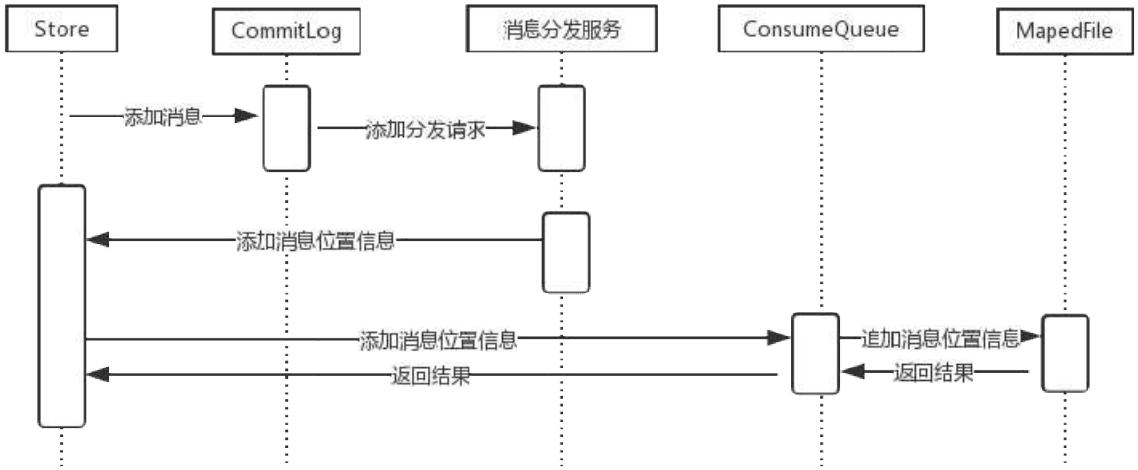


序号	字段	说明	字节数	备注
1	CommitLog Offset	CommitLog 的起始物理偏移量 physical offset	8	
2	Size	消息的大小	4	
3	Message Tag Hashcode	消息 Tag 的哈希值	8	用于订阅时消息过滤（订阅时如果指定了 Tag，会根据 HashCode 来快速查找到订阅的消息）

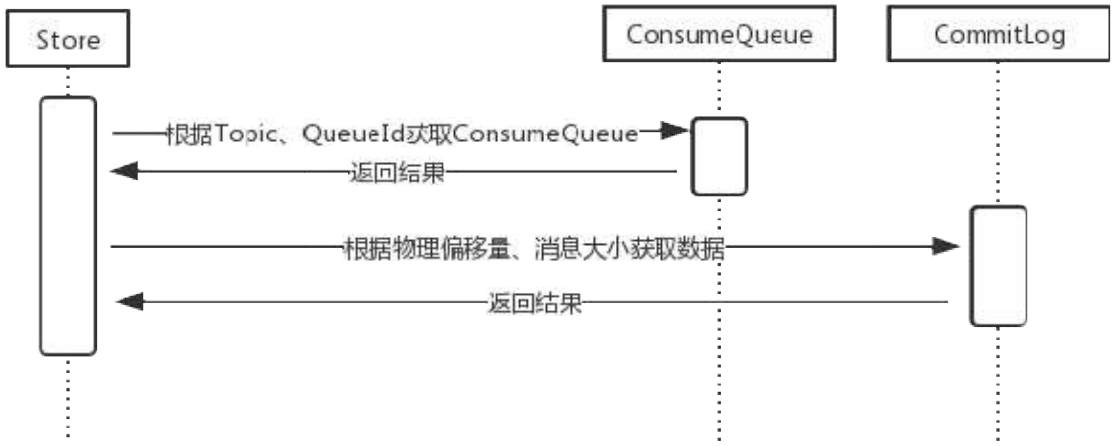
ConsumeQueue 是定长的结构，每条数据大小为 20 个字节，每个文件默认大小为 600 万个字节。

Consumer 消费消息的时候 ,需要 2 个步骤 :首先读取 ConsumeQueue 得到 offset ,然后读取 CommitLog 得到消息内容。

添加消息时，添加消息到 commitLog 后会向分发服务添加一个分发请求，分发服务调用 MessageStore 添加消息位置信息，根据消息的 Topic、QueueId 获取 ConsumeQueue，消息的位置信息追加到对应的消费队列中，最终保存的二进制文件中，主要流程如下图：



获取消息时， 首先根据 Topic、QueueId 获取 ConsumeQueue，然后根据消息逻辑 offset，获取消息的物理偏移量、消息的 Size，最后根据消息的物理偏移量、消息的 Size 获取 CommitLog 数据，主要流程如下图：



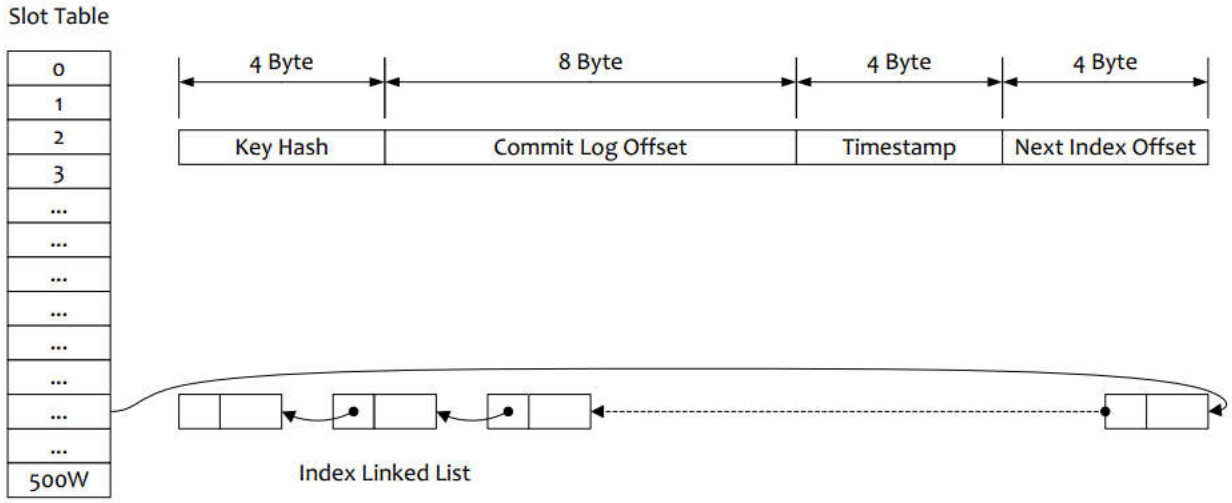
1.5 索引

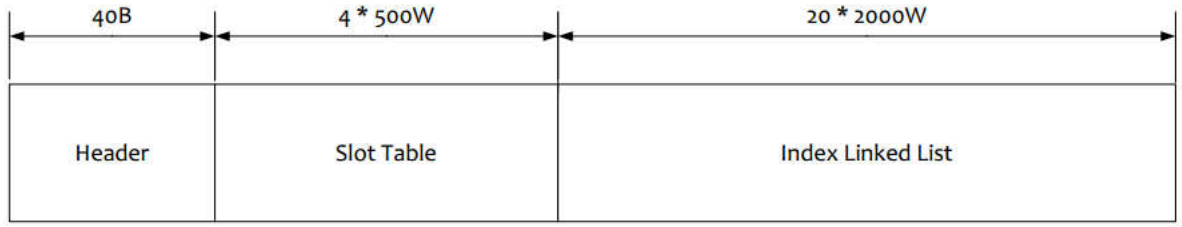
IndexService（索引服务）

IndexService 用于创建索引文件集合，当用户想要查询某个 topic 下某个 key 的消息时，能够快速响应。

Index File（索引文件）

IndexFile 存储消息索引的文件，文件结构如下：





索引文件由三个部分组成：Header（索引文件头信息）、Slot Table（槽位信息）、Index Linked List（消息的索引内容）

Index Header：索引文件头信息由 40 个字节的数据组成。

序号	字段	说明	字节数	备注
1	BeginTimestamp	索引文件开始时间	8	第一个索引创建的时间
2	EndTimestamp	索引文件结束时间	8	最后一个索引创建的时间
3	BeginPhyOffset	索引文件开始的物理偏移量	8	第一个索引对应的 CommitLog 物理偏移量
4	EndPhyOffset	索引文件结束的物理偏移量	8	最后一个索引对应的 CommitLog 物理偏移量
5	HashSlotCount	索引文件占用的槽位数	4	
6	IndexCount	索引的个数	4	

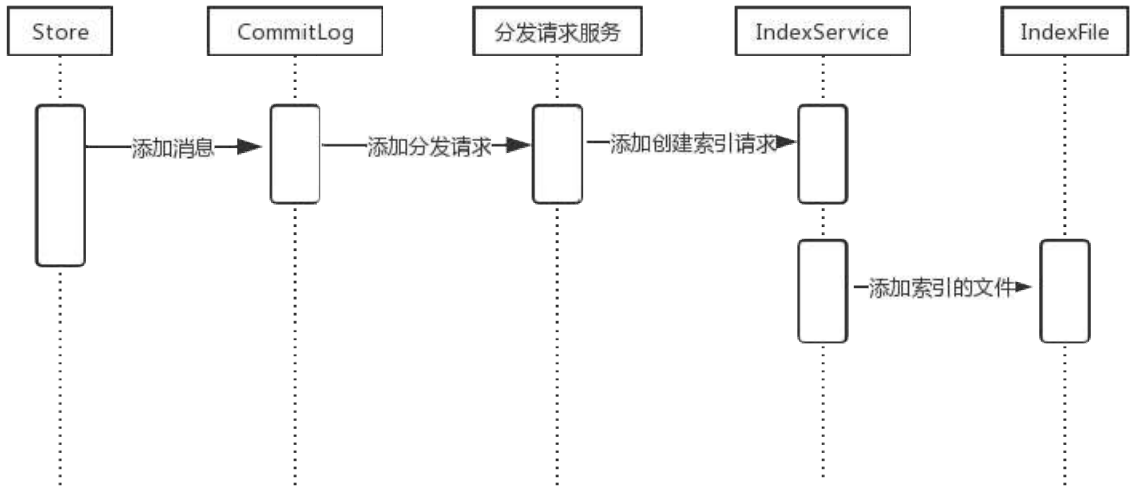
Slot Table

Index Linked List：消息的索引内容链表，默认每个文件有 2000W 消息索引内容组成，每个消息索引内容为 20 个字节的数据。

序号	字段	说明	字节数	备注
1	KeyHash	key 的哈希值	4	topic-key(key 是消息的 key)的 hashCode 组成
2	PhyOffset	commitLog 的物理偏移量	8	
3	Timestamp	索引创建的时间	4	
4	NextIndexOffset	下一个索引的索引地址	4	

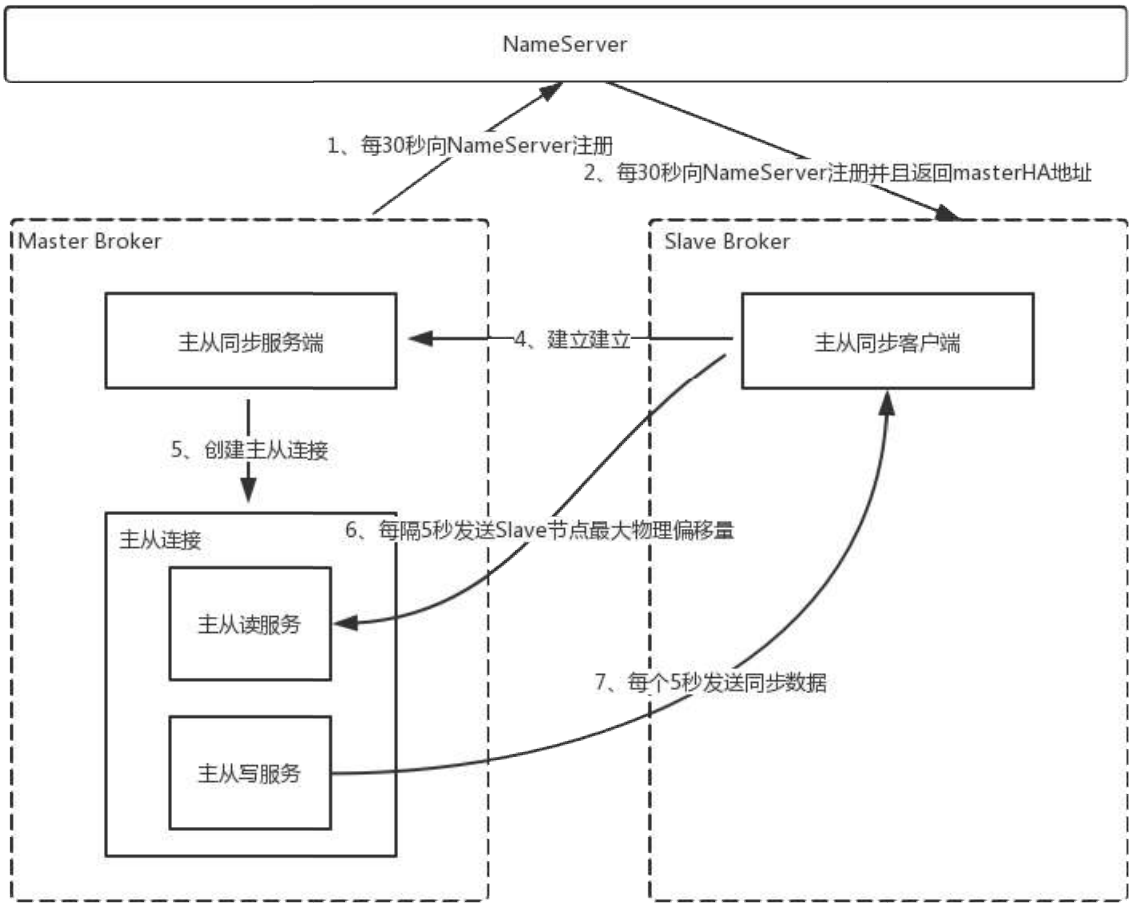
IndexFile 的创建过程：

首先在 DispatchMessageService 写入 ConsumeQueue 后，会再调用 indexService.putRequest，添加索引请求；IndexService 定时获取创建索引请求，调用 IndexService 的 buildIndex 进行创建索引。



1.6 主从同步

在集群模式的部署方式中，Master 与 Slave 配对是通过指定相同的 brokerName 参数来配对，Master 的 BrokerId 必须是 0，Slave 的 BrokerId 必须是大于 0 的数。一个 Master 下面可以挂载多个 Slave，同一个 Master 下的多个 Slave 通过指定不同的 BrokerId 来区分。



主从同步服务

存储模块启动时，会启动主从同步服务，主从同步服务主要的组成部分是：主从同步服务端、主从同步客户端

主从同步服务端

接收 slave 节点的连接请求，接收到请求后会建立主从连接，接受和传递主从之间数据。

主从连接

主从连接主要由主从写服务、主从读服务组成，主从写服务主要用于 master 传输同步数据，主从读服务主要用于接受 slave 节点发送的 offset 信息。

主从写服务传输的数据结构：

序号	字段	说明	字节数	备注
1	Offset	commitLog 物理偏移量	8	同步 commitLog 物理偏移量

2	BodySize	传输数据的大小	4	
3	BodyData	传输数据的内容	BodyLength	

主从同步客户端

连接 master 节点，定时上报 offset 以及接收 master 节点传输的同步数据。

主从同步客户端上报的数据结构：

序号	字段	说明	字节数	备注
1	Offset	commitLog 物理偏移量	8	slave 节点的最大 commitLog 物理偏移量

主从同步客户端上报 offset 时，会获取当前最大 CommitLog 文件物理偏移量。如果 HAClient 是首次上报 offset，并且上报的 offset 为 0，master 节点会获取最后一个 CommitLog 文件进行传输，其余的 CommitLog 文件不会进行同步。上报的 offset 不为 0，master 节点会从上报的 offset 进行同步。

1.7 刷盘

RocketMQ 刷盘有两种方式，分为：同步刷盘、异步刷盘。

同步刷盘：在消息到达 MQ 后，RocketMQ 需要将数据持久化，同步刷盘是指数据到达内存之后，必须刷到 commitlog 日志之后才算成功，然后返回 producer 数据已经发送成功。

异步刷盘：数据到达内存之后,返回 producer 说数据已经发送成功，然后再写入 commitlog 日志。

RocketMQ 默认是使用异步刷盘。

逻辑队列刷盘服务(FlushConsumeQueueService)：用于将 ConsumeQueue 的 File 文件写入入里磁盘，首先判断是否到达了刷盘时间，如果到达了，那么全盘通刷；否则，遍历所有的 ConsumeQueue，调用 cq.commit(flushConsumeQueueLeastPages)进行刷盘 ,flushConsumeQueueLeastPages 是目前文件的未刷盘大小达到 flushConsumeQueueLeastPages*OS_PAGE_SIZE(1024*4)个，才进行刷盘。

逻辑队列刷盘服务：定时将 ConsumeQueue 的数据从内存写入到文件。

1.8 文件清理

存储服务启动时，会启动定时清理文件服务，定时清除服务会每分钟定时清理 CommitLog、ConsumeQueue 文件。

清理CommitLog文件服务

清理 CommitLog 文件，需要满足以下任意一条件：

1、消息文件过期（默认 72 小时），且到达清理时点（默认是凌晨 4 点），删除过期文件。

2、消息文件过期（默认 72 小时），且磁盘空间达到了水位线（默认 75%），删除过期文件。

3、磁盘已经达到必须释放的上限（85%水位线）的时候，则开始批量清理文件（无论是否过期），直到空间充足。

注：若磁盘空间达到危险水位线（默认 90%），出于保护自身的目的，broker 会拒绝写入服务。

清理ConsumeQueue文件服务

定时清理小于最小 CommitLog 物理偏移量的 ConsumeQueue 的文件。

附件一 BoltMQ 开发者联系方式

姓名	联系方式	更新日期
郜焱磊	gyl_adaihao@163.com	2017/11/21
田玉粮	idistyl@gmail.com	2017/11/21
尹同强	tongqiangyin@gmail.com	2017/11/21
罗继	gunsluo@gmail.com	2017/11/21
周飞	he236555699@163.com	2017/11/21
戎志宏	hzrong007@163.com	2017/11/21