Rocket MQ初步认知（一）

> 学习背景：前端开发工程师，懂一点java，项目要求接触RocketMQ，本文为初步认知
>
> 作者：Nick

RocketMQ是一个消息中间件。其产生的动机在github官方文档有如下阐述：

> 在早期阶段，我们在ActiveMQ 5.x（小于5.3）的基础上构建了分布式消息传递中间件。我们的国际业务使用它来进行异步通信，搜索，社交网络活动流，数据管道，甚至在我们的交易订单流程中。随着我们的贸易业务吞吐量越来越不可思议，来自我们的消息传送群集的压力也变得越来越明显。

> 基于我们的观察和研究，随着越来越多的队列和虚拟主题的使用，ActiveMQ IO模块成为瓶颈。在某些情况下，较慢的消费者可能会减慢生产者。我们尽最大努力通过节流，断路器或降级来处理这个问题，但它不能优雅地扩展。所以我们开始专注于当时流行的消息传递解决方案Kafka。不幸的是，Kafka不能满足我们的要求，如低延迟和高可靠性，详见这里。

> 在这种情况下，我们决定创新一个新的消息中间件，以处理一系列广泛的用例，从传统的发布/订阅场景到苛刻的大容量实时事务系统，不容许消息丢失。我们还创建了一个基于RocketMQ的基础产品，这是一个名为阿里云平台的平台即服务（PaaS）产品。今天，超过100家公司在他们的业务解决方案中使用RocketMQ开源版本。我们相信RocketMQ可以让更多的人受益，所以我们想在世界各地分享。

阐述中提到的kafka,ActiveMQ以及我们将要研究的RocketMQ都是消息中间件，何是消息中间件？

消息中间件

- 理解：
  - 消息中间件利用高效可靠的消息传递机制进行平台无关的数据交流，分布式系统中重要的组件，主要解决应用耦合，异步消息，流量削锋等问题。实现高性能，高可用，可伸缩和最终一致性架构。是大型分布式系统不可缺少的中间件。
  - 传统消息系统数据有较大丢失风险，消息中间件最大的作用是解耦系统之间的依赖及异步化各系统间的调用。且作为系统间消息存储和转发环节，具有高可靠性。
- 特点：
  - 异步处理：发/接双方不需同时在线，消息可以分发，可以堆叠。
  - 解耦合：防止引入过多的API给系统的稳定性带来风险；调用方使用不当会给被调用方系统造成压力，被调用方处理不当会降低调用方系统的响应能力。

理解完消息中间件，我们来看一下RocketMQ的特点：

Rocket MQ 特点：

官方解释：

> 产品基于高可用分布式集群技术，提供消息发布订阅、消息轨迹查询、定时（延时）消息、资源统计、监控报警等一系列消息云服务，是企业级互联网架构的核心产品

这里提到集群和分布式，再理解下：

分布式和集群

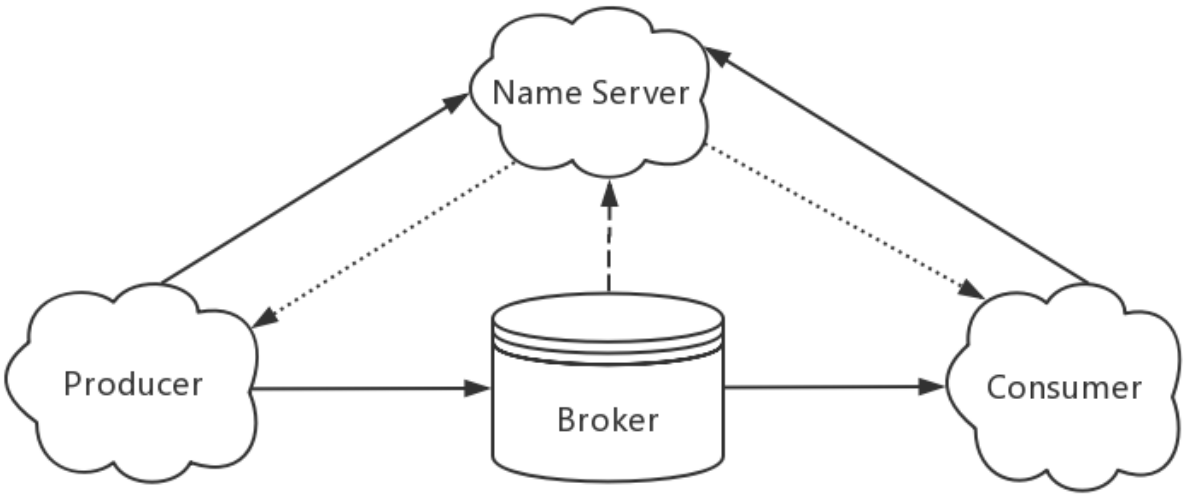
- 概念：
  - 分布式：一个业务分拆多个子业务，部署在不同的服务器上。
  - 集群：同一个业务，部署在多个服务器上，将几台服务器集中在一起，实现同一业务
- 区别

  分布式的每一个节点，都完成不同的业务，一个节点垮了，这个节点的业务就不可访问了。而集群，有一个组织性，一台服务器垮了，其它的服务器可以顶上来。
  所以说，分布式中的每一个节点，都可以做集群。而集群并不一定就是分布式的

> 举例：如新浪网，访问的人多了，他可以做一个集群，前面放一个响应服务器，后面几台服务器完成同一业务，如果有业务访问的时候，响应服务器看哪台服务器的负载不是很重，就将给哪一台去完成。

到这里，对于RocketMQ是个什么东西有一个模糊的理解，在具体实践前我们需要了解一些概念（我仅介绍单机搭建时会涉及的知识点）

Rocket MQ架构及术语理解



图中有四个角色，也是Rocket MQ中的重要概念。

术语概念

- Producer
  - 消息生产者，负责产生消息，一般由业务系统负责产生消息。
- Consumer
  - 消息消费者，负责消费消息，一般是后台系统负责异步消费。
- NameServer
  - 无状态节点，用来保存活跃的 broker 列表，和topic列表。
- Broker
  - 消息中转角色，负责存储消息，转发消息。

producer和consumer很好理解一个发送消息，一个接受消息。nameserver和broker是重点要说的，它们是服务器层，构成了集群的逻辑。这里先解释一个概念。

> 无状态节点
> 无状态和状态的区别，即为服务器应答过程中是否基于上次请求所构筑的上下文环境，个人理解：无状态就是发送过来的请求结果都一样，和上一次没啥因果关系。有状态就是我这一次发的请求和上一次有一定关系，服务器得保留每次的请求，以便对下一次请求产生影响

nameserver是无状态节点，所以可以扩展成集群。

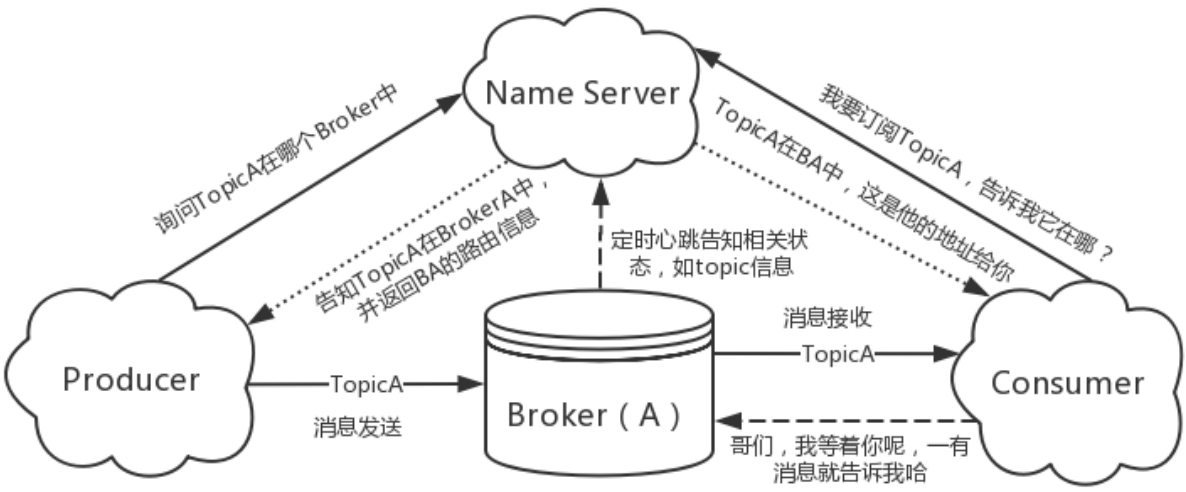
- topic
    - 消息的逻辑管理单位。

实际生产模式下（关闭了broker的自动创建、订阅topic的功能）

- broker控制topic的增删，发送未知topic的producer会报错（autoCreateTopicEnable=false）
- broker控制topic的订阅，未指定的订阅组，consumer无法拉取消费该topic的信息（autoCreateSubscriptionGroup=false ）

工作流程

1. 首先需要搭建一个NameServer节点，就像一个信号塔，producer，consumer以及broker之间的联系都需要通过访问NameServer，获取到各自需要的路由信息后才能进行联系。
2. NameServer节点搭建成功后，启动Broker。启动前可以修改Broker的默认配置文件，包括前文说到的topic相关的参数等，修改完成后，启动Broker，配置即生效。
3. 启动完NameServer和Broker后，可以在控制台里进行topic的增加和订阅（具体控制台的操作后文再说）。然后producer就可以发送已有的topic类型的消息，另一边consumer（订阅组）被broker准许后，每当consumer订阅的topic下接收到消息（消息从producer发送到broker中），consumer便可以成功拉取到此消息（消息从broker到consumer）并进行消费。



该段简要的描述了RocketMQ工作的流程，更多更复杂的概念咱们后文再涉及...

下一篇：Rocket MQ单机环境搭建