Kafka

A distributed streaming platform

# 概述

Kafka 有三种模式，PUBLISH & SUBSCRIBE、PROCESS、STORE

▪ PUBLISH & SUBSCRIBE

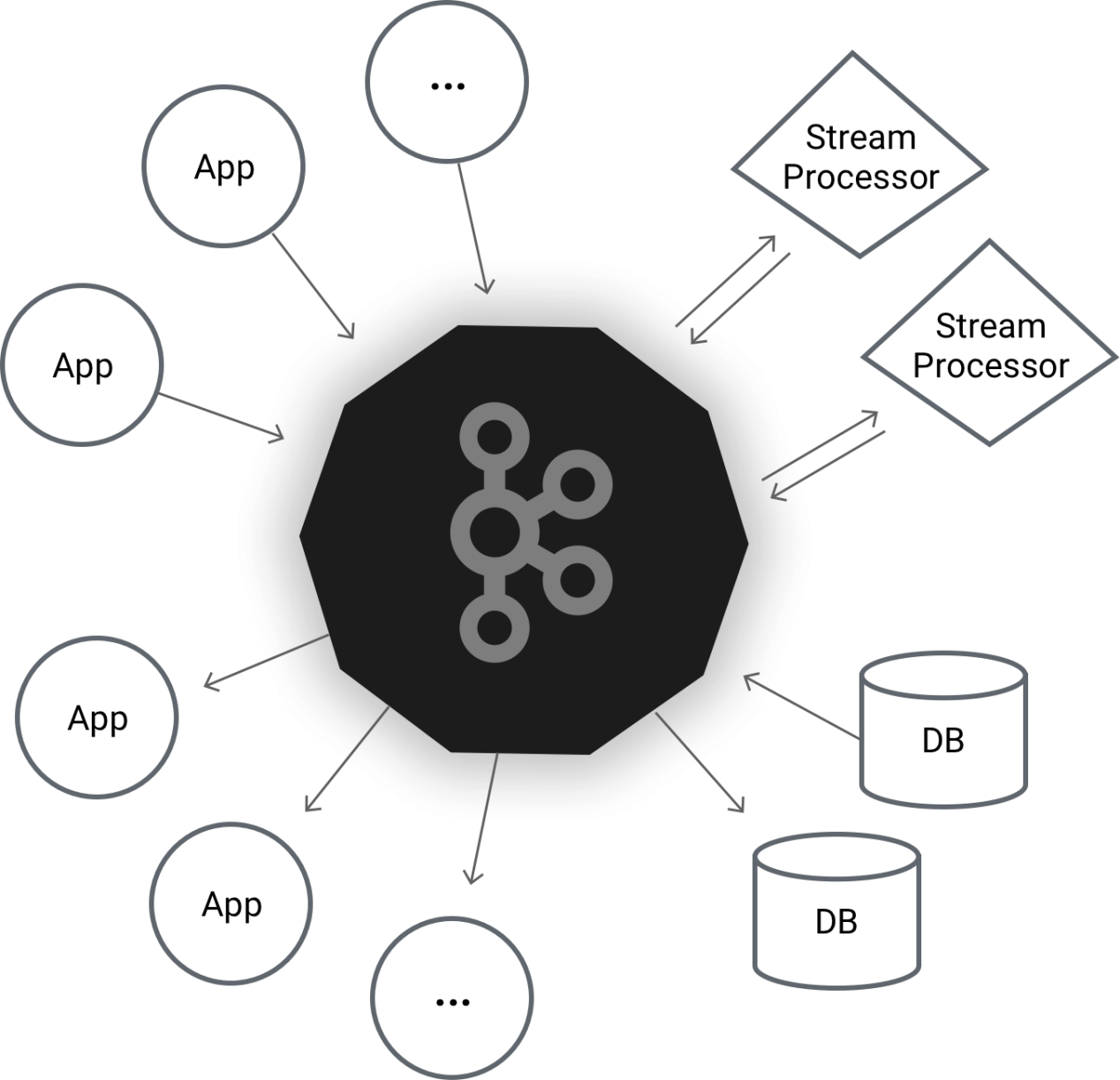
和其他消息发布订阅系统类似，例如(ActiveMQ,RabbitMQ etc.);

▪ PROCESS

实时流处理;

▪ STORE

分布式数据存储。



Kafka 用户构建实时数据管道和数据流应用程序。它具有水平可扩展性，容错性，快速性，并在数千家公司的生产中运行。

# 简介

ApacheKafka是一个分布式流媒体平台。这到底是什么意思呢？

流媒体平台有三个关键功能：

▪ publish & subscribe，类似于消息队列或企业消息传递系统;

▪ 以容错的可持久化的方式存储数据;

▪ 实时处理。

Kafka通常用于两大类应用：

▪ 构建可在系统或应用程序之间可靠获取数据的实时流数据管道;

▪ 构建转换或相应数据流的实时流应用程序。

首先了解几个概念：

▪ Kafka作为一个集群运行在一个或多个可跨多个数据中心的服务器上;

▪ Kafka集群以成为topic的类型存储数据流;

▪ 每条记录都包含一个key，一个key和一个时间戳。

Kafka有四个核心API：

▪ Producer API：

允许应用发布数据流到一个后多个kafka topics;

▪ Consumer API:

允许一个应用订阅一个或多个topics并处理来自发布者的数据流;

▪ Streams API：

允许应用充当流处理器，消费来自一个或多个topics的输入流并生生成到一个或多个输出topics的输出流，从而有效地将输入流转换为输出流;

▪ Connector API：

允许构建和运行将Kafka topics连接到现有应用或数据系统的可重用生产者或消费者。例如，关系数据库的连接器可能捕获对表的每个更改。

在Kafka中，客户端和服务器之间的通信是通过简单，高性能，语言无关的TCP协议完成的。此协议向后兼容。我们为Kafka提供Java客户端，单客户端有多种语言版本。

## Topics and Logs

Topic: Kafka为记录数据流的核心抽象。

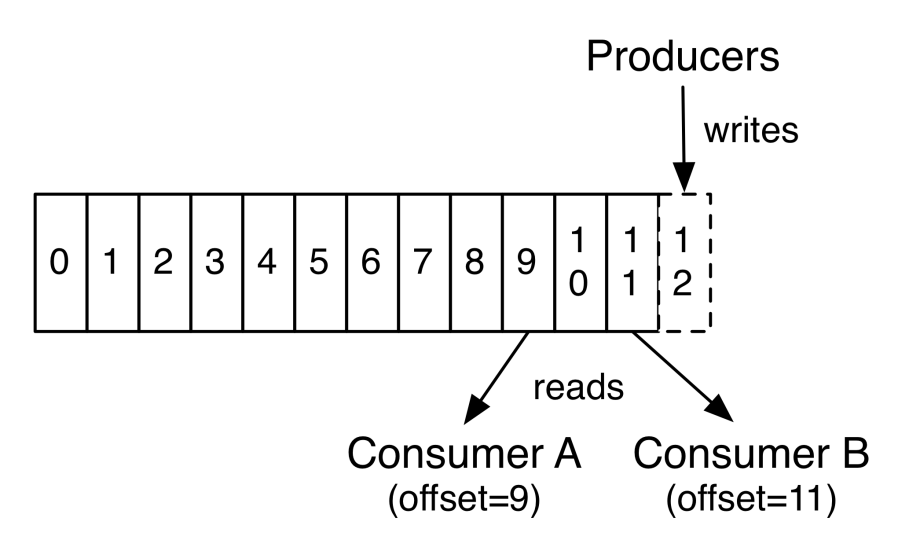
Topic是发布数据的类别或订阅源名称。Kafka的topic总是多用户；也就是说，一个topic可以有0个，1个或多个消费者订阅写入数据。

对于每个topic，kafka集群都维护一个如下所示的分区日志：



每个 Partition 都是一个有序的，不可变得记录序列，不断附加到结构化的提交日志中。分区中的记录每个都分配了一个称为偏移量的顺序ID号，它唯一地标识分区中的每个记录。

Kafka集群将所有记录持久化-无论是否已使用-使用可配置的保存策略。例如，如果保留策略设置为两天，则在发布记录后的两天内，它可供使用，之后将被丢弃以释放空间。Kafka的性能在数据大小方面实际上是恒定的，因此长时间存储数据不是问题。



实际上，基于每个消费者保留的唯一元数据是该消费者在日志中的偏移量或位置。这种偏移量由消费者控制：通常消费者在读取记录时会线性地提高其偏移量，但事实上，由于 该位置由消费者控制，因此它可以按照自己喜欢的任何顺序消费记录。例如，消费者可以重置为较旧的偏移量来重新处理过去的数据，或者跳到最近的记录并从“现在”开始消费。

这些功能组合意味着Kafka消费者非常廉价-它们可以来来往往对集群或其它消费者没有太大影响。例如，你可以使用我们的命令行工具“tail”任何topic内容，而无需更改任何现有消费者所消费的内容。

日志中的分区有多重用途。首先，它们允许日志扩展到超出合适单个服务器的大小。每个单独的分区必须适合托管它的服务器，单topic可能有许多分区，因此它可以处理任意数量的数据。其次，它们充当了并行性大单位-更多的是bit。

## Distribution

日志的分区分布在kafka集群中的服务器上，每个服务器处理数据并请求共享的分区。每个分区都在可配置数量的服务器上进行复制，以实现容错。

每个分区都有一个服务器充当“leader”，0个或多个服务器充当“followers”。Leader处理分区的所有读取和写入请求，而followers被动地复制leader。如果leader失败，其中一个followers将自动成为新的leader。每个服务器都充当其某些分区的leader和其他服务器的followers，因此负载在集群中得到很好的平衡。

## Geo-Replication

Kafka MirrorMaker为你的集群提供geo-replication支持。使用MirrorMaker，可以跨多个数据中心或云区域复制数据。你可以在active/passive中使用它进行备份和恢复；或者在active/active方案中，将数据放置在离用户较近的位置，或支持数据位置的要求。

## Producers

生产者将数据发布到他们选择的topic。生产负责选择分配给topic中哪个分区的记录。这可以以循环方式完成，仅仅是为了平衡负载，或者可以根据一些语义分区功能(例如基于记录中的某些key)来完成。更多关于在一秒钟内使用分区的信息。

## Consumers

消费者使用消费者组名称标记自己，并且发布到topic的每个记录被传递到每个订阅消费者组中的一个消费者实例。消费者实例可以在单独的进程中，也可以在不同的机器。

如果所有消费者实例具有相同的消费者组，则记录将有效地在使用者实例上进行负载平衡。

如果所有消费者实例具有不同的消费者组，则每个记录将广播到所有消费者进程。



两个服务器kafka集群，托管四个分区(P0-P3)，包含两个消费者组。消费者组A有两个消费者实例，B组有四个消费者实例。

然而，更常见的是，我们发现topic具有少量的消费者群体，每个“logical subscriber”一个。每个组由许多用于可伸缩性和容错的消费者实例组成。这只不过是publish-subscribe语义，其中订阅者是消费者集群而不是单个进程。

在kafka中实现消费的方式是通过在消费者实例上划分日志中的飞去，以便每个实例在任何时间点都是分配的“fair share”的独立消费者。维护组中成员资格的过程由kafka协议动态处理。如果新实例加入该组，它们将从该组的其它成员接管一些分区；如果实例死亡，其分区将分发给其余实例。

Kafka仅提供分区内记录的排序，而不是topic中不同分区之间的记录。对于大多数应用程序而言，按分区排序与按key分区数据的能力相结合就足够了。但是如果你需要对记录进行总排序，则可以使用仅包含一个分区的topic来实现，但这意味着每个消费者组只有一个消费者进程。

## Multi-tenancy

你可以将kafka部署为multi-tenant解决方案。通过配置哪些topic可以生成或使用数据来启用multi-tenancy。同时也支持quotas。管理员可以定义和枪支执行quotas，以控制客户端使用的代理资源。有关更多信息，清参与安全性文档([security documentation](https://kafka.apache.org/documentation/" \l "security))。

## Guarantees

在high-level提供一下保证：

▪ 生产者发送到特定topic分区的消息将按其发送顺序追加。也就是说，如果记录M1由与记录M2相同的生产者发送，并且首先发送M1，则M1将具有比M2更低的偏移量并且在日志中更早出现。

▪ 消费者实例按照它们存储在日志中的顺序查看记录。

▪ 对于具有复制因子N的topic，我们将容忍最多N-1个服务器故障，而不会丢失任何提交到日志的记录。

## Kafka as a Messageing System

Kafka 的流概念与传统的企业消息系统相比如何？

消息系统上有两种模型：queue和publish-subscribe。在队列中，消费者池可以从服务器读取并且每个记录转到其中一个；在发布-定于中，记录被广播给所有消费者。这两种模型中的每一种都有优点和缺点。队列的优势在于它允许你在多个消费者实例上划分数据处理，从而可以扩展你的处理。不行的是，一旦一个进程读取它已经失效的数据，队列就不是多用户。发布-定于允许你将数据广播到多个进程，但由于每条消息都发送给每个订阅者，因此无法进行扩展处理。

Kafka的消费组集群概念扩展了这两个概念。与队列一样，消费者组允许你将处理划分为一组进程(消费者组的成员)。与发布-订阅一样，kafka允许你向多个消费者组广播消息。

Kafka模型的优势在于每个topic都具有这些属性-它可以扩展处理并且也是 multi-subscriber-不需要选择其中任何一个。

与传统的消息系统相比，kafka具有更强的顺讯保证。

传统队列在服务器上按顺序保留记录，如果多个消费者从队列中消费，则服务器按照存储顺序分发记录。但是，虽然服务器按顺序分发记录，但是记录时异步传递给消费者的，因此他们可能会在不同的消费者处出现故障。这实际上意味着在存在并行消费的情况下丢失记录的顺序。消息传递系统通常同构具有“exclusive consumer”概念来解决这个问题，该概念只允许一个进程从队列中消费，单当然这意味着处理中没有并行性。

Kafka做的更好。通过在topic中具有并行性概念-分区，kafka能够在消费者流程池中提供顺序保证和负载均衡。这是通过将topic中的分区分配给消费者组中的一个消费者使用。通过这样做，我们确保消费者是该分区的唯一读者并按顺序使用数据，由于有许多分区，这仍然可以平衡许多消费者实例的负载。但请注意，消费组组中的消费者实例不能超过分区。

## Kafka as a Storage System

任何允许发布与消费消息分离的消息的消息队列实际上充当了正在进行的消息的存储系统。Kafka的不同之处在于它是一个非常好的存储系统。

写入kafka的数据将写入磁盘并进行复制以实现容错。Kafka允许生产者等待确认，以便在完全复制之前写入不被认为是完整的，并且即使写入的服务器失败也保证写入仍然存在。

磁盘结构kafka很好地使用了可伸缩性-无论服务器上有50KB还是50TB的持久数据，kafka都会执行相同的操作。

由于认真对待存储并允许客户端控制器读取位置，你可以将kafka视为一种专用于高性能，低延迟提交日志存储，复制和传播的专用分布式文件系统。

有关kafka的提交日志存储和复制设计的详细信息，请阅读此页面([this](https://kafka.apache.org/documentation/" \l "design))