技术翻译

已有文章 2384 篇
当前位置： 译文列表 » 服务器端开发 ， 投递原文

在 2384 篇翻译的文章中搜索 搜 索

分布式发布订阅消息系统 Kafka 架构设计

53
顶

英文原文：Kafka Architecture Design
标签： Kafka
帝都老白 推荐于 4年前 （共 48 段，翻译完成于 03-08）（25评） 369人收藏此文章， 我要收藏

参与翻译(4人)： fbm, 我不是会员, K6F, nesteaa

仅中文 | 中英文对照 | 仅英文 | 打印此文章

我们为什么要搭建该系统

Kafka是一个消息系统，原本开发自LinkedIn，用作LinkedIn的活动流（activity stream）和运营数据处理管道（pipeline）的基础。现在它已为多家不同类型的公司 作为多种类型的数据管道（data pipeline）和消息系统使用。

活动流数据是所有站点在对其网站使用情况做报表时要用到的数据中最常规的部分。活动数据包括页面访问量（page view）、被查看内容方面的信息以及搜索情况等内容。这种数据通常的处理方式是先把各种活动以日志的形式写入某种文件，然后周期性地对这些文件进行统计分析。运营数据指的是服务器的性能数据（CPU、IO使用率、请求时间、服务日志等等数据）。运营数据的统计方法种类繁多。

近年来，活动和运营数据处理已经成为了网站软件产品特性中一个至关重要的组成部分，这就需要一套稍微更加复杂的基础设施对其提供支持。

翻译的不错

fbm 顶 哦！ 4年前

9人顶

活动流和运营数据的若干用例

- "动态汇总（News feed）"功能。将你朋友的各种活动信息广播给你
- 相关性以及排序。通过使用计数评级（count rating）、投票（votes）或者点击率（ click-through）判定一组给定的条目中那一项是最相关的.
- 安全：网站需要屏蔽行为不端的网络爬虫（crawler），对API的使用进行速率限制，探测出扩散垃圾信息的企图，并支撑其它的行为探测和预防体系，以切断网站的某些不正常活动。
- 运营监控：大多数网站都需要某种形式的实时且随机应变的方式，对网站运行效率进行监控并在有问题出现的情况下能触发警告。
- 报表和批处理：将数据装载到数据仓库或者Hadoop系统中进行离线分析，然后针对业务行为做出相应的报表，这种做法很普遍。

翻译的不错

fbm 顶 哦！ 4年前

6人顶

活动流数据的特点

这种由不可变（immutable）的活动数据组成的高吞吐量数据流代表了对计算能力的一种真正的挑战，因其数据量很容易就可能会比网站中位于第二位的数据源的数据量大10到100倍。

传统的日志文件统计分析对报表和批处理这种离线处理的情况来说，是一种很不错且很有伸缩性的方法；但是这种方法对于实时处理来说其时延太大，而且还具有较高的运营复杂度。另一方面，现有的消息队列系统（messaging and queuing system）却很适合于在实时或近实时（near-real-time）的情况下使用，但它们对很长的未被处理的消息队列的处理很不给力，往往并不将数据持久化作为首要的事情考虑。这样就会造成一种情况，就是当把大量数据传送给Hadoop这样的离线系统后， 这些离线系统每个小时或每天仅能处理掉部分源数据。Kafka的目的就是要成为一个队列平台，仅仅使用它就能够既支持离线又支持在线使用这两种情况。

Kafka支持非常通用的消息语义（messaging semantics）。尽管我们这篇文章主要是想把它用于活动处理，但并没有任何限制性条件使得它仅仅适用于此目的。

翻译的不错

fbm 顶 哦！ 4年前

7人顶

部署

下面的示意图所示是在LinkedIn中部署后各系统形成的拓扑结构。

要注意的是，一个单个的Kafka集群系统用于处理来自各种不同来源的所有活动数据。它同时为在线和离线的数据使用者提供了一个单个的数据管道，在线活动和异步处理之间形成了一个缓冲区层。我们还使用kafka，把所有数据复制（replicate）到另外一个不同的数据中心去做离线处理。

我们并不想让一个单个的Kafka集群系统跨越多个数据中心，而是想让Kafka支持多数据中心的数据流拓扑结构。这是通过在集群之间进行镜像或“同步”实现的。这个功能非常简单，镜像集群只是作为源集群的数据使用者的角色运行。这意味着，一个单个的集群就能够将来自多个数据中心的数据集中到一个位置。下面所示是可用于支持批量装载（batch loads）的多数据中心拓扑结构的一个例子：

请注意，在图中上面部分的两个集群之间不存在通信连接，两者可能大小不同，具有不同数量的节点。下面部分中的这个单个的集群可以镜像任意数量的源集群。要了解镜像功能使用方面的更多细节，请访问这里.

翻译的不错

fbm 顶 哦！ 4年前

6人顶

主要的设计元素

Kafka之所以和其它绝大多数信息系统不同，是因为下面这几个为数不多的比较重要的设计决策：

1. Kafka在设计之时为就将持久化消息作为通常的使用情况进行了考虑。
2. 主要的设计约束是吞吐量而不是功能。
3. 有关哪些数据已经被使用了的状态信息保存为数据使用者（consumer）的一部分，而不是保存在服务器之上。
4. Kafka是一种显式的分布式系统。它假设，数据生产者（producer）、代理（brokers）和数据使用者（consumer）分散于多台机器之上。

以上这些设计决策将在下文中进行逐条详述。

翻译的不错

fbm 顶 哦！ 4年前

4人顶

基础知识

翻译的不错

fbm 顶 我! 4年前

6人顶

首先来看一些基本的术语和概念。

消息指的是通信的基本单位。由消息生产者（producer）发布关于某话题（topic）的消息，这句话的意思是，消息以一种物理方式被发送给了作为代理（broker）的服务器（可能是另外一台机器）。若干的消息使用者（consumer）订阅（subscribe）某个话题，然后生产者所发布的每条消息都会被发送给所有的使用者。

Kafka是一个显式的分布式系统 —— 生产者、使用者和代理都可以运行在作为一个逻辑单位的、进行相互协作的集群中不同的机器上。对于代理和生产者，这么做非常自然，但使用者却需要一些特殊的支持。每个使用者进程都属于一个使用者小组（consumer group） 。准确地讲，每条消息都只会发送给每个使用者小组中的一个进程。因此，使用者小组使得许多进程或多台机器在逻辑上作为一个单个的使用者出现。使用者小组这个概念非常强大，可以用来支持JMS中队列（queue）或者话题（topic）这两种语义。为了支持队列 语义，我们可以将所有的使用者组成一个单个的使用者小组，在这种情况下，每条消息都会发送给一个单个的使用者。为了支持话题语义，可以将每个使用者分到它自己的使用者小组中，随后所有的使用者将接收到每一条消息。在我们的使用当中，一种更常见的情况是，我们按照逻辑划分出多个使用者小组，每个小组都是有作为一个逻辑整体的多台使用者计算机组成的集群。在大数据的情况下，Kafka有个额外的优点，对于一个话题而言，无论有多少使用者订阅了它，一条条消息都只会存储一次。

消息持久化（Message Persistence）及其缓存

翻译的不错

fbm 顶 我! 4年前

5人顶

不要害怕文件系统！

在对消息进行存储和缓存时，Kafka严重地依赖于文件系统。 大家普遍认为“磁盘很慢”，因而人们都对持久化结（persistent structure）构能够提供说得过去的性能抱有怀疑态度。实际上，同人们的期望值相比，磁盘可以说是既很慢又很快，这取决于磁盘的使用方式。设计的很好的磁盘结构往往可以和网络一样快。

磁盘性能方面最关键的一个事实是，在过去的十几年中，硬盘的吞吐量正在变得和磁盘寻道时间严重不一致了。结果，在一个由6个7200rpm的SATA硬盘组成的RAID-5磁盘阵列上，线性写入（linear write）的速度大约是300MB/秒，但随即写入却只有50k/秒，其中的差别接近10000倍。线性读取和写入是所有使用模式中最具可预计性的一种方式，因而操作系统采用预读（read-ahead）和后写（write-behind）技术对磁盘读写进行探测并优化后效果也不错。预读就是提前将一个比较大的磁盘块中内容读入内存，后写是将一些较小的逻辑写入操作合并起来组成比较大的物理写入操作。关于这个问题更深入的讨论请参考这篇文章ACM Queue article；实际上他们发现，在某些情况下，顺序磁盘访问能够比随即内存访问还要快！

翻译的不错

fbm 顶 我! 4年前

5人顶

为了抵消这种性能上的波动，现代操作系统变得越来越积极地将主内存用作磁盘缓存。所有现代的操作系统都会乐于将所有空闲内存转做磁盘缓存，即时在需要回收这些内存的情况下会付出一些性能方面的代价。所有的磁盘读写操作都需要经过这个统一的缓存。想要舍弃这个特性都不太容易，除非使用直接I/O。因此，对于一个进程而言，即使它在进程内的缓存中保存了一份数据，这份数据也可能在OS的页面缓存（pagecache）中有重复的一份，结构就成了一份数据保存了两次。

更进一步讲，我们是在JVM的基础之上开发的系统，只要是了解过一些Java中内存使用方法的人都知道这两点：

1. Java对象的内存开销（overhead）非常大，往往是对象中存储的数据所占内存的两倍（或更糟）。
2. Java中的内存垃圾回收会随着堆内数据不断增长而变得越来越不明确，回收所花费的代价也会越来越大。

翻译的不错

fbm 顶 我! 4年前

3人顶

由于这些因素，使用文件系统并依赖于页面缓存要优于自己在内存中维护一个缓存或者什么别的结构 —— 通过对所有空闲内存自动拥有访问权，我们至少将可用的缓存大小翻了一倍，然后通过保存压缩后的字节结构而非单个对象，缓存可用大小接着可能又翻了一倍。这么做下来，在GC性能不受损失的情况下，我们可在一台拥有32G内存的机器上获得高达28到30G的缓存。而且，这种缓存即使在服务重启之后会仍然保持有效，而不象进程内缓存，进程重启后还需要在内存中进行缓存重建（10G的缓存重建时间可能需要10分钟），否则就需要以一个全空的缓存开始运行（这么做它的初始性能会非常糟糕）。这还大大简化了代码，因为对缓存和文件系统之间的一致性进行维护的所有逻辑现在都是在OS中实现的，这事OS做起来要比我们在进程中做那种一次性的缓存更加高效，准确性也更高。如果你使用磁盘的方式更倾向于线性读取操作，那么随着每次磁盘读取操作，预读就能非常高效使用随后准能用得着的数据填充缓存。

翻译的不错

fbm 顶 我! 4年前

3人顶

这就让人联想到一个非常简单的设计方案：不是要在内存中保存尽可能多的数据并在需要时将这些数据刷新（flush）到文件系统，而是我们要做完全相反的事情。所有数据都要立即写入文件系统中持久化的日志中但不进行刷新数据的任何调用。实际中这么做意味着，数据被传输到OS内核的页面缓存中了，OS随后会将这些数据刷新到磁盘的。此外我们添加了一条基于配置的刷新策略，允许用户对把数据刷新到物理磁盘的频率进行控制（每当接收到N条消息或者每过M秒），从而可以为系统硬件崩溃时“处于危险之中”的数据在量上加个上限。

这种以页面缓存为中心的设计风格在一篇讲解Varnish的设计思想的文章中有详细的描述（文风略带有助于身心健康的傲气）。

- 3
- 4
- 5
- >

网友评论 共25条 发表评论 回页面顶部

YANGL 发表于 2013-03-09 09:28 回复
赞！

块块 发表于 2013-03-09 09:40 回复
作者很用心，写得很详细，资源珍贵，需要用心看

eyu 发表于 2013-03-09 09:43 回复
mark

耀哥 发表于 2013-03-09 11:47 回复
“使用文件系统并依赖于页面缓存要优于自己在内存中维护一个缓存或者什么别的结构 ” 赞一个

郑柯 来自 Android 发表于 2013-03-09 13:25 回复
很用心，很长，收藏了细品

绿风 发表于 2013-03-09 22:15 回复
mark

s3051024 发表于 2013-03-10 08:29 回复
this framework is scala based

kimmking 发表于 2013-03-10 22:15 回复
good job

t0591 发表于 2013-03-12 12:06 回复
很好，详细

dayan_ 发表于 2013-03-12 13:16 回复
谢谢分享

豆粥 发表于 2013-04-21 21:04 回复
需要反复读几遍，慢慢消化

- 1
- 2
- >

杨子江 发表于 2013-08-13 13:34 回复

使用者

是不是用消费者更加适合呢 哈哈

dclink 发表于 2014-05-16 16:57 回复

正好需

要用kafka，太感谢楼主了

工信布 发表于 2014-08-02 10:54 回复

屎

xiaotao 发表于 2014-10-31 15:46 回复

apache

kafka技术分享系列(目录索引)
http://blog.csdn.net/lizhitao/article/details/39499283

Johnhe 发表于 2015-04-27 00:05 回复

> 引用来自“耀哥”的评论
> “使用文件系统并依赖于页面缓存要优于自己在内存中维护一个缓存或者什么别的结构 ” 赞一个

什么意思呢。没明白。文件系统指的是什么啊？把数据放在硬盘上 不放在内存上。

Johnhe 发表于 2015-04-27 00:07 回复

> 引用来自“耀哥”的评论
> “使用文件系统并依赖于页面缓存要优于自己在内存中维护一个缓存或者什么别的结构 ” 赞一个

是什么意思呢。把数据放在硬盘上，类似solr的数据？而不放在内存里。

scylla 发表于 2016-01-08 15:42 回复

mark

sixi_yanyan 发表于 2016-05-27 17:17 回复

赞！

hello5orld 发表于 2016-08-02 11:35 回复

对于刚

使用kafka的我来说，这篇文章确实挺高深的。

发表评论

回评论顶部 | 回页面顶部