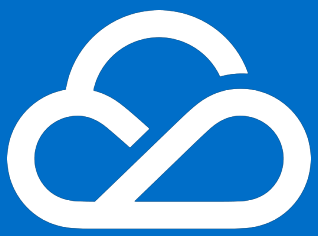
腾讯云消息服务

消息系统CKafka

产品文档



腾讯云

【版权声明】

【商标声明】

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

CKafka产品概述

产品概述

Cloud Kafka是基于腾讯自研CMQ消息队列引擎，提供高吞吐性能、高可扩展性的消息队列服务。Cloud Kafka完美兼容Apache kafka 0.9版本接口，在性能、扩展性以及业务安全保障、运维等方面具有超强优势，让您在享受低成本、超强功能的同时，免除繁琐运维工作。

应用场景

1. 网页追踪

   Cloud Kafka通过实时处理网站活动（PV，搜索，用户其他活动等），并根据类型发布到topic中，这些信息流可以被用于实时监控或离线统计分析等。

   由于每个用户的page view中会生成许多活动信息，因此网站活动跟踪需要很高的吞吐量，Cloud Kafka可以完美满足高吞吐、离线处理等要求。

2. 日志聚合

   Cloud Kafka 提供了低延迟处理、易于支持多个数据源和分布式的数据处理（消费）的特性。相比于中心化的日志聚合系统，Cloud Kafka

   可以在提供同样性能的条件下，实现更强的持久化保证以及更低的端到端延迟。

   因此，Cloud Kafka 的特性决定它非常适合作为”日志收集中心”；多台主机/应用可以将操作日志”批量”“异步”的发送到 Cloud Kafka 集群中，而无需保存在本地或者 DB 中；Cloud Kafka

   可以批量提交消息/压缩消息，这对生产者而言，几乎感觉不到性能的开支。此时

   消费者可以使hadoop 等其他系统化的存储和分析系统对拉取日志进行统计分析。

3. 大数据场景

   对于一些大数据相关的业务场景而言，需要对大量并发数据进行处理和汇总，此时对集群的处理性能和扩展性都有很高的要求。而Cloud Kafka在实现上，其数据分发机制，磁盘存储空间的分配、消息格式的处理、服务器选择以及数据压缩等方面，也决定其适合处理海量的实时消息，并能汇总分布式应用的数据，方便系统运维。

在具体的大数据场景中，Cloud

Kafka能够很好地支持离线数据、流式数据的处理，并能够方便地进行数据聚合、分析等操作。

优势

1.解耦

有效解耦生产者、消费者之间的关系。在确保同样的接口约束的前提下，允许独立扩展或修改生产者/消费者间的处理过程。

2.可扩展性

由于消息的处理过程被解耦，只需要水平扩展处理过程，即可有效增加消息的入队效率和处理效率，十分灵活。

4.削峰填谷

消息队列能够抵挡突增的访问压力，而不会因为突发的超负荷的请求而完全崩溃，有效提升系统健壮性。

5.可恢复性

当系统的部分组件出现故障时，整个系统不会因此收到影响，增加了系统的容错能力。及时某一处理消息的进程故障，队列中的消息依然可以在系统恢复后被处理。

6.顺序读写

Cloud Kafka能够保证一个Partition内消息的有序性，和大部分的消息队列一致，Cloud Kafka可以保证数据按照顺序进行处理，极大提升磁盘效率。

7.异步通信

在业务无需立即处理消息的场景下，Cloud Kafka提供了消息的异步处理机制，访问量高时仅将消息放入队列中，在访问量降低后再对消息进行处理，缓解系统压力。

名词解释

序号	名称	解释
1	Broker	Cloud Kafka集群中的服务器
2	Topic	消息类别，Cloud Kafka是面向消息的
3	Partition	一个物理上分区的概念，一个Topic可以包含一个或者多个partition，Cloud Kafka以partition作为分配单位

序号	名称	解释
4	Replica	partition 的副本，用于保障 partition 的高可用
5	Offset	消息在partition的唯一序号
6	Producer	生产者，负责发布消息
7	Consumer	消费者，从集群中消费消息
8	Consumer group	消费者分组，每个consumer必须属于一个consumer group。每条消息能被多个consumer group消费，但只能被该group中的一个consumer消费
9	Zookeeper	用于存储集群的meta数据、进行leader选举、故障容错等

CKafka常见问题

Cloud Kafka兼容哪一版的开源Kafka？

目前CKafka服务可以完美兼容0.9及以上版本的开源Kafka api，实现用户零成本上云。

什么是主题（TOPIC）？

Topic是每条发布到 Cloud Kafka 集群的消息所属的类别，即 Cloud Kafka 是面向 topic 的。用户需要先创建topic 然后才能读写。

什么是分区（PARTITION）？

Partition 是物理上的概念，每个 topic 会被分成一个或多个 partition。partition可以用来水平扩展topic的吞吐，发布的消息将被写入不同partition，并被若干消费者同时读取。由于Cloud Kafka 分配的单位是 partition，因此在本质上，topic的并行吞吐量和partition个数成正比。

Cloud Kafka和CMQ有什么区别？

CMQ提供金融级的高可靠、高数据持久性消息传输，保证数据强一致性。
Cloud Kafka适用于要求更高吞吐率，对可靠性要求相对较低的场景（如日志聚合等业务）。此外，Cloud Kafka完美兼容kafka的老用户，可以做到零迁移成本，实例完全独占。

Kafka客户端是否可以直接连接Cloud Kafka服务？

Cloud Kafka可以兼容0.9及以上版本的开源Kafka，您可以通过Kafka客户端连接消息中心，并且把代码部署在腾讯云服务中生产或消费消息。

Cloud Kafka如何保证安全性？

Cloud Kafka通过如下安全特性确保安全性：

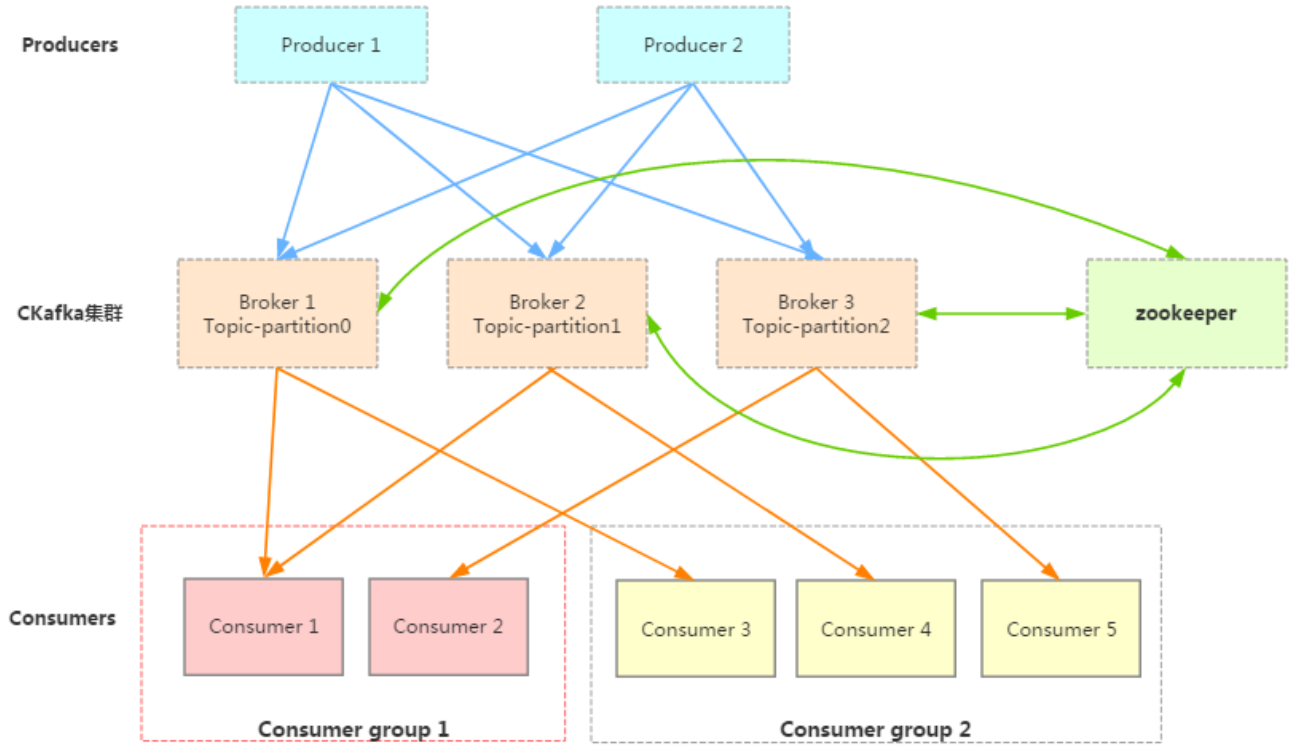
租户隔离：实例的网络访问在账户间天然隔离。
权限控制：Cloud Kafka额外应用层上做了来源ip白名单的鉴权机制。

安全防护：提供多纬度的安全防护、防 DDoS 攻击等服务；

CKafka技术原理

技术原理

Cloud Kafka的架构图如下所示：



一个典型的Cloud Kafka集群如上所示。其中的生产者Producer可能是网页活动产生的消息、或是服务日志等信息。生产者通过push模式将消息发布到Cloud Kafka的Broker集群，消费者通过pull模式从broker中消费消息。消费者Consumer被划分为若干个Consumer Group，此外，集群通过Zookeeper管理集群配置，进行leader选举，故障容错等。

高吞吐的实现

Cloud Kafka中存在大量的网络数据持久化到磁盘和磁盘文件通过网络发送的过程。这一过程的性能直接影响Kafka的整体吞吐量，主要通过以下几点实现。

1. 高效使用磁盘
2. 磁盘中顺序读写数据，提高磁盘利用率

   写message

   消息写到page cache，由异步线程刷盘

   读message

消息直接从page cache转入socket发送出去
当从page cache没有找到相应数据时，此时会产生磁盘IO,从磁盘加载消息到page cache,然后直接从socket发出去

3. Broker的零拷贝(Zero Copy)机制
   使用sendfile系统调用，将数据直接从页缓存发送到网络上
4. 减少网络开销
5. 数据压缩降低网络负载
6. 批处理机制：Producer批量向Broker写数据、Consumer批量从Broker拉数据

数据持久化

Cloud Kafka的数据持久化主要通过如下原理实现：

1. topic中partition存储分布
   在Cloud Kafka文件存储中，同一topic有多个不同partition，每个partition在物理上对应一个文件夹，用户存储该partition中的消息和索引文件。例如，创建两个topic，topic1中存在5个partition，topic2中存在10个partition，则整个集群上会相应生成5+10=15个文件夹。

2. partition中文件存储方式
   partition物理上由多个segment组成，每个segment大小相等，顺序读写，快速删除过期segment,提高磁盘利用率。

水平扩展Scale Out

一个Topic可分多个Partition, 分布在一个或多个Broker上
一个消费者可订阅其中一个或者多个Partition
Producer负责将消息均衡分配到对应的Partition
Partition内消息有序的

Consumer Group设计

Cloud Kafka不删除已消费的消息
任何consumer必须属于一个group

同一Consumer Group中的多个Consumer不同时消费同一个partition

不同Group同时消费同一条消息，多元化 (队列模式、发布订阅模式）

多副本

为何需要多副本设计？

增强系统可用性、可靠性

Replica均匀分布到整个集群

Replica的算法如下：

1、将所有Broker（假设共n个Broker）和待分配的Partition排序

2、将第i个Partition分配到第（i mod n）个Broker上

3、将第i个Partition的第j个Replica分配到第（(i + j) mode n）个Broker上

Leader Election选举机制

Cloud Kafka在Zookeeper中动态维护了一个ISR（in-sync replicas），

ISR里的所有Replica都跟上了leader

只有ISR里的成员才有被选为Leader的可能。

ISR中f+1个Replica，一个Partition能在保证不丢失已commit的消息的前提下

容忍f个Replica的失败。

共有2f+1个Replica（包含Leader和Follower），commit之前必须保证有f+1个

Replica复制完消息，为了保证正确选出新的Leader，fail的Replica不能超过f个。