Kafka学习笔记

# 概述

一种快速，可扩展，容错的消息传递系统

Apache™Kafka是一种快速，可扩展，耐用且容错的发布 - 订阅消息传递系统。由于它具有更高的吞吐量，可靠性和复制性，Kafka通常用于代替传统的消息代理，如JMS和AMQP。

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](https://kafka.apache.org/protocol.html)。

## 基本概念

1. 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。
2. Kafka集群以称为主题的类别存储记录流。
3. 每条记录由一个键，一个值和一个时间戳组成。

## 重要特性

| **特性** | **描述** |
| --- | --- |
| 可扩展性 | 分布式系统可轻松扩展，无需停机 |
| 持久性 | 在磁盘上保留消息，并提供群集内复制 |
| 可靠性 | 复制数据，支持多个订阅者，并在发生故障时自动平衡消费者 |
| 性能 | 发布和订阅的高吞吐量，磁盘结构即使有大量TB的存储消息也能提供持续的性能 |

# Kafka的架构设计

## Kafka的组件

1. Topic
2. Producer
3. Consumer
4. Brokers

在高级别Kafka提供以下保证：

1. 生产者发送到特定主题分区的消息将按其发送顺序附加。也就是说，如果记录M1由与记录M2相同的生成者发送，并且首先发送M1，则M1将具有比M2更低的偏移并且在日志中更早出现。
2. 消费者实例按照它们存储在日志中的顺序查看记录。
3. 对于具有复制因子N的主题，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。

## Topic

Kafka集群以称为主题的类别存储记录流，每条记录由一个键，一个值和一个时间戳组成。

主题是发布记录的类别或订阅源名称。

对于每个主题，Kafka群集都维护一个分区日志，如下所示：



每个分区都是一个有序的，不可变的记录序列，不断附加到结构化的提交日志中。分区中的记录每个都被分配一个称为*偏移*的顺序ID号，它唯一地标识分区中的每个记录。

Kafka集群持久地保留所有已发布的记录 - 无论它们是否已被消耗 - 使用可配置的保留期。

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。

消费组



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，他们将从该组的其他成员接管一些分区; 如果实例死亡，其分区将分发给其余实例。

### 分区

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分区的共享。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。

分区的设计：

领导者、“追随者”；

### Producer

### Consumer



### Brokers

# Kafka MirrorMaker

# 消息传统

消息传统上有两种模型：[排队](http://en.wikipedia.org/wiki/Message_queue)和[发布 - 订阅](http://en.wikipedia.org/wiki/Publish%E2%80%93subscribe_pattern)。

# Kafka的安全

安全包含的内容：

1. Kafka安全性包含多种需求 - 需要加密流经Kafka的数据，防止恶意代理向Kafka发布数据，以及管理对个人或组级别特定主题的访问的能力。
2. 因此，Kafka的最新更新通过SSL支持线加密，基于Kerberos的身份验证以及通过Apache Ranger或其他可插拔授权系统的粒度授权选项。

# Metrics

# 配额-多租户

# Kafka 技术点

1. 机架感知提高了弹性和可用性，使得副本被隔离，因此可以保证它们跨越多个机架或可用区域。
2. 自动副本leader选举，通过检测不均匀分布，自动，均匀地分配集群功能中的领导者，一些经纪人提供与其他人相比更多的数据并进行调整。
3. 消息时间戳，因此Kafka中的每条消息现在都有一个时间戳字段，指示消息的生成时间。
4. SASL改进包括外部认证服务器以及在一台服务器上支持多种类型的SASL认证

## Kafka幂等producer

幂等producer指producer.send的逻辑是幂等的，即发送相同的Kafka消息，broker端不会重复写入消息。

### exactly once semantics，EOS

Kafka 0.11.0.0版本正式支持精确一次处理语义(exactly once semantics，下称EOS)。

Kafka的EOS主要体现在3个方面：

1. 幂等producer：保证发送单个分区的消息只会发送一次，不会出现重复消息
2. 事务(transaction)：保证原子性地写入到多个分区，即写入到多个分区的消息要么全部成功，要么全部回滚
3. 流处理EOS：流处理本质上可看成是“读取-处理-写入”的管道。此EOS保证整个过程的操作是原子性。即流处理EOS保证的是端到端(E2E)消息处理的EOS。

注意，流处理EOS只适用于Kafka Streams

启用方法：

1. 启用幂等producer：在producer程序中设置属性enable.idempotence=true，但不要设置transactional.id。注意是不要设置，而不是设置成空字符串或"null"
2. 启用事务支持：在producer程序中设置属性transcational.id为一个指定字符串(你可以认为这是你的事务名称，故最好起个有意义的名字)，同时设置enable.idempotence=true
3. 启用流处理EOS：在Kafka Streams程序中设置processing.guarantee=exactly\_once

幂等producer提供的语义保证是有条件的：

1. 单分区幂等性：幂等producer无法实现多分区上的幂等性。若要实现多分区上的原子性，需要引入事务。
2. 单会话幂等性：幂等producer无法跨会话实现幂等性。即使同一个producer宕机并重启也无法保证消息的EOS语义

### 幂等producer的设计与实现

### 参考资料

1. [**关于Kafka幂等producer的讨论**](http://www.cnblogs.com/huxi2b/p/7717775.html)**：**<http://www.cnblogs.com/huxi2b/p/7717775.html>

## Kafka日志留存策略

### 参考资料

1. 《关于Kafka日志留存策略的讨论.docx》

## [**Kafka Java consumer动态修改topic订阅**](http://www.cnblogs.com/huxi2b/p/7040617.html)

### 参考资料

1. Kafka Java consumer动态修改topic订阅：<http://www.cnblogs.com/huxi2b/p/7040617.html>

# Kafka管理、监控

## 管理、监控的内容

1. Topic管理：创建/删除主题
2. 安全管理：管理用户权限，更容易、更快速地分发安全令牌
3. 支持通过Kafka REST服务器/ API发布/消费数据

## 管理与监控的框架、贡献者

框架：

贡献者：

1. Ambari可视化Kafka运营指标的视图
2. Hortonworks的Kafka管理面板

# 常见的应用场景

1. 流处理
2. 网站活动跟踪
3. 度量标准收集和监控
4. 日志聚合

# 参考资料