Apache Kafka 2.4 正式发布，重要功能详细介绍

2019年12月18日 Apache Kafka 2.4 正式发布了，这个版本有很多新功能，本文将介绍这个版本比较重要的功能，完整的更新可以参见 release notes

Kafka broker, producer, 以及 consumer 新功能

KIP-392: 允许消费者从最近的副本获取数据

在 Kafka 2.4 版本之前，消费者只能从 leaders 分区那里获取数据。在多数据中心部署的集群中，这通常意味着使用者不得不承担昂贵的跨数据中心的网络成本，以便从 leaders
分区那里获取数据。有了 KIP-392 特性之后，Kafka 现在支持从 follower 副本读取数据。这使得 broker 能够将消费者重定向到附近的 follower 副本，从而节省成本。

KIP-429: Kafka 消费者增量平衡协议

KIP-429 这个 KIP 在原有的紧迫再平衡协议（eager rebalance
protocol）的基础上，增加了消费者增量平衡协议（Incremental Rebalance Protocol）。与 eager 协议不同，eager
协议总是在重新平衡之前撤销所有已分配的分区，然后尝试重新分配它们，而 incremental 协议允许消费者在重新平衡事件期间保留其分区，从而尽量减少消费者组成员之间的分区迁移。因此，通过 scaling out/down

操作触发的端到端重新平衡时间更短，这有利于重量级、有状态的消费者，比如 Kafka Streams 应用程序。

KIP-455: 为 Replica Reassignment 参见管理者 API

作为对现有基于 ZooKeeper API 的替换，新的 API 支持增量副本重新分配（incremental replica reassignments）和取消正在进行的重新分配。这也解决了当前基于 ZooKeeper API 的局限性，比如安全性增强和可审核性。新的 API 通过 AdminClient 暴露给我们使用。

KIP-480: Sticky Partitioner

在 Kafka 2.4 版本之前，在没有指定分区和键的情况下，生产者默认 partitioner 以循环方式对数据进行分区（具体参见 《Key为null时Kafka如何选择分区(Partition)》）。这导致将一个大的 batches 拆成许多小的 batches，导致更多的请求以及排队过程中产生更长的延时。

KIP-480 实现了一个新的 partitioner，在没有指定分区或键而导致批处理满时选择 Sticky Partitioner。使用 Sticky Partitioner 有助于改进消息批处理，减少延迟，并减少 broker 的负载。在 KIP 上讨论的一些基准测试显示，延迟降低了50%，CPU 利用率降低了5-15%。

KIP-482: Kafka 协议支持可选的标记字段

Kafka 远程过程调用（RPC）协议有自己的二进制数据序列化格式。Kafka 协议目前不支持可选字段，也不支持附加额外字段。

为了支持这些场景，KIP-482 将可选的标记字段添加到 Kafka 序列化格式中。标记字段（Tagged fields）总是可选的。KIP-482 还为可变长度的对象实现了更高效的序列化。

KIP-504: Add new Java Authorizer Interface

这个 KIP 定义了一个与 broker 中的其他可插拔接口一致的 Java authorizer API。当前 Scala authorizer API 中的一些限制在不破坏兼容性的情况下是无法修复的，在新的 API 中已经解决了这些限制。额外的请求上下文现在可以提供给授权方，以支持基于安全协议或监听器的授权。

该 API 还支持使用批处理的方式进行异步 ACL 更新。新的可插拔 authorizer API 只需要依赖于客户端的 JAR。增加了一个新的开箱即用的授权器，利用了新 API 支持的特性。提供给授权方的附加上下文用于改进审计日志记录。当为一个资源添加多个 ACL 时，批处理更新使用新的授权器提高 ACL 更新的效率。此外，授权器实现现在可以动态配置，而无需重新启动 broker。

KIP-525: 在 CreateTopics 响应中返回主题元数据和配置

在 Kafka 2.4 版本之前，CreateTopics API 响应只返回成功或失败状态。有了 KIP-525 之后，现在这个 API 响应返回额外的元数据，包括创建主题的实际配置，这就使得我们在创建主题之后不需要再发送额外请求来获取主题的配置。

此外，在调用 CreateTopics API 带上 validateOnly=true 参数时可以获得用于主题创建的缺省 broker 配置。这对于在用于创建主题的管理工具中显示缺省配置非常有用。

KAFKA-7548: KafkaConsumer 不应该丢弃已经获取的暂停分区（paused partitions）的数据

当用户在消费者 API 中暂停某个分区时，该分区被认为是“不可获取的（unfetchable）”。当 consumer 已经获取了某个分区的数据，并且该分区被暂停，那么在 consumer 下一个轮询中，来自“不可获取”分区的所有数据将被丢弃。在常规操作期间，暂停和恢复分区是常见的，这可能导致在不必要的情况下丢弃预取的数据。

当分区从暂停变成恢复后，Kafka 将生成新的获取请求并将其发送给 broker，以便再次获得相同的分区数据。根据分区暂停和恢复的频率，这可能会影响 consumer 轮询的许多方面，包括 broker/consumer 吞吐量、消费者获取请求的数量，以及与 NIO 相关的垃圾收集(GC)问题。这个问题现在通过保留暂停分区已经 fetch 的数据来解决，以便在用户恢复分区后，能够在消费者轮询中返回该数据。

Kafka Connect 新功能

KIP-382: MirrorMaker 2.0

KIP-382 引入了 MirrorMaker 2.0 (MM2)，这是一个基于 Connect 框架的多集群、跨数据中心复制引擎。该工具包括几个用于灾难恢复的特性，包括跨集群消费者检查点和偏移同步。自动主题重命名和周期检测支持双向主动复制和其他复杂拓扑。

这个 KIP 引入了 RemoteClusterUtils 新的类，用于客户端来解析检查点、心跳以及来自其他集群的“远程主题”。

KIP-440: 扩展 Connect Converter 使得其支持头部信息

KIP-440 中引入了新的功能，使得 Kafka Connect 中支持头部信息（header），这样就可以将 Kafka Connect 与依赖于头部信息进行序列化/反序列化的 Kafka 生产者和消费者一起使用。

KIP-507: Securing Internal Connect REST Endpoints

KIP-507 为内部 REST endpoint 带来开箱即用的身份验证和授权，Connect workers 使用这个端点将任务配置转发给 leader。如果没有这个功能，可以使用此端点将任意任务配置写入 Connect cluster。

KIP-481: SerDe Improvements for Connect Decimal type in JSON

KIP-481 里面在 JSON converter 里面加入了 decimal.format，主要目的是将 Connect 里面的 DECIMAL 逻辑类型值序列化为数字而不是 base64 形式的字符串。这个新选项的默认值为 base64，以遍兼容之前的行为，但是我们也可以将其更改为 number，以便将 decimal 值序列化为普通的 JSON 数字。JSON converter 使用这两种格式都会自动反序列化，因此在更改

源连接转换器使用数字格式之前，请确保升级消费者应用程序和接收连接器。

Kafka Streams 的新功能

KIP-213: KTable 支持非键 join

以前，Streams 领域特定语言（DSL）只允许基于 KTables 的主键（primary key）进行表与表之间的 join。现在，对于一个 KTable (左输入)与另一个 KTable (右输入)进行 join，该 join 基于一个指定的外键作为其值字段的一部分，join 结果是一个新的 KTable，新 KTable 的主键是左 KTable 的主键。

KIP-307: 允许使用 KStreams DSL 定义自定义处理器名称

在 Kafka 2.4 版本之前，通过 Kafka Streams DSL 构建新拓扑时，处理器（processor）的名称是自动生成的。如果处理器名称随机生成，那么包含许多操作的复杂拓扑可能很难理解，有了这个 KIP ，用户可以为处理器取一个有意义的名称，这样便于后面的管理。

KIP-470: TopologyTestDriver 测试输入和输出可用性的改进

TopologyTestDriver 允许我们测试 Kafka Streams 的逻辑，这比利用实际的生产者和消费者要快得多，这使得模拟不同的实时场景成为可能。Kafka 2.4.0 引入了 TestInputTopic 和 TestOutputTopic 类来简化测试接口。

度量、监视和操作性的改进

- KIP-412 adds support to dynamically alter a broker's log levels using the Admin API.
- KIP-495 allows users to dynamically alter log levels in the Connect framework.
- KIP-521 changes Connect to also send log messages to a file and rolls that file every day.
- KIP-460 modifies the PreferredLeaderElection RPC to support unclean leader election in addition to preferred leader election.
- KIP-464 allows you to leverage num.partitions and default.replication.factor from the AdminClient#createTopics API.
- KIP-492 supports the security provider config, which can be used to configure custom security algorithms.
- KIP-496 adds an API to delete consumer offsets and expose it via the AdminClient.
- KIP-503 adds metrics to monitor the number of topics/replicas marked for deletion.
- KIP-475 adds metrics to measure the number of tasks on a connector.
- KIP-471 exposes a subset of RocksDB's statistics in Kafka Streams metrics, which enables users to find bottlenecks and tune RocksDB accordingly.
- KIP-484 adds new metrics for the group and transaction metadata loading duration.
- KIP-444 adds a few new metrics at the Streams instance level such as static version/commit-id as well as dynamic state.

ZooKeeper 升级到 3.5.x

ZooKeeper 已经升级到 3.5 x，ZooKeeper 3.5.x 中添加了对 TLS 加密的支持，这使我们能够在 Kafka brokers 和 ZooKeeper 之间配置 TLS 加密。

支持 Scala 2.13

Apache Kafka 2.4.0 现在支持 Scala 2.13，同时仍与 Scala 2.12 和 2.11 兼容。

本文翻译自 What's New in Apache Kafka 2.4