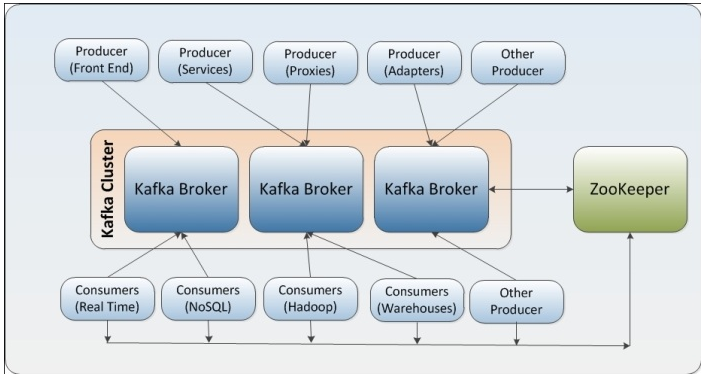
Apache Kafka是个开源的分布式消息发布订阅系统，具有以下特征：

* **消息持久化（persistent messaging）：Kafka提供时间复杂度为O(1)的消息持久化能力，即使是TB级别以上的数据也能保证常数时间复杂度的访问性能。在Kafka中，消息被持久化在磁盘中，同时也在集群之间复制，防止数据丢失**
* **高吞吐率（high throughput）：即使在普通的机器上也能提供每秒上百MB读写的能力。**
* **分布式（distributed）：Kafka支持Kafka服务器间的消息分区，以及基于消费者机器集群的分布式消费，维护每个分区内的顺序。Kafka集群可以在不停机的情况下弹性地增减结点。**
* **实时（real time）：生产者线程产生的消息应该立即被消费者线程可见，这是基于事件系统的关键特性，例如Complex Event Processing (CEP) system。**
* **支持在线水平扩展（scale out）**



**使用Kafka的案例**

Kafka通常被用于：

* **日志聚合（log aggregation）：从服务器收集日志文件，将它们放到文件服务器或HDFS中处理。使用Kafka可以将日志数据或事件数据抽象为信息流，从而避免了对文件格式等的依赖。同时提供了低延迟处理能力，并支持多数据源和分布式处理。**
* **流处理（stream processing）：Kafka可用于对数据进行多阶段处理的场景，例如，一个主题的原始数据被消费，经过增强或转换处理形成新的主题供后续的消费者使用。这样的处理过程称为流处理。**
* **提交日志（commit logs）：Kafka可用来处理大规模分布式系统的外部提交日志。在Kafka集群之间复制日志可以帮助故障节点恢复其状态。**
* **点击流跟踪（click stream tracking）：另一个常用Kafka的场景是，捕捉用户点击流数据，如页面视图、搜索等这样的real-time publish-subscribe feed。这些数据被发布到中心主题，由于数据量巨大，每个活动类型一个主题。这些主题可以被很多类型的消费者订阅，如实时处理和监控应用。**
* **消息处理（messaging）：message broker用于将数据处理和数据提供者解耦。Kafka可以作为message broker，它具有更好的吞吐量、内建分区、复制和容错能力。**

**Kafka的一些高级特性**

* **为分区提供了复制因子。保证了所有提交的信息不会丢失，即使某个borker失效时分区中还有未消费的数据，至少有一个副本是可用的。默认生产者发送信息请求会阻塞直到信息提交到所有活跃的副本中，也可以通过配置指定生产者将信息提交到某个broker。**
* **消费者采用长轮询模式（long-pulling model），并且会阻塞直到有可用的提交信息，避免了频繁轮询。**

安装：

1、**tar -xzf kafka\_2.11-0.10.1.0.tgz**

2、**bin/zookeeper-server-start.sh config/zookeeper.properties**

3、**bin/kafka-server-start.sh config/server.properties**

创建topic:

**bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test**

没有分区，且没有进行冗余备份

**bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 3 --partitions 3 --topic my-replicated-topic**

创建了三个分区三份冗余备份

查看topic信息：

**bin/kafka-topics.sh --list --zookeeper localhost:2181**

**bin/kafka-topics.sh --list --zookeeper localhost:2181 –topic test**

**--describe**

生产者Producer:

**bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test**

消费者consumer：

**bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --from-beginning**

添加broker：

> **cp config/server.properties config/server-1.properties**

> **cp config/server.properties config/server-2.properties**

config/server-1.properties:

broker.id=1

listeners=PLAINTEXT://:9093

log.dir=/tmp/kafka-logs-1

config/server-2.properties:

broker.id=2

listeners=PLAINTEXT://:9094

log.dir=/tmp/kafka-logs-2

> **bin/kafka-server-start.sh config/server-1.properties &**

...

> **bin/kafka-server-start.sh config/server-2.properties &**

Java对应kafka api的批量发送：

List<KeyedMessage<String, String>> messages = new ArrayList<KeyedMessage<String, String>>(100);

for (int i = 0; i <= 10000; i++) {

KeyedMessage<String, String> message =

new KeyedMessage<String, String>(KafkaProperties.TOPIC, i + "", "Message" + i);

messages.add(message);

if (i % 100 == 0) {

producer.send(messages);

messages.clear();

}

}

producer.send(messages);