Apache Kafka

By: **IBM Cloud Education**

18 February 2020

Integration

Apache Kafka

Learn about Apache Kafka, the open source streaming technology behind some of the most popular real-time, event-driven user experiences on the web, including AirBnB, Netflix, and

Pinterest.

What is Apache Kafka?

Apache Kafka (Kafka) is an open source, distributed streaming platform that enables (among other things) the development of real-time, event-driven applications. So, what does that mean?

Today, billions of data sources continuously generate streams of data records, including streams of *events*. An event is a digital record of an action that happened and the time that it happened. Typically, an event is an action that drives another action as part of a process. A customer placing an order, choosing a seat on a flight, or submitting a registration form are all examples of events. An event doesn't have to involve a person—for example, a connected thermostat's report of the temperature at a given time is also an event.

These streams offer opportunities for applications that respond to data or events in real-time. A streaming platform enables developers to build applications that continuously consume and process these streams at extremely high speeds, with a high level of fidelity and accuracy based on the correct order of their occurrence.

LinkedIn developed Kafka in 2011 as a high-throughput message broker for its own use, then open-sourced and donated Kafka to the Apache Software Foundation (link resides outside IBM). Today, Kafka has evolved into the most widely-used streaming platform, capable of ingesting and processing *trillions* of records per day without any perceptible performance lag as volumes scale. Fortune 500 organizations such as Target, Microsoft, AirBnB, and Netflix rely on Kafka to deliver real-time, data-driven experiences to their customers.

The following video provides further information about Kafka (9:10):

How Kafka works

Kafka has three primary capabilities:

1. It enables applications to publish or subscribe to data or event streams.
2. It stores records accurately (i.e., in the order in which they occurred) in a fault-tolerant and durable way.
3. It processes records in real-time (as they occur).

Developers can leverage these Kafka capabilities through four APIs:

- **Producer API:** This enables an application to publish a stream to a Kafka *topic*. A topic is a named log that stores the records in the order they occurred relative to one another. After a record is written to a topic, it can't be altered or deleted; instead, it remains in the topic for a preconfigured amount of time—for example, for two days—or until storage space runs out.

- **Consumer API:** This enables an application to subscribe to one or more topics and to ingest and process the stream stored in the topic. It can work with records in the topic in real-time, or it can ingest and process past records.

- **Streams API:** This builds on the Producer and Consumer APIs and adds complex processing capabilities that enable an application to perform continuous, front-to-back stream processing—specifically, to consume records from one or more topics, to analyze or aggregate or transform them as required, and to publish resulting streams to the same topics or other topics. While the Producer and Consumer APIs can be used for simple stream processing, it's the Streams API that enables development of more sophisticated data- and event-streaming applications.

- **Connector API:** This lets developers build *connectors*, which are reusable producers or consumers that simplify and automate the integration of a data source into a Kafka cluster. Check out some ready-made connectors for popular data stores (link resides outside IBM).

Kafka performance

Kafka is a distributed platform—it runs as a fault-tolerant, highly available cluster that can span multiple servers and even multiple data centers. Kafka topics are partitioned and replicated in such a way that they can scale to serve high volumes of simultaneous consumers without impacting performance. As a result, according to Apache.org, "Kafka will perform the same whether you have 50KB or 50TB of persistent storage on the server."

Kafka use cases

Kafka is used primarily for creating two kinds of applications:

- **Real-time streaming data pipelines:** Applications designed specifically to move millions and millions of data or event records between enterprise systems—at scale and in real-time—and move them reliably, without risk of corruption, duplication of data, and other problems that typically occur when moving such huge volumes of data at high speeds.

- **Real-time streaming applications:** Applications that are driven by record or event streams and that generate streams of their own. If you spend any time online, you encounter scores of these applications every day, from the retail site that continually updates the quantity of a product at your local store, to sites that display personalized recommendations or advertising based on clickstream analysis.

Kafka vs. RabbitMQ

RabbitMQ is a very popular open source *message broker*, a type of middleware that enables applications, systems, and services to communicate with each other by translating messaging protocols between them.

Because Kafka began as a kind of message broker (and can, in theory, still be used as one) and because RabbitMQ supports a publish/subscribe messaging model (among others), Kafka and RabbitMQ are often compared as alternatives. But, the comparisons aren't really practical, and they often dive into technical details that are beside the point when choosing between the two. For example, that Kafka topics can have multiple subscribers, whereas each RabbitMQ message can have only one; or that Kafka topics are durable, whereas RabbitMQ messages are deleted once consumed.

The bottom line is:

- *Kafka is a stream processing platform* that enables applications to publish, consume, and process high volumes of record streams in a fast and durable way; and

- *RabbitMQ is a message broker* that enables applications that use different messaging protocols to send messages to, and receive messages from, one another.

Apache technologies often used with Kafka

Kafka is frequently used with several other Apache technologies as part of a larger streams processing, event driven architecture or big data analytics solution.

Apache Spark

Apache Spark is an analytics engine for large-scale data processing. You can use Spark to perform analytics on streams delivered by Apache Kafka and to produce real-time stream processing applications, such as the aforementioned click-stream analysis.

Apache NiFi

Apache NiFi is a data flow management system with a visual, drag-and-drop interface. Because NiFi can run as a Kafka producer and a Kafka consumer, it’s an ideal tool for managing data flow challenges that Kafka can’t address.

Apache Flink

Apache Flink is an engine for performing computations on event streams at scale, with consistently high speed and low latency. Flink can ingest streams as a Kafka consumer, perform operations based on these streams in real-time, and publish the results to Kafka or to another application.

Apache Hadoop

Apache Hadoop is a distributed software framework that lets you store massive amounts of data in a cluster of computers for use in big data analytics, machine learning, data mining, and other data-driven applications that process structured and unstructured data. Kafka is often used to create a real-time streaming data pipeline to a Hadoop cluster.

Kafka and IBM Cloud®

Kafka will remain part of application modernization as the demand for better customer experiences and more applications impacts business and IT operations. When it comes to meeting such demands, a move toward greater automation also helps. Ideally, it would start with small, measurably successful projects, which you can then scale and optimize for other processes and in other parts of your organization.

Working with IBM, you'll have access to AI-powered automation capabilities, including prebuilt workflows, to help accelerate innovation by making every process more intelligent.

Take the next step:

- Learn about IBM Event Streams for IBM Cloud, a fully managed Kafka-as-a-Service event streaming platform that allows you to build event-driven applications in the IBM Cloud.

- IBM Event Streams is also available as part of the AI-powered automation software IBM Cloud Pak® for Integration.

- Explore IBM Watson® IoT Platform, IBM Streaming Analytics and IBM Cloud Functions, which IBM Event Streams also seamlessly integrates with for event-driven application development.

- Try Event Streams for free without any commitment. To see what an Apache Kafka application looks like, check out IBM samples (link resides outside IBM) in Java, Node.js, and Python on GitHub.

- Check out a container-native version of Event Streams (link resides outside IBM).

- Take our integration maturity assessment to evaluate your integration maturity level across critical dimensions and discover the actions you can take to get to the next level.

- Download our agile integration guide, which explores the merits of a container-based, decentralized, microservices-aligned approach for integrating solutions.

Get started with an IBM Cloud account today.