What is Streaming Data?

Streaming Data is data that is generated continuously by thousands of data sources, which typically send in the data records simultaneously, and in small sizes (order of Kilobytes). Streaming data includes a wide variety of data such as log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services, and telemetry from connected devices or instrumentation in data centers.

This data needs to be processed sequentially and incrementally on a record-by-record basis or over sliding time windows, and used for a wide variety of analytics including correlations, aggregations, filtering, and sampling. Information derived from such analysis gives companies visibility into many aspects of their business and customer activity such as –service usage (for metering/billing), server activity, website clicks, and geo-location of devices, people, and physical goods –and enables them to respond promptly to emerging situations. For example, businesses can track changes in public sentiment on their brands and products by continuously analyzing social media streams, and respond in a timely fashion as the necessity arises.

Benefits of Streaming Data

Streaming data processing is beneficial in most scenarios where new, dynamic data is generated on a continual basis. It applies to most of the industry segments and big data use cases. Companies generally begin with simple applications such as collecting system logs and rudimentary processing like rolling min-max computations. Then, these applications evolve to more sophisticated near-real-time processing. Initially, applications may process data streams to produce simple reports, and perform simple actions in response, such as emitting alarms when key measures exceed certain thresholds. Eventually, those applications perform more sophisticated forms of data analysis, like applying machine learning algorithms, and extract deeper insights from the data. Over time, complex, stream and event processing algorithms, like decaying time windows to find the most recent popular movies, are applied, further enriching the insights.

Streaming Data Examples

- Sensors in transportation vehicles, industrial equipment, and farm machinery send data to a streaming application. The application monitors performance, detects any potential defects in advance, and places a spare part order automatically preventing equipment down time.
- A financial institution tracks changes in the stock market in real time, computes value-at-risk, and automatically rebalances portfolios based on stock price movements.
- A real-estate website tracks a subset of data from consumers' mobile devices and makes real-time property recommendations of properties to visit based on their geo-location.
- A solar power company has to maintain power throughput for its customers, or pay penalties. It implemented a streaming data application that monitors of all of panels in the field, and schedules service in real time, thereby minimizing the periods of low throughput from each panel and the associated penalty payouts.
- A media publisher streams billions of clickstream records from its online properties, aggregates and enriches the data with demographic information about users, and optimizes content placement on its site, delivering relevancy and better experience to its audience.
- An online gaming company collects streaming data about player-game interactions, and feeds the data into its gaming platform. It then analyzes the data in real-time, offers incentives and dynamic experiences to engage its players.

Comparison between Batch Processing and Stream Processing

Before dealing with streaming data, it is worth comparing and contrasting *stream processing* and *batch processing*. *Batch processing*can be used to compute arbitrary queries over different sets of data. It usually computes results that are derived from all the data it encompasses, and enables deep analysis of big data sets. MapReduce-based systems, like Amazon EMR, are examples of platforms that support batch jobs. In contrast, *stream processing* requires ingesting a sequence of data, and incrementally updating metrics, reports, and summary statistics in response to each arriving data record. It is better suited for real-time monitoring and response functions.

	Batch processing	Stream processing
Data scope	Queries or processing over all or most of the data in the dataset.	Queries or processing over data within a rolling time window, or on just the most recent data record.
Data size	Large batches of data.	Individual records or micro batches consisting of a few records.

Performance	Latencies in minutes to hours.	Requires latency in the order of seconds or milliseconds.
Analyses	Complex analytics.	Simple response functions, aggregates, and rolling metrics.

Many organizations are building a hybrid model by combining the two approaches, and maintain a real-time layer and a batch layer. Data is first processed by a streaming data platform such as Amazon Kinesis to extract real-time insights, and then persisted into a store like S3, where it can be transformed and loaded for a variety of batch processing use cases.

Challenges in Working with Streaming Data

Streaming data processing requires two layers: a storage layer and a processing layer. The storage layer needs to support record ordering and strong consistency to enable fast, inexpensive, and replayable reads and writes of large streams of data. The processing layer is responsible for consuming data from the storage layer, running computations on that data, and then notifying the storage layer to delete data that is no longer needed. You also have to plan for scalability, data durability, and fault tolerance in both the storage and processing layers. As a result, many platforms have emerged that provide the infrastructure needed to build streaming data applications including Amazon Kinesis Streams, Amazon Kinesis Firehose, Apache Kafka, Apache Flume, Apache Spark Streaming, and Apache Storm.

Working with Streaming Data on AWS

Amazon Web Services (AWS) provides a number options to work with streaming data. You can take advantage of the managed streaming data services offered by Amazon Kinesis, or deploy and manage your own streaming data solution in the cloud on Amazon EC2.

Amazon Kinesis is a platform for streaming data on AWS, offering powerful services to make it easy to load and analyze streaming data, and also enables you to build custom streaming data applications for specialized needs. It offers two services: Amazon Kinesis Firehose, and Amazon Kinesis Streams.

In addition, you can run other streaming data platforms such as –Apache Kafka, Apache Flume, Apache Spark Streaming, and Apache Storm –on Amazon EC2 and Amazon EMR.

Amazon Kinesis Streams

Amazon Kinesis Streams enables you to build your own custom applications that process or analyze streaming data for specialized needs. It can continuously capture and store terabytes of data per hour from hundreds of thousands of sources. You can then build applications that consume the data from Amazon Kinesis Streams to power real-time dashboards, generate alerts, implement dynamic pricing and advertising, and more. Amazon Kinesis Streams supports your choice of stream processing framework including Kinesis Client Library (KCL), Apache Storm, and Apache Spark Streaming. Learn more about Amazon Kinesis Streams »

Amazon Kinesis Firehose

Amazon Kinesis Firehose is the easiest way to load streaming data into AWS. It can capture and automatically load streaming data into Amazon S3 and Amazon Redshift, enabling near real-time analytics with existing business intelligence tools and dashboards you're already using today. It enables you to quickly implement an ELT approach, and gain benefits from streaming data quickly. Learn more about Amazon Kinesis Firehose »

Other Streaming Solutions on Amazon EC2

You can install streaming data platforms of your choice on Amazon EC2 and Amazon EMR, and build your own stream storage and processing layers. By building your streaming data solution on Amazon EC2 and Amazon EMR, you can avoid the friction of infrastructure provisioning, and gain access to a variety of stream storage and processing frameworks. Options for streaming data storage layer include Apache Kafka and Apache Flume. Options for stream processing layer Apache Spark Streaming and Apache Storm.