Cloudera Streaming Analytics 1.5.0

Cloudera Streaming Analytics Overview

Date published: 2019-12-17
Date modified: 2021-09-29

Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Streaming Analytics in Cloudera.....4

What is Apache Flink?..... 5

Core Features of Flink.....6

Streaming Analytics in Cloudera

Cloudera Streaming Analytics (CSA) offers real-time stream processing and streaming analytics powered by Apache Flink. Flink implemented on CDP provides a flexible streaming solution with low latency that can scale to large throughput and state. Additionally to Flink, CSA includes SQL Stream Builder to offer data analytical experience using SQL queries on your data streams.

Key features of Cloudera Streaming Analytics

SQL Stream Builder

SQL Stream Builder is a job management interface to compose and run Streaming SQL on streams, as well as to create durable data APIs for the results.

Cloudera Platform

Implementing Flink on the Cloudera Platform allows you to easily integrate with Runtime components, and have all the advantages of cluster and service management with Cloudera Manager.

Streaming Platform

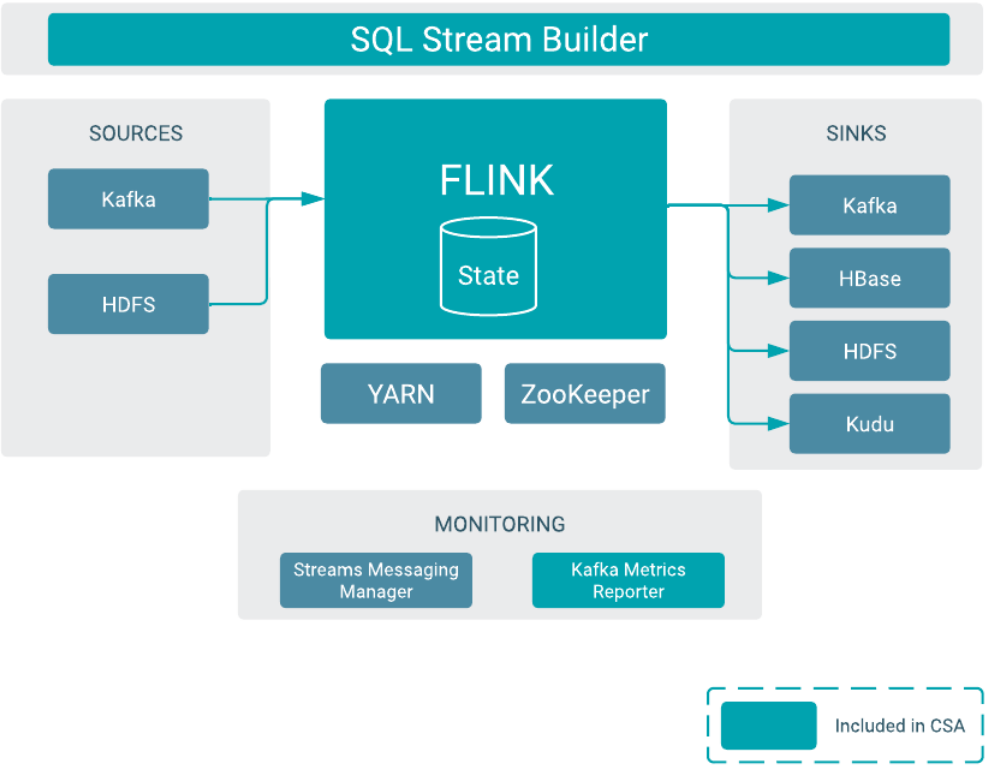
For streaming analytics, CSA fits into a complete streaming platform augmented by Apache Kafka, Schema Registry, Streams Messaging Manager in the Cloudera Runtime stack.

Supported Connectors

CSA offers Kafka, HBase, HDFS, Kudu and Hive as connectors to choose based on the requirements of your application deployment.

Monitoring Solutions

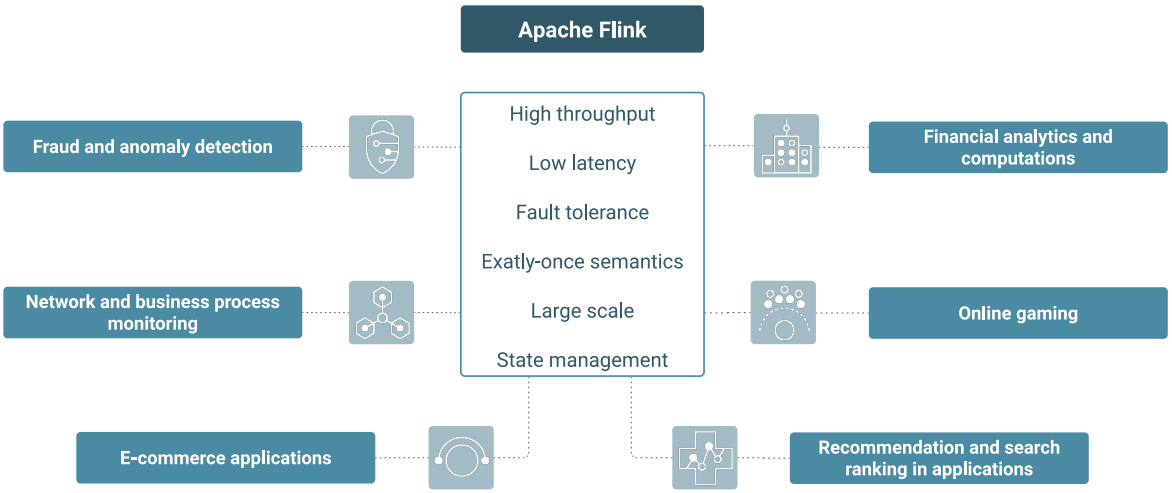
Within CSA, Kafka Metrics Reporter, Streams Messaging Manager and the reworked Flink Dashboard helps you monitor and troubleshoot your Flink applications.



What is Apache Flink?

Flink is a distributed processing engine and a scalable data analytics framework. You can use Flink to process data streams at a large scale and to deliver real-time analytical insights about your processed data with your streaming application.

Flink is designed to run in all common cluster environments, perform computations at in-memory speed and at any scale. Furthermore, Flink provides communication, fault tolerance, and data distribution for distributed computations over data streams. A large variety of enterprises choose Flink as a stream processing platform due to its ability to handle scale, stateful stream processing, and event time.

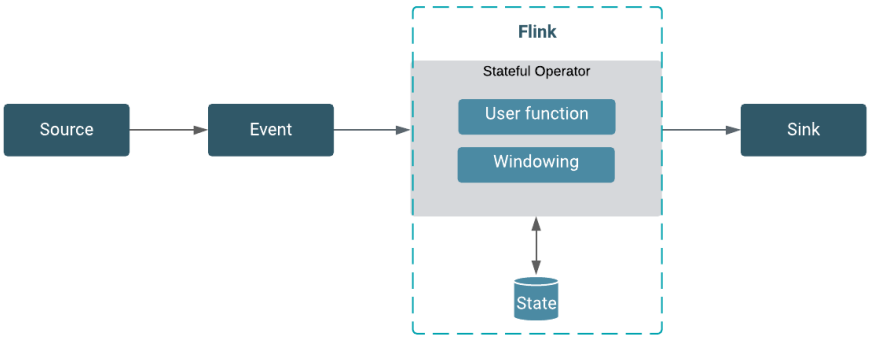


Core Features of Flink

Learn more about the specific details of Flink architecture, the DataStream API, how Flink handles event time and watermarks, and how state management works in Flink.

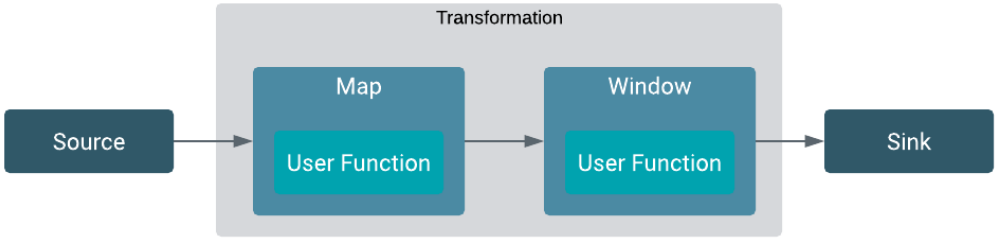
Architecture

The two main components for the task execution process are the Job Manager and Task Manager. The Job Manager on a master node starts a worker node. On a worker node the Task Managers are responsible for running tasks and the Task Manager can also run more than one task at the same time. The resource management for the tasks are completed by the Job manager in Flink. In a Flink cluster, Flink jobs are executed as YARN applications. HDFS is used to store recovery and log data, while ZooKeeper is used for high availability coordination for jobs.



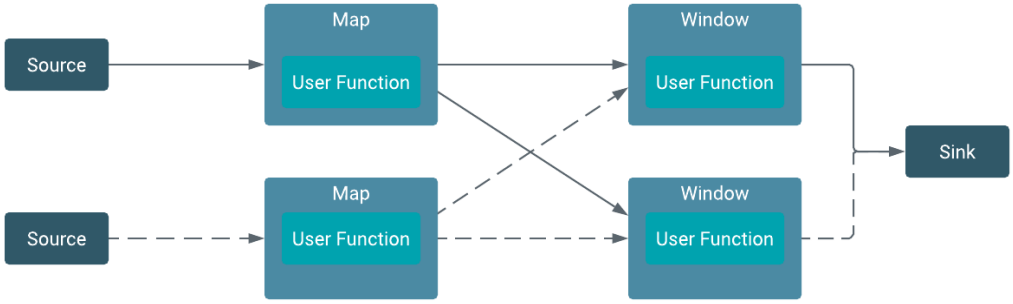
DataStream API

The DataStream API is used as the core API to develop Flink streaming applications using Java or Scala programming languages. The DataStream API provides the core building blocks of the Flink streaming application: the datastream and the transformation on it. In a Flink program, the incoming data streams from a source are transformed by a defined operation which results in one or more output streams to the sink.



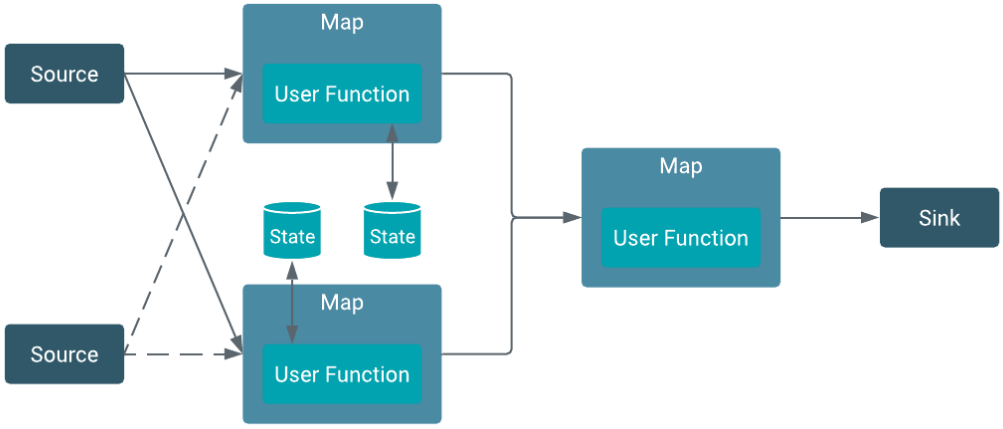
Operators

Operators transform one or more DataStreams into a new DataStream. Programs can combine multiple transformations into sophisticated data flow topologies. Other than the standard transformations like map, filter, aggregation, you can also create windows and join windows within the Flink operators. On a dataflow one or more operations can be defined which can be processed in parallel and independently to each other. With windowing functions, different computations can be applied to different streams in the defined time window to further maintain the processing of events. The following image illustrates the parallel structure of dataflows.



State and state backend

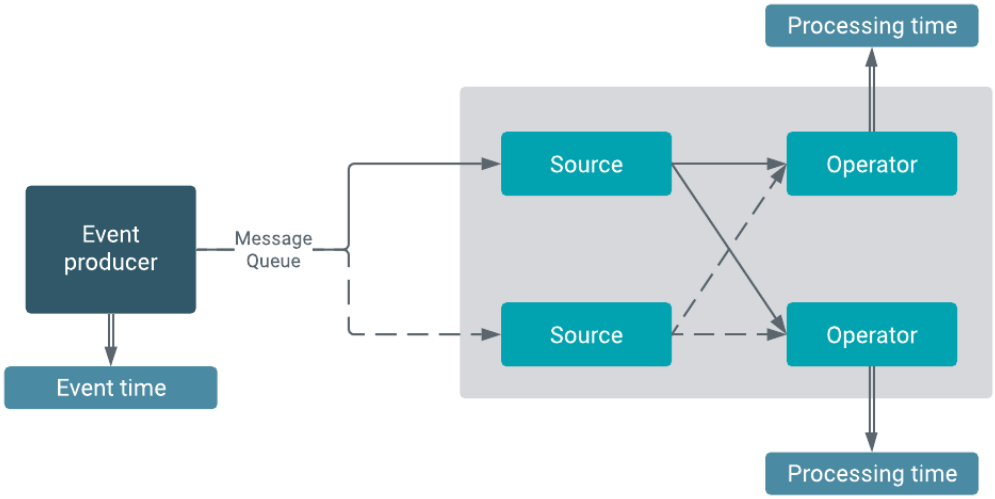
Stateful applications process dataflows with operations that store and access information across multiple events. You can use Flink to store the state of your application locally in state backends that guarantee lower latency when accessing your processed data. You can also create checkpoints and savepoints to have a fault-tolerant backup of your streaming application on a durable storage.



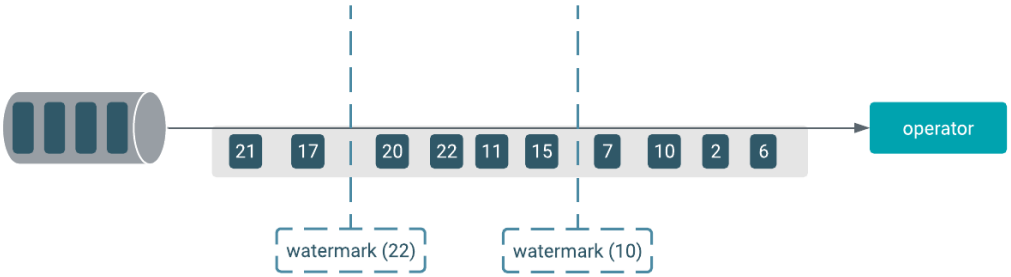
Event time and watermark

In time-sensitive cases where the application uses alerting or triggering functions, it is important to distinguish between event time and processing time. To make the designing of applications easier,

you can create your Flink application either based on the time when the event is created or when it is processed by the operator.

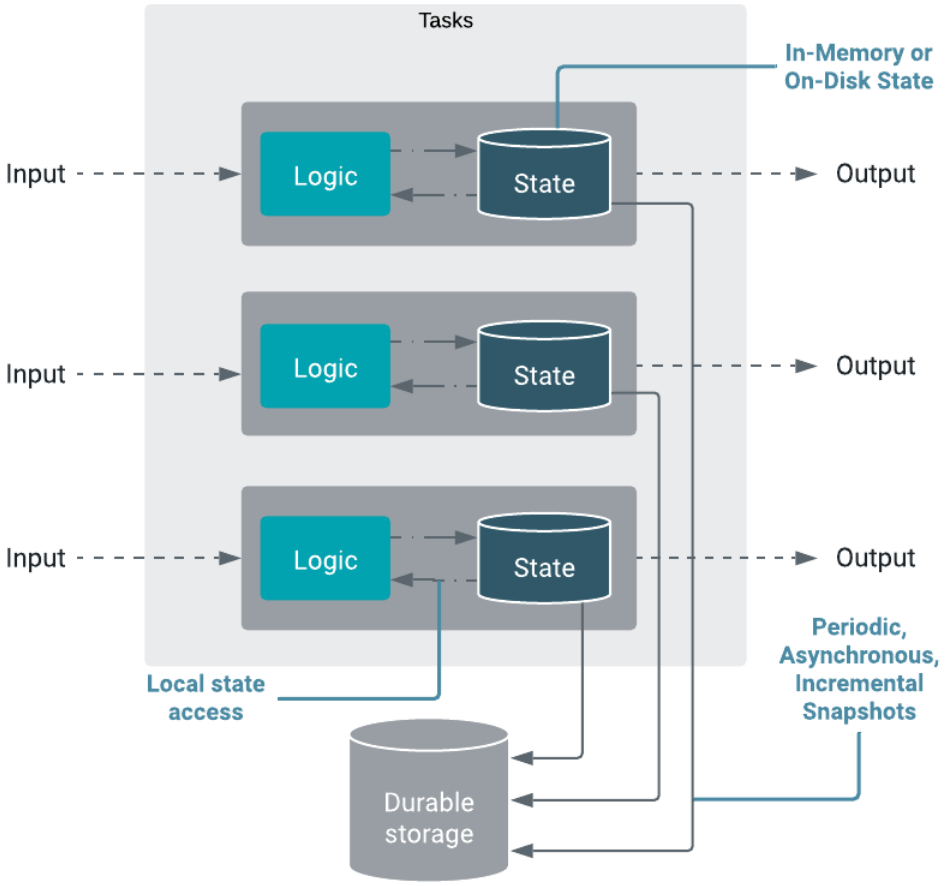


With only the event time, it is not clear when the events are processed in the application. To track the time for an event time based application, watermark can be used.



Checkpoints and savepoints

Checkpoints and savepoints can be created to make the Flink application fault tolerant throughout the whole pipeline. Flink contains a fault tolerance mechanism that creates snapshots of the data stream continuously. The snapshot includes not only the dataflow, but the state attached to it. In case of failure, the latest snapshot is chosen and the system recovers from that checkpoint. This guarantees that the result of the computation can always be consistently restored. While checkpoints are created and managed by Flink, savepoints are controlled by the user. A savepoint can be described as a backup from the executed process.



Related Information

Flink application structure
Configuring RocksDB state backend
Enabling checkpoints for Flink applications
Enabling savepoints for Flink applications