

Cloudera DataFlow Overview

Date published: 2021-04-06

Date modified: 2025-07-17

CLOUDERA

Legal Notice

© Cloudera Inc. 2025. 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

Cloudera Data Flow key features.....	4
Cloudera Data Flow key concepts.....	5
Cloudera Data Flow Architecture.....	7
Comparison of deployments and functions in Cloudera Data Flow.....	8

Cloudera Data Flow key features

Cloudera Data Flow is a cloud-native universal data distribution service powered by Apache NiFi that enables you to connect to any data source, process and deliver data to any destination. For more details on features and functionalities, see the below list.

Flow and resource isolation

Cloudera Data Flow allows you to easily isolate data flows from each other and guarantee a set of resources to each data flow without requiring administrators to create additional NiFi clusters. For each flow deployment, Cloudera Data Flow creates a dedicated, auto-scaling NiFi cluster on the shared Kubernetes resources in an environment. This way flow deployments can scale independently from each other, allowing you to isolate flow deployments and assign resources to deployments as needed.

Flow isolation can be useful when you want to guarantee a set of resources for a specific data flow or when you want to isolate failure domains.

Auto-scaling flow deployments

Cloudera Data Flow offers two types of auto-scaling capabilities for Apache NiFi data flows. Flow deployments may automatically scale up and down based on CPU utilization within the boundaries that are set in the deployment wizard. In this case Cloudera Data Flow scales flow deployments by adding or removing NiFi pods on the Kubernetes cluster as needed, as well as scaling the Kubernetes cluster up or down within boundaries specified during Cloudera Data Flow enablement.

On top of scaling based on CPU utilization, you may also enable Flow Metrics Scaling in the deployment wizard. This feature adds or removes NiFi pods to the Kubernetes cluster based on anticipated traffic on connection(s) where data first enters the flow. Scaling happens automatically, driven by a prediction algorithm, that uses a backpressure prediction metric to forecast how full a queue will be within a predefined period of time. The metric targets only connections which are attached to source processors. You do not need to configure anything during flow deployment. When Flow Metrics Scaling is enabled, both this metric and CPU utilization are considered. Whichever metric calls for higher scale will be obeyed.

Fault tolerant flow deployments

Flow deployments use persistent volumes to store NiFi repositories in a durable way. In case of an instance or pod failure, Cloudera Data Flow automatically spins up new pods and re-attaches the persistent volumes to ensure data processing continues from where it was interrupted.

Quick flow deployment with predefined ReadyFlows

You can quickly deploy a predefined set of data flows with minimal configuration called ReadyFlows. ReadyFlows provide you with an easy way to implement the most common data flow use cases.

Serverless NiFi Flows with Cloudera Data Flow Functions

Cloudera Data Flow Functions allows you to deploy NiFi flows not only as long running auto scaling Kubernetes clusters but also as functions on cloud providers' serverless compute services including AWS Lambda, Azure Functions, and Google Cloud Functions. Cloudera Data Flow Functions targets use cases that do not require always running NiFi flows, enables developers to focus more on business logic and less on operational management, and establishes a true pay for value model with a serverless architecture.

Central monitoring dashboard and KPIs

You can monitor your flow deployments across environments and cloud providers on a single dashboard. You can track important flow performance metrics by defining KPI alerts for your flow deployments.

Universal connectivity

You can connect to any data source or target using NiFi's rich processor library, including on-premise data sources, cloud data storage, cloud data warehouses, log data sources, cloud data analytics services, or cloud business process services.

Role-based access control

You can control which users are entitled to perform actions like enabling the data service, creating new flow deployments or new drafts by assigning predefined roles like Flow Administrator, Flow Developer or Flow User to individual Cloudera users or groups.

Projects allow you to further restrict access to a subset of resources that are assigned to them. Similarly, Collections allow for fine-grained access control to a subset of flow definitions within the Catalog.

Secure inbound connections

You can easily provision secure, stable, and scalable endpoints, making it easy for any application to send data to flow deployments.

Parameter groups

You can create groups of parameters and share them between data flows. Parameter groups allow you to centrally manage, share and reuse common parameters that your data flows depend on. When developing new data flows, developers and administrators alike can re-use these common parameters for a simplified development and deployment experience.

Continuous integration (CI) / Continuous deployment (CD)

The Cloudera Data Flow service is built with automation in mind. Any action that is performed on the UI can be turned into a CLI statement for automation. Deploying a new NiFi flow is as easy as executing a single CLI command.

Cloudera Data Flow key concepts

Learn about the key concepts and terms used in Cloudera Data Flow.

Catalog

The **Catalog** is where your flow definitions are stored and where you manage the Cloudera Data Flow flow definition lifecycle from import through versioning to deletion. The **Catalog** is also the place from where you can initiate new deployments.

Collection

A **Collection** is a logical container within the Cloudera Data Flow **Catalog** that groups flow definitions to enable both organization and fine-grained access control. Flow definitions assigned to a collection are only visible to users with the appropriate permissions for that collection. A flow definition can belong to only one collection at a time, and removing it from a collection makes it accessible to all users with sufficient Catalog permissions.

Deployments

The **Deployments** view is the central monitoring component within Cloudera Data Flow showing all flow deployments across environments at a glance. For each flow deployment, you can open the **Deployment Details** pane, which shows you the KPIs you have defined, system metrics, as well as system events and alerts.

Deployment Manager

The Deployment Manager allows you to review and modify flow deployment parameters, settings for size and scaling, and KPI and alert definitions. It also allows you to initiate NiFi version upgrades, access the NiFi canvas of your flow deployments as well as terminate them. Click the **Actions Manage Deployment** in the Deployment Details pane to access the Deployment Manager.

Environment

Cloudera Data Flow works in the context of Cloudera environments. You can enable the Cloudera Data Flow service for any supported environment you have registered with Cloudera. The enablement process creates the Kubernetes infrastructure required by Cloudera Data Flow and each environment maps to one Kubernetes cluster.

Once Cloudera Data Flow has been enabled for an environment, you can start deploying flow definitions to it.

Flow definition

A flow definition represents the data flow logic developed in Cloudera Data Flow's Flow Designer and published to the **Catalog**; or developed in Apache NiFi and exported by using the Download Flow Definition action on a NiFi process group or the root canvas. Flow definitions typically leverage parameterization to make the flows portable between for example development and production NiFi environments.

To run an existing NiFi data flow in Cloudera Data Flow, you have to export it as a flow definition and upload it to the Cloudera Data Flow **Catalog**.

Flow deployment

A flow deployment represents a NiFi cluster running on Kubernetes and executing a specific flow definition. When you initiate the flow deployment process from the **Catalog**, a deployment wizard helps you turn a flow definition into a flow deployment. When using the wizard, specify your environment, provide configuration parameters, auto-scaling settings and KPI definitions for your flow deployment.

Function

A function is a flow that is uploaded into the Cloudera Data Flow **Catalog** and that can be run in serverless mode by serverless cloud provider services.

KPI

Apache NiFi has multiple metrics to monitor the different statistics of the system such as memory usage, CPU usage, data flow statistics, and so on. Key Performance Indicators (KPIs) are representations of those metrics for a NiFi component in Cloudera Data Flow. They provide a critical monitoring tool for a real-time view into your data flow performance.

NiFi node

In a flow deployment, a NiFi node is a pod provisioned in the underlying Kubernetes (K8s) cluster. It does not directly relate to a Virtual Machine (VM) of the underlying K8s cluster.

You specify the allowed number of VMs in the K8s cluster when you configure the minimum and maximum number of K8s nodes, and autoscaling, while enabling Cloudera Data Flow for a Cloudera environment.

You specify the allowed number of NiFi nodes during flow deployment. These nodes are then provisioned in the K8s cluster on one or several VMs, depending on resource allocation. Auto-scaling of the NiFi nodes of a deployment may or may not trigger the auto-scaling of the underlying K8s cluster, if configured and depending on the current resources allocation.

Parameter group

A parameter group is a set of shared parameters that can be reused within the project it is currently assigned to. Using shared parameter groups facilitates flow development and flow deployment.

Project

A Project is a container for a set of Cloudera Data Flow resources that restricts visibility of resources associated with it.

ReadyFlow

A ReadyFlow is a predefined, out-of-the-box data flow which can be immediately deployed by providing a small set of required parameters.

ReadyFlow Gallery

The **ReadyFlow Gallery** is where you find all available ReadyFlows. To use a ReadyFlow, you need to add it from the **ReadyFlow Gallery** to the **Catalog** and then use it to create a flow deployment.

Resource

Flow deployments, flow drafts, parameter groups, inbound connections, custom NAR configs, and custom Python configs are collectively called resources in Cloudera Data Flow. You can view and manage them from the **Resources** view.

Workspace

The **Workspace** view displays all resources within an Environment, making it easier to switch between them and managing them.

Cloudera Data Flow Architecture

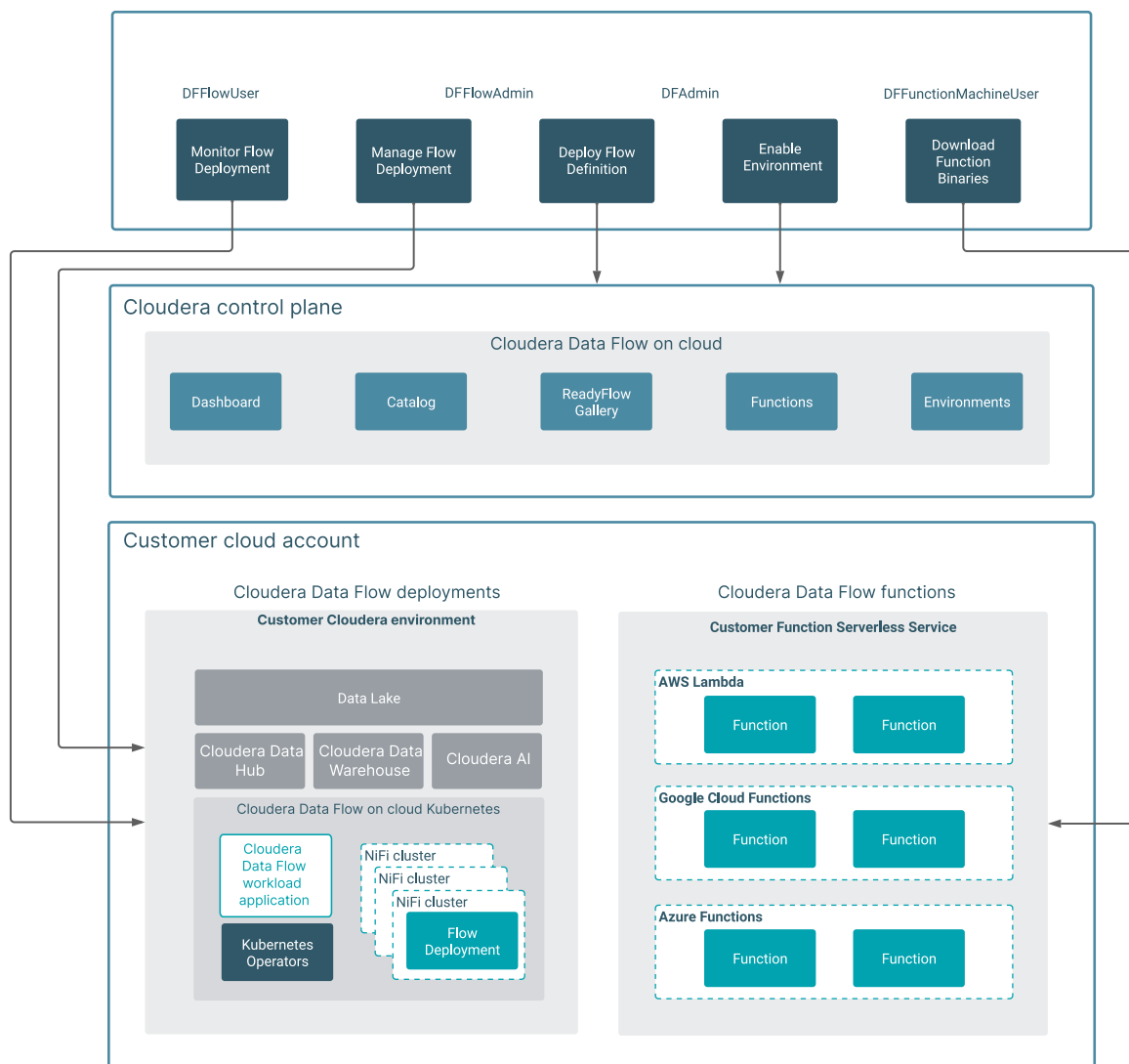
Cloudera Data Flow follows a two-tier architecture where product capabilities like the Dashboard, Catalog and Environment management are hosted on the Cloudera Control Plane while the flow deployments processing your data are provisioned in a Cloudera environment which represents infrastructure in your cloud provider account. Learn more about the service architecture, and how Cloudera Data Flow enables the various service users to achieve their goals.

When you enable Cloudera Data Flow for one of your registered Cloudera environments, Cloudera Data Flow creates and configures the required infrastructure including a Kubernetes cluster, Kubernetes Operators and the Cloudera Data Flow workload application in your cloud account. After Cloudera Data Flow has been successfully enabled for an environment, users can deploy Flow Definitions into this environment. Deploying a Flow Definition creates a dedicated NiFi cluster on Kubernetes allowing you to treat NiFi flows as isolated flow deployments.

Flow deployments run the NiFi flow logic and process data in your cloud account. Therefore data that is being processed by a flow deployment does not traverse the Cloudera Control Plane. Flow deployments send heartbeats containing health and performance information to the Control Plane where this data is visualized and presented in the Dashboard.

The Cloudera Data Flow Functions feature allows you to deploy NiFi flows stored in the Cloudera Data Flow Catalog as functions executed within AWS Lambda, Azure Functions and/or Google Cloud Functions. Leveraging Cloudera Data Flow Functions does not require to have Cloudera Data Flow enabled in a Cloudera environment. When a Cloudera Data Flow function is executed, the function will interact with the Control Plane to retrieve the flow definition and to send monitoring information.

Learn more about the specific details of Cloudera Data Flow architecture from the diagram below.



Comparison of deployments and functions in Cloudera Data Flow

This topic explains the differences between deployments and functions in the context of Cloudera Data Flow. It helps you to understand when to use one or the other from a feature as well as from a cost perspective.

Cloudera Data Flow allows you to run NiFi flows on Kubernetes clusters, providing you with an opportunity to run NiFi flows efficiently at scale. However, for a specific set of use cases, it is more beneficial to go one step further by using Cloudera Data Flow Functions. Cloudera Data Flow Functions is a Cloudera Data Flow extension to run NiFi flows as functions for event-driven use cases, where Apache NiFi provides a no-code UI for building and running functions efficiently.

Cloudera Data Flow Deployments provides a cloud-native runtime to run your Apache NiFi flows through auto-scaling Kubernetes clusters. It also provides a centralized monitoring and alerting capability that results in improved Software Development Life Cycle (SDLC) for developers. To run a Cloudera Data Flow deployment, a Kubernetes cluster (or Cloudera Data Flow environment) needs to be provisioned and at least one virtual machine needs to be

always up and running. Additional resources will be provisioned depending on your requirements and according to resource consumption by the running flows. The total cost of ownership (TCO) consists of the cloud provider's and Cloudera's costs. The cloud provider's costs include running the Kubernetes cluster through its native service and are based on the number of virtual machines being used. Cloudera's costs are based on a Concurrent User (CCU) pricing for the running flows which is based on the time period while the flows are running.

Cloudera Data Flow Functions provides a cloud-native runtime to run your Apache NiFi flows as functions on the serverless compute services of three cloud providers (AWS Lambda, Azure Functions, and Google Cloud Functions). It is particularly powerful when the flow does not require NiFi resources to be always up and running. The use cases are event-driven object store processing, microservices that power serverless web applications, IoT data processing, asynchronous API gateway request processing, batch file processing, job automation with cron/timer scheduling, and so on. For these use cases, the NiFi flows need to be treated like jobs with a distinct start and end. The start is based on a trigger event like a file landing in an object store, the start of a cron event, a gateway endpoint being invoked, and so on. Resources are only provisioned by the cloud provider for the duration required by the flow for processing the trigger event. The TCO consists of the cloud provider's costs depending on the amount and duration (in milliseconds) of resources provisioned with the function and of Cloudera's costs, based on the number of function invocations and the execution time in case the processing of a single event takes more than one second.

	Cloudera Data Flow Deployments	Cloudera Data Flow Functions
Supported cloud providers	AWS, Azure	AWS, Azure, Google Cloud
Runtime	AWS EKS Azure AKS	AWS Lambda Azure Functions Google Cloud Functions
Cloudera pricing	CCU based – Per minute for resources used by the running flows	Per invocation
Supports NiFi clustering	Yes	No
Auto-scaling	Yes – NiFi cluster scales up and down based on CPU consumption. Minimum number of one pod for a given running flow.	Yes – Based on how many concurrent events need to be processed. Multiple instances of the function can be executed at the same time.
Supports all NiFi components	Yes	Yes
Resource limitations	Yes – A given NiFi pod can be provisioned with a maximum of 12vCore and 24 GB of memory	Yes – A function instance cannot be provisioned with more than a given amount of memory and CPU (depends on the cloud provider)
Duration limitations	No – A given flow can be running forever	Yes – A function execution cannot exceed about 15 minutes (depending on the cloud provider)
Access to the NiFi UI	Yes – The NiFi UI can be accessed for a running flow	No – Cloudera Data Flow Functions is powered by Stateless NiFi which does not provide a UI
Serverless	No – There is always a minimum amount of resources up and running (for the Kubernetes cluster and for a running flow)	Yes – No infrastructure to manage and no running resources when no events to be processed
In-memory processing	No – Data going through NiFi requires writes on attached volumes	Yes – Unless specified otherwise, processing is in-memory to improve performances
Multiple sources / multiple destinations	Yes – There is no limit in the flow definition complexity	No – Cloudera Data Flow Functions is designed for simple event processing where a given event is processed and sent to one destination

From a pure technical point of view, any flow that can run as a Cloudera Data Flow function can also run as a Cloudera Data Flow deployment. Choosing one or the other depends on various considerations:

Trigger

Using Cloudera Data Flow Functions assumes that the use case has a trigger mechanism that can be implemented at the cloud provider level. Most of the use cases are covered by the existing triggers but some are not. For example, a flow that listens for events over TCP or UDP do not have a corresponding trigger available.

Cost

There are many parameters to take into account to perform an exhaustive cost analysis between the two options (the flow design, the required resources, the event execution time, and so on), but Cloudera Data Flow Functions is more cost efficient up to one million events processed per month.

SLA

Depending on how the function is configured, a cold start may happen. In a serverless architecture, a cold start refers to the required time for provisioning the resources that will run the function. With Cloudera Data Flow Functions, it is the addition of the time required for provisioning the container and the time required for Stateless NiFi to start and load the components required for executing the NiFi flow. It may go from a few seconds to a minute depending on the function's configuration. A cold start only happens when the function has not been triggered for some time (it depends on the cloud provider). It is also possible to completely remove any cold start at additional cost on the cloud provider's side.

Serverless

Cloudera Data Flow Functions relies on the cloud provider for provisioning resources when the events need to be processed. There are no infrastructure considerations required for upgrades, patches, maintenance, monitoring, and so on. Cloudera Data Flow Functions delegates the scalability to the cloud provider and can virtually scale infinitely to handle very bursty use cases.

Use cases

A few example use cases for Cloudera Data Flow Functions

- Serverless data processing pipelines: Develop and run your data processing pipelines when files are created or updated in any of the cloud object stores (for example: when a photo is uploaded to object storage, a data flow is triggered which runs image resizing code and delivers resized image to different locations to be consumed by web, mobile, and tablets).
- Serverless workflows/orchestration: Chain different low-code functions to build complex workflows (for example: automate the handling of support tickets in a call center).
- Serverless scheduled tasks: Develop and run scheduled tasks without any code on pre-defined timed intervals (for example: offload an external database running on-premises into the cloud once a day every morning at 4:00 a.m.).
- Serverless IOT event processing: Collect, process, and move data from IOT devices with serverless IOT processing endpoints (for example: telemetry data from oil rig sensors that need to be filtered, enriched, and routed to different services are batched every few hours and sent to a cloud storage staging area).
- Serverless Microservices: Build and deploy serverless independent modules that power your applications microservices architecture (for example: event-driven functions for easy communication between thousands of decoupled services that power a ride-sharing application).
- Serverless Web APIs: Easily build endpoints for your web applications with HTTP APIs without any code using Cloudera Data Flow Functions and any of the cloud providers' function triggers (for example: build high performant, scalable web applications across multiple data centers).
- Serverless Customized Triggers: With the Cloudera Data Flow Functions State feature, build flows to create customized triggers allowing access to on-premises or external services (for example: near real time offloading of files from a remote SFTP server).
- Serverless Stream Processing: Easily process messages in real time as they arrive in your messaging queue service (Kinesis, Kafka, Pub/Sub, EventHub, and so on).

For further functions case studies, see [AWS Lambda](#), [Azure Functions](#), and [Google Cloud Functions](#).

You should consider Cloudera Data Flow Deployments over Cloudera Data Flow Functions when:

- A single event (file, request, message, etc) needs to go to multiple destinations. Cloudera Data Flow Functions is better suited for use cases with a single source and a single destination.
- More than 1 million invocations / processing seconds per month are expected. In this case, Cloudera Data Flow Deployments would likely be more cost effective.
- Use cases are Listen based (other than HTTP) such as ListenTCP, ListenUDP, and so on.
- The data needs to be persisted across restarts (for example, the data source is not replayable).
- Buffering / merging of multiple events before sending to destination is required. Cloudera Data Flow Functions provides single event processing.
- Data to be processed is extremely large (multiple GB). While Cloudera Data Flow Functions can also be used, the addition of file store / ephemeral storage will incur additional costs.
- The use case cannot afford a cold start and should always offer very low latency. Cloudera Data Flow Functions can be configured with always running instances at additional cost.
- The workload would benefit from the NiFi clustering and auto-scaling capabilities.
- The user would like to use command and control features of the NiFi UI to stop and update parts of the running flow as well as to monitor the processing of the data in the NiFi UI.
- The user has a large set of use cases and running everything on a single Kubernetes cluster might provide a lower TCO.