Cloudera Data Flow

Troubleshooting

Date published: 2021-04-06
Date modified: 2025-09-30

https://docs.cloudera.com/

Legal Notice

Contents

Overview

You may encounter some common errors while using , and it is useful to understand how to recognize and correct them.

Troubleshooting errors that occur when enabling for an environment

Learn how to recognize and correct common errors that occur when you are enabling for an environment.

Troubleshooting errors that occur when is disabled for an environment

Learn how to recognize and correct common errors that occur when you are disabling for an environment.

Troubleshooting errors that occur when is enabled for an environment

Learn how to recognize and correct common errors with environments for which has been enabled.

Troubleshooting errors that occur during flow deployment

Learn how to recognize and address common errors with your flow deployments.

Troubleshooting errors that occur when enabling for an environment fails

Learn how to recognize and correct common errors that occur when you are enabling for an environment.

Your ability to deploy flow definitions depends on being enabled and in good health in your target environment. If a deployment is unhealthy in an environment, it is typically because enabling or disabling for an environment has failed. Review the environment troubleshooting information to understand common errors and their solutions.

When you enable for an existing environment, creates the required infrastructure in your cloud environment and installs core services as well as the software in the Kubernetes cluster. The enablement process can be divided into three parts:

1. Provisioning Cloud Infrastructure
2. Installing core software and services
3. Installing software and services

To understand why the enablement process is failing, the first step is to identify at which stage the enablement process failed.

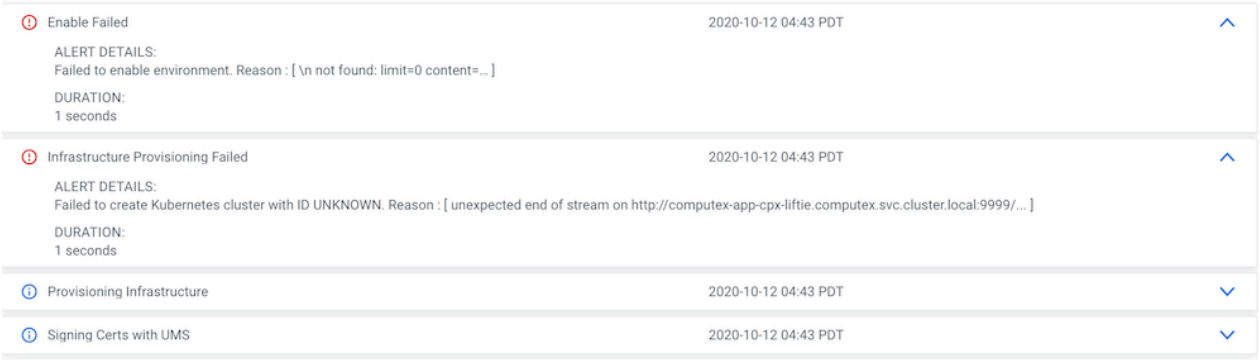
Identifying where the enablement process failed

Use the information provided in the hover state of an environment and the status messages that have been logged in the environment's Event History to identify where the enablement process has failed.

1. In DataFlow, navigate to the Environments page and find the environment where the enablement process failed.
2. Hover over the status icon and take note of the error message.
3. Click the environment, select the Alerts tab, and review the error and info events that have been logged during enablement.

Enablement fails during infrastructure provisioning or core software installation

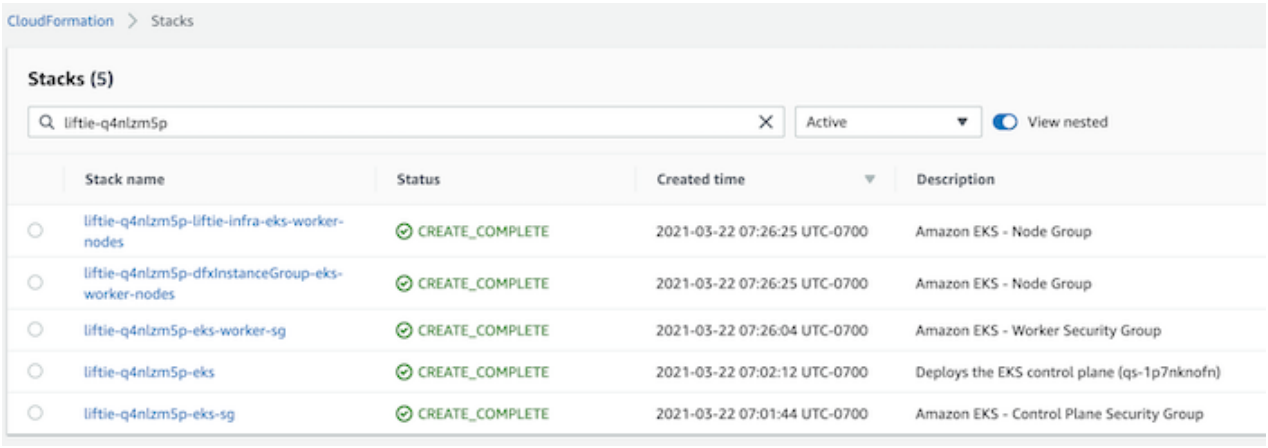
If you only see a status message in the Event History indicating that the infrastructure provisioning has started but you cannot see a corresponding status message confirming that the infrastructure has been provisioned successfully, was either not able to create the required infrastructure or install the core software and services afterwards.



The infrastructure provisioning failed error message indicates that there was either an issue with creating AWS infrastructure or setting up core services.

Validating infrastructure creation – for AWS users

uses CloudFormation scripts to create the required infrastructure in your AWS account. To validate whether the requested resources have been created successfully, log in to your AWS account, navigate to CloudFormation and search for the Kubernetes cluster ID that you have extracted from the environment events and looks similar to liftie-q4nlzm5p. Verify that the CloudFormation scripts completed successfully.

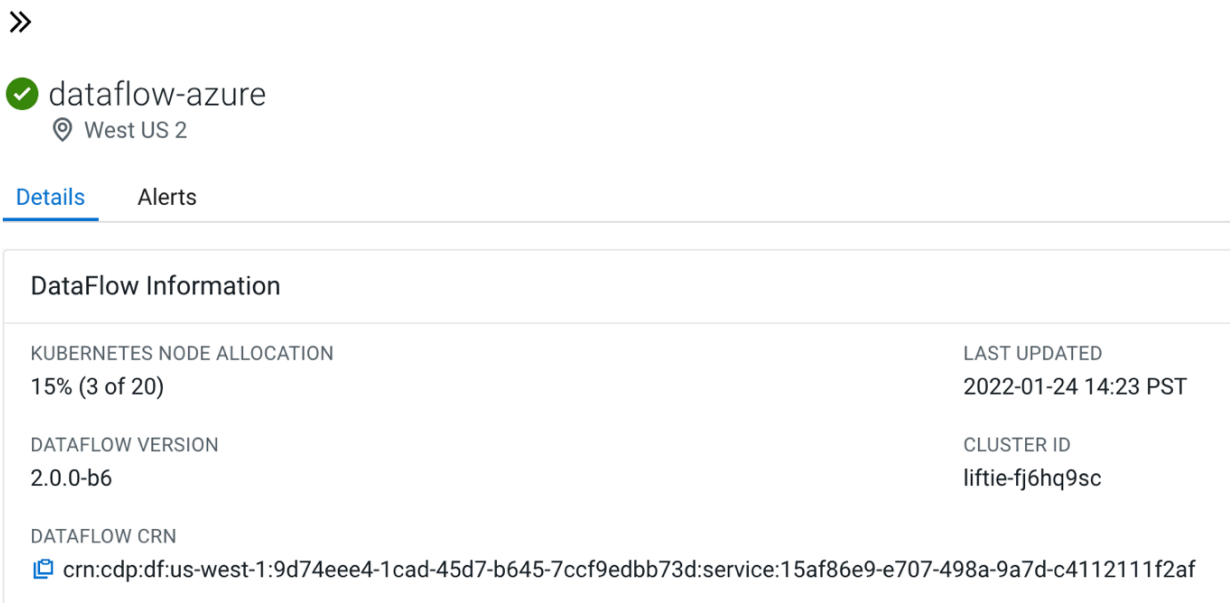


If the CloudFormation script did not complete successfully, make sure that the cross account role for your environment has been assigned appropriate permissions.

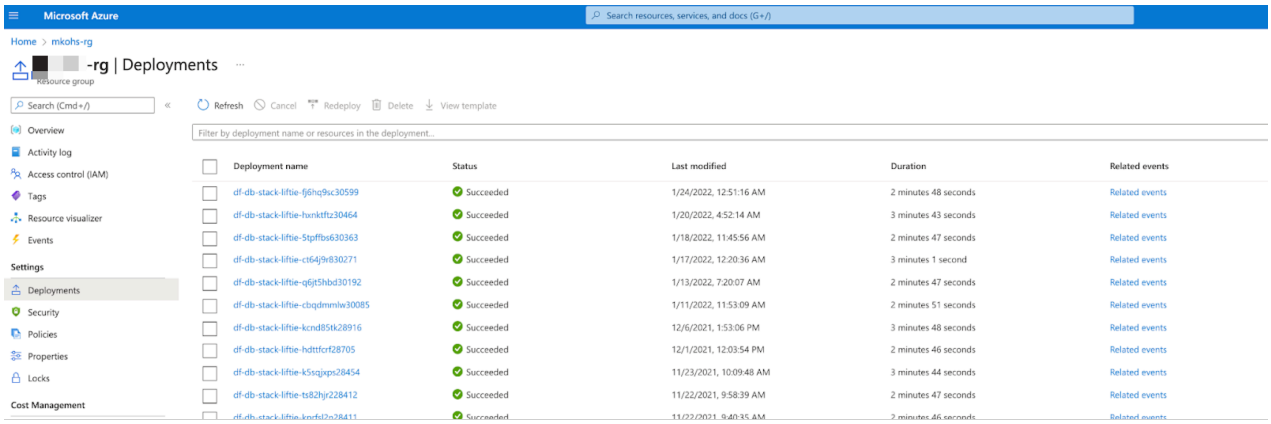
If the CloudFormation script completed successfully but enabling failed before completing the infrastructure setup, this might be an indication that the Kubernetes cluster cannot communicate with the control plane or other public endpoints like container image repositories. Make sure that the VPC and subnets you are using for meet the and prerequisites.

Validating infrastructure creation – for Azure users

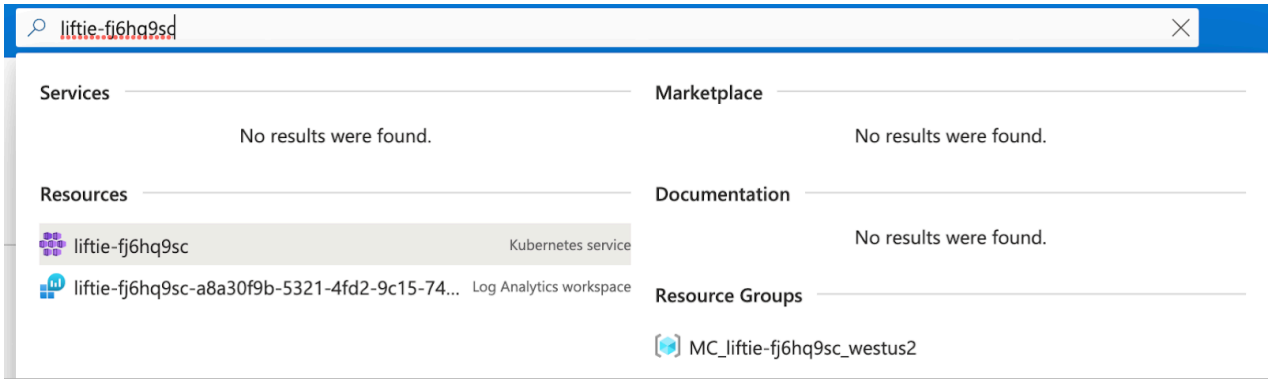
uses Azure Resource Manager to orchestrate infrastructure creation in Azure. When you enable for a environment, a Cluster ID is generated that can be used to track all associated resources in Azure. Navigate to Environments in , select your Azure environment and copy the Cluster ID, which will look similar to liftie-fq6hq9sc.



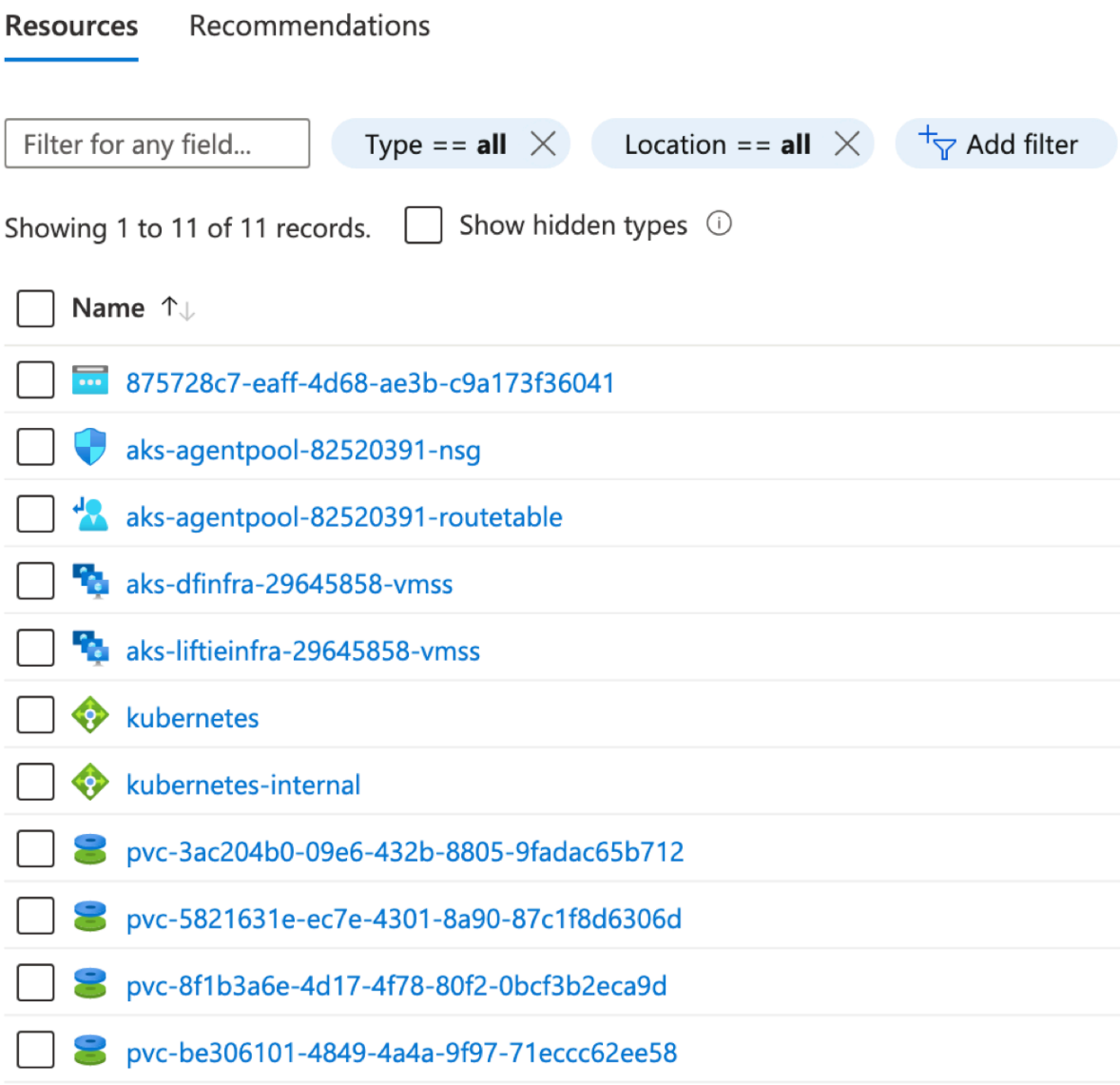
First, make sure that ’s Azure PostgreSQL database has been created successfully. In the Azure Portal, navigate to your resource group, explore Deployments under Settings, and find the database deployment associated with the previously obtained Cluster ID.



Next, validate that the AKS Kubernetes cluster and associated infrastructure has been created successfully. In the Azure Portal, use the Cluster ID to search for associated resources in your Azure subscription.



You should find a Kubernetes service as well as a new resource group called MC_<Cluster ID>_<AZURE Region>, which contains the virtual machine scale sets, load balancers, IP addresses, and storage disks that have been created for the Kubernetes cluster.



If the resources were not created successfully, make sure that the app-based credential for your environment has been assigned the appropriate permissions. For more information, see *Prerequisites for the provisioning credential*.

If the infrastructure resources were created successfully, but enabling failed before completing the infrastructure setup, it might be an indication that the Kubernetes cluster cannot communicate with the control plane or other public endpoints like container image repositories. Make sure that the virtual network and subnets you are using for meet the and prerequisites. For more information, see *VNet and subnets*.

Enablement fails during software and service installation

If you see a status message that indicates that the required Infrastructure has been provisioned successfully but the enablement process still failed, this is an indication that installing and setting up the software and services has failed.

The Infrastructure Provisioned status event indicates that the Kubernetes cluster has been created and core services have been setup successfully.

To ensure that this is not a transient issue, use the Retry Enablement action to start the enablement process again. Retry Enablement terminates all existing resources and provisions new infrastructure.

If retrying does not help and enabling still fails after successfully provisioning the infrastructure, copy the error message from the Event History and open a support case with .

Related Concepts

Overview

Related Information

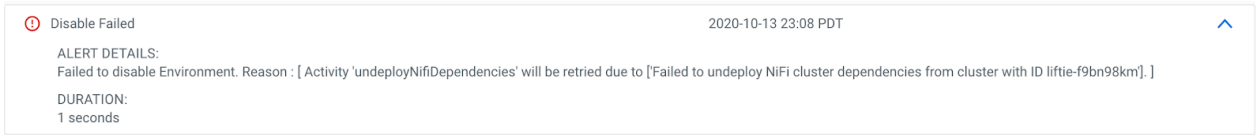
Prerequisites for the provisioning credential

VNet and subnets

Troubleshooting errors that occur when disabling for an environment fails

Learn how to recognize and correct common errors that occur when you are disabling for an environment.

can be in Bad Health due to a failed disablement process. If you see error messages in the Event History indicating that disabling has failed, use the Retry Disable Process to try again to ensure that the issue is not transient.



If the issue persists, select Environments Action Reset Environment to clear 's state for the environment. Resetting the Environment allows you to enable for the same environment again at a later stage.

Note:

Resetting does not terminate the associated cloud infrastructure. After resetting for an environment, you must manually delete the cloud resources that were created during enablement.

Related Concepts

Overview

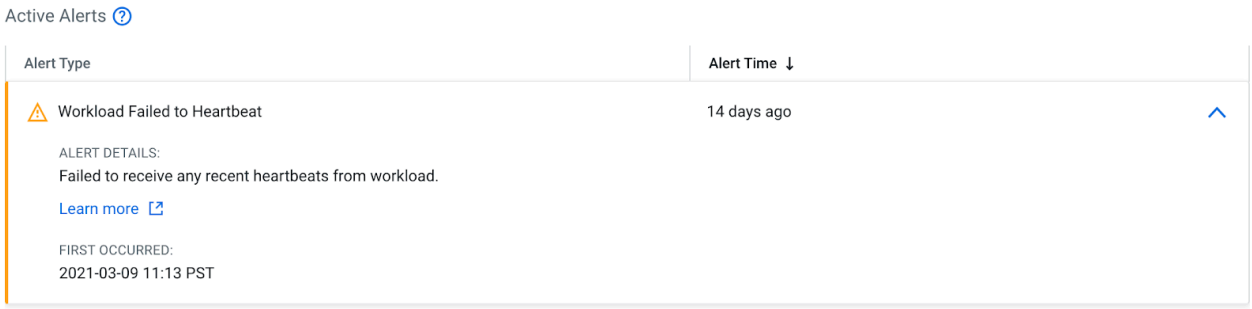
Troubleshooting errors that occur after successful enablement

Learn how to recognize and correct common errors with environments for which has been enabled.

After you have successfully enabled for an environment there are several reasons why 's health can become Concerning or Bad.

Concerning health due to “Workload Failed to Heartbeat”

If health is concerning for one of your environments, hover over the status icon and check the Alerts tab in the environment details to see details about the issue. Workload Failed to Heartbeat means that the Control Plane has not received a recent heartbeat from the workload application running in your cloud account.



fails to receive heartbeats from a particular environment

Heartbeat failures can have several reasons. Make sure that:

- The associated environment has been started and is running.
- There was no networking related change in your VPC/subnet configuration and that your networking setup still meets the requirements outlined in the *Networking*.

If the issue persists, open a support case with .

Concerning Health due to Nearing Maximum Kubernetes Limit

When your Kubernetes cluster is close to its maximum node count, will show Concerning Health for the particular environment and display an Active Alert with more details about the boundaries. To return to Good Health you can adjust the maximum node number through the Edit Configuration option in the Environment actions menu.

Bad Health due to issues with the associated environment

Certain issues with the associated environment will result in reporting Bad Health for an environment. Once is in Bad Health you can no longer create or termnate flow deployments in the environment.

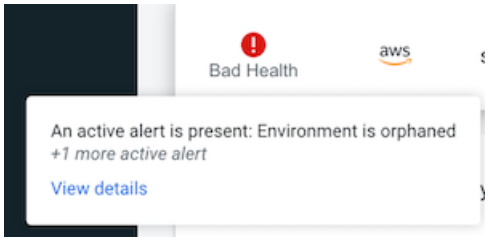
Bad Health due to environment state

If the associated environment is either unhealthy or in a starting/stopping state, will report Bad Health. To return to Good Health make sure that:

- The associated environment has been started and is running.
- FreeIPA and DataLake are both running.

Bad Health due to environment having been deleted

If the associated environment has been deleted without disabling first, will report Bad Health indicating that it has been orphaned and required services such as FreeIPA are no longer available in the environment.



You are not able to create new Flow Deployments for . You cannot recover health in this situation. Terminate your Flow Deployments and use the Disable Environment action to terminate and associated cloud infrastructure and enable again for a different environment.

Related Concepts

Overview

Troubleshooting flow deployment errors

Learn how to recognize and address common errors with your flow deployments.

Setting up `kubectl` to connect to the Kubernetes cluster

It is helpful to have access to the Kubernetes cluster using command line tools such as `kubectl` when you are troubleshooting deployment or upgrade failures. To set up `kubectl` access, follow these steps:

1. In , from the Environments page, select the service for which you want to add or remove user access.
2. Click Manage .
3. From the Actions menu, click Manage Kubernetes API Server User Access.
4. Add the AWS IAM role that you will authenticate as to the list of authorized users for by entering the ARN in the Add User dialog.
5. Use the Download Kubeconfig action to retrieve the kubeconfig file for connecting to your cluster.
6. Set up your `kubectl` to use the downloaded kubeconfig file:

```
export KUBECONFIG=[***PATH/TO/DOWNLOADED/KUBECONFIG/FILE***]
```

7. Run kubectl get ns and validate that your output looks similar to:

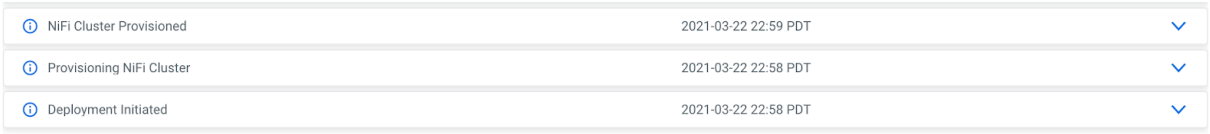
```
NAME                        STATUS   AGE
cadence                     Active   37h
cert-manager                Active   37h
cfm-operator-system         Active   37h
default                     Active   38h
dfx-dev-environment-ns      Active   36h
dfx-idbrokers3-ns           Active   36h
dfx-kafkatos3-ns            Active   22h
dfx-local                   Active   37h
dfx-ops                     Active   37h
kube-node-lease             Active   38h
kube-public                 Active   38h
kube-system                 Active   38h
liftie                      Active   37h
logging                     Active   37h
monitoring                  Active   37h
nfs-provisioner-system      Active   37h
nginx-ingress               Active   37h
prometheus-operator-system  Active   37h
```

With `kubectl` being set up correctly, you are able to access NiFi and other logs directly through the CLI.

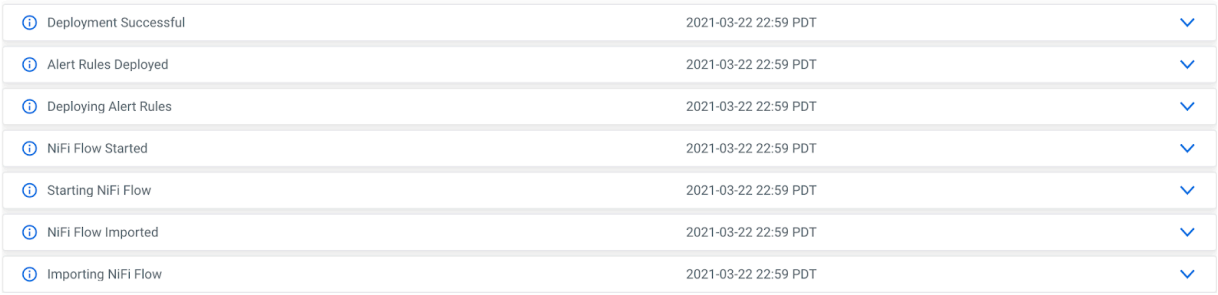
Understanding flow deployment failures

The Flow Deployment process consists of two phases:

1. Scheduling resources on Kubernetes and creating a new NiFi cluster:



2. Importing, configuring, and starting the NiFi flow definition:



If your flow deployment fails, check the Event History to see where exactly the issue occurred.

Deployment fails during Phase 1

If the issue occurs during Phase 1 while scheduling resources on Kubernetes and creating the NiFi cluster, you can get more details on why the deployment failed by looking at the application logs.

Identify the application pod by running:

```
kubectl get pods
--namespace dfx-local
```

The result should look similar to

```
NAME                                  READY   STATUS      RESTARTS   AGE
create-db-zxb2q                       0/1     Completed   0          38h
dfx-local-deployment-7f8b466c68-xwrbf 3/3     Running     0          38h
```

Copy the `dfx-local-deployment` pod ID and run to view the application logs.

```
kubectl logs \
  -f dfx-local-deployment-7f8b466c68-xwrbf \
  -c dfx-local \
  --namespace dfx-local
```

Note:

`-f` tails the log file as is writing to it.

A common reason for flow deployment issues is that the Kubernetes cluster does not have enough resources available for the deployment pods to be scheduled. Pods that are stuck in pending are an indicator for not enough resources being available.

You can explore flow deployments and their resources by running:

```
get pods --namespace dfx-deployment_name-ns
```

A healthy deployment should look similar to this with one or more NiFi pods (depending on the Sizing & Scaling settings), a ZooKeeper and a Prometheus pod.

```
NAME                        READY   STATUS    RESTARTS   AGE
dfx-nifi-0                  5/5     Running   0          22h
dfx-zookeeper-0             1/1     Running   0          22h
prometheus-prometheus-0     4/4     Running   0          22h
```

If one of the pods is stuck in Pending you can explore the pod further and identify any potential issues by looking at its events.

If in the above screenshot dfx-nifi-0 was Pending and you wanted to find out why, you would run the following command to get detailed information about the containers in the pod.

```
kubectl describe pod dfx-nifi-0 -n dfx-deployment_name-ns
```

Find the Events section and check if there are any messages about why a container could not be scheduled.

If the flow deployment failed because of insufficient resources in the Kubernetes cluster, you can increase the Kubernetes cluster size by using the Edit Configuration action of the affected environment.

Deployment fails during Phase 2

If the issue occurs during Phase 2, check the NiFi canvas of the deployment for any error messages. To get there, open the deployment details, click Manage Deployment and in the following Deployment Manager page select the view in NiFi action.

If a processor or controller service failed to start, make sure that you have provided the correct values for the deployment parameters. You can adjust parameter values in the NiFi canvas and restart processors or controller services as needed. Once you have identified the issue, note down the correct parameter values and start a new deployment.

To view the NiFi log for a particular deployment run the following `kubectl` command.

```
kubectl logs -f dfx-nifi-0 -c app-log --namespace dfx-deployment_name-ns
```

Note:

`-f` tails the log file as NiFi is writing to it.

Troubleshooting using a diagnostic bundle

You can collect diagnostic logs in a bundle and optionally send them to support either using the `start-get-diagnostics-collection` CLI command, or using the Unified Diagnostics feature of the . For more information, see *Collecting diagnostic bundle using CLI* and *Collecting diagnostic bundle using Unified Diagnostics*.

Related Concepts

Overview

Related Tasks

Collecting and sending a diagnostic bundle using Unified Diagnostics

Related reference

Collecting and sending a diagnostic bundle using the CLI

Collecting and sending a diagnostic bundle using the CLI

Learn about collecting diagnostic logs and uploading them using the CLI to facilitate troubleshooting both environment and deployment issues.

```
cdp df start-get-diagnostics-collection \
--df-service-crn '[***DATAFLOW SERVICE CRN***]' \
--description 'some description' \
--destination 'CLOUD_STORAGE' \
--environment-components '["CFM_OPERATOR", "CERT_MANAGER"]' \
--start-time [***YYYY-MM-DDTHH:MM***] \
--end-time [***YYYY-MM-DDTHH:MM***] \
--deployments '["[***CRN1***]", "[***CRN2***]"]' \
--collection-scope [***ALL|ENVIRONMENT|DEPLOYMENT***] \
--include-nifi-diagnostics [***YES|NO***]
```

--df-service-crn

Description

Provide the Cloud Resource Name (CRN) of the service for which you want to collect diagnostic data.

Possible values

Valid CRN

Importance

Required

--description

Description

Provide a free text description of the case for which diagnostic data is being collected. The provided description is persisted in the database and returned in the listing for future reference.

Possible values

Free text

Importance

Required

--destination

Description

Specify the upload destination for the generated bundle.

Possible values

- CLOUD_STORAGE - uploads the bundle to cloud storage associated with the Service environment. A bundle path is included in the listing output with the format <bucket/container>/<datalake-name>/cdf/….
- SUPPORT - uploads bundle to the UDX backend where the bundle is associated with the provided case number and is automatically attached to the case.

Importance

Required

--environment-components

Description

Specify a list of Environment Components for which you want to collect logs. This list operates as an allowed list when present, only collecting logs for specified components. If this list is not present, it implies log collection from all components.

Possible values

Environment component	Namespace
CFM_OPERATOR	cfm-operator-system
CADENCE	cadence
CERT_MANAGER	cert-manager
DFX_LOCAL	dfx-local
DFX_LOGGING	dfx-logging
FLUXCD	flux-system
NFS_PROVISIONER	nfs-provisioner-system
NGINX	nginx-ingress
REDIS	redis
VAULT	vault
VPA	vpa
ZOOKEEPER_OPERATOR	zookeeper-operator-system

Importance

Optional

--start-time

Description

Specify the time from which you want logs to be collected. No logs generated prior to the start time are collected.

If no end or start time is specified, log collection defaults to logs from the past 24 hours. Time zone is assumed to be UTC. Collection throws an error and does not begin if the provided end time is earlier than the provided start time.

Possible values

Date and time in *YYYY-MM-DD*T*HH:MM* format. For example, 2023-02-03T11:00

Importance

Optional

--end-time

Description

Specify the time to which you want logs to be collected. No logs generated after the end time are collected.

If no end or start time is specified, log collection defaults to logs from the past 24 hours. Time zone is assumed to be UTC. Collection throws an error and does not begin if the provided end time is earlier than the provided start time.

Possible values

Date and time in *YYYY-MM-DD*T*HH:MM* format. For example, 2023-02-03T11:01

Importance

Optional

--deployments

Description

Specify a list of Deployment CRNs for which logs should be collected. This list operates as an allowed list when present, only collecting logs for specified deployments. If this list is not present, it allows log collection from all Flow Deployments in the environment.

Possible values

List of valid CRNs

Importance

Optional

--collection-scope

Description

Specify the scope of data collection.

Possible values

ALL	collects logs from environment components and deployments, honoring the provided allowed lists.
ENVIRONMENT	collects logs from environment components, honoring the environment-components allowed list, and no logs from deployments except for those specified in the deployments allowed list.
DEPLOYMENT	collects logs from deployments, honoring the deployments allowed list, and no logs from environment components except for those specified in the environment-components allowed list.

Importance

Optional

--include-nifi-diagnostics

Description

Set this value to 'true' if you want to get a heap and thread dump for a diagnostic bundle. The --de ployments field defines the deployments for which the heap and thread dump are collected.

The following information is collected:

Resource	What is collected
HelmReleases	yaml, events
HelmCharts	yaml
HelmRepositories	yaml
Certificates	yaml
Values Secrets	yaml
PrometheusRules	yaml
Nifis	yaml
StatefulSets	yaml, events
Deployments	yaml, events
Services	yaml
ReplicaSets	yaml, events

Resource	What is collected
DaemonSets	yaml, events

These resources are organized in a namespaced fashion under the AdditionalClusterDetails directory. Note that the logs are stored in a different directory.

Possible values

true or false

Importance

Optional

Default collection:

```
cdp df start-get-diagnostics-collection \
--df-service-crn '[***DATAFLOW SERVICE CRN***]' \
--description 'some description' \
--destination 'CLOUD_STORAGE'
```

Support upload:

```
cdp df start-get-diagnostics-collection \
 --df-service-crn '[***DATAFLOW SERVICE CRN***]' \
 --description 'some description' \
 --destination 'SUPPORT' \
 --case-number '123456'
```

Note that case-number is required when destination is SUPPORT

List collection attempts:

```
cdp df list-diagnostics \
--df-service-crn '[***DATAFLOW SERVICE CRN***]'
```

Related Tasks

Collecting and sending a diagnostic bundle using Unified Diagnostics

Related Information

Send a diagnostic bundle using the

Collecting and sending a diagnostic bundle using Unified Diagnostics

You can trigger diagnostic bundle collection for your environment in order to send Data Lake, FreeIPA, and service and environment logs to Support for support case resolution.

Before you begin

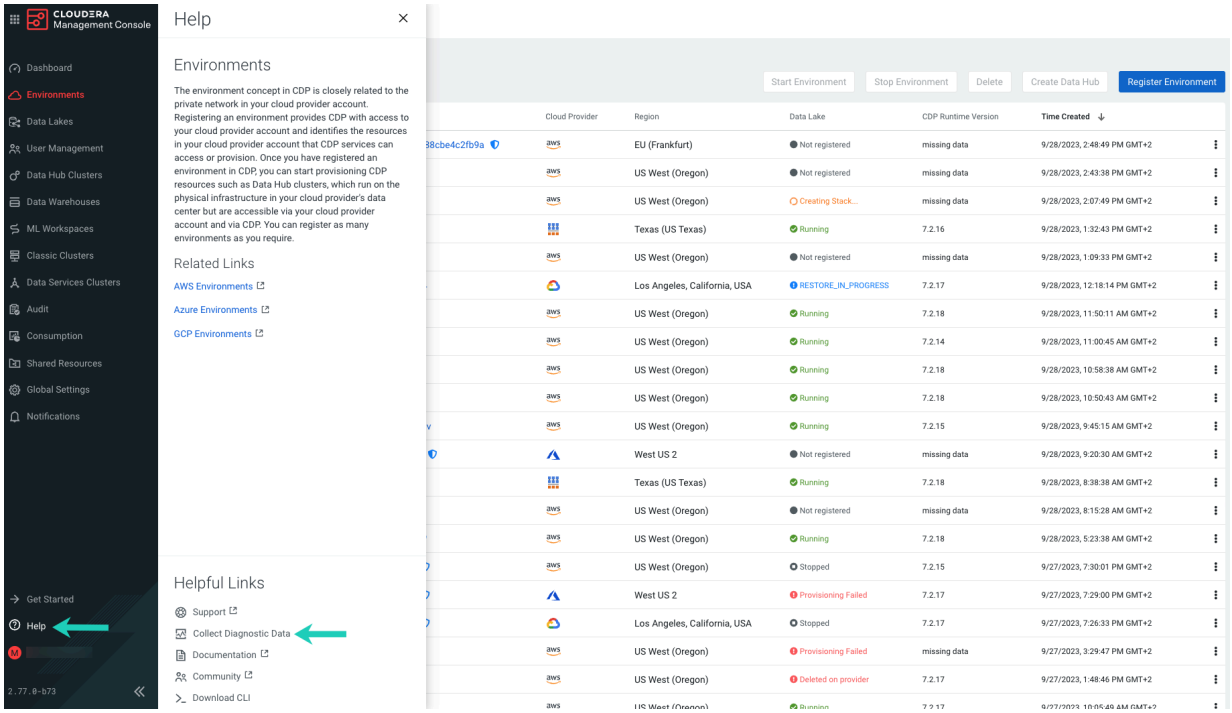
Required role: You need the EnvironmentAdmin or Owner role for the environment for which you would like to create a bundle.

- FreeIPA deployment time must be September 2020 or later.
- The VMs from which diagnostic data is to be collected must be running.
- Salt minions must be started properly on the nodes.

Procedure

1. Log in to the web interface.

2. Navigate to the Management Console.
3. Click Help in the bottom left corner and select Collect Diagnostic Data.



4. Click the Collect New Diagnostic Data.
5. On the **General Information** page, provide the following:

   Envionment

   Select the environment for which you would like to generate a diagnostic bundle. This is a required field.

   Case Number

   Enter the related support case number. You can obtain the case number from the MyCloudera portal or from your email by checking support case notification emails. This is a required field.

   Description

   Provide a free text description of the case for which you would like to generate diagnostic data. The provided description is persisted in the database and returned in the listing for future reference. This is a required field.

   Time Range

   You can specify specific points in time (date and time) from which to start and end the collection. This is an optional field.

6. Click Next.

7. On the **Selected Services** page under DataFlow, you can configure the diagnostic bundles you want to generate for environment components and deployments:

   Environment Components

   Select a list of Environment Components from which you want to collect logs. This list operates as an allowed list when present, only collecting logs for specified components.

   If you leave this list empty, it implies you want to collect logs from all components.

   Select Components

   Specify a list of Deployment CRNs for which you want to collect logs. This list operates as an allowed list when present, only collecting logs for specified deployments.

   If you leave this list empty, it implies log collection from all Flow Deployments in the environment.

   Collection Scope

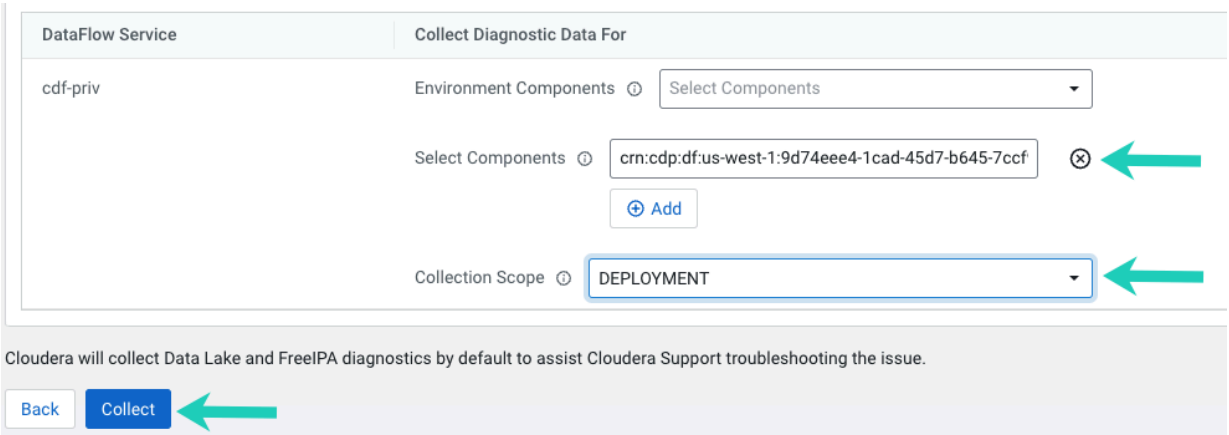
   Specify whether you want to collect environment logs, deployment logs, or both.

   Regardless of the selection you make here, logs for environment components and deployments you specified in their respective allowed lists will be collected.

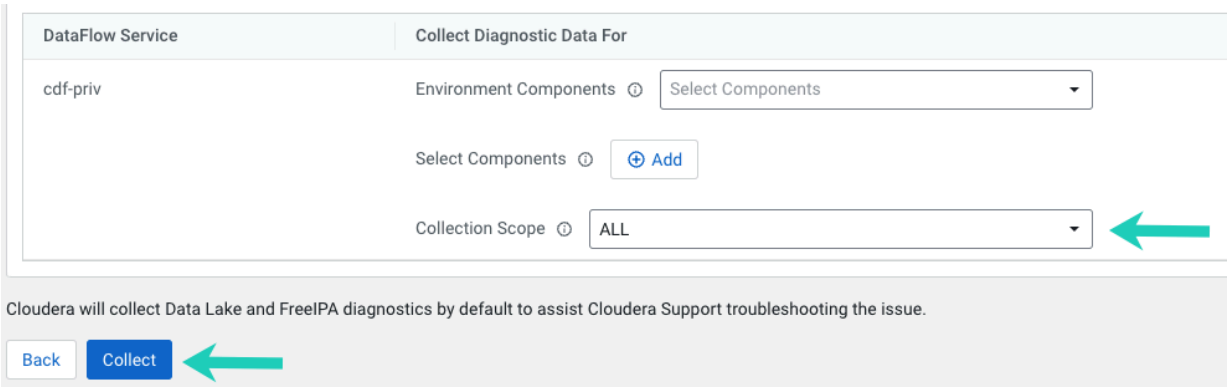
   To generate diagnostic data for a specific Deployment, enter the CRN of the Deployment for which you want to collect logs in the Select Components field and select DEPLOYMENT as Collection Scope.

   Note: To obtain the CRN of a Deployment:

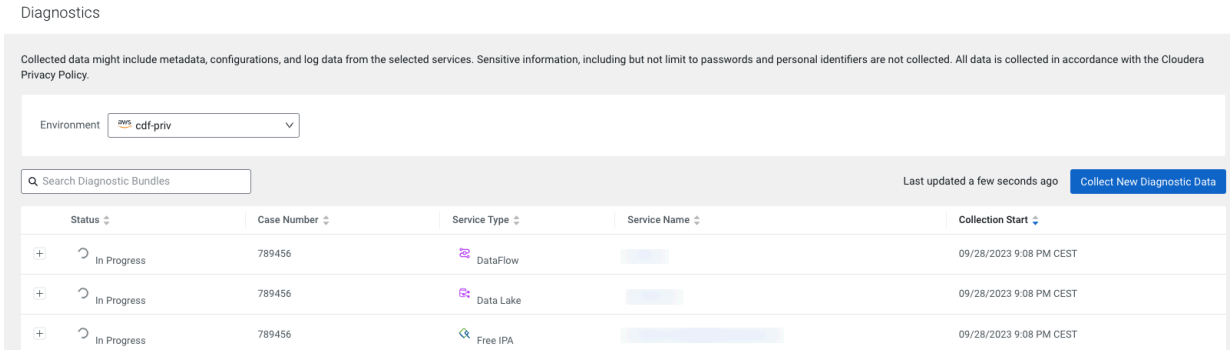
   a. Navigate to DataFlow on the web interface.
   b. On the **Dashboard** select the deployment you want to collect diagnostic data for.
   c. Click Copy under CRN #.



To obtain all diagnostic data for a Service, set Collection Scope to ALL.

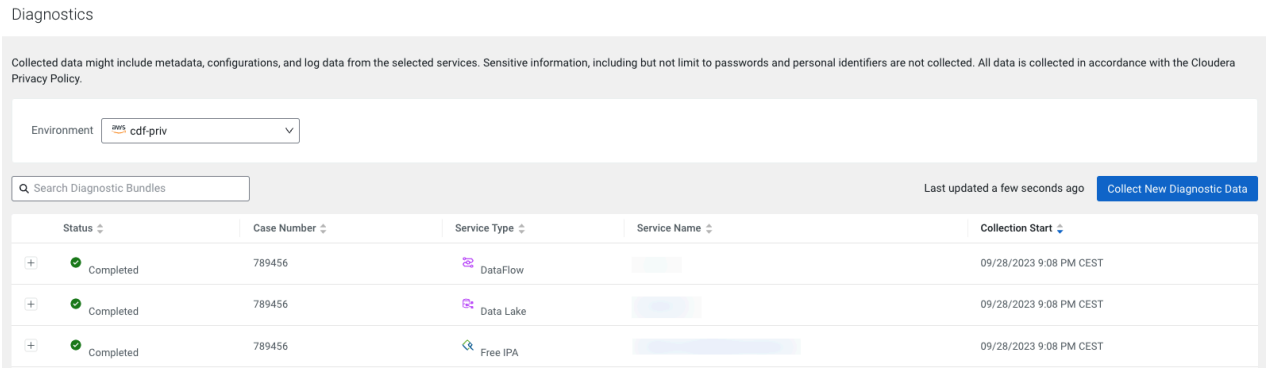


8. Click Collect.
   New entries appear on the page, allowing you to track the status of the log collection. Since a separate bundle is created for each service, there is one entry for each of the following: Data Lake, FreeIPA, and one entry for each service. While the collection is in progress, the collection status is In progress:



Results

Once the collection is complete, the status changes to Completed. This means that Support can now access the diagnostic bundles:



Related reference

Collecting and sending a diagnostic bundle using the CLI

Downloading NiFi application log

You can download the NiFi application log from the CDF Deployment Manager to use it for troubleshooting.

About this task

This feature allows you to download the NiFi application log that is currently being written. As the log file is rotated and the old file is archived once the file size reaches 10 MB, this is the theoretical maximum you can download using this method. For information on downloading archived log files, see *Diagnostic bundle collection*.

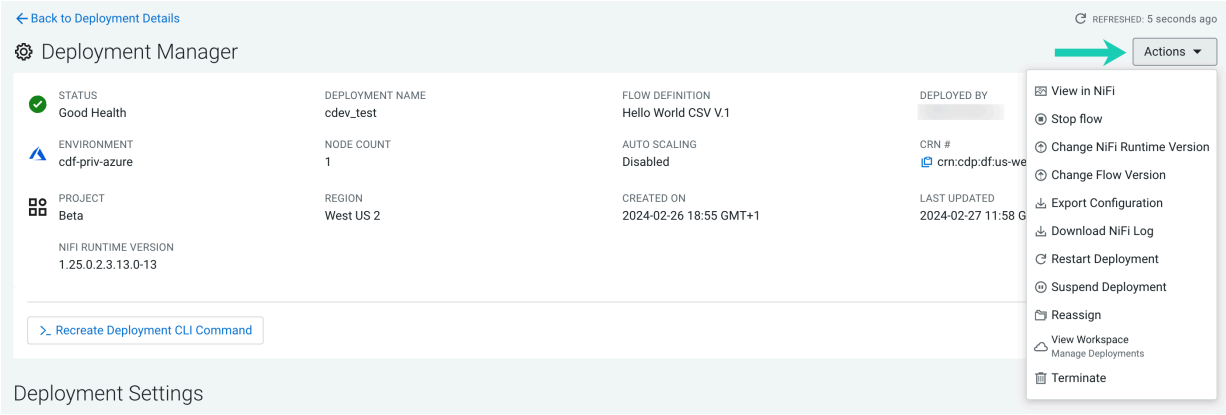
Before you begin

You need DFFlowAdmin permission to perform this action.

Procedure

1. Open by clicking the tile in the sidebar.
2. Click Deployments from the left navigation pane.
3. Select a flow deployment to expand the Deployment Details pane and get deployment, KPI, system metrics, alerts, and event history information.

4. In Deployment Details, click Actions .
   You are redirected to the **Deployment Manager** page.
5. Click Actions in the **Deployment Manager** page.



6. Select Download NiFi Log.

Results

The current NiFi application log is downloaded to your computer in tar.gz format.