Cloudera Data Engineering 1.5.2

Cloudera Data Engineering Release Notes

Date published: 2020-07-30
Date modified: 2023-11-02

CLOUDΞRA

Legal Notice

Contents

What's new in Cloudera Data Engineering Private Cloud

This release of Cloudera Data Engineering (CDE) on CDP Private Cloud 1.5.2 includes the following features:

Creating a Git repository in Cloudera Data Engineering (Technical Preview)

You can now use Git repositories to collaborate, manage project artifacts, and promote applications from lower to higher environments. Cloudera currently supports Git providers such as GitHub, GitLab, and Bitbucket. Repository files can be accessed when you create a Spark or Airflow job. You can then deploy the job and use CDE's centralized monitoring and troubleshooting capabilities to tune and adjust your workloads.

For more information, see Creating a Git repository in Cloudera Data Engineering.

Using GPUs to accelerate CDE Spark jobs and sessions (Technical Preview)

CDE supports accelerating the Spark jobs and sessions using GPUs. You can leverage the power of GPUs to benefit from faster execution times and reduced infrastructure costs without changing the existing CDE application code. By enabling GPU support, data engineers can make use of GPU resources available to the CDE service. You can configure GPU resource quota per virtual cluster which can be requested for running a Spark job or session.

For more information, see Accelerating CDE Jobs and Sessions using GPUs.

Elastic Quota for Virtual Clusters

Elastic quota for virtual clusters is now generally available (GA). You can configure elastic quota to a virtual cluster (VC) to get a minimum guaranteed and maximum capacity of resources (CPU and memory) as guaranteed quota and maximum quota. The guaranteed quota dictates the minimum amount of resources available for allocation for a VC at all times. The resources above the guaranteed quota and within the VC's maximum quota can be used by any VC on demand if the cluster capacity allows for it.

Elastic quotas allow the VC to acquire unused capacity in the cluster when their guaranteed quota limit gets exhausted. This ensures efficient use of resources in the cluster. At the same time, the maximum quota limits the threshold amount of resources a VC can claim in the cluster at any given time.

For more information, see Creating virtual clusters.

Updated CDE Home page

The new Home page of the CDE user interface has been updated with easy access to link commonly performed tasks. The new landing page now provides convenient quick-access links to create sessions and jobs, schedule a job, run an ad-hoc job, upload a DAG, build a pipeline, create new file resources and Python environment, monitor job run, and display Virtual Clusters.

View Job run timeline

You can now view the intermediate stages of the job run at every stage during its life cycle in real-time. In case of a job failure, you can view the specific event and component where the job run failed. This reduces turnaround time during the debugging process for the job run failure.

For more information, see Viewing Job run timeline.

Job log retention policy (Technical Preview)

You can now configure the job log retention policy. The retention policy lets you specify how long to retain the logs and after which the logs are deleted to save storage costs and improve performance. By default, in CDE there is no expiration period and logs are retained forever.

For more information, see Creating virtual clusters.

Support for Apache Iceberg

Apache Iceberg 1.3 is now generally available (GA) in CDE when deployed on CDP Private Cloud Base version 7.1.9 or higher. See Apache Iceberg in Cloudera Data Platform.

Apache Iceberg is in Technical Preview for Cloudera Data Engineering CDE when deployed on CDP Private Cloud Base version 7.1.7 SP2 or 7.1.8, because interoperability requirements between CDP Private Cloud Base and CDE Private Cloud are not met. Tables that are converted to Iceberg Table format can only be accessed through CDE.

Data Connectors

Data connectors are is generally available (GA). Data connectors enable you to access different storage using only a few configurations specific to storage. Data Connectors are bound to a CDE service.

For more information, see Using Ozone storage with Cloudera Data Engineering.

Hive Warehouse Connector tables

Hive Warehouse Connector (HWC) tables are now supported in Spark 3 of CDE.

Known issues and limitations in Cloudera Data Engineering on CDP Private Cloud

This page lists the current known issues and limitations that you might run into while using the Cloudera Data Engineering (CDE) service.

DEX-11300: Editing the configuration of a job created using a GIT repository shows Resources instead of Repository

> Jobs which use application file from *Repositories* when edited, shows Resources as a source under Select application file. This issue does not affect the functionality of the job but could confuse as it displays the source as a Resource for the application even if the selected file is from a repository. Though it would show Resource in this case, in the backend it is selected from the chosen repository.

DEX-11508: The file modification time in the Modified column is not updated on the Repositories page

> The GIT Repository modified time does not show the correct modified time after syncing the files. When you click Sync Repository, the syncing of files is done successfully even if the modified time shown is old.

DEX-11583: Updating the log retention time to a decimal value using API shows unclear message

> When you update the log retention time with a decimal value or an incorrect value using API, an error message similar to the following is displayed:

```
* Connection #0 to host console-cde-bt5xp3.apps.shared-os-qe-01.
kcloud.example.com left intact
{"status":"error","message":"json: cannot unmarshal number 1.5 in
to Go struct field AppInstanceLogRetentionType.config.logRetenti
on.retentionPeriod of type int"}%
```

> You must set the log retention time only with a whole number greater than 0 using API.

DEX-11340: Sessions go to unknown state if you start the CDE upgrade process before killing live Sessions

> If spark sessions are running during the CDE upgrade then they are not be automatically killed which can leave them in an unknown state during and after the upgrade.

> You must kill the running Spark Sessions before you start the CDE upgrade.

DEX-10939: Running the `prepare-for-upgrade` command puts the workload side database into read-only mode

Running the `prepare-for-upgrade` command puts the workload side database into read-only mode. If you try to edit any resources or jobs or run jobs in any virtual cluster under the CDE service for which the `prepare-for-upgrade` command was executed, The MySQL server   is running with the --read-only option so it cannot execute this statement error is displayed.

This means that all the APIs that perform `write` operations will fail for all virtual clusters. This is done to ensure that no changes are done to the data in the cluster after the `prepare-for-upgrade` command is executed, so that the new restored cluster is consistent with the old version.

You must ensure that you have sufficient time to complete the entire upgrade process before running the `prepare-for-upgrade` command.

DOCS-17844: Logs are lost if the log lines are longer than 50000 characters in fluentd

This issue occurs when the Buffer_Chunk_Size parameter for the `fluent-bit` is set to a value that is lesser than the size of the log line.

The values that are currently set are:

```
Buffer_Chunk_Size=50000
Buffer_Max_Size=50000
```

When required, you can set higher values for these parameters in the `fluent-bit` configuration map which is present in the *dex-app-xxxx* namespace.

DOCS-18585: Changes to the log retention configuration in the existing virtual cluster do not reflect the new configuration

When you edit the log retention policy configuration for an existing virtual cluster, the configuration changes are not applied.

When you edit the log retention policy configuration, you must restart the runtime-api-server pod using the `kubectl rollout restart deployment/<deployment-name> -n <namespace>` command to apply the changes.

For example:

```
kubectl rollout restart deployment/dex-app-fww6lrgm-api -n dex-a
pp-fww6lrgm
```

DEX-11231: In OpenShift, the Spark 3.3 virtual cluster creation fails due to Airflow pods crashing

This is an intermittent issue during virtual cluster installation in the OCP cluster where the airflow-scheduler and airflow-webserver pods are stuck in the *CrashLoopBackOff* state. This leads to virtual cluster installation failure.

Retry the virtual cluster installation because the issue is intermittent.

DEX-10576: Builder job does not start automatically when the resource is restored from an archive

For the airflow python environment resource, the restoration does not work as intended. Though the resource is restored, the build process is not triggered. Even if the resource was activated during backup, it is not reactivated automatically. This leads to job failure during restoration or creation, if there is a dependency on this resource.

You can use the CDE API or CLI to download the requirements.txt file and upload it to the resource. You can activate the environment if required.

```
# cde resource download --name <python-environment-name> --resou
rce-path requirements.txt
```

```
# cde resource upload --name <python-environment-name> --local-
path requirements.txt
```

DEX-10147: Grafana issue if the same VC name is used under different CDE services which share same environment

In CDE 1.5.1, when you have two different CDE services with the same name under the same environment, and you click the Grafana charts for the second CDE service, metrics for the Virtual Cluster in the first CDE service will display.

After you have upgraded CDE, you must verify other things in the upgraded CDE cluster except the data shown in Grafana. After you verified that everything in the new upgraded CDE service, the old CDE service must be deleted and the Grafana issue will be fixed.

DEX-10116: Virtual Cluster installation fails when Ozone S3 Gateway proxy is enabled

Virtual Cluster installation fails when Ozone S3 gateway proxy is enabled. Ozone s3 gateway proxy gets enabled when more than one Ozone S3 Gateway is configured in the CDP Private Cloud Base cluster.

Add the `127.0.0.1 s3proxy-<environment-name>.<private-cloud-control-plane-name>-services.svc.cluster.local` entry in the /etc/hosts of all nodes in the CDP Private Cloud Base cluster where the Ozone S3 gateway is installed. For example, if the private cloud environment name is cdp-env-1 and private cloud control plane name is cdp, then add the `127.0.0.1 s3proxy-cdp-env-1.cdp-services.svc.cluster.local` entry in /etc/hosts.

DEX-10052: Logs are not available for python environment resource builder in CDP Private Cloud

When creating a python environment resource and uploading the requirements.txt file, the python environment is built using a k8s job that runs in the cluster. These logs cannot be viewed currently for debugging purposes using CDE CLI or UI. However, you can view the events of the job.

None

DEX-10051: Spark sessions is hung at the Preparing state if started without running the cde-utils.sh script

You might run into an issue when creating a spark session without initialising the CDE virtual cluster and the UI might hang in a Preparing state.

Run the cde-utils.sh to initialise the virtual cluster as well as the user in the virtual cluster before creating a Spark long-running session.

DEX-9783: While creating the new VC, it shows wrong CPU and Memory values

When clicking on the Virtual Cluster details for a Virtual Cluster that is in the Installing state, the configured CPU and Memory values that are displayed are inaccurate for until the VC is created.

Refresh the Virtual Cluster details page to get the correct values, five minutes after the Virtual Cluster installation has started.

DEX-9692: CDE UI does not work when the port 80 is blocked on the k8s cluster

If your environment has blocked port 80 at the ingress level. Then the CDE UI does not work

None

DEX-11294: Spark UI does not render in CDE UI

Spark UI will not work for job runs that are using a Git repository as a resource.

None

DEX-9961: CDE Service installation is failing when retrieving `aws_key_id`

CDE Service installation is failing when retrieving `aws_key_id` with the Could not add shared cluster overrides,   error: unable to retrieve aws_key_id from the env service error.

1. Restart the Ozone service on the Cloudera Data Platform Base cluster and make sure all the components are healthy.

2. Create a new environment in Cloudera Data Platform Private Cloud using the Management
   Console.
3. Use the same environement for creating the CDE Service.

DEX-8996: CDE service stuck at the initialising state when a user who does not have correct permission tries to create it

When a CDE user tries to create a CDE service, it gets stuck at the initializing state and does not
fail. Additionally, cleanup cannot be done from the UI and must be done on the backend.

Only the user who has the correct permission should create a CDE service. If you experience any
issue, delete the stuck CDE service from the database.

DEX-8682: CDE PvC 1.5.0 : CDP upgrade to 1.5.0 with OCP upgrade (4.8 to 4.10) Jobs UI is not opening

Upgrading the OCP version from 4.8 to 4.10 while CDE service is enabled, causes the Jobs UI to
not open. This is due to OCP 4.10 upgrading to the Kubernetes version 1.23 which removes the old
ingress APIs used.

Back up CDE jobs in the CDE virtual cluster, and then delete the CDE service and CDE virtual
cluster. Restore it after the upgrade. For more information about backup and restore CDE jobs, see
Backing up and restoring CDE jobs.

DEX-8614: Sometimes Spark job is not getting killed even though its parent Airflow job gets killed

Sometimes if an issue is encountered while sending the request to kill a spark batch to the Livy API
and the error is logged but not propagated properly to the Airflow job. In such cases, the underlying
spark job might still be running, though the airflow job considers that the job is killed successfully.

Kill the spark job manually using CDE user interface, CLI, or API.

DEX-8601: ECS 1.4.x to 1.5.0 Upgrade: jobs fail after upgrade

Upgrading the ECS version while CDE service is enabled, causes the jobs launched in the old CDE
virtual cluster fail. This is due to ECS upgrading to the kubernetes version 1.23 which removes the
old ingress APIs used.

Back up CDE jobs in the CDE virtual cluster, and then delete the CDE service and CDE virtual
cluster. Restore it after the upgrade. For more information about backup and restore CDE jobs, see
Backing up and restoring CDE jobs.

DEX-8226: Grafana Charts of new virtual clusters will not be accessible on upgraded clusters if virtual clusters are created on existing CDE service

If you upgrade the cluster from 1.3.4 to 1.4.x and create a new virtual clusters on the existing CDE
Service, Grafana Charts will not be displayed. This is due to broken APIs.

Create a new CDE Service and a new virtual cluster on that service. Grafana Charts of the virtual
cluster will be displayed.

DEX-7000: Parallel Airflow tasks triggered at exactly same time by the user throws the 401:Unauthorized error

Error 401:Unauthorized causes airflow jobs to fail intermittently, when parallel Airflow tasks using
CDEJobRunOperator are triggered at the exact same time in an Airflow DAG.

Using the below steps, create a workaround bashoperator job which will prevent this error from
occurring. This job will keep running indefinitely as part of the workaround and should not be
killed.

1. Navigate to the Cloudera Data Engineering Overview page by clicking the Data Engineering tile
   in the Cloudera Data Platform (CDP) console.
2. In the CDE Services column, select the service containing the virtual cluster where you want to
   create the job.
3. In the Virtual Clusters column on the right, click the View Jobs icon on the virtual cluster where
   you want to create the job.
4. In the left hand menu, click Jobs.

5. Click Create Job.
6. Provide the job details:

   a. Select Airflow for the job type.
   b. Specify the job name as bashoperator-job.
   c. Save the following python script to attach it as a DAG file.

   ```
   from dateutil import parser
   from airflow import DAG
   from airflow.utils import timezone
   from airflow.operators.bash_operator import BashOperator
   default_args = {
       'depends_on_past': False,
   }
   with DAG(
       'bashoperator-job',
       default_args = default_args,
       start_date = parser.isoparse('2022-06-17T23:52:00.123Z')
   .replace(tzinfo=timezone.utc),
       schedule_interval = None,
       is_paused_upon_creation = False
       ) as dag:
       [ BashOperator(task_id = 'task1', bash_command = 'sleep
   infinity'),
        BashOperator(task_id = 'task2', bash_command = 'sleep in
   finity') ]
   ```

   d. Select File, click Select a file to upload the above python, and select a file from an existing resource.
7. Select the Python Version, and optionally select a Python Environment.
8. Click Create and Run.

DEX-7001: When Airflow jobs are run, the privileges of the user who created the job is applied and not the user who submitted the job

Irrespective of who submits the Airflow job, the Airflow job is run with the user privileges who created the job. This causes issues when the job submitter has lesser privileges than the job owner who has higher privileges.

Spark and Airflow jobs must be created and run by the same user.

Changing LDAP configuration after installing CDE breaks authentication

If you change the LDAP configuration after installing CDE, as described in Configuring LDAP authentication for CDP Private Cloud, authentication no longer works.

Re-install CDE after making any necessary changes to the LDAP configuration.

HDFS is the default filesystem for all resource mounts

For any jobs that use local filesystem paths as arguments to a Spark job, explicitly specify file:// as the scheme. For example, if your job uses a mounted resource called test-resource.txt, in the job definition, you would typically refer to it as /app/mount/test-resource.txt. In CDP Private Cloud, this should be specified as file:///app/mount/test-resource.txt.

Scheduling jobs with URL references does not work

Scheduling a job that specifies a URL reference does not work.

Use a file reference or create a resource and specify it

Limitations

Access key-based authentication will not be enabled in upgraded clusters prior to CDP PVC 1.3.4 release

After you upgrade to PVC 1.3.4 version from earlier versions, you must create the CDE Base service and Virtual Cluster again to use the new Access Key feature. Otherwise, the Access Key feature will not be supported in the CDE Base service created prior to the 1.3.4 upgrade.

Fixed issues in Cloudera Data Engineering on CDP Private Cloud

Review the list of issues that are resolved in the Cloudera Data Engineering (CDE) service in the CDP Data Services 1.5.1 release.

DEX-10055: Interacting with a killed session builder in CDP Private Cloud

With this fix, the session will no longer be unresponsive.

DEX-7513: Patching an airflow DAG keeps the total run number even though the history is not retained

With this fix, you can view the old run logs.

DEX-9895: DEX VC API response shows Spark version as 2.4.7

With this fix, in CDE 1.5.1, Spark 3.2.3 is displayed as the default version in a CDE Spark Virtual Cluster as expected.

Deprecation Notices for Airflow and Spark Python environments APIs

Certain features and functionality are deprecated in Aiflow Python environments and Spark Python environments. You must review these API changes that will be deprecated and removed in future releases.

Deprecation notice for Airflow Python environment APIs (technical preview)

Airflow Python environment (technical preview) is improved in CDE Private Cloud 1.5.2 and offers a more maintainable and reliable user experience through a revamped API. The existing resource-based approach is now deprecated and the new endpoints and new workflow should be used.

Deprecated endpoints/methods is available in the CDE API Docs.

- POST /admin/airflow/airflow-python-env is deprecated. This API exists for compatibility reasons, use the /admin/airflow/env APIs group instead, to activate an airflow python environment resource.
- GET /admin/airflow/airflow-python-env/active is deprecated. This API exists for compatibility reasons, use the /admin/airflow/env APIs group instead to get the currently activated airflow python environment name.
- For all /resources endpoint, the airflow-python-env resource type is deprecated.
- For all /resources endpoint, the pythonEnv.pyPiMirror is deprecated, use pythonEnv.pipRepository.url instead.

Deprecation Notice for Spark Python environment APIs

For improved maintainability and extendable API, the following change is made in the CDE Resource API:

- For all /resources endpoint, pythonEnv.pyPiMirror is deprecated. You must use pythonEnv.pipRepository.url instead.

Creating Cloudera Data Engineering Virtual Cluster without installing Atlas in your CDP Base cluster

If the Cloudera Data Engineering Virtual Cluster creation fails because Atlas is not installed, you must identify the CDE Namespace and set an environment variable prior to creating the Virtual Cluster.

Procedure

1. Identify the CDE Namespace

   a. In the Cloudera Data Platform (CDP) console, click the Data Engineering tile. The CDE Home page displays.

   b. In the CDE Services column, click for the CDE service you want to create a VC.

   c. Note the Cluster ID shown on the page and identify the CDE Namespace. For example, if the Cluster ID is cluster-sales8098, then the CDE Namespace is *dex-base-sales8098*.

2. Use this CDE Namespace (*dex-base-sales8098*) to run Kubernetes commands using kubectl or OpenShift's command line oc.

   kubectl

   ```
   kubectl set env deployment/dex-base-configs-manager -c dex-base-configs-
   manager ATLAS_CONFIGS_DISABLED=true --namespace <CDE Namespace>
   ```

   oc

   ```
   oc set env deployment/dex-base-configs-manager -c dex-base-configs-manager
    ATLAS_CONFIGS_DISABLED=true --namespace <CDE Namespace>
   ```

Compatibility for Cloudera Data Engineering and Runtime components

Learn about Cloudera Data Engineering (CDE) and compatibility for Runtime components across different versions.

Table 1: CDE compatibility with Runtime component details

Runtime Version	Spark 2.4.x	Spark 3.2.x	Spark 3.3.x	Airflow	Iceberg	Kubernetes
7.1.7 SP 2	• Spark 2.4.7 • Scala 2.11 • Python 2.7 • Python 3.6	• Spark 3.2.3 • Scala 2.12.10 • Python 3.6	NA	• Airflow 2.6.3 • Python 3.8	Iceberg 0.14.1	1.25
7.1.8	• Spark 2.4.7 • Scala 2.11 • Python 2.7 • Python 3.6	• Spark 3.2.3 • Scala 2.12.10 • Python 3.6	NA	• Airflow 2.6.3 • Python 3.8	Iceberg 0.14.1	1.25
7.1.9	• Spark 2.4.7 • Spark 2.4.8 • Scala 2.11 • Python 2.7 • Python 3.6	• Spark 3.2.3 • Scala 2.12.10 • Python 3.6	• Spark 3.3.2 • Scala 2.12.15 • Python 3.8	• Airflow 2.6.3 • Python 3.8	Iceberg 1.3.0	1.25