Cloudera Data Engineering 1.5.4

Spark Connect Sessions (Technical Preview)

Date published: 2020-07-30
Date modified: 2024-05-30

https://docs.cloudera.com/

Legal Notice

Contents

Spark Connect Sessions

You can learn what a Spark Connect Session is, certain known limitations and the supported Runtime component versions.

What a Spark Connect Session is

A session is an interactive short-lived development environment for running Spark commands. A Spark Connect Session is a type of CDE Session that exposes the Spark Connect interface. A Spark Connect Session allows you to connect to Spark from any remote Python environment.

Supported versions of Cloudera Runtime components

Ensure that you are using the following software versions of the Runtime components before you use Spark Connect Sessions:

- Spark 3.4.1
- CDP Runtime 7.1.8

Supported Spark Connectors

The following Spark Connectors are supported with the previously listed Runtime component versions:

- Hive
- HDFS
- Hive tables Parquet storage
- Hive tables ORC storage
- Ranger - table-level access controls

Limitations

Spark Connect Sessions do not support the following:

- Profile support: Spark Connect does not support profiles in the configuration files even though the CDE clients support "Profiles" in the configuration files.
- Documentation links within the Spark Connect UI point to incorrect documents.
- Session creation allows a mix of uppercase and lowercase letters in the session names. However, using uppercase letters causes Spark Connect Sessions to connect incorrectly. As a workaround, use only lowercase letters in session names.
- Access control support: Spark Connect Sessions do not support access control. After a session is created, anyone with access to the virtual cluster can connect to it.

Configuring Spark Connect Sessions

Learn about how to configure a Spark Connect Session with CDE.

Before you begin

Before you create a Spark Connect Session, perform the following steps:

1. Create a CDE Service.
2. Create a CDE Virtual cluster. You must select All Purpose (Tier 2) in the Virtual Cluster option and Spark 3.4.1 as the Spark version.
3. Initialize the virtual cluster.

4. Initialize users in virtual clusters.
5. If you are using an OpenShift cluster, then run the following command:

```
$ oc -n openshift-ingress-operator annotate ingresscontrollers/default i
ngress.operator.openshift.io/default-enable-http2=true
```

Procedure

1. Perform the following steps on each user's machine:
   a) Create the ~/.cde/config.yaml configuration file and add the vcluster-endpoint and cdp-endpoint parameters. This allows the client machine to identify a virtual cluster. For more information, see vcluster-endpoint and cdp-endpoint.

      For example,

      ```
      cdp-endpoint: https://console-cdp.apps.example.com
      credentials-file: /Users/user1/.cde/credentials
      vcluster-endpoint: https://ffws6v27.cde-c9b822vr.apps.example.com/dex/
      api/v1
      ```

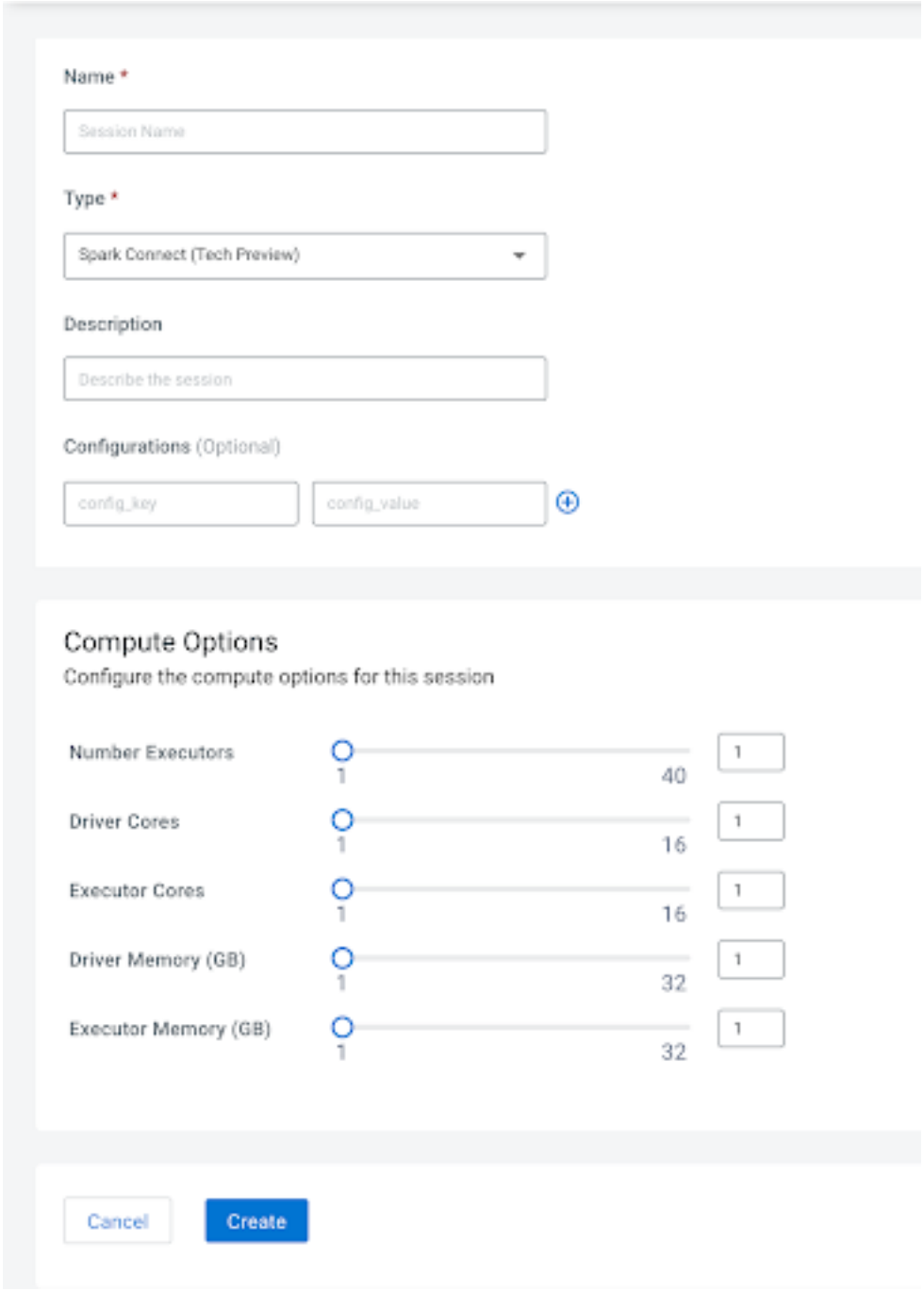
   b) Create an access key and update the credentials-file parameter in the ~/.cde/config.yaml configuration file with the path where the credentials file is located. This allows the client machine to acquire the short-lived access tokens.

      Note: Access keys configured with the default profile are supported.

      For example,

      ```
      [default]
      cdp_access_key_id=571ff....
      cdp_private_key=dvbYd....
      ```

2. Create a Spark Connect Session using one of the following methods:
   - Using the UI: Create a new session as per Creating Sessions in Cloudera Data Engineering but when you select the session type, select Spark Connect (Tech Preview) from the Type drop-down list.



   - Using the CLI: Create a Spark Connect Session by running the following command:

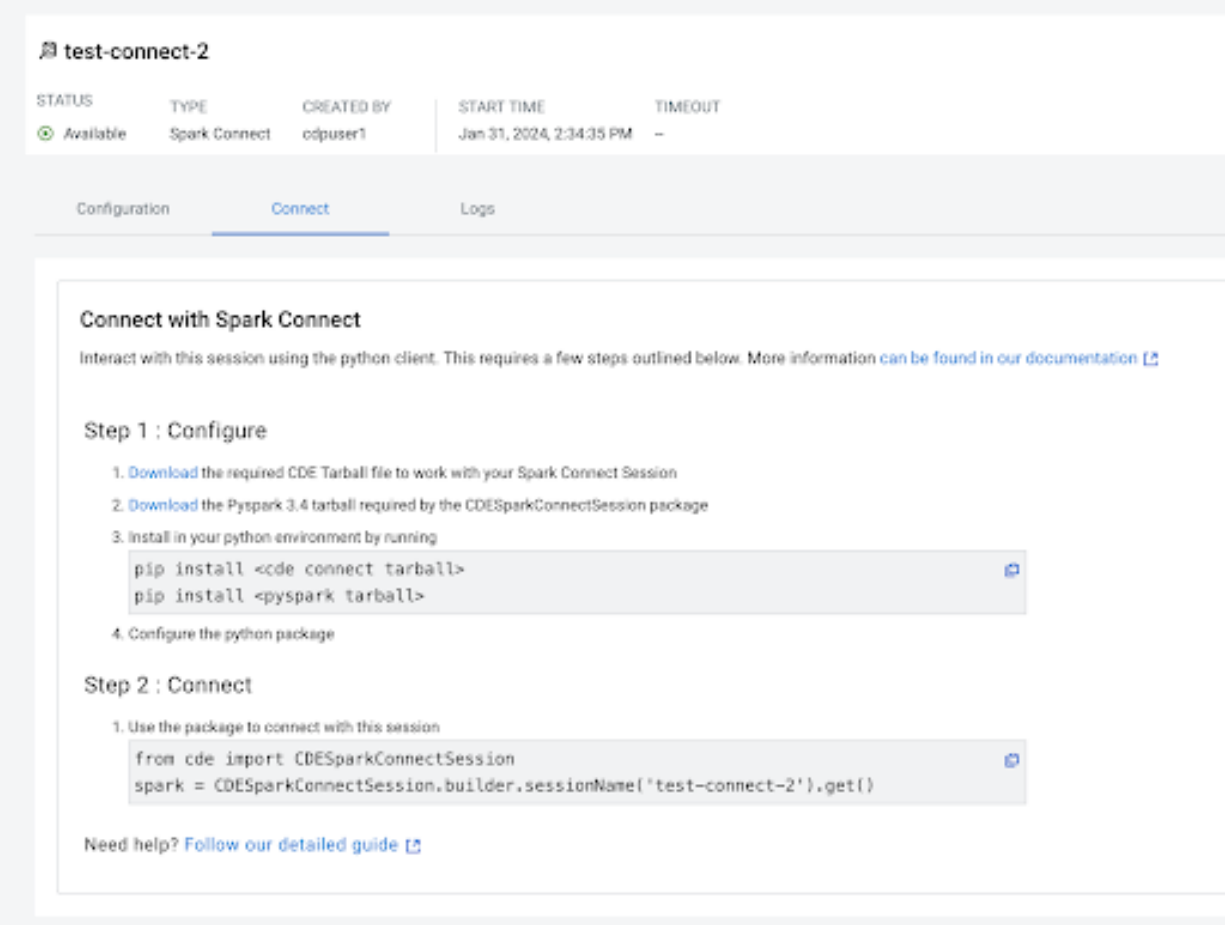
```
cde session create --name [***SPARK-SESSION-NAME***] --type spark-connec
t
```

   Note:

   To get all the attributes of a cde session command, run the `cde session -h` command.

3. On the CDE Home page, click Sessions and then select the Spark Connect Session that you have created.

4. Go to the Connect tab and download the required CDE Tar file and Pyspark 3.4 Tar file as displayed on the screen.



Note: The Copy Link option can be used to retrieve a URL and download the client using cURL.

5. Create a new Python virtual environment or use your existing one and install the Tar file after activating your Python virtual environment.

```
python3 -m venv cdeconnect
. cdeconnect/bin/activate

pip install [***cdeconnect tarball***]
pip install [***pyspark tarball***]
```

6. If you have used the self-signed certificates while Initializing the virtual cluster, then you must configure the certificates for the CDE Virtual Cluster, Spark Connect gRPC server, and the control plane hosts to be trusted. Append all the certificates belonging to those hosts to the Python "certifi cacerts ca" truststore. Usually, the path of the truststore is venv/lib/python3.7/site-packages/certifi/cacert.pem. For trusting gRPC connections, export the following variable:

Note: If you do not have self-signed certificates, manually create cacerts.pem file.

```
# In bash_profile or terminal
export GRPC_DEFAULT_SSL_ROOTS_FILE_PATH=venv/lib/python3.7/site-packages/c
ertifi/cacert.pem

# In a Jupyter notebook use the inbuilt %env magic
```

```
%env GRPC_DEFAULT_SSL_ROOTS_FILE_PATH=~/<path-to-cert>
```

Sample code to connect to Spark Connect Session

After configuring Spark Connect Sessions, learn how you can run the CLI commands from a remote Python host to connect to a session and execute Spark SQL commands through an example.

You can use the following sample code to connect to the Spark Connect session. Use the spark variable to interact with Spark as you connect to the CDE jobs or sessions.

```
> python
Python 3.9.13 (main, Jul 29 2022, 12:22:24)
[Clang 13.0.0 (clang-1300.0.27.3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from cde import CDESparkConnectSession
>>> spark = CDESparkConnectSession.builder.sessionName('connect-session').ge
t()
>>> spark.version
'3.4.1.1.20.7180.0-33'
>>> spark.sql("use retaildb").show()
++
||
++
++

>>> spark.sql("select * from products_external").show()
+----------+-------------------+--------------------+-------------------+
------------+--------------------+
|product_id|product_category_id|        product_name|product_description|pro
duct_price|       product_image|
+----------+-------------------+--------------------+-------------------+--
----------+--------------------+
|         1|                  2|Quest Q64 10 FT. ...|                   |
     59.98|http://images.acm...|
|         2|                  2|Under Armour Men'...|                   |
     129.99|http://images.acm...|
|         3|                  2|Under Armour Men'...|                   |
     89.99|http://images.acm...|
|         4|                  2|Under Armour Men'...|                   |
     89.99|http://images.acm...|
|         5|                  2|Riddell Youth Rev...|                   |
    199.99|http://images.acm...|
|         6|                  2|Jordan Men's VI R...|                   |
     134.99|http://images.acm...|
|         7|                  2|Schutt Youth Recr...|                   |
     99.99|http://images.acm...|
|         8|                  2|Nike Men's Vapor ...|                   |
    129.99|http://images.acm...|
|         9|                  2|Nike Adult Vapor ...|                   |
      50.0|http://images.acm...|
|        10|                  2|Under Armour Men'...|                   |
     129.99|http://images.acm...|
|        11|                  2|Fitness Gear 300 ...|                   |
    209.99|http://images.acm...|
|        12|                  2|Under Armour Men'...|                   |
    139.99|http://images.acm...|
|        13|                  2|Under Armour Men'...|                   |
     89.99|http://images.acm...|
|        14|                  2|Quik Shade Summit...|                   |
     199.99|http://images.acm...|
```

```
|        15|                   2|Under Armour Kids...|                    |
     59.99|http://images.acm...|
|        16|                   2|Riddell Youth 360...|                    |
    299.99|http://images.acm...|
|        17|                   2|Under Armour Men'...|                    |
    129.99|http://images.acm...|
|        18|                   2|Reebok Men's Full...|                    |
      29.97|http://images.acm...|
|        19|                   2|Nike Men's Finger...|                    |
    124.99|http://images.acm...|
|        20|                   2|Under Armour Men'...|                    |
    129.99|http://images.acm...|
+----------+-------------------+--------------------+--------------------+-
-----------+--------------------+
only showing top 20 rows
```

Troubleshooting errors when working with Spark Connect Session

While working with the Spark Connect Sessions in Cloudera Data Engineering (CDE), you might encounter errors. Learn how you can troubleshoot those errors.

Condition

If the session is killed or the driver exits due to an error when the code is being executed, Spark Connect shows the following error.

```
pyspark.errors.exceptions.connect.SparkConnectGrpcException: <_MultiThreaded
Rendezvous of RPC that terminated with:
 status = StatusCode.UNKNOWN
 details = "Stream removed"
 debug_error_string = "UNKNOWN:Error received from peer  {grpc_message:"Str
eam removed", grpc_status:2, created_time:"2024-01-31T13:28:23.35214+05:30"}
"
>
```

Remedy

Procedure

Check the actual error from the session driver logs using UI or CDE CLI.

Creating cacerts.pem file

You can create a cacerts.pem file including all the self-signed certificates in it.

About this task

You can collect all the self-signed certificates of the control plane, virtual cluster, Spark Connect, and CDE service hostnames and add all of them to a cacerts.pem file.

Procedure

1. Identify the control plane hostname.

   The control plane hostname is the host name present in the URL used to access the private cloud control plane. For example, if `https://console-cdp.apps.example.cloudera.com` is the control plane URL, then `console-cdp.apps.example.cloudera.com` is the control plane hostname.
2. Identify the virtual cluster hostname as follows:
   a) Select the CDE service containing the virtual cluster that you want to activate.
   b) Click Cluster Details.
   c) Click JOBS API URL to copy the URL to your clipboard.
   d) Paste the URL into a text editor to identify the endpoint host.

      For example, if the copied URL is `https://dfdj6kgx.cde-2cdxw5x5.apps.ecs-demo.example.com/dex/api/v1`, then the endpoint host is `dfdj6kgx.cde-2cdxw5x5.apps.ecs-demo.example.com`.
3. Identify the Spark Connect hostname: Using the virtual cluster hostname identified in the preceeding step, prefix "`sc-`" to this hostname to obtain the Spark Connect hostname. For example, if the virtual cluster hostname is `dfdj6kgx.cde-2cdxw5x5.apps.ecs-demo.example.com`, then the Spark Connect hostname is `sc-dfdj6kgx.cde-2cdxw5x5.apps.ecs-demo.example.com`.
4. Identify the CDE service hostname as follows:
   a) Select the CDE service containing the virtual cluster that you want to activate.
   b) Click Cluster Details.
   c) Click Grafana Charts to copy the URL to your clipboard.
   d) Paste the URL into a text editor to identify the endpoint host.

      For example, if the copied URL is `https://service.cde-2cdxw5x5.apps.ecs-demo.example.com/grafana/249u4dnkfnkdf` then the endpoint host is `service.cde-2cdxw5x5.apps.ecs-demo.example.com`.
5. Run the following commands to add all the certificates into a single cacerts.pem file:

```
openssl s_client -connect [***control plane hostname***]:443 2>/dev/null
 </dev/null |  sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >> c
acerts.pem
openssl s_client -connect [***virtual cluster hostname***]:443 2>/dev/null
 </dev/null |  sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >> c
acerts.pem
openssl s_client -connect [***spark connect hostname***]:443 2>/dev/null
 </dev/null |  sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >> c
acerts.pem
openssl s_client -connect [***cde service hostname***]:443 2>/dev/null </
dev/null |  sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >> cacer
ts.pem
```