TRIFACTA

Install Guide

Version: 6.4.1
Doc Build Date: 08/30/2019

For third-party license information, please select **About Trifacta** from the User menu.

1. *Install Overview . 4*
    1.1 *Install for High Availability . . 4*
    1.2 *Install On-Premises . . 8*
    1.3 *Configure Server Access through Proxy . 19*
2. *Install Software 19*
    2.1 *Install Dependencies without Internet Access . 19*
    2.2 *Install on CentOS and RHEL . 21*
    2.3 *Install on Ubuntu . 25*
    2.4 *Install for Docker 29*
    2.5 *License Key . 38*
    2.6 *Install Desktop Application 40*
    2.7 *Start and Stop the Platform . 42*
    2.8 *Login 45*
3. *Install Reference 45*
    3.1 *Install SSL Certificate 46*
    3.2 *Change Listening Port . 50*
    3.3 *Supported Deployment Scenarios for Cloudera . 51*
    3.4 *Supported Deployment Scenarios for Hortonworks . 53*
    3.5 *Uninstall 56*
4. *Configure for Hadoop . 57*
    4.1 *Configuration by Hadoop Distribution . 64*
        4.1.1 *Configure for Cloudera . 65*
            4.1.1.1 *Configure Publishing to Cloudera Navigator . 66*
        4.1.2 *Configure for Hortonworks . 72*
    4.2 *Configure Hadoop Authentication . 75*
        4.2.1 *Configure for Kerberos Integration . 78*
        4.2.2 *Configure for Secure Impersonation . 81*
    4.3 *Enable HttpFS 86*
    4.4 *Enable Integration with Compressed Clusters . 89*
    4.5 *Enable Integration with Cluster High Availability . 91*
    4.6 *Configure for Hive 95*
        4.6.1 *Configure for Hive with Sentry 103*
        4.6.2 *Configure for Hive with Ranger 106*
    4.7 *Configure for KMS 109*
        4.7.1 *Configure for KMS for Sentry 110*
        4.7.2 *Configure for KMS for Ranger 112*

Install Overview

Contents:

- *Basic Install Workflow*
- *Installation Scenarios*
  - *Install On-Premises*
  - *Install for AWS*
  - *Install for Azure*
  - *Install from AWS Marketplace*
  - *Install from AWS with EMR*
  - *Install for Azure Marketplace*
  - *Install Desktop Application*
- *Notation*

Basic Install Workflow

1. Review the pre-installation checklist and other system requirements. See *Install Preparation*.
2. Review the requirements for your specific installation scenario in the following sections.
3. Install the software. See *Install Software*.
4. Install the databases. See *Install Databases*.
5. Configure your installation.
6. Verify operations.

Notation

In this guide, JSON settings are provided in dot notation. For example, `webapp.selfRegistration` refers to a JSON block `selfRegistration` under `webapp`:

```
{
...
  "webapp": {
    "selfRegistration": true,
    ...
  }
...
}
```

Install for High Availability

Contents:

- *Limitations*
- *Overview*
  - *Job interruption*
  - *Installation Topography*
- *Order of Installation*
- *Configuration*

The Trifacta® platform can be installed across multiple nodes for high availability failover. This section describes the general process for installing the platform across multiple, highly available nodes.

- The Trifacta platform can also integrate with a highly available Hadoop cluster. For more information, see *Enable Integration with Cluster High Availability*.

Limitations

The following limitations apply to this feature:

- This form of high availability is not supported for Marketplace installations.

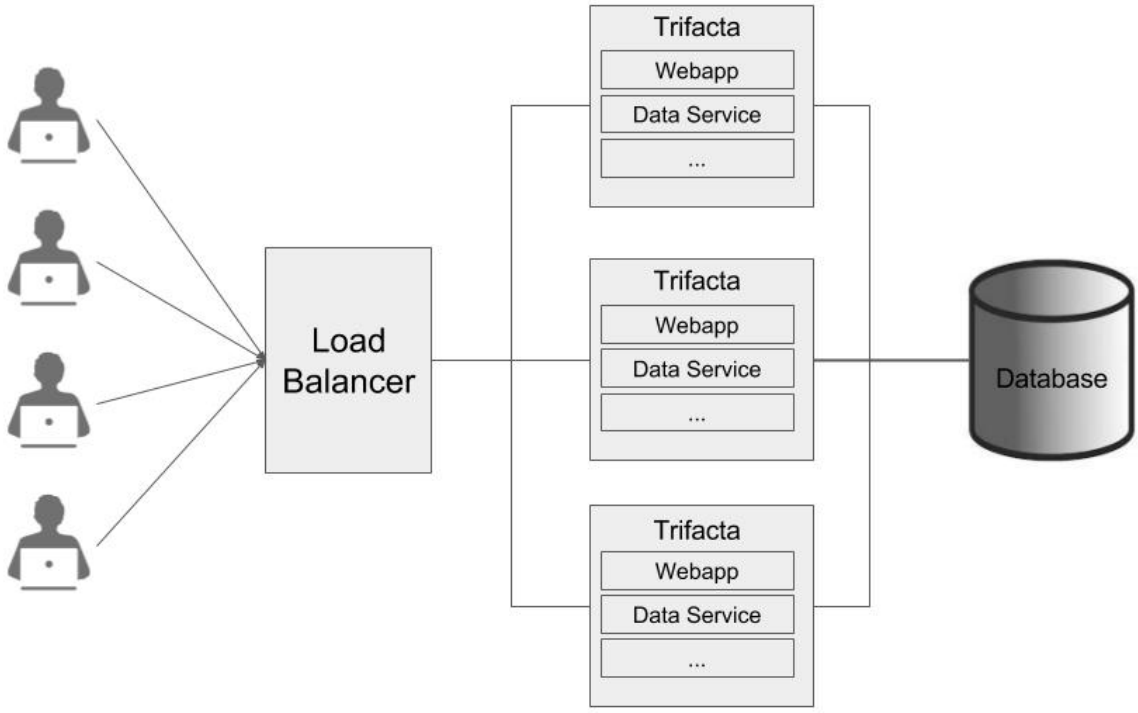
- Job canceling does not work.
- When HA is enabled, the restart feature in the Admin Settings page does not work. You must restart using the command line.
- The platform must be installed on `/opt/trifacta` on every failover node.
- This feature does not apply to the following components:
  - Hadoop cluster (See previous link.)
  - webhdfs/httpfs
  - Sentry
  - Navigator
  - Atlas
  - any other application/infrastructure with which the Trifacta platform can integrate

For more information, see *Configure for High Availability*.

Overview

The Trifacta platform supports an Active-Active HA deployment model, which works well at scale. The architecture features a single load balancer sitting in front of multiple nodes running the Trifacta platform. Each node:

- communicates with the same database
- shares the `/opt/trifacta/conf` and `/opt/trifacta/logs` directories through NFS.



- **Database:** PostGreSQL supports HA. The HA-enabled database runs outside of the cluster of platform nodes and appears to each node as a single database. No application code changes are required.
- **Load balancer:** HAProxy is used for its capabilities on health checking the other HA nodes. This load balance periodically checks the health of the other nodes in the setup.
  - If the health for a given node fails, then the load balancer stops routing traffic to that node while continuing to poll its health.
  - If the node recovers, the load balancer resumes sending traffic to it.
  - Node health is described below.
- **Synchronized configuration:** All nodes share the `/opt/trifacta/conf` mount point, which allows the same configuration files to be visible and accessible on each node.

Job interruption

In case of a failover event, any in-progress job should be marked as failed.

Failover events/scenarios around jobs:

#	Job	Event	Resulting job state
1	In progress	The batch job runner is fine, but executor running the job fails.	Failed
2	In progress	The batch job runner or the node dies.	In Progress
3	Queued	The batch job runner or the node dies.	In Progress[1]
4	Pending	The batch job runner or the node dies.	In Progress[1 2]

[1] It may not be "In Progress". However, the job has not truly failed.

2 A nuance around #3. There is a feature flag that can be enabled and is enabled by default, which causes pending jobs to be marked as failed on (re)start of batch job runner. However, because this feature indiscriminately marks *all* pending jobs as failed, it cannot be safely enabled in an environment that has multiple running batch job runners.

Installation Topography

The Trifacta platform supports a single load balancer placed in front of multiple nodes, each of which runs the same version of Trifacta Wrangler Enterprise. Content between nodes is shared using an NFS resource mount.

- **master node:** This node is the default one used for hosting and serving the Trifacta platform. Example node information:

```
NFS Server Hostname: server.local
NFS Server IP Address: 192.168.1.101
```

- **client node(s):** These nodes are failover nodes in case the master node is unavailable. Example node information:

```
NFS Client Hostname: client.local
NFS Client IP Address: 192.168.1.102
```

- **load balancer:** This documentation references set up for HAProxy as an example. If you are using a different load balancer, please consult the documentation that came with your product.

Shared resources:

Each node shares the following resources:

- Trifacta databases
- Directories shared via NFS mount:

```
/opt/trifacta/logs
/opt/trifacta/conf
```

Order of Installation

Steps:

1. All nodes must meet the system requirements. See *System Requirements*.
2. All nodes must have the appropriate ports opened. See *System Ports*.
3. Install the databases.

> **NOTE:** The databases must be installed in a location that is accessible to all nodes.

> **NOTE:** When installing databases for high availability access, you should deploy standard access and replication techniques that are consistent with the policies of your enterprise.

See *Install Databases*.

4. Complete the installation process for the server node.

   NOTE: After install, do not start the Trifacta node.

   See *Install Software*.
5. Repeat the above process for each of the client nodes.
6. The software is installed on all nodes. No node is running the software.

Configuration

Additional configuration is required.

NOTE: Starting and stopping the platform in high availability mode requires additional steps.

For more information, see *Configure for High Availability*.

Install On-Premises

Contents:

- *Scenario Description*
- *Preparation*
- *Deploy the Cluster*
  - *Prepare the cluster*
- *Deploy the Trifacta node*
- *Install Workflow*
- *Configure for Hadoop*
  - *Apply cluster configuration files - non-edge node*
  - *Apply cluster configuration files - edge node*
  - *Modify Trifacta configuration changes*
  - *Configure Spark Job Service*
  - *Configure Spark*
  - *Enable High Availability*
- *Configure for Trifacta platform*
  - *Set base storage layer*
- *Verify Operations*
  - *Prepare Your Sample Dataset*
  - *Store Your Dataset*
  - *Verification Steps*
- *Documentation*
- *Next Steps*

To install Trifacta® Wrangler Enterprise inside your enterprise infrastructure, please review and complete the following sections in the order listed below.

Scenario Description

- Installation of Trifacta Wrangler Enterprise on a server on-premises

- Installation of Trifacta databases on a server on-premises
- Integration with a supported Hadoop cluster on premises.
- Base storage layer of HDFS

Preparation

1. **Review Planning Guide:** Please review and verify *Install Preparation* and sub-topics.
2. **Acquire Assets:** Acquire the installation package for your operating system and your license key. For more information, contact *Trifacta Support*.
    1. If you are completing the installation without Internet access, you must also acquire the offline versions of the system dependencies. See *Install Dependencies without Internet Access*.
3. **Deploy Hadoop cluster:** In this scenario, the Trifacta platform does not create a Hadoop cluster. See below.

    > **NOTE:** Installation and maintenance of a working Hadoop cluster is the responsibility of the Trifacta Wrangler Enterprise customer. Guidance is provided below on the requirements for integrating the platform with the cluster.

4. **Deploy Trifacta node:**Trifacta Wrangler Enterprise must be installed on an edge node of the cluster. Details are below.

Limitations: For more information on limitations of this scenario, see *Product Limitations* in the *Install Preparation* area.

Deploy the Cluster

In your enterprise infrastructure, you must deploy a cluster using a supported version of Hadoop to manage the expected data volumes of your Trifacta jobs. For more information on suggested sizing, see *Sizing Guidelines* in the *Install Preparation* area.

When you configure the platform to integrate with the cluster, you must acquire information about the cluster configuration. For more information on the set of information to collect, see *Pre-Install Checklist* in the *Install Preparation* area.

> **NOTE:** By default, smaller jobs are executed on the Trifacta Photon running environment . Larger jobs are executed using Spark on the integrated Hadoop cluster. Spark must be installed on the cluster. For more information, see *System Requirements* in the *Install Preparation* area.

The Trifacta platform supports integration with the following cluster types. For more information on the supported versions, please see the listed sections below.

- See *Supported Deployment Scenarios for Cloudera*.
- See *Supported Deployment Scenarios for Hortonworks*.

Prepare the cluster

Before installing software, please complete the following steps if you are integrating with a Hadoop cluster.

Before you begin, please verify or complete the following:

1. On the Hadoop cluster:
    1. Create a user [`hadoop.user` (default=`trifacta`)] and a group for it [`hadoop.group` (default=`trifactausers`)].
    2. Create the following directories:
        1. `/trifacta`

2. `/user/trifacta`
3. Change the ownership of `/trifacta` and `/user/trifacta` to `trifacta:trifacta` or the corresponding values for the Hadoop user in your environment.

> **NOTE:** You must verify that the `[hadoop.user]` user has complete ownership and full access to Read, Write and Execute on these directories recursively.

2. Verify that WebHDFS is configured and running on the cluster.

3. Software installation is completed on a dedicated node in the cluster. The user installing the Trifacta software must have sudo access.

4. If you are installing on a server with an older instance of Postgres, you should remove the older version or change the default ports.

For more information, see *Prepare Hadoop for Integration with the Platform*.

Additional users may be required. For more information, see *Required Users and Groups* in the *Install Preparation* area.

Deploy the Trifacta node

An edge node of the cluster is required to host the Trifacta platform software. For more information on the requirements of this node, see *System Requirements*.

Install Workflow

Please complete these steps listed in order:

1. **Install software:** Install the Trifacta platform software on the cluster edge node. See *Install Software*.
2. **Install databases:** The platform requires several databases for storage.

   > **NOTE:** The default configuration assumes that you are installing the databases on a PostgreSQL server on the same edge node as the software using the default ports. If you are changing the default configuration, additional configuration is required as part of this installation process.

   For more information, see *Install Databases*.
3. **Start the platform:** For more information, see *Start and Stop the Platform*.
4. **Login to the application:** After software and databases are installed, you can login to the application to complete configuration:
   1. See *Login*.
   2. As soon as you login, you should change the password on the admin account. In the left menu bar, select **Settings > Admin Settings**. Scroll down to Manage Users. For more information, see *Change Admin Password*.

> **Tip:** At this point, you can access the online documentation through the application. In the left menu bar, select **Help menu > Product Docs**. All of the following content, plus updates, is available online. See Documentation below.

Configure for Hadoop

After you have performed the base installation of the Trifacta® platform, please complete the following steps if you are integrating with a Hadoop cluster.

Apply cluster configuration files - non-edge node

> **NOTE:** Installation on a non-edge node is not supported. Legacy customers can continue to use a non-edge node, but this deployment is not recommended.

If the Trifacta platform is being installed on a non-edge node, you must copy over the Hadoop Client Configuration files from the cluster.

> **NOTE:** When these files change, you must update the local copies. For this reason, it is best to install on an edge node.

1. Download the Hadoop Client Configuration files from the Hadoop cluster. The required files are the following:
    1. `core-site.xml`
    2. `hdfs-site.xml`
    3. `mapred-site.xml`
    4. `yarn-site.xml`
    5. `hive-site.xml` (if you are using Hive)
2. These configuration files must be moved to the Trifacta deployment. By default, these files are in `/etc/hadoop/conf`:

```
sudo cp <location>/*.xml /opt/trifacta/conf/hadoop-site/
sudo chown trifacta:trifacta /opt/trifacta/conf/hadoop-site/*.xml
```

For more information, see *Configure for Hadoop*.

Apply cluster configuration files - edge node

If the Trifacta platform is being installed on an edge node of the cluster, you can create a symlink from a local directory to the source cluster files so that they are automatically updated as needed.

1. Navigate to the following directory on the Trifacta node:

```
cd /opt/trifacta/conf/hadoop-site
```

2. Create a symlink for each of the Hadoop Client Configuration files referenced in the previous steps. Example:

```
ln -s /etc/hadoop/conf/core-site.xml core-site.xml
```

3. Repeat the above steps for each of the Hadoop Client Configuration files.

For more information, see *Configure for Hadoop*.

Modify Trifacta configuration changes

1. To apply this configuration change, login as an administrator to the Trifacta node. Then, edit `trifacta-conf.json`. Some of these settings may not be available through the *Admin Settings Page*. For more information, see *Platform Configuration Methods*.
2. **HDFS:** Change the host and port information for HDFS as needed. Please apply the port numbers for your distribution:

   ```
   "hdfs.namenode.host": "<namenode>",
   "hdfs.namenode.port": <hdfs_port_num>
   "hdfs.yarn.resourcemanager": {
   "hdfs.yarn.webappPort": 8088,
   "hdfs.yarn.adminPort": 8033,
   "hdfs.yarn.host": "<resourcemanager_host>",
   "hdfs.yarn.port": <resourcemanager_port>,
   "hdfs.yarn.schedulerPort": 8030
   ```

   For more information, see *Configure for Hadoop*.

3. Save your changes and restart the platform.

Configure Spark Job Service

The Spark Job Service must be enabled for both execution and profiling jobs to work in Spark.

Below is a sample configuration and description of each property. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*..

```
"spark-job-service" : {
  "systemProperties" : {
    "java.net.preferIPv4Stack": "true",
    "SPARK_YARN_MODE": "true"
  },
  "sparkImpersonationOn": false,
  "optimizeLocalization": true,
  "mainClass": "com.trifacta.jobserver.SparkJobServer",
  "jvmOptions": [
"-Xmx128m"
],
  "hiveDependenciesLocation": "%(topOfTree)s/hadoop-deps/cdh-5.4/build
/libs",
  "env": {
    "SPARK_JOB_SERVICE_PORT": "4007",
    "SPARK_DIST_CLASSPATH": "",
    "MAPR_TICKETFILE_LOCATION": "<MAPR_TICKETFILE_LOCATION>",
    "MAPR_IMPERSONATION_ENABLED": "0",
    "HADOOP_USER_NAME": "trifacta",
    "HADOOP_CONF_DIR": "%(topOfTree)s/conf/hadoop-site/"
  },
  "enabled": true,
  "enableHiveSupport": true,
  "enableHistoryServer": false,
  "classpath": "%(topOfTree)s/services/spark-job-server/server/build/libs
/spark-job-server-bundle.jar:%(topOfTree)s/conf/hadoop-site/:%(topOfTree)
s/services/spark-job-server/build/bundle/*:%(topOfTree)s/%
(hadoopBundleJar)s",
  "autoRestart": false,
},
```

The following properties can be modified based on your needs:

NOTE: Unless explicitly told to do so, do not modify any of the above properties that are not listed below.

Property	Description
`sparkImpersonationOn`	Set this value to `true`, if secure impersonation is enabled on your cluster. See *Configure for Secure Impersonation*.
`jvmOptions`	This array of values can be used to pass parameters to the JVM that manages Spark Job Service.
`hiveDependenciesLocation`	If Spark is integrated with a Hive instance, set this value to the path to the location where Hive dependencies are installed on the T rifacta node. For more information, see *Configure for Hive*.
`env.SPARK_JOB_SERVICE_PORT`	Set this value to the listening port number on the cluster for Spark. Default value is `4007`. For more information, see *System Ports*.
`env.HADOOP_USER_NAME`	The username of the Hadoop principal used by the platform. By default, this value is `trifacta`.

`env.HADOOP_CONF_DIR`	The directory on the Trifacta node where the Hadoop cluster configuration files are stored. Do not modify unless necessary.
`enabled`	Set this value to `true` to enable the Spark Job Service.
`enableHiveSupport`	See below.

After making any changes, save the file and restart the platform. See *Start and Stop the Platform*.

Configure service for Hive

Depending on the environment, please apply the following configuration changes to manage Spark interactions with Hive:

Environment	spark.enableHiveSupport
Hive is not present	`false`
Hive is present but not enabled.	`false`
Hive is present and enabled	`true`

If Hive is present on the cluster and either enabled or disabled: the `hive-site.xml` file must be copied to the correct directory:

```
cp /etc/hive/conf/hive-site.xml /opt/trifacta/conf/hadoop-site/hive-site.
xml
```

At this point, the platform only expects that a `hive-site.xml` file has been installed on the Trifacta node . A valid connection is not required. For more information, see *Configure for Hive* .

Configure Spark

After the Spark Job Service has been enabled, please complete the following sections to configure it for the Trifact a platform.

Yarn cluster mode

All jobs submitted to the Spark Job Service are executed in YARN cluster mode. No other cluster mode is supported for the Spark Job Service.

Configure access for secure impersonation

The Spark Job Service can run under secure impersonation. For more information, see *Configure for Secure Impersonation*.

When running under secure impersonation, the Spark Job Service requires access to the following folders. Read, write, and execute access must be provided to the Trifacta user and the impersonated user.

Folder Name	Platform Configuration Property	Default Value	Description
Trifacta Libraries folder	`"hdfs.pathsConfig.libraries"`	`/trifacta/libraries`	Maintains JAR files and other libraries required by Spark. No sensitive information is written to this location.

Trifacta Temp files folder	`"hdfs.pathsConfig.tempFiles"`	`/trifacta/tempfiles`	Holds temporary progress information files for YARN applications. Each file contains a number indicating the progress percentage. No sensitive information is written to this location.
Trifacta Dictionaries folder	`"hdfs.pathsConfig.dictionaries"`	`/trifacta/dictionaries`	Contains definitions of dictionaries created for the platform.

Identify Hadoop libraries on the cluster

The Spark Job Service does not require additional installation on the Trifacta node or on the Hadoop cluster. Instead, it references the spark-assembly JAR that is provided with the Trifacta distribution.

This JAR file does not include the Hadoop client libraries. You must point the Trifacta platform to the appropriate libraries.

Steps:

1. In platform configuration, locate the `spark-job-service` configuration block.
2. Set the following property:

```
"spark-job-service.env.HADOOP_CONF_DIR":
"<path_to_Hadoop_conf_dir_on_Hadoop_cluster>",
```

Property	Description
spark-job-service.env.HADOOP_CONF_DIR	Path to the Hadoop configuration directory on the Hadoop cluster.

3. In the same block, the `SPARK_DIST_CLASSPATH` property must be set depending on your Hadoop distribution.
    1. **For Cloudera 5.x:** This property can be left blank.
    2. **For Hortonworks 2.x:** This property configuration is covered later in this section.
4. Save your changes.

Locate Hive dependencies location

If the Trifacta platform is also connected to a Hive instance, please verify the location of the Hive dependencies on the Trifacta node. The following example is from Cloudera 5.10:

NOTE: This parameter value is distribution-specific. Please update based on your Hadoop distribution.

```
"spark-job-service.hiveDependenciesLocation":"%(topOfTree)s/hadoop-deps
/cdh-5.10/build/libs",
```

For more information, see *Configure for Spark*.

Enable High Availability

> **NOTE:** If high availability is enabled on the Hadoop cluster, it must be enabled on the Trifacta platform, even if you are not planning to rely on it. See *Enable Integration with Cluster High Availability*.

Configure for Trifacta platform

Set base storage layer

The platform requires that one backend datastore be configured as the base storage layer. This base storage layer is used for storing uploaded data and writing results and profiles.

> **NOTE:** By default, the base storage layer for Trifacta Wrangler Enterprise is set to HDFS. You can change it now, if needed. After this base storage layer is defined, it cannot be changed again.

See *Set Base Storage Layer*.

Verify Operations

> **NOTE:** You can try to verify operations using the Trifacta Photon running environment at this time. While you can also try to run a job on the Hadoop cluster, additional configuration may be required to complete the integration. These steps are listed under Next Steps below.

Prepare Your Sample Dataset

To complete this test, you should locate or create a simple dataset. Your dataset should be created in the format that you wish to test.

Characteristics:

- Two or more columns.
- If there are specific data types that you would like to test, please be sure to include them in the dataset.
- A minimum of 25 rows is required for best results of type inference.
- Ideally, your dataset is a single file or sheet.

Store Your Dataset

If you are testing an integration, you should store your dataset in the datastore with which the product is integrated.

> **Tip:** Uploading datasets is always available as a means of importing datasets.

- You may need to create a connection between the platform and the datastore.
- Read and write permissions must be enabled for the connecting user to the datastore.

- For more information, see *Connections Page*.

Verification Steps

Steps:

1. Login to the application. See *Login*.
2. In the application menu bar, click **Library**.
3. Click **Import Data**. See *Import Data Page*.
    1. Select the connection where the dataset is stored. For datasets stored on your local desktop, click **Upload**.
    2. Select the dataset.
    3. In the right panel, click the Add Dataset to a Flow checkbox. Enter a name for the new flow.
    4. Click **Import and Add to Flow**.
    5. **Troubleshooting:** At this point, you have read access to your datastore from the platform. If not, please check the logs, permissions, and your Trifacta® configuration.
4. In the left menu bar, click the Flows icon. Flows page, open the flow you just created. See *Flows Page*.
5. In the Flows page, click the dataset you just imported. Click **Add new Recipe**.
6. Select the recipe. Click **Edit Recipe**.
7. The initial sample of the dataset is opened in the Transformer page, where you can edit your recipe to transform the dataset.
    1. In the Transformer page, some steps are automatically added to the recipe for you. So, you can run the job immediately.
    2. You can add additional steps if desired. See *Transformer Page*.
8. Click **Run Job**.
    1.
    2. If options are presented, select the defaults.
    3. To generate results in other formats or output locations, click **Add Publishing Destination.** Configure the output formats and locations.
    4. To test dataset profiling, click the Profile Results checkbox. Note that profiling runs as a separate job and may take considerably longer.
    5. See *Run Job Page*.
    6. **Troubleshooting:** Later, you can re-run this job on a different running environment. Some formats are not available across all running environments.
9. When the job completes, you should see a success message under the Jobs tab in the Flow View page.
    1. **Troubleshooting:** Either the Transform job or the Profiling job may break. To localize the problem, try re-running a job by deselecting the broken job type or running the job on a different running environment (if available). You can also download the log files to try to identify the problem. See *Job Details Page*.
10. Click **View Results** from the context menu for the job listing. In the Job Details page, you can see a visual profile of the generated results. See *Job Details Page*.
11. In the Output Destinations tab, click a link to download the results to your local desktop.
12. Load these results into a local application to verify that the content looks ok.

Checkpoint: You have verified importing from the selected datastore and transforming a dataset. If your job was successfully executed, you have verified that the product is connected to the job running environment and can write results to the defined output location. Optionally, you may have tested profiling of job results. If all of the above tasks completed, the product is operational end-to-end.

Documentation

> **Tip:** You should access online documentation through the product. Online content may receive updates that are not present in PDF content.

You can access complete product documentation online and in PDF format. From within the Trifacta application, select **Help menu > Product Docs**.

Next Steps

After you have accessed the documentation, the following topics are relevant to on-premises deployments. Please review them in order.

> **NOTE:** These materials are located in the *Configuration Guide*.

Topic	Description
Required Platform Configuration	This section covers the following topics, some of which should already be completed: • *Set Base Storage Layer* - The base storage layer must be set once and never changed. • *Create Encryption Key File* - If you plan to integrate the platform with any relational sources, you must create an encryption key file and store it on the Trifacta node • *Running Environment Options* - Depending on your scenario, you may need to perform additional configuration for your available running environment(s) for executing jobs. • *Profiling Options* - In some environments, tweaks to the settings for visual profiling may be required. You can disable visual profiling if needed. • *Configure for Spark* - If you are enabling the Spark running environment, please review and verify the configuration for integrating the platform with the Hadoop cluster instance of Spark.

Configure for Hadoop	• *Configuration by Hadoop Distribution*:
	  • *Configure for Cloudera*
	  • *Configure for Hortonworks*
	• *Configure Hadoop Authentication*

Enable Integration with Compressed Clusters	If the Hadoop cluster uses compression, additional configuration is required.
Enable Integration with Cluster High Availability	If you are integrating with high availability on the Hadoop cluster, please complete these steps. • If you are integrating with high availability on the Hadoop cluster, HttpFS must be enabled in the platform. HttpFS is required in other, less-common cases. See *Enable HttpFS*.
Configure for Hive	Integration with the Hadoop cluster's instance of Hive.
Configure for KMS	Integration with the Hadoop cluster's key management system (KMS) for encrypted transport. Instructions are provided for distribution-specific versions of Hadoop.

Configure Security	A list of topics on applying additional security measures to the Trifacta platform and how integrates with Hadoop.
Configure SSO for AD-LDAP	Please complete these steps if you are integrating with your enterprise's AD/LDAP Single Sign-On (SSO) system.

Configure Server Access through Proxy

When you attempt to launch the application, you may receive an error message similar to the following:

```
No internet connection
Remote server timed out.
```

In some environments, your desktop machine may need to connect to the Internet through a proxy server. If you are using Wrangler Enterprise desktop application, it needs to know the proxy server to which to connect in order to access the Trifacta® node.

Please complete the following configuration steps to access the Trifacta servers.

Steps:

1. In the No internet connection dialog, click **Configure Proxy Settings**.
2. Please provide the following configuration information for your proxy server:
    1. **Proxy Host:** The URL of the proxy server. Please include the protocol identifier (e.g. `http://` or `https://`).
    2. **Proxy Port:** The port number to use to connect to the proxy server. In a URL, this value appears after a colon (e.g. `http://myproxy.example.com:8080`).
    3. **Username:** (optional) If your proxy requires a username to access, please enter it here.
    4. **Password:** (optional) Password associated with the user name.
3. Click **Save Proxy Settings and Restart**.

When the application restarts, you should be able to connect to the login screen.

> **NOTE:** If you continue to have difficulties connecting to the Internet, please contact your network administrator or Internet provider.

Install Software

To install Trifacta® Wrangler Enterprise, please review and complete the following sections in the order listed below.

Topics:

- *Install Dependencies without Internet Access*
- *Install on CentOS and RHEL*
- *Install on Ubuntu*
- *Install for Docker*
- *License Key*
- *Install Desktop Application*
- *Start and Stop the Platform*
- *Login*

Install Dependencies without Internet Access

Offline dependencies should be included in the URL location that Trifacta® provided to you. Please use the `\*deps\*` file.

> **NOTE:** If your installation server is connected to the Internet, the required dependencies are automatically downloaded and installed for you. You may skip this section.

Use the steps below to acquire and install dependencies required by the Trifacta platform. If you need further assistance, please contact *Trifacta Support*.

Install software dependencies without Internet access for CentOS or RHEL:

1. In a CentOS or RHEL environment, the dependencies repository must be installed into the following directory:

   ```
   /var/local/trifacta
   ```

2. The following commands configure Yum to point to the repository in `/var/local/trifacta`, which yum knows as `local`. Repo permissions are set appropriately. Commands:

   ```
   tar xvzf <DEPENDENCIES_ARCHIVE>.tar.gz
   mv local.repo /etc/yum.repos.d
   mv trifacta /var/local
   chown -R root:root /var/local/trifacta
   chmod -R o-w+r /var/local/trifacta
   ```

3. The following command installs the RPM while disable all repos other than local, which prevents the installer from reaching out to the Internet for package updates:

   > **NOTE:** The disabling of repositories only applies to this command.

   ```
   sudo yum --disablerepo=* --enablerepo=local install <INSTALLER>.rpm
   ```

4. If the above command fails and complains about a missing repo, you can add the missing repo to the `enablerepo` list. For example, if the `centos-base` repo is reported as missing, then the command would be the following:

   ```
   sudo yum --disablerepo=* --enablerepo=local,centos-base install
   <INSTALLER>.rpm
   ```

5. If you do not have a supported version of a Java Developer Kit installed on the Trifacta node, you can use the following command to install OpenJDK, which is included in the offline dependencies:

```
sudo yum --disablerepo=* --enablerepo=local,centos-base install
java-1.8.0-openjdk-1.8.0 java-1.8.0-openjdk-devel
```

Install database dependencies without Internet access:

If you are installing the databases on a node without Internet access, you can install the dependencies using either of the following commands:

> **NOTE:** This step is only required if you are installing the databases on the same node where the software is installed.

For PostgreSQL:

```
sudo yum --disablerepo=* --enablerepo=local install postgresql96-server
```

For MySQL:

```
sudo yum --disablerepo=* --enablerepo=local install mysql-community-
server
```

> **NOTE:** You must also install the MySQL JARs on the Trifacta node. These instructions are provided later.

Database are installed after the software is installed. For more information, see *Install Databases*.

Install dependencies without Internet access in Ubuntu:

If you are trying to perform a manual installation of dependencies in Ubuntu, please contact *Trifacta Support*.

Install on CentOS and RHEL

Contents:

- *Preparation*
- *Installation*
  - *1. Install Dependencies*
  - *2. Install JDK*
  - *3. Install Trifacta package*
  - *4. Verify Install*
  - *5. Install License Key*
  - *6. Store install packages*
  - *7. Install and configure Trifacta databases*
- *Configuration*

This guide takes you through the steps for installing Trifacta® Wrangler Enterprise software on CentOS or Red Hat.

For more information on supported operating system versions, see *System Requirements*.

Preparation

Before you begin, please complete the following.

> **NOTE:** Except for database installation and configuration, all install commands should be run as the root user or a user with similar privileges. For database installation, you will be asked to switch the database user account.

Steps:

1. Set the node where Trifacta Wrangler Enterprise is to be installed.
    1. Review the *System Requirements* and verify that all required components have been installed.
    2. Verify that all required system ports are opened on the node. See *System Ports*.
2. Review the *Desktop Requirements*.

    > **NOTE:** Trifacta Wrangler Enterprise requires the installation of Google Chrome on each desktop. For more information, see *Desktop Requirements*.

3. Review the *System Dependencies*.

    > **NOTE:** If you are installing on node without access to the Internet, you must download the offline dependencies before you begin. See *Install Dependencies without Internet Access*.

4. Acquire your *License Key*.
5. Install and verify operations of the datastore, if used.

    > **NOTE:** Access to the Spark cluster is required.

6. Verify access to the server where the Trifacta platform is to be installed.
7. **Cluster Configuration:** Additional steps are required to integrate the Trifacta platform with the cluster. See *Prepare Hadoop for Integration with the Platform*.

Installation

1. Install Dependencies

Without Internet access

If you have not done so already, you may download the dependency bundle with your release directly from Trifacta . For more information, see *Install Dependencies without Internet Access*.

With Internet access

Use the following to add the hosted package repository for CentOS/RHEL, which will automatically install the proper packages for your environment.

```
# If the client has curl installed ...
curl https://packagecloud.io/install/repositories/trifacta/dependencies
/script.rpm.sh | sudo bash

# Otherwise, you can also use wget ...
wget -qO- https://packagecloud.io/install/repositories/trifacta
/dependencies/script.rpm.sh | sudo bash
```

2. Install JDK

By default, the Trifacta node uses OpenJDK for accessing Java libraries and components. In some environments, basic setup of the node may include installation of a JDK. Please review your environment to verify that an appropriate JDK version has been installed on the node.

> **NOTE:** Use of Java Development Kits other than OpenJDK is not currently supported. However, the platform may work with the Java Development Kit of your choice, as long as it is compatible with the supported version(s) of Java. See *System Requirements*.

> **NOTE:** OpenJDK is included in the offline dependencies, which can be used to install the platform without Internet access. For more information, see *Install Dependencies without Internet Access*.

The following commands can be used to install OpenJDK. These commands can be modified to install a separate compatible version of the JDK.

```
sudo yum install java-1.8.0-openjdk-1.8.0 java-1.8.0-openjdk-devel
```

> **NOTE:** If `java-1.8.0-openjdk-devel` is not included, the batch job runner service, which is required, fails to start.

JAVA_HOME:

By default, the `JAVA_HOME` environment variable is configured to point to a default install location for the OpenJDK package.

> **NOTE:** If you have installed a JDK other than the OpenJDK version provided with the software, you must set the `JAVA_HOME` environment variable on the Trifacta node to point to the correct install location.

The property value must be updated in the following locations:

1. Edit the following file: `/opt/trifacta/conf/env.sh`

2. Save changes.

3. Install Trifacta package

> **NOTE:** If you are installing without Internet access, you must reference the local repository. The command to execute the installer is slightly different. See *Install Dependencies without Internet Access*.

> **NOTE:** Installing the Trifacta platform in a directory other than the default one is not supported or recommended.

Install the package with yum, using root:

```
sudo yum install <rpm file>
```

4. Verify Install

The product is installed in the following directory:

```
/opt/trifacta
```

JAVA_HOME:

The platform must be made aware of the location of Java.

Steps:

1. Edit the following file: `/opt/trifacta/conf/trifacta-conf.json`
2. Update the following parameter value:

```
"env": {
  "JAVA_HOME": "/usr/lib/jvm/java-1.8.0-openjdk.x86_64"
},
```

3. Save changes.

5. Install License Key

Please install the license key provided to you by Trifacta. See *License Key*.

6. Store install packages

For safekeeping, you should retain all install packages that have been installed with this Trifacta deployment.

7. Install and configure Trifacta databases

The Trifacta platform requires installation of several databases. If you have not done so already, you must install and configure the databases used to store Trifacta metadata. See *Install Databases*.

Configuration

After installation is complete, additional configuration is required.

> **The Trifacta platform requires additional configuration for a successful integration with the datastore. Please review and complete the necessary configuration steps. For more information, see *Configure*.**

Install on Ubuntu

Contents:

- *Preparation*
- *Installation*
  - *1. Install Dependencies*
  - *2. Install JDK*
  - *3. Install Trifacta package*
  - *4. Verify Install*
  - *5. Install License Key*
  - *6. Store install packages*
  - *7. Install and configure Trifacta databases*
- *Configuration*

This guide takes you through the steps for installing Trifacta® Wrangler Enterprise software on Ubuntu.

For more information on supported operating system versions, see *System Requirements*.

Preparation

Before you begin, please complete the following.

> **NOTE:** Except for database installation and configuration, all install commands should be run as the root user or a user with similar privileges. For database installation, you will be asked to switch the database user account.

Steps:

1. Set the node where Trifacta Wrangler Enterprise is to be installed.
   1. Review the *System Requirements* and verify that all required components have been installed.
   2. Verify that all required system ports are opened on the node. See *System Ports*.
2. Review the *Desktop Requirements*.

   > **NOTE:** Trifacta Wrangler Enterprise requires the installation of Google Chrome on each desktop.

3. Review the *System Dependencies*.

> **NOTE:** If you are installing on node without access to the Internet, you must download the offline dependencies before you begin. See *Install Dependencies without Internet Access*.

4. Acquire your *License Key*.
5. Install and verify operations of the datastore, if used.

> **NOTE:** Access to the cluster may be required.

6. Verify access to the server where the Trifacta platform is to be installed.
7. **Cluster configuration:** Additional steps are required to integrate the Trifacta platform with the cluster. See *Prepare Hadoop for Integration with the Platform*.

Installation

1. Install Dependencies

Without Internet access

If you have not done so already, you may download the dependency bundle with your release directly from Trifacta . For more information, see *Install Dependencies without Internet Access*.

With Internet access

Use the following to add the hosted package repository for Ubuntu, which will automatically install the proper packages for your environment.

> **NOTE:** Install curl if not present on your system.

Then, execute the following command:

> **NOTE:** Run the following command as the root user. In proxied environments, the script may encounter issues with detecting proxy settings.

```
curl https://packagecloud.io/install/repositories/trifacta/dependencies
/script.deb.sh | sudo bash
```

Special instructions for Ubuntu installs

These steps manually install the correct and supported version of the following:

- nodeJS
- nginX

Due to a known issue resolving package dependencies on Ubuntu, please complete the following steps prior to installation of other dependencies or software.

1. Login to the Trifacta node as an administrator.
2. Execute the following command to install the appropriate versions of nodeJS and nginX.

    1. Ubuntu 14.04:

```
sudo apt-get install nginx=1.12.2-1~trusty nodejs=10.13.0-
1nodesource1
```

2. Ubuntu 16.04

```
sudo apt-get install nginx=1.12.2-1~xenial nodejs=10.13.0-
1nodesource1
```

3. Continue with the installation process.

2. Install JDK

By default, the Trifacta node uses OpenJDK for accessing Java libraries and components. In some environments, basic setup of the node may include installation of a JDK. Please review your environment to verify that an appropriate JDK version has been installed on the node.

> **NOTE:** Use of Java Development Kits other than OpenJDK is not currently supported. However, the platform may work with the Java Development Kit of your choice, as long as it is compatible with the supported version(s) of Java. See *System Requirements*.

> **NOTE:** OpenJDK is included in the offline dependencies, which can be used to install the platform without Internet access. For more information, see *Install Dependencies without Internet Access*.

The following commands can be used to install OpenJDK. These commands can be modified to install a separate compatible version of the JDK.

```
sudo apt-get install openjdk-8-jre-headless
```

JAVA_HOME:

By default, the `JAVA_HOME` environment variable is configured to point to a default install location for the OpenJDK package.

> **NOTE:** If you have installed a JDK other than the OpenJDK version provided with the software, you must set the `JAVA_HOME` environment variable on the Trifacta node to point to the correct install location.

The property value must be updated in the following locations:

1. Edit the following file: `/opt/trifacta/conf/env.sh`
2. Save changes.

3. Install Trifacta package

> **NOTE:** If you are installing without Internet access, you must reference the local repository. The command to execute the installer is slightly different. See *Install Dependencies without Internet Access*.

NOTE: Installing the Trifacta platform in a directory other than the default one is not supported or recommended.

Install the package with apt, using root:

```
sudo dpkg -i <deb file>
```

The previous line may return an error message, which you may ignore. Continue with the following command:

```
sudo apt-get -f -y install
```

4. Verify Install

The product is installed in the following directory:

```
/opt/trifacta
```

JAVA_HOME:

The platform must be made aware of the location of Java.

Steps:

1. Edit the following file: `/opt/trifacta/conf/trifacta-conf.json`
2. Update the following parameter value:

```
"env": {
  "JAVA_HOME": "/usr/lib/jvm/java-1.8.0-openjdk.x86_64"
},
```

3. Save changes.

5. Install License Key

Please install the license key provided to you by Trifacta. See *License Key*.

6. Store install packages

For safekeeping, you should retain all install packages that have been installed with this Trifacta deployment.

7. Install and configure Trifacta databases

The Trifacta platform requires installation of several databases. If you have not done so already, you must install and configure the databases used to store Trifacta metadata. See *Install Databases*.

Configuration

After installation is complete, additional configuration is required.

> **The Trifacta platform requires additional configuration for a successful integration with the datastore. Please review and complete the necessary configuration steps. For more information, see *Configure*.**

Install for Docker

Contents:

- *Deployment Scenario*
- *Limitations*
- *Requirements*
  - *Docker Daemon*
- *Preparation*
- *Acquire Image*
  - *Acquire from FTP site*
  - *Build your own Docker image*
- *Configure Docker Image*
- *Start Server Container*
  - *Import Additional Configuration Files*
  - *Import license key file*
  - *Import Hadoop distribution libraries*
  - *Import Hadoop cluster configuration files*
  - *Install Kerberos client*
  - *Perform configuration changes as necessary*
- *Start and Stop the Container*
  - *Stop container*
  - *Restart container*
  - *Recreate container*
  - *Stop and destroy the container*
- *Verify Deployment*
- *Configuration*

This guide steps through the process of acquiring and deploying a Docker image of the Trifacta® platform in your Docker environment. Optionally, you can build the Docker image locally, which enables further configuration options.

Deployment Scenario

- Trifacta Wrangler Enterprise deployed into a customer-managed environment: On-premises, AWS, or Azure.
- PostgreSQL 9.6 installed either:
  - Locally
  - Remote server

Limitations

- You cannot upgrade to a Docker image from a non-Docker deployment.
- You cannot switch an existing installation to a Docker image.
- Supported distributions of Cloudera or Hortonworks:

    - *Supported Deployment Scenarios for Cloudera*
    - *Supported Deployment Scenarios for Hortonworks*
- The base storage layer of the platform must be HDFS. Base storage of S3 is not supported.
- High availability for the Trifacta platform in Docker is not supported.
- SSO integration is not supported.

Requirements

Support for orchestration through Docker Compose only

- Docker version 17.12 or later
- Docker-Compose 1.11.2 or later. Version must be compatible with your version of Docker.

Docker Daemon

	Minimum	Recommended
CPU Cores	2 CPU	4 CPU
Available RAM	8 GB RAM	10+ GB RAM

Preparation

1. Review the *Desktop Requirements*.

   > **NOTE:** Trifacta Wrangler Enterprise requires the installation of Google Chrome on each desktop. For more information, see *Desktop Requirements*.

2. Acquire your *License Key*.

Acquire Image

You can acquire the latest Docker image using one of the following methods:

1. Acquire from FTP site.
2. Build your own Docker image.

Acquire from FTP site

Steps:

1. Download the following files from the FTP site:
    1. `trifacta-docker-setup-bundle-x.y.z.tar`
    2. `trifacta-docker-image-x.y.z.tar`

       > **NOTE:** `x.y.z` refers to the version number (e.g. `6.4.0`).

2. Untar the `setup-bundle` file:

```
tar xvf trifacta-docker-setup-bundle-x.y.z.tar
```

3. Files are extracted into a `docker` folder. Key files:

File	Description
docker-compose-local-postgres.yaml	Runtime configuration file for the Docker image when PostgreSQL is to be running on the same machine. More information is provided below.
docker-compose-local-mysql.yaml	Runtime configuration file for the Docker image when MySQL is to be running on the same machine. More information is provided below.
docker-compose-remote-db.yaml	Runtime configuration file for the Docker image when the database is to be accessed from a remote server. **NOTE:** You must manage this instance of the database. More information is provided below.
README-running-trifacta-container.md	Instructions for running the Trifacta container **NOTE:** These instructions are referenced later in this workflow.
README-building-trifacta-container.md	Instructions for building the Trifacta container **NOTE:** This file does not apply if you are using the provided Docker image.

4. Load the Docker image into your local Docker environment:

```
docker load < trifacta-docker-image-x.y.z.tar
```

5. Confirm that the image has been loaded. Execute the following command, which should list the Docker image:

```
docker images
```

6. You can now configure the Docker image. Please skip that section.

Build your own Docker image

As needed, you can build your own Docker image.

Requirements

- Docker version 17.12 or later
- Docker Compose 1.11.2 or newer. It should be compatible with above version of Docker.

Build steps

1. Acquire the RPM file from the FTP site:

   > **NOTE:** You must acquire the el7 RPM file for this release.

2. In your Docker environment, copy the `trifacta-server\*.rpm` file to the same level as the `Dockerfile`.
3. Verify that the `docker-files` folder and its contents are present.
4. Use the following command to build the image:

   ```
   docker build -t trifacta/server-enterprise:latest .
   ```

5. This process could take about 10 minutes. When it is completed, you should see the build image in the Docker list of local images.

   > **NOTE:** To reduce the size of the Docker image, the Dockerfile installs the trifacta-server RPM file in one stage and then copies over the results to the final stage. The RPM is not actually installed in the final stage. All of the files are properly located.

6. You can now configure the Docker image.

Configure Docker Image

Before you start the Docker container, you should review the properties for the Docker image. In the provided image, please open the appropriate `docker-compose` file:

File	Description
docker-compose-local-postgres.yaml	Database properties in this file are pre-configured to work with the installed instance of PostgreSQL, although you may wish to change some of the properties for security reasons.
docker-compose-local-mysql.yaml	Database properties in this file are pre-configured to work with the installed instance of MySQL, although you may wish to change some of the properties for security reasons.
docker-compose-remote-db.yaml	The Trifacta databases are to be installed on a remote server that you manage. **NOTE:** Additional configuration is required.

> **NOTE:** You may want to create a backup of this file first.

Key general properties:

NOTE: Avoid modifying properties that are not listed below.

Property	Description
image	This reference must match the name of the image that you have acquired.
container_name	Name of container in your Docker environment.
ports	Defines the listening port for the Trifacta application. Default is `3005`. **NOTE:** If you must change the listening port, additional configuration is required after the image is deployed. See *Change Listening Port* For more information, see *System Ports*.

Database properties:

These properties pertain to the installation of the database to which the Trifacta application connects.

Property	Description
DB_INIT	If set to `true`, database initialization steps are performed at startup. **NOTE:** This step applies only if you are starting the container for the first time, and PostgreSQL databases will be installed locally.
DB_TYPE	Set this value to `postgresql` or `mysql`.
DB_HOST_NAME	Hostname of the machine hosting the databases. Leave value as `localhost` for local installation.
DB_HOST_PORT	(Remote only) Port number to use to connect to the databases. Default is `5432`. **NOTE:** If you are modifying, additional configuration is required after installation is complete. See *Change Database Port*.
DB_ADMIN_USERNAME	Admin username to be used to create DB roles/databases. Modify this value for remote installation. **NOTE:** If you are modifying this value, additional configuration is required. Please see the documentation for your database version.
DB_ADMIN_PASSWORD	Admin password to be used to create DB roles/databases. Modify this value for remote installation.

Kerberos properties:

If your Hadoop cluster is protected by Kerberos, please review the following properties.

Property	Description
KERBEROS_KEYTAB_FILE	Full path inside of the container where the Kerberos keytab file is located. Default value: `/opt/trifacta/conf/trifacta.keytab` **NOTE:** The keytab file must be imported and mounted to this location. Configuration details are provided later.
KERBEROS_KRB5_CONF	Full path inside of the container where the Kerberos `krb5.conf file` is located. Default: `/opt/krb-config/krb5.conf`

Hadoop distribution client JARs:

Please enable the appropriate path to the client JAR files for your Hadoop distribution. In the following example, the Cloudera path has been enabled, and the Hortonworks path has been disabled:

```
# Mount folder from outside for necessary hadoop client jars
# For CDH
- /opt/cloudera:/opt/cloudera
# For HDP
#- /usr/hdp:/usr/hdp
```

Please modify these lines if you are using Hortonworks.

Volume properties:

These properties govern where volumes are mounted in the container.

NOTE: These values should not be modified unless necessary.

Property	Description

volumes.conf	Full path in container to the Trifacta configuration directory. Default: `/opt/trifacta/conf`
volumes.logs	Full path in container to the Trifacta logs directory. Default: `/opt/trifacta/logs`
volumes.license	Full path in container to the Trifacta license directory. Default: `/trifacta-license`

Start Server Container

After you have performed the above configuration, execute the following to initialize the Docker container:

```
docker-compose -f <docker-compose-filename>.yaml run trifacta initfiles
```

When the above is started for the first time, the following directories are created on the localhost:

Directory	Description
./trifacta-data	Used by the Trifacta container to expose the `conf` and `logs` directories.

Import Additional Configuration Files

After you have started the new container, additional configuration files must be imported.

Import license key file

The Trifacta license file must be staged for use by the platform. Stage the file in the following location in the container:

NOTE: If you are using a non-default path or filename, you must update the `<docker-compose-filename> .yaml` file.

```
trifacta-license/license.json
```

Import Hadoop distribution libraries

If the container you are creating is on the edge node of your Hadoop cluster, you must provide the Hadoop libraries.

1. You must mount the Hadoop distribution libraries into the container. For more information on the libraries, see the documentation for your Hadoop distribution.
2. The Docker Compose file must be made aware of these libraries. Details are below.

Import Hadoop cluster configuration files

Some core cluster configuration files from your Hadoop distribution must be provided to the container. These files must be copied into the following directory within the container:

```
./trifacta-data/conf/hadoop-site
```

For more information, see *Configure for Hadoop* in the Configuration Guide.

Install Kerberos client

If Kerberos is enabled, you must install the Kerberos client and keytab on the node container. Copy the keytab file to the following stage location:

```
/trifacta-data/conf/trifacta.keytab
```

See *Configure for Kerberos Integration* in the Configuration Guide.

Perform configuration changes as necessary

The primary configuration file for the platform is in the following location in the launched container:

```
/opt/trifacta/conf/trifacta-conf.json
```

NOTE: Unless you are comfortable working with this file, you should avoid direct edits to it. All subsequent configuration can be applied from within the application, which supports some forms of data validation. It is possible to corrupt the file using direct edits.

Configuration topics are covered later.

Start and Stop the Container

Stop container

Stops the container but does not destroy it.

NOTE: Application and local database data is not destroyed. As long as the `<docker-compose-filename> .yaml` properties point to the correct location of the `*-data` files, data should be preserved. You can start new containers to use this data, too. Do not change ownership on these directories.

```
docker-compose -f <docker-compose-filename>.yaml stop
```

Restart container

Restarts an existing container.

```
docker-compose -f <docker-compose-filename>.yaml start
```

Recreate container

Recreates a container using existing local data.

```
docker-compose -f <docker-compose-filename>.yaml up --force-recreate -d
```

Stop and destroy the container

Stops the container and destroys it.

The following also destroys all application configuration, logs, and database data. You may want to back up these directories first.

```
docker-compose -f <docker-compose-filename>.yaml down
```

Local PostgreSQL:

```
sudo rm -rf trifacta-data/ postgres-data/
```

Local MySQL or remote database:

```
sudo rm -rf trifacta-data/
```

Verify Deployment

1. Verify access to the server where the Trifacta platform is to be installed.
2. **Cluster Configuration:** Additional steps are required to integrate the Trifacta platform with the cluster. See *Prepare Hadoop for Integration with the Platform*.
3. Start the platform within the container. See *Start and Stop the Platform*.

Configuration

After installation is complete, additional configuration is required. You can complete this configuration from within the application.

Steps:

1. Login to the application. See *Login*.
2. The primary configuration interface is the Admin Settings page. From the left menu, select **Settings menu > Settings > Admin Settings**. For more information, see *Admin Settings Page* in the Admin Guide.
3. Workspace-level configuration can also be applied. From the left menu, select **Settings menu > Settings > Workspace Admin**. For more information, see *Workspace Admin Page* in the Admin Guide.

The Trifacta platform requires additional configuration for a successful integration with the datastore. Please review and complete the necessary configuration steps. For more information, see *Configure* in the Configuration Guide.

License Key

Contents:

- *Download license key file*
- *Acquire license key*
- *Install your license key*
- *Update your license key*
- *Changing the license key location*
- *Expired license*
- *Invalid license key file*

Download license key file

If you have not done so already, the license key file is available where you have acquired the installation package. Please download `license.json`.

Acquire license key

A valid license key (`license.json`) is provided to each customer prior to installation. Your license key file is a JSON file that contains important information on your license.

NOTE: If your license key has expired, please contact *Trifacta Support*.

Install your license key

If you are updating your license, you may want to save your previous license key to a new location before overwriting.

NOTE: Do not maintain multiple license key files in this directory.

To apply your license key, copy the key file to the following location in the Trifacta® deployment:

```
/opt/trifacta/license
```

Update your license key

After you have installed your license key, you can update your license with a new one through the Admin Settings page. See *Admin Settings Page*.

Changing the license key location

By default, the license key file in use must be named: `license.json`.

If needed, you can change the path and filename of the license key. The property is the following:

```
"license.location"
```

See *Admin Settings Page*.

Expired license

NOTE: If your license expires, you cannot use the product until a new and valid license key file has been applied. When administrators attempt to login to the application, they are automatically redirected to a location from which they can upload a new license key file.

Invalid license key file

When you start the Trifacta platform, you may see the following:

Your license key is missing or has expired. Please contact *Trifacta Support*.

Install Desktop Application

Contents:

- *Install Process*
  - *Download*
  - *Setup*
  - *Install for Windows*
  - *Windows Command Line Installation and Configuration*
  - *Launch the Application*
  - *Documentation Note*
- *Troubleshooting*
  - *Cannot connect to server*
  - *"Does Not Support Your Browser" error*

If your environment does not support the use of Chrome, you can install the Wrangler Enterprise desktop application to provide the same access and functionality as the Trifacta® application. This desktop application connects to the enterprise Trifacta instance and provides the same capabilities without requiring a locally installed version of Chrome browser.

Wrangler Enterprise desktop application is a hybrid desktop application. Your local application instance accesses registered data files located in the datastore to which the Trifacta node is connected.

Install Process

> **NOTE:** The Wrangler Enterprise desktop application is a 64-bit Microsoft Windows application. It requires a 64-bit version of Windows to execute. The application also supports Single Sign On (SSO), if it is enabled.

Download

To begin, you must download the following Windows MSI file (`TrifactaEnterpriseSetup.msi`) from the location where your software was provided.

If you are planning to automate installation to desktops in your environment, please also download `setTrifactaServer.ps1`.

Setup

Before you begin, you should perform any necessary configuration of the Trifacta node before deploying the instances of the application. See *Configure for Desktop Application*.

Install for Windows

Steps:

1. On your Windows desktop, double-click the MSI file.
2. Follow the on-screen instructions to install the software.

Windows Command Line Installation and Configuration

As an alternative, you can perform installation and initial configuration from the command line. Download the MSI and the PS1 files to a local directory that is accessible.

> **NOTE:** For command line install, you must download from the `setTrifactaServer.ps1` from the download location.

Install software:

```
msiexec /i <path_to_TrifactaEnterpriseSetup.msi> /passive
```

Configure URL of Trifacta node:

```
setTrifactaServer.ps1 -trifactaServer <server_url> -installDir
<local_dir>
```

Parameter	Description
`trifactaServer`	(Required) URL of the server hosting the Trifacta platform. Format: `<http|https>://<host>:<port>`
`installDir`	(Optional) Specifies the installation directory in the local environment. If not specified, installation directory defaults to use the same path as the installer.
common installer parameters	This command supports the following Windows installer parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see *about_CommonParameters* here: *http://go.microsoft.com/fwlink/?LinkID=113216*.

After this install is completed, desktop users should be able to use the application normally.

Launch the Application

Steps:

1. When installation is complete, double-click the application icon.
2. For the server, please enter the full URL including port number of the Trifacta instance to which you are connecting.
    1. By default, the server is available over port `3005`. For more information, please contact your IT administrator.
    2. If you connect to the Internet through a proxy, additional configuration is required. See *Configure Server Access through Proxy*.

NOTE: If you make a mistake in specifying the URL to the server, please uninstall and reinstall the MSI. This step clears the local application cache, and you can enter the appropriate path through the application. See *Uninstall* below.

3. When the proper URL and port number are provided, you may launch the application.
4. If your environment contains multiple server deployments, you can select the one to which to connect:

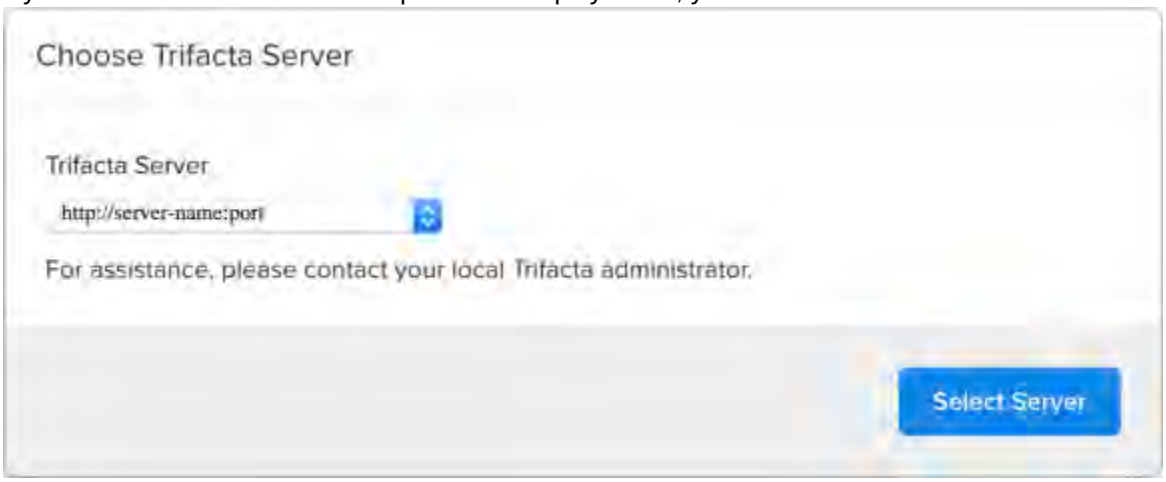


Figure: Choose Server

5. Login with your Trifacta account. See *Login*.

Documentation Note

Unless specifically noted, all features described for Trifacta Wrangler Enterprise or the Trifacta application apply to the Wrangler Enterprise desktop application.

Uninstall

To uninstall from your Windows machine, use the Add or Remove Programs control panel.

Troubleshooting

Cannot connect to server

If you are unable to connect to the server, please do the following:

1. Verify that you are connecting to the appropriate URL.
    1. If you are connecting to the incorrect URL, please uninstall the application and re-install using the MSI file. See *Uninstall* above.
2. Verify if you need to connect to the server through a proxy server. If so, additional configuration is required. See *Configure Server Access through Proxy*.
3. Check your firewall settings.

"Does Not Support Your Browser" error

This error message indicates that you are trying to connect to an instance of the server that does not support the Wrangler Enterprise desktop application. Please verify that your connection URL is pointed to a supported instance of the server.

Start and Stop the Platform

Contents:

- *Start*
  - *Verify operations*
- *Restart*
- *Stop*
- *Debugging*
- *Troubleshooting*
  - *Error - SequelizeConnectionRefusedError: connect ECONNREFUSED*

Tip: The Restart Trifacta button in the Admin Settings page is the preferred method for restarting the platform.

NOTE: The restart button is not available when high availability is enabled for the Trifacta® node.

See *Admin Settings Page*.

Start

NOTE: These operations must be executed under the root user.

Command:

```
service trifacta start
```

Verify operations

Steps:

1. Check logs for errors:

   ```
   /opt/trifacta/logs/*.log
   ```

   1. You can also access logs through the Trifacta® application for each service. See *System Services and Logs*.
2. Login to the Trifacta application. If available, perform a simple transformation operation. See *Login*.
3. Run a simple job. See *Verify Operations*.

Restart

Command:

```
service trifacta restart
```

When the login page is available, the system has been restarted. See *Login*.

Tip: If you have made any configuration changes, you should verify operations. See *Verify Operations*.

Stop

Command:

```
service trifacta stop
```

Debugging

You can verify operations of WebHDFS. Command:

```
curl -i  "http://<hadoop_node>:<port_number>/webhdfs/v1/?
op=LISTSTATUS&user.name=trifacta"
```

Troubleshooting

Error - SequelizeConnectionRefusedError: connect ECONNREFUSED

If you have attempted to start the platform after an operating system reboot, you may receive the following error message, and the platform start fails to complete:

```
2016-10-04T14:03:17.883Z - error: [ENVIRONMENT] Environment Sanity Test
Failed
2016-10-04T14:03:17.883Z - error: [ENVIRONMENT] Exception Type: Error
2016-10-04T14:03:17.883Z - error: [ENVIRONMENT] Exception Message:
SequelizeConnectionRefusedError: connect ECONNREFUSED
```

Solution:

NOTE: This solution applies to PostgreSQL 9.6 only. Please modify for your installed database version.

This error can occur when the operating system is restarted. Please execute the following commands to check the PostgreSQL configuration and restart the databases.

```
chkconfig postgresql-9.6 on
```

Then, restart the platform as normal.

```
service trifacta restart
```

Login

> **NOTE:** Administrators of the platform should change the default password for the admin account. See *Change Admin Password*.

To login to the Trifacta® application, navigate to the following in your browser:

`http://<host_name>:<port_number>`

where:

- `<host_name>` is the host of the Trifacta application.
- `<port_number>` is the port number to use. Default is `3005`.

If you do not have an account, click **Register**.

- If self-registration is enabled, you you may be able to immediately login after registering.
- If Kerberos or secure impersonation is enabled, an administrator must apply a Hadoop principal value to the account before you can login. Please contact your Trifacta administrator.
- System administrators can enable self-registration. See *Configure User Self-Registration*.

After you login, you are placed in the Flows page, where you can create and manage your datasets and flows. See *Flows Page*.

- If you are using S3 as your base storage layer and per-user authentication has been enabled, you must provide the AWS credentials to connect to your storage. From the left navigation bar, select **Settings > Storage** and then select the AWS option. See *Configure Your Access to S3*.

- For a basic walkthrough of the Trifacta application, see *Workflow Basics*.

To logout:

From the Settings menu, select **Logout**.

Install Reference

These appendices provide additional information during installation of Trifacta® Wrangler Enterprise.

Topics:

- *Install SSL Certificate*
- *Change Listening Port*

- *Supported Deployment Scenarios for Cloudera*
- *Supported Deployment Scenarios for Hortonworks*
- *Supported Deployment Scenarios for AWS*
- *Supported Deployment Scenarios for Azure*
- *Uninstall*

Install SSL Certificate

Contents:

- *Pre-requisites*
- *Configure nginx*
- *Modify listening port for Trifacta platform*
- *Add secure HTTP headers*
- *Enable secure cookies*
- *Troubleshooting*

You may optionally configure an SSL certificate to secure connections to the web application of the Trifacta® platform.

Pre-requisites

1. A valid SSL certificate for the FQDN where the Trifacta application is hosted
2. Root access to the Trifacta server
3. Trifacta platform is up and running

Configure nginx

There are two separate Nginx services on the server: one service for internal application use, and one service that functions as a proxy between users and the Trifacta application. To install the SSL certificate, all configuration are applied to the proxy process only.

Steps:

1. Log into the Trifacta server as the **centos** user. Switch to the **root** user:

   ```
   sudo su
   ```

2. Enable the proxy nginx service so that it starts on boot:

   ```
   systemctl enable nginx
   ```

3. Create a folder for the private key and limit access to it:

   ```
   sudo mkdir /etc/ssl/private/ && sudo chmod 700 /etc/ssl/private
   ```

4. Copy the following files to the server. If you copy and paste the content, please ensure that you do not miss characters or insert unwanted characters.

    1. The `.key` file should go into the `/etc/ssl/private/` directory.
    2. The .crt file and the CA bundle/intermediate certificate bundle should go into the `/etc/ssl/certs/` directory.

    > **NOTE:** The delivery name and format of these files varies by provider. Please verify with your provider's documentation if this is unclear.

    3. Your certificate and the intermediate/authority certificate must be combined into one file for nginx. Here is an example of how to combine them together:

    ```
    cat example_com.crt bundle.crt >> ssl-bundle.crt
    ```

5. Update the permissions on these files. Modify the following filenames as necessary:

    ```
    sudo chmod 600 /etc/ssl/certs/ssl-bundle.crt
    sudo chmod 600 /etc/ssl/private/your-private-cert.key
    ```

6. Use the following commands to deploy the example SSL configuration file provided on the server:

    > **NOTE:** Below, some values are too long for a single line. Single lines that overflow to additional lines are marked with a `\`. The backslash should not be included if the line is used as input.

    ```
    cp /opt/trifacta/conf/ssl-nginx.conf.sample /etc/nginx/conf.d
    /trifacta.conf && \
    rm /etc/nginx/conf.d/default.conf
    ```

7. Edit the following file:

    ```
    /etc/nginx/conf.d/trifacta.conf
    ```

8. Please modify the following key directives at least:

Directive	Description
`server_name`	FQDN of the host, which must match the SSL certificate's Common Name
`ssl_certificate`	Path to the file of the certificate bundle that you created on the server. This value may not require modification.
`ssl_certificate_key`	Path to the .key file on the server.

    Example file:

```
server {
  listen          443;
  ssl             on;
  server_name     EXAMPLE.CUSTOMER.COM;
  # Don't limit the size of client uploads.
  client_max_body_size 0;
  access_log      /var/log/nginx/ssl-access.log;
  error_log       /var/log/nginx/ssl-error.log;
  ssl_certificate      /etc/ssl/certs/ssl-bundle.crt;
  ssl_certificate_key  /etc/ssl/certs/EXAMPLE-NAME.key;
  ssl_protocols        SSLv3 TLSv1 TLSv1.1 TLSv1.2;
  ssl_ciphers RC4:HIGH:!aNULL:!MD5;
  ssl_prefer_server_ciphers on;
  keepalive_timeout    60;
  ssl_session_cache    shared:SSL:10m;
  ssl_session_timeout  10m;
  location / {
    proxy_pass  http://localhost:3005;
    proxy_next_upstream error timeout invalid_header http_500
http_502 http_503 http_504;
    proxy_set_header        Accept-Encoding   "";
    proxy_set_header        Host            $host;
    proxy_set_header        X-Real-IP       $remote_addr;
    proxy_set_header        X-Forwarded-For
$proxy_add_x_forwarded_for;
    proxy_set_header        X-Forwarded-Proto $scheme;
    add_header              Front-End-Https   on;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_redirect     off;
  }
  proxy_connect_timeout       6000;
  proxy_send_timeout          6000;
  proxy_read_timeout          6000;
  send_timeout                6000;
}
server {
  listen          80;
  return 301 https://$host$request_uri;
}
```

9. Save the file.
10. To apply the new configuration, start or restart the nginx service:

```
service nginx restart
```

Modify listening port for Trifacta platform

If you have changed the listening port as part of the above configuration change, then the `proxy.port` setting in Trifacta platform configuration must be updated. See *Change Listening Port*.

Add secure HTTP headers

If you have enabled SSL on the platform, you can optionally insert the following additional headers to all requests to the Trifacta node:

Header	Protocol	Required Parameters
X-XSS-Protection	HTTP and HTTPS	`proxy.securityHeaders.enabled=true`
X-Frame-Options	HTTP and HTTPS	`proxy.securityHeaders.enabled=true`
Strict-Transport-Security	HTTPS	`proxy.securityHeaders.enabled=true` and `proxy.securityHeaders.httpsHeaders=true`

NOTE: SSL must be enabled to apply these security headers.

Steps:

To add these headers to all requests, please apply the following change:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the following setting and change its value to `true`:

```
"proxy.securityHeaders.httpsHeaders": false,
```

3. Save your changes and restart the platform.

Enable secure cookies

If you have enabled SSL on the platform, you can optionally enable the use of secure cookies.

NOTE: SSL must be enabled.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the following setting and change its value to `true`:

```
"webapp.session.cookieSecureFlag": false,
```

3. Save your changes and restart the platform.

Troubleshooting

Problem - SELinux blocks proxy service from communicating with internal app service

If the Trifacta platform is installed on SELinux, the operating system blocks communications between the service that manages the proxy between users and the application and the service that manages internal application communications.

To determine if this problem is present, execute the following command:

```
sudo cat /var/log/audit/audit.log | grep nginx | grep denied
```

The problem is present if an error similar to the following is returned:

```
type=AVC msg=audit(1555533990.045:1826142): avc:  denied  { name_connect
} for  pid=25516 comm="nginx" dest=3005 scontext=system_u:system_r:
httpd_t:s0
```

For more information on this issue, see *https://www.nginx.com/blog/using-nginx-plus-with-selinux*.

Solution:

The solution is to enable the following network connection through the operating system:

```
sudo setsebool -P httpd_can_network_connect 1
```

Restart the platform.

Change Listening Port

If you need to change the listening port for the Trifacta® platform, please complete the following instructions.

> **Tip:** This change most typically applies if you are enabling use of SSL. For more information, see *Install SSL Certificate*.

> **NOTE:** By default, the platform listens on port `3005`. All client browsing devices must be configured to enable use of this port or any port number that you choose to use.

Steps:

1. Login to the Trifacta node as an admin.
2. Edit the following file:

   ```
   /opt/trifacta/conf/nginx.conf
   ```

3. Edit the following setting:

```
server {
  listen 3005;
  ...
```

4. Save the file.
5. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
6. Locate the following setting:

```
"proxy.port": 3005,
```

7. Set this value to the same value you applied in `nginx.conf`.
8. Save your changes and restart the platform.

Supported Deployment Scenarios for Cloudera

Contents:

- *Supported Cloudera Distributions*
- *Supported Deployments*
  - *Deployment System*
  - *Running Environment*
  - *Platform Security*
  - *High Availability*
  - *Metadata Publishing*
- *Connectivity*
  - *Hadoop Connectivity*
  - *External Connectivity*
- *Notes*

Supported Cloudera Distributions

> **NOTE:** By default, Cloudera may be installed with Java JDK 1.7 or earlier. If so, you must upgrade each node in the cluster to Java JDK 1.8. For more information, see *https://www.cloudera.com/documentation/enterprise/latest/topics/cdh_ig_jdk_installation.html*.

For this release, the Trifacta® platform supports the following Cloudera versions.

> **NOTE:** Cloudera 6.0 and later requires use of native Hadoop libraries from the cluster. See *Configure for Spark*.

- Cloudera 6.2.x (recommended)
- Cloudera 6.1.x
- Cloudera 6.0.x

> **NOTE:** Spark 2.4 is not supported on Cloudera 6.0. Please use Spark 2.2. See *Configure for Spark*.

- Cloudera 5.16.x

> **NOTE:** Cloudera 5.14.x and 5.15.x are no longer supported. For best results, please upgrade your Hadoop distribution.

Notes:

- **Update Date:** July 29, 2019
- The Trifacta platform supports all variants of patch or point releases (X.Y.* and X.Y.*.* releases) through the Hadoop vendor's backwards compatibility policy.
- For individual versions of Hadoop components (such as HDFS, Spark, and Hive), the Trifacta platform sup ports the component version that is bundled with the vendor's package for the supported Hadoop distribution.
- For more information on how to set up your Hadoop distribution, please consult the documentation provided with your distribution or contact your distribution vendor.

Supported Deployments

> **NOTE:** Unless otherwise noted, all items listed below are supported across all Hadoop distribution versions listed above. Unlisted items are not supported. Please contact *Trifacta Support* or your sales representative for items not listed here.

Deployment System

> **NOTE:** The Trifacta platform software must be installed on a gateway node of the Cloudera cluster. For more information, see *System Requirements*.

Item	Description
Physical On Premise Machines	Supported.
VMWare / VXServer	Supported.

> **NOTE:** Deployment to an Amazon EC2 is supported. See *Supported Deployment Scenarios for AWS*.

Running Environment

Item	Description
Spark	Supported.
Trifacta Photon	Supported.

Platform Security

Item	Description
HDFS File Permissions	Supported.
HDFS Privileges	Supported through Sentry.

Hive Privileges	Supported through Sentry.
Kerberos-Enabled Hadoop Cluster	Supported. See *Configure for Kerberos Integration*.
Secure User Impersonation	Supported. See *Configure for Secure Impersonation*.

High Availability

Item	Description
Name Node, Resource Manager, HttpFS	Supported. See *Enable Integration with Cluster High Availability*.

Metadata Publishing

Item	Description
Cloudera Navigator	Not supported.
Hive Publishing	Supported. See *Configure for Hive*.
Redshift Publishing	Supported. See *Run Job Page*. See *Publishing Dialog*.

Supported File Formats

See *Supported File Formats*.

Connectivity

Hadoop Connectivity

The Trifacta platform supports connectivity for execution to the following Hadoop environments for this vendor's distributions. Connectivity exceptions are listed below:

Running Environment	HDFS Reader	HDFS Writer	Hive Reader w/ HiveServer2
Spark	Supported.	Supported.	Supported.

Profiling Environment	HDFS Reader	HDFS Writer
Profiling on Spark	Supported.	Supported.

External Connectivity

Storage Platform	HDFS Reader	HDFS Writer
S3	Supported.	Supported.

Storage Platform	Amazon S3 Reader	Amazon S3 Writer
Spark Profiling	Supported.	Supported.

Notes

- none.

Supported Deployment Scenarios for Hortonworks

Contents:

- *Supported Hortonworks Distributions*
- *Supported Deployments*
  - *Deployment System*
  - *Running Environment*
  - *Platform Security*
  - *High Availability*
  - *Metadata Publishing*
- *Connectivity*
  - *Hadoop Connectivity*
  - *External Connectivity*
- *Notes*

Supported Hortonworks Distributions

For the following release, the Trifacta® platform supports the following Hortonworks versions.

> **NOTE:** Hortonworks 3.0 and later requires use of native Hadoop libraries. See *Configure for Spark*.

- Hortonworks 3.1.x

  > **NOTE:** Spark 2.4 is not supported on Hortonworks 3.1. Please use Spark 2.3. See *Configure for Spark*.

- Hortonworks 3.0.x

  > **NOTE:** Spark 2.4 is not supported on Hortonworks 3.0. Please use Spark 2.3. See *Configure for Spark*.

- Hortonworks 2.6.x

> **NOTE:** Hortonworks 2.4.x and Hortonworks 2.5.x are no longer supported. For best results, please upgrade your Hadoop distribution.

Notes:

- **Update Date:** July 29, 2019
- The Trifacta platform supports all variants of patch or point releases (X.Y.* and X.Y.*.* releases) through the Hadoop vendor's backwards compatibility policy.
- For individual versions of Hadoop components (such as HDFS, Spark, and Hive), the Trifacta platform supports the component version that is bundled with the vendor's package for the supported Hadoop distribution.
- For more information on how to set up your Hadoop distribution, please consult the documentation provided with your distribution or contact your distribution vendor.

Supported Deployments

NOTE: The Trifacta platform software must be installed on a Ambari/Hadoop client of the Hortonworks cluster. For more information, see *System Requirements*.

NOTE: After the Trifacta software has been installed, additional configuration is required for integration with the Hortonworks Data Platform. See *Configure for Hortonworks*.

NOTE: Unless otherwise noted, all items listed below are supported across all versions listed above. Unlisted items are not supported. Please contact *Trifacta Support* or your sales representative for items not listed here.

Deployment System

Item	Description
Physical On Premise Machines	Supported.
VMWare / VXServer	Supported.

NOTE: Deployment to an Amazon EC2 is supported. See *Supported Deployment Scenarios for AWS*.

Running Environment

Item	Description
Spark	Supported.
Trifacta Photon	Supported.

Platform Security

Item	Description
HDFS File Permissions	Supported.
HDFS Privileges	Supported through Ranger.
Hive Privileges	Supported through Ranger.
Kerberos-Enabled Hadoop Cluster	Supported. See *Configure for Kerberos Integration*.
Secure User Impersonation	Supported. See *Configure for Secure Impersonation*.

High Availability

Item	Description
Name Node, Resource Manager, HttpFS	Supported. See *Enable Integration with Cluster High Availability*.

Metadata Publishing

Item	Description
Hive Publishing	Supported. See *Configure for Hive*.
Redshift Publishing	Supported. See *Run Job Page*. See *Publishing Dialog*.

File Formats

See *Supported File Formats*.

Connectivity

Hadoop Connectivity

The Trifacta platform supports connectivity for execution to the following Hadoop environments for this vendor's distributions.

Running Environment	HDFS Reader	HDFS Writer	Hive Reader w/ HiveServer2
Spark	Supported.	Supported.	Supported.

Profiling Environment	HDFS Reader	HDFS Writer
Profiling on Spark	Supported.	Supported.

External Connectivity

Storage Platform	HDFS Reader	HDFS Writer
S3	Supported.	Supported.

Storage Platform	Amazon S3 Reader	Amazon S3 Writer
Spark Profiling	Supported.	Supported.

Notes

- None.

Uninstall

To remove Trifacta® Wrangler Enterprise, execute as root user one of the following commands on the Trifacta node.

> **NOTE:** All platform and cluster configuration files are preserved. User metadata is preserved in the Trifact a database.

CentOS/RHEL:

```
sudo rpm -e trifacta
```

Ubuntu:

```
sudo apt-get remove trifacta
```

Configure for Hadoop

Contents:

- *Before You Begin*
  - *Key deployment considerations*
  - *Platform configuration*
  - *Specify Trifacta user*
- *Configuration for Hadoop*
  - *Data storage*
  - *Configure ResourceManager settings*
  - *Default Hadoop job results format*
  - *Specify distribution client bundle*
  - *Authentication*
  - *Hadoop KMS*
  - *Hadoop distribution version*
  - *Hive access*
  - *High availability environment*
  - *Acquire Hadoop cluster configuration files*
- *Debugging*

The Trifacta® platform supports integration with a number of Hadoop distributions, using a range of components within each distribution. This page provides information on the set of configuration tasks that you need to complete to integrate the platform with your Hadoop environment.

Before You Begin

Key deployment considerations

1. **Hadoop cluster:** The Hadoop cluster should already be installed and operational. As part of the install preparation, you should have prepared the Hadoop platform for integration with the Trifacta platform. See *Prepare Hadoop for Integration with the Platform*.
   1. For more information on the components supported in your Hadoop distribution, See *Install Reference*.
2. **Storage:** on-premises, cloud, or hybrid.
   1. The Trifacta platform can interact with storage that is in the local environment, in the cloud, or in some combination. How your storage is deployed affects your configuration scenarios. See *Storage Deployment Options*.
3. **Base storage layer:** You must configure one storage platform to be the base storage layer. Details are described later.

NOTE: Some deployments require that you select a specific base storage layer.

After you have defined the base storage layer, it cannot be changed. Please review your *Storage Deployment Options* carefully. The required configuration is described later.

Platform configuration

After the Trifacta platform and its databases have been installed, you can perform platform configuration. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

NOTE: Some platform configuration is required, regardless of your deployment. See *Required Platform Configuration*.

Specify Trifacta user

NOTE: Where possible, you should define or select a user with a userID value greater than 1000. In some environments, lower userID values can result in failures when running jobs on Hadoop.

Set the Hadoop username `[hadoop.user` (default=`trifacta)]` for the Trifacta platform to use for executing jobs:

```
"hdfs.username": [hadoop.user],
```

If the Trifacta software is installed in a Kerberos environment, additional steps are required, which are described later.

Configuration for Hadoop

In the sections below are a series of questions about the Hadoop environment with which the Trifacta platform is integrating. Based on your answer, additional configuration may be required.

Data storage

The Trifacta platform supports access to the following Hadoop storage layers:

- HDFS
- S3

Set the base storage layer

At this time, you should define the base storage layer from the platform. See *Set Base Storage Layer*.

Required configuration for each type of storage is described below.

HDFS

If output files are to be written to an HDFS environment, you must configure the Trifacta platform to interact with HDFS.

- Hadoop Distributed File Service (HDFS) is a distributed file system that provides read-write access to large datasets in a Hadoop cluster. For more information, see *http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html*.

If your deployment is using HDFS, do not use the `trifacta/uploads` directory. This directory is used for storing uploads and metadata, which may be used by multiple users. Manipulating files outside of the Trifacta application can destroy other users' data. Please use the tools provided through the interface for managing uploads from HDFS.

Below, replace the value for `[hadoop.user` (default=`trifacta)]` with the value appropriate for your environment. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

```
"hdfs": {
  "username": "[hadoop.user]",
  ...
  "namenode": {
    "host": "hdfs.example.com",
    "port": 8080
  },
},
```

Parameter	Description
`username`	Username in the Hadoop cluster to be used by the Trifacta platform for executing jobs.
`namenode.host`	Host name of namenode in the Hadoop cluster. You may reference multiple namenodes.
`namenode.port`	Port to use to access the namenode. You may reference multiple namenodes. **NOTE:** Default values for the port number depend on your Hadoop distribution. See *System Ports*.

Individual users can configure the HDFS directory where exported results are stored.

NOTE: Multiple users cannot share the same home directory.

See *Storage Config Page*.

Access to HDFS is supported over one of the following protocols:

- See *WebHDFS* below.
- See *HttpFS* below.

WebHDFS

If you are using HDFS, it is assumed that WebHDFS has been enabled on the cluster. Apache WebHDFS enables access to an HDFS instance over HTTP REST APIs. For more information, see *https://hadoop.apache.org/docs/r1.0.4/webhdfs.html*.

The following properties can be modified:

```
"webhdfs": {
  ...
  "version": "/webhdfs/v1",
  "host": "",
  "port": 50070,
  "httpfs": false
},
```

Parameter	Description
`version`	Path to locally installed version of WebHDFS. **NOTE:** For `version`, please leave the default value unless instructed to do otherwise.
`host`	Hostname for the WebHDFS service. **NOTE:** If this value is not specified, then the expected host must be defined in `hdfs.namenode.host`.
`port`	Port number for WebHDFS. The default value is `50070`. **NOTE:** The default port number for SSL to WebHDFS is `50470` .
`httpfs`	To use HttpFS instead of WebHDFS, set this value to `true`. The port number must be changed. See *HttpFS* below.

Steps:

1. Set `webhdfs.host` to be the hostname of the node that hosts WebHDFS.
2. Set `webhdfs.port` to be the port number over which WebHDFS communicates. The default value is `50070`. For SSL, the default value is `50470`.
3. Set `webhdfs.httpfs` to false.
4. For `hdfs.namenodes`, you must set the `host` and `port` values to point to the active namenode for WebHDFS.

HttpFS

You can configure the Trifacta platform to use the HttpFS service to communicate with HDFS, in addition to WebHDFS.

NOTE: HttpFS serves as a proxy to WebHDFS. When HttpFS is enabled, both services are required.

In some cases, HttpFS is required:

- High availability requires HttpFS.
- Your secured HDFS user account has access restrictions.

If your environment meets any of the above requirements, you must enable HttpFS. For more information, see *Enable HttpFS*.

S3

The Trifacta platform can integrate with an S3 bucket. See *Enable S3 Access*.

Configure ResourceManager settings

Configure the following:

```
"yarn.resourcemanager.host": "hadoop",
"yarn.resourcemanager.port": 8032,
```

NOTE: Do not modify the other host/port settings unless you have specific information requiring the modifications.

For more information, see *System Ports*.

Default Hadoop job results format

For smaller datasets, the platform recommends using the Trifacta Photon running environment.

For larger datasets, if the size information is unavailable, the platform recommends by default that you run the job on the Hadoop cluster. For these jobs, the default publishing action for the job is specified to run on the Hadoop cluster, generating the output format defined by this parameter. Publishing actions, including output format, can always be changed as part of the job specification.

As needed, you can change this default format. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

```
"webapp.defaultHadoopFileFormat": "csv",
```

Accepted values: `csv`, `json`, `avro`, `pqt`

For more information, see *Run Job Page*.

Specify distribution client bundle

The Trifacta platform ships with client bundles supporting a number of major Hadoop distributions. You must configure the jarfile for the distribution to use. These distributions are stored in the following directory:

```
/opt/trifacta/hadoop-deps
```

Configure the bundle distribution property (`hadoopBundleJar`) in platform configuration. Examples:

Hadoop Distribution	`hadoopBundleJar` property value
Cloudera	`"hadoop-deps/cdh-x.y/build/libs/cdh-x.y-bundle.jar"`
Hortonworks	`"hadoop-deps/hdp-x.y/build/libs/hdp-x.y-bundle.jar"`

where:

`x.y` is the major-minor build number (e.g. 5.4)

> **NOTE:** The path must be specified relative to the install directory.

Authentication

Kerberos

The Trifacta platform supports integration with Kerberos security. The platform can utilize Kerberos' secure impersonation to broker interactions with the Hadoop environment.

See *Configure for Kerberos Integration*.

See *Configure for Secure Impersonation*.

Single Sign-On

The Trifacta platform can integrate with your SSO platform to manage authentication to the Trifacta application. See *Configure SSO for AD-LDAP*.

Hadoop KMS

If you are using Hadoop KMS to encrypt data transfers to and from the Hadoop cluster, additional configuration is required. See *Configure for KMS*.

Hadoop distribution version

> **Tip:** If there is no bundle for the distribution you need, you might try the one that is the closest match in terms of Apache Hadoop baseline. For example, CDH5 is based on Apache 2.3.0, so that client bundle will probably run ok against a vanilla Apache Hadoop 2.3.0 installation. For more information, see *Trifacta Support*.

Cloudera distribution

Some additional configuration is required. See *Configure for Cloudera*.

Hortonworks distribution

After install, integration with the Hortonworks Data Platform requires additional configuration. See *Configure for Hortonworks*.

Hive access

Apache Hive is a data warehouse service for querying and managing large datasets in a Hadoop environment using a SQL-like querying language. For more information, see *https://hive.apache.org/*.

See *Configure for Hive*.

High availability environment

You can integrate the platform with the Hadoop cluster's high availability configuration, so that the Trifacta platform can match the failover configuration for the cluster.

> **NOTE:** If you are deploying high availability failover, you must use HttpFS, instead of WebHDFS, for communicating with HDFS, which is described in a previous section.

For more information, see *Enable Integration with Cluster High Availability*.

Acquire Hadoop cluster configuration files

> **NOTE:** If the Trifacta node has been properly configured as a Hadoop Edge node, these files should already exist on the local node. The location of these files on the Hadoop cluster may vary based on Hadoop distribution, version, and enabled components. For more information, please contact your Hadoop administrator.

To enable the platform to use YARN installations, you must provide a set of client `*-site.xml` files.

`core-site.xml`

`hdfs-site.xml`

`httpfs-site.xml`

`mapred-site.xml`

`yarn-site.xml`

`hive-site.xml`

> **NOTE:** The above file is required if you are integrating with Hive, using the Spark running environment, or both. For more information, see *Configure for Hadoop*.

> **NOTE:** If these configuration files change in the Hadoop cluster, the versions installed on the Trifacta node should be updated, or components may fail to work. You may be better served by setting permissions on these files so that they can be read by the `[hadoop.user` (default=`trifacta)]` user and then creating a symlink from the Trifacta platform node.

Locate Client Configuration

For CDH 5:

In Cloudera Manager, select **Actions > Download Client Configuration**.

Configuration files are also available in `/etc/hadoop/conf` on any cluster edge node.

For HDP 2:

Client configuration files can be retrieved from an existing client node. Acquire `*-site.xml` files from `/etc/hadoop/conf`.

If you are using Hortonworks, you must complete the following modification to the site configuration file that is hosted on the Trifacta node.

> **NOTE:** Before you begin, you must acquire the full version and build number of your Hortonworks distribution. On any of the Hadoop nodes, navigate to `/usr/hdp`. The version and build number is a directory in this location, named in the following form: `A.B.C.D-XXXX`.

In the Trifacta deployment, edit the following file:

```
/opt/trifacta/conf/hadoop-site/mapred-site.xml
```

Perform the following global search and replace:

1. Search:

   ```
   ${hdp.version}
   ```

2. Replace with your hard-coded version and build number:

   ```
   A.B.C.D-XXXX
   ```

Save the file.

Restart the Trifacta platform.

For YARN:

YARN maintains site site configuration files in a similar location. These XML files should be retrieved, too.

Deploy Client Configuration

After you've collected the Hadoop client configuration, copy all `*-site.xml` files to the following:

```
<installation root>/conf/hadoop-site/
```

Restart services. See *Start and Stop the Platform*.

Debugging

You can review system services and download log files through the Trifacta application.

See *System Services and Logs*.

Configuration by Hadoop Distribution

The following sections contain additional configuration steps required to integrate the Trifacta® platform with supported versions of each specific distribution.

Topics:

- *Configure for Cloudera*
- *Configure for Hortonworks*

Configure for Cloudera

Contents:

- *Pre-requisites*
- *Configure Trifacta platform*
  - *Configure Hive Locations*
  - *Configure SSL for Hive*
- *Restart*

This section provides additional configuration requirements for integrating the Trifacta® platform with the Cloudera platform.

- This section applies only to the versions of CDH that Trifacta supports. For more information, see *Supported Deployment Scenarios for Hortonworks*.

> **NOTE:** Except as noted, the following configuration items apply to the latest supported version of Cloudera platform.

Pre-requisites

Before you begin, it is assumed that you have completed the following tasks:

1. Successfully installed a supported version of Cloudera platform into your enterprise infrastructure.
2. Installed the Trifacta software in your environment. For more information, see *Install Software*.
3. Reviewed the mechanics of platform configuration. See *Required Platform Configuration*.
4. Configured access to the Trifacta database. See *Configure the Databases*.
5. Performed the basic cluster integration configuration. See *Configure for Hadoop*.
6. You have access to platform configuration through the Trifacta node or through the Admin Settings page. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

Configure Trifacta platform

Configure Hive Locations

If you are enabling an integration with Hive on the cluster, there are some distribution-specific parameters that must be set. For more information, see *Configure for Hive*.

Configure SSL for Hive

CDH supports two methods of enabling SSL communications with Hive:

1. **SASL-QOP method:** Enable encryption between Hive JDBC and HiveServer 2 using SASL-QOP. This method is available by default with the Trifacta platform.
2. **TLS/SSL method:** Use TLS/SSL encryption for JDBC connections to HiveServer 2.

To determine the method in use:

1. In Cloudera Manager configuration, search for: `tls`.
2. If the options for TLS/SSL are enabled, please complete the following configuration steps.
3. If these options are not enabled, the cluster can still use the SASL-QOP method. For more information on this method, see *Configure for Hive* .

Enable TLS/SSL Method

Steps:

1. The default Hive JDBC driver provided with your Trifacta installation must be replaced with the drive provided by Cloudera. Please complete the following commands, noting the wildcards (*) in the JAR path:

   > **NOTE:** The current driver must be removed or replaced in the working directory. Do not leave it in the directory.

   ```
   cd /opt/trifacta/services/data-service/build/dependencies
   rm *hive*jdbc*
   cp /opt/cloudera/parcels/CDH-5.8*/jars/hive-jdbc-1.1.0-cdh5.8.0*jar
   .
   ```

2. Enable the Hive connection. For the Hive connection string options, you must specify something like the following:

   ```
       "connectStrOpts": ";ssl=true;sslTrustStore=</path/to
   /truststore>;trustStorePassword=<storePassword>"
   ```

   > **NOTE:** The truststore specified above must exist on the Trifacta node and be accessible to the Trif acta user through the listed password. This truststore must contain the certificate for the Hive server.

3. Save the parameters file. For more information on creating the connection, see *Configure for Hive*.
4. Restart the platform. See *Start and Stop the Platform*.
5. Verify that you can read from a Hive source through the application. See *Hive Browser*.

Restart

To apply your changes, restart the platform. See *Start and Stop the Platform*.

Configure Publishing to Cloudera Navigator

Contents:

- *Publishing Behavior*
- *Supported Versions*
- *Limitations*
- *Pre-requisites*
- *Enable Navigator Publish*
- *Additional Configuration for SSL*
- *Configure Custom Source Types*
- *Validate*

The Trifacta® platform can be configured to publish metadata about recipe and jobs to Cloudera Navigator, which provides data governance over the Cloudera cluster. This section describes how to enable and configure this integration.

- Cloudera Navigator is an integrated data management solution for the Cloudera platform, providing security, governance, discovery, and analysis across diverse datasets in the cluster. For more information, see *https://www.cloudera.com/resources/datasheet/cloudera-navigator-datasheet.html*.

Publishing Behavior

When this integration is enabled, recipe and job information is automatically published for all jobs executed on Photon or Spark. The following behaviors are applied to publishing:

- When a job completes, the Trifacta platform automatically attempts to publish a link to the job to Navigator.
- Job results are submitted to a queue for Cloudera Navigator to execute. The publishing time may take a while to complete.
- If the publication is successful, there is no need to execute any additional publishing to Navigator.

> **NOTE:** If the publication fails, you must re-run the job in the Trifacta platform. Ad-hoc publishing to Navigator of completed jobs is not supported.

- Success or failure of the publication to Cloudera Navigator can be found in the `job.log` file for the Trifacta job.

Supported Versions

Component	Supported Version(s)
Cloudera	5.16
Cloudera Navigator	2.15.1
Navigator API	9 or later (13 is recommended)

Limitations

- The Trifacta platform produces its own entities to reference S3 objects. In Navigator, this feature is undocumented, and ID generation for S3 endpoint proxies is broken in the current release of the Navigator SDK (*https://github.com/cloudera/navigator-sdk/issues/91*).
- Jobs that read or write from a Hive table using the Hive-remote JDBC connector cannot link with Navigator's entity for the Hive table. Instead, Navigator links to a JDBC table entity created by the Trifacta platform.

Jobs published to the following targets cannot also be published to Cloudera Navigator:

- Tableau Server

> **NOTE:** In Cloudera Navigator, platform jobs display only a single source of data, even if the job references multiple data sources.

Pre-requisites

1. The Trifacta platform must be installed, configured, and integrated with an existing instance of the Cloudera platform. Please see the Cloudera Navigator documentation for additional details.
2. The Trifacta node must have the Cloudera Manager port opened. The default port is `7187`.
3. You must have a Navigator user account with write permissions into the appropriate Navigator project.
4. **To enable SSL use:**
    1. A Java keystore and a sample CA certificate must be created on the node hosting Cloudera Manager.
    2. A valid, self-signed certificate must be created on the node hosting Cloudera Manager.
    3. In the order listed, the above certificates must be imported into the Java keystore.
    4. Retain the server path and the passwords for the keystore and certificates.
    5. For more information, see the documentation that was provided for your Cloudera Manager release.

Enable Navigator Publish

Please complete the following steps to enable publication to Cloudera Navigator.

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the `clouderaNavigator` properties.
3. Edit the following properties:

Property	Description
`"clouderaNavigator.enabled"`	When set to `true`, publication to Navigator is enabled.
`"clouderaNavigator.baseURL"`	Base URL of the Navigator instance where you are publishing. **NOTE:** The port number must be specified as part of the `baseURL`. Default value is `7187`.
`"clouderaNavigator.username"`	Username of the Navigator account to use to connect.
`"clouderaNavigator.password"`	Password of the Navigator account
`"clouderaNavigator.namespace"`	Namespace in Navigator where metadata is published.
`"clouderaNavigator.apiVersion"`	The version of the Cloudera Navigator API to use. **Tip:** This API version appears as part of the base URL. It should not be modified unless directed to do so.

4. If you are using HTTPS to connect to Cloudera Navigator, additional configuration is required. See below.
    1. Otherwise, set `clouderaNavigator.https.enabled` to `false`.

5. Save your changes.

Additional Configuration for SSL

To enable communication over SSL with Cloudera Navigator, please complete the following steps in Cloudera Manager and on the Trifacta node.

> **NOTE:** Before you begin, you must create valid certificates and import them into the Java keystore in the node hosting Cloudera Manager.

Steps:

1. Launch Cloudera Manager.
2. Select **MGMT**.
3. Click **Configuration**.
4. Click **Scope > Navigator Metadata Server Category > Security**.
5. Set Enable TLS/SSL for Navigator Metadata Server to `true`.
6. Set TLS/SSL for Navigator Metadata Server to the path where the Java keystore was created. The following is the default path:

```
/opt/cm_keystore.jks
```

7. Set TLS/SSL Keystore File Password to the password to the Java keystore.
8. Set TLS/SSL Keystore Key Password to the password to the certificate.
9. Restart the MGMT service.
10. The JKS file that you created must be transferred to an accessible location on the Trifacta node.
11. Login to the application.
12. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
13. Configure the following properties:

Property	Description
`"clouderaNavigator.https.enabled"`	Set this value to `true` to enable HTTPS communications with Navigator.
`"clouderaNavigator.https.trustStore.type"`	The format of the Java keystore file. Set this value to `jks`.
`"clouderaNavigator.https.trustStore.location"`	The absolute path on the Trifacta node to the Java keystore file. In the previous example, this value was the following: `/opt/cm_keystore.jks`
`"clouderaNavigator.https.trustStore.password"`	The password to the Java keystore file

14. Change the `clouderaNavigator.baseURL` value to use HTTPS.
15. Save your changes and restart the platform. See *Start and Stop the Platform*.

Configure Custom Source Types

In Cloudera Navigator, every listed entity is associated with a source. A **source** is a specified resource that is part of each job listing. Example sources include HDFS namenodes and Hive metadata servers. Each source has a specified source type. For more information on source types, see *https://github.com/cloudera/navigator-sdk/blob/master/model/src/main/java/com/cloudera/nav/sdk/model/SourceType.java*

For each source type:

- If you have only one source for a type, no further configuration is required.
- If you have multiple source types, you must specify your custom sources. For example, if you have multiple HDFS clusters, you must specify them in your custom sources. See below.

> **NOTE:** If you have multiple sources for a single type and have not completed the following configuration, the Trifacta platform publication job to Cloudera Navigator fails. In the job log, you can review the source and the source type identifiers that caused the failure.

Steps:

1. To list all of your Cloudera Navigator source types, visit the following URL:

```
http://<navigator_instance_url>:<port_number>/api/v<apiVersion>
/entities?query=type%3Asource
```

   where:

Property	Description
<navigator_instance_url>	The URL of your instance of Cloudera Navigator
<port_number>	The port number of your instance of Cloudera Navigator. Default value is `7187`.
<apiVersion>	The version number of the API in use.

2. Example: cURL with jq:

```
curl "http://username:password@cloudera-navigator-host:7187/api/v13
/entities?query=type%3Asource" | jq '[.[] | {type: .sourceType,
identity: .identity}]'
```

3. From the returned list, you can determine the source Id's to use for each source type.
4. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
5. Locate the following settings.

> **NOTE:** Leave these settings empty if you intend to use the default settings from Navigator.

Setting	Description
clouderaNavigator.customsources.YARN	Navigator identifier for the YARN resource manager to use
clouderaNavigator.customsources.SPARK	Navigator identifier for the Spark instance to use
clouderaNavigator.customsources.S3	Navigator identifier for the S3 bucket to use
clouderaNavigator.customsources.HIVE	Navigator identifier for the Hive metadata server to use
clouderaNavigator.customsources.HDFS	Navigator identifier for the HDFS cluster to use

6. You can specify the cluster to use by listing source type and mapping pairs under `customSources`. If you have multiple sources of a single type, this step disambiguates between them for the Trifacta platform.

    1. First you must acquire the identifiers to use from Cloudera Navigator via API.
        1. Navigator endpoint: `/v13/entities?query=type%3Asource`
        2. Request method: `GET`
        3. Example response:

```
[
  {
    "type": "S3",
    "identity": "9"
  },
  {
    "type": "YARN",
    "identity": "6"
  },
  {
    "type": "SPARK",
    "identity": "5"
  },
  {
    "type": "CLUSTER",
    "identity": "1"
  },
  {
    "type": "HDFS",
    "identity": "10"
  },
  {
    "type": "HDFS",
    "identity": "11"
  },
  {
    "type": "HDFS",
    "identity": "12"
  }
]
```

2. In the above example response are three HDFS clusters. Below, one of these clusters havs been specified by Id (11) to be used.

```
{
    ...,
    "clouderaNavigator": {
      ...,
      "customSources": {
        "HDFS": "11"
      }
    },
    ...
}
```

7. Save your changes and restart the platform.
8. Verify your mappings by running a job on the named source. See below.

Validate

Steps:

1. If you haven't done so already, restart the platform to apply the configuration changes. See *Start and Stop the Platform*.
2. Run a job.
3. When the job completes, open the job through the Jobs page. See *Jobs Page*.
4. Acquire the jobGroup Id for the job. It is the final value in the URL. In the following example, the jobGroup Id is 3:

```
http://example.com:3005/jobs/3
```

5. Login to Navigator. Search for the following string:

```
trifacta.<jobGroupId>
```

NOTE: It may take up to 30 minutes for results to be published to Navigator.

6. When you see one or more entries, such as the following, the job has been successfully published:

```
trifacta.14.wrangle.29
trifacta.14.filewriter.30
trifacta.14.filewriter.31
```

7. The above entries indicate the individual jobs within the job group that have been completed.

Configure for Hortonworks

Contents:

- *Hortonworks Cluster Configuration*
  - *Configure for Ranger*
  - *Configure for Spark Profiling*
  - *Set up directory permissions*
- *Configure Trifacta platform*
  - *Configure WebHDFS port*
  - *Configure Resource Manager port*
  - *Configure location of Hadoop bundle JAR*
  - *Configure Hive Locations*
- *Restart*

This section provides additional configuration requirements for integrating the Trifacta® platform with the Hortonworks Data Platform.

- This section applies only to the versions of HDP that Trifacta supports. For more information, see *Supported Deployment Scenarios for Hortonworks*.

> **NOTE:** Except as noted, the following configuration items apply to the latest supported version of Hortonworks Data Platform.

Pre-requisites

Before you begin, it is assumed that you have completed the following tasks:

1. Successfully installed a supported version of Hortonworks Data Platform into your enterprise infrastructure.
2. Installed the Trifacta software in your environment. For more information, see *Install Software*.
3. Reviewed the mechanics of platform configuration. See *Required Platform Configuration*.
4. Configured access to the Trifacta database. See *Configure the Databases*.
5. Performed the basic Hadoop integration configuration. See *Configure for Hadoop*.
6. You have access to platform configuration either via the Trifacta node or through the Admin Settings page.

Hortonworks Cluster Configuration

The following changes need to be applied to Hortonworks cluster configuration files or to configuration areas inside Ambari.

> **Tip:** Ambari is the recommended method for configuring your Hortonworks cluster.

Configure for Ranger

Configure Ranger to use Kerberos

If you have deployed Ranger in a Kerberized environment, you must verify and complete the following changes in Ambari.

Steps:

1. If you have enabled Ranger, navigate to **Hive > Configs > Settings**.
   1. Choose Authorization: **Ranger**.
   2. Hiveserver2 Authentication: **Kerberos**.

2. If you have enabled Ranger and Hive, navigate to **Hive > Configs > Advanced > General**.
    1. hive.security.authorization.manager: **org.apache.ranger.authorization.hive.authorizer.RangerHiveAuthorizerFactory**
3. Navigate to **Hive > Configs > Advanced > Advanced hive-site**.
    1. hive.security.authentication.manager: **org.apache.hadoop.hive.ql.security.SessionStateUserAuthenticator**
    2. hive.conf.restricted.list: **hive.security.authenticator.manager,hive.security.authorization.manager,*hive.users.in*.admin.role,hive.security.authorization.enabled**
4. Navigate to **Hive > Configs > Advanced > Custom hive-site**. Changes in this area update `hive-site.xml`.
    1. hadoop.proxyuser.trifacta.groups: `[hadoop.group` (default=`trifactausers)]`
    2. hadoop.proxyuser.trifacta.hosts: *****
    3. hive2.jdbc.url:**<your_jdbc_url>**
    4. hive.metastore.sasl.enabled: **true**
5. Save your configuration changes.

Configure for Spark Profiling

If you are using Spark for profiling, you must add environment properties to your cluster configuration. See *Configure for Spark*.

Additional configuration for Spark profiling on S3

If you are using S3 as your datastore and have enabled Spark profiling, you must apply the following configuration, which adds the `hadoop-aws` JAR and the `aws-java-sdk` JAR to the extra class path for Spark.

Steps:

1. In Ambari, navigate to **Spark2 > Configs**.
2. Add a new parameter to **Custom Spark2-defaults**.
3. Set the parameter as follows, which is specified for HDP 2.5.3.0, build 37:

```
spark.driver.extraClassPath=/usr/hdp/2.5.3.0-37/hadoop/hadoop-aws-2.7.3.2.5.3.0-37.jar:/usr/hdp/2.5.3.0-37/hadoop/lib/aws-java-sdk-s3-1.10.6.jar
```

4. Restart Spark from Ambari.
5. Restart the Trifacta platform.

Set up directory permissions

On all Hortonworks cluster nodes, verify that the YARN user has access to the YARN working directories:

```
chown yarn:hadoop /mnt/hadoop/yarn
```

If you are upgrading from a previous version of Hortonworks, you may need to clear the YARN user cache for the `[hadoop.user` (default=`trifacta)]` user:

```
rm -rf /mnt/hadoop/yarn/local/usercache/trifacta
```

Configure Trifacta platform

The following changes need to be applied to the Trifacta node.

Except as noted, these changes are applied to the following file in the Trifacta deployment:

```
/opt/trifacta/conf/trifacta-conf.json
```

Configure WebHDFS port

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. **WebHDFS:** Verify that the port number for WebHDFS is correct:

```
"webhdfs.port": <webhdfs_port_num>,
```

3. Save your changes.

Configure Resource Manager port

Hortonworks uses a custom port number for Resource Manager. You must update the setting for the port number used by Resource Manager. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

> **NOTE:** By default, Hortonworks uses 8050 for Resource Manager. Please verify that you have the correct port number.

```
"yarn.resourcemanager.port": 8032,
```

Save your changes.

Configure location of Hadoop bundle JAR

1. Set the value for the Hadoop bundle JAR to the appropriate distribution. The following is for Hortonworks 2.6:

```
"hadoopBundleJar": "hadoop-deps/hdp-2.6/build/libs/hdp-2.6-bundle.
jar"
```

2. Save your changes.

Configure Hive Locations

If you are enabling an integration with Hive on the Hadoop cluster, there are some distribution-specific parameters that must be set. For more information, see *Configure for Hive*.

Restart

To apply your changes, restart the platform. See *Start and Stop the Platform*.

After restart, you should verify operations. For more information, see *Verify Operations*.

Configure Hadoop Authentication

Contents:

- *End-User Authentication*
- *End-User Authorization*
  - *Security Scenarios for HDFS Access*
  - *Security Scenarios for Hive Access*

Depending on your Hadoop security environment, the following sections describe implications for the platform and provide links to appropriate documentation.

End-User Authentication

Depending on use of Single Sign On, Trifacta users access the application using the following credentials.

Security Features	Implications
Single Sign On (SSO)	Users access application using the LDAP/AD principal associated with their account. For more information, see *Configure SSO for AD-LDAP*.
All other security scenarios	Users access application using their Trifacta userId.

End-User Authorization

The following security scenarios apply to accessing Hadoop-based data storage.

Security Scenarios for HDFS Access

Depending on the following security features implemented in your Hadoop environment, your interactions with HDFS may have different implications.

Security Features	Implications
No Kerberos authentication	• All Trifacta users use the `[hadoop.user` (default=`trifacta`)`]` Hadoop user to access HDFS. • No security is applied.
• Kerberos authentication • No secure impersonation	• All Trifacta users authenticate and then use delegation token for all requests to HDFS. • If you receive an error when attempting to contact HDFS, your delegation token may have failed due to configuration error. Please contact your Trifacta administrator. • All Trifacta users use the `[hadoop.user]` Hadoop user to access HDFS.

• Kerberos authentication • Secure impersonation	• All Trifacta users authenticate with the `[hadoop.user]` user keytab. The `[hadoop.user]` user retrieves a delegation token on behalf of the user's Hadoop principal. • If you receive an error when attempting to contact HDFS, your delegation token may have failed due to a configuration error. Please contact your Trifacta administrator. • Trifacta users securely impersonate using their assigned Hadoop principal on HDFS.

For more technical information:

- See *Configure for Kerberos Integration*.
- See *Configure for Secure Impersonation*.

Security Scenarios for Hive Access

Depending on the following security features implemented in your Hadoop environment, your interactions with Hive may have different implications.

Security Features	Implications
No additional security features	• All Trifacta users use the `[hadoop.user]` Hadoop user to access Hive. • No security is applied.
• Kerberos authentication • No secure impersonation	• Trifacta users authenticate with the `[hadoop.user]` user keytab for all requests to Hive. • If you receive an error when attempting to contact Hive, authentication likely failed due to a configuration error. Please contact your Trifacta administrator.
• Kerberos authentication • Secure impersonation	• Trifacta users authenticate with the `[hadoop.user]` user keytab and then send proxying requests on behalf of the user's Hadoop principal. • If you receive an error when attempting to contact Hive, authentication likely failed due to a configuration error. Please contact your Trifacta administrator. • Hive is responsible for respecting proxy permissions, with the `hive` user itself proxying as `[hadoop.user]` proxying as the user's Hadoop principal.
• Kerberos authentication • Secure authentication • Sentry role-based access (Cloudera only) • Ranger role-based access (Hortonworks only)	• Trifacta users authenticate with the `[hadoop.user]` user keytab and then send proxying requests on behalf of the user's Hadoop principal. • If you receive an error when attempting to contact Hive, authentication likely failed due to a configuration error. Please contact your Trifacta administrator. • Hive executes access to the physical data file on HDFS as the Unix or LDAP user `hive`, which should be part of the group `[hadoop.group` (default=`trifactausers`)].
• Sentry role-based access (Cloudera only)	• Hive authorizes access with a Sentry lookaside. The `[hadoop.user]` user as well as the user's Hadoop principal should be configured with appropriate privileges and roles in Sentry.

• Kerberos authentication • No secure authentication • Sentry role-based access (Cloudera only) • Ranger role-based access (Hortonworks only)	• Trifacta users authenticate with the `[hadoop.user]` user keytab. • If you receive an error when attempting to contact Hive, authentication likely failed due to a configuration error. Please contact your Trifacta administrator. • Hive executes access to the physical data file on HDFS as the Unix or LDAP user `hive`, which should be part of the group `[hadoop.group` (default=`trifactausers)]`.

For more technical information:

- See *Configure for Kerberos Integration*.
- See *Configure for Secure Impersonation*.
- See *Configure for Hive with Sentry*.
- See *Configure for Hive with Ranger*.

Configure for Kerberos Integration

Hiemdal

Contents:

- *Pre-requisites for Kerberos integration*
- *Configure the KDC*
- *Create keytab in Active Directory environments*
- *Configure the Trifacta platform for Kerberos*
- *Configure Kerberos-delegated relational connections*

This document describes how to set up a Trifacta® user in Kerberos.

- Kerberos provides authentication services across a wide variety of platforms. See *http://www.kerberos.org/*.

Pre-requisites for Kerberos integration

Before you begin, please verify the following:

1. The `[hadoop.user` (default=`trifacta)]` user is created and enabled on each node in the Hadoop cluster.

   > **NOTE:** If LDAP is enabled, the `trifacta` user should be created in the same realm as the cluster.

2. On the Trifacta host, the directory `/opt/trifacta` is owned by the `[hadoop.user]` user .
3. The `[hadoop.user]` user exists on each node in the Hadoop cluster.

   > **NOTE:** The `[hadoop.user]` must have the same user ID and group ID on each node in the cluster. Depending on your cluster's configuration, this requirement may require an LDAP command. Configuring LDAP is beyond the scope of this document.

4. The `[hadoop.user]` user must be a member of any special group that is permitted to access HDFS or to run Hadoop jobs.

Configure the KDC

Steps:

1. On your KDC node, configure a Kerberos principal for the Trifacta platform:
    1. The principal's identifier has two parts: its **name** and its **realm**. For example, the principal `trifacta@HADOOPVAL.MSSVC.LOCAL` has the name `trifacta` and the realm `HADOOPVAL.MSSVC.LOCAL`.
    2. Retain the name and principal for later configuration.
2. Create a keytab file for the Trifacta principal. Command:

    ```
    kadmin xst -k trifacta.keytab <full principal identifier>
    ```

    where:
    `<full_principal_identifier>` is the principal identifier in Kerberos.

    On the KDC, you may have to run `kadmin.local` instead of `kadmin`. The rest of the arguments should remain the same.

    NOTE: If you're creating a keytab file in an AD environment, alternative instructions may need to be applied. See below.

3. Verify that the keytab is working. Command:

    ```
    klist -e -k -t trifacta.keytab
    ```

4. Copy the keytab to the Trifacta node in the following directory:
    `/opt/trifacta/conf/trifacta.keytab`
5. Configure the keytab file so that it is owned by the `[hadoop.user]` user. It should only be readable by that user.

    NOTE: Verify that all user principals that use the platform are also members of the group of the keytab user.

Create keytab in Active Directory environments

Some additional instructions are provided for the following environments.

For MIT Kerberos

See *https://kb.iu.edu/d/aumh* :

```
> ktutil
  ktutil:  addent -password -p username@EXAMPLE.COM -k 1 -e rc4-hmac
  Password for username@EXAMPLE.COM: [enter your password]
  ktutil:  addent -password -p username@EXAMPLE.COM -k 1 -e aes256-cts
  Password for username@EXAMPLE.COM: [enter your password]
  ktutil:  wkt username.keytab
  ktutil:  quit
```

For Heimdal Kerberos

```
  > ktutil -k username.keytab add -p username@EXAMPLE.COM -e arcfour-
hmac-md5 -V 1
```

If the keytab created in Heimdal does not work, you may need an `aes256-cts` entry. In this case, locate a machine with MIT Kerberos, and use the MIT Kerberos method instead.

Configure the Trifacta platform for Kerberos

You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

Locate the `kerberos` section, which controls Kerberos authentication.

Example configuration:

Substitute your own values in place of the example values as appropriate.

```
"kerberos.enabled": true,
"kerberos.principal": "trifacta",
"kerberos.kdc": "kdc.mssvc.local",
"kerberos.realm": "HADOOPVAL.MSSVC.LOCAL",
"kerberos.keytab": "/opt/trifacta/conf/trifacta.keytab"
"kerberos.principals.hive": "<UNUSED>",
"kerberos.principals.namenode": "nn/_HOST@EXAMPLE.COM"
"kerberos.principals.resourcemanager": "<YOUR_VALUE_HERE>",
```

Parameter	Description
enabled	To enable Kerberos authentication, set this value to `true`.
principal	The name part of the principal you created in the KDC
kdc	The host of the KDC
realm	Realm of the KDC
keytab	Directory in the Trifacta deployment where the Kerberos keytab file is stored

principals	List of jobtrackers and namenodes that are governed by Kerberos **NOTE:** `kerberos.principals.hive` is unused. This value must be inserted into the Hive connection definition. See *Create Hive Connections*. **NOTE:** If you don't know the values to use here, see *Set principal values* below. **NOTE:** If you don't specify principal names in the `principals` definition section, the default names are used: `mapred/<jobtracker host>@<realm>`. You should specify the principals explicitly.

At this point, you should be able to load files from HDFS and run jobs against the kerberized Hadoop cluster.

Set principal values for YARN

Check the following Hadoop config properties in `yarn-site.xml` :

```
principals.jobtracker = yarn.resourcemanager.principal
principals.namenode = dfs.namenode.kerberos.principal
```

Configure Kerberos-delegated relational connections

When Kerberos has been enabled in the platform, you can apply the global keytab to be used for SSO connections to relational sources of data. For more information, see *Enable SSO for Relational Connections*.

Configure for Secure Impersonation

Contents:

- *Users and groups for secure impersonation*
- *Hadoop configuration for secure impersonation*
- *HDFS Directories*
- *Trifacta configuration for secure impersonation*
- *Provisioning impersonated users*

In a Hadoop environment, **secure impersonation** enables the Trifacta® platform and its users to act as the signed-in user when performing actions on Hadoop. When enabled, you can leverage the permissions infrastructure in your Hadoop cluster to control privacy level, collaboration, and data sharing for your user base. For the Trifacta user, their jobs and job outputs are owned by the specified Trifacta user, instead of the Hadoop user `[hadoop.user]`. The Trifacta system account is required even in secure impersonation mode.

- This configuration is optional.
- For more information on Hadoop secure impersonation, see *http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/Superusers.html*.

Please complete these steps to to enable secure impersonation.

NOTE: Trifacta secure impersonation requires Kerberos to be applied to the Hadoop cluster. However, you can use Kerberos without enabling secure impersonation, if desired. See *Configure for Kerberos Integration*.

Users and groups for secure impersonation

On the Hadoop cluster, the Trifacta platform requires a common Unix or LDAP group containing the `[hadoop.user` (default=`trifacta)]` and all Trifacta users.

NOTE: In UNIX environments, usernames and group names are case-sensitive. Please be sure to use the case-sensitive names for users and groups in your Hadoop configuration and Trifacta configuration file.

NOTE: If the HDFS user has restrictions on its use, it is not suitable for use with secure impersonation. Instead, you should enable HttpFS and use a separate HttpFS-specific user account instead. For more information, see *Configure for Hadoop*.

Assuming this group is named `trifactausers`:

- Create a Unix or LDAP group `trifactausers`
- Make user `[hadoop.user]` a member of `trifactausers`
- Verify that all user principals that use the platform are also members of the `trifactausers` group.

Hadoop configuration for secure impersonation

In your Kerberos configuration, you must configure the user `[hadoop.user]` as a secure impersonation proxy user for Trifacta users.

NOTE: The following addition must be made to your Hadoop cluster configuration file. This file must be copied to the Trifacta node with the required other cluster configuration files. See *Configure for Hadoop*.

In `core-site.xml` on the Hadoop cluster, add the following configuration, replacing the values for `[hadoop.user]` and `[hadoop.group` (default=`trifactausers)]` with the values appropriate for your environment:

```
<!-- Trifacta secure impersonation -->
<property>
  <name>hadoop.proxyuser.[hadoop.user].hosts</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.proxyuser.[hadoop.user].groups</name>
  <value>[hadoop.group]</value>
</property>
```

HDFS Directories

Verify that the shared upload and job results directories are owned and writeable to the `trifactausers` group.

For more information on HDFS directories and their permissions, see *Prepare Hadoop for Integration with the Platform*.

Stricter directory permissions in an impersonated environment

By default, the directories and sub-directories of the locations for uploaded data and job results are set to `730` in an environment with secure impersonation enabled. This configuration allows impersonated users to do the following:

> **NOTE:** Stricter permissions sets can adversely affect users' ability to access shared flows.

- The `7` user-permission implies that individual users have full permissions over their own directories.
    - Individual users can read only data in their own upload directory below `/trifacta/uploads`.
- The `3` user-permission is used because the top-level directory is owned by the `[hadoop.user]` user. Each impersonated user in the `trifactausers` group requires write and execute permissions on their own directory to create it and manage it. This permission set implies that the `trifactausers` group has read and execute permissions over a user's upload directory.
- Without access-level controls, these permissions are inherited from the parent directory and have the following implications:
    - Since impersonated group users have execute permissions, they can list all directories in this area.
    - Since impersonated group users have write permissions, they can theoretically write to any other user's upload directory, although this directory is not configurable.

Within the upload area, each user of the Trifacta platform is assigned an individual directory. For simplicity, the permissions on these directories are automatically applied to the sub-directories. In an impersonated environment, an individual directory is owned by the Hadoop principal for the user, so if two or more users share the same Hadoop principal, they have theoretical access to each others' directories. This simple scheme can be replaced by a more secure method using access-level controls.

> **NOTE:** To enable these stricter permissions, access-level controls must be enabled on your Hadoop cluster. For more information, please see the documentation for your Hadoop distribution.

If access-level controls are enabled for your impersonated environment, you can apply stricter permissions on these sub-directories for additional security.

Steps:

The following steps apply `730` permissions to the top-level directory and `700` to all user sub-directories. With these stricter permissions on sub-directories, no one other than the user, including the `trifacta` user, can access the user's sub-directory.

1. The `/trifacta/uploads` value is the default value for upload location in HDFS.
    1. In an individual deployment, the directory setting is defined in platform configuration . You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*. Locate the value for `hdfs.pathsConfig.fileUpload` .

NOTE: The following steps can also be applied to the directory where job results are written. By default, this directory is `/trifacta/queryResults`. For more secure controls over job results, you should also retrieve the value for `hdfs.pathsConfig.batchResults`.

2. Replace the values in the following steps with the value from your configuration.

2. Before you begin, you should consider resetting all access-level controls on the upload directories and sub-directories:

```
hdfs dfs -setfacl -b /trifacta/uploads
```

3. The following command removes the application of the permissions from the `uploads` directory and any sub-directory to members of the default group. So, an individual group member's permissions are not automatically shared with the group:

```
hdfs dfs -setfacl -R -m default:group::--- /trifacta/uploads
```

where:
`<groupId>` is the group name for the impersonated users for the Trifacta platform. The default value is `trifactausers`.

4. The following command is required to enable all users to access dictionaries:

```
hdfs dfs -setfacl -R -m default:group::rwx /trifacta/uploads/0
```

5. The following step sets the permissions at the top level to `730` :

```
hdfs dfs -chmod 730 /trifacta/uploads
```

6. Sub-directory permissions are a combination of these permissions and any relevant access-level controls.
7. **Apply to queryResults directory:** Repeat the above steps for the `/trifacta/queryResults` directory as needed.
8. **ACL for Hive:** If you need to apply access controls to Hive, you can use the following:

```
hdfs dfs -setfacl -R -m default:user:hive:rwx /trifacta/queryResults
```

User directories for YARN

For YARN deployments, each Hadoop user must have a home directory for which the user has write permissions. This directory must be located in the following location within HDFS:

```
/user/<username>
```

where:

- <username> is the Hadoop principal to use.

> **NOTE:** For jobs executed on the default Trifacta Photon running environment, user output directories must be created with the same permissions as you want for the transform and sampling jobs executed on the server. Users may be able to see the output directories of other users, but output job files are created with the user umask setting (`hdfs.permissions.userMask`), as defined in platform configuration.

Example for Hadoop principal `myUser`:

```
hdfs dfs -mkdir /user/myUser
hdfs dfs -chown myUser /user/myUser
```

Optional:

```
hdfs dfs -chmod -R 700 /user/myUser
```

Trifacta configuration for secure impersonation

You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

Set the following parmeter to `true`.

```
"hadoopImpersonation" : true,
```

If you have enabled the Spark running environment for job execution, you must enable the following parameter as well:

```
"spark-job-service.sparkImpersonationOn" : true,
```

For more information, see *Configure Spark Running Environment*.

Umask permissions

Under secure impersonation, the Trifacta platform utilizes two separate umask permission sets. If secure impersonation is not enabled, the Trifacta platform utilizes the systemUmask for all operations.

> **NOTE:** Umask settings are three-digit codes for defining the bit switches for read, write, and execute permissions for users, groups, and others (in that order) for a file or directory. These settings are inverse settings. For example, the umask value of `077` enables read, write, and execute permissions for users and disables all permissions for groups and others. For more information, see *https://en.wikipedia.org/wiki/Umask*.

Name	Property	Description

userUmask	`hdfs.permissions.userMask`	Controls the output permissions of files and directories that are created by impersonated users. These permissions define private permission settings for individual users.
systemUmask	`hdfs.permissions.systemUmask`	Controls the output permissions of files and directories that are created by the Trifacta system user. These permissions also control resources for the admin user and resources that are shared between Trifacta users.

Notes:

- In a secure impersonation environment, systemUmask should be defined as `027` (the default value), which enables read access to shared resources for all users in the Trifacta group.
  - For greater security, it is possible to set the userUmask to `077`, which locks down individual user directories under `/trifacta/queryResults`. However, secure impersonation requires more permissions on the systemUMask to enable sharing of resources.
  - Please note that the permission settings for the admin user are controlled by systemUmask.

Provisioning impersonated users

> **NOTE:** A newly created user in the platform cannot log in unless provisioned by a platform administrator, even if self-registration is enabled. The administrator must apply the Hadoop principal to the account.

To provision users as admin, log in and visit the Admin Settings page from the drop-down menu on the top right. Locate the Users section and click **Edit Users**. If you have the Secure Hadoop Impersonation flag on with Kerberos enabled, you should see a Hadoop Principal column. From here, you can assign each user a Hadoop principal. Multiple users can share the same Hadoop principal, but each Trifacta user must have a Hadoop principal assigned to them.

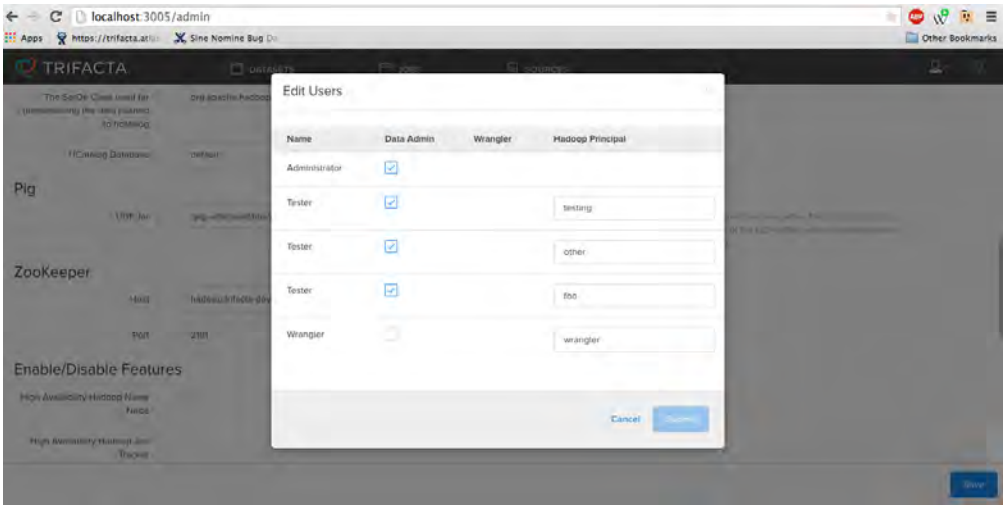


Figure: Editing users

Enable HttpFS

Contents:

- *Pre-requisites*

- *Configuration*
- *Enable SSL*

This section describes how to enable the Trifacta® platform to use the HttpFS service for communicating with Hadoop HDFS. HttpFS is commonly used in the following scenarios:

1. **High Availability.** WebHDFS does not support High Availability failover. You must use HttpFS instead.
2. **HDFS user is not available for secure impersonation.** If you have enabled secure impersonation in an environment where the HDFS superuser is restricted from use, you can enable HttpFS and use the HttpFS superuser for secure impersonation.

Pre-requisites

Before you begin, please verify that you have done the following in your environment:

- Enabled HDFS in your Hadoop cluster.
- Installed hadoop-httpfs into your Hadoop cluster.
- HttpFS has been enabled on a known port on the cluster.

> **NOTE:** If you are enabling HttpFS for use with High Availability, you should avoid enabling the HttpFS service on the primary namenode of the cluster. For more information, see *Enable Integration with Cluster High Availability*.

> **NOTE:** By default, HttpFS is available on port `14000`. Please verify the port number in use for your cluster.

- Started HttpFS service on the cluster.

Configuration

You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

Steps:

1. The configuration settings for HttpFS are within the HDFS configuration area:

```
"hdfs.webhdfs.host": "",
"hdfs.webhdfs.port": 14000,
"hdfs.webhdfs.httpfs": true,
```

2. Set `hdfs.webhdfs.httpfs` to `true`.
3. Specify the host and port for the HttpFS service. You can use one of the following methods:
    1. Specify `hdfs.webhdfs.host` and `hdfs.webhdfs.port` values to point to the node hosting HttpFS.
    2. Leave the `hdfs.webhdfs.host` value empty, in which case the platform falls back to using the namenode host as the WebHDFS host. Modify that value if required.

NOTE: By default, the platform expects this service to be available on port `14000`. Please apply the value that matches your cluster environment.

4. Save your changes and restart the platform.

Enable SSL

Optionally, you can enable secure (SSL) communications between the platform and HttpFS.

NOTE: The most secure method requires the creation and deployment of an SSL certificate for the HDFS instance. These steps provide instructions for how to do so.

If this certificate is not available, you can still enable communication over SSL over WebHDFS or HttpFS. Please skip steps 1 and 2 and complete the secure configuration without certificate export.

Steps:

1. Deploy a PEM file certificate that can be read by the `[os.user` (default=`trifacta)]` user account on the Trifacta node.

   NOTE: The following security configuration requires export of and access to an SSL certificate in PEM file format for the HDFS instance. Creation and deployment of this certificate exceeds the scope of this document. Please see the documentation provided with your Hadoop distribution.

   Certificates are commonly stored in Java keystores. They can be exported to PEM file format using the following command:

   ```
   keytool -exportcert -rfc  -alias <node_alias>   -storepass <pwd> -
   keystore cacerts -file <filename.pem>
   ```

   where:
   <pwd> is the keystore password.
   `<filename.pem>` is the output filename for the certificate.
   `<node_alias>` is the alias for the certificate in the keystore.
2. Place this generated certificate on the Trifacta node in a place where it is readable by the `[os.user` (default=`trifacta)]` user. The following location is suitable:

   ```
   /opt/trifacta
   ```
3. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
4. Locate the following setting and enable it:

Setting	Description
`"hdfs.webhdfs.ssl.enabled": true,`	Set to `true` to enable SSL communications with WebHDFS or (if enabled) HttpFS.

5. There is no need to update the port number. Port `14000` applies to HTTP and HTTPS.
6. **Security Level:** The level of security is determined by the following configuration options:

    1. Secure without certificate export:

Setting	Description
`"hdfs.webhdfs.ssl.certificateValidationRequired": false,`	Set to `false` to disable use of trusted certificate validation.
`"hdfs.webhdfs.ssl.certificatePath: "",`	Leave this value empty.

    2. Secure with certificate:

Setting	Description
`"hdfs.webhdfs.ssl.certificateValidationRequired": false,`	Set to `true` to require SSL use of trusted certificate validation.
`"hdfs.webhdfs.ssl.certificatePath: "",`	Configure the path on the Trifacta node to the location where you stored the certificate.

7. Save your changes and restart the platform.

Enable Integration with Compressed Clusters

Contents:

- *Pre-requisites*
- *Enable integration with compression*
- *Specify codecs*
- *Configure platform*

The Trifacta® platform can be configured to integrate with fully compressed Hadoop clusters. The following cluster compression methods are supported:

- Gzip
- Bzip2
- Snappy

Supported running environments:

- Trifacta Photon
- Spark

For more information, see *Running Environment Options*.

Hadoop clusters can be configured to enable compression of intermediate and/or final output data by default. The settings that are usually used to do so can be found in `mapred-site.xml` and `core-site.xml`.

Pre-requisites

> **NOTE:** If you have not done so already, you must retrieve cluster configuration files and store them on the Trifacta node. For more information, see *Configure for Hadoop*.

Enable integration with compression

Steps:

1. Edit the local version of `mapred-site.xml`. This file is typically located in `/etc/conf/hadoop`.
2. Add the following properties:

```
<configuration>
  ...
  <property>
    <name>mapreduce.map.output.compress</name>
    <value>true</value>
  </property>

  <property>
    <name>mapreduce.map.output.compress.codec</name>
    <value>org.apache.hadoop.io.compress.SnappyCodec</value>
  </property>

  <property>
    <name>mapreduce.output.fileoutputformat.compress</name>
    <value>true</value>
  </property>

  <property>
    <name>mapreduce.output.fileoutputformat.compress.type</name>
    <value>BLOCK</value>
  </property>

  <property>
    <name>mapreduce.output.fileoutputformat.compress.codec</name>
    <value>org.apache.hadoop.io.compress.SnappyCodec</value>
  </property>
  ...
</configuration>
```

3. Save the file and complete the following steps.

Specify codecs

One or more compression/decompression methods (codecs) must be specified in `core-site.xml`.

Steps:

1. Edit the local version of `mapred-site.xml`. This file is typically located in `/etc/conf/hadoop`.

2. Specify the codecs to use in the `io.compression.codecs` property. Supported values:

Code	Value
Gzip	`org.apache.hadoop.io.compress.GzipCodec`
Bzip2	`org.apache.hadoop.io.compress.BZip2Codec`
Snappy	`org.apache.hadoop.io.compress.SnappyCodec`

3. In the following example, all three codecs have been specified:

```
<configuration>
  ...
  <property>
    <name>io.compression.codecs</name>
   <value>org.apache.hadoop.io.compress.GzipCodec,org.apache.hadoop.
io.compress.BZip2Codec,org.apache.hadoop.io.compress.SnappyCodec<
/value>
  </property>
  ...
</configuration>
```

4. Save the file.

Configure platform

Apply the following changes from within the application to enable the Trifacta platform to communicate with the compressed cluster.

Steps:

1. Login to the application.
2. In the Admin Settings page, set the following settings:

Setting	Description
`hadoopDefaultClusterCompression.enabled`	To enable integration with a compressed cluster, set this value to `true`.

`hadoopDefaultClusterCompression.compression`	Set this value to the type of compression applied on the cluster:
	`none` - (default) no cluster compression
	`gzip`
	`bzip2`
	`snappy`

3. Save your changes and restart the platform.

Enable Integration with Cluster High Availability

Contents:

- *Enable HttpFS*
- *Enable HA Service*
- *Configure HA for Individual Components*
- *Update Active Namenode*
- *Configure HA in a Kerberized Environment*
- *Platform Restart*

In a Hadoop cluster, **high availability** provides failover support for one or more configured nodes. This section describes how to enable the Trifacta® platform to utilize the highly available set of nodes within the Hadoop cluster. High availability enables access to each node of the cluster configured for it, in the event of machine crash or software installation or upgrade.

If high availability is enabled on the Hadoop cluster, you must enable it on the Trifacta platform, which prevents integration conflicts between Hadoop-specific components in the platform and their cluster equivalents.

Enable HttpFS

The WebHDFS service does not directly support high availability. You must enable the related HttpFS service and specify a WebHDFS namenode to point to the server hosting the HttpFS service.

NOTE: Avoid enabling the HttpFS service on the primary namenode of the cluster. In the event that the node hosting the namenode fails over, the HttpFS service is no longer available. You may be required to manually set the active namenode and restart the Trifacta platform.

For more information, see *Enable HttpFS*.

Enable HA Service

To begin, you must enable the High Availability service in the Trifacta platform for the supported components. In platform configuration, each component has its own feature flag under `feature.highAvailability`.

You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

In the following example configuration, high availability has been disabled for resourcemanagers and enabled for namenodes:

NOTE: In almost all cases, `feature.highAvailability.resourceManager` should be set to `false`. For more information, see *Example - Configure resource manager* below.

```
"feature.highAvailability.namenode": true,
"feature.highAvailability.resourceManager": false,
```

Configure HA for Individual Components

High availability in Hadoop works by specifying a **nameservice** for a highly available component and then enumerating the hosts and ports as **children** of that nameservice node. These values must be explicitly specified in the platform configuration.

> **Service names and child names should be specified in the file as they appear in the cluster's configuration files.**

Example - Configure namenode

In the following example, the nameservice `namenodeha` provides high availability through two namenodes: `nn1` a nd `nn2`. In a high availability environment, these hosts are used for submitting jobs and writing data to HDFS.

```
"hdfs": {
  ...
  "highAvailability": {
    "serviceName": "namenodeha",
    "namenodes": {
      "nn1": {
        "host": "nn1.hadoop.mycompany.org",
        "port": 8020
      },
      "nn2": {
        "host": "nn2.hadoop.mycompany.org",
        "port": 8020
      }
    }
  }
},
```

Example - Configure resource manager

> **NOTE:** Set `feature.highAvailability.resourcemanager=true` only if the cluster file `yarn-site.xml` enables `yarn.resourcemanager.hostname.highlyavailableyarn`. This setting enables the cluster high availability for resourcemanager.
>
> Otherwise, set `feature.highAvailability.resourcemanager=false` for all environments. For HA environments, the resourcemanager hosts specified in the configuration below set the HA servers that are used by the Trifacta platform.

The following example specifies two failover nodes for the resource manager: `rm1` and `rm2`.

```
"yarn": {
 "resourceManagers": {
   "rm1": {
     "host": "rm1.yarn.mycompany.org",
     "port": 8032,
     "schedulerPort": 8030,
     "adminPort": 8033,
     "webappPort": 8042
   },
  "rm2": {
     "host": "rm2.yarn.mycompany.org",
     "port": 8032,
     "schedulerPort": 8030,
     "adminPort": 8033,
     "webappPort": 8042
   }
  }
}
```

Update Active Namenode

The active namenode used by the service must be configured explicitly. This value must be updated whenever the active namenode changes. Otherwise, HDFS becomes unavailable.

> **NOTE:** If the HttpFS service has been tied to the primary namenode of the cluster and that node fails, this setting must be manually configured to the new node and the platform must be restarted. Avoid tying HttpFS to the primary namenode.

In this example, the active namenode has been set to the `nn1` value in the previous configuration:

```
"webhdfs": {
 "proxy": { ... },
 "version": "/webhdfs/v1",
 "port": 14000,
 "httpfs": true
},
...
"namenode": {
 "host": "nn1.hadoop.mycompany.org",
 "port": 8020
 },
},
```

Configure HA in a Kerberized Environment

If you are enabling high availability in a Kerberized environment, additional configuration is required.

NOTE: WebHDFS does not support high availability/failover. You must enable HttpFS instead. For more information, see *Enable HttpFS*.

Steps:

1. If you have not done so already, acquire `httpfs-site.xml` from your Hadoop cluster.
2. Add the following settings to the file, replacing `[hadoop.user` (default=`trifacta)]` with the value appropriate for your environment:

```
<property>
 <name>httpfs.authentication.type</name>
 <value>org.apache.hadoop.security.token.delegation.web.
KerberosDelegationTokenAuthenticationHandler</value>
</property>
<property>
 <name>httpfs.authentication.delegation-token.token-kind</name>
 <value>WEBHDFS delegation</value>
</property>
<property>
 <name>httpfs.proxyuser.[hadoop.user].hosts</name>
 <value>*</value>
</property>
<property>
 <name>httpfs.proxyuser.[hadoop.user].groups</name>
 <value>*</value>
</property>
```

3. The above change must also be applied to the `httpfs-site.xml` configuration file for the cluster.

Save your changes and restart the platform.

Platform Restart

When high availability has been enabled, you must restart the platform from the command line. For more information, see *Start and Stop the Platform*.

Configure for Hive

Contents:

- *Pre-requisites*
- *Limitations*
- *Configure for Hive*
- *Enable appending to Hive tables without full permissions*
- *Validate Configuration*

This section describes how to enable the Trifacta® platform to read sources in Hive and write results back to Hive.

- A Hive source is a single table in a selected Hive database.

- Apache Hive is a data warehouse system for managing queries against large datasets distributed across a Hadoop cluster. Queries are managed using HiveQL, a SQL-like querying language. See *https://hive.apache.org/*.
- The platform can publish results to Hive as part of any normal job or on an ad-hoc basis for supported output formats.
- Hive is also used by the Trifacta platform to publish metadata results. This capability shares the same configuration described below.

Supported Versions:

Hive Version	Master namenode	Notes
Hive 1.x	HiveServer2	All supported Hadoop deployments
Hive 2.x	HiveServer2 (Interactive)	Supported on HDP 2.6 only.

Pre-requisites

1. HiveServer2 and your Hive databases must already be installed in your Hadoop cluster.

   NOTE: For JDBC interactions, the Trifacta platform supports HiveServer2 only.

2. You have verified that Hive is working correctly.
3. You have acquired and deployed the `hive-site.xml` configuration file into your Trifacta deployment. See *Configure for Hadoop*.

Limitations

1. Only one global connection to Hive is supported.
2. Changes to the underlying Hive schema are not picked up by the Trifacta platform and will break the source and datasets that use it.
3. During import, the JDBC data types from Hive are converted to Trifacta data types. When data is written back to Hive, the original Hive data types may not be preserved. For more information, see *Type Conversions*.
4. Publish to partitioned tables in Hive is supported.
   1. The schema of the results and the partitioned table must be the same.
   2. If they do not match, you may see an SchemaMismatched Exception error in the UI. You can try a drop and republish action on the data. However, the newly generated table does not have partitions.
   3. For errors publishing to partitioned columns, additional information may be available in the logs.

NOTE: Running user-defined functions for an external service, such as Hive, is not supported from within a recipe step. As a workaround, you may be able to execute recipes containing such external UDFs on the Photon running environment. Performance issues should be expected on larger datasets.

Configure for Hive

Hive user

The user with which Hive connects to read from the backend datastore should be a member of the user group `[hive.group` (default=`trifactausers)]` or whatever group is used to access storage from the Trifacta platform.

Verify that the Unix or LDAP group `[os.group` (default=`trifacta)]` has read access to the Hive warehouse directory.

Hive user for Spark:

> **NOTE:** If you are executing jobs in the Spark running environment, additional permissions may be required. If the Hive source is a reference or references to files stored elsewhere in backend storage, the Hive user or its group must have read and execute permissions on the source directories or files.

Enable Data Service

In platform configuration, the Trifacta data service must be enabled. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

Please verify the following:

```
"data-service.enabled": true,
```

Locate the Hive JDBC Jar

In platform configuration, you must verify that the following parameter is pointing to the proper location for the Hive JDBC JAR file. The example below identfies the location for Cloudera 5.10:

> **NOTE:** This parameter varies for each supported distribution and version.

```
"data-service.hiveJdbcJar": "hadoop-deps/cdh-5.10/build/libs/cdh-5.10-
hive-jdbc.jar",
```

Enable Hive Support for Spark Job Service

If you are using the Spark running environment for execution and profiling jobs, you must enable Hive support within the Spark Job Service configuration block.

> **NOTE:** The Spark running environment is the default running environment. When this change is made, the platform requires that a valid `hive-site.xml` cluster configuration file be installed on the Trifacta node.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the following setting and verify that it is set to `true`:

   ```
   "spark-job-service.enableHiveSupport" : true,
   ```

3. Modify the following parameter to point to the location where Hive dependencies are installed. This example points to the location for Cloudera 5.10:

NOTE: This parameter value is distribution-specific. Please update based on your specific distribution.

```
"spark-job-service.hiveDependenciesLocation":"%(topOfTree)s/hadoop-
deps/cdh-5.10/build/libs",
```

4. Save your changes.

Enable Hive Database Access for Spark Job Service

The Spark Job Services requires read access to the Hive databases. Please verify that the Spark user can access the required Hive databases and tables.

For more information, please contact your Hive administrator.

Configure managed table format

The Trifacta platform publishes to Hive using managed tables. When writing to Hive, the platform pushes to an externally staged table. Then, from this staging table, the platform selects and inserts into a managed table.

By default, the platform published to managed tables in Parquet format. As needed, you can apply the following values into platform configuration to change the format to which the platform writes when publishing a managed table:

- PARQUET (default)
- AVRO

To change the format, please modify the following parameter.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the following parameter and modify it using one of the above values:

```
"data-service.hiveManagedTableFormat": "PARQUET",
```

3. Save your changes and restart the platform.

Additional configuration for Hive 3.0

NOTE: Hive 3.0 is supported only on Hortonworks HDP 3.x using the Hive Warehouse Connector to read from Hive.

Tables in Hive 3.0 are ACID-compliant, transactional tables. Since Spark cannot natively read transactional tables, the Trifacta platform must utilize Hive Warehouse Connector to query the Hive 3.0 datastore for tabular data.

- The Hive Warehouse Connector connects to LLAP, which can run the Hive queries.
- Low Latency Analytics Processing (LLAP) is a Hortonworks framework that uses long-lived daemons in YARN for Hive query execution and in-memory caching of Hive data.
- For more information, see *https://hortonworks.com/blog/top-5-performance-boosters-with-apache-hive-llap/*.

NOTE: If Ranger is deployed on the cluster, Spark respects any column- or row-level security that Ranger enforces on the Hive tables. Queries for unauthorized data in a table fail in the Trifacta platform.

Please complete the following steps to integrate the Trifacta platform with Hive 3.0 through HDP 3.x and LLAP.

NOTE: Before you begin, please verify that you have performed the extra configuration for using Spark on HDP 3.x. For more information, see *Configure for Spark*.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Enable use of the Hive Warehouse Connector:

```
"spark-job-service.useHiveWarehouseConnector": true
```

3. Add the Hive Warehouse Connector to the Spark Job Service classpath. Example:

NOTE: If you have already configured for HDP 3.x, then the `(sparkBundleJar)` update below may have already been added.

```
classpath: "%(topOfTree)s/services/spark-job-server/server/build
/libs/spark-job-server-bundle.jar:%(sparkBundleJar)s:/etc/hadoop
/conf/:/etc/hive/conf/:%(topOfTree)s/%(hadoopBundleJar)s:/usr/hdp
/current/hive_warehouse_connector/*"
```

4. The following properties and values must be inserted in the `spark.props` section:

NOTE: These properties must be added to the Trifacta platform configuration. They cannot be read from Ambari.

```
"spark.datasource.hive.warehouse.load.staging.dir": "/tmp",
"spark.datasource.hive.warehouse.metastoreUri": "thrift://hdp30.
example:9083",
"spark.driver.extraLibraryPath": "/usr/hdp/current/hadoop-client/lib
/native:/usr/hdp/current/hadoop-client/lib/native/Linux-amd64-64",
"spark.executor.extraJavaOptions": "-XX:+UseNUMA",
"spark.executor.extraLibraryPath": "/usr/hdp/current/hadoop-client
/lib/native:/usr/hdp/current/hadoop-client/lib/native/Linux-amd64-
64",
"spark.hadoop.hive.llap.daemon.service.hosts": "@llap0",
"spark.hadoop.hive.zookeeper.quorum": "hdp30.example:2181",
"spark.sql.hive.hiveserver2.jdbc.url": "jdbc:hive2://hdp30.example:
2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2-
```

```
interactive",
"spark.sql.hive.hiveserver2.jdbc.url.principal": "hive
/_HOST@HORTONWORKS",
"spark.yarn.security.credentials.hiveserver2.enabled": "true",
"spark.yarn.jars": "local:/usr/hdp/current/spark2-client/jars/*"
```

The properties listed below require information from your HDP cluster. For the other properties, please use the listed values, unless otherwise required.

Property	Description
"spark.datasource.hive.metastoreUri"	URI for the Hive metastore. Copy the value from hive.metastore.uris. Example value: `thrift://mycluster-1.com:9083`
"spark.hadoop.hive.zookeeper.quorum"	A list of Zookeeper hosts used by LLAP. Copy the value from Advanced hive-site in Ambari: `hive.zookeeper.quorum`
"spark.sql.hive.hiveserver2.jdbc.url"	The URL for HiveServer2 Interactive. In Ambari, copy the value from the following: **Services > Hive > Summary > HIVESERVER2 INTERACTIVE JDBC URL**.
"spark.sql.hive.hiveserver2.jdbc.url.principal"	This property must be equal to `hive.server2.authentication.kerberos.principal`. In Ambari, copy the value for this property from the following: **Services > Hive >Configs > Advanced > Advanced hive-site**. The property value is in `hive.server2.authentication.kerberos.principal`.

For more information on these properties, see *https://docs.hortonworks.com/HDPDocuments/HDP3/HDP-3.1.0/integrating-hive/content/hive_configure_a_spark_hive_connection.html* .

5. Save your changes and restart the platform.

Create Hive Connection

NOTE: High availability for Hive is supported through configuration of the Hive connection.

For more information, see *Create Hive Connections*.

Optional Configuration

Depending on your Hadoop environment, you may need to perform additional configuration to enable connectivity with your Hadoop cluster.

Additional Configuration for Secure Environments

For secure impersonation

NOTE: You should have already configured the Trifacta platform to use secure impersonation. For more information on basic configuration, see *Configure for Secure Impersonation*.

You must add the Hive principal value to your Hive connection. Add the following principal value to the Connect String Options textbox.

```
"connectStrOpts": ";principal=<principal_value>",
```

For Kerberos with secure impersonation

NOTE: You should have already enabled basic Kerberos integration. For more information, see *Configure for Kerberos Integration*.

NOTE: If you are enabling Hive in a Kerberized environment, you must also enable secure impersonation. When connecting to Hive, Kerberos without secure impersonation is not supported. You should have already configured the Trifacta platform to use secure impersonation. For more information on basic configuration, see *Configure for Secure Impersonation*.

Additional Configuration for Sentry

The Trifacta platform can be configured to use Sentry to authorize access to Hive. See *Configure for Hive with Sentry*.

Enable appending to Hive tables without full permissions

Optionally, you can enable users to publish to Hive tables for which they do not have CREATE or DROP permissions.

If they have read (SELECT) and append (INSERT) permissions on a Hive schema, they can be permitted to append to the production schema using a separate schema that matches the production one. The Trifacta platform does the following:

1. CREATE a staging table in the schema specified in the User Profile.

   NOTE: These schemas must be created. Users must be given CREATE and DROP permissions on them.

2. INSERT the output data into the staging table.
3. Via INSERT, copy over data from the staging table to the production schema, effectively performing an append on the production table.

Steps:

NOTE: This feature must be enabled.

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Search for the following setting:

```
"feature.showHiveStagingDB": true,
```

3. Save your changes and restart the platform.

Use:

1. Staging schemas must be created in Hive.
2. Each user must insert the name of the staging schema in their user profile once. For more information, see *User Profile Page*.
3. When users generate results to Hive, they choose to publish to the production schema as an `append` operation.
    1. For more information, see *Run Job Page*.
    2. For more information, see *Publishing Dialog*.
4. Because this feature is enabled, the platform uses the specified staging schema and publishing mechanism to perform the `append` to the production schema.

Validate Configuration

> **NOTE:** The platform cannot publish to a default database in Hive that is empty. Please create at least one table in your default database.

Build example Hive dataset

Steps:

1. Download and unzip the following dataset: *Dataset-HiveExampleData*.
2. Store the dataset in the following example directory:

```
/tmp/hiveTest_5mb
```

3. Use the following command to create your table:

```
create table test (name string, id bigint, id2 bigint, randomName
string, description string, dob string, title string, corp string,
fixedOne bigint, fixedTwo int) row format delimited fields
terminated by ',';
```

4. Add the example dataset to the above test table:

```
load data local inpath '/tmp/hiveTest_5mb' into table test;
```

Check basic connectivity

Steps:

1. After restart, login to the Trifacta application. See *Login*.
2. If the platform is able to connect to Hive, you should see a Hive tab in the Import Data page. Create a new dataset and verify that the data from the Hive data has been ingested in the Transformer page.
3. If not, please check `/opt/trifacta/logs/data-service.log` for errors.

4. For more information, see *Verify Operations*.

Configure for Hive with Sentry

Contents:

- *Pre-requisites*
- *Secure Impersonation with Trifacta platform and Hive with Sentry*
- *Users and Groups for Sentry*
- *Configuration*
- *Basic Authentication*
- *Verify Operations*
- *Troubleshooting*

This section describes how to ensure that the Trifacta® platform is configured correctly to connect to Hive when Sentry is enabled for Hive. Sentry provides role-based authorization for Hive and other Hadoop components on the Cloudera platform.

- For more information, see *http://www.cloudera.com*.

Pre-requisites

Before you begin, please verify that your enterprise has deployed both Hive and Sentry according to recommended configuration practices. For more information, please consult the documentation that was provided with your Hadoop distribution.

NOTE: Before you begin, you must integrate the Trifacta platform with Hive. See *Configure for Hive*.

1. Enable the Sentry Service. Then, configure Hive to use the Sentry Service. See *http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/sg_sentry_service_config.html#concept_amg_2l2_xq_unique_2* .
2. (recommended) Enable secure impersonation. See below.

Secure Impersonation with Trifacta platform and Hive with Sentry

Secure impersonation ensures consistent and easily traceable security access to the data stored within your Hadoop cluster.

NOTE: Although not required, secure impersonation is highly recommended for connecting the platform with Hive.

The Trifacta platform requires the following additional configuration changes to maintain secure impersonation and work with Hive data:

1. Enable the platform with secure impersonation. See *Configure for Secure Impersonation* for details.
2. Give the local Hive user access to the Unix or LDAP group `[ldap.group` (default=`trifactausers)]`.
3. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
4. Set the following umask in `trifacta-conf.json`:

```
"hdfs.permissions.userUmask" = 027,
```

5. Verify that the Unix or LDAP group `[ldap.group]` has read access to the hive warehouse directory as specified in the following section. For more information, see *http://www.cloudera.com/content/www/en-us/documentation/enterprise/latest/topics/sg_sentry_service_config.html#concept_mlr_qxm_vq_unique_1* .
6. (Optional) Configuring Sentry to sync HDFS permissions will maintain user level access control for the underlying data files. For more information, see *http://www.cloudera.com/documentation/enterprise/5-4-x/topics/sg_hdfs_sentry_sync.html*

Users and Groups for Sentry

For Sentry, the following definitions and relationships apply.

Definition	Description
User	Individual account, as identified by the underlying authentication system
Group	A set of users maintained by the authentication system
Role	A set of privileges stored as a template to combine multiple access rules
Privilege	An instruction or rule allowing access to an object. Examples of Privileges include access to databases, tables, and the operations that can be executed.

In Sentry:

- Privileges can only be granted to Roles.
- A Group can be assigned to one or more Roles.
    - Users are assigned to a Group through the underlying authentication mechanism (e.g. operating system or LDAP).

Configuration

NOTE: Before you begin, you should determine the privileges that must be granted to Trifacta users based on your environment and needs.

1. Start Beeline as an administrative user.
2. Create a role for users of the Trifacta platform:

```
CREATE  ROLE trifactaUserRole;
```

3. Grant that role to the `[ldap.group` (default=`trifactausers)]` group associated with the platform:

```
GRANT ROLE trifactaUserRole TO GROUP trifacta;
```

4. Grant all privileges to this role for the filesystem area under which platform output is generated. The full URI is required. Example:

NOTE: Modify the grants as needed for your environment.

```
GRANT ALL ON URI 'hdfs://domain_example:8020/trifacta/queryResults
/user1@example.com/' to ROLE trifactaUserRole;
```

NOTE: If the above URI changes, the above grant must be reapplied to the new URI.

Basic Authentication

When the Trifacta platform is enabled with secure impersonation and submits requests to Hive, the following steps occur:

1. The platform authenticates as the `[hadoop.user]` user through Kerberos.
2. The Hive server authorizes access to the underlying table through Sentry as the Hadoop principal user assigned to the Trifacta user.

   NOTE: This Hadoop principal is the user that should be configured with appropriate privileges and roles in Sentry.

3. The Hive server executes access to the physical data file on HDFS as the Unix or LDAP user `hive`, which should be part of the designated group `[hadoop.group` (default=`trifactausers)]`.

NOTE: Since Sentry assigns privileges and roles to Unix groups, a common practice is to assign the Hadoop principal users (used by Trifacta users) to dedicated Unix groups that are **separate** from the Unix group `[os.group` (default=`trifacta)]` to use within Sentry. Sentry should not grant any privileges and roles to the Unix group `trifacta`.

NOTE: In UNIX environments, usernames and group names are case-sensitive. Please verify that you are using the case-sensitive names for users and groups in your Hadoop configuration and Trifacta configuration file.

Verify Operations

After you have completed your configuration changes, you should restart the platform. See *Start and Stop the Platform*.

To verify platform operations, run a simple job. For more information, see *Verify Operations*.

Troubleshooting

Cannot publish to Hive in a Kerberized environment with secure impersonation using Sentry

If you have deployed Sentry to manage access to a Kerberized environment using secure impersonation, you may encounter the following error when trying to write your results back to the Hadoop cluster:

NOTE: This issue is known to appear in Cloudera 5.7. It may not appear in later releases.

```
2015-09-02 20:49:54.111Z - WARN : com.trifacta.dataservice.
Controller      : Bad Request: org.springframework.jdbc.
BadSqlGrammarException: StatementCallback; bad SQL grammar [CREATE TABLE
`test_trifacta`  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.
AvroSerDe' STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.avro.
AvroContainerInputFormat' OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.
avro.AvroContainerOutputFormat' LOCATION 'hdfs://domain_example:8020
/trifacta/queryResults/user1@example.com/test/143/original_98.avro'
TBLPROPERTIES ('avro.schema.literal'='{"type":"record","name":"
GenericTrifactaRecord","fields":[{"name":"name","type":["null","
string"]},{"name":"id","type":["null","string"]},{"name":"id2","type":
["null","long"]},{"name":"randomname","type":["null","string"]},{"name":"
description","type":["null","string"]},{"name":"dob","type":["null","
string"]},{"name":"title","type":["null","string"]},{"name":"corp","
type":["null","string"]},{"name":"fixedone","type":["null","long"]},
{"name":"fixedtwo","type":["null","long"]}]}')]; nested exception is org.
apache.hive.service.cli.HiveSQLException: Error while compiling
statement: FAILED: SemanticException No valid privileges

Required privileges for this query: Server=server1-
>URI=hdfs://domain_example:8020/trifacta/queryResults/user1@example.com
/test/143/original_98.avro->action=*;
```

In this case, Sentry is failing to validate the URI permissions to allow the user (`user1@example.com`) to access the HDFS path, as the permissions have not been specifically granted to the required role. Sentry queries for authorization, fails, and throws the above exception.

The solution is to grant all access privileges for the user's Sentry role to Trifacta results directory for the target user. In the following example, access is granted to the `role2` role:

```
GRANT ALL ON URI 'hdfs://domain_example:8020/trifacta/queryResults
/user1@example.com/' to ROLE role2;
```

Since permissions in Sentry are recursive through the directories, the target directory for the specific job is covered. For more information on Sentry permissions, See Terminologies section in *http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cdh_sg_sentry.html*.

Configure for Hive with Ranger

Contents:

- *Pre-requisites*
- *Secure Impersonation with Trifacta platform and Hive with Ranger*
- *Users and Groups for Ranger*
- *Policies in Ranger*
- *Configuration*
- *Verify Operations*

This section describes how to ensure that the Trifacta® platform is configured correctly to connect to Hive when Ranger is enabled for Hive. Ranger provides role-based authorization for Hive and other Hadoop components on the Hortonworks platform.

Ranger effectively functions as a whitelist of URI's; by default, access is denied for any object in Hive. When a URI is requested, Ranger checks HDFS for permissions for the authenticated user. If HDFS denies access, then Ranger checks its defined set of URI's for the permission and, if a match is found, grants access for the authenticated user.

- For more information, see *http://hortonworks.com/blog/best-practices-for-hive-authorization-using-apache-ranger-in-hdp-2-2/*

Pre-requisites

Before you begin, please verify that your enterprise has deployed both Hive and Ranger according to recommended configuration practices. For more information, please consult the documentation that was provided with your Hadoop distribution.

NOTE: Before you begin, you must integrate the Trifacta platform with Hive. See *Configure for Hive*.

Secure Impersonation with Trifacta platform and Hive with Ranger

Secure impersonation ensures consistent and easily traceable security access to the data stored within your Hadoop cluster.

NOTE: Although not required, secure impersonation is highly recommended for connecting the platform with Hive.

Since secure impersonation for the combination of HiveServer2 and Ranger is not supported by Ranger, you must apply the following additional configuration changes to the Trifacta platform to enable secure impersonation in the environment:

1. Enable the platform with secure impersonation. See *Configure for Secure Impersonation* for details.
2. Add the hive service user `hive` to the Unix or LDAP group `[os.group` (default=`trifacta)]`.
3. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
4. Set the following parameter:

```
"hdfs.permissions.userUmask" = 027
```

5. Ensure that the Unix or LDAP group has read access to the Hive warehouse directory, as described in the following section. For more information, see *http://hortonworks.com/blog/best-practices-for-hive-authorization-using-apache-ranger-in-hdp-2-2/*.

Users and Groups for Ranger

When the Trifacta platform is enabled with secure impersonation and submits requests to Hive, the following steps occur:

1. The platform authenticates as the `[hadoop.user.principal` (default=`trifacta)]` user through Kerberos.

2. The Hive server authorizes access to the underlying table through Ranger as the Hadoop principal user assigned to `[hadoop.user.principal]`.

   > **NOTE:** This Hadoop principal is the user that should be configured through policies in Ranger to have the appropriate privileges.

3. The Hive server executes access to the physical data file on HDFS as the Unix user `hive`, which should be part of the group `[hadoop.group` (default=`trifactausers)]`.

> **NOTE:** Since Ranger assigns access to databases, tables, and columns to Unix users and groups, a common practice is to assign the Hadoop principal users (used by Trifacta users) to dedicated Unix groups that are separate from the Unix group `[os.group` (default=`trifacta)]` use within Ranger. Ranger should not grant any privileges and roles to the Unix group `[os.group` (default=`trifacta)]`.

> **NOTE:** In UNIX environments, usernames and group names are case-sensitive. Please verify that you are using the case-sensitive names for users and groups in your Hadoop configuration and Trifacta configuration file.

Policies in Ranger

In Ranger, you can configure access through policies. A Ranger **policy** is a combination of:

- Specified database, table, or tabled column
- Permissions associated with that specified object.
- Assignment of permissions to individual users or groups

Required Permissions

> **NOTE:** In general, to manage access through Ranger, permissions to underlying Hadoop components such as HDFS or Hive should be minimized within those components. All permissions in Ranger are additive, which means that you should be careful about overlapping users and groups.

The following components require these permissions at a minimum to be assigned to the Hadoop principal:

Component	Permissions
HDFS	Read, Write, Execute
Hive	Select, Update. Create (for Hive publishing)

Configuration

> **NOTE:** The following configuration is required for integration of HDP 2.6 or later with Hive publishing when Ranger is enabled.

1. In the Ambari console, navigate to the following: **HDFS > Configs > Advanced > Advanced ranger-hdfs-plugin-properties**.
2. Set the following to `true`: **Enable Ranger for HDFS**.
3. From the left nav bar, navigate to the following: **Ranger > Configs > Ranger Plugin tab**.
4. Set the following to `true`: **Hive Ranger Plugin**.

5. Please verify that the other Ambari properties are set to integrate Hive through Ranger. For more information, see the HDP documentation.
6. Restart the HDP cluster.
7. Open Ranger.
8. In the policies area, create the following two policies:

```
trifacta_policies
hive_warehouse
```

9. Set the following properties on these policies:
    1. Policy Type: `Access`
    2. Enabled: `true`
    3. Resource path:
        1. For `trifacta_policies`, set this value to either of the following values:

```
/trifacta
/trifacta/queryResults
```

        2. For `hive_warehouse`, set this value to the location of the Hive warehouse. The following is the default value:

```
/user/hive/warehouse
```

    4. Recursive: `true`

> **NOTE:** Policies must be recursive.

    5. Audit Logging: `yes`
    6. Allow conditions:
        1. Select group: `Hadoop`, `Trifacta`
        2. Select user: `Trifacta`
        3. Permissions: `Read`, `Write`, `Execute`
10. Save the policies.

Verify Operations

After you have completed your configuration changes, you should restart the platform. See *Start and Stop the Platform*.

To verify platform operations, run a simple job. For more information, see *Verify Operations*.

Configure for KMS

Hadoop KMS is a key management system that enables encrypted transport to and from the Hadoop cluster. This section describes how to configure the Trifacta® platform for integration with KMS.

NOTE: The Trifacta platform supports encryption at rest only through the KMS solution provided with the Hadoop distribution. Generic encryption at rest is not supported.

NOTE: If KMS is enabled on the cluster, you must configure KMS for the Trifacta platform regardless of other security features enabled on the cluster.

- For more information on KMS, see *https://hadoop.apache.org/docs/stable/hadoop-kms/index.html*.

NOTE: The required configuration for integrating with each Hadoop distribution may vary. Please be sure to review the details.

Pre-requisites

1. You have installed the Trifacta software. See *Install*.
2. You have performed the basic configuration steps for Hadoop. See *Configure for Hadoop*.
3. You have enabled any required secure authentication services.
    1. See *Configure for Kerberos Integration*.
    2. See *Configure for Secure Impersonation*.

Configure by Distribution Type

KMS is a cluster-wide configuration. If you are enabling Kerberos, secure impersonation, or encryption at rest on the cluster, you must perform the KMS site configuration changes in the pages for your specific Hadoop distribution.

Cloudera/Sentry: See *Configure for KMS for Sentry*.

Hortonworks/Ranger: See *Configure for KMS for Ranger*.

Configure for KMS for Sentry

Contents:

- *Configure Hadoop Cluster*
    - *Enable HDFS Encryption*
    - *Java KMS Configuration*
    - *Java KeyStore KMS Configuration*
    - *HDFS Configuration*
- *Validate*

This section describes how to configure the Trifacta® platform for integration with KMS system for Cloudera. It assumes that access to the cluster is gated by Sentry.

Before you begin, please verify the pre-requisites. See *Configure for KMS*.

Configure Hadoop Cluster

NOTE: These changes should be applied through the management console for the Hadoop cluster before pushing the client configuration files to the nodes of the cluster.

In the following sections:

- `[hadoop.user` (default=`trifacta)]` - the userID accessing the cluster component
- `[hadoop.group` (default=`trifactausers)]` -the appropriate group of user accessing the cluster component

Enable HDFS Encryption

On the Cloudera cluster, you may enable HDFS encryption using a designated Java KeyStore. For more information, see *http://www.cloudera.com/documentation/enterprise/latest/topics/sg_hdfs_encryption_wizard.html?scroll=concept_n2p_5vq_vt#concept_fcq_phr_wt_unique_1* .

Java KMS Configuration

Additional configuration for the Java KMS is required. See *http://www.cloudera.com/documentation/enterprise/latest/topics/cdh_sg_kms.html*.

Java KeyStore KMS Configuration

In the `kms-site.xml` configuration file, please locate the following properties:

> **NOTE:** If you have deployed Cloudera Manager for your cluster, do not modify these properties in the file. Make any modifications through the Cloudera Manager console.

```
<property>
  <name>hadoop.kms.authentication.kerberos.keytab</name>
  <value>${user.home}/kms.keytab</value>
</property>
```

In Cloudera Manager, you may wish to change the following safety value value. Navigate to **KMS service > Configuration > Advanced > Key Management Server Proxy Advanced Configuration Snippet (Safety Valve) for kms-site.xml**. Modify the following:

```
<property>
    <name>hadoop.kms.aggregation.delay.ms</name>
    <value>10000</value>
</property>
```

In the `kms-site.xml` file, insert the following properties, which are required properties for the Key Management Server Advanced Configuration safety value:

```
<property>
  <name>hadoop.kms.authentication.kerberos.principal</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].groups</name>
  <value>[hadoop.group]</value>
</property>
```

```
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].hosts</name>
  <value>*</value>
</property>
```

HDFS Configuration

In `httpfs-site.xml`, please insert the following properties, which are the safety value for HttpFS Advanced Configuration:

```
<property>
  <name>httpfs.proxyuser.[hadoop.user].groups</name>
  <value>[hadoop.group]</value>
</property>
<property>
  <name>httpfs.proxyuser.[hadoop.user].hosts</name>
  <value>*</value>
</property>
```

Save the files.

Validate

After the configuration is complete, you can try to import a dataset from a source stored in a cluster location managed by KMS, assuming that any required authentication configuration has been completed. See *Import Data Page*.

For more information, see *Configure Hadoop Authentication*.

Configure for KMS for Ranger

Contents:

- *Configure Hadoop Cluster*
  - *Add Trifacta user properties to KMS site file*
  - *Configuration for Ranger*
- *Validate*

This section describes how to configure the Trifacta® platform for integration with KMS system for Hortonworks Data Platform. It assumes that access to the cluster is gated by Ranger.

Before you begin, please verify the pre-requisites. See *Configure for KMS*.

Configure Hadoop Cluster

> **NOTE:** These changes should be applied through the management console for the Hadoop cluster before pushing the client configuration files to the nodes of the cluster.

In the following sections:

- `[hadoop.user` (default=`trifacta)]` - the userID accessing the cluster component
- `[hadoop.group` (default=`trifactausers)]` -the appropriate group of user accessing the cluster component

Add Trifacta user properties to KMS site file

In Ambari on the Hortonworks cluster, navigate to **KMS > Configs > Advanced > kms-site**. Add the following properties:

```
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].users</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].groups</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].hosts</name>
  <value>*</value>
</property>
```

Configuration for Ranger

If you are using Ranger's Key Management System, additional configuration is required.

- For more information on installing KMS, see *http://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.3.2/bk_Ranger_KMS_Admin_Guide/content/ch02s01.html* .

NOTE: These changes apply to the Hortonworks cluster only. Make changes through Ambari; avoid editing configuration files directly. Configuration files do not need to be shared with the Trifacta platform.

KMS Configuration for Hive

If you are using Hive, please add the Hive users and groups information to `kms-site.xml`:

```
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].users</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].groups</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.[hadoop.user].hosts</name>
  <value>*</value>
</property>
```

Verify Kerberos authentication for KMS

If Kerberos is deployed, edit `kms-site.xml` and verify the following properties in `kms-site.xml`:

```
<property>
  <name>hadoop.kms.authentication.type</name>
  <value>kerberos</value>
  <description> Authentication type for the KMS. Can be either &quot;
simple&quot; or &quot;kerberos&quot;.</description>
</property>
<property>
  <name>hadoop.kms.authentication.kerberos.keytab</name>
  <value>/etc/security/keytabs/spnego.service.keytab</value>
  <description> Path to the keytab with credentials for the configured
Kerberos principal.</description>
</property>
<property>
  <name>hadoop.kms.authentication.kerberos.principal</name>
  <value>HTTP/FQDN for KMS host@YOUR HADOOP REALM</value>
  <description> The Kerberos principal to use for the HTTP endpoint. The
principal must start with 'HTTP/' as per the Kerberos HTTP SPNEGO
specification.</description>
</property>
```

Verify users for KMS

If you are using Kerberos KMS authentication, verify the following properties in `kms-site.xml`:

```
<property>
  <name>hadoop.kms.proxyuser.hdfs.users</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.hdfs.groups</name>
  <value>*</value>
</property>
<property>
  <name>hadoop.kms.proxyuser.hdfs.hosts</name>
  <value>*</value>
</property>
```

Configure connection to the KMS node

> **NOTE:** The following changes need to be applied to the Hortonworks cluster configuration files and then shared with the Trifacta node. For more information on the files required by the platform, see *Configure for Hadoop*.

Changes to `core-site.xml` for KMS

Edit `core-site.xml` and make the following change:

```
hadoop.security.key.provider.path=kms://http@<KMS_HOST>:9292/kms
```

Changes to `hdfs-site.xml` for KMS

Edit `hdfs-site.xml` and make the following change:

```
dfs.encryption.key.provider.uri=kms://http@<KMS_HOST>:9292/kms
```

Changes `dbks-site.xml` for KMS

> **NOTE:** The following changes is required only if Ranger's KMS system is enabled. If so, this change enables access to files that are stored in secured folders.

Edit `dbks-site.xml` and make the following change:

> **NOTE:** If the existing value is `hdfs`, you may leave it as-is.

```
update property hadoop.kms.blacklist.DECRYPT_EEK='-'
```

Save the files.

Validate

After the configuration is complete, you can try to import a dataset from a source stored in a cluster location managed by KMS, assuming that any required authentication configuration has been completed. See *Import Data Page*.

For more information, see *Configure Hadoop Authentication*.