



TRIFACTA

Install Guide for Azure

Version: 9.2
Doc Build Date: 07/29/2022

Copyright © Trifacta Inc. 2022 - All Rights Reserved. CONFIDENTIAL

These materials (the “Documentation”) are the confidential and proprietary information of Trifacta Inc. and may not be reproduced, modified, or distributed without the prior written permission of Trifacta Inc.

EXCEPT AS OTHERWISE PROVIDED IN AN EXPRESS WRITTEN AGREEMENT, TRIFACTA INC. PROVIDES THIS DOCUMENTATION AS-IS AND WITHOUT WARRANTY AND TRIFACTA INC. DISCLAIMS ALL EXPRESS AND IMPLIED WARRANTIES TO THE EXTENT PERMITTED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE AND UNDER NO CIRCUMSTANCES WILL TRIFACTA INC. BE LIABLE FOR ANY AMOUNT GREATER THAN ONE HUNDRED DOLLARS (\$100) BASED ON ANY USE OF THE DOCUMENTATION.

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

1. <i>Install Overview</i>	4
1.1 <i>Install for High Availability</i>	6
1.2 <i>Install for Azure</i>	10
2. <i>Install Software</i>	14
2.1 <i>Install Dependencies without Internet Access</i>	15
2.2 <i>Install for Docker</i>	19
2.3 <i>Install on CentOS and RHEL</i>	29
2.4 <i>Install on Ubuntu</i>	35
2.5 <i>License Key</i>	41
3. <i>Start and Stop the Platform</i>	44
4. <i>Login</i>	47
5. <i>Install Configuration</i>	48
5.1 <i>Install Config for Azure</i>	49
6. <i>Install Reference</i>	60
6.1 <i>Install SSL Certificate</i>	61
6.2 <i>Change Listening Port</i>	66
6.3 <i>Supported Deployment Scenarios for Azure</i>	67
6.4 <i>Uninstall</i>	69

Install Overview

Contents:

- *Required Documents*
 - *Basic Install Workflow*
 - *Installation Scenarios*
 - *Install On-Premises*
 - *Install for AWS*
 - *Install for Azure*
 - *Install for Docker*
 - *Install Errata*
 - *Notation*
-

Required Documents

If you do not have access to online documentation, please verify that you have the following PDF documents, which are part of or are referenced during the installation process.

Tip: You should be able to install and configure the Trifacta platform using only the Install Guide. However, if you have additional requirements or require further explanation than what is provided in the Install Guide, these documents are important references.

NOTE: For AWS Marketplace or Azure Marketplace installs, the content available through the Marketplace should contain all documentation required to complete the installation.

Document	Starting Online Link	Description
Planning Guide PDF	<i>Install Planning</i>	Requirements and pre-install preparation for the Trifacta node in your install environment Tip: If you have not done so already, please review this Guide and verify requirements against your environment. In particular, please verify the <i>Product Support Matrix</i> and <i>System Requirements</i> sections.
Databases Guide PDF	<i>Install Databases</i>	Install or upgrade the Trifacta databases for one of the supported database versions.
Install Guide PDF	<i>Install Overview</i>	Instructions for installing the software and configuring it for basic operations NOTE: There are separate Install Guides for each supported infrastructure. Please verify that you are using the appropriate one.
Configuration Guide PDF	<i>Configure</i>	Configure integrations with running environments, backend storage, and other data sources, as well as instructions for configuring the Trifacta platform
Admin Guide PDF	<i>Admin</i>	For the installation process, this Guide contains useful topics on backup and recovery and verifying operations of the Trifacta platform.

User Guide PDF	<i>Workflow Basics</i>	After installation and configuration is complete, this Guide can be helpful for references on how to use the product.
-------------------	----------------------------	---

Basic Install Workflow

1. Prepare the environment for your installation scenario.
2. Install the software.
3. Install the databases.
4. Start the platform and login.
5. Configure your installation.
6. Verify operations.

The install workflow is described in detailed in the page for your installation scenario.

Install Errata

Notation

In this guide, JSON settings may be provided in dot notation in either of the following forms.

For example, `webapp.selfRegistration` refers to a JSON block `selfRegistration` under `webapp`:

Form 1:

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

Form 2:

```
"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* in the Configuration Guide.

Limitations

The following limitations apply to this feature:

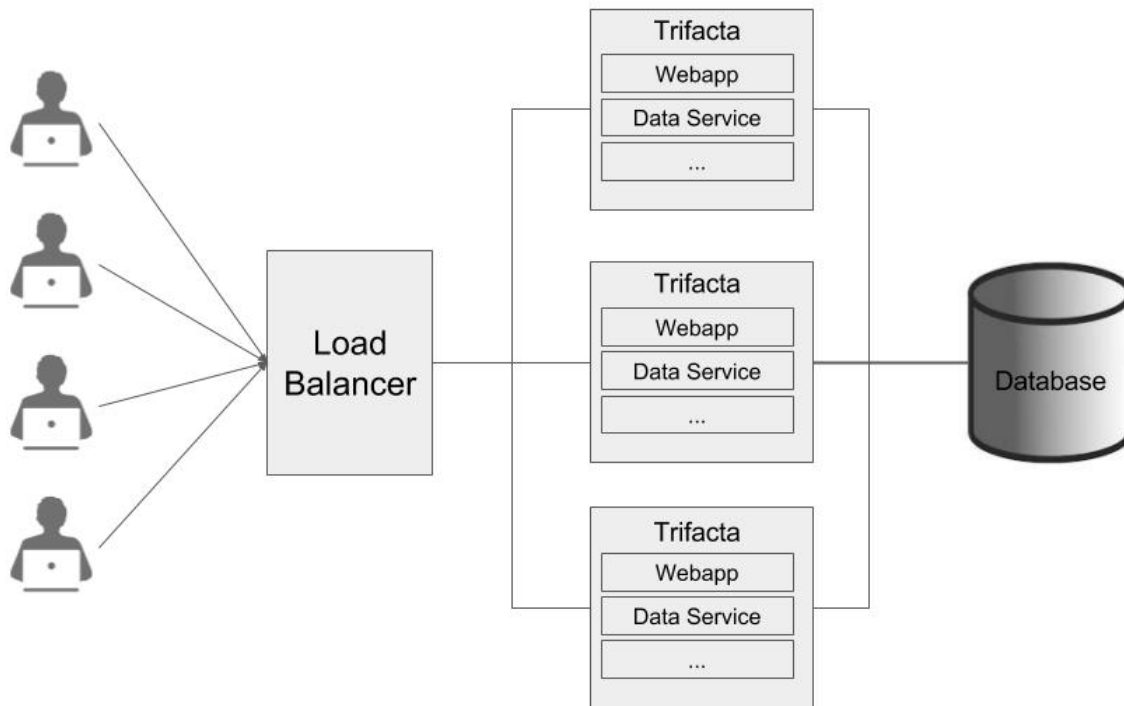
- This form of high availability is not supported for Marketplace installations.
- Job canceling does not work.
- When HA is enabled, the restart feature in the Admin Settings page does not work. You must restart using the command line.
- The platform must be installed on `/opt/trifacta` on every failover 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* in the Configuration Guide.

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 Self-Managed Enterprise Edition. 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* in the Planning Guide.
2. All nodes must have the appropriate ports opened. See *System Ports* in the Planning Guide.
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* in the Databases Guide.

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* in the Configuration Guide.

Install for Azure

Contents:

- *Scenario Description*
 - *Limitations*
 - *Deployment limitations*
 - *Product limitations*
 - *Prerequisites*
 - *Azure Desktop Requirements*
 - *Azure Prerequisites*
 - *Preparation*
 - *Deploy the Cluster*
 - *Deploy the Trifacta node*
 - *Install Workflow*
 - *Next Steps*
-

This install process applies to installing Trifacta® on an Azure infrastructure that you manage.

Azure Marketplace deployments:

NOTE: Content in this section does not apply to deployments from the Azure Marketplace. For more information, see the Azure Marketplace.

Scenario Description

NOTE: All hardware in use for supporting the platform is maintained within the enterprise infrastructure on Azure.

- Installation of Trifacta on a node in Microsoft Azure
- Installation of Trifacta databases on the same node
- Integration with a supported cluster for running jobs.
- Base storage layer and backend datastore of ADLS Gen1, ADLS Gen2, or WASB
- High availability or failover of the Trifacta node is not supported in Azure.

For more information on deployment scenarios, see *Supported Deployment Scenarios for Azure*.

Limitations

Deployment limitations

The following limitations apply to installations of Trifacta Self-Managed Enterprise Edition on Azure:

NOTE: If you are installing the databases in PostgreSQL, you must use PostgreSQL 11 for Azure installations. Please follow the instructions for installing PostgreSQL 12, modifying them to work with version 11.

NOTE: If you are using Azure Databricks as a datasource, please verify that openJDKv1.8.0_302 or earlier is installed on the Trifacta node. Java 8 is required. If necessary, downgrade the Java version and restart the platform. There is a known issue with TLS v1.3.

- The application user credentials are used to access to the cluster. Details are provided below.
- ADLS Gen1/Storage Blob access is only for the cluster's primary storage. Additional storage accounts are not supported.
- HDFS must be set as the base storage layer of the Trifacta platform. Details are provided later.
 - S3 integration and AWS-based integrations such as Redshift are not supported.
- Use of HttpFS is not supported.
- Security features such as Kerberos and secure impersonation are not supported.

Product limitations

For general limitations on Trifacta, see *Product Limitations*.

Prerequisites

Please acquire the following assets:

- **Install Package:** Acquire the installation package for your operating system.
 - **License Key:** As part of the installation package, you should receive a license key file. See *License Key* for details.
 - For more information, contact *Alteryx Support*.
- **Offline system dependencies:** 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*.

Azure Desktop Requirements

- All desktop users must be able to connect to the instance through the enterprise infrastructure.

Azure Prerequisites

Depending on which of the following Azure components you are deploying, additional prerequisites and limitations may apply:

- Cluster:
 - *Configure for Azure Databricks* in the Configuration Guide
- Storage:
 - *ADLS Gen1 Access* in the Configuration Guide
 - *WASB Access* in the Configuration Guide
 - *ADLS Gen2 Access* in the Configuration Guide
- Authentication:
 - *Configure SSO for Azure AD* in the Configuration Guide

Preparation

Before you begin, please verify that you have completed the following:

1. **Read:** Please read this entire document before you begin.
2. **Cluster sizing:** Before you begin, you should allocate sufficient resources for the cluster. For guidance, please contact your Trifacta representative.

3. **Node:** Review the system requirements for the node hosting the Trifacta platform. See *System Requirements* in the Planning Guide.
 - a. The required set of ports must be enabled for listening. See *System Ports* in the Planning Guide.
 - b. This node should be dedicated for Trifacta use.
4. **Databases:**
 - a. The platform utilizes a set of databases that must be accessed from the Trifacta node. Databases are installed as part of the workflow described later.

Deploy the Cluster

Cluster types: Deploy and provision a cluster of one of the supported types. The Trifacta platform supports integrations with multiple cluster types.

NOTE: Before you deploy, you should review cluster sizing options. For guidance, please contact your Trifacta representative.

Backend storage layer: Primary storage of the cluster may be set to an existing ADLS Gen1, ADLS Gen2, or WASB layer.

For more information, see *Supported Deployment Scenarios for Azure*.

Deploy the Trifacta node

In your Azure infrastructure, you must deploy a suitable VM for the installation of the Trifacta platform.

- For more information, see *System Requirements* in the Planning Guide.
- A set of ports must be opened on the VM for the platform. For more information, see *System Ports* in the Planning Guide.
- When you configure the platform to integrate with the cluster, you must acquire some information about the cluster resources. For more information on the set of information to collect, see *Product Support Matrix* in the Planning Guide.

For more information on the supported cluster distributions, see *Supported Deployment Scenarios for Azure*.

Install Workflow

NOTE: These steps are covered in greater detail later in this section.

The installation and configuration process requires the following steps. To continue, see Next Steps below.

1. **Install software:** Install the Trifacta platform software on the Trifacta 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* in the Databases Guide.

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:
 - a. See *Login*.

- b. As soon as you login, you should change the password on the admin account. In the left nav bar, select **User menu > Admin console > Admin settings**. Scroll down to Manage Users. For more information, see *Change Admin Password* in the Configuration Guide.

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

5. **Install configuration:** After you are able to successfully login to the Trifacta application, you must configure the product to work with your backend storage layer and the running environment on the cluster. See *Install Configuration*.

Next Steps

To continue, please install the Trifacta software on the Trifacta node.

NOTE: Please complete the installation steps for the operating system version that is installed on the Trifacta node.

See *Install Software*.

Install Software

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

Install Dependencies without Internet Access

Contents:

- *Install CentOS or RHEL dependencies without Internet access*
 - *Install software dependencies on CentOS or RHEL*
 - *Install database dependencies on CentOS or RHEL*
 - *Install database client on CentOS or RHEL*
- *Install Ubuntu dependencies without Internet access*
 - *Install software dependencies on Ubuntu*
 - *Install database dependencies on Ubuntu*
 - *Install database client on Ubuntu*

Offline dependencies should be included in the URL location that Alteryx® provided to you. Please use the `*dependencies*` 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 *Alteryx Support*.

Install CentOS or RHEL dependencies without Internet access

Install software dependencies on 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 xvfz <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:

Java 8:

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

6. **For CentOS 8.x:** If you are installing on CentOS 8.x, you must complete the following manual dependency install for NodeJS.

```
sudo yum --disablerepo=* --enablerepo=local nodejs-12.16.1-lnodesource.x86_64.rpm
```

Install database dependencies on CentOS or RHEL

If you are installing the databases on a CentOS node without Internet access, you can install the dependencies using the appropriate command:

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

For PostgreSQL 12.3:

```
sudo yum --disablerepo=* --enablerepo=local install postgresql12-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.

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

Install database client on CentOS or RHEL

If you are installing the databases on a remote server from the Trifacta node, then you must install the database client for your database distribution on the Trifacta node.

NOTE: The server dependencies include both server and client. This step is only required if you are installing the database server on a remote node from the Trifacta node.

PostgreSQL DB client:

Please complete the following steps to install the database client. Please modify the commands for CentOS /RHEL 8 and PostgreSQL 12.3.

1. Login to the Trifacta node.
2. If you have not done so already, download and unzip the dependencies for your distribution.

3. Remove the client if it exists on the node. The following removes the PostgreSQL 9.6 client:

```
yum list installed | grep postgresql
sudo yum erase postgresql96.x86_64 postgresql96-libs.x86_64
```

4. Verify that `psql` and `pgdump` are not present:

```
which psql
which pg_dump
```

5. Install the client from the RPM. The following installs the PostgreSQL 12 client for CentOS/RHEL 7.x:

```
cd /opt/trifacta/rpms/el/7/
sudo yum --disablerepo=* --enablerepo=local install postgresql-client-12
```

6. Verify that `psql` and `pgdump` are not present:

```
which psql
which pg_dump
```

7. Verify that `psql` is working and can connect to the remote DB server:

```
psql --host=<remote_db_hostname> --username=<username> --dbname=<database_name>
```

Above requires a password. For more information on default database names and usernames, see *Manual Database Install*.

MySQL 5.7 DB client:

You must license, download, and install the MySQL database client separately. For more information, see *Install Database Client for MySQL*.

Install Ubuntu dependencies without Internet access

Install software dependencies on Ubuntu

In an Ubuntu environment, you can use the following sequence of commands to install the dependencies without Internet access. The following example is for Release 7.6.0 and Ubuntu 16.04 (Xenial).

1. Login to the Trifacta node.
2. If you have not done so already, download and unzip the dependencies for your distribution.
3. Execute the following commands to unzip the TAR file and install the dependencies:

```
cd /opt
sudo mv trifacta-server-deps-7.6.0-ubuntu-16.04.tar.gz .
sudo gunzip trifacta-server-deps-7.6.0-ubuntu-16.04.tar.gz
sudo tar xvf trifacta-server-deps-7.6.0-ubuntu-16.04.tar
```

Install database dependencies on Ubuntu

If you are installing the databases on an Ubuntu node without Internet access, you can install the dependencies using the appropriate command:

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

1. Execute the following command on the TAR file to view the available PostgreSQL dependencies:

```
tar tvf trifacta-server-deps-7.6.0-ubuntu-16.04.tar.gz | grep -i pg | awk '{print $6}'
```

2. The available PostgreSQL dependencies are displayed:

```
trifacta-repo/postgresql-client-12_12.5-1.pgdg16.04+1_amd64.deb  
trifacta-repo/postgresql-12_12.5-1.pgdg16.04+1_amd64.deb
```

For PostgreSQL 12.3:

```
sudo dpkg -i postgresql-12_12.5-1.pgdg16.04+1_amd64.deb
```

For MySQL:

You must license, download, and install the MySQL database software separately.

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

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

Install database client on Ubuntu

If you are installing the databases on a remote server from the Trifacta node, then you must install the database client for your database distribution on the Trifacta node.

NOTE: The server dependencies include both server and client. This step is only required if you are installing the database server on a remote node from the Trifacta node.

DB client for PostgreSQL 12.3:

```
sudo dpkg -i postgresql-client-12_12.5-1.pgdg16.04+1_amd64.deb
```

DB client for MySQL 5.7:

You must license, download, and install the MySQL database client separately. For more information, see *Install Database Client for MySQL*.

Install for Docker

Contents:

- *Deployment Scenario*
 - *Limitations*
 - *Requirements*
 - *Infrastructure*
 - *Docker Daemon*
 - *Database client*
 - *Preparation*
 - *Acquire Image*
 - *Acquire from FTP site*
 - *Build your own Docker image*
 - *Configure Docker Image*
 - *Setup Container*
 - *Import Additional Configuration Files*
 - *Import license key file*
 - *Additional setup for Azure*
 - *Additional setup for Hadoop on-premises*
 - *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 Self-Managed Enterprise Edition deployed into a customer-managed environment: On-premises, AWS, or Azure.
- PostgreSQL 12.3 or MySQL 5.7 installed either:
 - Locally
 - Remote server
- On-premises Hadoop:
 - Connected to a supported Hadoop cluster.
 - Kerberos integration is supported.

Limitations

NOTE: For Docker installs and upgrades, only the dependencies for the latest supported version of each supported major Hadoop distribution are available for use after upgrade. For more information on the supported versions, please see the `hadoop-deps` directory in the installer. Dependencies for versions other than those available on the installer are not supported.

- 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. See *Supported Deployment Scenarios for Cloudera*.
- The base storage layer of the platform must be S3 or ABFS.
- 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 version must be compatible with the following version(s) of Docker Compose.
- Docker-Compose 1.24.1. Version must be compatible with your version of Docker.

Infrastructure

Before you begin a Dockerized install, please verify that your enterprise infrastructure meets the following integration requirements.

NOTE: The Docker image contains all components and requirements for the Trifacta node for the appropriate infrastructure. In the following pages, you should verify connectivity, account permissions, cluster and datastore availability, and other aspects of connecting the Trifacta node to your infrastructure resources.

Infrastructure	Documentation
On-premises Hadoop	<i>Install On-Premises</i>
AWS	<i>Install for AWS</i>
Azure	<i>Install for Azure</i>

Docker Daemon

	Minimum	Recommended
CPU Cores	8 CPU	16 CPU
Available RAM	64 GB RAM	128 GB RAM

Database client

Installation or upgrade of the product in a Dockerized environment requires installation of appropriate database client on the Trifacta node.

Database vendor	Description
PostgreSQL 12.3	The database client is included as part of the image and is automatically installed.
MySQL 5.7	<p>The database client must be downloaded and installed by the customer. It is not available in the Docker image. The database client must be referenced through the Docker image file.</p> <div> <p>NOTE: Before you perform an upgrade of your deployment that connects to a MySQL database, please contact <i>Alteryx Customer Success Services</i>.</p> </div>

Preparation

1. Review the *Browser Requirements* in the Planning Guide.

NOTE: Trifacta Self-Managed Enterprise Edition requires the installation of a supported browser on each desktop.

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:
 - a. `trifacta-docker-setup-bundle-x.y.z.tar`
 - b. `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
<code>docker-compose-local-postgres.yaml</code>	Runtime configuration file for the Docker image when PostgreSQL is to be running on the same machine. More information is provided below.
<code>docker-compose-local-mysql.yaml</code>	Runtime configuration file for the Docker image when MySQL is to be running on the same machine. More information is provided below.
<code>docker-compose-remote-db.yaml</code>	Runtime configuration file for the Docker image when the database is deployed on a remote server. <div>NOTE: You must manage this instance of the database.</div> More information is provided below.
<code>docker-compose-remote-db-postgres-s3.yaml</code>	Runtime configuration file for the Docker image when the Postgres database is deployed in AWS, and Trifacta is configured for S3 + EMR.
<code>docker-compose-remote-db-postgres-databricks-adls.yaml</code>	Runtime configuration file for the Docker image when the Postgres database is deployed in Azure, and Trifacta is configured for ADLS Gen2 + Databricks.
<code>README-running-trifacta-container-aws.md</code>	Instructions for running the Trifacta container on AWS

	NOTE: These instructions are referenced later in this workflow.
README-running-trifacta-container-azure.md	Instructions for running the Trifacta container on Azure 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.

- Load the Docker image into your local Docker environment:

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

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

```
docker images
```

- 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 version must be compatible with the following version(s) of Docker Compose.
- Docker Compose 1.24.1. It should be compatible with above version of Docker.

Build steps

- Acquire the RPM file from the FTP site:

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

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

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

- 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
<code>docker-compose-local-postgres.yaml</code>	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.
<code>docker-compose-local-mysql.yaml</code>	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.
<code>docker-compose-remote-db.yaml</code>	The Trifacta databases are to be installed on a remote server that you manage. <div>NOTE: Additional configuration is required.</div>
<code>docker-compose-remote-db-postgres-s3.yaml</code>	The Trifacta databases are to be installed on a remote Postgres server in AWS that you manage.
<code>docker-compose-remote-db-postgres-databricks-adls.yaml</code>	The Trifacta databases are to be installed on a remote Postgres server in Azure that you manage.

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
<code>image</code>	This reference must match the name of the image that you have acquired.
<code>container_name</code>	Name of container in your Docker environment.
<code>ports</code>	Defines the listening port for the Trifacta application. Default is 3005. <div>NOTE: If you must change the listening port, additional configuration is required after the image is deployed. See <i>Change Listening Port</i>.</div>

Database properties:

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

Property	Description
----------	-------------

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 <i>Change Database Port</i> in the Databases Guide.
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.
DB_AZURE_INSTANCE_NAME	Name of Azure Database for PostgreSQL Server. This setting is applicable only when the setup is on Azure with Databricks and ADLS Gen2.

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:

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

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: <div>/opt/trifacta/conf</div>
volumes.logs	Full path in container to the Trifacta logs directory. Default: <div>/opt/trifacta/logs</div>
volumes.license	Full path in container to the Trifacta license directory. Default: <div>/trifacta-license</div>

Setup Container

Steps:

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

```
docker-compose -f <docker-compose-filename>.yaml run --no-deps --rm trifacta initfiles
```

2. 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 <code>conf</code> and <code>logs</code> directories.
./trifacta-license	Place the <code>license.json</code> file in this directory.

3. Generate `.sql` file containing sql statements to create users and databases necessary for Trifacta services:

```
docker-compose -f <docker-compose-filename>.yaml run --no-deps --rm trifacta initdatabase
```

4. The following file is created on localhost:

Directory	Description
./trifacta-data/db_setup /trifacta_<database_type>_DB_objects.sql	Used by the Trifacta container to expose the <code>conf</code> and <code>logs</code> directories.

5. Create users and database:

- Postgres database:

```
docker-compose -f <docker-compose-filename>.yaml run postgresdb sh -c
"PGPASSWORD=<DB_ADMIN_PASSWORD> psql --username=<DB_ADMIN_USERNAME> --host=<DB_HOST_NAME> --
port=<DB_HOST_PORT> --dbname=postgres -f /opt/trifacta/db_setup/trifacta_<DB_TYPE>_DB_objects.
sql"
```

- MySQL database:

```
docker-compose -f <docker-compose-filename>.yaml run mysql db sh -c "mysql --host=<DB_HOST_NAME>
--port=<DB_HOST_PORT> --user=<DB_ADMIN_USERNAME> --password=<DB_ADMIN_PASSWORD> --
database=mysql < /opt/trifacta/db_setup/trifacta_mysql_<DB_TYPE>_objects.sql"
```

6. Run configuration and database migrations:

NOTE: During installation, the following command also creates required tables in the above databases.

```
docker-compose -f <docker-compose-filename>.yaml run --no-deps --rm trifacta run-migrations
```

7. Start Trifacta container:

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

NOTE: If the Trifacta container is running but nothing is listening at port 3005, please confirm that you have started the container using the appropriate `docker-compose` commands.

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
```

Additional setup for Azure

For more information on setup on Azure using ADLS Gen2 storage, see *ADLS Gen2 Access*.

Additional setup for Hadoop on-premises

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* in the Planning Guide.
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 **User menu > Admin console > Admin settings**. For more information, see *Admin Settings Page* in the Admin Guide.
3. In the Admin Settings page, you should do the following:
 - a. Configure password criteria. See *Configure Password Criteria*.
 - b. Change the Admin password. See *Change Admin Password*.
4. Workspace-level configuration can also be applied. From the left menu, select **User menu > Admin console > Workspace settings**. For more information, see *Workspace Settings 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.

Install on CentOS and RHEL

Contents:

- *Preparation*
 - *Required version of RPM for CentOS*
 - *Installation*
 - *Python setup tools*
 - *1. Install Dependencies*
 - *2. Install JDK*
 - *3. Install Trifacta package*
 - *4. Verify Install*
 - *5. Install License Key*
 - *6. Install Hadoop dependencies*
 - *7. Set File Ownership*
 - *8. Store install packages*
 - *Install Hadoop Dependencies*
 - *Included Dependencies*
 - *Acquire Other Dependencies*
 - *Install Dependencies*
 - *Next Steps*
 - *Install and configure Trifacta databases*
 - *Install configuration*
-

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

For more information on supported operating system versions, see *Product Support Matrix* in the Planning Guide.

Preparation

Before you install software, please review and verify 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. Review key sections of the Planning Guide:
 - a. Review the *System Requirements* and verify that all required components have been installed.
 - b. Verify that all required *System Ports* are opened on the node.
 - c. Review the *System Dependencies* in the Planning Guide.
 - d. **Cluster Configuration:** Additional steps are required to integrate the Trifacta platform with the cluster. See *Prepare Hadoop for Integration with the Platform* in the Planning Guide.
2. Acquire your *License Key*.
3. Install and verify operations of the datastore, if used.

NOTE: Access to the Spark cluster is required.

4. Verify access to the server where the Trifacta platform is to be installed.

Required version of RPM for CentOS

The installer for the Trifacta platform on CentOS/RHEL requires RPM version 4.11.3-40. Please upgrade if necessary.

NOTE: On CentOS/RHEL 7.4 or earlier, the installer may fail to launch on earlier versions of RPM.

Installation

Python setup tools

The Python setup tools can be useful for debugging startup issues.

Tip: These tools are useful. They are not required.

To install:

CentOS/RHEL 8.x:

```
yum install python3-setuptools
```

CentOS/RHEL 7.x:

```
yum install python-setuptools
```

1. Install Dependencies

Without Internet access

If you have not done so already, you may download the dependency bundle with your release directly from Alteryx . 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
```

Additional dependencies for CentOS 8.x

If you are installing on CentOS 8.x, you must complete the following manual dependency installs.

NodeJS:

```
yum -y --disablerepo="*" --enablerepo="trifacta_dependencies" install nodejs
```

PostgreSQL:

NOTE: This step is required only if you are installing the Trifacta platform onto CentOS 8.x and are using PostgreSQL to host the Trifacta databases. Otherwise, you may skip this step.

```
yum -y --disablerepo="*" --enablerepo="trifacta_dependencies" install postgresql96-server
```

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. For more information, see *System Requirements* in the Planning Guide.

Tip: 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.

NOTE: If this package is not included, the batch job runner service, which is required, fails to start.

Java 8:

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

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-openjdk.x86_64"  
},
```

3. Save changes.

5. Install License Key

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

6. Install Hadoop dependencies

If you are integrating with a supported Hadoop cluster, you must install the dependencies for the Hadoop cluster on the Trifacta node. See below.

7. Set File Ownership

All files in the Trifacta install directory and sub-directories must be owned by the same user that is used to run the Trifacta platform. Mismatches in ownership and execution permissions can cause services to fail to start.

Steps:

Before you upgrade, please complete the following:

1. Login to the Trifacta node as the root user.
2. Execute the following command. The user that is being granted ownership of the install directory is `trifacta`, which is the default user that runs the platform. If you are using a different user to run your Trifacta deployment, please substitute that name.


```
chown -R trifacta:trifacta /opt/trifacta
```

8. Store install packages

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

Install Hadoop Dependencies

If you are integrating Hadoop cluster, the associated Hadoop dependencies must be installed on the Trifacta® node.

Included Dependencies

The Hadoop dependencies for the latest supported version of each Hadoop distribution are included in the Trifacta software distribution.

Supported Versions:

- *Supported Deployment Scenarios for Cloudera*
- *Configure for EMR in the Configuration Guide*

Not required for:

NOTE: If you are integrating with one of the following running environments, please skip installing Hadoop dependencies.

Azure running environments:

- Azure Databricks

Acquire Other Dependencies

Hadoop dependencies for other versions of the Hadoop distribution can be acquired from the Trifacta FTP site using one of the following methods.

Via a web browser

1. Log in: <https://ftp.trifacta.com/login>
2. Browse to the following directory:

```
Releases/Trifacta_x.y/hadoop/
```

where: `x.y` corresponds to the release number that you are installing (e.g. Release 6.8).

3. Download the following file: `hadoop_deps.tar.gz`

Via WGET

Example is for Release 6.8:

```
wget --user CustomerUsername --ask-password https://ftp.trifacta.com/Releases/Trifacta_6.8/hadoop/hadoop_deps.tar.gz
```

Via SFTP

Example is for Release 6.8:

```
sftp CustomerUsername@ftp.trifacta.com:Releases/Trifacta_6.8/hadoop/hadoop-deps.tar.gz .
```

Via CURL

Example is for Release 6.8:

```
curl -O -C - -u CustomerUsername:CustomerPassword https://ftp.trifacta.com/Releases/Trifacta_6.8/hadoop/hadoop-deps.tar.gz
```

Via FTP/FTPS

1. Access the FTP server via your preferred FTP client.
2. Browse to the following directory:

```
Releases/Trifacta_x.y/hadoop/
```

where: `x.y` corresponds to the release number that you are installing (e.g. Release 6.8).

3. Download the following file: `hadoop_deps.tar.gz`

Install Dependencies

If needed, transfer the download to the Trifacta node.

Extract it to the following directory:

```
sudo tar -vxf hadoop-deps.tar --directory /opt/trifacta/
```

NOTE:

After you extract the files to the target directory, verify that the ownership of the new directory (`/opt/trifacta/hadoop-deps/`) and its subfolders match the ownership settings for the rest of the Trifacta installation in `/opt/trifacta`.

Next Steps

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* in the Databases Guide.

Install configuration

After installation is complete, additional configuration is required to make the platform operational. See *Install Configuration*.

Install on Ubuntu

Contents:

- *Preparation*
 - *Installation*
 - *1. Install Dependencies*
 - *2. Install JDK*
 - *3. Install Trifacta package*
 - *4. Verify Install*
 - *5. Install License Key*
 - *6. Install Hadoop dependencies*
 - *7. Set File Ownership*
 - *8. Store install packages*
 - *Install Hadoop Dependencies*
 - *Included Dependencies*
 - *Acquire Other Dependencies*
 - *Install Dependencies*
 - *Next Steps*
 - *Install and configure Trifacta databases*
 - *Install configuration*
-

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

For more information on supported operating system versions, see *Product Support Matrix* in the Planning Guide.

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. Review key sections of the Planning Guide:
 - a. Review the *System Requirements* and verify that all required components have been installed.
 - b. Verify that all required *System Ports* are opened on the node.
 - c. Review the *System Dependencies* in the Planning Guide.
 - d. **Cluster configuration:** Additional steps are required to integrate the Trifacta platform with the cluster. See *Prepare Hadoop for Integration with the Platform* in the Planning Guide.
2. Acquire your *License Key*.
3. Install and verify operations of the datastore, if used.

NOTE: Access to the cluster may be required.

4. Verify access to the server where the Trifacta platform is to be installed.

Installation

Tip: The Python setup tools can be useful for debugging startup issues. To install:

1. Install Dependencies

Without Internet access

If you have not done so already, you may download the dependency bundle with your release directly from Alteryx . 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
- Supervisor

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 nodeJS, nginx, and Supervisor:

- a. Ubuntu 16.04 (Xenial):

```
sudo apt-get install supervisor=3.2.4 nginx=1.17.7-1~xenial nodejs=14.15.4-1nodesource1
```

- b. Ubuntu 18.04 (Bionic Beaver):

```
sudo apt-get install supervisor=3.2.4 nginx=1.17.7-1~bionic nodejs=14.15.4-1nodesource1
```

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. For more information, see *System Requirements* in the Planning Guide.

Tip: 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:

NOTE: If you encounter errors running the following command, execute the next command anyway. If that command completes without error, the installation is ok.

```
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. Please note the Java version number (1.8) below, which can be modified for other supported versions of Java.

```
"env": {  
  "JAVA_HOME": "/usr/lib/jvm/java-1.8-openjdk.x86_64"  
},
```

3. Save changes.

5. Install License Key

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

6. Install Hadoop dependencies

If you are integrating with a supported Hadoop cluster, you must install the dependencies for the Hadoop cluster on the Trifacta node. See below.

7. Set File Ownership

All files in the Trifacta install directory and sub-directories must be owned by the same user that is used to run the Trifacta platform. Mismatches in ownership and execution permissions can cause services to fail to start.

Steps:

Before you upgrade, please complete the following:

1. Login to the Trifacta node as the root user.
2. Execute the following command. The user that is being granted ownership of the install directory is `trifacta`, which is the default user that runs the platform. If you are using a different user to run your Trifacta deployment, please substitute that name.

```
chown -R trifacta:trifacta /opt/trifacta
```

8. Store install packages

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

Install Hadoop Dependencies

If you are integrating Hadoop cluster, the associated Hadoop dependencies must be installed on the Trifacta® node.

Included Dependencies

The Hadoop dependencies for the latest supported version of each Hadoop distribution are included in the Trifacta software distribution.

Supported Versions:

- *Supported Deployment Scenarios for Cloudera*
- *Configure for EMR in the Configuration Guide*

Not required for:

NOTE: If you are integrating with one of the following running environments, please skip installing Hadoop dependencies.

Azure running environments:

- Azure Databricks

Acquire Other Dependencies

Hadoop dependencies for other versions of the Hadoop distribution can be acquired from the Trifacta FTP site using one of the following methods.

Via a web browser

1. Log in: <https://ftp.trifacta.com/login>
2. Browse to the following directory:

```
Releases/Trifacta_x.y/hadoop/
```

where: `x.y` corresponds to the release number that you are installing (e.g. Release 6.8).

3. Download the following file: `hadoop_deps.tar.gz`

Via WGET

Example is for Release 6.8:

```
wget --user CustomerUsername --ask-password https://ftp.trifacta.com/Releases/Trifacta_6.8/hadoop/hadoop-deps.tar.gz
```

Via SFTP

Example is for Release 6.8:

```
sftp CustomerUsername@ftp.trifacta.com:Releases/Trifacta_6.8/hadoop/hadoop-deps.tar.gz .
```

Via CURL

Example is for Release 6.8:

```
curl -O -C - -u CustomerUsername:CustomerPassword ftps://ftp.trifacta.com/Releases/Trifacta_6.8/hadoop/hadoop-deps.tar.gz
```

Via FTP/FTPS

1. Access the FTP server via your preferred FTP client.
2. Browse to the following directory:

```
Releases/Trifacta_x.y/hadoop/
```

where: `x.y` corresponds to the release number that you are installing (e.g. Release 6.8).

3. Download the following file: `hadoop-deps.tar.gz`

Install Dependencies

If needed, transfer the download to the Trifacta node.

Extract it to the following directory:

```
sudo tar -vxf hadoop-deps.tar --directory /opt/trifacta/
```

NOTE:

After you extract the files to the target directory, verify that the ownership of the new directory (`/opt/trifacta/hadoop-deps/`) and its subfolders match the ownership settings for the rest of the Trifacta installation in `/opt/trifacta`.

Next Steps

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* in the Databases Guide.

Install configuration

After installation is complete, additional configuration is required to make the platform operational. See *Install Configuration*.

License Key

Contents:

- *License limits*
 - *Download license key file*
 - *Acquire license key*
 - *Install your license key*
 - *Upload through the application*
 - *Command Line*
 - *Update your license key*
 - *Changing the license key location*
 - *License key violations*
 - *Too many users*
 - *License expiration date reached*
 - *Expired license*
 - *Invalid license key file*
-

Access to Trifacta is governed by a license key file that must be uploaded and installed on the Trifacta node.

License limits

Access to the product is determined by two factors:

- Number of users vs. number of users permitted by the license
- Expiration date of the license

How users are counted:

The number of users of the product is determined by:

- Number of active and disabled/suspended users
 - The default admin user account is counted as a valid user. Do not delete this account. For more information, see *Create Admin Account*.
- Deleted users may remain in the system for a period of time. These users are not counted against the license limit.

For more information, see "License key violations" below.

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 *Alteryx 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.

Upload through the application

Steps:

1. Navigate to the URL for the Trifacta application.
2. Login as an administrator.
3. From the menu, select **User menu > Admin console > Admin Settings**.
4. Scroll down to the bottom of the page. Click **Upload License**.
5. Navigate your local environment to select your license key file. Click **Open**.

The license key file is updated.

Command Line

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* in the Admin Guide.

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* in the Admin Guide.

License key violations

Too many users

If you have created more users in the Trifacta application than are supported by your license key, a notification banner is displayed.

NOTE: Although you are permitted to continue to use the application, you must remove users from your user base to maintain compliance with your license key. For more information on adjusting your license key, please contact *Alteryx Support*.

NOTE: Usage of the APIs is not blocked when user count limits are violated.

License expiration date reached

If you attempt to use Trifacta or its APIs after your license key has expired, access is prevented.

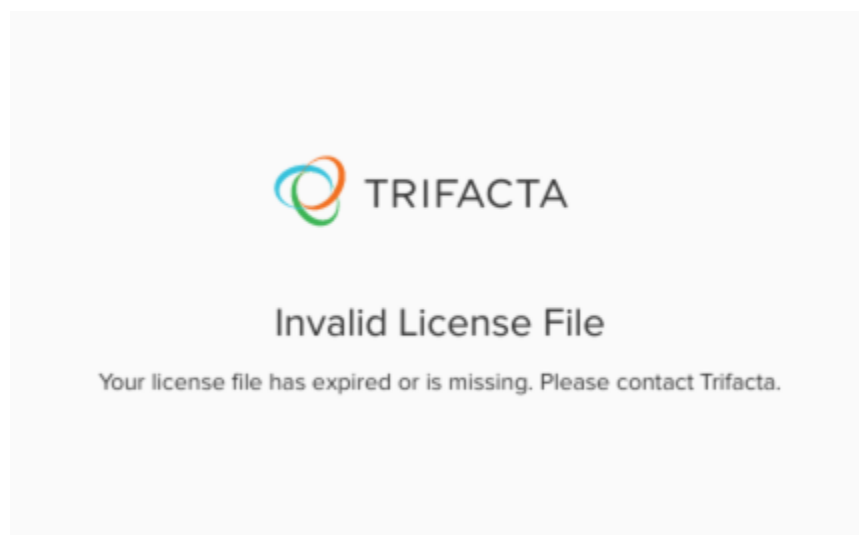
Please contact *Alteryx Support* to extend your license.

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 *Alteryx Support*.

Start and Stop the Platform

Contents:

- *Command Line*
 - *Start*
 - *Restart*
 - *Stop*
- *Configure Platform Restart*
- *Troubleshooting*
 - *Error - "ImportError: No module named pkg_resources" error in supervisord*
 - *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* in the Admin Guide.

Command Line

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
```

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

Restart

Command:

```
service trifacta restart
```

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

Stop

Command:

```
service trifacta stop
```

Configure Platform Restart

By default, the Trifacta platform waits for a period of time for the Trifacta application to restart before re-activating the user interface. As needed, you can review and modify the following settings, which define the parameters of these restarts.

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 parameters, and adjust settings as needed:

```
"webapp.waitForRestart.initialWait": 45000,  
"webapp.waitForRestart.intervalWait": 5000,  
"webapp.waitForRestart.maxChecks": 60,
```

Setting	Description
webapp.waitForRestart.initialWait	Number of seconds to wait for the Trifacta application before checking it for a successful restart. Default is 45 000 milliseconds (45 seconds).
webapp.waitForRestart.intervalWait	After the initial wait period has failed, this value is the number of seconds to wait before checking the Trifacta application for a successful restart. Default is 5 000 milliseconds (5 seconds).
webapp.waitForRestart.maxChecks	Total number of checks for a successful restart before failing the Trifacta application.

3. Save your changes and restart the application.

Troubleshooting

You can verify operations of WebHDFS. Command:

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

Error - "ImportError: No module named pkg_resources" error in supervisord

When you start the platform for the first time, you may receive the following error:

```
Traceback (most recent call last):
File "/usr/local/bin/supervisord", line 5, in <module>
from pkg_resources import load_entry_point
ImportError: No module named pkg_resources
```

This error occurs when the supervisord process is starting. The Trifacta platform fails to complete startup.

Solution:

This issue is caused by a missing package for supervisord. The simplest solution is to install the Python setup tools on the Trifacta node. Commands are listed below.

NOTE: These commands must be executed as root user.

CentOS/RHEL:

```
yum install python-setuptools
```

Ubuntu:

```
wget https://bootstrap.pypa.io/ez_setup.py -O - | python
```

After installation is complete, restart the platform.

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 12 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-12 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* in the Admin Guide.

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**.

NOTE: If you enter a mismatched password and password confirmation, registration fails as expected. If you correct the mismatch and try to register again, registration may fail again. The workaround is to clear your browser cache and register again. This is a known issue.

- If self-registration is enabled, 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* in the Configuration Guide.

After you login, you are placed in the Home page. See *Home Page* in the User Guide.

Tip: When you login for the first time, you can immediately import a dataset to begin transforming it.

- 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 **User menu > Preferences > Storage** and then select the AWS option. See *Configure Your Access to S3* in the Configuration Guide.
- For a basic walkthrough of the Trifacta application, see *Workflow Basics* in the User Guide.

You cannot login to the application using an unsupported browser version. For more information on supported versions, see *Browser Requirements*.

Product Documentation:

NOTE: After you log in the Trifacta application, you can access online documentation for your product. Select **Help menu > Documentation**.

To log out:

From the User menu, select **Log out**.

Install Configuration

After the Trifacta® platform has been installed, you must complete the configuration steps listed here before starting the platform. Please complete the steps that apply to your deployment of the Trifacta node.

If you are using a PDF version of this content, please complete the following configuration with the Configuration Guide PDF reference available.

Configuration Interfaces

To review the interfaces for configuration that are available through the Trifacta application, please complete the following steps.

Steps:

1. Login to the application. See *Login*.
2. The primary configuration interface is the Admin Settings page. From the left menu, select **User menu > Admin console > Admin settings**. For more information, see *Admin Settings Page* in the Admin Guide.

NOTE: In the Admin Settings page, you should do the following configuration steps:

- a. Configure password criteria. See *Configure Password Criteria* in the Admin Guide.
- b. Change the Admin password. See *Change Admin Password* in the Configuration Guide.

3. Workspace-level configuration can also be applied. From the left menu, select **User menu > Admin console > Workspace settings**. For more information, see *Workspace Settings Page* in the Admin Guide.

Containerized Installs

This section does not apply if you are using a containerized version of the software. Containerized versions:

- *Install for Docker*

Install Config for Azure

Contents:

- *Configure in Azure*
 - *Create registered application*
 - *Create Key Vault in Azure*
 - *Enable Key Vault access for the Trifacta platform*
 - *Create or modify Azure backend datastore*
 - *Create or modify running environment cluster*
 - *Configure the Platform*
 - *Configure the Platform for Azure*
 - *Base platform configuration*
 - *Set base storage layer*
 - *Integrate with running environment*
 - *Configure platform authentication*
 - *Verify Operations*
 - *Documentation*
 - *Next Steps*
-

After installation of the Trifacta platform software and databases in your Microsoft Azure infrastructure, please complete these steps to perform the basic integration between the Trifacta node and Azure resources like the backend storage layer and running environment cluster.

NOTE: This section includes only basic configuration for required platform functions and integrations with Azure. Please use the links in this section to access additional details on these key features.

Tip: When you save changes from within the Trifacta platform, your configuration is automatically validated, and the platform is automatically restarted.

Configure in Azure

These steps require admin access to your Azure deployment.

Create registered application

To create an Azure Active Directory (AAD) application, please complete the following steps in the Azure console.

Steps:

1. Create registered application:
 - a. In the Azure console, navigate to **Azure Active Directory > App Registrations**.
 - b. Create a New App. Name it `trifacta`.

NOTE: Retain the Application ID and Directory ID for configuration in the Trifacta platform.

2. Create a client secret:
 - a. Navigate to **Certificates & secrets**.
 - b. Create a new Client secret.
-

NOTE: Retain the value of the Client secret for configuration in the Trifacta platform.

3. Add API permissions:
 - a. Navigate to **API Permissions**.
 - b. Add Azure Key Vault with the `user_impersonation` permission.

For additional details, see *Configure for Azure*.

Please complete the following steps in the Azure portal to create a Key Vault and to associate it with the Trifacta registered application.

NOTE: A Key Vault is required for use with the Trifacta platform.

Create Key Vault in Azure

Steps:

1. Log into the Azure portal.
2. Goto: <https://portal.azure.com/#create/Microsoft.KeyVault>
3. Complete the form for creating a new Key Vault resource:
 - a. Name: Provide a reasonable name for the resource. Example:

```
<clusterName>-<applicationName>-<group/organizationName>
```

- Or, you can use `trifacta`.
- b. Location: Pick the location used by the cluster.
 - c. For other fields, add appropriate information based on your enterprise's preferences.
 4. To create the resource, click **Create**.

NOTE: Retain the DNS Name value for later use.

Enable Key Vault access for the Trifacta platform

Steps:

In the Azure portal, you must assign access policies for application principal of the Trifacta registered application to access the Key Vault.

Steps:

1. In the Azure portal, select the Key Vault you created. Then, select **Access Policies**.
2. In the Access Policies window, select the Trifacta registered application.
3. Click **Add Access Policy**.
4. Select the following secret permissions (at a minimum):
 - a. Get
 - b. Set
 - c. Delete
 - d. Recover
5. Select the Trifacta application principal.
6. Assign the policy you just created to that principal.

For additional details, see *Configure Azure Key Vault*.

Create or modify Azure backend datastore

In the Azure console, you must create or modify the backend datastore for use with the Trifacta platform. Supported datastores:

NOTE: You should review the limitations for your selected datastore before configuring the platform to use it. After the base storage layer has been defined in the platform, it cannot be modified.

Datastore	Notes
ADLS Gen2	Supported for use with Azure Databricks cluster only. See <i>ADLS Gen2 Access</i> .
ADLS Gen1	See <i>ADLS Gen1 Access</i> .
WASB	Only WASBS protocol is supported only. See <i>WASB Access</i> .

Create or modify running environment cluster

In the Azure console, you must create or modify the running environment where jobs are executed by the Trifacta platform. Supported running environments:

NOTE: You should review the limitations for your selected running environment before configuring the platform to use it.

Running Environment	Notes
Azure Databricks	See <i>Configure for Azure Databricks</i> .

Configure the Platform

Please complete the following sections as soon as you can access the Trifacta application .

This section contains a set of configuration steps required to enable basic functionality in the Trifacta® platform, as well as the methods by which you can apply the configuration.

Configure the Platform for Azure

Please complete the following steps to configure the Trifacta platform and to integrate it with Azure resources.

Base platform configuration

Please complete the following configuration steps in the Trifacta® platform.

NOTE: If you are integrating with Azure Databricks and are Managed Identities for authentication, please skip this section. That configuration is covered in a later step.

NOTE: Except as noted, these configuration steps are required for all Azure installs. These values must be extracted from the Azure portal.

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. Azure registered application values:

```
"azure.applicationId": "<azure_application_id>",  
"azure.directoryId": "<azure_directory_id>",  
"azure.secret": "<azure_secret>",
```

Parameter	Description
azure.applicationId	Application ID for the Trifacta registered application that you created in the Azure console
azure.directoryId	The directory ID for the Trifacta registered application
azure.secret	The Secret value for the Trifacta registered application

3. Configure Key Vault:

```
"azure.keyVaultUrl": "<url_of_key_vault>",
```

Parameter	Description
azure.keyVaultUrl	URL of the Azure Key Vault that you created in the Azure console

4. Save your changes and restart the platform.

For additional details:

- See *Configure for Azure*.
- See *Configure Azure Key Vault*.

Set base storage layer

The Trifacta platform supports integration with the following backend datastores on Azure.

- ADLS Gen2
- ADLS Gen1
- WASB

ADLS Gen2

Please complete the following configuration steps in the Trifacta® platform.

NOTE: Integration with ADLS Gen2 is supported only on Azure Databricks.

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 ADLS Gen2 as the base storage layer:

```
"webapp.storageProtocol": "abfss",  
"hdfs.enabled": false,  
"hdfs.protocolOverride": "",
```

Parameter	Description
webapp.storageProtocol	Sets the base storage layer for the platform. Set this value to <code>abfss</code> . NOTE: After this parameter has been saved, you cannot modify it. You must re-install the platform to change it.
hdfs.enabled	For ADLS Gen2 access, set this value to <code>false</code> .
hdfs.protocolOverride	For ADLS Gen2 access, this special parameter should be empty. It is ignored when the storage protocol is set to <code>abfss</code> .

3. Configure ADLS Gen2 access mode. The following parameter must be set to `system`.

```
"azure.adlsgen2.mode": "system",
```

4. Set the protocol whitelist and base URIs for ADLS Gen2:

```
"fileStorage.whitelist": ["abfss"],  
"fileStorage.defaultBaseUri": ["abfss://filesystem@storageaccount.dfs.core.windows.net/"],
```

Parameter	Description
fileStorage.whitelist	A comma-separated list of protocols that are permitted to read and write with ADLS Gen2 storage. NOTE: The protocol identifier <code>"abfss"</code> must be included in this list.
fileStorage.defaultBaseUri	For each supported protocol, this param must contain a top-level path to the location where Trifacta platform files can be stored. These files include uploads, samples, and temporary storage used during job execution. NOTE: A separate base URI is required for each supported protocol. You may only have one base URI for each protocol.

5. Save your changes.

6. The Java VFS service must be enabled for ADLS Gen2 access. For more information, see *Configure Java VFS Service* in the Configuration Guide.

For additional details, see *ADLS Gen2 Access*.

ADLS Gen1

ADLS Gen1 access leverages HDFS protocol and storage, so additional configuration is required.

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 ADLS Gen1 as the base storage layer:

```
"webapp.storageProtocol": "adl",  
"hdfs.enabled": false,
```

Parameter	Description
webapp.storageProtocol	Sets the base storage layer for the platform. Set this value to <code>adl</code> . NOTE: After this parameter has been saved, you cannot modify it. You must re-install the platform to change it.
hdfs.enabled	For ADLS Gen1 storage, set this value to <code>false</code> .

3. These parameters specify the base location and protocol for storage. Only one datastore can be specified:

```
"fileStorage": {  
  "defaultBaseUri": [  
    "<baseURIOfYourLocation>"  
  ],  
  "whitelist": ["adl"]  
}
```

Parameter	Description
filestorage.defaultBaseURIs	Set this value to the base location for your ADLS Gen1 storage area. Example: <code>adl://<YOUR_STORE_NAME>.azuredatalakestore.net</code>
whitelist	This list must include <code>adl</code> .

4. Save your changes.

For additional details, see *ADLS Gen1 Access*.

WASB

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 WASB as the base storage layer:

```
"webapp.storageProtocol": "wasbs",  
"hdfs.enabled": false,
```

Parameter	Description
webapp.storageProtocol	Sets the base storage layer for the platform. Set this value to <code>wasbs</code> . NOTE: After this parameter has been saved, you cannot modify it. You must re-install the platform to change it. wasb protocol is not supported.

hdfs.enabled	For WASB blob storage, set this value to <code>false</code> .
--------------	---

3. Save your changes.
4. In the following sections, you configure where the platform acquires the SAS token to use for WASB access from one of the following:
 - a. From platform configuration
 - b. From the Azure key vault

Configure SAS token for WASB

When integrating with WASB, the platform must be configured to use a SAS token to gain access to WASB resources. This token can be made available in either of the following ways, each of which requires separate configuration.

Via Trifacta platform configuration:

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 and specify the following parameter:

```
"azure.wasb.fetchSasTokensFromKeyVault": false,
```

Parameter	Description
azure.wasb.fetchSasTokensFromKeyVault	For acquiring the SAS token from platform configuration, set this value to <code>false</code> .

3. Save your changes and restart the platform.

Via Azure Key Vault:

To require the Trifacta platform to acquire the SAS token from the Azure key vault, please complete the following configuration 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 and specify the following parameter:

```
"azure.wasb.fetchSasTokensFromKeyVault": true,
```

Parameter	Description
azure.wasb.fetchSasTokensFromKeyVault	For acquiring the SAS token from the key vault, set this value to <code>true</code> .

Define WASB stores

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. Locate the `azure.wasb.stores` configuration block.
3. Apply the appropriate configuration as specified below.

Tip: The default container must be specified as the first set of elements in the array. All containers listed after the first one are treated as extra stores.

```
"azure.wasb.stores":
[
{
```

```

    "sasToken": "<DEFAULT_VALUE1_HERE>",
    "keyVaultSasTokenSecretName": "<DEFAULT_VALUE1_HERE>",
    "container": "<DEFAULT_VALUE1_HERE>",
    "blobHost": "<DEFAULT_VALUE1_HERE>"
  },
  {
    "sasToken": "<VALUE2_HERE>",
    "keyVaultSasTokenSecretName": "<VALUE2_HERE>",
    "container": "<VALUE2_HERE>",
    "blobHost": "<VALUE2_HERE>"
  }
]
},

```

Parameter	Description	SAS Token from Azure Key Vault	SAS Token from Platform Configuration
sasToken	<p>Set this value to the SAS token to use, if applicable.</p> <p>Example value:</p> <pre>?sv=2019-02-02&ss=bfqt&srt=sco&sp=rwdlacup&se=2022-02-13T00:00:00Z&st=2020-02-13T00:00:00Z&spr=https&sig=<redacted></pre>	<p>Set this value to an empty string.</p> <p>NOTE: Do not delete the entire line. Leave the value as empty.</p>	<p>See below for the command to execute to generate a SAS token.</p>
keyVaultSasTokenSecretName	<p>Set this value to the secret name of the SAS token in the Azure key vault to use for the specified blob host and container.</p> <p>If needed, you can generate and apply a per-container SAS token for use in this field for this specific store. Details are below.</p>		<p>Set this value to an empty string.</p> <p>NOTE: Do not delete the entire line. Leave the value as empty.</p>
container	<p>Apply the name of the WASB container.</p> <p>NOTE: If you are specifying different blob host and container combinations for your extra stores, you must create a new Key Vault store. See above for details.</p>		
blobHost	<p>Specify the blob host of the container.</p> <p>Example value:</p> <pre>storage-account.blob.core.windows.net</pre> <p>NOTE: If you are specifying different blob host and container combinations for your extra stores, you must create a new Key Vault store. See above for details.</p>		

4. Save your changes and restart the platform.

For additional details, see [WASB Access](#).

Checkpoint: At this point, you should be able to load data from your backend datastore, if data is available. You can try to run a small job on Photon, which is native to the Trifacta node. You cannot yet run jobs on an integrated cluster.

Integrate with running environment

The Trifacta platform can run jobs on the following running environments.

NOTE: You may integrate with only one of these environments.

Base configuration for Azure running environments

The following parameters should be configured for all Azure running environments.

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. Parameters:

```
"webapp.runInTrifactaServer": true,  
"webapp.runinEMR": false,  
"webapp.runInDataflow": false,
```

Parameter	Description
webapp.runInTrifactaServer	When set to <code>true</code> , the platform recommends and can run smaller jobs on the Trifacta node, which uses the embedded Photon running environment. <div>Tip: Unless otherwise instructed, the Photon running environment should be enabled.</div>
webapp.runinEMR	For Azure, set this value to <code>false</code> .
webapp.runInDataflow	For Azure, set this value to <code>false</code> .

3. Save your changes.

Integrate with Azure Databricks

The Trifacta platform can be configured to integrate with supported versions of Azure Databricks clusters to run jobs in Spark.

NOTE: Before you attempt to integrate, you should review the limitations around this integration. For more information, see *Configure for Azure Databricks*.

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. Configure the following parameters to enable job execution on the specified Azure Databricks cluster:

```
"webapp.runInDatabricks": true,  
"webapp.runWithSparkSubmit": false,
```

Parameter	Description
webapp.runInDatabricks	Defines if the platform runs jobs in Azure Databricks. Set this value to <code>true</code> .
webapp.runWithSparkSubmit	For all Azure Databricks deployments, this value should be set to <code>false</code> .

3. Configure the following Azure Databricks-specific parameters:

```
"databricks.serviceUrl": "<url_to_databricks_service>",
```

Parameter	Description
databricks.serviceUrl	URL to the Azure Databricks Service where Spark jobs will be run (Example: https://westus2.azuredatabricks.net)

NOTE: If you are using instance pooling on the cluster, additional configuration is required. See *Configure for Azure Databricks*.

4. Save your changes and restart the platform.

For additional details, see *Configure for Azure Databricks*.

Checkpoint: At this point, you should be able to load data from your backend datastore and run jobs on an integrated cluster.

Configure platform authentication

The Trifacta platform supports the following methods of authentication when hosted in Azure.

Integrate with Azure AD SSO

The platform can be configured to integrate with your enterprise's Azure Active Directory provider. For more information, see *Configure SSO for Azure AD*.

Non-SSO authentication

If you are not applying your enterprise SSO authentication to the Trifacta platform, platform users must be created and managed through the application.

Self-managed:

Users can be permitted to self-register their accounts and manage their password reset requests:

NOTE: Self-created accounts are permitted to import data, generate samples, run jobs, and generate and download results. Admin roles must be assigned manually through the application.

- See *Configure User Self-Registration* in the Configuration Guide
- See *Enable Self-Service Password Reset* in the Configuration Guide

Admin-managed:

If users are not permitted to create their accounts, an admin must do so:

- See *Create User Account* in the Admin Guide
 - See *Create Admin Account* in the Admin Guide

Checkpoint: Users who are authenticated or have been provisioned user accounts should be able to login to the Trifacta application and begin using the product.

Verify Operations

NOTE: You can try to verify operations using the Trifacta Photon running environment at this time.

After you have applied a configuration change to the platform and restarted, you can use the following steps to verify that Trifacta® is working correctly.

Documentation

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

Next Steps

The following install and configuration topics were not covered in this workflow. If these features apply, please reference the following topics in the Configuration Guide for more information.

Topic	Description	Configuration Guide sections
User Access	You can enable self-service user registration or create users through the admin account.	<i>Required Platform Configuration</i>
Relational Connections	The platform can integrate with a variety of relational datastores.	<i>Create Encryption Key File</i> <i>Relational Access</i>
Compressed Clusters	The platform can integrate with some compressed running environments.	<i>Enable Integration with Compressed Clusters</i>
High Availability	The platform can integrate with a highly available cluster.	<i>Enable Integration with Cluster High Availability</i>
	The Trifacta node can be configured to use other nodes in case of a failure.	<i>Configure for High Availability</i>
Features	Some features must be enabled and can be configured through platform configuration.	<i>Configure Features</i> Feature flags: <i>Miscellaneous Configuration</i>
Services	Some platform services support additional configuration options.	<i>Configure Services</i>

Install Reference

These appendices provide additional information during installation of Trifacta®.

Install SSL Certificate

Contents:

- *Prerequisites*
- *Configure nginx*
- *Modify listening port for Trifacta platform*
- *Add secure HTTP headers*
- *Enable secure cookies*
- *Disable default port*
- *Update certificates*
- *Troubleshooting*

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

NOTE: For security reasons, a self-signed certificate is considered an insecure origin by most browsers, which do not cache results from these origins. As a result, each user visit to the Trifacta application requires re-downloading of static assets from the application. For better performance, you should consider deploying a signed certificate. Low-cost solutions such as Let's Encrypt can be deployed to manage your certificates on the Trifacta node.

Prerequisites

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.

NOTE: Do not apply these configuration changes to the nginx files in `/opt/trifacta/conf`. Those files apply to the internal nginx server, which is not covered by SSL.

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.
 - a. The .key file should go into the /etc/ssl/private/ directory.
 - b. 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.

- c. 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      TLSv1.2 TLSv1.3;
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           $http_upgrade;
    proxy_set_header    Connection       "upgrade";
    proxy_redirect      off;
}
proxy_connect_timeout 6000;
proxy_send_timeout    6000;
proxy_read_timeout    6000;
send_timeout          6000;
}
server {
    listen      80;
    return 301 https://$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	<code>proxy.securityHeaders.enabled=true</code>
X-Frame-Options	HTTP and HTTPS	<code>proxy.securityHeaders.enabled=true</code>
Strict-Transport-Security	HTTPS	<code>proxy.securityHeaders.enabled=true</code> and <code>proxy.securityHeaders.httpsHeaders=true</code>

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.

Disable default port

If you wish to access through the default port (3005), you must do so external to the platform and through the node itself.

NOTE: The Trifacta platform requires access to the default port internally. You cannot disable external access to this port through the platform. You must disable through the operating system.

For more information, please see the documentation provided with your operating system distribution.

Update certificates

To replace a certificate with an updated one, please do the following.

Steps:

1. Copy in the new certificate to the Trifacta node.
2. Edit the nginx configuration file:

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

3. In the configuration file, replace the values for the following settings to point to the new certificate:
 - a. `ssl_certificate`
 - b. `ssl_certificate_key`
 - c. For more information, see "Configure nginx" above.
4. Save the file, 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 Azure

Contents:

- *Azure Deployment Scenarios*
- *Azure Installations*
- *Azure Integrations*

Azure Deployment Scenarios

The following are the Azure deployment scenarios.

Deployment Scenario	Trifacta node installation	Base Storage Layer	Storage - WASB	Storage - ADLS Gen1 /Gen2	Storage - Azure SQL Database	Storage - SQL DW	Cluster
Trifacta® Self-Managed Enterprise Edition install for WASB	Azure	WASB	read/write	read only	read	read/write	Azure Databricks
Trifacta Self-Managed Enterprise Edition install for ADLS Gen1	Azure	HDFS	read only	read/write	read	read/write	Azure Databricks
Trifacta Self-Managed Enterprise Edition install for ADLS Gen2	Azure	ABFSS	read only	<ul style="list-style-type: none">• read/write to ADLS Gen2• read only to ADLS Gen1	read	read/write	Azure Databricks

Legend and Notes:

Column	Notes
Deployment Scenario	Description of the Azure-connected deployment
Trifacta node installation	Location where the Trifacta node is installed in this scenario.
Base Storage Layer	When the Trifacta platform is first installed, the base storage layer must be set. <div>NOTE: After you have begun using the product, you cannot change the base storage layer.</div>
Storage - WASB	For read/write access to WASB, the base storage layer must be set to WASB.
Storage - ADLS Gen1	For read/write access to ADLS Gen1, the base storage layer must be set to HDFS.
Storage - ADLS Gen2	For read/write access to ADLS Gen2, the base storage layer must be set to ABFSS.
Storage - Azure SQL Database	For Azure installs, you can optionally create a connection to an Azure SQL Database instance.
Storage - SQL DW	For Azure installs, you can optionally create a connection to an Azure-hosted instance of SQL DW.
Cluster	List of Hadoop cluster types that are supported for integration and job execution at scale. <ul style="list-style-type: none">• The Trifacta platform can integrate with at most one cluster. It cannot integrate with two different clusters at the same time.

- Smaller jobs can be executed on the Trifacta Photon running environment, which is hosted on the Trifacta node itself.
- For more information, see *Running Environment Options* in the Configuration Guide.

Azure Installations

For more information, see *Install for Azure* in the Install Guide.

Azure Integrations

The following table describes the different Azure components that can host or integrate with the Trifacta platform. Combinations of one or more of these items constitute one of the deployment scenarios listed in the following section.

Azure Service	Description	Base Storage Layer	Other Required Azure Services
Azure Databricks	Optionally, you can integrate the Trifacta platform with an Azure Databricks cluster.	Base storage layer can be HDFS (for ADLS Gen1), ABFSS (for ADLS Gen2), or WASB.	
WASB	Windows Azure Storage Blobs (WASB) extends HDFS to enable access to storage blobs that have not been deployed into the cluster.	Base Storage Layer = WASB	Azure Databricks Key Vault
ADLS Gen1	Active Data Lake Store (ADLS) provides a highly scalable file-based storage system within the cluster.	Base Storage Layer = HDFS	Azure Databricks Key Vault
ADLS Gen2	Azure Data Lake Storage Gen2 is a set of capabilities dedicated to big data analytics, built on Azure Blob storage.	Base Storage Layer = ABFSS	Azure Databricks Key Vault

The following database connections are optional.

Database Name	Description
Hive	<div> NOTE: Access to Hive is not supported on Azure Databricks. </div>
SQL DW	You can read from and write to SQL Data Warehouse, a scalable data warehouse solution for Azure.
Azure SQL Database	You can read from Azure SQL Database, a SQL Server variant for Azure.

Uninstall

To remove Trifacta®, 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 Trifacta database.

CentOS/RHEL:

```
sudo rpm -e trifacta
```

Ubuntu:

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



Copyright © 2022 - Trifacta, Inc.
All rights reserved.