



# TRIFACTA

## Install Guide for AWS

Version: 8.7.1  
Doc Build Date: 09/01/2022

**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. *Install* 4
  - 1.1 *Install Overview* 5
    - 1.1.1 *Install for AWS* 7
    - 1.1.2 *Install for High Availability on AWS* 12
  - 1.2 *Install Software* 23
    - 1.2.1 *Install Dependencies without Internet Access* 24
    - 1.2.2 *Install for Docker* 28
    - 1.2.3 *Install on CentOS and RHEL* 39
    - 1.2.4 *Install on Ubuntu* 46
    - 1.2.5 *License Key* 52
  - 1.3 *Start and Stop the Platform* 55
  - 1.4 *Login* 58
  - 1.5 *Install Reference* 59
    - 1.5.1 *Install SSL Certificate* 60
    - 1.5.2 *Change Listening Port* 65
    - 1.5.3 *Supported Deployment Scenarios for AWS* 66
    - 1.5.4 *Uninstall* 70

# Install

This section contains content related to the installation and configuration of Trifacta® products.

# 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 <div><b>Tip:</b> 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.</div>
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 <div><b>NOTE:</b> There are separate Install Guides for each supported infrastructure. Please verify that you are using the appropriate one.</div>
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 AWS

## Contents:

- *Scenario Description*
  - *Limitations*
    - *Deployment Limitations*
    - *Product Limitations*
  - *Prerequisites*
    - *AWS desktop requirements*
    - *AWS prerequisites*
  - *Preparation*
    - *AWS Information*
    - *Internet access*
  - *Deploy the Cluster*
  - *Deploy the EC2 Node*
  - *Install Workflow*
  - *Next Steps*
- 

This install process applies to installing Trifacta® Self-Managed Enterprise Edition on an AWS infrastructure that you manage.

## AWS Marketplace deployments:

**NOTE:** Content in this section does not apply to deployments from the AWS Marketplace. For more information on installing from the Marketplace, see the AWS Marketplace listing.

## High availability deployments on AWS:

Installation on AWS for high availability utilizes an image-based deployment method. These instructions do not apply.

For more information, see *Install for High Availability on AWS*.

## Scenario Description

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

- Installation of Trifacta on an EC2 server in AWS
- Installation of Trifacta databases on AWS
- Integration with a supported EMR cluster.
- Base storage layer and backend datastore of S3

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

## Limitations

### Deployment Limitations

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

- When publishing single files to S3, you cannot apply an `append` publishing action.
- The following limitations apply to EMR integration only:
  - No support for Hive integration
  - No support for secure impersonation or Kerberos

## Product Limitations

For general limitations of Trifacta, see *Product Limitations* in the Planning Guide.

## 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*.

## AWS desktop requirements

- All desktop users must be able to connect to the EC2 instance through the enterprise infrastructure.

## AWS prerequisites

Depending on which of the following AWS components you are deploying, additional prerequisites and limitations may apply. Please review these sections as well.

- *Configure for EMR* in the Configuration Guide
- *S3 Access* in the Configuration Guide
- *Amazon Redshift Connections* in the Configuration Guide

## Preparation

Before you install Trifacta on AWS, please verify that you have completed the following:

1. **Read:** Please read this entire document before you begin.
2. **VPC:** Enable and deploy a working AWS VPC.
  - a. In your VPC, you must define a subnet where you plan to deploy the Trifacta node.
3. **S3:** Enable and deploy an AWS S3 bucket to use as the base storage layer for the platform. In the bucket, the platform stores metadata in the following location:

```
<S3_bucket_name>/trifacta
```

See <https://s3.console.aws.amazon.com/s3/home>.

4. **IAM Policies:** Create IAM policies for access to the S3 bucket. Required permissions are the following:
  - The system account or individual user accounts must have full permissions for the S3 bucket:

```
Delete*, Get*, List*, Put*, Replicate*, Restore*
```

- These policies must apply to the bucket and its contents. Example:

```
"arn:aws:s3:::my-trifacta-bucket-name"
"arn:aws:s3:::my-trifacta-bucket-name/*"
```



- See <https://console.aws.amazon.com/iam/home#/policies>
5. **EC2 instance role:** Create an EC2 instance role for your S3 bucket policy. See <https://console.aws.amazon.com/iam/home#/roles>.
  6. **EC2 instance:** Deploy an AWS EC2 with SELinux where the Trifacta software can be installed.
    - 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.

**NOTE:** The EC2 node must meet the system requirements for installing the platform. For more information, see *System Requirements* in the Planning Guide.

7. **EMR cluster:** An existing EMR cluster is required.
  - a. **Cluster sizing:** Before you begin, you should allocate sufficient resources for sizing the cluster. For guidance, please contact your Trifacta representative.
  - b. See Deploy the Cluster below.
8. **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.

**NOTE:** If installing databases on Amazon RDS, an admin account to RDS is required. For more information, see *Install Databases on Amazon RDS*.

## AWS Information

Before you begin installation, please acquire the following information from AWS:

- **EMR:**
  - AWS region for the EMR cluster, if it exists.
  - ID for EMR cluster, if it exists
    - If you are creating an EMR cluster as part of this process, please retain the ID.
    - The EMR cluster must allow access from the Trifacta node. This configuration is described later.
- **Subnet:** Subnet within your virtual private cloud (VPC) where you want to launch the Trifacta platform.
  - This subnet should be in the same VPC as the EMR cluster.
  - Subnet can be private or public.
  - If it is private and it cannot access the Internet, additional configuration is required. See below.
- **S3:**
  - Name of the S3 bucket that the platform can use
  - Path to resources on the S3 bucket
- **EC2:**
  - Instance type for the Trifacta node

## Internet access

From AWS, the Trifacta platform requires Internet access for the following services:

**NOTE:** Depending on your AWS deployment, some of these services may not be required.

- AWS S3
- Key Management System [KMS] (if sse-kms server side encryption is enabled)
- Secure Token Service [STS] (if temporary credential provider is used)
- EMR (if integration with EMR cluster is enabled)

**NOTE:** If the Trifacta platform is hosted in a VPC where Internet access is restricted, access to S3, KMS and STS services must be provided by creating a VPC endpoint. If the platform is accessing an EMR cluster, a proxy server can be configured to provide access to the AWS ElasticMapReduce regional endpoint.

## Deploy the Cluster

In your AWS infrastructure, you must deploy a supported version of EMR across a recommended number of nodes to support the expected data volumes of your Trifacta jobs.

- For more information on the supported EMR distributions, see *Supported Deployment Scenarios for AWS*.
- For more information on suggested sizing, see *Sizing Guidelines* in the Planning Guide.

**NOTE:** Cluster information including cluster configuration files must be accessible to the Trifacta node. These requirements are described below.

## Deploy the EC2 Node

An EC2 node of the cluster must be deployed to host the Trifacta platform software. Here are some guidelines for deploying the EC2 cluster from the EC2 cluster:

1. **Instance size:** Select the instance size.
2. **Network:** Configure the VPC, subnet, firewall and other configuration settings necessary to communicate with the instance.
3. **Auto-assigned Public IP:** You must create a public IP to access the Trifacta platform.
4. **EC2 role:** Select the EC2 role that you created.
5. **Local storage:** Select a local EBS volume. The default volume includes 100GB storage.

**NOTE:** The local storage environment contains the Trifacta databases, the product installation, and its log files. No source data is ever stored within the product.

6. **Security group:** Use a security group that exposes access to port 3005, which is the default port for the platform.
7. **Create an AWS key-pair for access:** This key is used to provide SSH access to the platform, which may be required for some admin tasks.
8. Save your changes.

## 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 for High Availability on AWS

## Contents:

- *Limitations*
  - *Prerequisites*
    - *AWS infrastructure*
    - *Trifacta assets*
  - *Install Steps*
    - *Configure Docker image*
    - *Configure AWS Kubernetes*
    - *Configure DB credential secrets*
    - *Configure the deployment*
    - *Edit values overrides*
    - *Install*
  - *Other Commands*
    - *Scale platform*
    - *Restart platform*
    - *Delete pods*
  - *Backups*
    - *Database*
    - *EFS mounts*
  - *Configuration*
    - *Set S3 as base storage layer*
    - *Upload the license file*
    - *Configure for EMR*
- 

In the Amazon AWS infrastructure, the Trifacta® platform can be deployed in a high availability failover mode across multiple nodes. This section describes the process for installing the platform across multiple, highly available nodes.

**NOTE:** This section applies to customer-managed deployments of the Trifacta platform on AWS.

- **On-Premises for Hadoop:** For more information, see *Install for High Availability*.

## Limitations

The following limitations apply to this feature:

- This form of high availability is not supported for Marketplace installations.
- During installation, the platform is configured to use the same account to access AWS resources. Per-user authentication must be set up afterward.

## Prerequisites

Before you begin, please verify that you have met the following requirements.

### AWS infrastructure

- **AWS account**
- **EKS cluster** (see below)

- **S3 bucket:**
  - S3 is required for the base storage layer.
  - A set of permissions must be enabled for the accounts or IAM roles used to access the bucket. For more information, see *S3 Access* in the Configuration Guide.
- **EMR cluster.** For more information, see *Configure for EMR* in the Configuration Guide.
- **Amazon RDS database:**
  - The Trifacta databases must be hosted on the same instance and port in Amazon RDS.
  - PostgreSQL 9.6 or 12.3
  - To ensure sufficient database connections, the instance size must be larger than `m4.large`.
  - The actual databases are installed as part of the installation process.
- **EFS mounts:**
  - One mount is required for each:
    - `conf`
    - `logs`
  - Minimum 10GB of free space is recommended for each mount.
  - For more information on managing your EFS mount points, see <https://docs.aws.amazon.com/efs/latest/ug/creating-using-create-fs.html>.

## EKS cluster

- Kubernetes version 1.15+
- Subnets are available across multiple zones

**NOTE:** You should avoid using a default namespace. This namespace should be shared by other apps using your cluster.

Instance types:

**Tip:** Instance sizes should be larger than `m4.2xlarge`.

	Minimum	Recommended
Cores	8	16
RAM	12 GB	16 GB
Disk space	10 GB minimum	10 GB minimum

**NOTE:** If you are publishing to S3, additional disk space should be reserved for a higher number of concurrent users or larger data volumes. For more information on fast upload to decrease disk requirements, see *S3 Access* in the Configuration Guide.

For more information on installing and managing an EKS cluster, see <https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html>.

## CLIs

The following command line interfaces are referenced as part of this install process:

- `awscli`
- `aws-iam-authenticator`
- `kubect`
- `helm` (version 3)

- docker

## Trifacta assets

The following assets are available from the

1. **Trifacta image file**
2. **Trifacta helm package**
  - a. TGZ file
  - b. Override template file for configuring initial values
3. **Trifacta license key file**  
application. For more information, see

## Install Steps

### Configure Docker image

Please complete the following steps to download and configure the Docker image for use.

#### Steps:

1. Create an AWS Elastic Container Registry (ECR) repository to store information, see
2. Download the image file from the

```
trifacta-docker-image-ha.x.y.z.tar
```

where:

x.y.z

3. Load the image file into your ECR repository. For more information, see <https://docs.aws.amazon.com/AmazonECR/latest/userguide/docker-push-ecr-image.html>
4. The image file has been loaded into the repository.

## Configure AWS Kubernetes

#### Prerequisites:

- AWS Kubernetes cluster is operational.
- These steps use the AWS CLI and

#### Steps:

1. Configure the AWS CLI to use the eks-admin user for your Kubernetes cluster. For more information, see <https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html>
2. Update the Kubernetes configuration (

```
aws eks update-kubeconfig --name <eks-cluster-name> --region <aws-region>
```

where:

<eks-cluster-name>

<aws-region>

**Tip:**  
configuration.

3. Switch to the namespace in the above cluster:

**NOTE:** You should avoid using a default namespace. This namespace should be shared by other apps using your cluster.

```
kubectl config set-context --current --namespace=<namespace>
```

4. Verify that you are ready to use the namespace in the cluster:

```
kubectl get pods
```

5. The cluster is ready for use.

## Configure DB credential secrets

For each of the Trifacta databases that you have installed, you must set up database credential secrets. Please use the following pattern for configuring your database secrets.

**NOTE:** Except for db-credentials-admin, each of these secrets maps to a specific Trifacta database. db-credentials-admin is the username/password of the admin user of the RDS instance. The admin credentials are used to create and initialize all Trifacta databases.

```
kubectl create secret generic db-credentials-webapp --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-scheduling-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-time-based-trigger-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-artifact-storage-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-authorization-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-configuration-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-job-metadata-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-secure-token-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-job-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-contract-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-orchestration-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-optimizer-service --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-batch-job-runner --from-literal=username=<db_username> --from-literal=password=<db_password>
kubectl create secret generic db-credentials-admin --from-literal=username=<db_username> --from-literal=password=<db_password>
```

where:

- <db\_username> = username to access the specified database.
- <db\_password> = password corresponding to the specified database.

## Configure the deployment

### Steps:

1. Unpack the tar file obtained from the FTP site:

```
untar trifacta-ha-setup-bundle-x.y.z.tar
```

where:

`x.y.z` maps to the Release number (Release 7.6.0).

2. The package contains:
  - a. A values override template file: `values.override.template.yaml`
  - b. A Trifacta helm packagegz file
3. Create a copy of the value overrides template file:

```
cp values.override.template.yaml values.override.yaml
```

4. Edit the `values.override.yaml` file. Instructions are below.

### Edit values overrides

Example file:



```

# Template for minimal configuration
# to get a High-availability deployment of Trifacta up and running

replicaCount: 2
image:
  repository: "<PATH TO IMAGE_REPO>"
loadBalancer:
  ssl:
    # ARN To certificate in ACM
    certificateARN: arn:aws:acm:XXXX:certificate/XXXXXXX
nfs:
  conf:
    server: "<NFS SERVER HOST>"
    path: "/"
  logs:
    server: "<NFS SERVER HOST>"
    path: "/"
database:
  host: "<DATABASE HOST>"
  port: "5432"
  type: postgresql

triconfOverrides:
  "aws.accountId" : "<AWS ACCT_ID>"
  "aws.credentialProvider": "<AWS CRED PROVIDER>"
  "aws.systemIAMRole": "arn:aws:iam:XXXX:role/XXXXXXX"
  "aws.s3.bucket.name": "<AWS S3 BUCKET NAME>"
  "aws.s3.key": "<AWS S3 KEY>"
  "aws.s3.secret": "<AWS S3 SECRET>"

# Enable a fluentd Statefulset to collect application logs.
fluentd:
  enabled: true
  # Specify values overrides for fluentd chart here

# Enable a fluent DaemonSet to collect node, K8s dataplane and cluster logs
fluentd-daemonset:
  enabled: false
  # Specify values overrides for the fluentd-daemonset chart here

# Cluster details must be specified if fluentd logging is enabled
global:
  cluster:
    name: "<CLUSTER NAME>" # EKS Cluster name
    region: "<CLUSTER REGION>" # EKS Cluster region

```

**Tip:** Paths to values are listed below in JSON notation (`item.item.item`).

Value	Description
<code>replicaCount</code>	Number of replica nodes of the Trifacta node to maintain as failovers.
<code>image.repository</code>	AWS path to the ECR image repository that you created.

## Configure SSL

By default, SSL is enabled, and a certificate is required.

SSL certificate requirements:

- SSL security is served through the AWS LoadBalancer that serves the Trifacta platform.
  - For more information on the supported SSL configurations, see the `values.yaml` file provided in the Trifacta helm package.
- The SSL certificate must be issued for the FQDN of the Trifacta platform.

Value	Description
loadBalancer.ssl.certificateARN	The ARN for the SSL certificate in the AWS Certificate Manager.

The certificate ARN value references the ARN stored in the AWS Certificate Manager, or you can import your own certificate into ACM. For more information, see <https://docs.aws.amazon.com/acm/latest/userguide/import-certificate.html>.

### To disable:

To disable SSL, please apply the following configuration changes:

```
loadBalancer:
  ssl:
    enabled: false
```

### EFS Mount points

The following values are used to define the locations of the mount points for storing configuration and log data.

**NOTE:** You should have reserved at least 10 GB for each mount point.

Value	Description
nfs.conf.server	Host of the NFS server for the configuration mount point
nfs.conf.path	On the conf server, the path to the storage area. Default is the root location.
nfs.logs.server	Host of the NFS server for the logging mount point
nfs.logs.path	On the conf server, the path to the storage area. Default is the root location.

### Databases

Value	Description
database.host	Host of the Amazon RDS databases <div> <b>NOTE:</b> All Trifacta databases must be hosted on the same RDS instance and available through the same port. </div>
database.port	Port number through which to access the RDS databases. The default value is 5432.
database.type	The type of database. Please leave this value as <code>postgresql</code> .

### trifacta-conf.json overrides

Below you can specify values that are applied to `trifacta-conf.json`, which is the platform configuration file. For more information on these settings, see *Configure for AWS* in the Configuration Guide.

Value	Description
triconfOverrides.aws.accountId	The AWS account identifier to use when connecting to AWS resources.

triconfOverrides.aws.credentialProvider	<p>The type of credential provider to use for individuals authenticating to AWS resources.</p> <div> <p><b>NOTE:</b> During installation, the platform is configured to use the same account to access AWS resources. Per-user authentication must be set up afterward.</p> </div> <p>Supported values:</p> <ul style="list-style-type: none"> <li>• <code>default</code> - credentials are submitted as an AWS key/secret combination.</li> <li>• <code>temporary</code> - credentials are submitted using the same IAM role for all users.</li> </ul> <div> <p><b>Tip:</b> Using a temporary credential provider is recommended.</p> </div> <p>Details are below.</p>
triconfOverrides.aws.systemIAMRole	When the credential provider is set to <code>temporary</code> , this value defines the system-wide IAM role to use to access AWS.
triconfOverrides.aws.s3.key	When the credential is set to <code>default</code> , this value defines the AWS key to use for authentication.
triconfOverrides.aws.s3.secret	When the credential is set to <code>default</code> , this value defines the AWS secret for the AWS key.
triconfOverrides.aws.s3.bucket.name	<p>The default S3 bucket to use.</p> <div> <p><b>NOTE:</b> The AWS account must have read/write access to this bucket.</p> </div>

After the platform is operational, you can apply additional configuration changes to this file through the command line or through the application. For more information, see *Platform Configuration Methods* in the Configuration Guide.

## Configure fluentd

When enabled, a separate set of fluentd pods is launched to collect and forward Trifacta logs.

Value	Description
fluentd.enabled	<p>When set to <code>true</code>, a fluentd Statefulset is deployed to collect application logs.</p> <p>You can specify value overrides to fluentd chart in the following manner:</p> <div> <pre>fluentd:   image:     repository: fluent/fluentd-kubernetes-daemonset     tag: "v1.10.4-debian-cloudwatch-1.0"</pre> </div> <p>See <code>charts/fluentd/values.yaml</code> in the helm package for supported values.</p>
fluentd-daemonset.enabled	When set to <code>true</code> , a fluentd DaemonSet is deployed to collect node, Kubernetes, dataplane, and cluster logs.

If either of the above fluentd logging options is enabled, the following must be specified:

Value	Description
global.cluster.name	This value is the name of the EKS cluster that you created.

global.cluster.region	This value is the name of the region where the EKS cluster was created.
-----------------------	---

## Configure fluentd

Optionally, you can enable fluentd to collect application logs.

## Log destinations:

The logs source for fluentd logs is the Trifacta log directory.

The log destination must be configured. For more information on the fluentd output plugins, see <https://www.fluentd.org/dataoutputs>.

1. Create a `logdestination.conf` configuration file containing a ConfigMap for your log destination:

```
kubectl create configmap fluentd-log-destination --from-file logdestination.conf
```

2. The `logdestination.conf` file must be in a fluentd configuration. Below, you can see an example `logdestination.conf` file, which pushes Trifacta logs to AWS Cloudwatch:

```
<label @NORMAL>
  <match app.*>
    @type cloudwatch_logs
    @id out_cloudwatch_logs_application
    region "#{ENV.fetch('REGION')}"
    log_group_name "/aws/containerinsights/#{ENV.fetch('CLUSTER_NAME')}/application"
    log_stream_name_key stream_name
    auto_create_stream true
    json_handleryajl
    <buffer>
      flush_interval 5
      chunk_limit_size 2m
      queued_chunks_limit_size 32
      retry_forever true
    </buffer>
  </match>
</label>
```

For more information on fluentd configuration file syntax, see <https://docs.fluentd.org/configuration/config-file>.

3. When configured, the `logdestination.conf` file is added as an add-on to the prepackaged fluentd configuration for the Trifacta platform.

## Install

### Install software

After you have configured the values override file, you can use the following command to install the deployment using helm:

```
helm install trifecta <trifacta-helm-package-tgz-file> --namespace <namespace> --values <path-to-values-override-file>
```

where:

- `trifacta-helm-package-tgz-file` = the name of the Helm package that you downloaded from the Trifacta FTP site.
- `namespace` = the AWS Kubernetes namespace value.
- `path-to-values-override-file` = the path in your local environment to the values override file.

### Acquire service URL

Use the following command to retrieve the service URL.

**NOTE:** The service URL is used to access the Trifacta application, where you complete the configuration process. Users create Trifacta objects through the Trifacta application.

```
kubectl get svc trifacta -o json | jq -r '.status.loadBalancer.ingress[0].hostname'
```

### Verify access to the application

Copy and paste the service URL into a supported version of a supported web browser. For more information on supported web browsers, see *Browser Requirements* in the Planning Guide.

**Tip:** You can map CNAME/ALIAS against this service URL through Route53 configurations. For more information, see <https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/getting-started.html>.

The login screen for the Trifacta application should be displayed. Login to the application using the admin credentials.

**You should change the administrator password as soon as you log in. For more information, see *Change Admin Password* in the Admin Guide.**

For more information, see *Login*.

## Other Commands

### Scale platform

Scale the number of Trifacta platform pods through kubectl:

```
kubectl scale statefulset trifacta --replicas=<Desired number of pods>
```

### Restart platform

Restart the Trifacta platform through kubectl:

```
kubectl rollout restart statefulset trifacta
```

### Delete pods

Use the following to delete Trifacta pods:

```
kubectl delete statefulset trifacta
```

## Backups

### Database

By default, Amazon RDS performs periodic backups of your installed databases.

For more information on manual backup of the databases, see *Backup and Recovery* in the Admin Guide.

### EFS mounts

For more information on backing up your EFS mounts through AWS, see <https://docs.aws.amazon.com/efs/latest/ug/awsbackup.html>.

## Configuration

### Set S3 as base storage layer

You must configure access to S3.

**NOTE:** If you are publishing to S3, 50 GB or more is recommended for storage per node. Additional disk space should be reserved for a higher number of concurrent users or larger data volumes. You can also enable fast upload to decrease disk requirements.

For more information, see *S3 Access* in the Configuration Guide.

S3 must be set as the base storage layer. For more information, see *Set Base Storage Layer* in the Configuration Guide.

### Upload the license file

When the platform is first installed, a temporary license is provided. This license key must be replaced by the license key that was provided to you. For more information, see *License Key* in the Admin Guide.

### Configure for EMR

Additional configuration is required to enable the Trifacta platform to run jobs on the EMR cluster. For more information, see *Configure for EMR* in the Configuration Guide.

# 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 `enable_repo` list. For example, if the `centos-base` repo is reported as missing, then the command would be the following:



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

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

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

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-12.16.1-1nodesource.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 or Hortonworks:
  - *Supported Deployment Scenarios for Cloudera*
  - *Supported Deployment Scenarios for Hortonworks*
- 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><b>NOTE:</b> Before you perform an upgrade of your deployment that connects to a MySQL database, please contact Alteryx Customer Success Services.</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>	<p>Runtime configuration file for the Docker image when the database is deployed on a remote server.</p> <div> <p><b>NOTE:</b> You must manage this instance of the database.</p> </div> <p>More information is provided below.</p>

<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.
<i>README-running-trifacta-container-aws.md</i>	Instructions for running the Trifacta container on AWS  <b>NOTE:</b> These instructions are referenced later in this workflow.
<i>README-running-trifacta-container-azure.md</i>	Instructions for running the Trifacta container on Azure  <b>NOTE:</b> These instructions are referenced later in this workflow.
<i>README-building-trifacta-container.md</i>	Instructions for building the Trifacta container  <b>NOTE:</b> This file does not apply if you are using the provided Docker image.

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

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

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

```
docker images
```

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

## Build your own Docker image

As needed, you can build your own Docker image.

### Requirements

- Docker version 17.12 or later. Docker 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

1. Acquire the RPM file from the FTP site:

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

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

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

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

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

6. You can now configure the Docker image.

## Configure Docker Image

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

File	Description
<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><b>NOTE:</b> 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.



ports	Defines the listening port for the Trifacta application. Default is 3005. <div> <b>NOTE:</b> 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 <code>postgresql</code> or <code>mysql</code> .
DB_HOST_NAME	Hostname of the machine hosting the databases. Leave value as <code>localhost</code> for local installation.
DB_HOST_PORT	(Remote only) Port number to use to connect to the databases. Default is 5432. <div> <b>NOTE:</b> If you are modifying, additional configuration is required after installation is complete. See <i>Change Database Port</i> in the Databases Guide.         </div>
DB_ADMIN_USERNAME	Admin username to be used to create DB roles/databases. Modify this value for remote installation. <div> <b>NOTE:</b> If you are modifying this value, additional configuration is required. Please see the documentation for your database version.         </div>
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: <div> <code>/opt/trifacta/conf/trifacta.keytab</code> </div> <div> <b>NOTE:</b> The keytab file must be imported and mounted to this location. Configuration details are provided later.         </div>
KERBEROS_KRB5_CONF	Full path inside of the container where the Kerberos <code>krb5.conf</code> file is located. Default: <div> <code>/opt/krb-config/krb5.conf</code> </div>

### Hadoop distribution client JARs:

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

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

Please modify these lines if you are using Hortonworks.

**Volume properties:**

These properties govern where volumes are mounted in the container.

**NOTE:** These values should not be modified unless necessary.

Property	Description
volumes.conf	Full path in container to the Trifacta configuration directory. Default: <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 conf and logs 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 mysqldb 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 postgresql196-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.

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

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

## JAVA\_HOME:

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

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

The property value must be updated in the following locations:

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

## 3. Install Trifacta package

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

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

Install the package with yum, using root:

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

#### 4. Verify Install

The product is installed in the following directory:

```
/opt/trifacta
```

#### JAVA\_HOME:

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

##### Steps:

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

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

3. Save changes.

#### 5. Install License Key

Please install the license key provided to you by 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*
- *Supported Deployment Scenarios for Hortonworks*
- *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 ftps://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*.

# 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 (1.8.0) below, which can be modified for other supported versions of Java.

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

3. Save changes.

#### 5. Install License Key

Please install the license key provided to you by 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*
- *Supported Deployment Scenarios for Hortonworks*
- *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*.

# 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.**

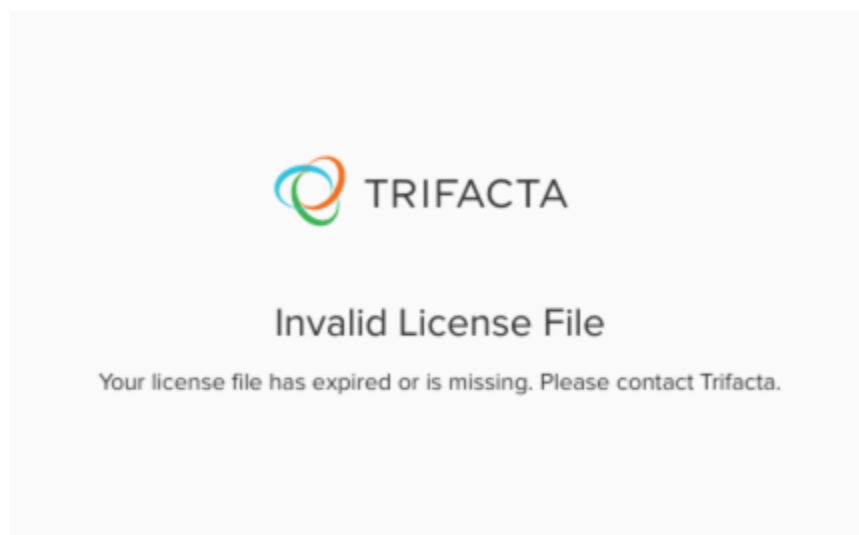
Please contact *Alteryx Support* to extend your license.

## 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 45000 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 5000 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 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.

## 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 SSLv3 TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers RC4:HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    keepalive_timeout 60;
    ssl_session_cache shared:SSL:10m;
    ssl_session_timeout 10m;
    location / {
        proxy_pass http://localhost:3005;
        proxy_next_upstream error timeout invalid_header http_500 http_502 http_503 http_504;
        proxy_set_header    Accept-Encoding    "";
        proxy_set_header    Host                $host;
        proxy_set_header    X-Real-IP           $remote_addr;
        proxy_set_header    X-Forwarded-For     $proxy_add_x_forwarded_for;
        proxy_set_header    X-Forwarded-Proto   $scheme;
        add_header           Front-End-Https    on;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_redirect       off;
    }
    proxy_connect_timeout    6000;
    proxy_send_timeout       6000;
    proxy_read_timeout       6000;
    send_timeout             6000;
}
server {
    listen      80;
    return 301 https://$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	proxy.securityHeaders.enabled=true and proxy.securityHeaders.httpsHeaders=true
---------------------------	-------	---

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

#### Steps:

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

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

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

3. Save your changes and restart the platform.

#### Enable secure cookies

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

**NOTE:** SSL must be enabled.

#### Steps:

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

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

3. Save your changes and restart the platform.

#### 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 AWS

## Contents:

- *AWS Deployment Scenarios*
- *AWS Installations*
  - *Data Preparation for Amazon Redshift and S3 on AWS Marketplace (AMI)*
  - *Trifacta on AWS Marketplace with EMR*
  - *Trifacta on EC2 Instance*
- *AWS Integrations*

## AWS Deployment Scenarios

The following are the basic AWS deployment scenarios.

### Trifacta platform deployed through AWS Marketplace:

Deployment Scenario	Trifacta node installation	Base Storage Layer	Storage - S3	Storage - Redshift	Cluster	Notes
Data Preparation for Amazon Redshift and S3 AWS install through AWS Marketplace CloudFormation template	EC2	S3	read/write	read/write	None	Data Preparation for Amazon Redshift and S3 does not support integration with any running environment clusters. All job execution occurs on the Trifacta node in the Trifacta Photon running environment. This scenario is suitable for smaller user groups and data volumes.
Trifacta AWS install through AWS Marketplace CloudFormation template - with integration to EMR cluster	EC2	S3	read/write	read/write	EMR	This deployment scenario integrates by default with an EMR cluster, which is created as part of the process.  It does not support integration with a Hadoop cluster.

### Trifacta platform installed on AWS:

Deployment Scenario	Trifacta node installation	Base Storage Layer	Storage - S3	Storage - Redshift	Cluster	Notes
Trifacta AWS install with S3 read access	EC2	HDFS	read only	Not supported	EMR	When HDFS is the base storage layer, the only accessible AWS resources is read-only access to S3.
Trifacta AWS install with S3 read/write access	EC2	S3	read/write	read/write	EMR	
Trifacta AWS install with S3 read/write access	EC2	S3	read/write	read/write	AWS Databricks	

### Trifacta platform installed on-premises and integrated with AWS resources:

Deployment Scenario	Trifacta node installation	Base Storage Layer	Storage - S3	Storage - Redshift	Cluster	Notes
---------------------	----------------------------	--------------------	--------------	--------------------	---------	-------

Trifacta on-premises install with S3 read access	On-premises	HDFS	read only	Not supported	Hadoop	When HDFS is the base storage layer, the only accessible AWS resources is read-only access to S3.
Microsoft Azure						Integration with AWS-based resources is not supported.

## Legend and Notes:

Column	Notes
<b>Deployment Scenario</b>	Description of the AWS-connected deployment
<b>Trifacta node installation</b>	Location where the Trifacta node is installed in this scenario. All AWS installations are installed on EC2 instances.
<b>Base Storage Layer</b>	When the Trifacta platform is first installed, the base storage layer must be set.  <b>NOTE:</b> After you have begun using the product, you cannot change the base storage layer.  <b>NOTE:</b> Read/write access to AWS-based resources requires that S3 be set as the base storage layer.
<b>Storage - S3</b>	Trifacta supports read access to S3 when the base storage layer is set to HDFS. For read/write access to S3, the base storage layer must be set to S3.  <b>NOTE:</b> Users can enable access to other S3 buckets by creating new connections through the Trifacta application. For more information, see <i>Connection Types</i> .
<b>Storage - Redshift</b>	For access to Redshift, the base storage layer must be set to S3.
<b>Cluster</b>	List of cluster types that are supported for integration and job execution at scale. <ul style="list-style-type: none"> <li>The Trifacta platform can integrate with at most one cluster. It cannot integrate with two different clusters at the same time.</li> <li>Access to an EMR cluster requires S3 to be the base storage layer.</li> <li>Smaller jobs can be executed on the Trifacta Photon running environment, which is hosted on the Trifacta node itself.</li> <li>For more information, see <i>Running Environment Options</i> in the Configuration Guide.</li> </ul>
<b>Notes</b>	Any additional notes

## AWS Installations

### Data Preparation for Amazon Redshift and S3 on AWS Marketplace (AMI)

Through the Amazon Marketplace, you can license and deploy an AMI of Data Preparation for Amazon Redshift and S3, which does not require integration with a clustered running environment. All job execution happens within the AMI on the EC2 instance that you deploy. For more information, see the Data Preparation for Amazon Redshift and S3 listing on AWS Marketplace.

## Trifacta on AWS Marketplace with EMR

You can deploy an AMI of the Trifacta platform onto an EC2 instance. For more information, see the Trifacta listing for AWS Marketplace.

You can deploy it in either of the following ways:

1. Auto-create a 3-node EMR cluster.
2. Integrate it later with your pre-existing EMR cluster.

For more information, see the Trifacta Self-Managed Enterprise Edition listing on AWS Marketplace.

## Trifacta on EC2 Instance

When the Trifacta platform is installed on AWS, it is deployed on an EC2 instance. Through the EC2 console, there are a few key parameters that must be specified.

**NOTE:** After you have created the instance, you should retain the instanceId from the console, which must be applied to the configuration in the Trifacta platform.

## AWS Integrations

The following table describes the different AWS 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.

AWS Service	Description	Base Storage Layer	Other Required AWS Services
EC2	Amazon Elastic Compute Cloud (EC2) can be used to host the Trifacta node in a scalable cloud-based environment. The following deployments are supported: <ul style="list-style-type: none"><li>• Trifacta Self-Managed Enterprise Edition with or without access to an EMR cluster</li><li>• Data Preparation for Amazon Redshift and S3 on an AMI</li></ul>	Base storage layer can be S3 or HDFS.  If set to HDFS, only read access to S3 is permitted.	
S3	Amazon Simple Storage Service (S3) can be used for reading data sources, writing job results, and hosting the Trifacta databases.  Access to specific S3 buckets can be enabled through new connections.	Base storage layer can be S3 or HDFS.  If set to HDFS, only read access to S3 is permitted.	
Redshift	Amazon Redshift provides a scalable data warehouse platform, designed for big data analytics applications. The Trifacta platform can be configured to read and write from Amazon Redshift database tables.	Base Storage Layer = S3	S3
EMR	For more information on supported versions of EMR, see <i>Configure for EMR</i> in the Configuration Guide.	Base Storage Layer = S3	EC2 instance
Amazon RDS	Optionally, the Trifacta databases can be installed on Amazon RDS. For more information, see <i>Install Databases on Amazon RDS</i> in the Databases Guide.	Base Storage Layer = S3	

### AWS Marketplace integrations:

AWS Service	Description	Base Storage Layer	Other Required AWS Services
-------------	-------------	--------------------	-----------------------------

AMI	Through the AWS Marketplace, you can license and install an Amazon Machine Image (AMI) instance of Data Preparation for Amazon Redshift and S3. This product is intended for smaller user groups that do not need large-scale processing of Hadoop-based clusters.	Base Storage Layer = S3  <b>NO TE:</b> HD FS is not sup por ted.	EC2 instance
EMR	Through the AWS Marketplace, you can license and install an AMI specifically configured to work with Amazon Elastic Map Reduce (EMR), a Hadoop-based data processing platform.	Base Storage Layer = S3	AMI

# 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.