Cloudera Public Cloud

# Getting Started in Cloudera on cloud

**Date published: 2019-08-22**
**Date modified: 2025-08-18**

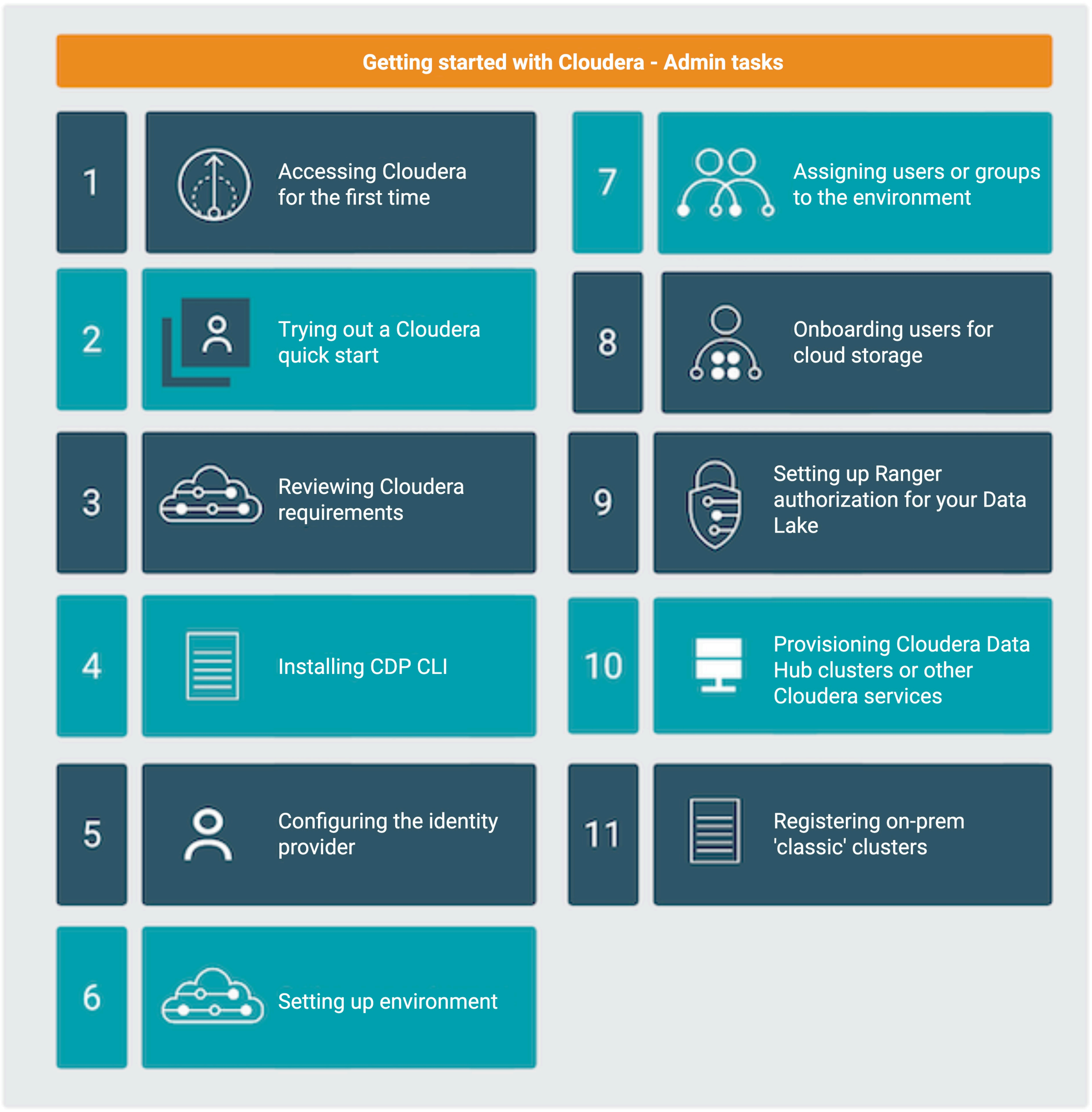
**https://docs.cloudera.com/**

# Legal Notice

# Contents

# Getting started as an admin

Refer to this section if you are a Cloudera admin who is trying to get started in Cloudera.


Getting started with Cloudera - Admin tasks
1 Accessing Cloudera for the first time
2 Trying out a Cloudera quick start
3 Reviewing Cloudera requirements
4 Installing CDP CLI
5 Configuring the identity provider
6 Setting up environment
7 Assigning users or groups to the environment
8 Onboarding users for cloud storage
9 Setting up Ranger authorization for your Data Lake
10 Provisioning Cloudera Data Hub clusters or other Cloudera services
11 Registering on-prem 'classic' clusters

## Accessing Cloudera for the first time

You have the following options to access the Cloudera web interface:

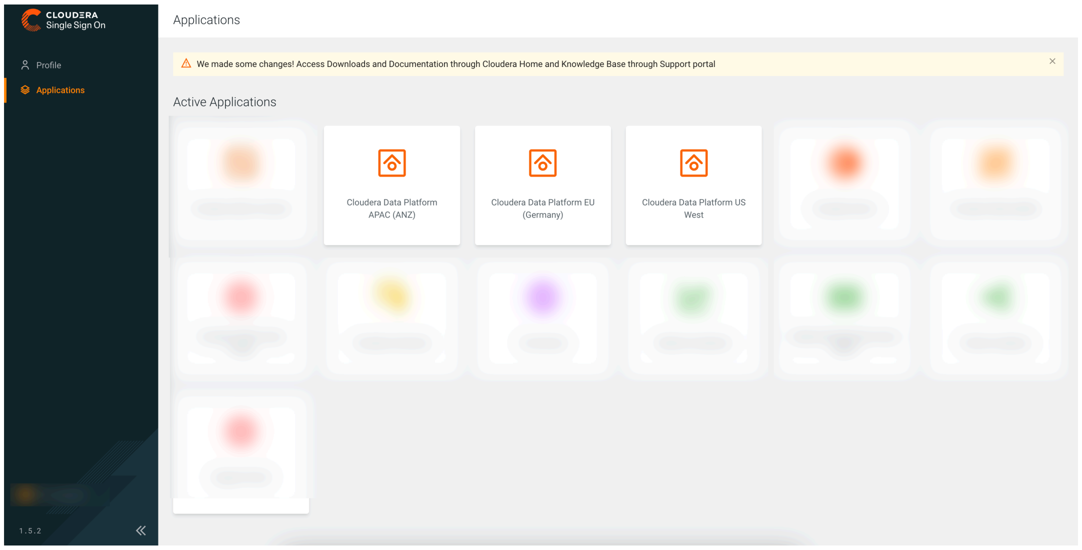
**Using direct link to the Cloudera platform**

Access the Cloudera web interface at https://console.cdp.cloudera.com (if your Cloudera account was created in the Cloudera Control Plane region us-west-1) or https://console.<control-plane-region>.cdp.cloudera.com (if your Cloudera account was created in any other Cloudera Control Plane region). When logging in for the first time, log in by using your MyCloudera credentials.

On the Cloudera **Sign In** page, click Log in with Cloudera SSO and sign in with your MyCloudera credentials.

**Accessing the Cloudera platform from Cloudera Single Sign On portal**

After logging in to the Cloudera Single Sign On portal with your Cloudera credentials, you can access Cloudera platform through the **Applications** page.


CLOUDERA
Single Sign On
Profile
Applications
Applications
Active Applications
Cloudera Data Platform APAC (ANZ)
Cloudera Data Platform EU (Germany)
Cloudera Data Platform US West
1.5.2
We made some changes! Access Downloads and Documentation through Cloudera Home and Knowledge Base through Support portal

## Trying out a Cloudera quick start

If you would like to quickly set up Cloudera for evaluation purposes, you can use our AWS Quick Start, Azure Quick Start, or Google Cloud Quick Start.

## Reviewing cloud provider requirements

You should review the cloud provider requirements for setting up a Cloudera environment:

- AWS: AWS Requirements, AWS Reference Network Architecture, and AWS environment validation tool.
- Azure: Azure Requirements
- GCP: Google Cloud Requirements

## Installing CDP CLI

You can install and configure CDP CLI. See CLI client setup.

## Setting up Identity provider

In order to add users from your organization to Cloudera, set up your identity provider. For instructions, refer to Onboarding users.

## Registering an environment

Register an environment for your organization. An environment determines the specific cloud provider region and virtual network in which resources can be provisioned, and includes the credential that should be used to access the cloud provider account. For instructions, refer to AWS environments, Azure environments, or Google Cloud environments documentation.

## Assigning users or groups to your environment

Once your environment is up and running, you should assign users or groups to the environment and then perform user sync. For instructions, refer to Enabling admin and user access to environments.

### Onboarding users and groups for cloud storage

The minimal setup for cloud storage defined in environment prerequisites spins up a Cloudera environment and Data Lake with no end user access to cloud storage. Adding users and groups to a Cloudera cluster involves ensuring they are properly mapped to IAM roles to access cloud storage. For instructions, refer to:

- Onboarding Cloudera users and groups for AWS cloud storage
- Onboarding Cloudera users and groups for Azure cloud storage
- Onboarding Cloudera users and groups for GCP cloud storage

### Setting up Ranger authorization for your Data Lake

Once your environment is up and running, you should log in to Ranger and create policies for access to specific tables and databases. You can either log in to Hive first and create resources and then create policies for them in Ranger, or you can create Ranger policies in advance.

For instructions on how to access your Data Lake cluster, refer to Accessing Data Lake services. For instructions on how to set up authorization in Ranger, refer to Using Ranger to provide authorization documentation.

### Provisioning compute resources

After performing these steps, you are set to start provisioning compute resources (Cloudera Data Hub clusters, Cloudera Data Warehouse, and so on). For more information, refer to the following documentation:

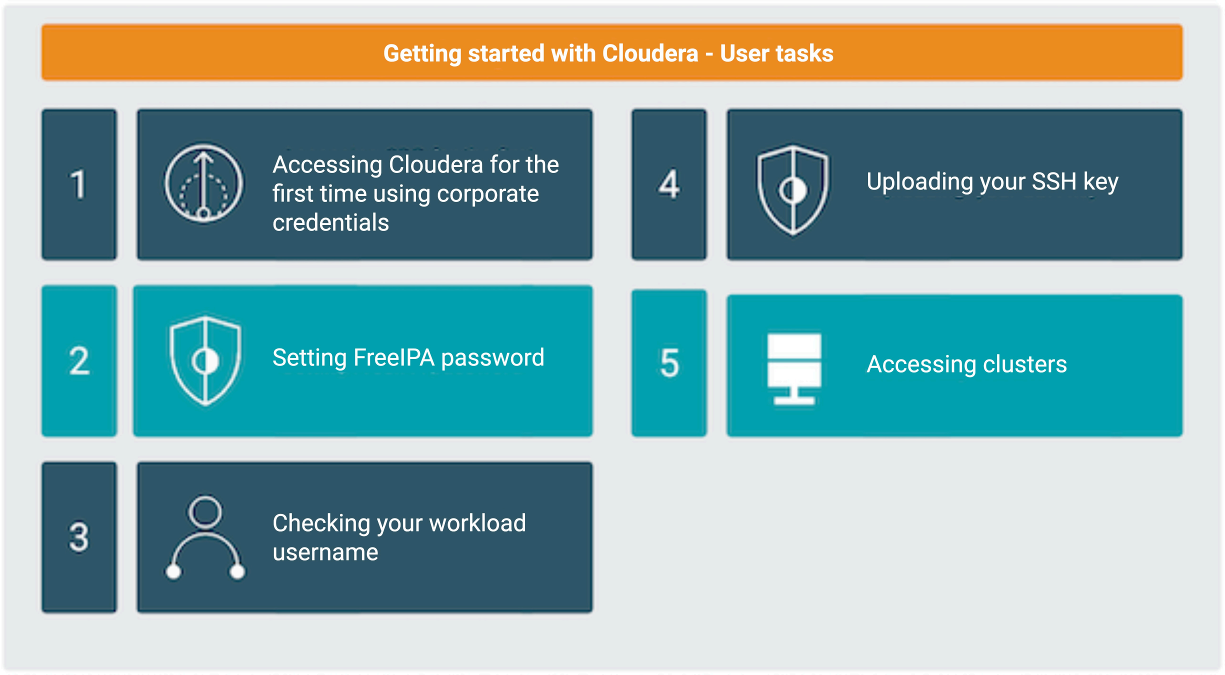
- Cloudera Data Engineering
- Cloudera DataFlow
- Cloudera Data Hub
- Cloudera Data Warehouse
- Cloudera AI
- Cloudera Operational Database

### Registering your existing clusters

You can optionally register your existing CDH and HDP clusters in Cloudera if you would like to generate a workload, data movement, and compute capacity plan and replicate your data. For instructions, refer to Managing classic clusters.

# Getting started as a user

Refer to this section if you are a non-admin Cloudera user who is trying to get started in Cloudera.


Getting started with Cloudera - User tasks
1
Accessing Cloudera for the first time using corporate credentials
2
Setting FreeIPA password
3
Checking your workload username
4
Uploading your SSH key
5
Accessing clusters

## Accessing Cloudera for the first time

Access the Cloudera platform after logging in to the Identity Provider used in your enterprise. Reach out to your Cloudera administrator for the required credentials and the enterprise specific steps.

For more information about the sign-in process, see Signing in to Cloudera on cloud as a user.

## Setting workload password

If you are planning to access certain resources such as:

- Access clusters via SSH
- Connect to clusters via JDBC or ODBC
- Access Data Analytics Studio (DAS)
- Access Cloudera AI workspaces

you must access these by using your workload password. Initially, you must set your workload password, and then you need to reset it each time a new environment is shared with you. For more information about when and how to set and reset your workload password, refer to Accessing non-SSO interfaces using IPA credentials.

## Checking your workload user name

If you are planning to access certain resources such as:

- Access clusters via SSH
- Connect to clusters via JDBC or ODBC
- Access Data Analytics Studio (DAS)
- Access Cloudera AI workbenches

you must access these by using your workload user name. To check your workload user name, navigate to the Cloudera Management Console > User Management > Users, find your user and check your Workload User Name.

### Uploading SSH key

As an alternative for using workload password for SSHing to workload clusters, you can also upload your SSH public key to Cloudera and use the matching SSH private key for access. For more information, refer to Managing SSH keys.

### Accessing resources

Your Cloudera administrator decided which Cloudera resources are available to you. You can access these resources from the Cloudera web interface. For more information, refer to the following documentation:

- Data Hub
- Cloudera Data Engineering
- Cloudera DataFlow
- Cloudera Data Hub
- Cloudera Data Warehouse
- Cloudera AI
- Cloudera Operational Database

# Creating and managing Cloudera deployments

In this topic, we provide an overview of best practices for deploying Cloudera and demonstrate how to create and manage Cloudera deployments through a simple yet powerful Terraform framework.

If you are looking for a high-level overview of best practices for setting up Cloudera by using our standardized Terraform-based Cloudera deployment patterns, continue reading this article.

**Note:**

Creating new Cloudera deployments, adding data services, and managing the platform are also possible via the Cloudera web interface and CDP CLI. These options enable you to create customized deployments with a high degree of flexibility.

## What is a Cloudera deployment

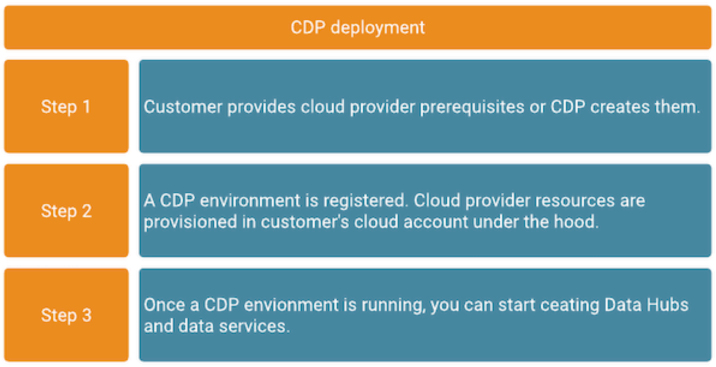
A Cloudera deployment is a set of Cloudera management services and data services including related cloud provider resources that exist in your AWS, Azure, or GCP account. It is a combination of the cloud infrastructure that may span multiple cloud providers and regions, and the Cloudera platform that abstracts this underlying cloud provider infrastructure into an integrated, unified, logical data platform layer.

Each Cloudera deployment consists of Cloudera services and the underlying cloud provider resources.

For Cloudera to be deployed, a set of cloud provider prerequisites needs to be provided first, including a virtual network and subnets, storage accounts, and access roles/identities and policies. These cloud provider prerequisites are typically customer-managed and exist in the cloud provider account independently of Cloudera services. As such, they may be shared with other, non-Cloudera cloud services.

Once the cloud provider prerequisites are present, a Cloudera environment can be deployed in the virtual network. Once your Cloudera environment is up and running, your core Cloudera and cloud provider infrastructure is in place and you can start creating Cloudera Data Hub clusters and data services in order to run workloads. When these services are created, additional cloud provider resources such as VM instances, security groups, and load balancers are deployed in your cloud account. For each service, you can select which subnets of the underlying virtual network and what storage locations within your specified storage accounts they should use.

These three high-level deployment steps are described in the following diagram:


CDP deployment
Step 1
Customer provides cloud provider prerequisites or CDP creates them.
Step 2
A CDP environment is registered. Cloud provider resources are provisioned in customer's cloud account under the hood.
Step 3
Once a CDP envionment is running, you can start ceating Data Hubs and data services.

Cloudera deployment can be performed by using either Cloudera web interface or Cloudera CLI, or Terraform-based Cloudera deployment patterns. Continue reading to learn about deploying Cloudera using Terraform.

**Note:** As a best practice, cloud provider prerequisites (such as a VPC/VNet and subnets) should be created and managed outside of Cloudera. The Terraform quickstart module provided creates all these cloud provider prerequisites, but in case you would like to use an existing AWS VPC or Azure VNet and subnets, you can achieve this by providing a few additional optional parameters.

**Note:** If you would like to understand the details of the automation tooling provided here or are looking for more flexibility for your automated Cloudera deployments, refer to Terraform module for deploying Cloudera.

### Related Information

Cloudera's Shared Responsibility Model

## Cloudera deployment patterns

To simplify the task of defining and creating Cloudera deployments, we provide and describe a set of predefined target architectures recommended by Cloudera. These target architectures are called **deployment patterns**.

In Cloudera's Terraform framework, each pattern is represented by a deployment template that allows you to quickly instantiate one of the reference deployments. The templates can be used as a starting point and modified according to your needs. You can learn more about the recommended configurations of Cloudera on cloud from the documentation of our end-to-end deployment patterns as well as our network reference architectures for AWS and Azure.

Currently, we provide templates that represent the following deployment patterns, each matching a different use case:

| | |
|---|---|
| **Private** | Production-like setup fully deployed on private subnets without public IPs or direct outbound internet access. Demonstrates a possible production deployment with typical network security features enabled.<br><br>**Note:** Since private subnets have no internet connectivity by default, such a setup of Cloudera would not function out of the box, unless additional network components such as Internet Gateways or NAT Gateways are present. For convenience, we deploy these additional components by default. To turn off this behavior (for example when deploying to an existing private network), you can set the optional parameter private_network_extensions=false. |
| **Semi-private** | Production-like setup with access over the public internet to the user interfaces and API endpoints only. It serves as a reference for production deployments without the need for configuring VPNs, jump hosts, and user-defined routing for outbound (egress) traffic |

| | |
|---|---|
| **Public** | Simple setup with access over public internet to all endpoints and with a minimal footprint. It can be used for quick testing, tutorial, demonstration, or simply to understand the internal workings of Cloudera on cloud. This setup is not secure enough for production but can be used for proof of concept. |

**Note:**

A real-life production Cloudera deployment often differs from the patterns described here. The examples that we provide intend to simplify the initial creation of a Cloudera deployment and serve as a reference for customized, customer-owned templates. While based on observed patterns and best practices, these templates are provided as-is and maintained by the Cloudera field community. If you plan to set up Cloudera for production, we assume that you customize the provided examples to match your IT networking and security guidelines.

## Cloudera deployment pattern definitions

Deployment patterns are predefined architectures recommended by Cloudera that simplify the task of defining and creating Cloudera deployments. There are many options available for deploying Cloudera, but as a best practice, Cloudera recommends that you use one of the following three deployment patterns: private, semi-private, or public.

These patterns are based on the identically named network reference architectures and extend them, by incorporating Cloudera's recommended configuration for deploying Cloudera in multiple availability zones, selecting the Data Lake scale, configuring storage access policies, and setting up fine-grained access control.

As can be expected, each of these deployment patterns brings a unique trade-off among various aspects, such as ease of setup, security provided, workloads supported, and so on. Read the following content to understand what specific networking, IAM, and storage cloud provider configurations, and Cloudera configurations are applied as part of the supported deployment patterns.

### Related Information

Overview of AWS resources used by Cloudera

Overview of Azure resources used by Cloudera

# Deploy Cloudera using Terraform

This guide demonstrates how to deploy Cloudera on AWS, Azure, or GCP by using one of the Cloudera deployment templates.

The templates use Terraform, an open source Infrastructure as Code (IaC) software tool for defining and managing cloud or data center infrastructure. You interface the templates via a simple configuration file residing in a GitHub repository.

For an overview of best practices for deploying Cloudera, refer to Creating and managing Cloudera deployments.

**Note:** As a best practice, cloud provider prerequisites (such as a VPC/VNet and subnets) should be created and managed outside of Cloudera. The provided Terraform quickstart module creates all these cloud provider prerequisites. However, if you would like to use an existing AWS VPC, Azure VNet and subnets, or GCP VPC, you can provide additional optional parameters.

## Prerequisites

Before deploying Cloudera, you should make sure that your cloud account meets the basic requirements and that you've installed a few prerequisites.

To meet these requirements and install the prerequisites, refer to the following documentation:

- Cloud provider requirements
- Prerequisites for deploying Cloudera

You should also familiarize yourself with the background information about Cloudera deployment patterns and deployment pattern definitions described in Creating and managing Cloudera deployments.

**Note:** Terraform does not support editing environment resources after deployment.

Next, you can follow the instructions below for deploying Cloudera.

## Deploy Cloudera

Setting up a Cloudera deployment involves cloning a GitHub repository, editing the configuration, and running Terraform commands to launch the deployment.

### Step 1: Clone the repository

The cdp-tf-quickstarts repository contains Terraform resource files to quickly deploy Cloudera on cloud and associated pre-requisite cloud resources. It uses the Cloudera Terraform Modules provided by Cloudera to do this.

Clone this repository and navigate to the local directory with the cloned repository:

```
git clone https://github.com/cloudera-labs/cdp-tf-quickstarts.git
cd cdp-tf-quickstarts
```

### Step 2: Edit the configuration file for the required cloud provider

In the cloned repository, change to the required cloud provider directory - AWS, Azure, or GCP.

Edit the input variables in the configuration file as required:

**For AWS**

```
cd aws
cp terraform.tfvars.template terraform.tfvars
vi terraform.tfvars
```

**For Azure**

```
cd azure
mv terraform.tfvars.template terraform.tfvars
vi terraform.tfvars
```

**For GCP**

```
cd gcp
mv terraform.tfvars.template terraform.tfvars
vi terraform.tfvars
```

Following is a sample configuration file indicating the values to be changed. The variables are explained after the sample. You should review and update all the variables.

**For AWS**

```
# ------- Global settings -------
```

```
env_prefix = "<ENTER_VALUE>" # Required name prefix for cloud and Cloudera
 resources, e.g. cldr1

# ------- Cloud Settings -------
aws_region = "<ENTER_VALUE>" # Change this to specify Cloud Provider re
gion, e.g. eu-west-1

# ------- Cloudera Environment Deployment -------
deployment_template = "<ENTER_VALUE>"  # Specify the deployment pattern
below. Options are public, semi-private or private
```

**For Azure**

```
# ------- Global settings -------
env_prefix = "<ENTER_VALUE>" # Required name prefix for cloud and Cloudera
 resources, e.g. cldr1

# ------- Cloud Settings -------
azure_region = "<ENTER_VALUE>" # Change this to specify Cloud Provider
region, e.g. eastus
# ------- Cloudera Environment Deployment -------
deployment_template = "<ENTER_VALUE>"  # Specify the deployment pattern
below. Options are public, semi-private or private
```

**For GCP**

```
# ------- Global settings -------
env_prefix = "<ENTER_VALUE>" # Required name prefix for cloud and Cloudera
 resources, e.g. cldr1

# ------- Cloud Settings -------
gcp_project = "<ENTER_VALUE>" # Change this to specify the GCP Project ID
gcp_region = "<ENTER_VALUE>" # Change this to specify Cloud Provider regi
on, e.g. europe-west2

# ------- Cloudera Environment Deployment -------
deployment_template = "<ENTER_VALUE>"  # Specify the deployment pattern
below. Options are public, semi-private or private
```

As a result of this step, your configuration file should look similar to the following:

**For AWS**

```
# ------- Global settings -------
env_prefix = "TEST-ENV" # Required name prefix for cloud and Cloudera res
ources, e.g. cldr1

# ------- Cloud Settings -------
aws_region = "EU-WEST-1" # Change this to specify Cloud Provider region,
 e.g. eu-west-1

# ------- Cloudera Environment Deployment -------
deployment_template = "PUBLIC"  # Specify the deployment pattern below. O
ptions are public, semi-private or private
```

**For Azure**

```
# ------- Global settings -------
env_prefix = "TEST-ENV" # Required name prefix for cloud and Cloudera res
ources, e.g. cldr1
```

```
# ------- Cloud Settings -------
azure_region = "WESTEUROPE" # Change this to specify Cloud Provider region
, e.g. eastus

# ------- Cloudera Environment Deployment -------
deployment_template = "PUBLIC"  # Specify the deployment pattern below. O
ptions are public, semi-private or private
```

**For GCP**

```
# ------- Global settings -------
env_prefix = "CDP-DEMO" # Required name prefix for cloud and Cloudera res
ources, e.g. cldr1

# ------- Cloud Settings -------
gcp_project = "MY-GCP-PROJECT-ID" # Change this to specify the GCP Project
 ID
gcp_region = "US-CENTRAL1" # Change this to specify Cloud Provider region,
 e.g. europe-west2
# ------- Cloudera Environment Deployment -------
deployment_template = "PUBLIC"  # Specify the deployment pattern below. O
ptions are public, semi-private or private
```

The following tables explain the mandatory inputs that need to be provided in the configuration file.

Table 1: Mandatory inputs

**For AWS**

| Input | Description | Default value |
|---|---|---|
| env_prefix | A string prefix that will be used to name the cloud provider and Cloudera resources created. | Not set |
| aws_region | The AWS region in which the cloud prerequisites and Cloudera will be deployed. For example, eu-west-1. For a list of supported AWS regions, see Supported AWS regions. | Not set |
| deployment_template | The selected deployment pattern. Values allowed:<br><br>private, semi-private and public. | public |

**For Azure**

| Input | Description | Default value |
|---|---|---|
| env_prefix | A string prefix that will be used to name the cloud provider and Cloudera resources created. | Not set |
| azure_region | The Azure region in which the cloud prerequisites and Cloudera will be deployed. For example, eastus. For a list of supported Azure regions, see Supported Azure regions. | Not set |
| deployment_template | The selected deployment pattern. Values allowed:<br><br>private, semi-private and public. | public |

### For GCP

| Input | Description | Default value |
|---|---|---|
| gcp_region | The GCP region in which the cloud prerequisites and Cloudera will be deployed. For example, eastus. For a list of supported GCP regions, see Supported GCP regions. | Not set. |
| gcp_project | GCP project ID that will be used for Cloudera. | Not set. |
| env_prefix | A string prefix that is used to name the created cloud provider and Cloudera resources. | Not set. |
| deployment_template | The selected deployment pattern. Values allowed: <br> private, semi-private, public. | public |

The following tables explain the optional inputs that can be added to the configuration file. While the mandatory input attributes are included in the configuration file and only their values need to be entered, optional attributes and values must be added manually.

Table 2: Optional inputs

### For AWS

| Input | Description | Default value |
|---|---|---|
| aws_key_pair | The name of an AWS keypair that exists in your account in the selected region. | Not set |
| ingress_extra_cidrs_and_ports | Inbound access to the UI and API endpoints of your deployment will be allowed from the CIDRs (IP ranges) and ports specified here. <br> Enter your machine’s public IP here, with ports 443 and 22. If unsure, you can check your public IP address here. | CIDRs are not set. <br> Ports are set to 443, 22 by default. |
| create_vpc | Flag to specify if the VPC should be created. | true |
| cdp_vpc_id | VPC ID for Cloudera environment. Required if create_vpc is false | Empty string |
| cdp_public_subnet_ids | List of public subnet ids. Required if create_vpc is false. Can be an empty list depending on deployment_template. | Empty list |
| cdp_private_subnet_ids | List of private subnet ids. Required if crea te_vpc is false. | Empty list |
| private_network_extensions | Enable creation of resources for connectivity to Cloudera Control Plane (public subnet and NAT Gateway) for Private Deployment. Only relevant for private deployment template. | true |

| | | |
|---|---|---|
| env_tags | Define environment-level tags for your resources, such as owner, project, and end date. For more information about custom tags, see the Defining custom tags documentation.<br><br>Using the owner, project, and end date example, define the environment-level tags as follows:<br><br>`env_tags = {`<br>`  owner   = "<ENTER_VALUE>"`<br>`  project = "<ENTER_VALUE>"`<br>`  enddate = "<ENTER_VALUE>"`<br>`}` | Not set |

## For Azure

| Input | Description | Default value |
|---|---|---|
| public_key_text | An SSH public key string to be used for the nodes of the Cloudera environment. | Not set |
| ingress_extra_cidrs_and_ports | Inbound access to the UI and API endpoints of your deployment will be allowed from the CIDRs (IP ranges) and ports specified here.<br><br>Enter your machine's public IP here, with ports 443 and 22. If unsure, you can check your public IP address here. | CIDRs are not set.<br><br>Ports are set to 443, 22 by default. |
| create_vnet | Flag to specify if the VNet should be created. | true |
| cdp_resourcegroup_name | Preexisting Azure resource group for Cloudera environment. Required if create_v net is false. | Empty string |
| cdp_vnet_name | Preexisting VNet name for Cloudera environment. Required if create_vnet is false. | Empty string |
| cdp_subnet_names | List of preexisting subnet names for Cloudera resources. Required if create_vnet is false. | Empty list |
| cdp_gw_subnet_names | List of preeexisting subnet names for Cloudera Gateway. Required if create_vnet is false. Can be an empty list depending on depl oyment_template. | Empty list |
| cdp_delegated_subnet_names | List of preeexisting subnet names for Postgres flexible servers. Can be an empty list depending on deployment_template. | Empty list |
| env_tags | Define environment-level tags for your resources, such as owner, project and end date. For more information about custom tags, see the Defining custom tags documentation.<br><br>Using the owner, project, and end date example, define the environment-level tags as follows:<br><br>`env_tags = {`<br>`  owner   = "<ENTER_VALUE>"`<br>`  project = "<ENTER_VALUE>"`<br>`  enddate = "<ENTER_VALUE>"`<br>`}` | Not set |

## For GCP

| Input | Description | Default value |
|---|---|---|
| create_vpc | Flag to specify if the VPC should be created. | True |

| | | |
|---|---|---|
| cdp_vpc_name | VPC name for Cloudera environment. Required if create_vpc is false. | Empty string. |
| cdp_subnet_names | List of subnet names for Cloudera resources. Required if create_vpc is false. | Empty list. |
| public_key_text | An SSH public key string used for the Cloudera environment nodes. If not specified, an SSH keypair is generated as part of the code. | Not set. |
| ingress_extra_cidrs_and_ports | Inbound access to the UI and API endpoints of your deployment will be allowed from the CIDRs (IP ranges) and ports specified here.<br><br>Enter your machine's public IP here, with ports 443 and 22. If unsure, you can check your public IP address here. If not specified, the public IP of the machine is looked up where the Terraform code is being executed. | CIDRs are not set.<br><br>Ports are set to 443 and 22 by default. |
| env_tags | Define environment-level tags for your resources, such as owner, project, and end date. For more information about custom tags, see the Defining custom tags documentation.<br><br>Using the owner, project, and end date example, define the environment-level tags as follows:<br><br>`env_tags = {`<br>`  owner   = "<ENTER_VALUE>"`<br>`  project = "<ENTER_VALUE>"`<br>`  enddate = "<ENTER_VALUE>"`<br>`}` | Not set. |

## Step 3: Launch the deployment

Run the Terraform commands to validate the configuration and launch the deployment:

```
terraform init
terraform apply
```

Terraform displays a plan with the list of cloud provider and Cloudera resources that will be created.

When you are prompted, type yes to instruct Terraform to perform the deployment. Typically, this takes about 60 minutes. Once the deployment is complete, Cloudera will print output similar to the following:

```
Apply complete! Resources: 46 added, 0 changed, 0 destroyed.
```

You can navigate to the Cloudera web interface at https://cdp.cloudera.com/ and see your deployment progress. Once the deployment completes, you can create Cloudera Data Hub clusters and data services.

## Clean up the Cloudera environment and infrastructure

If you no longer need the infrastructure provisioned by Terraform, run the following command to remove the deployment infrastructure and terminate all resources:

```
terraform destroy
```

# Cloud provider requirements

Review the requirements related to the AWS, Azure, or GCP account that you would like to use with Cloudera.

### AWS account

To follow this guide, you need to have access to an AWS account. In this guide, we assume that you have a newly created account or a sub-account with default settings and no network restrictions (custom routes to the Internet) or policy restrictions (AWS Organizations policies or Service Control Policies (SCPs)) in place. SCPs configured on the parent AWS Organization of your AWS account may impact certain steps described in this guide and may require that you follow a custom deployment path.

You also need the following account-level AWS settings:

- An AWS role that has permissions to create IAM objects (cross-account role and policy, IAM roles and policies, S3 buckets). You will also need to create credentials for your IAM user role. You will need these in the next section for configuring the Terraform Provider for AWS on your machine. See AWS security credentials.
- Select a supported AWS region for your deployment. See Supported AWS regions.
- A vCPU quota of at least 200 cores. You may need a higher limit for larger deployments. You can check your current vCPU quota under the name Running On-Demand Standard (A, C, D, H, I, M, R, T, Z) instances. Make sure that the quota value is 200 or larger. See the AWS documentation for requesting an EC2 vCPU limit increase.
- An elastic IP quota of at least 5 elastic IPs (for the public and semi-private patterns). The recommended quota is 10 elastic IPs.

### Azure account

To follow this guide, you need to have access to an Azure account. In this guide, we assume that you have a newly created account or a sub-account with default settings and no network restrictions (custom routes to the Internet) or policy restrictions (Azure Organizations policies or Service Control Policies (SCPs)) in place. SCPs configured on the parent Azure Organization of your Azure account may impact certain steps described in this guide and may require that you follow a custom deployment path.

You also need the following tenant and subscription-level Azure permissions and settings:

- You need to have write permissions for Azure AD in order to create the Azure service principal (App registration).
- Your user needs to have Contributor privileges at least at the scope of the Azure resource group in which you will deploy Cloudera; That is, your user needs to have permissions to create managed identities, grant role assignments at the scope of the resource group, and create VNet/subnets and storage accounts.
- Select a supported Azure region for your deployment. See Supported Azure regions.
- A Total Regional vCPU quota of at least 200 cores. You may need a higher limit for larger deployments. For requesting a compute quota increase, see the Azure documentation. Make sure that the Standard DSv3 Family vCPUs quota is also 200 cores or larger.
- A Public IP Addresses quota of at least 5 public IP addresses (for the public and semi-private patterns). The recommended quota is 10 IP addresses.
- Make sure that all services required by Cloudera are available in your selected Azure region. You may need to request that Azure Support whitelists a particular service (such as Azure Database for PostgreSQL) for your subscription in your selected region. See Overview of Azure resources used by Cloudera.

### GCP account

To follow this guide, you need to have access to a GCP account. In this guide, we assume that you have a newly created account or a sub-account with default settings and no network restrictions (custom routes to the Internet) or policy restrictions (GCP Organizations policies or Service Control Policies (SCPs)) in place. SCPs configured on the parent GCP Organization of your GCP account may impact certain steps described in this guide and may require that you follow a custom deployment path.

You also need the following tenant and subscription-level GCP permissions and settings:

- Contributor privileges for the GCP project, and the following permissions:
  - Compute Admin to create and manage Compute Engine instances.
  - VPC Admin to create and manage VPCs, subnets, and network resources.
  - Storage Admin to provision and manage Google Cloud Storage buckets.
  - IAM Admin to create managed identities and assign roles.
  - Service Account Admin to create and manage service accounts. Service accounts must have the following roles:
    - roles/storage.admin for managing GCS buckets;
    - roles/compute.admin for creating VMs and managing networks;
    - roles/resourcemanager.projectIamAdmin for managing project-level IAM roles.
  - Cloud API Access to access APIs and services required by Cloudera.
- The following GCP APIs should be enabled for your project:
  - Compute Engine API (compute.googleapis.com) to create and manage virtual machines.
  - Cloud Storage API (storage.googleapis.com) to provision storage buckets.
  - IAM API (iam.googleapis.com) to manage service accounts and roles.
  - Cloud Resource Manager API (cloudresourcemanager.googleapis.com) for project-level role assignments.
- Ensure a total regional vCPU quota of at least 200 cores is available. For larger deployments may require additional quota, which can be requested from GCP support.
- A minimum of 5 public IP addresses is required for public or semi-private deployments. We recommend a quota of 10 public IP addresses for flexibility.
- Select a supported GCP region for your deployment. For more information, see Supported GCP regions.
- Make sure that all services required by Cloudera are available in your selected GCP region. You may need to request that GCP Support whitelists a particular service (such PostgreSQL database) for your subscription in your selected region. For more information, see Overview of GCP resources used by Cloudera.

### Related Information

Overview of AWS resources used by Cloudera

## Prerequisites for deploying Cloudera

To set up Cloudera via deployment automation using this guide, the following prerequisites must be installed and configured in your local environment:

- Terraform version 1.3 or newer
- Terraform Provider for AWS, Azure, or GCP
- Terraform Provider for Cloudera

### Install Terraform

Install Terraform version 1.3 or newer. See installation instructions in the Terraform installation guide.

### Configure Terraform Provider for AWS, Azure, or GCP

For AWS examples, see Build Infrastructure | Terraform | HashiCorp Developer.

For Azure examples, see Build Infrastructure - Terraform Azure Example | Terraform | HashiCorp Developer.

For GCP examples, see Build infrastructure on GCP | Terraform | HashiCorp Developer.

### Configure Terraform Provider for Cloudera

Configure Terraform Provider for Cloudera by downloading or creating a Cloudera configuration file. You can find the required steps for Generating an API access key and Configuring Cloudera client in our documentation.

### Reviewing Cloud Provider Requirements

You should review the cloud provider requirements for setting up a Cloudera environment:

- AWS: AWS Requirements
- Azure: Azure Requirements
- GCP: Google Cloud Requirements

# Terraform module for deploying Cloudera

The Terraform Modules for Cloudera Prerequisites on AWS, Azure, and GCP contain Terraform resource files and example variable definition files for creating the prerequisite cloud provider resources required for deploying Cloudera. These modules use the official Terraform Providers for AWS, Azure, or GCP, all maintained by Hashicorp. They include a VPC/VNet configured with public and private subnets according to the network deployment pattern specified, data and log buckets/containers for the Cloudera environment, and several AWS and GCP IAM roles and policies or Azure managed identities to enable fine-grained permissions for access to the Cloudera Control Plane and AWS/Azure services.

Furthermore, the Terraform Module for Cloudera Deployment is used to create a Cloudera credential and deploy a Cloudera environment and a Data Lake.

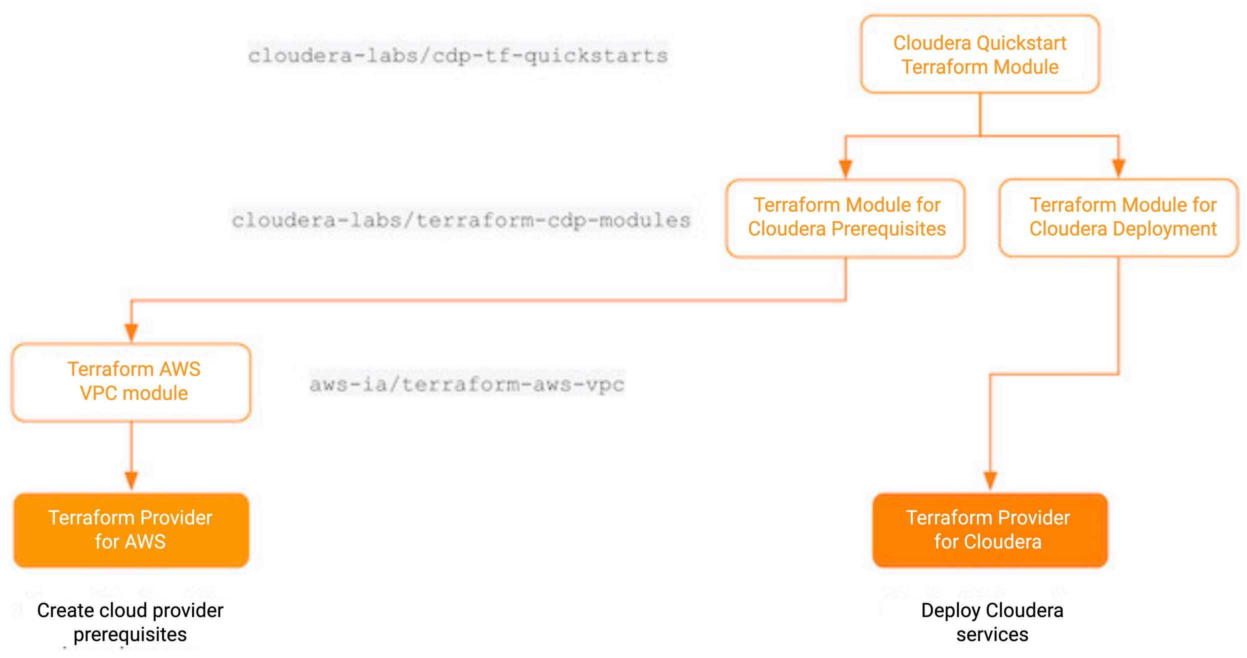
The aforementioned modules support the network deployment patterns described in Cloudera deployment pattern definitions below and are coupled with the Cloudera Quickstart Terraform Module that we provide for simplifying end-to-end setup including both the cloud prerequisites and the Cloudera services.

The Cloudera Terraform modules currently support the following commands:

- terraform apply to create or update an environment
- terraform destroy to tear down an environment

The Cloudera Terraform modules do not support the terraform import command to import existing Cloudera on Cloud infrastructure into a Terraform configuration.

The following diagram illustrates the hierarchy of modules and providers used by the onboarding automation tooling (AWS is used as an example):


cloudera-labs/cdp-tf-quickstarts
Cloudera Quickstart Terraform Module
cloudera-labs/terraform-cdp-modules
Terraform Module for Cloudera Prerequisites
Terraform Module for Cloudera Deployment
Terraform AWS VPC module
aws-ia/terraform-aws-vpc
Terraform Provider for AWS
Terraform Provider for Cloudera
Create cloud provider prerequisites
Deploy Cloudera services

In our Deploy Cloudera using Terraform onboarding guides, we use these modules to quickly deploy Cloudera.