Cloudera Public Cloud

Getting Started in CDP Public Cloud

Date published: 2019-08-22
Date modified:

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.

Accessing Cloudera for the first time

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.

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:

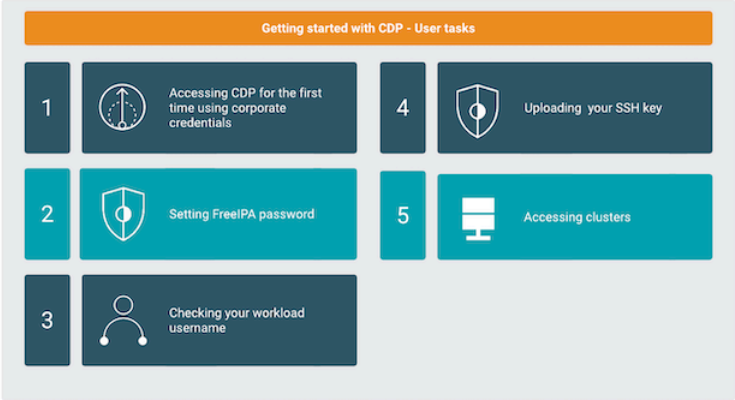
- Cloudera Data Engineering
- Cloudera DataFlow
- Cloudera Data Hub
- Cloudera Data Warehouse
- Cloudera AI
- Cloudera Operational Database

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.



Accessing Cloudera for the first time

Access the Cloudera web interface at https://console.cdp.cloudera.com and log in by using your corporate credentials or other credentials that you received from your Cloudera administrator.

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 is also possible via the Cloudera web interface and CDP CLI. These options enable you to create customized deployments with a high degree of flexibility.

Note:

This guide currently covers deploying Cloudera on AWS and Azure only. For instructions on how to quickly deploy Cloudera on GCP, refer to Cloudera quickstarts.

What is a Cloudera deployment

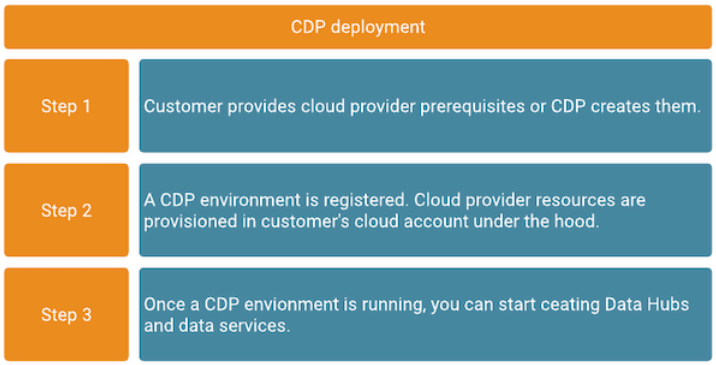
A Cloudera deployment is a set of Cloudera management services and data services including related cloud provider resources that exist in your AWS, Azure, or GCP account. It is a combination of the cloud infrastructure that may span multiple cloud providers and regions, and the Cloudera platform that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

In order 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:



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 Public 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. **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 Public 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.

Cloud provider prerequisites

This section summarizes the networking, IAM, and storage cloud provider configurations that are made when Cloudera is deployed based on one of the deployment patterns.

Networking

For AWS

	Private	Semi-private	Public
VPC	A new VPC is provisioned in your cloud provider account.	A new VPC is provisioned in your cloud provider account.	A new VPC is provisioned in your cloud provider account.
Subnets	1x /18 public subnet for network access (when using private_network_extension=true) 3x /18 private subnets (for cluster nodes)	3x /19 public subnets (for load balancers) 3x /19 subnets (for cluster nodes)	3x /18 public subnets (for load balancers and cluster nodes)
Public IPs	1 Elastic IP is allocated (when using private_network_extension=true)	3 Elastic IPs are allocated (for load balancers)	All deployed nodes have public IPs
Egress traffic	One AWS Internet Gateway and one AWS Public and Private NAT Gateway are created (when using private_network_extension=true)	One AWS Internet Gateway, three AWS Public and Private NAT Gateways (1 per public subnet)	One AWS Internet Gateway
Ingress traffic	Public Load Balancer (when using private_network_extension=true)	Public Load Balancer	Public Load Balancer

Security groups	2 Security Groups (Rules set up based on user input/ configuration)	2 Security Groups (Rules set up based on user input/ configuration)	2 Security Groups (Rules set up based on user input/ configuration)

For Azure

	Private	Semi-private	Public
Resource Group	A single new resource group is created.	A single new resource group is created.	A single new resource group is created.
VNet	A new VNet is provisioned in your cloud provider account.	A new VNet is provisioned in your cloud provider account.	A new VNet is provisioned in your cloud provider account.
Subnets	3x /18 subnets (for load balancers and cluster nodes)	1x /18 subnets (for load balancers)) 3x /18 subnets (for cluster nodes)	3x /18 subnets
Public IPs	No public IPs are assigned.	Load balancers have public hostnames. No public IPs are assigned to cluster nodes	All nodes deployed have public IPs.
Egress traffic	Managed by Azure	Managed by Azure	Managed by Azure
Ingress traffic	Managed by Azure	Managed by Azure	Managed by Azure
Security groups	2 Network Security Groups (Rules set up based on user input/configuration)	2 Network Security Groups (Rules set up based on user input/configuration)	2 Network Security Groups (Rules set up based on user input/configuration)

Identity and access management

For AWS

	Private	Semi-private	Public
Federated access	Cross-account policy	Cross-account policy	Cross-account policy
Storage access	IAM roles, policies, and instance profiles	IAM roles, policies, and instance profiles	IAM roles, policies, and instance profiles

For Azure

	Private	Semi-private	Public
Federated access	Azure service principal	Azure service principal	Azure service principal
Storage access	User-assigned managed identities and role assignments using built-in roles	User-assigned managed identities and role assignments using built-in roles	User-assigned managed identities and role assignments using built-in roles

Storage

For AWS

	Private	Semi-private	Public
S3 buckets	3 base locations	3 base locations	3 base locations

For Azure

	Private	Semi-private	Public

Azure storage accounts	One storage account for data, logs and backup (3 containers) One additional storage account for locally caching VM images	One storage account for data, logs and backup (3 containers) One additional storage account for locally caching VM images	One storage account for data, logs and backup (3 containers) One additional storage account for locally caching VM images

Environment and Data Lake

This section summarizes Cloudera networking, security, and other configurations that are made when Cloudera is deployed based on one of the deployment patterns.

Cloudera networking setup

For AWS

	Private	Semi-private	Public
Communication with Cloudera Control Plane	Reverse HTTPS tunnel (Cluster Connectivity Manager), no private link	Reverse HTTPS tunnel (Cluster Connectivity Manager), no private link	Reverse HTTPS tunnel (Cluster Connectivity Manager), no private link
Load balancer and node placement	2 load balancers are placed in private subnets. All cluster nodes are placed in private subnets.	2 load balancers are placed in the external subnets and all cluster nodes are placed in the internal subnets.	2 load balancers and all cluster nodes are placed in public subnets.
Multiple availability zones	Environment and Data Lake clusters are spread across three availability zones.	Environment and Data Lake clusters are spread across three availability zones.	Environment cluster is spread across three availability zones. Basic Data Lake cluster is deployed in one availability zone.
Ports open (in the external and internal network)	Ports 22 and 443 are open by default.	Ports 22 and 443 are open by default.	Ports 22 and 443 are open by default.

For Azure

	Private	Semi-private	Public
Communication with Cloudera Control Plane	Reverse HTTPS tunnel (Cluster Connectivity Manager), no private link	Reverse HTTPS tunnel (Cluster Connectivity Manager), no private link	Reverse HTTPS tunnel (Cluster Connectivity Manager), no private link
Load balancer and node placement	Azure Standard Network Load Balancers are created by the data lake	Azure Standard Network Load Balancers are created by the data lake	Azure Standard Network Load Balancers are created by the data lake
Availability zones	Zone placement is managed by Azure	Zone placement is managed by Azure	Zone placement is managed by Azure
Ports open (in the external and internal network)	Ports 22 and 443 are open by default.	Ports 22 and 443 are open by default.	Ports 22 and 443 are open by default.

Cloudera security setup

For AWS

	Private	Semi-private	Public
Fine-grained storage access control (RAZ)	Enabled	Enabled	Enabled
SSH access to cluster hosts	Root access is possible with a customer-provided keypair.	Root access is possible with a customer-provided keypair.	Root access is possible with a customer-provided keypair.

For Azure

	Private	Semi-private	Public
Fine-grained storage access control (RAZ)	Enabled	Enabled	Enabled
SSH access to cluster hosts	Root access is possible with a customer-provided keypair.	Root access is possible with a customer-provided keypair.	Root access is possible with a customer-provided keypair.

Cloudera versions and details

For AWS

	Private	Semi-private	Public
Data Lake Runtime version	Latest	Latest	Latest
Data Lake shape	Medium Duty	Medium Duty	Light Duty

For Azure

	Private	Semi-private	Public
Data Lake Runtime version	Latest	Latest	Latest
Data Lake shape	Medium Duty	Medium Duty	Light Duty

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 or Azure 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 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.

Prerequisites

Prior to 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 CDP deployment patterns and deployment pattern definitions described in Creating and managing Cloudera deployments.

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

Deploy Cloudera

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

Step 1: Clone the repository

The cdp-tf-quickstarts repository contains Terraform resource files to quickly deploy Cloudera Public 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 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. Currently AWS and Azure are available.

Next, edit the input variables in the configuration file as required:

For AWS

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

For Azure

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

Sample content of this file, with indicators of values to change are shown below. The variables are explained below 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
```

As an outcome 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 CDP resources
, 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 CDP resources
, 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
```

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: private, semi-private and public.	public

For Azure

Input	Description	Default value
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
env_prefix	A string prefix that will be used to name the cloud provider and Cloudera resources created.	Not set
deployment_template	The selected deployment pattern. Values allowed: private, semi-private and public.	public

The following tables explain the optional inputs. The optional inputs can optionally be added to the configuration file. While the mandatory inputs are present in the configuration file and only their values need to be provided, the optional inputs should 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. 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. 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 crea te_vpc is false	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

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. 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. 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_vnet is false	Empty string
cdp_vnet_name	VNet name for Cloudera environment. Required if create_vnet is false	Empty string
cdp_subnet_names	List of subnet names for Cloudera resources. Required if create_vnet is false	Empty list
cdp_gw_subnet_ids	List of subnet names for Cloudera Gateway. Required if create_vnet is false	Empty list

Step 3: Launch the deployment

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

```
terraform init
terraform apply
```

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

When you are prompted, type yes to tell Terraform to perform the deployment. Typically, this will take 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 progressing. 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 that's 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 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.

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 or Azure
- 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 or Azure

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

For Azure examples see Build Infrastructure - Terraform Azure Example | 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.

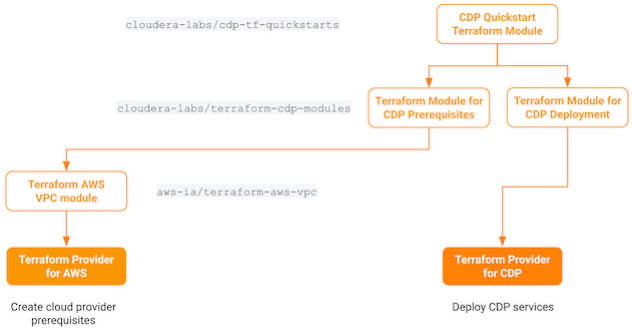
Terraform module for deploying Cloudera

The Terraform Modules for Cloudera Prerequisites on AWS and Azure 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 or Azure, both 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 a number of AWS 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 following diagram illustrates the hierarchy of modules and providers used by the onboarding automation tooling (AWS is used as an example):



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