Cloudera on premises / CDP Private Cloud (PvC)

Installation & Setup

:: Deployment Guide ::

Published: September 2025

By: **Kuldeep Sahu**, Partner Solutions Engineer, Cloudera Inc.

Audience

The intended audience for this document includes, but is not limited to, sales engineers, field consultants, professional services, IT managers, IT engineers, partners, and customers who are interested in learning about and deploying the Cloudera Data Platform Private Cloud (Cloudera on premises) on the bare metal/ virtual machines for digital transformation through cloud-native modern data analytics and AI/ML.

Purpose of this Document

This document describes the architecture, installation, configuration, and validated use cases for the on premise platforms using Cloudera Data Platform Private Cloud Base on bare metal based servers or virtual machines. A reference architecture is provided to configure the Cloudera Data Platform on VMs.

****

Table of Contents

Author History

Name	Version	Date
Kuldeep Sahu	1.0	23-May-2024

****

Cloudera on premises Installation and Configuration on on-premises Infrastructure:

This document provides all the required information for setup and install Cloudera on premises.

Important URLs:

Install Cloudera on premises Base and Data Service Clusters:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-installation.html
https://docs.cloudera.com/cdp-private-cloud-data-services/latest/installation-ecs/topics/cdppvc-installation-ecs-steps.html

Uninstall and cleanup Cloudera on premises Base, Data Service Clusters and PostgreSQL DB:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-uninstallation.html
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation-ecs/topics/cdppvc-installation-ecs-uninstall-pvc.html
https://kb.objectrocket.com/postgresql/how-to-completely-uninstall-postgresql-757

Internal documentation: Prerequisites list by Dennis Lee and PvC AWS Setup by Puneet Joshi
https://dennislee22.github.io/docs/cdppvc
https://docs.google.com/document/d/1OSKBChSTbc8NhuQ8YXRN-YxFnaVBj47Lz4cWro-zTVs/edit

==========

Introduction:

Cloudera on premises is an integrated analytics and data management platform deployed in private data centers. Cloudera Data Platform is a single platform that has two form factors CDP Public and Cloudera on premises.
It consists of Cloudera on premises Base and Cloudera on premises Data Services and offers broad data analytics and artificial intelligence functionality along with secure user access and data governance features.
Cloudera on premises (PvC) data services components run on containerized clusters and thus require a container orchestration engine to manage all the workloads.

There will be two major components in Cloudera on premises Installation:

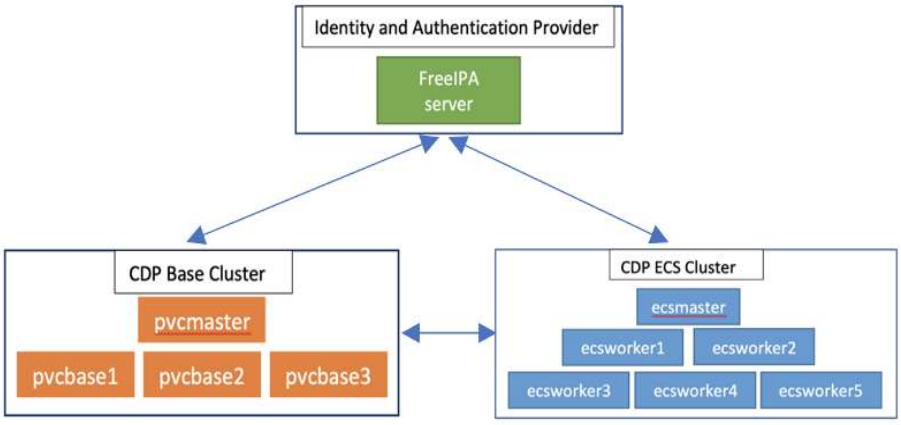
- Cloudera on premises Base Cluster
- Cloudera on premises Data Services Cluster

Cloudera on premises DS offers installation with two orchestration engines.

- RedHat Openshift Container Platform (OCP)
- Embedded Container Service (Cloudera managed-ECS)

In this document, we focus on Cloudera on premises Data Service Cluster setup with ECS.

Cloudera on premises setup consists of the following three parts.



- **FreeIPA server:-** It provides the Identity and Authentication to the cluster. It includes Kerberos as the authentication provider and LDAP as directory service provider. All the cluster nodes (both Base and ECS) act as FreeIPA agents. (FreeIPA server includes Private DNS Server, MIT Kerberos KDC, Directory Server, Chrony, Dogtag certificate system, SSSD)
- **CDP Base Cluster:**- It consists of all the prerequisite services that form the basis for CDP Data Lake for Data Services.
    - Atlas
    - Solr
    - HBase
    - HDFS
    - Hive (Metastore Server)
    - Hive-on-Tez (HiveServer2)
    - Hue (Required for CDW data service)
    - Iceberg Replication
    - Impala(Used as Client)
    - Kafka
    - Ozone (Required for CDE data service)
    - Phoenix
    - Ranger
    - Spark on YARN (Spark 2)
    - Spark 3
    - Tez
    - YARN
    - Yarn Queue Manager (Optional)
    - ZooKeeper

    Missing Components which we aren't considering as part of this setup guide but can be installed additionally, if needed:

    - Flink
    - Knox
    - Kudu
    - Livy
    - Nifi

- Nifi Registry
- QueueManager
- S3 Connecter
- Zeppelin

- **CDP ECS Data Services Cluster:-** This is the Rancher (RKE) based Kubernetes cluster that forms the basis for all the containerized workloads of CDP Data Services. It consists of ECS master, ECS agents, and Docker servers.

- **Service Dependencies:**

Service	Dependencies
Atlas	• HDFS • HBase • Kafka (Kafka broker role only) • Solr
HBase	• HDFS • ZooKeeper
HDFS	• Hive • Spark • Yarn
Hive	• HDFS • YARN • Tez
Hive-on-Tez	• HDFS • YARN • Hive • Tez
Hue	• HDFS • Hive
Impala	• HDFS • Hive
Kafka	ZooKeeper
Livy	• Spark • Impala

Service	Dependencies
Oozie	• YARN
Ozone	-
Ranger	• HDFS • Solr • Atlas
Solr	• HDFS • ZooKeeper
Spark on YARN	• YARN
Streams Messaging Manager	• Kafka
Streams Replication Manager	• Kafka
Tez	• YARN
YARN	• HDFS • ZooKeeper
Zeppelin	• HDFS • Spark-on-YARN • YARN
ZooKeeper	-

Let's have a look at the prerequisites before proceeding with the actual setup.

****

Solution Summary

This RA document details the process of installing Cloudera on premises on bare metal or VM based servers and configuration details of fully tested and validated workloads in the cluster.

Prerequisites:

Entitlements

Your License key must have the PvC DS entitlement. A current key without the entitlement will block access to ECS bits. Please raise a ticket or reach out to the Cloudera POC to get the necessary entitlements.

Virtual Machines

Administrator access to virtual machines.

Infrastructure Setup: Hardware and Software Requirements

Below table summarizes the machines used for this POC. This is a minimum requirement, One can increase the number of machines to achieve High Availability and Fault Tolerance. If this cluster is not meant to perform any benchmarking or performance test, one can proceed ahead with this infrastructure.

Note: ***The cluster configurations used in this document are designed and decided considering the installation/configuration and management of all 3 data services' i.e. CAI, CDW, CDE. with minimalistic workloads on a single ECS data services cluster for the PoC purpose. The hardware specs should be redetermined and recalculated for the clusters to set up for a different purpose from above mentioned.***

Note: ***Screenshots shown and versions used in this document are just for the reference purpose and may differ from the version to version used.***

Table 1.

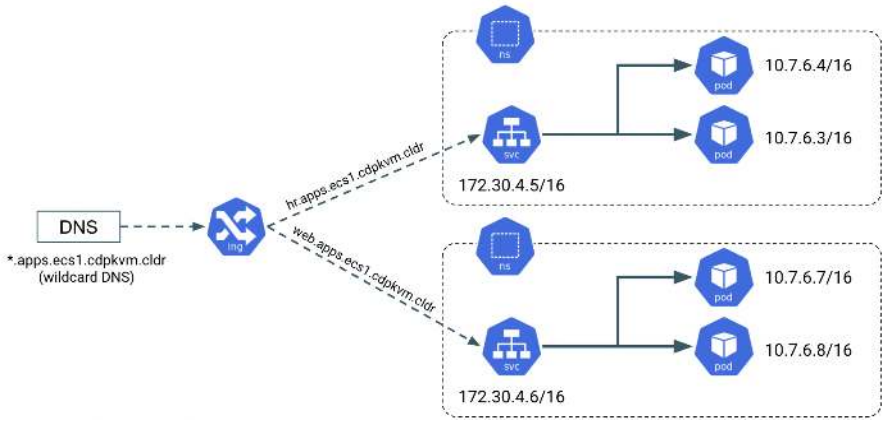
Count	CDP Role
1	FreeIPA Server (Will be used for FreeIPA, Kerberos, Private DNS, LDAP, NTP, KDC, and will be used as an ansible controller node for automation purpose)
1	Cloudera-Manager Server (with external PostgreSQL Database server, will be used for downloading bits as well)
4	CDP Base Cluster (1 Master and 3 Worker Nodes)
3-11	CDP ECS Data Services Cluster (1 Master and 3-10 Worker Nodes)

DNS Server (In case we are not going with FreeIPA)

An external DNS server must contain the forward and reverse zones of the company domain name. The external DNS server must be able to resolve the hostname of all Cloudera on premises hosts and the 3rd party components (includes Kerberos, LDAP server, external database, NFS server) and perform reverse DNS lookup.

Wildcard DNS entry must be configured; e.g. ***.apps.cldrsetup.local**. This helps to reduce Day-2 operational tasks to set separate DNS entries for each newly provisioned external-facing application/service.

The external DNS server is expected to be ready prior to installing the Cloudera on premises solution and its installation procedure is not covered in this document.



Kerberos + LDAP Server/AD + Certificate (Required only, in case we are not going with FreeIPA)

An external Kerberos server and the Kerberos key distribution center (KDC) (with a realm established) must be available to provide authentication to CDP services, users and hosts.

An external secured LDAP-compliant identity/directory server (LDAPS) is required to enable the Cloudera on premises solution to look up for the user accounts and groups in the directory. This is expected to be ready prior to installing the Cloudera on premises solution and its installation procedure is not covered in this document.

Auto-TLS should be enabled using certificates created and managed by a Cloudera Manager certificate authority (CA), or certificates signed by a trusted public CA or your own internal CA. Prepare the certificate of your choice.

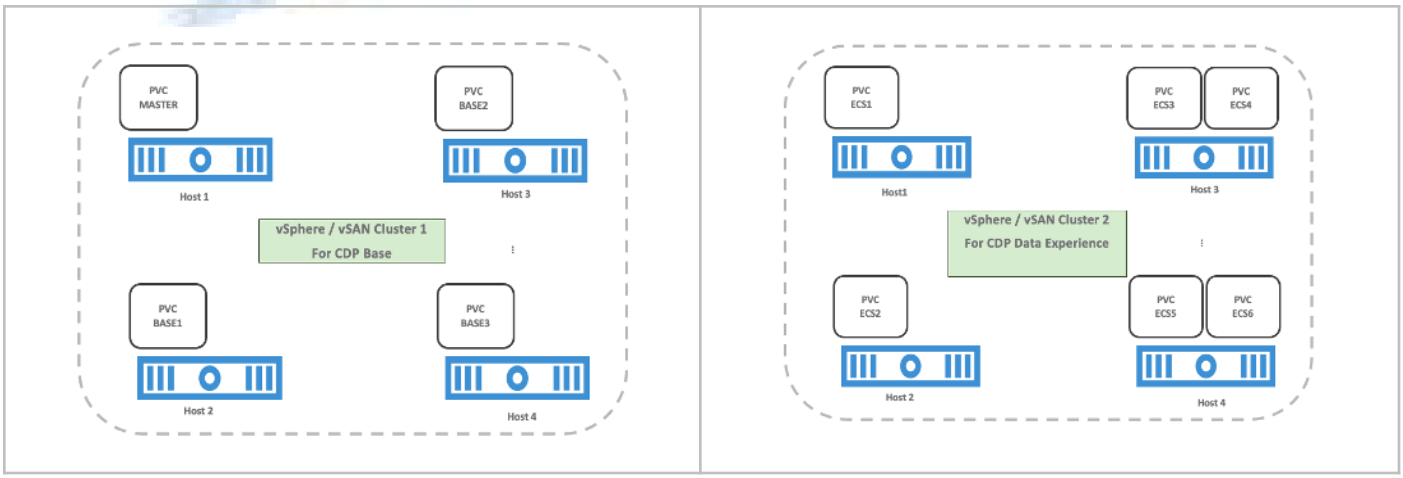
The total number of CA certificates must not exceed 10. Otherwise, pods will be evicted during initialization due to limited memory (1Gi) to process the configmap file.

External NFS (Preferable but optional; needed for CAI use case)

CAI requires an external NFS server to store the project files and directories. NFS version 4.1 must be supported.

The external NFS storage is expected to be ready prior to installing the Cloudera on premises solution. External NFS storage installation is not covered in this document.

This document covers the Cloudera on premises setup and testing of the Data Services.



****

Hardware Requirements

**Hardware specs e.g. CPU, memory, disk, etc. should be analyzed and re-determined as per the setup requirement e.g. POC, demo, HA, DR etc. Current setup is for POC/Demo purpose only.
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-requirements-supported-versions.html

Reverse Proxy Server: (Optional: For external URLs, best practice perspective)

Role	HostName	CPU	RAM	Disk	Partitions
reverse proxy	proxy	8	16GB	OS disk (100GB)	NA

FreeIPA/Kerberos & Private DNS Server: (In case we are not going with FreeIPA, External Kerberos/KDC is Reqd.)

Role	HostName	CPU	RAM	Disk	Partitions
ipaserver+ansible-controller	ipaserver	16	32GB	OS disk (250GB)	root partition
cldr-mngr, postgres db, bits	cldr-mngr	32	64GB	1.2TB	root partition, /var=600GB /opt=600GB

Note: Each of the nodes in the below configurations require a dedicated minimum allocation of 450GB each for /docker and /var, important to consider if dedicated mounts(disks) are used.

PvC Base Cluster we will be installing a 4-node cluster on VMs:

** Here BaseMaster Node will also host Gateway and Utility hosts' services as per public documentation at https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-runtime-cluster-hosts-role-assignments.html
** The Role assignment strategy for Control Plane Services' (e.g. HDFS, YARN, Spark, etc.) is discussed in the later steps of PvC Base Cluster Setup.

Role	HostName	CPU	RAM	Disk	Partitions
BASE CLUSTER					
Base-Master	pvcbase-master	32	64GB	root partition (600GB)	/hdfs /opt /var /yarn
Base-Worker	pvcbase-worker1	32	64GB	root partition (600GB)	/hdfs /opt /var /yarn
Base-Worker	pvcbase-worker2	32	64GB	root partition (600GB)	/hdfs /opt /var /yarn
Base-Worker	pvcbase-worker3	32	64GB	root partition (600GB)	/hdfs /opt /var /yarn

PvC Data Service (ECS) Cluster with CDW: we will be installing a 4-node cluster on VMs:

** Specs upgraded for concurrent tests and higher data volume tests and assumes only CDW services will be deployed
** Assuming Specs for 1 CDW Data Catalog, 1 CDV (DataViz) Small Instance, 1 Hive LLAP and 1 Impala Virtual Warehouse each with 1 coordinator and 2 executors.

Role	HostName	CPU	RAM	Disk	Partitions
ECS DS CLUSTER	**CDW**				
ECS-Master	pvcecs-master	32	128GB	root partition + 2TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker1	32	256GB	root partition + 2TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker2	32	256GB	root partition + 2TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker3	32	256GB	root partition + 2TB	/cdwdata, /docker, /lhdata, /var

PvC Data Service (ECS) Cluster with CAI: we will be installing a 3-node cluster on VMs:

** Specs upgraded for concurrent tests and higher data volume tests and assumes only CAI services will be deployed

** Assuming Specs for 1 CAI Workbench with 10 small and 2 Average sized CAI Concurrent Sessions.

Role	HostName	CPU	RAM	Disk	Partitions
ECS DS CLUSTER	**CAI**				
ECS-Master	pvcecs-master	32	128GB	root partition + 2 TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker1	32	128GB	root partition + 2 TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker2	32	128GB	root partition + 2 TB	/docker, /lhdata, /var

PvC Data Service (ECS) Cluster with CDE: we will be installing a 6-node cluster on VMs:

** Specs upgraded for concurrent tests and higher data volume tests and assumes only CDE services will be deployed.

** Assuming Specs for 1 CDE Virtual service along with 1 Virtual Cluster with 5 small and 2 Average sized CDE Concurrent Jobs.

Role	HostName	CPU	RAM	Disk	Partitions
ECS DS CLUSTER	**CDE**				
ECS-Master	pvcecs-master	32	64GB	root partition + 2TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker1	32	64GB	root partition + 2TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker2	32	64GB	root partition + 2TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker3	32	64GB	root partition + 2TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker4	32	64GB	root partition + 2TB	/docker, /lhdata, /var
ECS-Worker	pvcecs-worker5	32	64GB	root partition + 2TB	/docker, /lhdata, /var

PvC Data Service(ECS) Cluster with CDW+CDE+CAI:we will be installing a 11-node cluster on VMs:

** Specs upgraded for concurrent tests and higher data volume tests and assumes all 3 services will be deployed (CDW, CDE, CAI)

** Assuming Specs for 1 CDW Data Catalog, 1 CDV (DataViz Small) Instance, 1 Hive LLAP and 1 Impala Virtual Warehouse each with 1 coordinator and 2 executors.

** Assuming Specs for 1 CAI Workbench with 10 small and 2 Average sized CAI Concurrent Sessions.

** Assuming Specs for 1 CDE Virtual service along with 1 Virtual Cluster with 5 small and 2 Average sized CDE Concurrent Jobs.

Role	HostName	CPU	RAM	Disk	Partitions
ECS DS CLUSTER	**CAI+CDW+CDE**				
ECS-Master	pvcecs-master	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker1	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker2	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker3	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker4	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker5	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker6	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker7	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker8	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker9	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var
ECS-Worker	pvcecs-worker10	32	128GB	root partition + 1.8TB	/cdwdata, /docker, /lhdata, /var

Reference Architecture

Data Lake (Cloudera on premises Base) Reference Architecture

- Cloudera Data Platform: Cloudera on premises Base **7.3.1.400 SP2**
- Cloudera Data Platform: Cloudera on premises Data Services **1.5.5 CHF1**

This RA document explains the architecture and deployment procedures for Cloudera Data Platform Cloudera on premises on cluster using on premise Infrastructure for Big Data and Analytics. The solution provides the details to configure Cloudera on premises on the bare metal RHEL9 based infrastructure.

****

Software Requirements

Note: This document is written for the below specified versions and commands to be executed are specific for those versions. If you are planning to use some different versions, commands may need to be updated separately.

Table 1 lists the software components and the versions required for a single cluster of the Servers running in on premise, as tested, and validated in this document.

Below table summarizes the list of softwares/packages and their use for setting up Cloudera on premises cluster.

Table 2. Software Distributions and Firmware Versions

Software Component	Version or Release	Host to be Installed
OS: Red Hat Enterprise Linux Server (RHEL)	9.x	All Servers
OpenJDK	17.0.14.0.7-1 >=	All Servers
Python3	3.9.22 >=	All Servers
PostgreSQL DB	17 >=	Cldr-Mngr
Psycopg2-binary	2.9.10 >=	All Servers
Postgres-JDBC-Connector	42.7.7 >=	All Servers
Cloudera Manager	7.13.1-CHF4 **(7.13.1.400-68000784)**	Cldr-Mngr
Cloudera on premises Base (RunTime)	7.3.1.400 SP2 **(7.3.1-1.cdh7.3.1.p400.67986116)**	PvC Base Cluster Nodes
Cloudera on premises Data Services	1.5.5-CHF1 **(1.5.5-h2-b10)**	PvC Data Service Cluster Nodes
Hadoop (Includes YARN and HDFS)	3.1.1.7.3.1.400-100	PvC Base Cluster Nodes
Spark3	3.5.4.7.3.1.400-100	PvC Base Cluster Nodes
Ozone	1.4.0.7.3.1.100-39	PvC Base Cluster Nodes
FreeIPA Server	Latest: 4.12.4	IPA server node
FreeIPA Client	Latest: 4.12.4	All nodes except ipaserver
NFS Utility Package	Latest: 2.8.3	PvC Data Service Cluster Nodes
TLS	AutoTLS (Self-signed)	ipa server node
Kerberos + LDAP(IdP) + DNS	FreeIPA	ipa server node
Data Lake Storage	HDFS(All) Ozone Iceberg v2 (with HDFS & Ozone)	
ECS DB Configuration	Embedded	ECS Only
Vault	Embedded	ECS Only
Docker Registry Type	Embedded/ Cloudera Default	ECS Only
NFS for CAI	Embedded i.e Internal	ECS Only

Note: Please check the Cloudera on premises requirements and supported versions for information about hardware, operating system, and database requirements, as well as product compatibility matrices, here: https://supportmatrix.cloudera.com/ and here:

https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/index.html

Note: For Cloudera on premises Base and Experiences versions and supported features, go to: https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/private-release-notes/topics/rt-runtime-component-versions.html

Note: For Cloudera on premises Base requirements and supported version, go to: https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-requirements-supported-versions.html

Note: Dedicated ***NVMe/SSD drives*** are recommended to store ***Ozone metadata, Ozone mgmt*** configuration for the admin/mgmt. nodes and worker/data nodes and ***CDW data service storage*** for virtual warehouses for local attached Storage Tiering Cache.

Summary

The below table contains the names assigned to the VM instances and to some other required components. Going forward in this document will refer to them by name.

Note: The domain name, and the hostnames mentioned here are just for reference. You may choose to have the hostnames as per your requirements.

Table 3.

NodeName	Details
pvcbase-master	Cloudera on premises Base Master
pvcbase-worker1 to pvcbase-worker3	CDP Base Cluster Worker Nodes
ipaserver **(OR Existing LDAP/AD + DNS + Kerberos + KDC)**	FreeIPA Server
cldr-mngr	Cloudera-Manager and PostgreSQL DB Server
pvcecs-master	ECS Master Node
pvcecs-worker1 to pvcecs-worker10	ECS Worker Nodes
CLDRSETUP.LOCAL **(Replace with your ORG DOMAIN)**	Dummy Domain For POC Purpose

Once you have familiarized yourself with all the information mentioned above, you can start with the preliminary work for CDP Base Cluster setup.

****

Post OS Installation

Preliminary Work

Before getting into the actual installation of Cloudera on premises Base & Data Services clusters, we need to prepare our machines and perform some steps to meet the prerequisites.

Choose one of the nodes of the cluster or a separate node as the Ansible Admin/Controller Node for management. In this document, we configured the ipaserver for this purpose.

Procedure 1. Configure individual servers' static hostnames and prepare /etc/hosts file

Step 1. Setup the hostname ***for each individual node*** by logging in using the IP addresses provided by the infrastructure team, so we can refer to them with names instead of IP addresses for simplicity and ease of identification. While you are configuring the hostname, also follow **Step 2** while logging in to each host. **Replace your ORG DOMAIN**

```
[root@ipaserver ~]# sudo hostnamectl set-hostname --static ipaserver.cldrsetup.local
[root@cldr-mngr ~]# sudo hostnamectl set-hostname --static cldr-mngr.cldrsetup.local

[root@pvcbase-master ~]# sudo hostnamectl set-hostname --static pvcbase-master.cldrsetup.local
[root@pvcbase-worker1 ~]# sudo hostnamectl set-hostname --static pvcbase-worker1.cldrsetup.local
[root@pvcbase-worker2 ~]# sudo hostnamectl set-hostname --static pvcbase-worker2.cldrsetup.local
[root@pvcbase-worker3 ~]# sudo hostnamectl set-hostname --static pvcbase-worker3.cldrsetup.local

[root@pvcecs-master ~]# sudo hostnamectl set-hostname --static pvcecs-master.cldrsetup.local
[root@pvcecs-worker1 ~]# sudo hostnamectl set-hostname --static pvcecs-worker1.cldrsetup.local
[root@pvcecs-worker2 ~]# sudo hostnamectl set-hostname --static pvcecs-worker2.cldrsetup.local
[root@pvcecs-worker3 ~]# sudo hostnamectl set-hostname --static pvcecs-worker3.cldrsetup.local
[root@pvcecs-worker4 ~]# sudo hostnamectl set-hostname --static pvcecs-worker4.cldrsetup.local
[root@pvcecs-worker5 ~]# sudo hostnamectl set-hostname --static pvcecs-worker5.cldrsetup.local
[root@pvcecs-worker6 ~]# sudo hostnamectl set-hostname --static pvcecs-worker6.cldrsetup.local
[root@pvcecs-worker7 ~]# sudo hostnamectl set-hostname --static pvcecs-worker7.cldrsetup.local
[root@pvcecs-worker8 ~]# sudo hostnamectl set-hostname --static pvcecs-worker8.cldrsetup.local
[root@pvcecs-worker9 ~]# sudo hostnamectl set-hostname --static pvcecs-worker9.cldrsetup.local
[root@pvcecs-worker10 ~]# sudo hostnamectl set-hostname --static pvcecs-worker10.cldrsetup.local
```

Step 2. While you set the hostnames by logging in to each individual hosts, make sure to run the dnf update and install python3 dependencies as well, since these are fresh nodes:

** Python3 can be installed manually on bare minimum (ipaserver/ansible admin) and can be later installed using ansible on the rest of the nodes. (Only, If you don't want it to install on each individual node)

```
[root@ipaserver ~]# sudo dnf -y update
[root@ipaserver ~]# sudo dnf -y install wget telnet net-tools bind-utils iproute traceroute nc

##### Verify If Python3 and Pip3 are already installed.
[root@ipaserver ~]# python3 --version
[root@ipaserver ~]# pip3 --version

##### Install python3.9 - ON ALL NODES (if not Present already):
##### Check if Python could be installed using dnf command directly:
[root@ipaserver ~]# sudo dnf -y install python39 python3-pip
##### If Python could not be installed using dnf command directly:
[root@ipaserver ~]# sudo dnf -y groupinstall "Development Tools"
[root@ipaserver ~]# sudo dnf -y install epel-release openssl-devel bzip2-devel libffi-devel xz-devel
[root@ipaserver ~]# VERSION=3.9.22
[root@ipaserver ~]# wget https://www.python.org/ftp/python/$VERSION/Python-$VERSION.tgz
[root@ipaserver ~]# tar xvf Python-$VERSION.tgz
[root@ipaserver ~]# cd Python-$VERSION/
[root@ipaserver ~]# ./configure --enable-optimizations
[root@ipaserver ~]# sudo make altinstall
[root@ipaserver ~]# python3 --version
[root@ipaserver ~]# cd -
[root@ipaserver ~]# rm -rvf Python-$VERSION/ Python-$VERSION.tgz

##### Install pip3 - ON ALL NODES (if not Present already):
```

```
[root@ipaserver ~]# dnf install -y python3-pip
[root@ipaserver ~]# pip3 install --upgrade pip
[root@ipaserver ~]# pip3 --version
[root@ipaserver ~]# pip3 install psycopg2-binary

##### Verify Python and Pip Versions
[root@ipaserver ~]# python3 -V
Python 3.9.22
[root@ipaserver ~]# pip3 --version
```

Step 3. Log into the ipaserver Node using IP provided previously by the infrastructure team.

```
[root@ipaserver ~]# ssh 172.31.24.240
```

Step 4. Setup /etc/hosts on the ipaserver node; this is a pre-configuration to setup Private DNS as shown in the next section. In large scale production grade deployment, DNS server setup is highly recommended.

Populate the host file with IP addresses and corresponding hostnames on the ipaserver node by taking the private IP of machine and add an entry in /etc/hosts file as follows: ***(All of below mentioned IPs are private IP addresses)***

(We will later copy the same hosts file to all other nodes with the help of ansible)

```
[root@ipaserver ~]# sudo vi /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

# Free-IPA Server
172.31.24.240 ipaserver.cldrsetup.local ipaserver

# Cloudera Manager Server
172.31.1.38 cldr-mngr.cldrsetup.local cldr-mngr
172.31.1.38 postgresdb.cldrsetup.local postgresdb

# PvC Base Cluster Nodes
172.31.1.34 pvcbase-master.cldrsetup.local pvcbase-master
172.31.1.35 pvcbase-worker1.cldrsetup.local pvcbase-worker1
172.31.1.36 pvcbase-worker2.cldrsetup.local pvcbase-worker2
172.31.1.37 pvcbase-worker3.cldrsetup.local pvcbase-worker3

# PvC Data Services ECS Cluster Nodes
172.31.30.239 pvcecs-master.cldrsetup.local pvcecs-master
172.31.22.43  pvcecs-worker1.cldrsetup.local pvcecs-worker1
172.31.30.249 pvcecs-worker2.cldrsetup.local pvcecs-worker2
172.31.26.24  pvcecs-worker3.cldrsetup.local pvcecs-worker3
172.31.24.198 pvcecs-worker4.cldrsetup.local pvcecs-worker4
172.31.24.53  pvcecs-worker5.cldrsetup.local pvcecs-worker5
172.31.22.44  pvcecs-worker6.cldrsetup.local pvcecs-worker6
172.31.30.250 pvcecs-worker7.cldrsetup.local pvcecs-worker7
172.31.26.25  pvcecs-worker8.cldrsetup.local pvcecs-worker8
172.31.24.199 pvcecs-worker9.cldrsetup.local pvcecs-worker9
172.31.24.54  pvcecs-worker10.cldrsetup.local pvcecs-worker10
```

Step 5. Perform the basic validation of OS version and hostname/IP configurations:

```
## Ensure that the OS version is RHEL 9.x.
## To verify the version, run the below command. It should return RedHat Linux version 9.x.

[root@ipaserver ~]# cat /etc/*rel* |grep -E 'NAME|VERSION'
NAME="Red Hat Enterprise Linux"
VERSION="9.4 (Plow)"
VERSION_ID="9.4"
PRETTY_NAME="Red Hat Enterprise Linux 9.4 (Plow)"
CPE_NAME="cpe:/o:redhat:enterprise_linux:9::baseos"
REDHAT_BUGZILLA_PRODUCT_VERSION=9.4
REDHAT_SUPPORT_PRODUCT_VERSION="9.4"

## Verify Hostname and IP addresses
[root@ipaserver ~]# hostname -f
ipaserver.cldrsetup.local

[root@ipaserver ~]# hostname -i
```

```
172.31.24.240

[root@ipaserver ~]# cat /etc/hostname
ipaserver.cldrsetup.local

[root@ipaserver ~]# ip addr show eth0 | grep -e inet
    inet 10.0.2.2/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
    inet6 fe80::c3:16ff:fe00:9/64 scope link noprefixroute

[root@ipaserver ~]# ip addr show eth1 | grep -e inet
    inet 172.31.24.240/24 brd 172.31.1.255 scope global dynamic noprefixroute eth1
    inet6 fe80::65b3:25c6:8b2a:b4ae/64 scope link noprefixroute

[root@ipaserver ~]# ip addr show |grep $(hostname -i)
    inet 172.31.24.240/24 brd 172.31.1.255 scope global dynamic noprefixroute eth1

[root@ipaserver ~]# host -v -t A $(hostname) | grep -A2 ANSWER
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
--
;; ANSWER SECTION:
ipaserver.cldrsetup.local. 1200    IN      A       172.31.24.240

[root@ipaserver ~]#
[root@ipaserver ~]# uname -a
Linux ipaserver.cldrsetup.local 5.14.0-427.26.1.el9_4.x86_64 #1 SMP PREEMPT_DYNAMIC Fri Jul 5 11:34:54
EDT 2024 x86_64 x86_64 x86_64 GNU/Linux
```

Procedure 2. Setup ipaserver (which includes Private DNS Server, MIT Kerberos KDC, Directory Server, Chrony, Dogtag certificate system, SSSD)

Install and Setup of IPA services

In this step, a Private DNS server and other services like KDC, Directory Service will be configured on the ipaserver. Also, please note that the hostnames used in this installation can be modified as per your requirements.

Follow the on screen instructions and provide the inputs for the parameters as per the table below.

Parameter	Value
Server host name [ipaserver.cldrsetup.local]:	ipaserver.cldrsetup.local
Please confirm the domain name [cldrsetup.local]:	cldrsetup.local
Please provide a realm name [CLDRSETUP.LOCAL]:	CLDRSETUP.LOCAL
Directory Manager password:	<Password For Directory Manager> ***(cloudera123)***
Password (confirm):	<Confirm Password> ***(cloudera123)***
IPA admin password:	<Password For IPA Admin> ***(cloudera123)***
Password (confirm):	<Confirm Password> ***(cloudera123)***
Do you want to configure DNS forwarders? [yes]:	<ENTER>
Do you want to search for missing reverse zones?[yes]:	no
NetBIOS domain name [CLDRSETUP]:	CLDRSETUP
Do you want to configure chrony with NTP server or pool address? [no]:	yes
Enter NTP source server addresses separated by comma, or press Enter to skip:	<ENTER>

Parameter	Value
Enter a NTP source pool address, or press Enter to skip:	<ENTER>
Continue to configure the system with these values?[no]:	yes

Please keep the same password for both Directory manager and IPA admin so that there is no confusion in future while using the same. Also, note down the password separately.

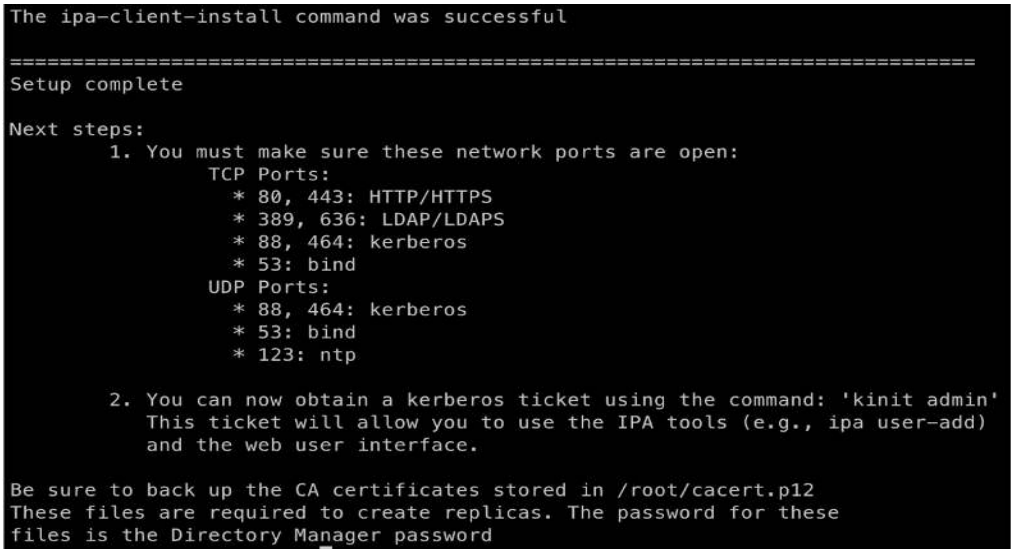
Step 1. Login to IPAServer node and Install ipa-server packages:

```
# Install ipa server dependencies packages through dnf using the below command.
[root@ipaserver ~]# sudo dnf install -y ipa-server bind bind-dyndb-ldap ipa-server-dns firewalld

# If required, use below command to set the java version
[root@ipaserver ~]# update-alternatives --config java

# Configure ipa-server and DNS by using command: ipa-server-install --setup-dns
[root@ipaserver ~]# ipa-server-install --setup-dns
The log file for this installation can be found in /var/log/ipaserver-install.log
==============================================================================
This program will set up the IPA Server.
Version 4.11.0

This includes:
  * Configure a stand-alone CA (dogtag) for certificate management
  * Configure the NTP client (chronyd)
  * Create and configure an instance of Directory Server
  * Create and configure a Kerberos Key Distribution Center (KDC)
  * Configure Apache (httpd)
  * Configure DNS (bind)
  * Configure SID generation
  * Configure the KDC to enable PKINIT

To accept the default shown in brackets, press the Enter key.

Enter the fully qualified domain name of the computer
on which you're setting up server software. Using the form
<hostname>.<domainname>
Example: ipaserver.cldrsetup.local

Server host name [ipaserver.cldrsetup.local]: <ENTER>

Warning: skipping DNS resolution of host ipaserver.cldrsetup.local
The domain name has been determined based on the host name.

Please confirm the domain name [cldrsetup.local]: <ENTER>

The kerberos protocol requires a Realm name to be defined.
This is typically the domain name converted to uppercase.

Please provide a realm name [CLDRSETUP.LOCAL]: <ENTER>
Certain directory server operations require an administrative user.
This user is referred to as the Directory Manager and has full access
to the Directory for system management tasks and will be added to the
instance of directory server created for IPA.
The password must be at least 8 characters long.

Directory Manager password: <cloudera123>
Password (confirm): <cloudera123>

The IPA server requires an administrative user, named 'admin'.
This user is a regular system account used for IPA server administration.

IPA admin password: <cloudera123>
Password (confirm): <cloudera123>

Checking DNS domain cldrsetup.local., please wait ...
Do you want to configure DNS forwarders? [yes]: no
No DNS forwarders configured
Do you want to search for missing reverse zones? [yes]: no
```

```
Trust is configured but no NetBIOS domain name found, setting it now.
Enter the NetBIOS name for the IPA domain.
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.
Example: EXAMPLE.

NetBIOS domain name [CLDRSETUP]: <ENTER>

Do you want to configure chrony with NTP server or pool address? [no]: yes
Enter NTP source server addresses separated by comma, or press Enter to skip: <ENTER>
Enter a NTP source pool address, or press Enter to skip: <ENTER>

The IPA Master Server will be configured with:
Hostname:       ipaserver.cldrsetup.local
IP address(es): 172.31.24.240
Domain name:    cldrsetup.local
Realm name:     CLDRSETUP.LOCAL

The CA will be configured with:
Subject DN:   CN=Certificate Authority,O=CLDRSETUP.LOCAL
Subject base: O=CLDRSETUP.LOCAL
Chaining:     self-signed

BIND DNS server will be configured to serve IPA domain with:
Forwarders:       No forwarders
Forward policy:   only
Reverse zone(s):  No reverse zone

Continue to configure the system with these values? [no]: yes

The following operations may take some minutes to complete.
Please wait until the prompt is returned.

Disabled p11-kit-proxy
Synchronizing time
No SRV records of NTP servers were found and no NTP server or pool address was provided.
Using default chrony configuration.
Attempting to sync time with chronyc.
Time synchronization was successful.
Configuring directory server (dirsrv). Estimated time: 30 seconds
  [1/43]: creating directory server instance
Validate installation settings ...
Create file system structures ...
Perform SELinux labeling ...
Create database backend: dc=cldrsetup,dc=local ...
Perform post-installation tasks ...
  [2/43]: tune ldbm plugin
  [3/43]: adding default schema
------
------
------
------
------
  [6/8]: restarting Directory Server to take MS PAC and LDAP plugins changes into account
  [7/8]: adding fallback group
  [8/8]: adding SIDs to existing users and groups
This step may take a considerable amount of time, please wait..
Done.
Configuring client side components
This program will set up an IPA client.
Version 4.11.0

Using the existing certificate '/etc/ipa/ca.crt'.
Client hostname: ipaserver.cldrsetup.local
Realm: CLDRSETUP.LOCAL
DNS Domain: cldrsetup.local
IPA Server: ipaserver.cldrsetup.local
BaseDN: dc=cldrsetup,dc=local

Configured /etc/sssd/sssd.conf
Systemwide CA database updated.
Adding SSH public key from /etc/ssh/ssh_host_rsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ed25519_key.pub
SSSD enabled
```

```
Configured /etc/openldap/ldap.conf
Configured /etc/ssh/ssh_config
Configured /etc/ssh/sshd_config.d/04-ipa.conf
Configuring cldrsetup.local as NIS domain.
Client configuration complete.
The ipa-client-install command was successful

==============================================================================
Setup complete

Next steps:
        1. You must make sure these network ports are open:
                TCP Ports:
                  * 80, 443: HTTP/HTTPS
                  * 389, 636: LDAP/LDAPS
                  * 88, 464: kerberos
                  * 53: bind
                UDP Ports:
                  * 88, 464: kerberos
                  * 53: bind
                  * 123: ntp

        2. You can now obtain a kerberos ticket using the command: 'kinit admin'
           This ticket will allow you to use the IPA tools (e.g., ipa user-add)
           and the web user interface.

Be sure to back up the CA certificates stored in /root/cacert.p12
These files are required to create replicas. The password for these
files is the Directory Manager password
The ipa-server-install command was successful
[root@ipaserver ~]#

##### Disable the firewall on ipaserver to be able to connect from rest of hosts
[root@ipaserver ~]# systemctl stop firewalld
[root@ipaserver ~]# systemctl disable firewalld
Removed "/etc/systemd/system/multi-user.target.wants/firewalld.service".
Removed "/etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service".
[root@ipaserver ~]#

##### If Fail, do: If the installation fails, then run the below command to uninstall and retry with the above command for installation.
[root@ipaserver ~]# ipa-server-install --uninstall
[root@ipaserver ~]# ipa-server-install --setup-dns (again)
```

The setup will take 10-15 Minutes. If everything goes fine then you should get an output similar to the below screenshot.



Step 2. Verify KDC setup: kerberos ticket is working fine by generating a ticket for the admin user.

```
##### Run the kinit admin command to authenticate as admin and enter the directory password provided during ipa server installation. The command should generate the ticket and should be listed by executing klist -e.
```

```
[root@ipaserver ~]# kinit admin
Password for admin@CLDRSETUP.LOCAL: <cloudera123>

[root@ipaserver ~]# klist -e
Ticket cache: KCM:0
Default principal: admin@CLDRSETUP.LOCAL

Valid starting       Expires              Service principal
05/13/2024 04:07:15  05/14/2024 03:30:48  krbtgt/CLDRSETUP.LOCAL@CLDRSETUP.LOCAL
        Etype (skey, tkt): aes256-cts-hmac-sha384-192, aes256-cts-hmac-sha384-192

##### try kinit admin@CLDRSETUP.LOCAL
##### (if fails anytime, run below commands)
[root@ipaserver ~]# ipactl stop && ipactl start && ipactl status

##### Verify the status of ipa services installed

[root@ipaserver ~]# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
named Service: RUNNING
httpd Service: RUNNING
ipa-custodia Service: RUNNING
pki-tomcatd Service: RUNNING
ipa-otpd Service: RUNNING
ipa-dnskeysyncd Service: RUNNING
ipa: INFO: The ipactl command was successful
```

This command should return the below output:

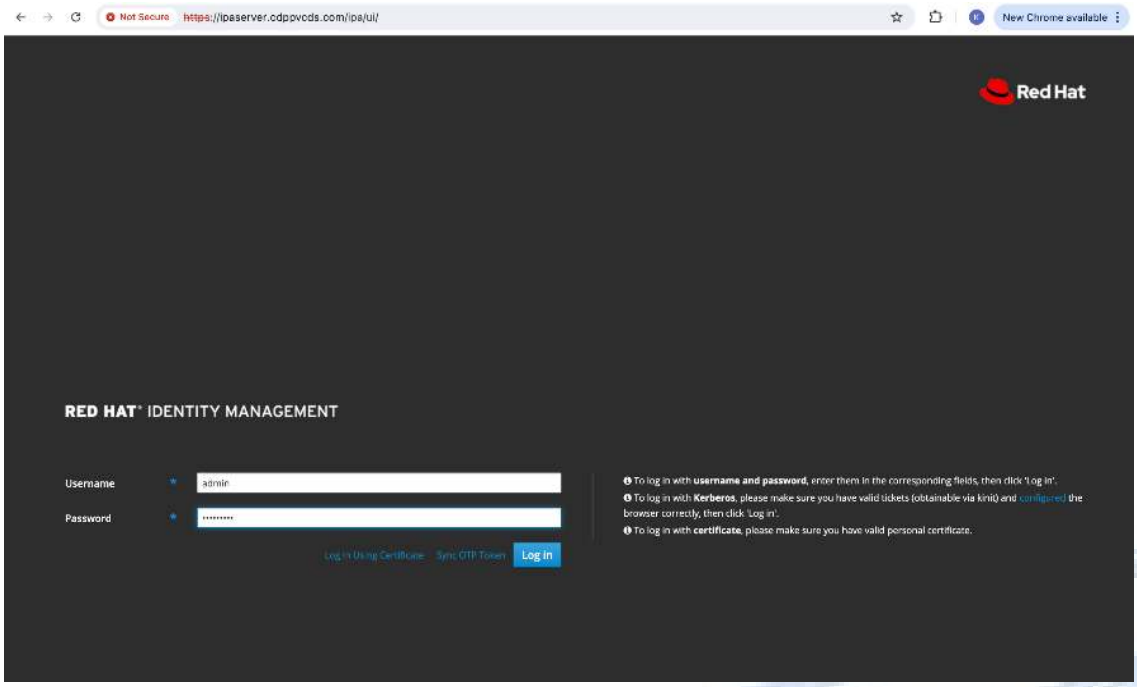
```
[root@ipaserver centos]# kinit admin
Password for admin@CDPPVCDS.COM:
[root@ipaserver centos]# klist -e
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: admin@CDPPVCDS.COM

Valid starting     Expires            Service principal
04/05/23 10:49:29  04/06/23 10:49:26  krbtgt/CDPPVCDS.COM@CDPPVCDS.COM
        Etype (skey, tkt): aes256-cts-hmac-sha1-96, aes256-cts-hmac-sha1-96
[root@ipaserver centos]#
```

Step 3. (Optional) Access WebUI for IPAServer Administration:

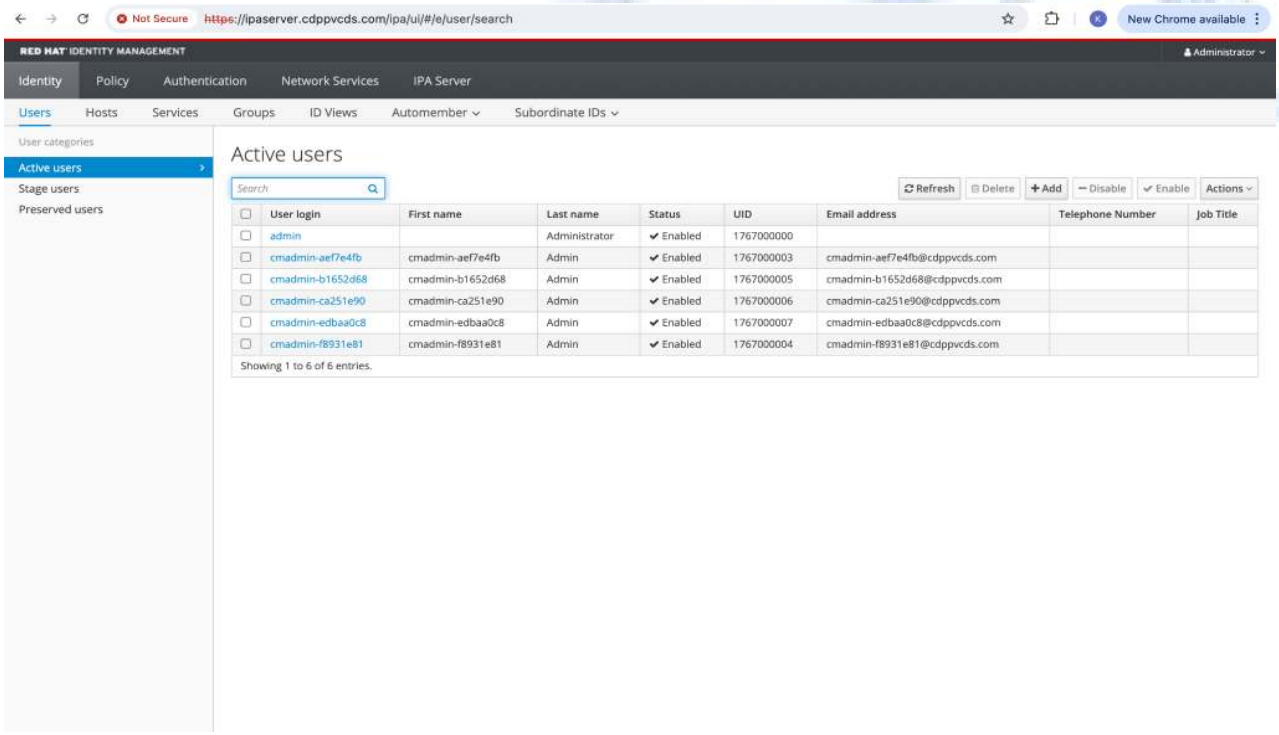
Note: Once the FreeIPA server is successfully installed, the FreeIPA Web UI is automatically set up as part of the installation process. You can access the Web UI directly by putting either the hostname or the IP address of the IPA server into your browser (You'll need to add the ***hostname<->IP*** mapping entry to your Laptop's /etc/hosts file:

```
##### Add IPAserver IP address mapping to your local system's (Laptop) /etc/hosts file, similar to as below.
ksahu@Kuldeeps-MacBook-Air % sudo vi /etc/hosts
35.83.155.109 ipaserver.cldrsetup.local ipaserver

##### Access the below URL on browser, and the IPA Admin console will open.
https://ipaserver.cldrsetup.local/ipa/ui/
```

Step 4. (Optional) You will see below WebUI for IPAServer Administration. Enter the same admin credentials set up for Kerberos KDC authentication: ***(i.e. admin/cloudera123)***



Step 5. (Optional) Below management console will get appear after the successful authentication:



Procedure 3. Set Up Password-less Login

To manage all the nodes in a cluster from the admin/controller node, password-less login needs to be set up. It assists in automating common tasks with Ansible, and shell-scripts without having to use passwords.

Enable the passwordless login across all the nodes when Red Hat Linux is installed across all the nodes in the cluster.

Step 1. Log into the ipaserver Node.

```
[root@ipaserver ~]# ssh 172.31.24.240
```

Step 2. Run the ssh-keygen command to create both public and private SSH key-pair on the ansible-controller node.

```
[root@ipaserver ~]# ssh-keygen -N '' -f ~/.ssh/id_rsa
[root@ipaserver ~]# ls -l /root/.ssh
[root@ipaserver ~]# chmod 600 /root/.ssh/id_rsa
```

```
❯ ssh-keygen -t rsa  -f /root/.ssh/id_rsa_new
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
```

Step 3. Run the following command from the ansible-controller/ipaserver node to copy the public key id_rsa.pub to all the nodes of the cluster. ssh-copy-id appends the keys to the remote-hosts .ssh/authorized_keys.

(NA in AWS EC2, need to be done manually, as right now password based authentication for non-root users is disabled)

```
[root@ipaserver ~]# for i in {1}; do echo "copying ipaserver.cldrsetup.local"; ssh-copy-id -i
~/.ssh/id_rsa.pub root@ipaserver.cldrsetup.local; done;
[root@ipaserver ~]# for i in {1}; do echo "copying cldr-mngr.cldrsetup.local"; ssh-copy-id -i
~/.ssh/id_rsa.pub root@cldr-mngr.cldrsetup.local; done;
[root@ipaserver ~]# for i in {1}; do echo "copying pvcbase-master.cldrsetup.local"; ssh-copy-id -i
~/.ssh/id_rsa.pub root@pvcbase-master.cldrsetup.local; done;
[root@ipaserver ~]# for i in {1..3}; do echo "copying pvcbase-worker$i.cldrsetup.local"; ssh-copy-id -i
~/.ssh/id_rsa.pub root@pvcbase-worker$i.cldrsetup.local; done;
[root@ipaserver ~]# for i in {1}; do echo "copying pvcecs-master.cldrsetup.local"; ssh-copy-id -i
~/.ssh/id_rsa.pub root@pvcecs-master.cldrsetup.local; done;
[root@ipaserver ~]# for i in {1..10}; do echo "copying pvcecs-worker$i.cldrsetup.local"; ssh-copy-id -i
~/.ssh/id_rsa.pub root@pvcecs-worker$i.cldrsetup.local; done;

##### Alternate way is to add pub key to authorized_keys file manually on ipaserver node and copy the entire
.ssh directory to all other NODES; otherwise login into each hosts and manually update authorized_keys:

[root@ipaserver ~]# cat /root/.ssh/id_rsa.pub >> /root/.ssh/authorized_keys

##### copy the entire .ssh directory to all NODES
[root@ipaserver ~]# scp -r /root/.ssh root@cldr-mngr.cldrsetup.local:/root/.
##### (provide root user password when prompted)

##### Download the id_rsa and id_rsa.pub to your local machine by either using scp or sftp (as it will be
required later)
```

Step 4. Enter yes for ***Are you sure you want to continue connecting (yes/no)***?

Step 5. Enter the password of the remote host.

Procedure 4. Set up Ansible (We will be using ipaserver as ansible controller/admin node)

Step 1. Login to IPAServer node and Install ansible-core

```
[root@ipaserver ~]# dnf install -y ansible-core
[root@ipaserver ~]# ansible --version
ansible [core 2.14.14]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.9/site-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /bin/ansible
  python version = 3.9.22 (main, Jan 24 2024, 00:00:00) [GCC 11.4.1 20231218 (Red Hat 11.4.1-3)]
(/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
[root@ipaserver ~]# echo "export ANSIBLE_HOST_KEY_CHECKING=False" >> ~/.bashrc && source ~/.bashrc
```

Step 2. Prepare the host inventory file for Ansible as shown below. Various host groups have been created based on any specific installation requirements of certain hosts.

```
[root@ipaserver ~]# vi /etc/ansible/hosts

[admin]
ipaserver.cldrsetup.local

[ipaserver]
ipaserver.cldrsetup.local

[cldr-mngr]
cldr-mngr.cldrsetup.local

[namenodes]
pvcbase-master.cldrsetup.local

[datanodes]
pvcbase-worker1.cldrsetup.local
pvcbase-worker2.cldrsetup.local
pvcbase-worker3.cldrsetup.local

[ecsmasternodes]
pvcecs-master.cldrsetup.local

[ecsnodes]
pvcecs-worker1.cldrsetup.local
pvcecs-worker2.cldrsetup.local
pvcecs-worker3.cldrsetup.local
pvcecs-worker4.cldrsetup.local
pvcecs-worker5.cldrsetup.local
pvcecs-worker6.cldrsetup.local
pvcecs-worker7.cldrsetup.local
pvcecs-worker8.cldrsetup.local
pvcecs-worker9.cldrsetup.local
pvcecs-worker10.cldrsetup.local

[nodes]
pvcbase-master.cldrsetup.local
pvcbase-worker1.cldrsetup.local
pvcbase-worker2.cldrsetup.local
pvcbase-worker3.cldrsetup.local
pvcecs-master.cldrsetup.local
pvcecs-worker1.cldrsetup.local
pvcecs-worker2.cldrsetup.local
pvcecs-worker3.cldrsetup.local
pvcecs-worker4.cldrsetup.local
pvcecs-worker5.cldrsetup.local
pvcecs-worker6.cldrsetup.local
pvcecs-worker7.cldrsetup.local
pvcecs-worker8.cldrsetup.local
pvcecs-worker9.cldrsetup.local
pvcecs-worker10.cldrsetup.local
```

Step 3. Verify the host group by running the following commands.

```
[root@ipaserver ~]# ansible datanodes -m ping
pvcbase-worker2.cldrsetup.local | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker3.cldrsetup.local | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker1.cldrsetup.local | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
```

```
    "ping": "pong"
}
```

Step 4. Copy /etc/hosts file to each node part of the cloudera deployment to resolve fqdn across the cluster

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/hosts dest=/etc/hosts"
```

Procedure 5. Set up Network configuration files and DNS Zones/Records

Step 1. We will update the network configuration file */etc/resolv.conf* on the IPA server to use the Name Server created in previous steps and will later copy this file to the rest of nodes using ansible (after installing freeipa-client, as it overrides resolv.conf and may lead to rework) to make them able to resolve FQDNs across the cluster:
(Open the file /etc/resolv.conf in edit mode and add the following. Make sure the new entry is added above any other nameserver entry. The contents of the file must look similar to the below.)

Note: Make sure that the */etc/resolv.conf* file on the ECS hosts *contains a maximum of 2 active search domains*.
https://docs.cloudera.com/data-warehouse/1.5.5/release-notes/topics/dw-private-cloud-known-issues-ecs-environments.html

```
[root@ipaserver ~]# cat /etc/resolv.conf
search ap-southeast-1.compute.internal cldrsetup.local
nameserver 172.31.24.240  # PrivateIP of FreeIPA Server must be first nameserver entry after search
nameserver 172.31.0.2     # DNS of AWS i.e. in case of PvC Configured on EC2
nameserver 127.0.0.1
[root@ipaserver ~]# cp /etc/resolv.conf /etc/resolv.conf.orig
```

```
; generated by /usr/sbin/dhclient-script
search ap-south-1.compute.internal cdppvcds.com
nameserver 172.31.40.119
```

Step 2. We will update the network configuration file */etc/sysconfig/network* on the IPA server to use the Name Server created in previous steps and will later copy this file to the rest of nodes to make them able to resolve FQDNs across the cluster:
(The changes in */etc/resolv.conf* above are temporary and would get overwritten if the machine is rebooted. In order to keep the nameserver entry persistent, open the file */etc/sysconfig/network* in edit mode and add below entries.)

```
[root@ipaserver ~]# cat /etc/sysconfig/network
NETWORKING=yes
NISDOMAIN=cldrsetup.local           # our DNS DOMAIN
DNS1=172.31.24.240           # PRIVATE_IP_OF_IPASERVER
NOZEROCONF=yes
[root@ipaserver ~]#
```

```
NETWORKING=yes
NISDOMAIN=cdppvcds.com
DNS1=172.31.40.119
NOZEROCONF=yes
```

Step 3. Copy **/etc/resolv.conf** file to each node to make them able to resolve FQDNs across the cluster:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/resolv.conf dest=/etc/resolv.conf" --become
```

Step 4. Copy **/etc/sysconfig/network** file to each node to make them able to resolve FQDNs across the cluster: (/etc/resolv.conf changes may vanished after the reboot, so to persist those changes, we need the below configuration)

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/sysconfig/network dest=/etc/sysconfig/network" --become
```

Step 5. Update Network config to make sure DNS entries persist, even after reboot:

```
# Extract DNS entries
[root@ipaserver ~]# grep '^nameserver' /etc/resolv.conf | awk '{print "DNS" NR "=" $2}' > /tmp/dns_entries.txt
```

```
# Update ifcfg-eth0 with DNS entries
[root@ipaserver ~]# while IFS= read -r line; do
  ansible all -m lineinfile -a "path=/etc/sysconfig/network-scripts/ifcfg-eth0 line='${line}' state=present
backup=true" --become
done < /tmp/dns_entries.txt

# Clean up
[root@ipaserver ~]# rm -vf /tmp/dns_entries.txt
```

Step 6. Setup Reverse DNS Zone on ipaserver, –from-ip is VPC-CIDR In this step we will be setting up a reverse DNS zone on the FreeIPA server for reverse lookup:

```
##### Take the CIDR block of the network in which the instances are created and create a reverse DNS zone by
executing the below command on the IPA Server machine.
##### ipa dnszone-add --name-from-ip=<YOUR_VPC_CIDR>

##### If your VPC has a CIDR 172.16.0.0/16, then the command looks as below.

[root@ipaserver ~]# ipa dnszone-add --name-from-ip=172.31.0.0/16
Zone name [31.172.in-addr.arpa.]:
  Zone name: 31.172.in-addr.arpa.
  Active zone: True
  Authoritative nameserver: ipaserver.cldrsetup.local.
  Administrator e-mail address: hostmaster
  SOA serial: 1715598489
  SOA refresh: 3600
  SOA retry: 900
  SOA expire: 1209600
  SOA minimum: 3600
  BIND update policy: grant CLDRSETUP.LOCAL krb5-subdomain 31.172.in-addr.arpa. PTR;
  Dynamic update: False
  Allow query: any;
  Allow transfer: none;

##### Once you execute the above command, accept the default value by hitting the enter key. It will create a
reverse DNS zone by name 16.172.in-addr.arpa. (with a trailing dot)
```

```
[root@ipaserver centos]# ipa dnszone-add --name-from-ip=172.31.0.0/16
Zone name [31.172.in-addr.arpa.]:
  Zone name: 31.172.in-addr.arpa.
  Active zone: TRUE
  Authoritative nameserver: ipaserver.cdppvcds.com.
  Administrator e-mail address: hostmaster
  SOA serial: 1680093921
  SOA refresh: 3600
  SOA retry: 900
  SOA expire: 1209600
  SOA minimum: 3600
  BIND update policy: grant CDPPVCDS.COM krb5-subdomain 31.172.in-addr.arpa. PTR;
  Dynamic update: FALSE
  Allow query: any;
  Allow transfer: none;
```

Step 7. Disable krb5 ccache config and verify:

```
##### OPEN /etc/krb5.conf on IPASERVER and comment ccache conf: (this step is not needed on any cluster node,
as CDP will manage the krb5.conf in further steps config)
#### After the setup is complete, we need to make a kerberos config change which gets enabled automatically
post the ipa server setup.

##### Open the file /etc/krb5.conf in edit mode and comment out the line related to ccache_name as shown
below.
[root@ipaserver ~]# vi /etc/krb5.conf
##### Comment the below ccache config
#default_ccache_name = KEYRING:persistent:%{uid}

##### After any changes of /etc/krb5.conf anytime, do run the below commands to restart all the IPA services.
[root@ipaserver ~]# ipactl restart
```

```
[root@ipaserver centos]# cat /etc/krb5.conf
includedir /etc/krb5.conf.d/
includedir /var/lib/sss/pubconf/krb5.include.d/

[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = CDPPVCDS.COM
 dns_lookup_realm = false
 dns_lookup_kdc = true
 rdns = false
 ticket_lifetime = 24h
 forwardable = true
 udp_preference_limit = 0
# default_ccache_name = KEYRING:persistent:%{uid}
```

Step 8. Prepare the commands for adding dnsrecord and configuring reverse lookup:

```
##### ADD The entry of all individual machines (separate IP separate command) to reverse DNS zone:

##### We need to create a record for each machine in the reverse DNS zone, created previously.
##### Use the below command as reference and make changes as per your configuration/machine's private IP and
Hostname.

##### Add the entry of this e.g. IPA server machine to the reverse DNS zone.
##### We need to add the IPV4 address in reverse order. The first two octets are already added in the reverse
zone above. Now we need to create a record for this machine inside that zone by using the last two octets.

##### In the command below you need to add the record by providing the last two octets of your machine's
private IPV4 in reverse order. Include the trailing dot after the machine name FQDN in the above command.

##### Generate the command as shown below and run the same for all the FreeIPA agents, that includes all the
nodes of Base and ECS cluster.
       ipa dnsrecord-add <2nd>.<1st>.in-addr.arpa. <4th>.<3rd> --ptr-rec <server FQDN>.
##### Example:
       ipa dnsrecord-add 16.172.in-addr.arpa. 226.31 --ptr-rec ipaserver.cldrsetup.local.

##### Following the same, The record for the machine should be created in the Reverse DNS zone.
```

```
[root@ipaserver centos]# ipa dnsrecord-add 31.172.in-addr.arpa. 119.40 --ptr-rec ipaserver.cdppvcds.com.
  Record name: 119.40
  PTR record: ipaserver.cdppvcds.com.
```

```
[root@cdpbase centos]# ipa dnsrecord-add 31.172.in-addr.arpa. 234.0 --ptr-rec cdpbase.cdppvcds.com.
  Record name: 234.0
  PTR record: cdpbase.cdppvcds.com.
```

```
[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 240.24  --ptr-rec ipaserver.cldrsetup.local.
  Record name: 240.24
  PTR record: ipaserver.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 139.27 --ptr-rec cldr-mngr.cldrsetup.local.
  Record name: 139.27
  PTR record: cldr-mngr.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 104.21 --ptr-rec pvcbase-master.cldrsetup.local.
  Record name: 104.21
  PTR record: pvcbase-master.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 185.16 --ptr-rec pvcbase-worker1.cldrsetup.local.
  Record name: 185.16
```

```
  PTR record: pvcbase-worker1.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 0.23 --ptr-rec pvcbase-worker2.cldrsetup.local.
  Record name: 0.23
  PTR record: pvcbase-worker2.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 240.18 --ptr-rec pvcbase-worker3.cldrsetup.local.
  Record name: 240.18
  PTR record: pvcbase-worker3.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 239.30 --ptr-rec pvcecs-master.cldrsetup.local.
  Record name: 239.30
  PTR record: pvcecs-master.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 43.22 --ptr-rec pvcecs-worker1.cldrsetup.local.
  Record name: 43.22
  PTR record: pvcecs-worker1.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 249.30 --ptr-rec pvcecs-worker2.cldrsetup.local.
  Record name: 249.30
  PTR record: pvcecs-worker2.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 24.26 --ptr-rec pvcecs-worker3.cldrsetup.local.
  Record name: 24.26
  PTR record: pvcecs-worker3.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 198.24 --ptr-rec pvcecs-worker4.cldrsetup.local.
  Record name: 198.24
  PTR record: pvcecs-worker4.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 53.24 --ptr-rec pvcecs-worker5.cldrsetup.local.
  Record name: 53.24
  PTR record: pvcecs-worker5.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 43.22 --ptr-rec pvcecs-worker6.cldrsetup.local.
  Record name: 43.22
  PTR record: pvcecs-worker6.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 249.30 --ptr-rec pvcecs-worker7.cldrsetup.local.
  Record name: 249.30
  PTR record: pvcecs-worker7.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 24.26 --ptr-rec pvcecs-worker8.cldrsetup.local.
  Record name: 24.26
  PTR record: pvcecs-worker8.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 198.24 --ptr-rec pvcecs-worker9.cldrsetup.local.
  Record name: 198.24
  PTR record: pvcecs-worker9.cldrsetup.local.

[root@ipaserver ~]# ipa dnsrecord-add 31.172.in-addr.arpa. 53.24 --ptr-rec pvcecs-worker10.cldrsetup.local.
  Record name: 53.24
  PTR record: pvcecs-worker10.cldrsetup.local.
```

Step 9. Verify the DNS records have been added successfully:

```
[root@ipaserver ~]# ipa dnsrecord-find 31.172.in-addr.arpa.
  Record name: @
  NS record: ipaserver.cldrsetup.local.

  Record name: 240.24
  PTR record: ipaserver.cldrsetup.local.

  Record name: 139.27
  PTR record: cldr-mngr.cldrsetup.local.

  Record name: 104.21
  PTR record: pvcbase-master.cldrsetup.local.

  Record name: 185.16
  PTR record: pvcbase-worker1.cldrsetup.local.

  Record name: 0.23
```

```
  PTR record: pvcbase-worker2.cldrsetup.local.

  Record name: 240.18
  PTR record: pvcbase-worker3.cldrsetup.local.

  Record name: 239.30
  PTR record: pvcecs-master.cldrsetup.local.

  Record name: 43.22
  PTR record: pvcecs-worker1.cldrsetup.local.

  Record name: 249.30
  PTR record: pvcecs-worker2.cldrsetup.local.

  Record name: 198.24
  PTR record: pvcecs-worker3.cldrsetup.local.

  Record name: 53.24
  PTR record: pvcecs-worker4.cldrsetup.local.

  Record name: 24.26
  PTR record: pvcecs-worker5.cldrsetup.local.

  Record name: 43.22
  PTR record: pvcecs-worker6.cldrsetup.local.

  Record name: 0.23
  PTR record: pvcecs-worker7.cldrsetup.local.

  Record name: 198.24
  PTR record: pvcecs-worker8.cldrsetup.local.

  Record name: 53.24
  PTR record: pvcecs-worker9.cldrsetup.local.

  Record name: 24.26
  PTR record: pvcecs-worker10.cldrsetup.local.

----------------------------
Number of entries returned 18
----------------------------
[root@ipaserver ~]#
```

Procedure 6. Configure freeipa-client on all other nodes to get them managed by ipa-server

Step 1: Install free-ipa client along with other packages needed on all hosts except ipaserver:

Note: Setup ipaserver client and krb5 libs on each node before copying resolv.conf, as installation of ipa-client will override this. *(UDP port 123 and TCP port 389 need to be enabled for ipa services, ntp and timesync)*

Note: Remove chrony from all hosts using ansible as it creates issues in installing and configuring ipa services successfully.

Note: Please review JAVA requirement in Cloudera on premises Base Requirements and Supported Versions sections: (We installed OpenJDK11 for this solution validation, ipa-client will also require and auto install java 11 on all hosts, if it is not present or any different version is installed e.g. java17)

https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-java-requirements.html

```
[root@ipaserver ~]# ansible all -m shell -a "sudo subscription-manager repos
--enable=rhel-9-for-x86_64-baseos-rpms && sudo subscription-manager repos
--enable=rhel-9-for-x86_64-appstream-rpms && sudo dnf install -y java-17-openjdk java-17-openjdk-devel
python3-pip wget telnet mlocate tar traceroute net-tools bind-utils traceroute nc && java -version &&
python3 -V && pip3 install --upgrade pip && pip3 -V && pip3 install psycopg2-binary && pip3 list |grep
psy"

[root@ipaserver ~]# ansible all -m shell -a "sudo dnf install -y
https://download.postgresql.org/pub/repos/yum/reporpms/EL-9-x86_64/pgdg-redhat-repo-latest.noarch.rpm &&
sudo dnf install -y postgresql14"

[root@ipaserver ~]# ansible all -m shell -a "sudo subscription-manager repos
--enable=rhel-9-for-x86_64-baseos-rpms && sudo subscription-manager repos
```

```
--enable=rhel-9-for-x86_64-appstream-rpms && sudo dnf install -y freeipa-client openldap-clients
krb5-workstation krb5-libs && chronyc tracking && chronyc sources" -l 'all:!admin'
```

Step 2: Install and Setup IPA services by configuring the free-ipa client on all machines (except ipaserver) and add all the machines to the DNS server, by running the command "***ipa-client-install***" to set up the IPA client.

Enter the values for these parameters as below. After entering these values, it should return the message as ***"The ipa-client-install command was successful".***

Parameter	Value
Do you want to configure chrony with NTP server or pool address? [no]:	yes
Enter NTP source server addresses separated by comma, or press Enter to skip:	<ENTER>
Enter a NTP source pool address, or press Enter to skip:	<ENTER>
Continue to configure the system with these values? [no]:	yes
User authorized to enroll computers:	admin
Password for admin@<Your_Domain>:	<Password created earlier> (***cloudera123***)

```
[root@pvcbase-master ~]# ipa-client-install --force-join
This program will set up IPA client.
Version 4.11.0

Discovery was successful!
Do you want to configure chrony with NTP server or pool address? [no]: yes
Enter NTP source server addresses separated by comma, or press Enter to skip: <ENTER>
Enter a NTP source pool address, or press Enter to skip: <ENTER>
Client hostname: cldr-mngr.cldrsetup.local
Realm: CLDRSETUP.LOCAL
DNS Domain: cldrsetup.local
IPA Server: ipaserver.cldrsetup.local
BaseDN: dc=cldrsetup,dc=local

Continue to configure the system with these values? [no]: yes
Synchronizing time
No SRV records of NTP servers were found and no NTP server or pool address was provided.
Using default chrony configuration.
Attempting to sync time with chronyc.
Time synchronization was successful.
User authorized to enroll computers: <admin>
Password for admin@CLDRSETUP.LOCAL: <cloudera123>
Successfully retrieved CA cert
    Subject:     CN=Certificate Authority,O=CLDRSETUP.LOCAL
    Issuer:      CN=Certificate Authority,O=CLDRSETUP.LOCAL
    Valid From:  2024-05-13 10:59:53+00:00
    Valid Until: 2044-05-13 10:59:53+00:00

Enrolled in IPA realm CLDRSETUP.LOCAL
Created /etc/ipa/default.conf
Configured /etc/sssd/sssd.conf
Systemwide CA database updated.
Hostname (pvcbase-master.cldrsetup.local) does not have A/AAAA record.
Adding SSH public key from /etc/ssh/ssh_host_rsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ed25519_key.pub
SSSD enabled
Configured /etc/openldap/ldap.conf
Configured /etc/ssh/ssh_config
Configured /etc/ssh/sshd_config.d/04-ipa.conf
Configuring cldrsetup.local as NIS domain.
Configured /etc/krb5.conf for IPA realm CLDRSETUP.LOCAL
Client configuration complete.
The ipa-client-install command was successful
[root@pvcbase-master ~]#
```

```
[root@cdpbase centos]# ipa-client-install --force-ntpd
Discovery was successful!
Client hostname: cdpbase.cdppvcds.com
Realm: CDPPVCDS.COM
DNS Domain: cdppvcds.com
IPA Server: ipaserver.cdppvcds.com
BaseDN: dc=cdppvcds,dc=com

Continue to configure the system with these values? [no]: yes
Synchronizing time with KDC...
Attempting to sync time using ntpd.  Will timeout after 15 seconds
Attempting to sync time using ntpd.  Will timeout after 15 seconds
Unable to sync time with NTP server, assuming the time is in sync. Please check that 123 UDP port is opened.
User authorized to enroll computers: admin
Password for admin@CDPPVCDS.COM:
Successfully retrieved CA cert
    Subject:     CN=Certificate Authority,O=CDPPVCDS.COM
    Issuer:      CN=Certificate Authority,O=CDPPVCDS.COM
    Valid From:  2023-03-29 11:23:01
    Valid Until: 2043-03-29 11:23:01

Enrolled in IPA realm CDPPVCDS.COM
Created /etc/ipa/default.conf
New SSSD config will be created
Configured sudoers in /etc/nsswitch.conf
Configured /etc/sssd/sssd.conf
Configured /etc/krb5.conf for IPA realm CDPPVCDS.COM
trying https://ipaserver.cdppvcds.com/ipa/json
[try 1]: Forwarding 'schema' to json server 'https://ipaserver.cdppvcds.com/ipa/json'
trying https://ipaserver.cdppvcds.com/ipa/session/json
[try 1]: Forwarding 'ping' to json server 'https://ipaserver.cdppvcds.com/ipa/session/json'
[try 1]: Forwarding 'ca_is_enabled' to json server 'https://ipaserver.cdppvcds.com/ipa/session/json'
Systemwide CA database updated.
Hostname (cdpbase.cdppvcds.com) does not have A/AAAA record.
Missing reverse record(s) for address(es): 172.31.0.234.
Adding SSH public key from /etc/ssh/ssh_host_rsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ed25519_key.pub
[try 1]: Forwarding 'host_mod' to json server 'https://ipaserver.cdppvcds.com/ipa/session/json'
SSSD enabled
Configured /etc/openldap/ldap.conf
NTP enabled
Configured /etc/ssh/ssh_config
Configured /etc/ssh/sshd_config
Configuring cdppvcds.com as NIS domain.
Client configuration complete.
The ipa-client-install command was successful
```

Step 3: Verify KDC setup: kerberos ticket generation is working fine by generating a ticket for the admin user from all individual hosts.

```
##### Run the kinit admin command to authenticate as admin and enter the directory password provided during
ipa server installation. The command should generate the ticket and should be listed by executing klist -e.

[root@ipaserver ~]# kinit admin
Password for admin@CLDRSETUP.LOCAL: <cloudera123>

[root@ipaserver ~]# klist -e
Ticket cache: KCM:0
Default principal: admin@CLDRSETUP.LOCAL

Valid starting       Expires              Service principal
05/13/2024 04:07:15  05/14/2024 03:30:48  krbtgt/CLDRSETUP.LOCAL@CLDRSETUP.LOCAL
        Etype (skey, tkt): aes256-cts-hmac-sha384-192, aes256-cts-hmac-sha384-192

##### try kinit admin@CLDRSETUP.LOCAL
##### (if fails anytime, run below commands

[root@ipaserver ~]# ipactl stop && ipactl start && ipactl status

##### Verify the status of ipa services installed

[root@ipaserver ~]# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
named Service: RUNNING
httpd Service: RUNNING
ipa-custodia Service: RUNNING
ntpd Service: RUNNING
pki-tomcatd Service: RUNNING
ipa-otpd Service: RUNNING
ipa-dnskeysyncd Service: RUNNING
```

```
ipa: INFO: The ipactl command was successful
```

This command should return the below output:

```
[root@ipaserver centos]# kinit admin
Password for admin@CDPPVCDS.COM:
[root@ipaserver centos]# klist -e
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: admin@CDPPVCDS.COM

Valid starting     Expires            Service principal
04/05/23 10:49:29  04/06/23 10:49:26  krbtgt/CDPPVCDS.COM@CDPPVCDS.COM
        Etype (skey, tkt): aes256-cts-hmac-sha1-96, aes256-cts-hmac-sha1-96
[root@ipaserver centos]#
```

Step 4: Verify the network configuration file ***/etc/resolv.conf*** on the IPA server to use the Name Server created in previous steps (after installing freeipa-client, as it overrides resolv.conf and may lead to rework) to make them able to resolve FQDNs across the cluster:
(Open the file /etc/resolv.conf in edit mode and verify the following. Make sure the new entry is added above any other nameserver entry. The contents of the file must look similar to the below.)

Note: Make sure that the ***/etc/resolv.conf*** file on the ECS hosts ***contains a maximum of 2 active search domains***.
https://docs.cloudera.com/data-warehouse/1.5.5/release-notes/topics/dw-private-cloud-known-issues-ecs-environments.html

```
[root@ipaserver ~]# cat /etc/resolv.conf
search ap-southeast-1.compute.internal cldrsetup.local
nameserver 172.31.24.240   # PrivateIP of FreeIPA Server must be first nameserver entry after search
nameserver 172.31.0.2      # DNS of AWS i.e. in case of PvC Configured on EC2
nameserver 127.0.0.1
[root@ipaserver ~]# cp /etc/resolv.conf /etc/resolv.conf.orig
```

```
; generated by /usr/sbin/dhclient-script
search ap-south-1.compute.internal cdppvcds.com
nameserver 172.31.40.119
```

Step 5: Verify the network configuration file ***/etc/sysconfig/network*** on the IPA server to use the Name Server created in previous steps:
(The changes in ***/etc/resolv.conf*** above are temporary and would get overwritten if the machine is rebooted. In order to keep the nameserver entry persistent, open the file ***/etc/sysconfig/network*** in edit mode and verify the entries below.)

```
[root@ipaserver ~]# cat /etc/sysconfig/network
NETWORKING=yes
NISDOMAIN=cldrsetup.local            # our DNS DOMAIN
DNS1=172.31.24.240           # PRIVATE_IP_OF_IPASERVER
NOZEROCONF=yes
[root@ipaserver ~]#
```

```
NETWORKING=yes
NISDOMAIN=cdppvcds.com
DNS1=172.31.40.119
NOZEROCONF=yes
```

Step 6: Copy **/etc/resolv.conf** file to each node again, to make them able to resolve FQDNs across the cluster:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/resolv.conf dest=/etc/resolv.conf"
```

Step 7: Copy **/etc/sysconfig/network** file again, to each node to make them able to resolve FQDNs across the cluster: (/etc/resolv.conf changes may vanished after the reboot, so to persist those changes, we need the below configuration)

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/sysconfig/network dest=/etc/sysconfig/network"
```

Step 8: Enable permissions for HDFS and for PAM Authentication:

```
[root@ipaserver ~]# ansible all -m shell -a "chmod 1777 /tmp && chmod 444 /etc/shadow"
```

Step 9: Login to IPAServer node and verify forward and reverse DNS lookup is working fine from each machine:

```
[root@ipaserver ~]# nslookup cldr-mngr.cldrsetup.local
Server:         172.31.24.240
Address:        172.31.24.240#53

Name:   cldr-mngr.cldrsetup.local
Address: 172.31.27.139

#(forward lookup)Running the below command should return the IPV4 of the machine in the Answer Section.

# dig <FQDN of the SERVER> A
# dig $(hostname) A | grep -A2 ANSWER
# Ex:- dig ipaserver.cldrsetup.local A

[root@ipaserver ~]# dig $(hostname -f) A | grep -A2 ANSWER
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
--
;; ANSWER SECTION:
ipaserver.cldrsetup.local. 1200    IN      A       172.31.24.240

#(reverse lookup) Running the below command should return the hostname of the machine in the Answer Section.

# dig -x <Private_IP_of_SERVER>
# dig -x $(hostname -i)|grep -A2 ANSWER
# Ex:- dig -x 172.31.40.119

[root@ipaserver ~]# dig -x $(hostname -i) | grep -A2 ANSWER
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
--
;; ANSWER SECTION:
240.24.31.172.in-addr.arpa. 86400 IN    PTR     ipaserver.cldrsetup.local.

[root@ipaserver ~]$
```

Step 10: Login on ipaserver, configure and validate wildcard DNS record is working fine and resolvable, which is required later for the ECS data service cluster (if not set properly, chances of ECS installation getting corrupt):

```
[root@ipaserver ~]#  ipa dnsrecord-add cldrsetup.local *.apps
Please choose a type of DNS resource record to be added
The most common types for this type of zone are: A, AAAA

DNS resource record type: A
A IP Address: 172.31.30.239       #Provide the IP address of ecs-master node
  Record name: *.apps
  A record: 172.31.30.239

[root@ipaserver ~]# nslookup console-cdp.apps.cldrsetup.local
Server:         172.31.24.240
Address:        172.31.24.240#53

Name:   console-cdp.apps.cldrsetup.local
Address: 172.31.30.239

[root@ipaserver ~]# dig console-cdp.apps.cldrsetup.local A | grep -A2 ANSWER
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
--
;; ANSWER SECTION:
console-cdp.apps.cldrsetup.local. 86400 IN A       172.31.30.239
```

```
[root@ipaserver ~]# dig -x 172.31.30.239 | grep -A2 ANSWER
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
--
;; ANSWER SECTION:
239.30.31.172.in-addr.arpa. 86400 IN    PTR     pvcecs-master.cldrsetup.local.

[root@ipaserver ~]#
```

Step 11: Download and copy postgresql-jdbc driver to all hosts:

```
[root@ipaserver ~]# wget https://jdbc.postgresql.org/download/postgresql-42.7.7.jar
[root@ipaserver ~]# chmod 644 postgresql-42.7.7.jar
[root@ipaserver ~]# ansible all -m copy -a "src=postgresql-42.7.7.jar
dest=/usr/share/java/postgresql-connector-java.jar"
[root@ipaserver ~]# ansible all -m shell -a "sudo ls -l /usr/share/java/postgresql-connector-java.jar"
[root@ipaserver ~]#
```

Procedure 7. Disable the Linux Firewall

Note: The default Linux firewall settings are too restrictive for any Hadoop deployment. Since the Cloudera on premises deployment will be in its own isolated network in the on premise environment, there is no need for that additional firewall. **(NA in AWS EC2)**

```
##### Either disable the firewall or update the rules: (ON ALL HOSTS)
[root@ipaserver ~]# ansible all -m command -a "firewall-cmd --zone=public --add-port=80/tcp --permanent"
[root@ipaserver ~]# ansible all -m command -a "firewall-cmd --zone=public --add-port=443/tcp --permanent"
[root@ipaserver ~]# ansible all -m command -a "firewall-cmd --reload"
[root@ipaserver ~]# ansible all -m command -a "systemctl disable firewalld && systemctl stop firewalld &&
systemctl status firewalld | grep -e disabled -e inactive"
[root@ipaserver ~]
```

Procedure 8. Disable SELinux

Note: SELinux must be disabled during the install procedure and cluster setup. SELinux can be enabled after installation and while the cluster is running.

Step 1: SELinux can be disabled by editing */etc/selinux/config* (in some systems it would be */etc/sysconfig/selinux*) To disable SELinux, change SELINUX=enforcing to SELINUX=disabled or SELINUX=permissive. follow these steps:

```
[root@ipaserver ~]# ansible all -m shell -a "sed -i 's/SELINUX=enforcing/SELINUX=disabled/g'
/etc/selinux/config"
[root@ipaserver ~]# ansible all -m shell -a "setenforce 0"
[root@ipaserver ~]# ansible all -m shell -a "getenforce"
```

Note: This command may fail if SELinux is already disabled. This requires reboot to take effect.
Note: While the suggested configuration is to disable SELinux as shown below, if for any reason SELinux needs to be enabled on the cluster, run the following command to make sure that the httpd can read the *Yum* profiles.

```
[root@ipaserver ~]# chcon -R -t httpd_sys_content_t /var/www/html/
```

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Procedure 9. Enable Syslog

Syslog must be enabled on each node to preserve logs regarding killed processes or failed jobs. Modern versions such as syslog-ng and rsyslog are possible, making it more difficult to be sure that a syslog daemon is present.

Step 1. Use one of the following commands to confirm that the service is properly configured:

```
[root@ipaserver ~]# ansible all -m command -a "rsyslogd -v"
[root@ipaserver ~]# ansible all -m command -a "service rsyslog status"
```

Procedure 10. Set ulimit

On each node, ulimit -n specifies the number of inodes that can be opened simultaneously. With the default value of 1024, the system appears to be out of disk space and shows no inodes available. This value should be set to 64000 on every node. Higher values are unlikely to result in an appreciable performance gain.

Step 1. For setting the ulimit on RedHat, edit **/etc/security/limits.conf** on admin node and add the following lines:

```
[root@ipaserver ~]# vi /etc/security/limits.conf
* soft nofile 1048576
* hard nofile 1048576
```

Step 2. Copy the /etc/security/limits.conf file from admin node (ipaserver) to all the nodes using the following command:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/security/limits.conf dest=/etc/security/limits.conf"
```

Step 3. Make sure that the /etc/pam.d/su file contains the following settings:

```
[root@ipaserver ~]# vi /etc/pam.d/su
#%PAM-1.0
auth            required        pam_env.so
auth            sufficient      pam_rootok.so
# Uncomment the following line to implicitly trust users in the "wheel" group.
#auth           sufficient      pam_wheel.so trust use_uid
# Uncomment the following line to require a user to be in the "wheel" group.
#auth           required        pam_wheel.so use_uid
auth            include         system-auth
auth            include         postlogin
account         sufficient      pam_succeed_if.so uid = 0 use_uid quiet
account         include         system-auth
password        include         system-auth
session         include         system-auth
session         include         postlogin
session         optional        pam_xauth.so
```

Step 4. Copy the /etc/pam.d/su file from admin node (ipaserver) to all the nodes using the following command:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/pam.d/su dest=/etc/pam.d/su"
```

Note: The ulimit values are applied on a new shell, running the command on a node on an earlier instance of a shell will show old values.

Procedure 11. Set TCP Retries

Adjusting the tcp_retries parameter for the system network enables faster detection of failed nodes. Given the advanced network-ing features of UCS, this is a safe and recommended change (failures observed at the operating system layer are most likely serious rather than transitory).

Note: On each node, setting the number of TCP retries to 5 can help detect unreachable nodes with less latency.

Step 1. Edit the file /etc/sysctl.conf on ipaserver node and add the following lines:

```
[root@ipaserver ~]# vi /etc/sysctl.conf
```

```
net.ipv4.tcp_retries2=5
```

Step 2. Copy the /etc/sysctl.conf file from admin node to all the nodes using the following command:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/sysctl.conf dest=/etc/sysctl.conf"
```

Procedure 12. Disable IPv6 Defaults

Step 1. Run the following command:

```
[root@ipaserver ~]# ansible all -m shell -a "echo 'net.ipv6.conf.all.disable_ipv6 = 1' >>
/etc/sysctl.conf"  -l 'all:!ecsmasternodes:!ecsnodes'
[root@ipaserver ~]# ansible all -m shell -a "echo 'net.ipv6.conf.default.disable_ipv6 = 1' >>
/etc/sysctl.conf" -l 'all:!ecsmasternodes:!ecsnodes'
[root@ipaserver ~]# ansible all -m shell -a "echo 'net.ipv6.conf.lo.disable_ipv6 = 0' >> /etc/sysctl.conf"
-l 'all:!ecsmasternodes:!ecsnodes'
```

Procedure 13. Disable Swapping

Step 1. Run the following to set VM swappiness to 1, by updating /etc/sysctl.conf file on all nodes:

```
[root@ipaserver ~]# ansible all -m shell -a "echo 'vm.swappiness=1' >> /etc/sysctl.conf"
```

Procedure 14. Disable Memory Overcommit

Step 1. Run the following on all nodes. Variable vm.overcommit_memory=0

```
[root@ipaserver ~]# ansible all -m shell -a "echo 'vm.overcommit_memory=0' >> /etc/sysctl.conf"
```

Step 2. Load the settings from default sysctl file /etc/sysctl.conf and verify the content of sysctl.conf:

```
[root@ipaserver ~]# ansible all -m shell -a "sysctl -p"     ## Reload sysctl.conf
[root@ipaserver ~]# ansible all -m shell -a "cat /etc/sysctl.conf"
net.ipv4.tcp_retries2=5
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 0
vm.swappiness=1
vm.overcommit_memory=0
[root@ipaserver ~]#
```

Procedure 15. Disable Transparent Huge Pages

Disabling Transparent Huge Pages (THP) reduces elevated CPU usage caused by THP.

Step 1. You must run the following commands for every reboot:

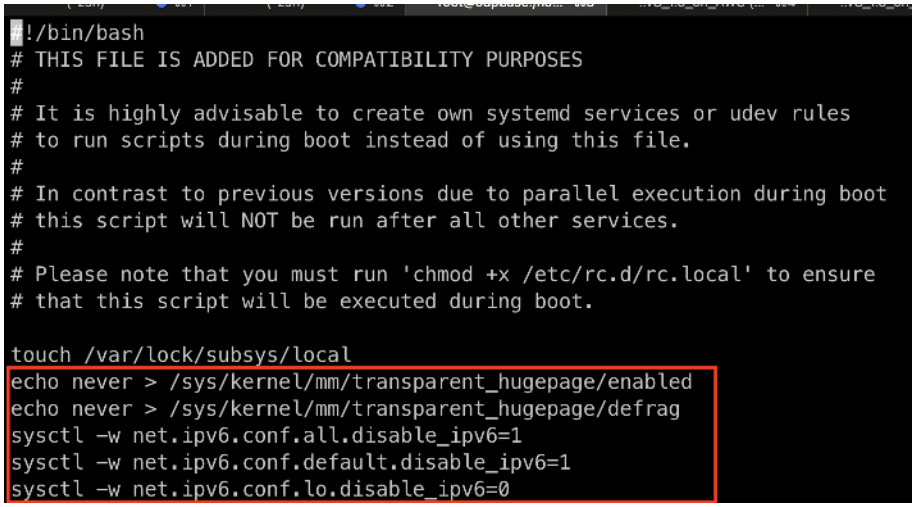
```
[root@ipaserver ~]# ansible all -m shell -a "echo never > /sys/kernel/mm/transparent_hugepage/enabled"
[root@ipaserver ~]# ansible all -m shell -a "echo never > /sys/kernel/mm/transparent_hugepage/defrag"
```

Step 2. On the Ansible-controller/ ipaserver node, run the following commands:

```
[root@ipaserver ~]# rm -f /root/thp_disable
[root@ipaserver ~]# echo "echo never > /sys/kernel/mm/transparent_hugepage/enabled" >> /root/thp_disable
[root@ipaserver ~]# echo "echo never > /sys/kernel/mm/transparent_hugepage/defrag" >> /root/thp_disable

##### Disable IPV6
[root@ipaserver ~]# echo "sysctl -w net.ipv6.conf.all.disable_ipv6=1" >> /root/thp_disable
[root@ipaserver ~]# echo "sysctl -w net.ipv6.conf.default.disable_ipv6=1" >> /root/thp_disable
[root@ipaserver ~]# echo "sysctl -w net.ipv6.conf.lo.disable_ipv6=0" >> /root/thp_disable
```

Step 3. Copy file to each node to copy the command to */etc/rc.d/rc.local* so they are executed automatically for every reboot:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/root/thp_disable dest=/root/thp_disable"

##### Append the content of file thp_disable to /etc/rc.d/rc.local:
[root@ipaserver ~]# ansible all -m shell -a "cat /root/thp_disable >> /etc/rc.d/rc.local"
[root@ipaserver ~]# ansible all -m shell -a "chmod +x /etc/rc.d/rc.local"
[root@ipaserver ~]# ansible all -m shell -a "cat /etc/rc.d/rc.local"
#!/bin/bash
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.
touch /var/lock/subsys/local
# Disable Transparent Huge Pages
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag
# Disable IPV6
sysctl -w net.ipv6.conf.all.disable_ipv6=1
sysctl -w net.ipv6.conf.default.disable_ipv6=1
sysctl -w net.ipv6.conf.lo.disable_ipv6=0
[root@ipaserver ~]#
```



Procedure 16. Disable tuned service

For Cloudera cluster with hosts are running RHEL/CentOS 7.x or 8.x or 9.x, disable the "tuned" service by running the following commands:

Step 1. Ensure that the tuned service is started.

```
[root@ipaserver ~]# ansible all -m shell -a "systemctl start tuned"
```

Step 2. Turn the tuned service off.

```
[root@ipaserver ~]# ansible all -m shell -a "tuned-adm off"
```

Step 3. Ensure that there are no active profiles.

```
[root@ipaserver ~]# ansible all -m shell -a "tuned-adm list"
# The output should contain the following line:
# pvcecs-worker4.cldrsetup.local | CHANGED | rc=0 >>
Available profiles:
- accelerator-performance     - Throughput performance based tuning with disabled higher latency STOP
states
- aws                         - Optimize for aws ec2 instances
- balanced                    - General non-specialized tuned profile
- desktop                     - Optimize for the desktop use-case
```

```
- hpc-compute                  - Optimize for HPC compute workloads
- intel-sst                    - Configure for Intel Speed Select Base Frequency
- latency-performance          - Optimize for deterministic performance at the cost of increased power
consumption
- network-latency              - Optimize for deterministic performance at the cost of increased power
consumption, focused on low latency network performance
- network-throughput           - Optimize for streaming network throughput, generally only necessary on
older CPUs or 40G+ networks
- optimize-serial-console      - Optimize for serial console use.
- powersave                    - Optimize for low power consumption
- throughput-performance       - Broadly applicable tuning that provides excellent performance across a
variety of common server workloads
- virtual-guest                - Optimize for running inside a virtual guest
- virtual-host                 - Optimize for running KVM guests
No current active profile.
```

Step 4. Shutdown and disable the tuned service.

```
[root@ipaserver ~]# ansible all -m shell -a "systemctl stop tuned"
[root@ipaserver ~]# ansible all -m shell -a "systemctl disable tuned"
```

Procedure 17. Turning off TCP checksum offload on the all ecs nodes' (both master and worker) network interface (Only Applicable for VMWare machines)

The default CNI (Canal) that comes with ECS (RKE2 from SUSE) does not support VM. Customers have reported escalations with the default CNI when they use VMWARE (WF for example). Customers may experience network connectivity issues or degraded performance when deploying ECS clusters on VMWARE environments with the default CNI. So, they use a workaround by turning off TCP checksum offload on the interface.

Note: Disabling TCP checksum offload may have implications on network performance or security. Evaluate the impact of this workaround in the specific environment and consider reverting the changes once a permanent solution or alternative workaround is available. Additionally, consult with the VMWARE documentation or support resources for guidance on network configuration and optimization in VMWARE environments.

Here's how you can perform the workaround for customers experiencing issues with the default CNI in ECS (RKE2 from SUSE) when using VMWARE infrastructure, by disabling TCP checksum offload on the network interface used by the affected ECS nodes:

Step 1: Identify the network interface:

```
##### Determine the network interface used by the affected ECS nodes. This can typically be found using
the `ifconfig` or `ip addr` command:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "ifconfig | grep flags | grep -v lo && ip
addr | grep mtu | grep -v lo"

[root@ipaserver ~]$ ifconfig | grep flags | grep -v lo
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
[root@ipaserver ~]$ ip addr | grep mtu | grep -v lo
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
[root@ipaserver ~]$
```

Step 2: Disable TCP Checksum Offload: Use the appropriate commands or configuration settings to disable TCP checksum offload on the identified network interface. For example, using ethtool:

```
##### Replace `<interface_name>` with the name of the network interface identified in step 1.
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo ethtool -K <interface_name> tx off"
##### After disabling TCP checksum offload, verify that the changes have been applied correctly and that the
affected ECS nodes no longer experience the reported issues.
```

Procedure 18. Create partitions on ECS nodes (master and worker) manually, if not present: (Do df -h) (Skip this)

Step 1: Create partitions on disk attached. We require three partitions i.e. /docker, /cdwdata and /lhdata:

```
[root@pvcecs-master ~]# lsblk
NAME    MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
xvda    202:0    0   600G  0 disk
├─xvda1 202:1    0     1M  0 part
├─xvda2 202:2    0   200M  0 part /boot/efi
├─xvda3 202:3    0   600M  0 part /boot
└─xvda4 202:4    0 599.2G  0 part /
xvdb    202:16   0   600G  0 disk
xvdc    202:32   0   600G  0 disk
xvdd    202:48   0   600G  0 disk
xvde    202:48   0   600G  0 disk

[root@pvcecs-master ~]# fdisk -l
Disk /dev/sda: 268.4 GB, 268435456000 bytes, 524288000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000a04a5

Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     2099199     1048576   83  Linux
/dev/sda2         2099200   524287999   261094400   8e  Linux LVM
/dev/xvdb         2099200   524287999   261094400   8e  Linux LVM
/dev/xvdc         2099200   524287999   261094400   8e  Linux LVM
/dev/xvdd         2099200   524287999   261094400   8e  Linux LVM
/dev/xvde         2099200   524287999   261094400   8e  Linux LVM

Disk /dev/sdb: 644.2 GB, 644245094400 bytes, 1258291200 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
[root@pvcecs-master ~]#

[root@pvcecs-master ~]# sudo parted /dev/xvdb mklabel gpt
Information: You may need to update /etc/fstab.

[root@pvcecs-master ~]# sudo parted /dev/xvdc mklabel gpt
[root@pvcecs-master ~]# sudo parted /dev/xvdd mklabel gpt

[root@pvcecs-master ~]# sudo parted -a opt /dev/xvdb mkpart primary ext4 0% 100%
Information: You may need to update /etc/fstab.

[root@pvcecs-master ~]# sudo parted -a opt /dev/xvdc mkpart primary ext4 0% 100%
[root@pvcecs-master ~]# sudo parted -a opt /dev/xvdd mkpart primary ext4 0% 100%

[root@pvcecs-master ~]# sudo mkfs.ext4 /dev/xvdb
mke2fs 1.46.5 (30-Dec-2021)
Found a gpt partition table in /dev/xvdb
Proceed anyway? (y,N) y
Creating filesystem with 157286400 4k blocks and 39321600 inodes
Filesystem UUID: 934fece5-074e-4102-8481-6ed3a3b10931
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
        102400000

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks):
done
Writing superblocks and filesystem accounting information: done

[root@pvcecs-master ~]# sudo mkfs.ext4 /dev/xvdc
[root@pvcecs-master ~]# sudo mkfs.ext4 /dev/xvdd

[root@pvcecs-master ~]# sudo mkdir -p /docker /cdwdata /lhdata

[root@pvcecs-master ~]# sudo mount /dev/xvdb /docker
[root@pvcecs-master ~]# sudo mount /dev/xvdc /cdwdata
```

```
[root@pvcecs-master ~]# sudo mount /dev/xvdd /lhdata

[root@pvcecs-master ~]# sudo blkid /dev/xvdb
[root@pvcecs-master ~]# sudo blkid /dev/xvdc
[root@pvcecs-master ~]# sudo blkid /dev/xvdd

[root@pvcecs-master ~]# cat /etc/fstab
UUID=497ad222-04fa-453f-b110-ba8001d38788       /       xfs     defaults        0       0
UUID=2e0d9ec9-a82a-43cc-a8e2-c6db30e7f6a4       /boot   xfs     defaults        0       0
UUID=7B77-95E7  /boot/efi       vfat    defaults,uid=0,gid=0,umask=077,shortname=winnt  0       2
[root@pvcecs-master ~]#

[root@pvcecs-master ~]# cat>> /etc/fstab
UUID=497ad222-04fa-453f-b110-ba8001d38788       /       xfs     defaults        0       0
UUID=2e0d9ec9-a82a-43cc-a8e2-c6db30e7f6a4       /boot   xfs     defaults        0       0
UUID=7B77-95E7  /boot/efi       vfat    defaults,uid=0,gid=0,umask=077,shortname=winnt  0       2
UUID="934fece5-074e-4102-8481-6ed3a3b10931"   /docker  xfs  defaults  0  0
UUID="52f4940d-3dca-4b90-a223-3bef6eee2b74"  /cdwdata   xfs     defaults        0       0
UUID="d7e9640b-cc12-49ac-b6c2-6eb3136679a1"  /lhdata   xfs     defaults        0       0

[root@pvcecs-master ~]# mount -av
[root@pvcecsmaster overlay2]# lvs
  LV       VG     Attr       LSize    Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
  home     centos -wi-ao----  183.24g
  root     centos -wi-ao----   50.00g
  swap     centos -wi-ao----   15.75g
  cdwdata vgcdp  -wi-ao----  600.00g
  docker  vgcdp  -wi-ao---- <600.00g
  var     vgcdp  -wi-ao---- <600.00g
  lhdata  vgcdp  -wi-ao----  1200.00g
[root@pvcecsmaster overlay2]#

[root@pvcecsmaster overlay2]# pvs
  PV         VG     Fmt  Attr PSize    PFree
  /dev/sda2  centos lvm2 a--  <249.00g    4.00m
  /dev/sdb   vgcdp  lvm2 a--  <600.00g <224.96g
  /dev/sdc   vgcdp  lvm2 a--  <600.00g <224.96g
  /dev/sdd   vgcdp  lvm2 a--  <1200.00g<224.96g
  /dev/sde   vgcdp  lvm2 a--  <600.00g <224.96g
[root@pvcecs-master ~]#
```

Procedure 19. Install httpd on Cloudera-Manager node i.e. cldr-mngr to host a local Parcel repository

Setting up the RHEL repository on the cloudera-manager node requires httpd.

Step 1. Install httpd on the cloudera-manager i.e. **cldr-mngr** node to host repositories:

Note: The Red Hat repository is hosted using HTTP on the admin node; this machine is accessible by all the hosts in the cluster.

```
[root@cldr-mngr ~]# dnf install -y httpd mod_ssl createrepo_c
```

Step 2. Generate CA certificate.

```
[root@cldr-mngr ~]# openssl req -newkey rsa:2048 -nodes -keyout /etc/pki/tls/private/httpd.key -x509 -days
3650 -out /etc/pki/tls/certs/httpd.crt \
-subj "/C=US/ST=North Carolina/L=Raleigh/O=Red Hat Inc/OU=CLDR/CN=cldr-mngr.cldrsetup.local"
...+.+..+...+.+......+...+............+..++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
*.......+.....++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.+.....+....+.........+...
..+......+...+.......+.....+....+..+....+......+..+......+....+...+..+.......+.............+.+..........
....+...+..+.........+....+..+...+.+..........+...+.+.....+.......+..+.......+........+.......+.........+
..+....+...........+...+.+......+........+....+......+......+...+...........+......+.....+...+......+.+..
.+.............+.+...........+........+.....+..+.......+..+.............+..+...+.+.............+...
.+...+.+.....+.........+.........+...........+.........+......+...............+.+..+.......+......+...
.........+...+.............+......+.....+.+..+.+......+........+......+.......+.....+...+............
....+...........+..........+..+...+....+...+...............+......+.........+.......+...+..+.+...+......
.....++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
...+.........+...........+.+.............+...+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++*....+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.+.......+...+..+...........+
....+.....+......+...+.+..+...........+.+......+...+...+...+......+.....+.................+......+...+.
.+.+.....+....+.................+..+.+.....+....+.............+...+.....+...+.................+.+.....
......+...+.+.....+.................+.........+...........+........+......+....+...........+.+........+
.............+..+...+.......+...+.............+...........+......+.........+..........+.........+.....
........+..+.+......+.........+..........+.+.....+....................+.............+..+....+++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++
-----

[root@cldr-mngr ~]# ls -l /etc/pki/tls/private/ /etc/pki/tls/certs/
/etc/pki/tls/certs/:
total 8
lrwxrwxrwx. 1 root root   49 Jul 28  2022 ca-bundle.crt ->
/etc/pki/ca-trust/extracted/pem/tls-ca-bundle.pem
lrwxrwxrwx. 1 root root   55 Jul 28  2022 ca-bundle.trust.crt ->
/etc/pki/ca-trust/extracted/openssl/ca-bundle.trust.crt
-rw-r--r--. 1 root root 1432 Mar  4 13:34 httpd.crt
-rw-r--r--. 1 root root 2260 Mar  1 16:36 postfix.pem

/etc/pki/tls/private/:
total 8
-rw-------. 1 root root 1700 Mar  4 13:33 httpd.key
-rw-------. 1 root root 3268 Mar  1 16:36 postfix.key
[root@cldr-mngr ~]#
```

Step 3. Create certificate directory to server content from.

```
[root@cldr-mngr ~]# mkdir -p /var/www/https/
[root@cldr-mngr ~]# echo secure content > /var/www/https/index.html
[root@cldr-mngr ~]# cat /var/www/https/index.html
secure content
```

Step 4. Edit httpd.conf file; add ServerName and make the necessary changes to the server configuration file:

```
[root@cldr-mngr ~]# vi /etc/httpd/conf/httpd.conf
ServerName cldr-mngr.cldrsetup.local:80
```

Step 5. Start httpd service.

```
[root@cldr-mngr ~]# systemctl start httpd
[root@cldr-mngr ~]# systemctl enable httpd
[root@cldr-mngr ~]# systemctl is-enabled httpd
```

****

Install Cloudera Data Platform Private Cloud (Cloudera on premises)

This chapter contains the following:

- Cloudera Runtime
- Install Cloudera on premises Base **7.3.1.400 SP2**
- Install CDP Data Services **1.5.5 CHF1**

Cloudera Runtime

Cloudera Runtime is the core open-source software distribution within Cloudera on premises Base. Cloudera Runtime includes approximately 50 open-source projects that comprise the core distribution of data management tools within CDP.

For more information review Cloudera Runtime Release notes:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/runtime-release-notes/topics/rt-Private Cloud-whats-new.html

Please review runtime cluster hosts and role assignments:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-runtime-cluster-hosts-role-assignments.html

Cloudera Data Platform Private Cloud Installation Requirements (Pre-requisites):

NTP/Chrony

Both Cloudera on premises Base and Cloudera on premises DS cluster should have their time synched with the NTP Clock time from the same NTP source. Also make sure, Active Directory server where Kerberos is setup for data lake and for other services must also be synced with the same NTP source.

JDK 11

The cluster must be configured with JDK 11, JDK8 is not supported. You can use Oracle, OpenJDK 11.04, or higher. JAVA 11 is a JKS requirement and must be met. In this setup we used OpenJDK 17.0.13.

Kerberos

Kerberos must be configured using an Active Directory (AD), RedHat FreeIPA or MIT KDC. Kerberos will be enabled for all services in the cluster.

Database Requirements

Cloudera Manager and Runtime come packaged with an embedded PostgreSQL database for use in non-production environments. The embedded PostgreSQL database is not supported in production environments. For production environments, you must configure your cluster to use dedicated external databases.

For detailed information about supported database go to: https://supportmatrix.cloudera.com/

Configure Cloudera Manager with TLS/SSL

TLS/SSL provides privacy and data integrity between applications communicating over a network by encrypting the packets transmitted between endpoints (ports on a host, for example). Configuring TLS/SSL for any system typically involves creating a private key and public key for use by server and client processes to negotiate an encrypted connection at runtime. In addition, TLS/SSL can use certificates to verify the trustworthiness of keys presented during the negotiation to prevent spoofing and mitigate other potential security issues.

For detailed information on encrypting data in transit, go to:

https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/security-encrypting-data-in-transit/topics/cm-security-guide-ssl-certs.html

The Auto-TLS feature automates all the steps required to enable TLS encryption at a cluster level. Using Auto-TLS, you can let Cloudera manage the Certificate Authority (CA) for all the certificates in the cluster or use the company's existing CA. In most cases, all the necessary steps can be enabled easily via the Cloudera Manager UI. This feature automates the following processes when Cloudera Manager is used as a Certificate Authority:

- Creates the root Certificate Authority or a Certificate Signing Request (CSR) for creating an intermediate Certificate Authority to be signed by company's existing Certificate Authority (CA)
- Generates the CSRs for hosts and signs them

Configuring TLS Encryption for Cloudera Manager Using Auto-TLS for detailed information:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/security-encrypting-data-in-transit/topics/cm-security-how-to-configure-cm-tls.html

Manually Configuring TLS Encryption for Cloudera Manager for detailed information:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/security-encrypting-data-in-transit/topics/cm-security-how-to-configure-cm-tls.html

TLS uses JKS-format (Java KeyStore)

Cloudera Manager Server, Cloudera Management Service, and many other CDP services use JKS formatted key-stores and certificates. Java 11 is required for JKS.

Licensing Requirements

The cluster must be setup with a license with entitlements for installing Cloudera on premises. 60 days evaluation license for Cloudera Data Platform Cloudera on premises Base does not allow you to set up Cloudera on premises Data Services.

Refer to the Cloudera on premises Base Requirements and Supported Versions for information about hardware, operating system, and database requirements, as well as product compatibility matrices.

Refer Cloudera Manager release note for new feature and support:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/manager-release-notes/topics/cm-whats-new-7113.html

Please review before install steps:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-before-you-install.html

Please review Cloudera on premises Base requirements and supported versions for information about hardware, operating system, and database requirements, as well as product compatibility matrices:

https://docs.cloudera.com/cdp-private-cloud-upgrade/latest/upgrade/topics/cdpdc-requirements-supported-versions.html

****

Cloudera on premises Cloudera Manager Server Setup

This section outlines the steps needed to set up a 6 node Cloudera on premises Base cluster. Below are the prerequisites which base cluster should have before installing/configuring Data Services.
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-installation.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-prod-installation.html

Ensure to verify compatibility matrix of Cloudera-Manager, Cloudera RunTime, DataServices/ECS, JDK, Python, PostgreSQL etc. all together:
https://supportmatrix.cloudera.com/

Procedure 1. Setup Cloudera Manager Repository

Note: These steps require a Cloudera username and password to access: https://archive.cloudera.com/p/cm7/

Step 1: From a host connected to the Internet, download the Cloudera's repositories as shown below and transfer it to the cldr-mngr node. We will directly login to ***cldr-mngr*** and perform below steps::

```
[root@ipaserver ~]# ssh root@cldr-mngr
[root@cldr-mngr ~]# mkdir -p /var/www/html/cloudera-repos/cloudera-manager/
[root@cldr-mngr ~]# cd /var/www/html/cloudera-repos/cloudera-manager/
```

Step 2: Download Cloudera Manager Repository:

```
#!/bin/bash

set -e

# ======================================================
#  Cloudera Manager Repository Download Script
# ======================================================

# ================================
# 1. Set Cloudera Archive Credentials
# ================================

USERNAME="<paywall-username>"
PASSWORD="<paywall-password>"

# ================================
# 2. Define Cloudera Manager Version & Build
# ================================

CM_VERSION="7.13.1.400"
BUILD_NUMBER="68000784"

# ================================
# 3. Define Base URL
# ================================

BASE_URL="https://${USERNAME}:${PASSWORD}@archive.cloudera.com/p/cm7/${CM_VERSION}/redhat9/yum"

# ================================
# 4. Prepare Target Local Repo Path
# ================================

cd /var/www/html/cloudera-repos/cloudera-manager/

# ================================
# 5. Download Repo & GPG Key Files
# ================================

wget ${BASE_URL}/cloudera-manager.repo
wget ${BASE_URL}/cloudera-manager-trial.repo
```

```
wget ${BASE_URL}/RPM-GPG-KEY-cloudera
wget https://${USERNAME}:${PASSWORD}@archive.cloudera.com/p/cm7/${CM_VERSION}/allkeys.asc
wget https://${USERNAME}:${PASSWORD}@archive.cloudera.com/p/cm7/${CM_VERSION}/allkeyssha256.asc

# ================================
# 6. Done
# ================================
echo "✔ Cloudera Manager repo and keys downloaded successfully."


# ================================
# 7. Verify Downloaded Files
# ================================
ls -lah

[root@cldr-mngr cloudera-manager]# pwd
/var/www/html/cloudera-repos/cloudera-manager

[root@cldr-mngr cloudera-manager]# ll
total 36
-rw-r--r--. 1 root root  2464 Apr 23 06:51 RPM-GPG-KEY-cloudera
-rw-r--r--. 1 root root 11019 Apr 23 06:51 allkeys.asc
-rw-r--r--. 1 root root  4901 Apr 23 06:51 allkeyssha256.asc
-rw-r--r--. 1 root root   248 Apr 14 11:16 cloudera-manager-trial.repo
-rw-r--r--. 1 root root   501 Apr 23 06:51 cloudera-manager.repo

[root@cldr-mngr cloudera-manager]#
```

Step 3: Edit cloudera-manager.repo file baseurl and GPG key with username and password provided by Cloudera and edit URL to match repository location **(OR)** Verify, if username and password are already present, so no action needed.

```
##### Verify if username and password are already present, so no action needed.
[root@cldr-mngr cloudera-manager]# vi cloudera-manager.repo
[cloudera-manager]
name=Cloudera Manager 7.13.1.400
baseurl=https://archive.cloudera.com/p/cm7/7.13.1.400/redhat9/yum/
gpgkey=https://archive.cloudera.com/p/cm7/7.13.1.400/redhat9/yum/RPM-GPG-KEY-cloudera
username=<username>
password=<password>
gpgcheck=1
enabled=1
autorefresh=0
type=rpm-md

[postgresql10]
name=Postgresql 10
baseurl=https://archive.cloudera.com/postgresql10/redhat9/
gpgkey=https://archive.cloudera.com/postgresql10/redhat9/RPM-GPG-KEY-PGDG-10
enabled=1
gpgcheck=1
module_hotfixes=true

##### If not, update the cloudera-manager.repo to look like below.
[root@cldr-mngr cloudera-manager]# vi cloudera-manager.repo
[cloudera-manager]
name=Cloudera Manager 7.13.1.400
baseurl=https://<username>:<password>@archive.cloudera.com/p/cm7/7.13.1.400/redhat9/yum/
gpgkey=https://<username>:<password>@@archive.cloudera.com/p/cm7/7.13.1.400/redhat9/yum/RPM-GPG-KEY-cloudera
gpgcheck=1
enabled=1
autorefresh=0
type=rpm-md
[root@cldr-mngr cloudera-manager]# cd
```

Step 4: Create directory to download cloudera manager agent, daemon, and server files

```
#!/bin/bash

set -e
```

```
# ========================================================
#  Cloudera Manager RPM Download Script
# ========================================================

# ================================
# 1. Create & Navigate to RPM Directory
# ================================

mkdir -p cm${CM_VERSION}/redhat9/yum/RPMS/x86_64/
cd cm${CM_VERSION}/redhat9/yum/RPMS/x86_64/

# ================================
# 2. Download Required RPMs
# ================================

for pkg in agent daemons server server-db-2; do
  wget ${BASE_URL}/RPMS/x86_64/cloudera-manager-${pkg}-${CM_VERSION}-${BUILD_NUMBER}.el9.x86_64.rpm
done

# ================================
# 3. Verify Downloaded Files
# ================================

ls -alh

[root@cldr-mngr x86_64]# pwd
/var/www/html/cloudera-repos/cloudera-manager/cm7.13.1/redhat9/yum/RPMS/x86_64
[root@cldr-mngr x86_64]# ll
total 1862092
-rw-r--r--. 1 root root  116384320 Apr 23 06:52 cloudera-manager-agent-7.13.1.400-68000784.el9.x86_64.rpm
-rw-r--r--. 1 root root 1790367515 Apr 23 06:52 cloudera-manager-daemons-7.13.1.400-68000784.el9.x86_64.rpm
-rw-r--r--. 1 root root      20933 Apr 23 06:52 cloudera-manager-server-7.13.1.400-68000784.el9.x86_64.rpm
-rw-r--r--. 1 root root      20933 Apr 23 06:52 cloudera-manager-server-db-7.13.1.400-68000784.el9.x86_64.rpm
[root@cldr-mngr x86_64]#
```

Step 5: Run createrepo command to create a local repository.

```
[root@cldr-mngr ~]# createrepo --baseurl http://$(hostname -i)/cloudera-repos/cloudera-manager/
/var/www/html/cloudera-repos/cloudera-manager/
```

Note: In a web browser please check and verify cloudera manager repository created by entering baseurl http://13.251.65.11/cloudera-repos/cloudera-manager/

Step 6: Copy cloudera-manager.repo file to /etc/yum.repos.d/ on all nodes to enable it to find the packages that are locally hosted on the admin node.

```
[root@cldr-mngr ~]# cp /var/www/html/cloudera-repos/cloudera-manager/cloudera-manager.repo
/etc/yum.repos.d/cloudera-manager.repo
```

Step 7: Edit cloudera-manager.repo. file as per the customer repository location configuration in the step above. Copy the updated repo file to the ipaserver node so it can be copied to the rest of servers using ansible.

```
[root@cldr-mngr ~]# vi /etc/yum.repos.d/cloudera-manager.repo
[cloudera-manager]
name=Cloudera Manager 7.13.1.400
baseurl=http://<ip_of_cldr_mngr>/cloudera-repos/cloudera-manager/
#Update IP of Repo/cldr-mngr server
gpgcheck=0
enabled=1
[root@cldr-mngr ~]# scp -r /etc/yum.repos.d/cloudera-manager.repo root@ipaserver:/etc/yum.repos.d/cloudera-manager.repo
```

Step 8: From the ansible control node copy the repo files to /etc/yum.repos.d/ of all the nodes of the cluster:

```
[root@ipaserver ~]# ansible all -m copy -a "src=/etc/yum.repos.d/cloudera-manager.repo
dest=/etc/yum.repos.d/cloudera-manager.repo"
```

Procedure 2. Set Up the Local Parcels for Cloudera on premises Base 7.3.1

From a host connected the internet, download Cloudera on premises Base 7.3.1 parcels for RHEL9 from the URL:

Step 1. Create a directory and Download CDH parcels as shown below:

```
#!/bin/bash

set -e

# ================================
# CDH Parcels Download Script
# ================================

# Credentials
USERNAME="<paywall-username>"
PASSWORD="<paywall-password>"

# CDH Version & Build
CDH_VERSION="7.3.1"
PARCEL_VERSION="7.3.1.400"
BUILD_NUMBER="67986116"    # must match build ID from Cloudera archive

# Base URL
BASE_URL="https://${USERNAME}:${PASSWORD}@archive.cloudera.com/p/cdh7/${PARCEL_VERSION}/parcels"

# Local repo directory
TARGET_DIR="/var/www/html/cloudera-repos/cdh${CDH_VERSION}"
mkdir -p "${TARGET_DIR}"
cd "${TARGET_DIR}"

# ================================
# Download parcels
# ================================
echo "Downloading CDH parcels for version ${CDH_VERSION}..."

# CDH parcel + checksums
wget -q
${BASE_URL}/CDH-${CDH_VERSION}-1.cdh${CDH_VERSION}.p${PARCEL_VERSION##*.}.${BUILD_NUMBER}-el9.parcel
wget -q
${BASE_URL}/CDH-${CDH_VERSION}-1.cdh${CDH_VERSION}.p${PARCEL_VERSION##*.}.${BUILD_NUMBER}-el9.parcel.sha1
wget -q
${BASE_URL}/CDH-${CDH_VERSION}-1.cdh${CDH_VERSION}.p${PARCEL_VERSION##*.}.${BUILD_NUMBER}-el9.parcel.sha25
6

# Key Trustee parcel + checksums
wget -q
${BASE_URL}/KEYTRUSTEE_SERVER-${PARCEL_VERSION}-1.keytrustee${PARCEL_VERSION}.p0.${BUILD_NUMBER}-el9.parce
l
wget -q
${BASE_URL}/KEYTRUSTEE_SERVER-${PARCEL_VERSION}-1.keytrustee${PARCEL_VERSION}.p0.${BUILD_NUMBER}-el9.parce
l.sha
wget -q
${BASE_URL}/KEYTRUSTEE_SERVER-${PARCEL_VERSION}-1.keytrustee${PARCEL_VERSION}.p0.${BUILD_NUMBER}-el9.parce
l.sha1
wget -q
${BASE_URL}/KEYTRUSTEE_SERVER-${PARCEL_VERSION}-1.keytrustee${PARCEL_VERSION}.p0.${BUILD_NUMBER}-el9.parce
l.sha256

# Manifest
wget -q ${BASE_URL}/manifest.json

# ================================
# Set permissions
# ================================
chmod -R ugo+rX "${TARGET_DIR}"

# ================================
# Verify downloaded files
# ================================
```

```
echo "Downloaded parcels:"
ls -lah "${TARGET_DIR}"

[root@cldr-mngr ~]# ll /var/www/html/cloudera-repos/cdh7/7.3.1.400/
total 8453728
-rw-r--r--. 1 root root 8656532124 Aug 21 15:12 CDH-7.3.1-1.cdh7.3.1.p400.67986116-el9.parcel
-rw-r--r--. 1 root root         40 Aug 21 15:12 CDH-7.3.1-1.cdh7.3.1.p400.67986116-el9.parcel.sha1
-rw-r--r--. 1 root root         64 Aug 21 15:12 CDH-7.3.1-1.cdh7.3.1.p400.67986116-el9.parcel.sha256
-rw-r--r--. 1 root root      70985 Aug 21 15:12 manifest.json
-rw-r--r--. 1 root root 1234567890 Aug 21 15:12
KEYTRUSTEE_SERVER-7.3.1.400-1.keytrustee7.3.1.400.p0.67986116-el9.parcel
-rw-r--r--. 1 root root         40 Aug 21 15:12
KEYTRUSTEE_SERVER-7.3.1.400-1.keytrustee7.3.1.400.p0.67986116-el9.parcel.sha
-rw-r--r--. 1 root root         64 Aug 21 15:12
KEYTRUSTEE_SERVER-7.3.1.400-1.keytrustee7.3.1.400.p0.67986116-el9.parcel.sha1
-rw-r--r--. 1 root root         96 Aug 21 15:12
KEYTRUSTEE_SERVER-7.3.1.400-1.keytrustee7.3.1.400.p0.67986116-el9.parcel.sha256

[root@cldr-mngr ~]#
```

Note: In a web browser please check and verify cloudera manager repository created by entering baseurl: http://13.251.65.11/cloudera-repos/cdh7.3.1/ (IP is of Cloudera-Manager)

Procedure 3. Set Up the Local Parcels for CDS 3.3 powered by Apache Spark

Step 1. From a host connected the internet, download CDS 3.3 Powered by Apache Spark parcels for RHEL9 from the URL: https://archive.cloudera.com/p/spark3/3.3.7191000.4/parcels/

Note: Although Spark 2 and Spark 3 can coexist in the same Cloudera on premises Base cluster, you cannot use multiple Spark 3 versions simultaneously. All clusters managed by the same Cloudera Manager Server must use exactly the same version of CDS 3.3 Powered by Apache Spark.

Step 2. Create a directory and download CDS parcels as shown below:

```
#!/bin/bash

set -e

# ================================
# CDS Parcels Download Script
# ================================

# ============================
# Set Variables
# ============================
USERNAME="<paywall-username>"
PASSWORD="<paywall-password>"
SPARK_VERSION="3.5.7191000.0"
SPARK_UPSTREAM="3.5.4"
BUILD_NUMBER="68499982"
BASE_URL="https://${USERNAME}:${PASSWORD}@archive.cloudera.com/p/spark3/${SPARK_VERSION}/parcels"

# ============================
# Create Local Repo Directory
# ============================
mkdir -p /var/www/html/cloudera-repos/spark3/${SPARK_VERSION}
cd /var/www/html/cloudera-repos/spark3/${SPARK_VERSION}

# ============================
# Download Spark3 Parcels and Manifest
# ============================
wget "${BASE_URL}/SPARK3-${SPARK_UPSTREAM}.${SPARK_VERSION}-30-1.p0.${BUILD_NUMBER}-el9.parcel"
wget "${BASE_URL}/SPARK3-${SPARK_UPSTREAM}.${SPARK_VERSION}-30-1.p0.${BUILD_NUMBER}-el9.parcel.sha1"
wget "${BASE_URL}/manifest.json"

# ============================
# Set Permissions
```

```
# ============================
chmod -R ugo+rX /var/www/html/cloudera-repos/spark3/

# ============================
# Verify
# ============================
ls -lh /var/www/html/cloudera-repos/spark3/${SPARK_VERSION}

[root@cldr-mngr ~]# ll /var/www/html/cloudera-repos/spark3/3.5.7191000.0/
total 1999000
-rw-r--r--. 1 root root 2046842555 Aug 21 15:20 SPARK3-3.5.4.3.5.7191000.0-30-1.p0.68499982-el9.parcel
-rw-r--r--. 1 root root         41 Aug 21 15:20 SPARK3-3.5.4.3.5.7191000.0-30-1.p0.68499982-el9.parcel.sha1
-rw-r--r--. 1 root root         64 Aug 21 15:20 SPARK3-3.5.4.3.5.7191000.0-30-1.p0.68499982-el9.parcel.sha256
-rw-r--r--. 1 root root       8962 Aug 21 15:20 manifest.json

[root@cldr-mngr ~]#
```

Step 4. In a web browser please check and verify cloudera manager repository created by entering baseurl:
http://13.251.65.11/cloudera-repos/spark3

Procedure 4. Install Python 3.9 ***see if directly python3 could be installed by dnf (Skip if installed previously, as we did)

For support and requirement on minimum python version please refer to:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-cm-install-python-3.9.html

Note: Python 3.9 is the default Python implementation provided by RHEL 9 and is usually installed by default. Perform this task to install or re-install it manually. We will run below ansible commands from ipaserver/ ansible control node. Rest of commands need to be run manually on each server.

Step 1. To install Python 3.9 standard package on RHEL 9 run following command:

```
[root@ipaserver ~]# ansible all -m shell -a "dnf install -y python3 python3-pip"
[root@ipaserver ~]# ansible all -m shell -a "python3 --version"
Python 3.9.22
[root@ipaserver ~]#
```

To install standard Python 3.9 binary on RHEL9 at standard or custom location, Follow steps below:

Step 2. Install the following packages before installing Python 3.9 from ansible control node:

```
[root@ipaserver ~]# ansible all -m shell -a "sudo dnf install gcc openssl-devel bzip2-devel libffi-devel
zlib-devel -y"
```

Step 3. Download Python 3.9 and decompress the package by running the following commands:

```
[root@ipaserver ~]# ansible all -m shell -a "cd /opt/ && curl -O
https://www.python.org/ftp/python/3.9.22/Python-3.9.22.tgz && tar -zxvf Python-3.9.22.tgz"
```

Step 4. Go to decompressed Python directory and Install Python 3.9 as follows:

```
[root@ipaserver ~]# ansible all -m shell -a "cd /opt/Python-3.9.22/ && ./configure --enable-optimizations
--enable-shared"
```

Note: By default, Python could be installed in any one of the following locations. If you are installing Python 3.9 in any other location, then you must specify the path using the --prefix option.

```
/usr/bin
/usr/local/python39/bin
/usr/local/bin
/opt/rh/rh-python39/root/usr/bin
```

Note: The --enabled-shared option is used to build a shared library instead of a static library.

```
[root@ipaserver ~]# ansible all -m shell -a "echo $LD_LIBRARY_PATH"
[root@ipaserver ~]# ansible all -m shell -a "echo 'export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib/' >> ~/.bashrc && source ~/.bashrc"
[root@ipaserver ~]# ansible all -m shell -a "cd /usr/local/bin/ && ls -l"
```

Step 5. Built Python 3.9 as follows:

```
[root@ipaserver ~]# ansible all -m shell -a "cd /opt/Python-3.9.22/ && make"
```

Step 6. Run the following command to put the compiled files in the default location or in the custom location that you specified using the --prefix option:

```
[root@ipaserver ~]# ansible all -m shell -a "cd /opt/Python-3.9.22/ && make install"
```

Step 7. Copy the shared compiled library files (libpython3.9.so) to the /lib64/ directory:

```
[root@ipaserver ~]# ansible all -m shell -a "cd /opt/Python-3.9.22/ && cp --no-clobber ./libpython3.9.so*
/lib64/"
```

Step 8. Change the permissions of the libpython3.9.so files as follows:

```
[root@ipaserver ~]# ansible all -m shell -a "chmod 755 /lib64/libpython3.9.so*"
```

Step 9. If you see an error such as error while loading shared libraries: libpython3.9.so.1.0: cannot open shared object file: No such file or directory, then run the following command:

```
[root@ipaserver ~]# ansible all -m shell -a "echo 'export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib/' >> ~/.bashrc && source ~/.bashrc"
```

Step 10. ***(For Hue)*** If you have installed Python 3.9 at a custom location, then you must append the custom path in Cloudera Manager > Clusters > Hue > Configuration > Hue Service Environment Advanced Configuration Snippet (Safety Valve) separated by colon (:) as follows ***(Later- after Base Cluster Installation)***:

```
Key: PATH
Value: [***CUSTOM-INSTALL-PATH***]:/usr/local/sbin:/usr/local/bin:/usr/sbin:
```

Step 11. Check Python version

```
[root@ipaserver ~]# ansible all -m command -a "python3 --version"
pvcbase-worker2.cldrsetup.local | CHANGED | rc=0 >>
Python 3.9.22
pvcbase-worker3.cldrsetup.local | CHANGED | rc=0 >>
Python 3.9.22
pvcbase-master.cldrsetup.local | CHANGED | rc=0 >>
Python 3.9.22
```

Procedure 5. Install and Configure Database for Cloudera Manager

Cloudera Manager uses various databases and datastores to store information about the Cloudera Manager configuration, as well as information such as the health of the system, or task progress.

Please review Database Requirement for Cloudera on premises Base.

This procedure highlights the installation and configuration steps with PostgreSQL. Please review Install and Configure Databases for Cloudera on premises Base for more details:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-install-config-postgresql-for-cdp.html

Note: If you already have a PostgreSQL database set up, you can skip to the section Configuring and Starting the PostgreSQL Server to verify that your PostgreSQL configurations meet the requirements for Cloudera Manager.
Note: We will be installing the external PostgreSQL DB server on the cldr-mngr host.

Step 1. Login on ***cldr-mngr server*** and Install PostgreSQL as shown in the steps below.

```
##### Install and configure POSTGRESQL DB on cldr-mngr server :
**** When you restart any process in future, the configuration for each of the services is redeployed
using information saved in the Cloudera Manager database. If this information is not available, your
cluster cannot start or function correctly. So, you must schedule and maintain regular backups of the
Cloudera Manager database to recover the cluster in the event of the loss of this database.
```

```
##### Install the repository RPM:
[root@cldr-mngr ~]# sudo dnf install -y
https://download.postgresql.org/pub/repos/yum/reporpms/EL-9-x86_64/pgdg-redhat-repo-latest.noarch.rpm

##### Disable the built-in PostgreSQL module:
[root@cldr-mngr ~]# sudo dnf -qy module disable postgresql

##### Install PostgreSQL:
[root@cldr-mngr ~]# sudo dnf install -y postgresql14 postgresql14-server postgresql14-libs
[root@cldr-mngr ~]#
```

Step 2. Install the PostgreSQL JDBC driver by running the following command on ansible-controller/ipaserver node. Rename the Postgres JDBC driver .jar file to postgresql-connector-java.jar and copy it to the /usr/share/java directory. The following copy command can be used if the Postgres JDBC driver .jar file is installed from the OS repositories: ***(Skip this, as we already performed it in prior steps)***

```
##### Install JDBC Connector
[root@ipaserver ~]# wget https://jdbc.postgresql.org/download/postgresql-42.7.7.jar

[root@ipaserver ~]# mv -v postgresql-42.7.7.jar postgresql-connector-java.jar

[root@ipaserver ~]# ansible all -m copy -a "src=postgresql-connector-java.jar
dest=/usr/share/java/postgresql-connector-java.jar"

[root@ipaserver ~]# ansible all -m shell -a "sudo chmod 644 /usr/share/java/postgresql-connector-java.jar"

[root@ipaserver ~]# ansible all -m shell -a "sudo ls -l /usr/share/java/postgresql*.jar"
ipaserver.cldrsetup.local | CHANGED | rc=0 >>
-rw-r--r-- 1 root root 1089312 Jun  3 08:06 /usr/share/java/postgresql-connector-java.jar

##### Alternate way (Not recommended)
[root@ipaserver ~]# ansible all -m shell -a "sudo dnf install postgresql-jdbc -y"
[root@ipaserver ~]# ansible all -m shell -a "java -version"
```

Step 4. Make sure that the data directory, which by default is /var/lib/pgsql/17/data/, is on a partition that has sufficient free space.

Note: Cloudera Manager supports the use of a custom schema name for the Cloudera Manager Server database. By default, PostgreSQL only accepts connections on the loopback interface. You must reconfigure PostgreSQL to accept connections from external hosts.

Step 5. Installing the psycopg2 Python package for PostgreSQL-backed Hue. ***(Skip this, as we already performed it in prior steps)***

Note: If you are installing Runtime 7 and using PostgreSQL as a backend database for Hue, then you must install the 2.9.3 version (or greater) of the psycopg2 package on all Hue hosts. The psycopg2 package is automatically installed as a dependency of Cloudera Manager Agent, but the version installed is often lower than 2.9.3

Step 6. Make sure the psycopg2 package dependencies for RHEL 9 is installed on all required hosts, by running the following commands:

```
##### Install the psycopg2-binary package as follows:
[root@ipaserver ~]# ansible all -m shell -a "pip3 install psycopg2-binary && pip3 list | grep psyc"

ipaserver.cldrsetup.local | CHANGED | rc=0 >>
Requirement already satisfied: psycopg2-binary in /usr/local/lib64/python3.9/site-packages (2.9.10)
psycopg2-binary      2.9.10
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with
the system package manager. It is recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv

[root@ipaserver ~]#
```

Step 7. Initialize the database:

```
# Initialize the DB
[root@cldr-mngr ~]# sudo /usr/pgsql-14/bin/postgresql-17-setup initdb
```

```
# Verify PG Version
[root@cldr-mngr ~]# cat /var/lib/pgsql/17/data/PG_VERSION
17

# Verify Psycopg Version
[root@cldr-mngr ~]# pip3 list |grep psycopg
psycopg2-binary      2.9.10

# data directory is very critical, if you want to cleanup postgres simply rename or remove
/var/lib/pgsql/14/data directory
```

Step 8. Make sure that LC_ALL is set to C.UTF-8 to enable UTF-8 CHARSET and initialize the database as follows:

```
[root@cldr-mngr ~]# echo 'LC_ALL="C.UTF-8"' >> /etc/locale.conf
```

Step 9. To enable MD5 authentication, edit /var/lib/pgsql/17/data/pg_hba.conf by adding the following lines, to enable connection from all outside hosts:
(Enable md5 auth to serve password authentication and TLS/SSL encryption from outside world)

```
[root@cldr-mngr ~]# vi /var/lib/pgsql/17/data/pg_hba.conf
host    all             all             0.0.0.0/0               md5 # Enable md5 authentication
host    ranger          rangeradmin     0.0.0.0/0               md5 # Allow ranger database connection
from any host
hostssl all             all             0.0.0.0/0               md5 # Allow SSL connection from client(s)
# replace 127.0.0.1 with host IP if PostgreSQL access from a different host is required.
# Edit section for replication privilege. HA not documented in this solution.

##### Backup the config so you can use it in case of re-setup.
[root@cldr-mngr ~]# cp /var/lib/pgsql/17/data/pg_hba.conf ~

##### If you have the file backed up, copy it
[root@cldr-mngr ~]# cp /var/lib/pgsql/17/data/pg_hba.conf /var/lib/pgsql/17/data/pg_hba.conf_orig
[root@cldr-mngr ~]# cp ~/pg_hba.conf /var/lib/pgsql/17/data/pg_hba.conf
```

Step 10. Configure settings to ensure your system performs as expected. Update these settings in the /var/lib/pgsql/17/data/postgresql.conf file. Settings vary based on cluster size and resources as follows:

```
[root@cldr-mngr ~]# vi /var/lib/pgsql/17/data/postgresql.conf
port = 5432                             # (change requires restart)     #### uncomment
listen_addresses = '*'                  # what IP address(es) to listen on;
max_connections = 1000                  # (change requires restart)
shared_buffers = 1024MB                 # min 128kB
wal_buffers = 16MB                      # min 32kB, -1 sets based on shared_buffers
max_wal_size = 6GB
min_wal_size = 512MB
checkpoint_completion_target = 0.9      # checkpoint target duration, 0.0 - 1.0 #### uncomment
standard_conforming_strings = off
jit = off

##### Backup the config so you can use it in case of re-setup.
[root@cldr-mngr ~]# cp /var/lib/pgsql/17/data/postgresql.conf ~

##### If you have the file backed up, copy it
[root@cldr-mngr ~]# cp /var/lib/pgsql/17/data/postgresql.conf /var/lib/pgsql/17/data/postgresql.conf_orig
[root@cldr-mngr ~]# cp ~/postgresql.conf /var/lib/pgsql/17/data/postgresql.conf
```

Note: Settings vary based on cluster size and resources.

Step 11. Start the PostgreSQL Server and configure it to start at boot.

```
[root@cldr-mngr ~]# systemctl start postgresql-17.service
[root@cldr-mngr ~]# systemctl enable postgresql-17.service
[root@cldr-mngr ~]# systemctl status postgresql-17.service -l
[root@cldr-mngr ~]# netstat -ltnupa | grep LIST | grep -E '5432|postgres'
```

Step 12. Create or verify login

```
[root@cldr-mngr ~]# sudo -u postgres psql
could not change directory to "/root": Permission denied
```

```
psql (14.13)
Type "help" for help.

postgres=# ALTER USER postgres PASSWORD 'postgres';
ALTER ROLE
postgres=# \q

[root@cldr-mngr ~]# psql -h cldr-mngr.cldrsetup.local -d postgres -U postgres
Password for user postgres:
psql (14.13)
Type "help" for help.

postgres=#\q
```

Step 13. Enable TLS 1.2 for PostgreSQL database before setting up Cloudera Manager.

```
##### Verify TLS is enabled or not:
[root@cldr-mngr ~]# sudo -u postgres psql
could not change directory to "/root": Permission denied
psql (14.13)
Type "help" for help.

postgres=# SELECT * FROM pg_stat_ssl;
  pid  | ssl | version | cipher | bits | client_dn | client_serial | issuer_dn
-------+-----+---------+--------+------+-----------+---------------+-----------
 41275 | f   |         |        |      |           |               |
(1 row)

postgres=# SHOW ssl;
 ssl
-----
 off
(1 row)

postgres=# \q

[root@cldr-mngr ~]# sudo dnf install -y mod_ssl

##### Stop Postgres DB service.
[root@cldr-mngr ~]# systemctl stop postgresql-17

[root@cldr-mngr ~]# cd /var/lib/pgsql/17/data/

##### Generate CA-signed certificates for clients to verify with openssl command line tool.
##### Update value for "-days 3650". Currently set for 3650 days = 10 years.

##### create a certificate signing request (CSR) and a public/private key file
[root@cldr-mngr data]# openssl req -new -nodes -text -out root.csr -keyout root.key -subj
'/C=US/ST=California/L=Santa Clara/O=Cloudera Inc/OU=CLDR/CN=cldr-mngr.cldrsetup.local'

##### Output for above command
[root@cldr-mngr data]# ls -ltr root*
total 8
-rw------- 1 root root 1704 Jun  3 07:43 root.key
-rw-r--r-- 1 root root 3589 Jun  3 07:43 root.csr
[root@cldr-mngr data]#

[root@cldr-mngr data]# chmod 400 root.key

[root@cldr-mngr data]# ls -ltr root*
total 8
-r-------- 1 root root 1704 Jun  3 07:43 root.key
-rw-r--r-- 1 root root 3589 Jun  3 07:43 root.csr
[root@cldr-mngr data]#

##### create a root certificate authority
[root@cldr-mngr data]# openssl x509 -req -in root.csr -text -days 3650 -extfile /etc/ssl/openssl.cnf
-extensions v3_ca -signkey root.key -out root.crt
Certificate request self-signature ok
```

```
subject=C = US, ST = California, L = Santa Clara, O = Cloudera Inc, OU = CLDR, CN =
cldr-mngr.cldrsetup.local

[root@cldr-mngr data]# ls -l
total 16
-r-------- 1 root root 1704 Jun  3 07:43 root.key
-rw-r--r-- 1 root root 3589 Jun  3 07:43 root.csr
-rw-r--r-- 1 root root 4592 Jun  3 07:46 root.crt
[root@cldr-mngr data]#

# create a server certificate signed by the new root certificate authority
[root@cldr-mngr data]# openssl req -new -nodes -text -out server.csr -keyout server.key -subj
"/CN=cldr-mngr.cldrsetup.local"

[root@cldr-mngr data]# ls -ltr root* server*
total 24
-r-------- 1 root root 1704 Jun  3 07:43 root.key
-rw-r--r-- 1 root root 3589 Jun  3 07:43 root.csr
-rw-r--r-- 1 root root 4592 Jun  3 07:46 root.crt
-rw------- 1 root root 1704 Jun  3 07:47 server.key
-rw-r--r-- 1 root root 3388 Jun  3 07:47 server.csr
[root@cldr-mngr data]#

[root@cldr-mngr data]# chmod 400 server.key

[root@cldr-mngr data]# ls -ltr root* server*
total 24
-r-------- 1 root root 1704 Jun  3 07:43 root.key
-rw-r--r-- 1 root root 3589 Jun  3 07:43 root.csr
-rw-r--r-- 1 root root 4592 Jun  3 07:46 root.crt
-r-------- 1 root root 1704 Jun  3 07:47 server.key
-rw-r--r-- 1 root root 3388 Jun  3 07:47 server.csr
[root@cldr-mngr data]#

[root@cldr-mngr data]# openssl x509 -req -in server.csr -text -days 3650 -CA root.crt -CAkey root.key
-CAcreateserial -out server.crt
Certificate request self-signature ok
subject=CN = cldr-mngr.cldrsetup.local

[root@cldr-mngr data]# ls -ltr root* server*
total 32
-r-------- 1 root root 1704 Jun  3 07:43 root.key
-rw-r--r-- 1 root root 3589 Jun  3 07:43 root.csr
-rw-r--r-- 1 root root 4592 Jun  3 07:46 root.crt
-r-------- 1 root root 1704 Jun  3 07:47 server.key
-rw-r--r-- 1 root root 3388 Jun  3 07:47 server.csr
-rw-r--r-- 1 root root   41 Jun  3 07:51 root.srl
-rw-r--r-- 1 root root 3933 Jun  3 07:51 server.crt
[root@cldr-mngr data]#

##### The above steps will create a server.crt and server.key file in that location.

[root@cldr-mngr data]# chown postgres:postgres server.crt server.key root.crt

##### Artifacts generated from above command:
[root@cldr-mngr data]# ls -l server\.* root\.*
-rw-r--r--. 1 postgres postgres 4586 Aug  9 14:34 root.crt
-rw-r--r--. 1 root     root     3584 Aug  9 14:33 root.csr
-r--------. 1 root     root     1704 Aug  9 14:33 root.key
-rw-r--r--. 1 root     root       41 Aug  9 14:37 root.srl
-rw-r--r--. 1 postgres postgres 3928 Aug  9 14:37 server.crt
-rw-r--r--. 1 root     root     3388 Aug  9 14:35 server.csr
-r--------. 1 postgres postgres 1704 Aug  9 14:35 server.key
[root@cldr-mngr data]#

##### Verify Key and Certs generated fine
[root@cldr-mngr data]# openssl rsa -noout -text -in server.key
Private-Key: (2048 bit, 2 primes)
modulus:
    00:b3:30:86:66:49:8d:c4:de:62:c6:17:e2:50:6c:
    88:91:10:49:26:6a:7f:a7:1d:6a:33:3a:71:0d:2c:
```

```
    f0:08:1b:3d:88:bc:73:43:b9:82:00:1a:a3:15:0f:
    08:ed:53:94:be:1e:25:7b:dd:99:66:c0:f5:2d:42:
    92:f0:d6:52:67:18:80:ab:a1:86:e1:aa:5c:53:47:
    41:3c:e2:2e:e1:dd:f8:5d:b7:e0:d0:39:26:f4:23:
    3d:78:71:9f:75:66:a0:0e:c7:9a:bc:c2:fb:db:1b:
    d1:fe:b2:2e:5d:a5:72:54:5f:04:54:1a:d8:76:77:
    a8:04:9d:05:9a:f6:25:5b:ed:73:88:6b:1a:e6:0f:
    09:62:d3:19:07:7c:2b:77:d0:5d:af:c3:bd:ff:44:
    7f:a9:08:b9:b2:e3:8c:5a:fd:90:dd:c7:bf:db:1e:
    c9:fe:72:16:e2:09:c2:0c:90:de:31:8b:06:58:e8:
    6c:37:7a:a4:bf:91:7e:ca:d4:15:60:d8:6f:b7:0b:
    e5:a1:5c:a2:30:98:d4:34:9c:69:88:57:f4:d1:b8:
    2a:1d:a1:c6:1f:5c:1d:10:56:5a:80:b5:5d:f3:f1:
    59:7f:4b:42:2c:82:3d:96:6d:5d:91:88:2a:de:12:
    6b:b4:65:f3:9d:c0:b8:02:4b:a6:21:bc:3b:5c:3f:
    32:3b
publicExponent: 65537 (0x10001)
privateExponent:
    12:78:80:8a:1f:af:dc:e8:bd:8e:c4:dc:7f:c4:c8:
    49:07:c0:3a:95:04:c6:91:aa:26:50:b2:61:94:cd:
    c3:50:27:86:26:42:cd:6a:dc:63:2d:5b:bd:2a:79:
    15:99:a5:7d:f9:76:8c:af:99:85:f5:82:f0:60:e9:
    eb:a8:74:03:0b:8c:0b:e5:11:15:c6:ed:50:6a:4a:
___
___
___
```

```
[root@cldr-mngr data]# openssl x509 -noout -text -in server.crt
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            40:26:f6:7b:84:d1:ad:30:65:2a:07:df:20:f8:4f:a3:91:0e:09:c7
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = US, ST = California, L = Santa Clara, O = Cloudera Inc, OU = CLDR, CN =
cldr-mngr.cldrsetup.local
        Validity
            Not Before: May 14 06:33:09 2024 GMT
            Not After : May 12 06:33:09 2034 GMT
        Subject: CN = cldr-mngr.cldrsetup.local
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b3:30:86:66:49:8d:c4:de:62:c6:17:e2:50:6c:
                    88:91:10:49:26:6a:7f:a7:1d:6a:33:3a:71:0d:2c:
                    f0:08:1b:3d:88:bc:73:43:b9:82:00:1a:a3:15:0f:
                    08:ed:53:94:be:1e:25:7b:dd:99:66:c0:f5:2d:42:
___
___
___
```

```
# Correct permissions for the private key file
[root@cldr-mngr data]# chmod 644 server.crt root.crt
[root@cldr-mngr data]# chmod 0600 /var/lib/pgsql/17/data/server.key

##### Edit Configuration file for PostgreSQL (postgresql.conf) to enable SSL
[root@cldr-mngr data]# cat <<EOF >> /var/lib/pgsql/17/data/postgresql.conf
ssl = on
ssl_ca_file = 'root.crt'
ssl_cert_file = 'server.crt'
ssl_key_file = 'server.key'
EOF

# Find the location of the private key file, typically in the data directory
[root@cldr-mngr data]# ls -l /var/lib/pgsql/17/data/server.key

##### Restart PostgreSQL database service to pick up the SSL related configuration changes and verify
login with SSL
[root@cldr-mngr data]# systemctl restart postgresql-17.service
[root@cldr-mngr data]# systemctl status postgresql-17.service -l
```

```
[root@cldr-mngr data]# psql -h cldr-mngr.cldrsetup.local -d postgres -U postgres
Password for user postgres: <postgres>
psql (14.13)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

postgres=# SELECT * FROM pg_stat_ssl;
  pid   | ssl | version |         cipher         | bits | client_dn | client_serial | issuer_dn
--------+-----+---------+------------------------+------+-----------+---------------+-----------
 43895 | t   | TLSv1.3 | TLS_AES_256_GCM_SHA384 |  256 |           |               |
(1 row)

postgres=# SHOW ssl;
 ssl
-----
 on
(1 row)

postgres=#
##### Verify SSL is actually applied for DB
postgres=# SELECT name, setting FROM pg_settings WHERE name LIKE '%ssl%';
postgres-#
postgres-#
                name                 |         setting
-------------------------------------+-------------------------
 ssl                                 | on
 ssl_ca_file                         | root.crt
 ssl_cert_file                       | server.crt
 ssl_ciphers                         | HIGH:MEDIUM:+3DES:!aNULL
 ssl_crl_dir                         |
 ssl_crl_file                        |
 ssl_dh_params_file                  |
 ssl_ecdh_curve                      | prime256v1
 ssl_key_file                        | server.key
 ssl_library                         | OpenSSL
 ssl_max_protocol_version            |
 ssl_min_protocol_version            | TLSv1.2
 ssl_passphrase_command              |
 ssl_passphrase_command_supports_reload | off
 ssl_prefer_server_ciphers           | on
(15 rows)
postgres-# \q

##### Copy /var/lib/pgsql/17/data/root.crt to /root/.postgresql/root.crt
[root@cldr-mngr data]# mkdir -p /root/.postgresql/
[root@cldr-mngr data]# cp /var/lib/pgsql/17/data/root.crt /root/.postgresql/root.crt

##### Copy the root.crt to all other hosts with the help of ansible, for this copy the root.crt file to
ipaserver/ansible control node
[root@cldr-mngr data]# scp -r /root/.postgresql/root.crt root@ipaserver:~/

##### Login to ipaserver and copy the root.crt Postgres DB certificate file to all other nodes  at
location /root/.postgresql/ with the help of ansible.

[root@ipaserver ~]# ls -l
[root@ipaserver ~]# chmod 644 root.crt
[root@ipaserver ~]# ansible all -m shell -a "mkdir -p /root/.postgresql/ && chmod -R 755 /root/.postgresql/"
[root@ipaserver ~]# ansible all -m copy -a "src=root.crt dest=/root/.postgresql/root.crt"

[root@cldr-mngr data]# psql -h cldr-mngr.cldrsetup.local -p 5432 -U postgres "dbname=postgres
sslmode=verify-full"
Password for user postgres:
psql (14.13)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.
postgres=#
postgres=# \q

[root@cldr-mngr data]# psql -h cldr-mngr.cldrsetup.local -p 5432 -U postgres "dbname=postgres
sslmode=verify-ca"
```

```
Password for user postgres:
psql (14.13)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.
postgres=#
postgres=# \q
[root@cldr-mngr data]#
```

Step 14. Create databases and service accounts for components that require databases. Following components requires databases:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-required-databases.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-configuring-starting-postgresql-server.html

Note: The databases must be configured to support the PostgreSQL UTF8 character set encoding.
Note: Record the values you enter for database names, usernames, and passwords. The Cloudera Manager installation wizard requires this information to correctly connect to these databases.

```
##### Create CM DB and USERS
[root@cldr-mngr data]# sudo -u postgres psql

CREATE ROLE scm LOGIN PASSWORD 'scm';
CREATE DATABASE scm OWNER scm ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE scm TO scm;

CREATE ROLE rman LOGIN PASSWORD 'rman';
CREATE DATABASE rman OWNER rman ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE rman TO rman;

CREATE ROLE hue LOGIN PASSWORD 'hue';
CREATE DATABASE hue OWNER hue ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE hue TO hue;

CREATE ROLE hive LOGIN PASSWORD 'hive';
CREATE DATABASE hive OWNER hive ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE hive TO hive;

CREATE ROLE oozie LOGIN PASSWORD 'oozie';
CREATE DATABASE oozie OWNER oozie ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE oozie TO oozie;

CREATE ROLE rangeradmin LOGIN PASSWORD 'rangeradmin';
CREATE DATABASE ranger OWNER rangeradmin ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE ranger TO rangeradmin;

/*For Ranger KMS, use rangerkms rather than rangeradmin user.*/
CREATE ROLE rangerkms LOGIN PASSWORD 'rangerkms';
CREATE DATABASE rangerkms OWNER rangerkms ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE rangerkms TO rangerkms;

CREATE ROLE schemaregistry LOGIN PASSWORD 'schemaregistry';
CREATE DATABASE schemaregistry OWNER schemaregistry ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE schemaregistry TO schemaregistry;

CREATE ROLE yqm LOGIN PASSWORD 'yqm';
CREATE DATABASE yqm OWNER yqm ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE yqm TO yqm;

/*For the SMM metadata store, create a database called smm with the password smm:*/
CREATE ROLE smm LOGIN PASSWORD 'smm';
CREATE DATABASE smm OWNER smm ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE smm TO smm;

CREATE ROLE das LOGIN PASSWORD 'das';
CREATE DATABASE das OWNER das ENCODING 'UTF8';
GRANT ALL PRIVILEGES ON DATABASE das TO das;
```

```
ALTER DATABASE hive SET standard_conforming_strings=off;
ALTER DATABASE oozie SET standard_conforming_strings=off;
SELECT 1;
SHOW ssl;
\q

--- Alternate commands
--- CREATE USER registry WITH PASSWORD 'registry';
--- GRANT ALL PRIVILEGES ON DATABASE "registry" to registry;
```

Note: If you plan to use Apache Ranger, please visit Configuring a PostgreSQL Database for Ranger or Ranger KMS for instructions on creating and configuring the Ranger database. **included above- install JDBC driver, create DB for ranger etc.**

Note: If you plan to use Schema Registry or Streams Messaging Manager, please visit Configuring the Database for Streaming Components for instructions on configuring the database. included above- create smm and registry db etc.

The following procedure describes how to install Cloudera Manager and then using Cloudera Manager to install Cloudera Data Platform Cloudera on premises Base 7.3.1.

Procedure 6. Install Cloudera Manager Server (CM-UI)

Cloudera Manager, an end-to-end management application, is used to install and configure Cloudera on premises Base. During CDP Installation, Cloudera Manager's Wizard will help to install Hadoop services and any other role(s)/service(s) on all nodes using the following procedure:

- Discovery of the cluster nodes
- Configure the Cloudera parcel or package repositories
- Install Hadoop, Cloudera Manager Agent (CMA) and Impala on all the cluster nodes.
- Install the Oracle JDK or OpenJDK if it is not already installed across all the cluster nodes.
- Assign various services to nodes.
- Start the Hadoop services

Note: Please see the JAVA requirements for Cloudera on premises Base.

Step 1. Install the Cloudera Manager Server packages by running following command:

```
[root@cldr-mngr data]# dnf install -y cloudera-manager-agent cloudera-manager-daemons cloudera-manager-server

# Recommendation: Always install agents via CM-UI only. Never install manually as it generates agent
config as localhost and leads to heartbeat error. If HeartBeat error comes up, then run below command to
update agent config (before start scm-server)
[root@cldr-mngr data]# sed -i 's/server_host=localhost/server_host=cldr-mngr.cldrsetup.local/g'
/etc/cloudera-scm-agent/config.ini
```

Step 2. Enable TLS 1.2 on Cloudera Manager Server.
https://docs.cloudera.com/cloudera-manager/7.11.3/installation/topics/cdpdc-enable-tls-12-cm-server.html

Step 3. Import the PostgreSQL root certificate in Step 5.

Step 4. If the Database host and Cloudera Manager Server host are located on the same machine, then perform the following steps to import the PostgreSQL database root certificate, as mentioned below in Step 5:

Step 5. Go to the path where root certificates are stored. By default it is /var/lib/pgsql/17/data/.

```
##### Configure CDP to use SSL Enabled DB

# Create a new directory in the following path by running the following command:
[root@cldr-mngr data]# mkdir -p /var/lib/cloudera-scm-server/.postgresql
[root@cldr-mngr data]# chmod 755 /var/lib/cloudera-scm-server/.postgresql
[root@cldr-mngr data]# cd /var/lib/cloudera-scm-server/.postgresql
```

```
# Copy the PostgreSQL root certificate to the new directory on the Cloudera Manager server host by running
the following command:
[root@cldr-mngr data]# cp /var/lib/pgsql/17/data/root.crt root.crt

# Change the ownership of the root certificate by running the following command:
[root@cldr-mngr data]# chown cloudera-scm: root.crt
[root@cldr-mngr data]# ls -lt
total 8
-rw-r--r-- 1 cloudera-scm cloudera-scm 4639 Mar  5 16:59 root.crt

# Include this root certificate path in the JDBC URL as follows:
# jdbc:postgresql://<DB HOSTNAME>:<DB-PORT>/<DB
NAME>?ssl=true&sslmode=verify-ca&sslrootcert=<PATH_TO_ROOT_CERTIFICATE>
#
jdbc:postgresql://cldr-mngr.cldrsetup.local:5432/scm?ssl=true&sslmode=verify-ca&sslrootcert=/var/lib/cloudera
-scm-server/.postgresql/root.crt

##### Changes required for Ranger SSL

[root@cldr-mngr data]# cp /usr/share/java/postgresql-connector-java.jar
/opt/cloudera/cm/lib/postgresql-connector.jar
[root@cldr-mngr data]# unlink /opt/cloudera/cm/lib/postgresql-42.*.jar
[root@cldr-mngr data]# ls -ltr /opt/cloudera/cm/lib/*postgres*
[root@cldr-mngr data]# chmod 755 /var/lib/cloudera-scm-server/
[root@cldr-mngr data]# ls -ltr /var/lib/cloudera-scm-server/.postgresql/*.crt
[root@cldr-mngr data]#
```

Step 6. Run the scm_prepare_database.sh script to check and generate database configuration file for cloudera-manager i.e. db.properties and test the database connection between cloudera-manager and database server:

```
# Run the script to configure PostgreSQL with TLS 1.2 enabled
###### sudo /opt/cloudera/cm/schema/scm_prepare_database.sh -h<DB HOSTNAME> --jdbc-url
"jdbc:postgresql://db_server_host:db_port/db_name?ssl=true&sslmode=verify-ca&sslrootcert=<PATH_TO_DB_ROOT_
CERTIFICATE>" <db_type:postgresql> <db_name> <db_role_user> <dn_user_password> --ssl
[root@cldr-mngr ~]# sudo /opt/cloudera/cm/schema/scm_prepare_database.sh -hcldr-mngr.cldrsetup.local
--jdbc-url
"jdbc:postgresql://cldr-mngr.cldrsetup.local:5432/scm?ssl=true&sslmode=verify-ca&sslrootcert=/var/lib/clouder
a-scm-server/.postgresql/root.crt" postgresql scm scm scm --ssl
JAVA_HOME=/usr/lib/jvm/java-17-openjdk-17.0.14.0.7-1.el9.x86_64
Verifying that we can write to /etc/cloudera-scm-server
Creating SCM configuration file in /etc/cloudera-scm-server
Executing:  /usr/lib/jvm/java-17-openjdk-17.0.14.0.7-1.el9.x86_64/bin/java -cp
/usr/share/java/mysql-connector-java.jar:/usr/share/java/oracle-connector-java.jar:/usr/share/java/postgre
sql-connector-java.jar:/opt/cloudera/cm/schema/../lib/* com.cloudera.enterprise.dbutil.DbCommandExecutor
/etc/cloudera-scm-server/db.properties com.cloudera.cmf.db.
[main] DbCommandExecutor              INFO  A JDBC URL override was specified. Using this as the URL to
connect to the database and overriding all other values.
[main] DbCommandExecutor              INFO  Successfully connected to database.
All done, your SCM database is configured correctly!
[root@cldr-mngr ~]#
```

Step 7. Upon successful connection, the scm_prepare_database.sh script writes the content of /etc/cloudera-scm-server/db.properties file as shown below, verify the content, should look like below:

```
[root@cldr-mngr ~]# cat /etc/cloudera-scm-server/db.properties
# Auto-generated by scm_prepare_database.sh on Tue Mar 5 08:02:56 PM PST 2024
#
# For information describing how to configure the Cloudera Manager Server
# to connect to databases, see the "Cloudera Manager Installation Guide."
#
com.cloudera.cmf.db.type=postgresql
com.cloudera.cmf.db.host=cldr-mngr.cldrsetup.local
com.cloudera.cmf.db.name=scm
com.cloudera.cmf.db.user=scm
com.cloudera.cmf.db.setupType=EXTERNAL
com.cloudera.cmf.db.password=scm
com.cloudera.cmf.orm.hibernate.connection.url=jdbc:postgresql://cldr-mngr.cldrsetup.local:5432/scm?ssl=tru
e&sslmode=verify-ca&sslrootcert=/var/lib/cloudera-scm-server/.postgresql/root.crt
```

```
[root@cldr-mngr ~]#
```

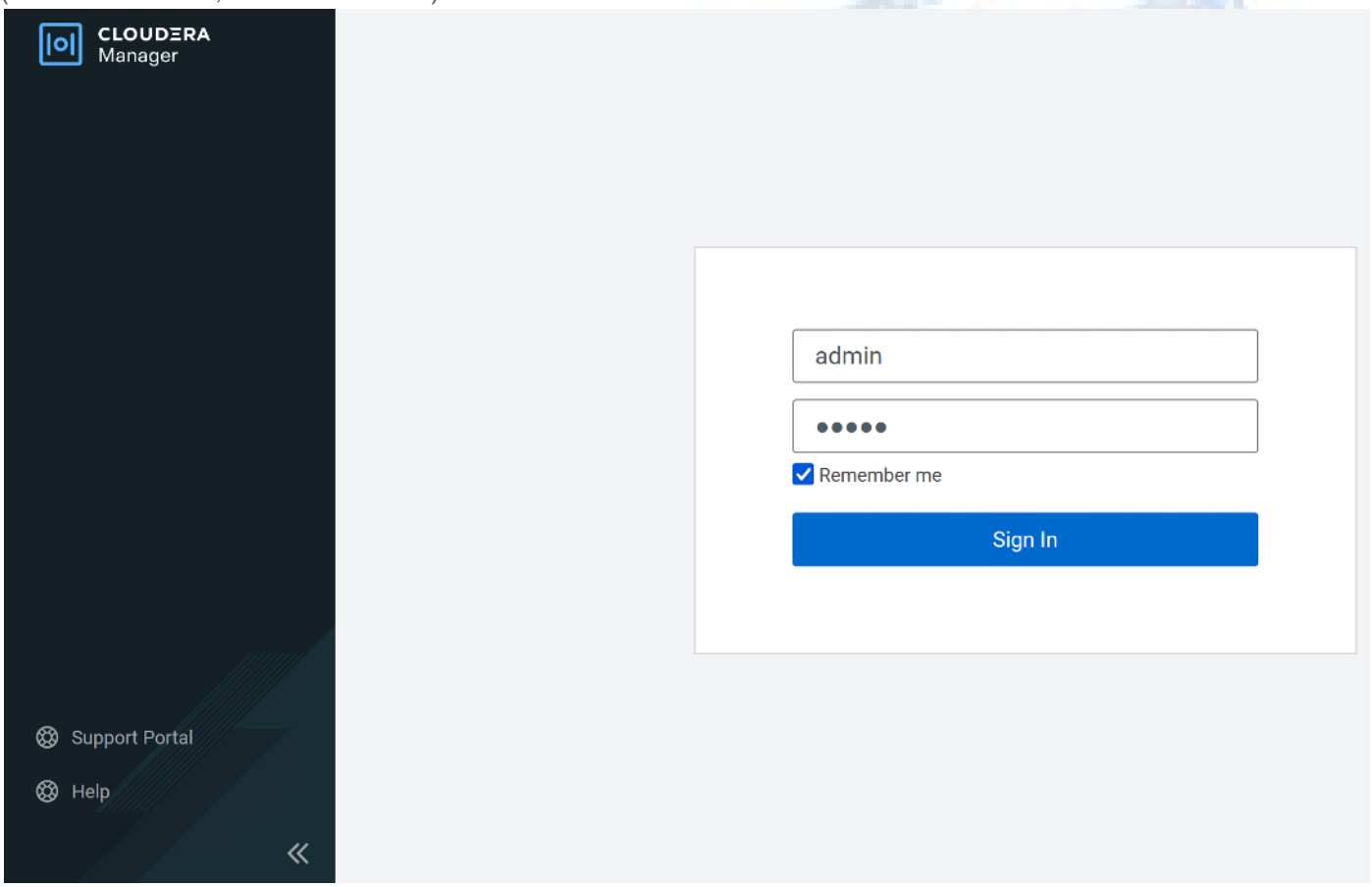
Step 8. Start the Cloudera Manager Server:

```
[root@cldr-mngr ~]# systemctl start cloudera-scm-server cloudera-scm-agent
[root@cldr-mngr ~]# systemctl enable cloudera-scm-server cloudera-scm-agent
[root@cldr-mngr ~]# systemctl status cloudera-scm-server cloudera-scm-agent -l
##### Run the below command to check the logs of cloudera-scm-server starting up. Wait until you see the
Started Jetty server message on the screen.
[root@cldr-mngr ~]# sudo tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log
```

Step 9. The Cloudera Manager should show the below logs before the UI actually comes up.

```
INFO WebServerImpl:org.eclipse.jetty.server.Server: Started @89711ms
INFO WebServerImpl:com.cloudera.server.cmf.WebServerImpl: Started Jetty server.
```

Step 10. Once the Cloudera-Manager(CM) installation is completed, open the endpoint URL http://<cldr-mgr_ip_addr>:7180/ of the Cloudera Manager WebUI and login to the CM using default credentials. (Username: admin, Password: admin)



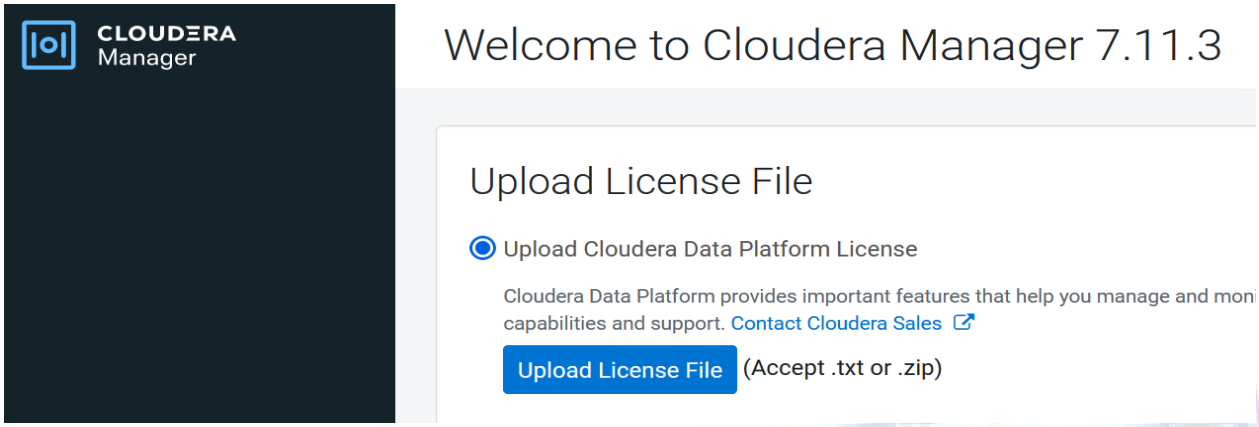
Note: Default username and password for Cloudera Manager is admin/admin.

Step 11. After logging in to Cloudera-Manager WebUI using the above credentials and uploading the Licence Key, open a new tab of CM-UI on browser and search for JAVA PATH in search bar present at left hand side:

```
# Override JAVA PATH in CM-UI> Search JAVA> Override the value> Save Changes> Restart Cloudera-SCM-Server
```

```
/usr/lib/jvm/java-17-openjdk-17.0.14.0.7-1.el9.x86_64 (Take the correct Java Path from your system, where java is installed)
```

Step 12. The Welcome to Cloudera Manager page appears. Since you would have received the CDP license before, select ***Upload Cloudera Data Platform License*** and upload the downloaded .txt or .zip file with the license information.



Step 13. Activate your license for Cloudera Data Platform by clicking the ***Continue*** button. Click Continue.

Step 14. The ***Add Private Cloud Base Cluster*** page appears. Next, we will enable AutoTLS for CM.

Step 15. As a prerequisite step to ***enabling AutoTLS***, login to the cldr-mngr node as user root, and verify ***cloudera-manager-agent*** software is installed and running successfully. Verify the logs in below file:

```
[root@cldr-mngr ~]# tail -f /var/log/cloudera-scm-agent/cloudera-scm-agent.log

##### Verify the same by running the below command. This should return the output stating the service is active and in running state.
[root@cldr-mngr ~]# systemctl status cloudera-scm-agent -l
```

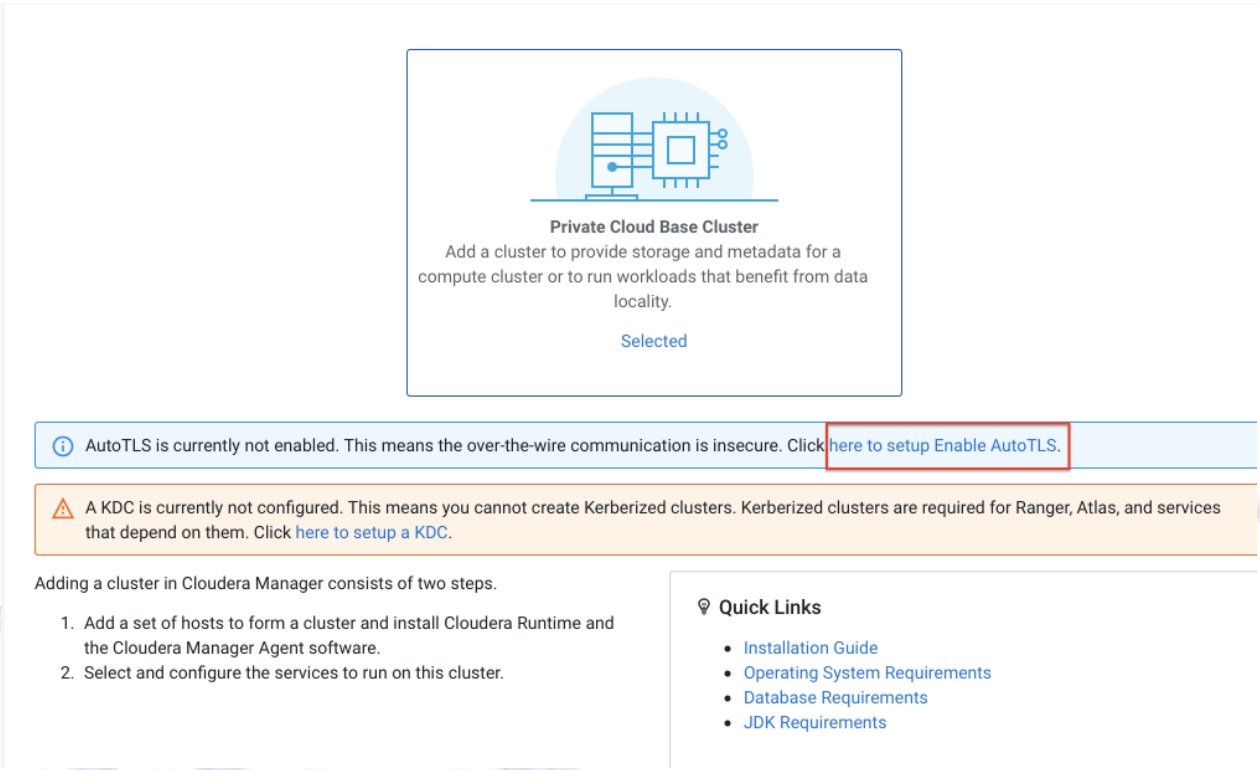
```
[root@cdpbase cloudera-scm-server]# systemctl status cloudera-scm-agent
● cloudera-scm-agent.service - Cloudera Manager Agent Service
   Loaded: loaded (/usr/lib/systemd/system/cloudera-scm-agent.service; enabled; vendor preset: disabled)
   Active: active (running) since Sat 2023-04-08 02:24:22 UTC; 9s ago
 Main PID: 3062 (cmagent)
   CGroup: /system.slice/cloudera-scm-agent.service
           └─3062 /usr/bin/python2 /opt/cloudera/cm-agent/bin/cm agent
```

****

Procedure 7. Enable AutoTLS

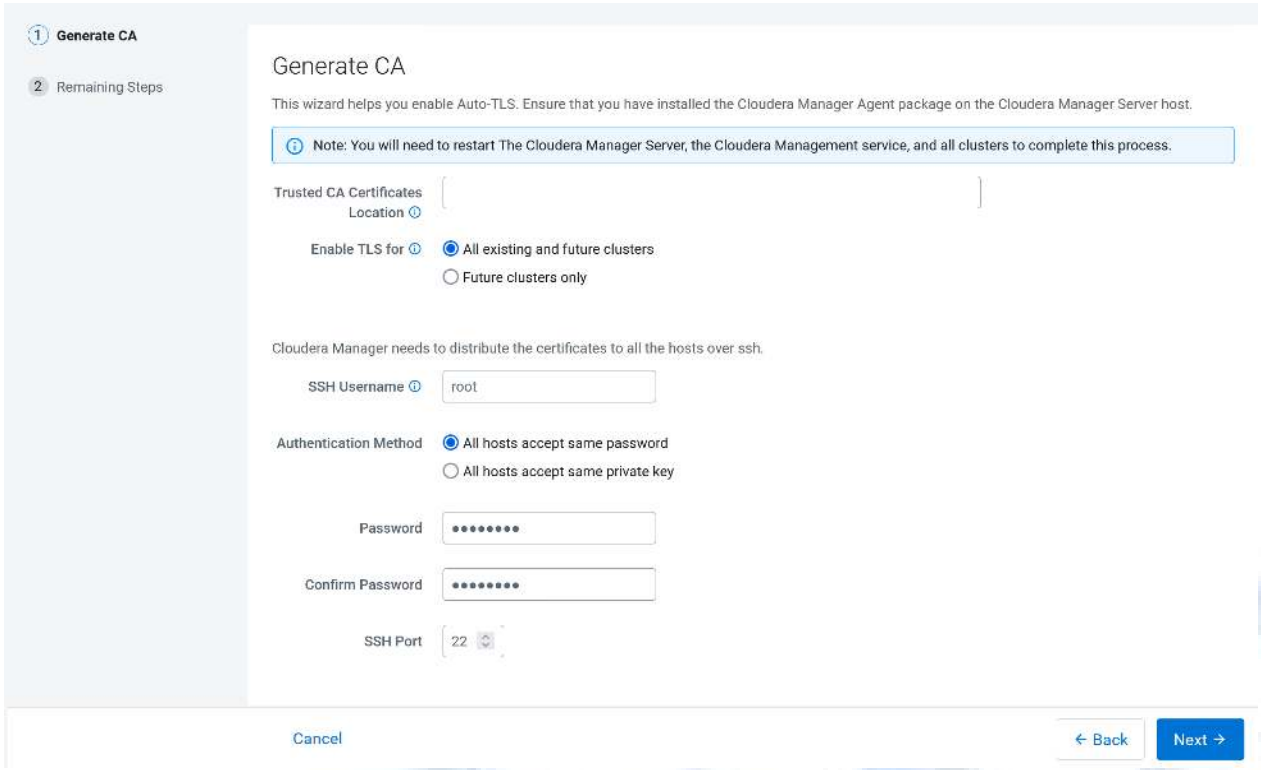
Auto-TLS is managed using the certmanager utility, which is included in the Cloudera Manager Agent software, and not the Cloudera Manager Server software. You must install the Cloudera Manager Agent software on the Cloudera Manager Server host to be able to use the utility. You can use certmanager to manage auto-TLS on a new installation. For more information, go to: Configuring TLS Encryption for Cloudera Manager Using Auto-TLS

Step 1. Click on the link ***here to setup Enable AutoTLS*** to set up AutoTLS through Cloudera Manager on the ***Add Private Cloud Base Cluster*** page.

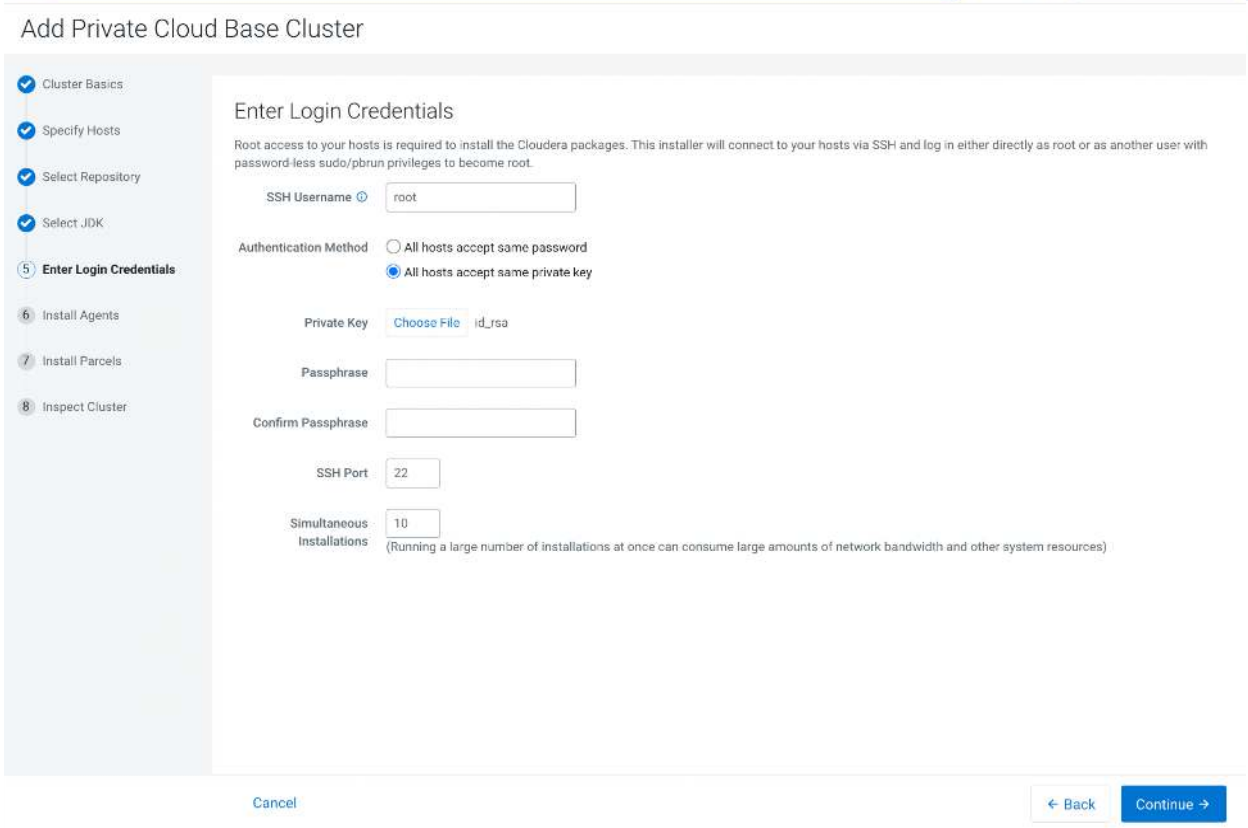


Step 2. Below screen will appear. Enter the values for the parameters as shown below. (**We will be using the private key approach**, you can use password option as well, both options should considerably work)

Component	Value
Enable TLS for	All existing and future clusters
SSH username	root
Authentication method	All hosts accept same private key / All hosts accept same password
Private Key (If using Key approach)	Choose the private key created and downloaded in earlier section
Password (If using Password approach)	Enter VM's root users' password
Confirm Password	Enter VM's root users' password (again)



a. Screenshot for using the Password based authentication method.



b. Screenshot for using the Private Key based authentication method.

Step 3. Click ***Next*** to continue. Below screen will appear, if all the values are entered properly.

Step 4. Click on Finish.

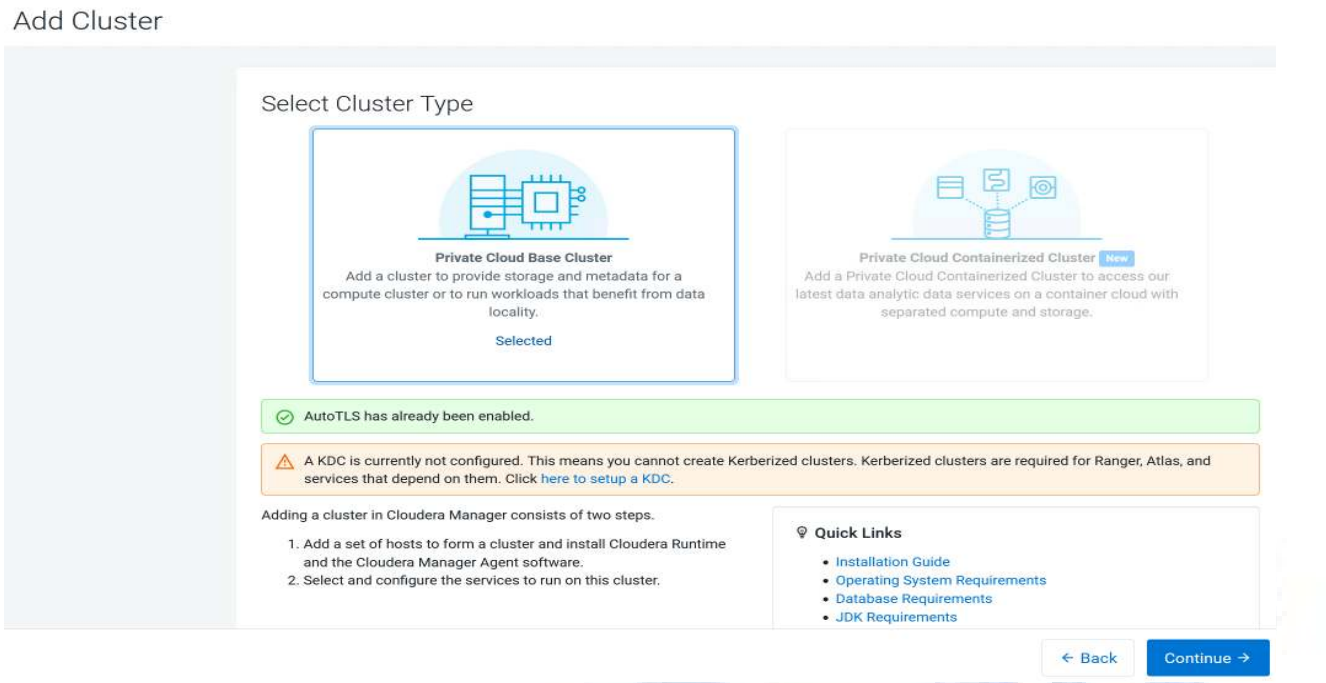
Step 5. After enabling the AutoTLS for the CM-UI through the browser, login to cldr-mngr node at backend as user root and restart Cloudera Manager Server, suggested in the previous screenshot.

```
[root@cldr-mngr ~]# systemctl restart cloudera-scm-server
[root@cldr-mngr ~]# systemctl status cloudera-scm-server -l

##### Run the below command to check the logs of cloudera-scm-server starting up. Wait until you see the
Started Jetty server message on the screen.
[root@cldr-mngr ~]# sudo tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log
```

Step 6. Once you see the message ***Started Jetty server*** in the logs, Login to Cloudera Manager using URL endpoint, ***http://<IP_for_CM_Server>:7180***, in a new incognito window.

Step 7. The URL should get redirected to https at 7183 port i.e. https://<CM_SRVR_IP_ADDR>:7183/ This means that the AutoTLS configuration is successful. You might get a warning message on the browser related to the certificate. You can ignore the warning and visit the website as this is not a signed certificate.

Step 8. Enter the default credentials (admin/admin) and click on ***Login***. You should see AutoTLS enabled as shown in the image below.



Procedure 8. Enable Kerberos:- Kerberos Integration with CDP

Cloudera Manager provides a wizard for integrating your organization's Kerberos with your cluster to provide authentication services. Cloudera Manager clusters can be integrated with MIT Kerberos, Red Hat Identity Management (or the upstream FreeIPA), or Microsoft Active Directory. For more information, see Enable Kerberos Authentication for CDP.

Note: In our lab, we configured RedHat FreeIPA based Kerberos authentication. We presume that FreeIPA is pre-configured with user(s) and proper authentication is set up for Kerberos Authentication.

Note: Before integrating Kerberos with your cluster, configure TLS encryption between Cloudera Manager Server and all Cloudera Manager Agent host systems in the cluster. During the Kerberos integration process, Cloudera Manager Server sends keytab files to the Cloudera Manager Agent hosts, and TLS encrypts the network communication, so these files are protected.

Note: For FreeIPA, you must have administrative privileges to the ipaserver instance for initial setup and for on-going management, or you will need to have the help of your LDAP administrator prior to and during the integration process. For example, administrative access is needed to access the FreeIPA Kerberos KDC, create principals, and troubleshoot Kerberos TGT/TGS-ticket-renewal and take care of any other issues that may arise.

Note: In case, you configure ***Active-Directory*** based Kerberos authentication. We presume that Active Directory is pre-configured with OU, user(s) and proper authentication is setup for Kerberos Authentication. LDAP users and bind users are expected to be in the same branch/OU.

Note: For ***Active Directory***, you must have administrative privileges to the Active Directory instance for initial setup and for on-going management, or you will need to have the help of your AD administrator prior to and during the integration process. For example, administrative access is needed to access the Active Directory KDC, create principals, and troubleshoot Kerberos TGT/TGS-ticket-renewal and take care of any other issues that may arise.

Step 1. Before proceeding further with KDC setup, we need to ensure that the changes to ***krb5.conf*** related to the default cache is not reversed. View the contents of the file ***/etc/krb5.conf*** after logging in to both ***ipaserver*** node and ***cldr-mngr*** node and check whether the property ***default_ccache_name*** is commented out. If not, then open the file and comment it out.

```
[root@ipaserver ~]# sudo vi /etc/krb5.conf
#default_ccache_name = KEYRING:persistent:%{uid}
```

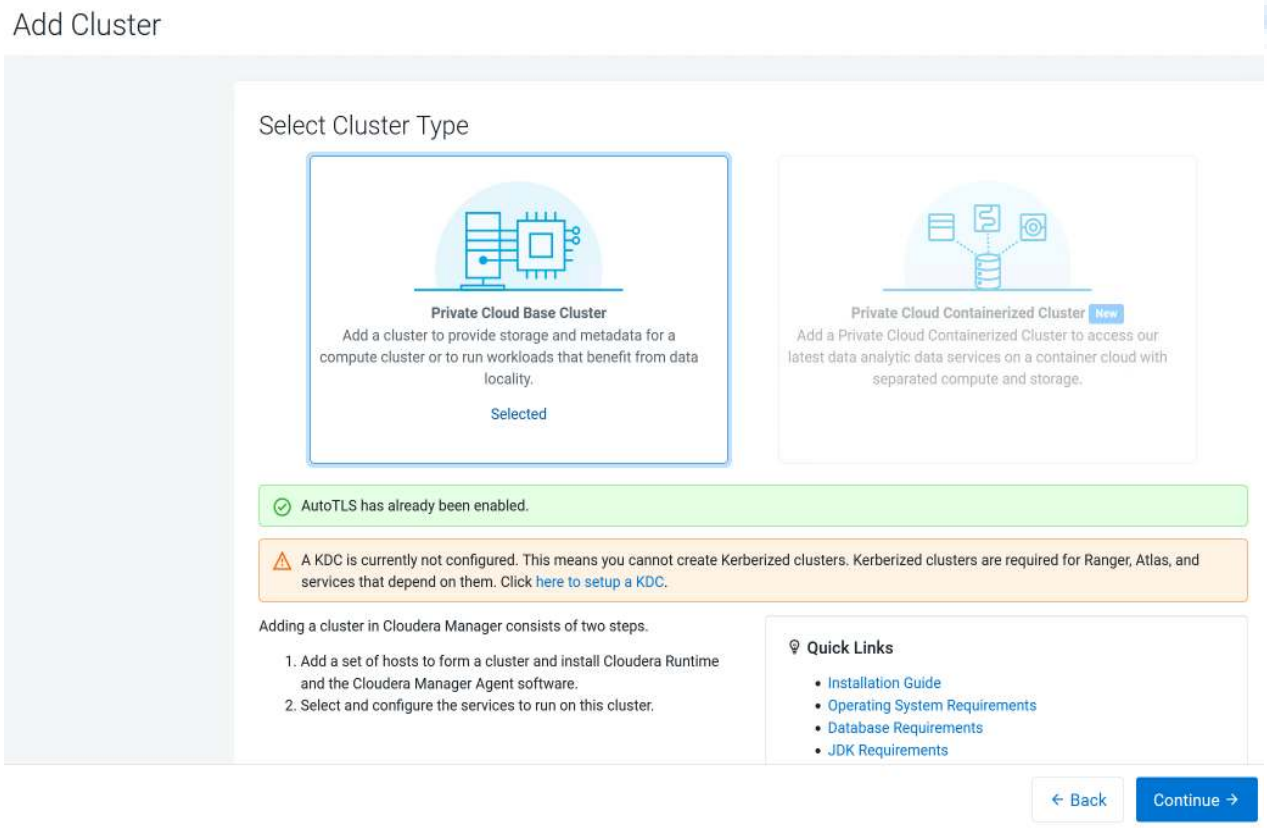
```
[libdefaults]
 default_realm = CDPPVCDS.COM
 dns_lookup_realm = false
 dns_lookup_kdc = true
 rdns = false
 ticket_lifetime = 24h
 forwardable = true
 udp_preference_limit = 0
# default_ccache_name = KEYRING:persistent:%{uid}
```

If you have made any changes, only then run the below commands to restart all the IPA services. If not, skip to the next step.

```
[root@ipaserver ~]# ipactl restart
```

Step 2. Now, move to CM-UI on your browser.

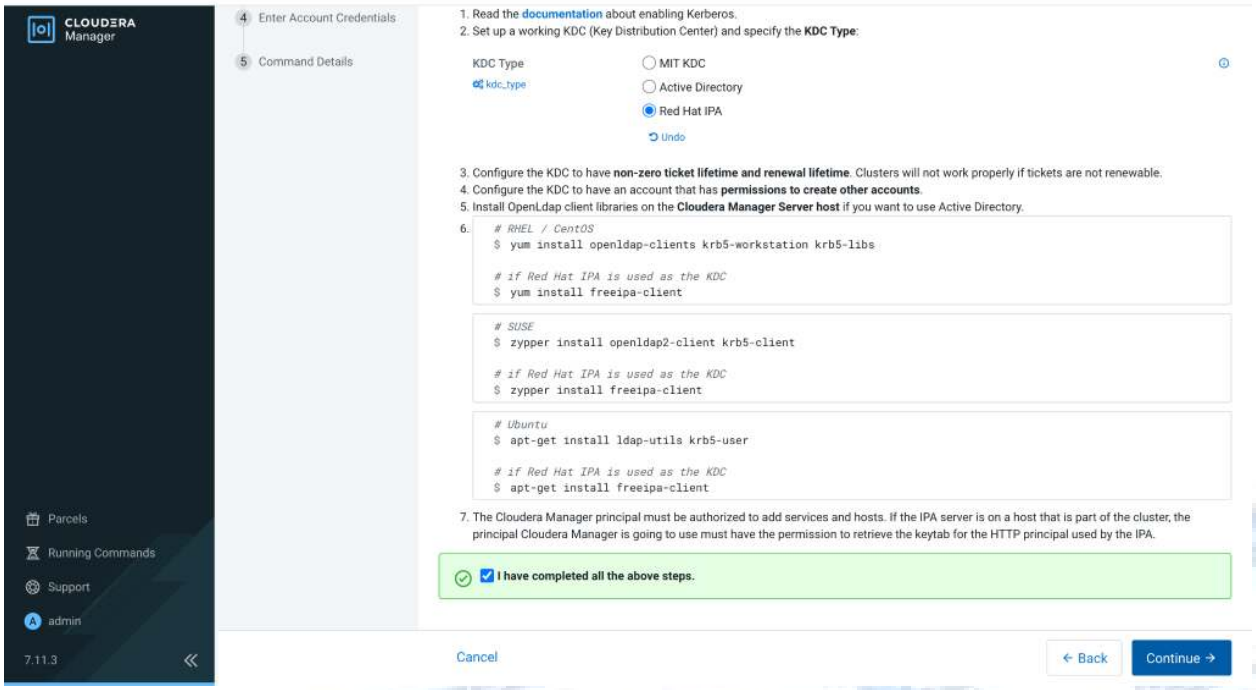
Step 3. In the Cloudera manager console click on ***here to set up a KDC***, on the same TLS page, to enable the kerberos authentication on the cluster.



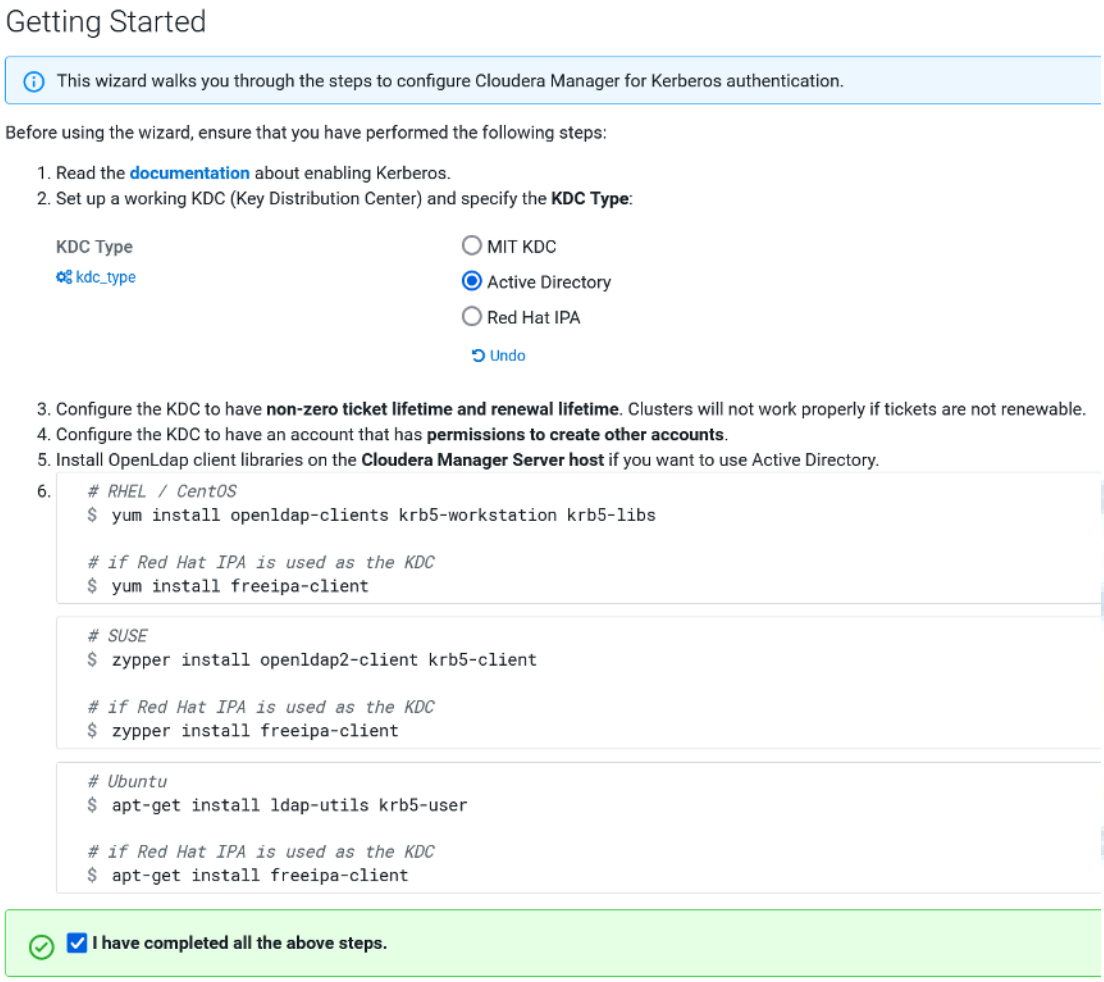
Step 4. Click Continue.

Step 5. Select ***RedHat IPA*** as shown below and check the box for ***I have completed all the above steps.***

We have already installed the required RedHat FreeIPA/ Kerberos dependency packages openldap-clients, krb5-workstation and krb5-libs in previous steps.



Step 6. Select *Active Directory* as shown below, **if you're proceeding with AD based Kerberos integration** and check the box for ***I have completed all the above steps.*** Setting up AD is beyond the scope of this document.
We have already installed the required RedHat FreeIPA/ Kerberos dependency packages openldap-clients, krb5-workstation and krb5-libs in previous steps. ***(Skip this step, in our case)***



Step 7. As recommended, install the following in all Cloudera Manager hosts by running the following command. Once completed, click the checkbox ***"I have completed all the above steps"*** and click ***Continue***.

(Skip the below command execution step, as we already installed the dependencies in prior steps)

```
[root@ipaserver ~]# ansible all -m command -a "dnf install -y openldap-clients krb5-workstation krb5-libs"
```

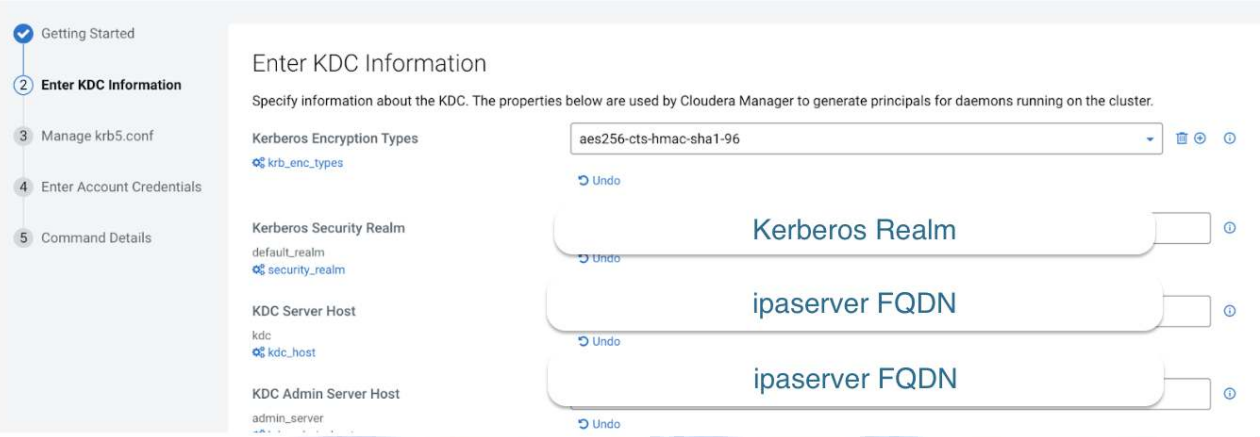
Step 8. For enabling Kerberos, under the Enter KDC Information page, provide below inputs for ***Enter KDC information*** for this Cloudera Manager. Use Table 6 as an example to fill-in the KDC setup information, provide below inputs and click Next.

Table 4. KDC Setup components and their corresponding value

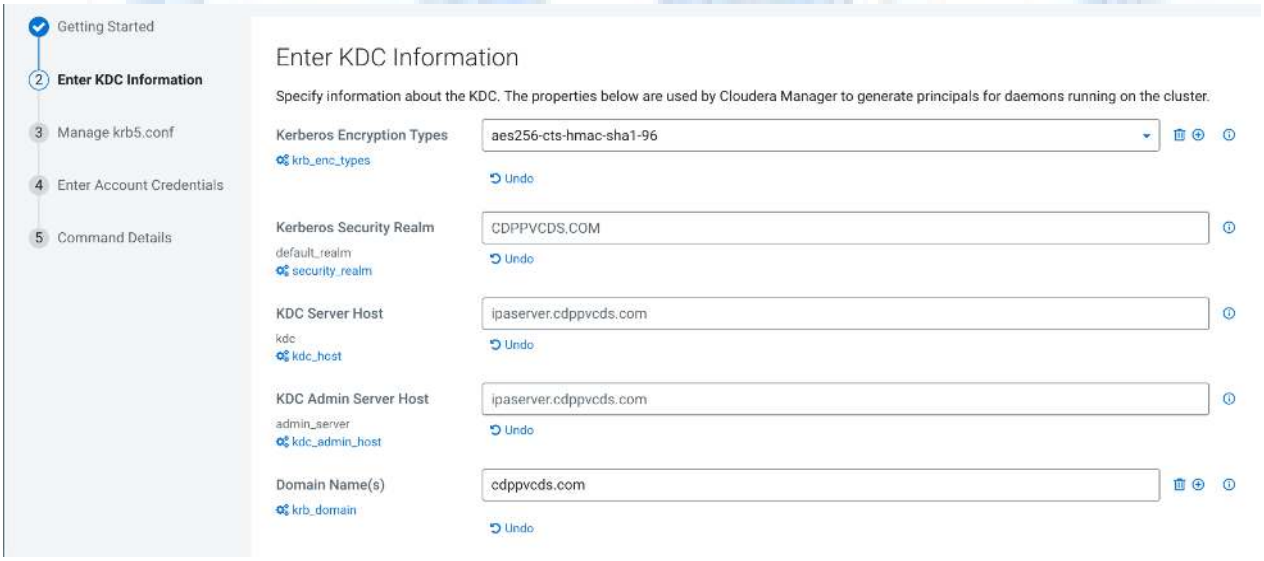
Component	Value
Kerberos Encryption Types	aes256-cts-hmac-sha1-96
Kerberos Security Realm	CLDRSETUP.LOCAL
KDC Server Host	ipaserver.cldrsetup.local
KDC Admin Server Host	ipaserver.cldrsetup.local

Component	Value
Domain Name(s)	cldrsetup.local
Base DN	dc=cldrsetup,dc=local
Active Directory Suffix **(Only for AD based Kerberos)**	OU=admin,DC=cldrsetup,DC=local
Active Directory Delete Accounts on Credential Regeneration **(Only for AD based Kerberos)**	Select (Check)

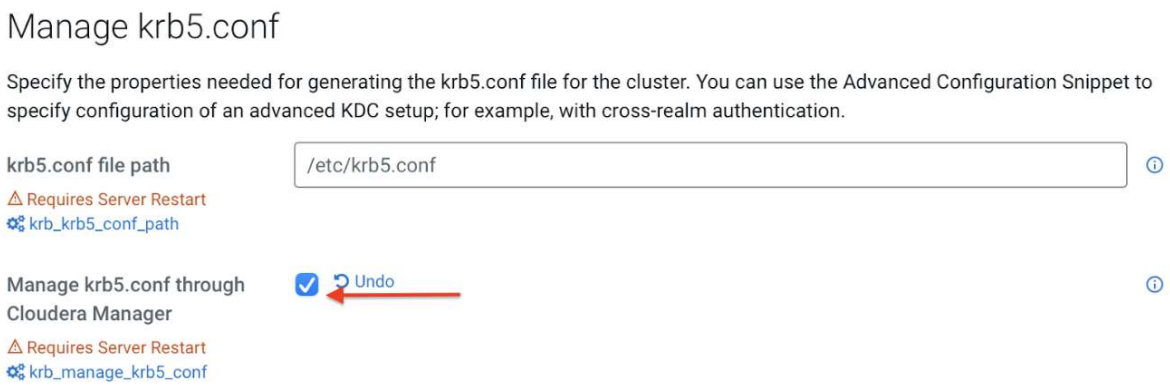
Check the picture below on where to populate the above mentioned fields:



Note: In this setup, we used Kerberos authentication with ***RedHat FreeIPA***.

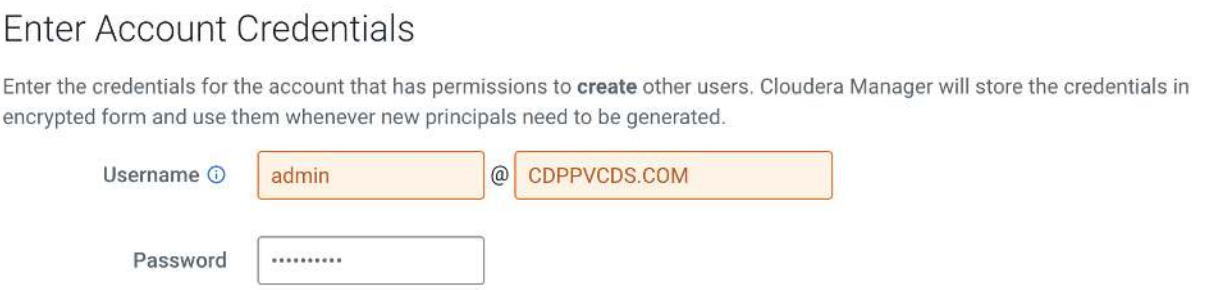


Step 9. On the Next Page, check the box for ***Manage "krb5.conf"*** to enable it through Cloudera Manager. This will install the krb5.conf file in all the hosts selected for the data lake.



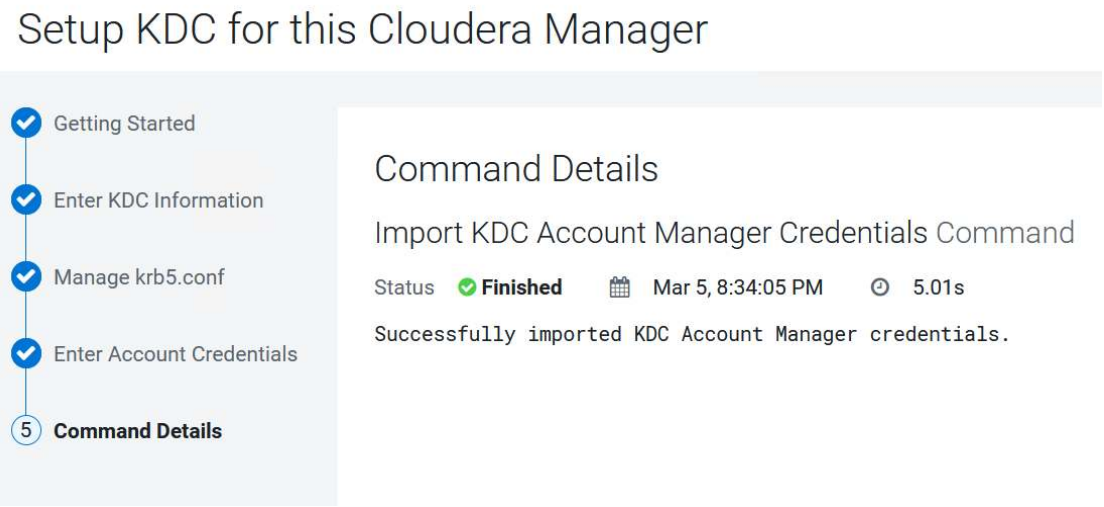
Step 10. Next, enter the details as per the configuration of FreeIPA you did before. i.e., provide the domain and password of the admin user configured earlier in the FreeIPA setup. Enter account credentials for the admin which you have created. This credential will be used to generate the keytabs. In our lab setup, "admin" user is created during the IPA server installation. Click Continue.

Enter the REALM portion of the principal in upper-case only to conform to Kerberos convention.



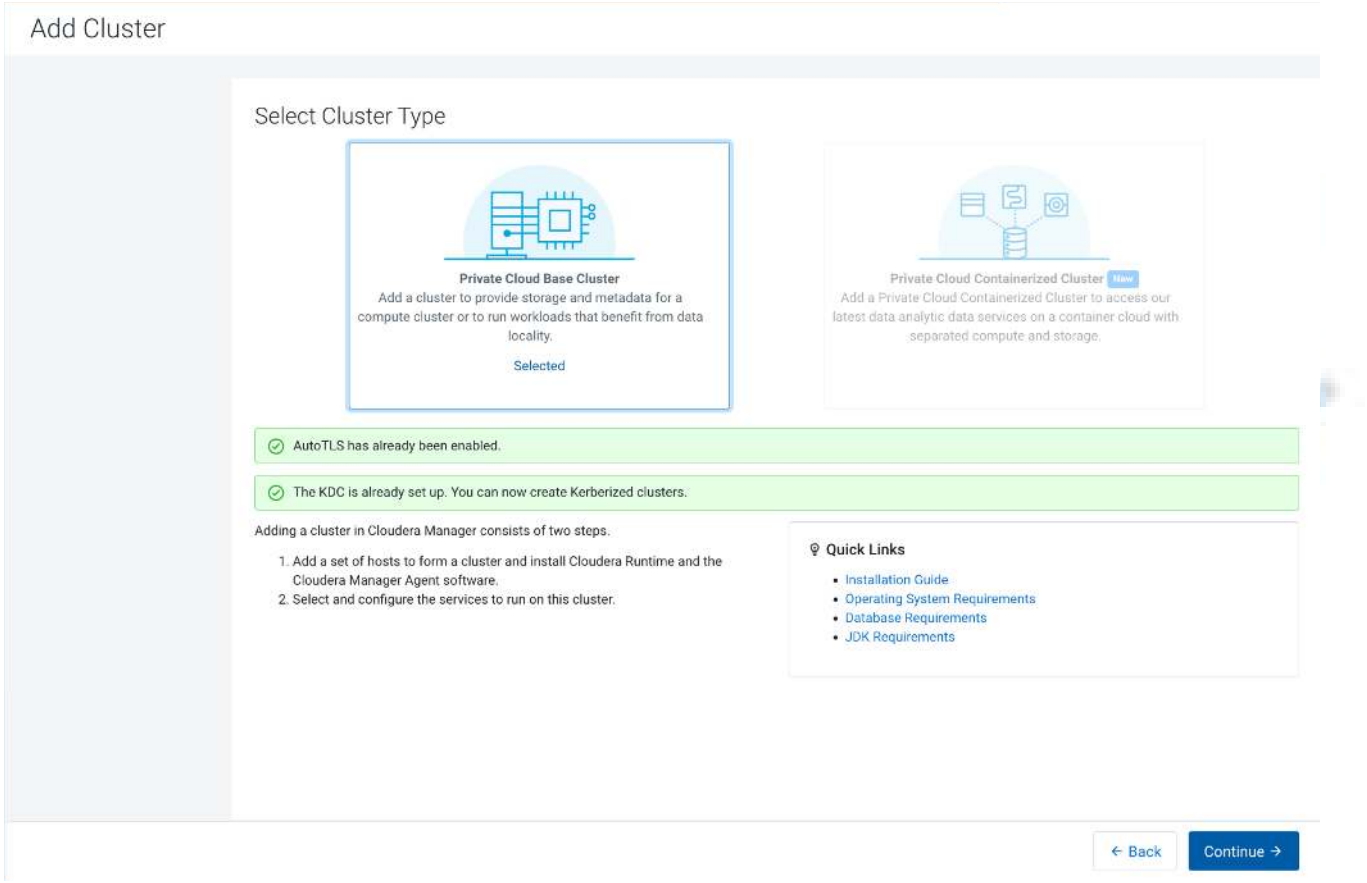
Step 11. Click Finish to complete the KDC setup.

Step 12. KDC Account manager credentials should get imported successfully as shown below.



```
##### For Red Hat IdM, make sure that all cluster hosts are joined to the IPA domain, after freeipa-client
is installed.
## Kerberos client OS-specific packages must be installed on all cluster hosts and client hosts that will
authenticate using Kerberos.

[root@ipaserver ~]# rpm -qa|grep ipa-client
ipa-client-4.6.8-5.el7.centos.16.x86_64
ipa-client-common-4.6.8-5.el7.centos.16.noarch
[root@ipaserver ~]#

##### If keytab error come up during kerberos configuration -- go to ipa server and run -
[root@ipaserver ~]# ipactl restart && ipactl status
```

Step 13. Once the KDC setup is completed, the Cloudera Manager wizard for adding a cluster will reflect the following:



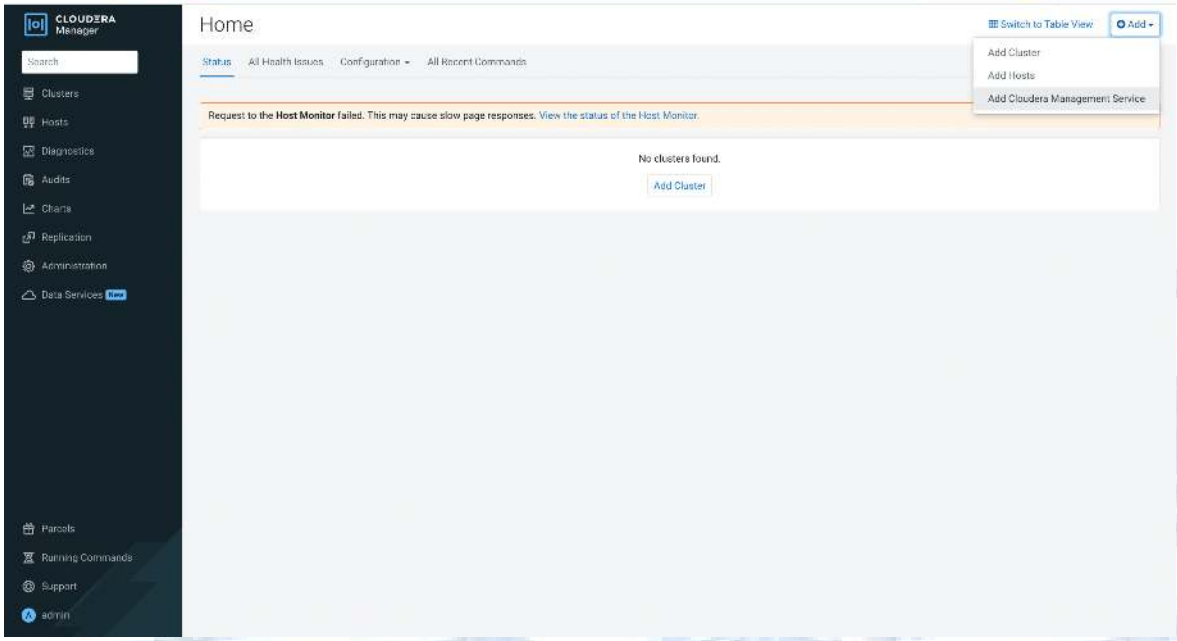
Step 14. Verify Kerberos configuration.

```
[root@ipaserver ~]# kinit admin@CLDRSETUP.LOCAL
Password for admin@CLDRSETUP.LOCAL: <cloudera123>
[root@ipaserver ~]# klist -e
Ticket cache: KCM:0
Default principal: admin@CLDRSETUP.LOCAL

Valid starting       Expires              Service principal
03/05/2024 20:35:11  03/06/2024 20:35:07  krbtgt/CLDRSETUP.LOCAL@CLDRSETUP.LOCAL
        renew until 03/12/2024 21:35:07
[root@ipaserver ~]#
```

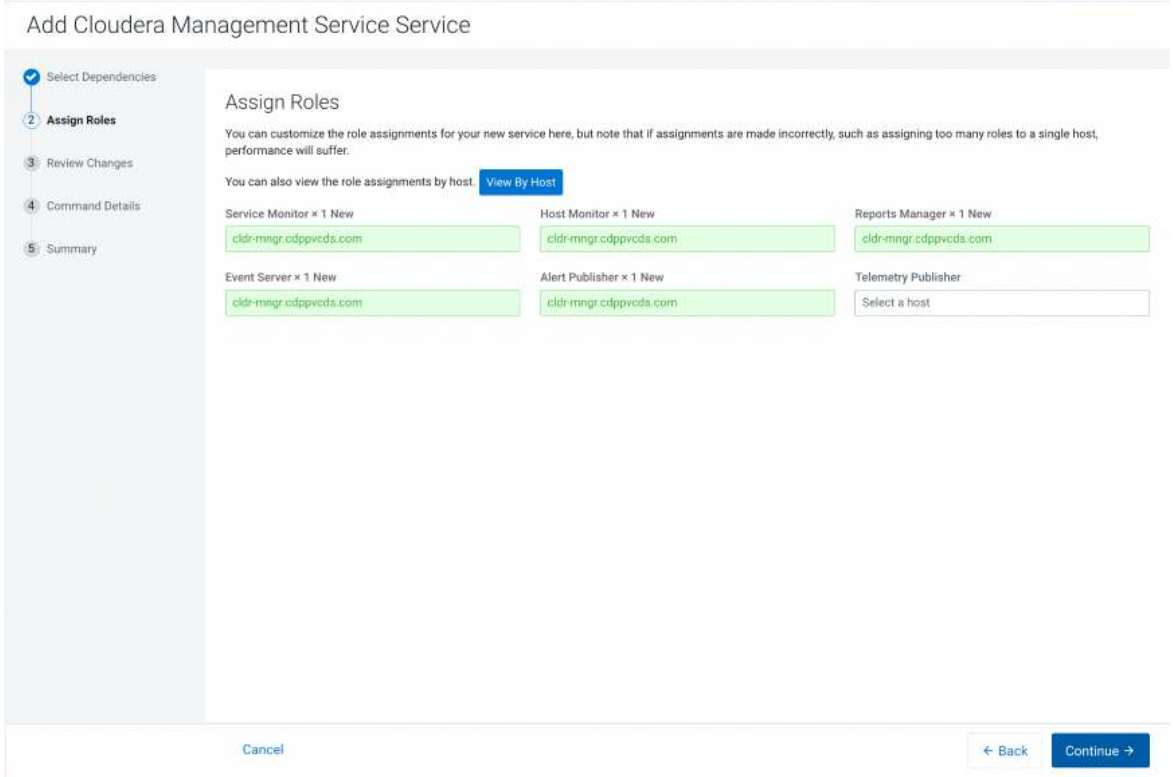
Step 15. Setup the *Cloudera Management Services*: **(Only if the status of hosts/services/charts are not visible / visible as (?) / or showing the errors on console)**

```
##### Setup the Cloudera Management Services (need rman DB details), it will start service monitor and other services and enable charts view. If Cloudera Management Services are not installed/ enabled or not working properly, status of hosts, or installed services will not be updated on CM-UI.
```

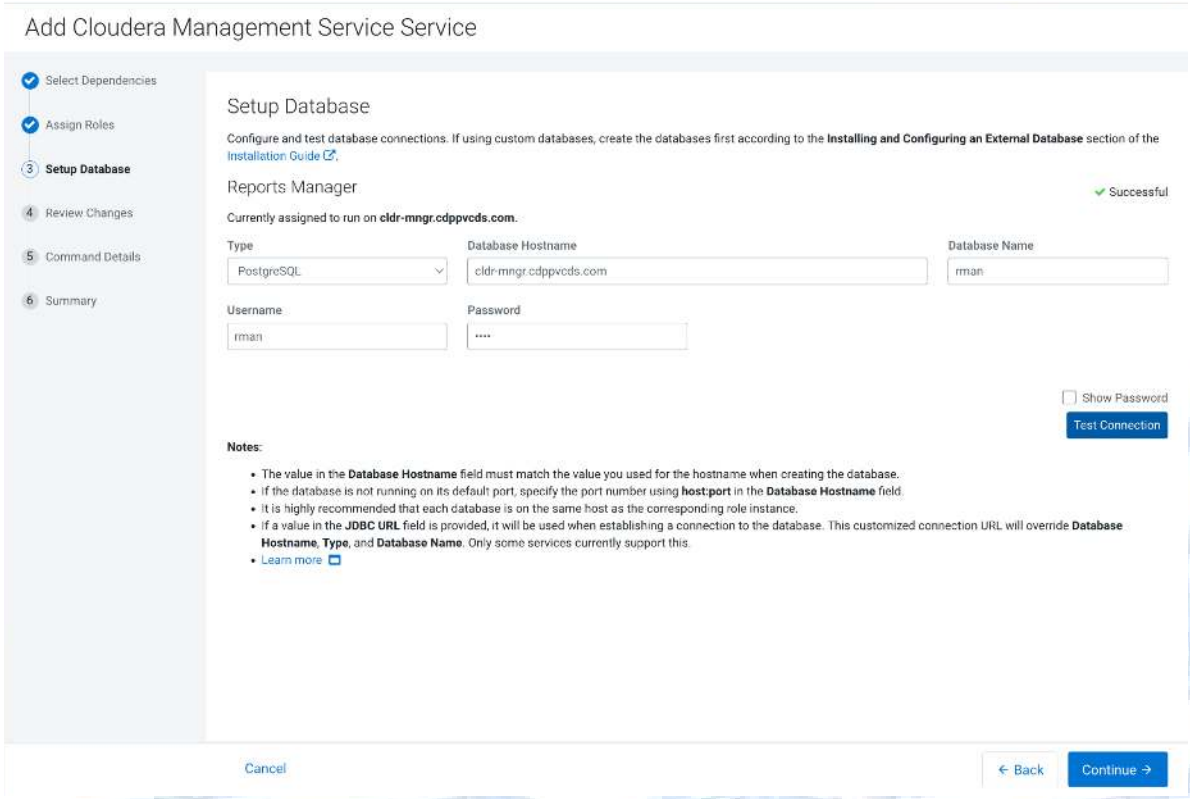
a) Go to —> WebUI -> Top Right Corner -> (+)Add -> Add Cloudera Management Service



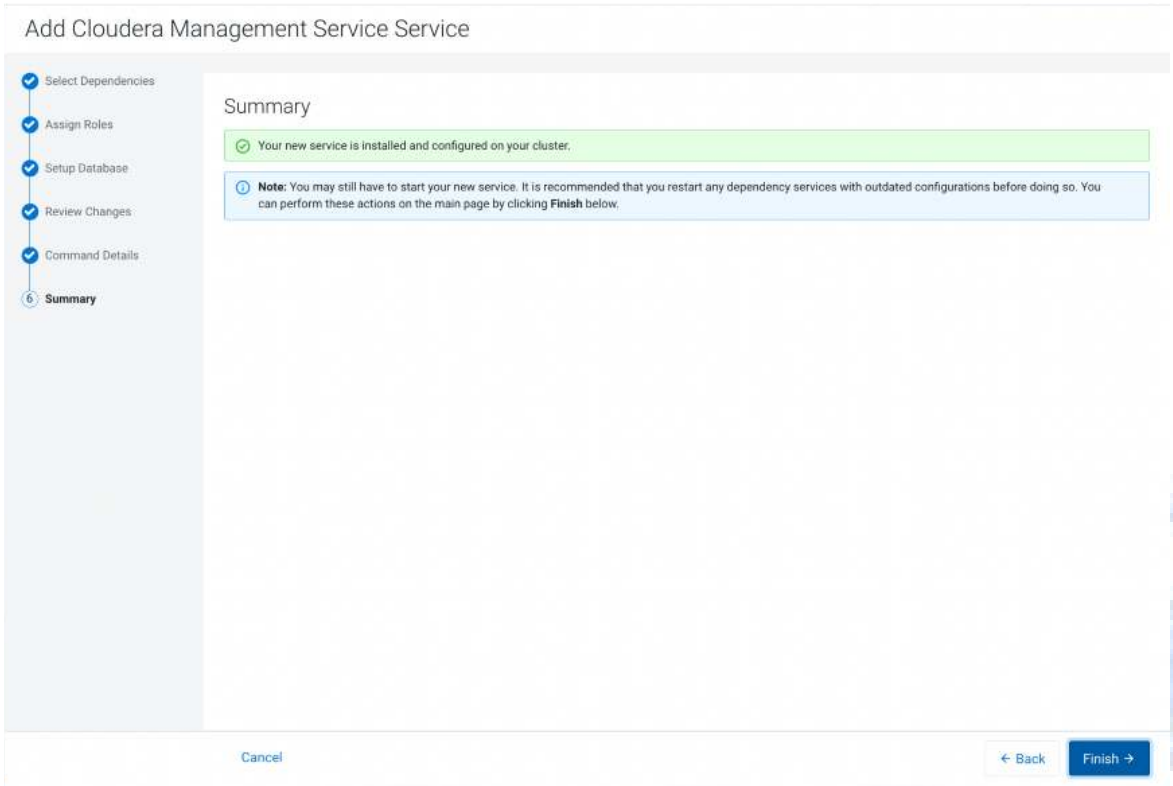
b) Assign Roles for different Cloudera Management Services to CLDR-MNGR Host (e.g. cldr-mngr.cldrsetup.local)



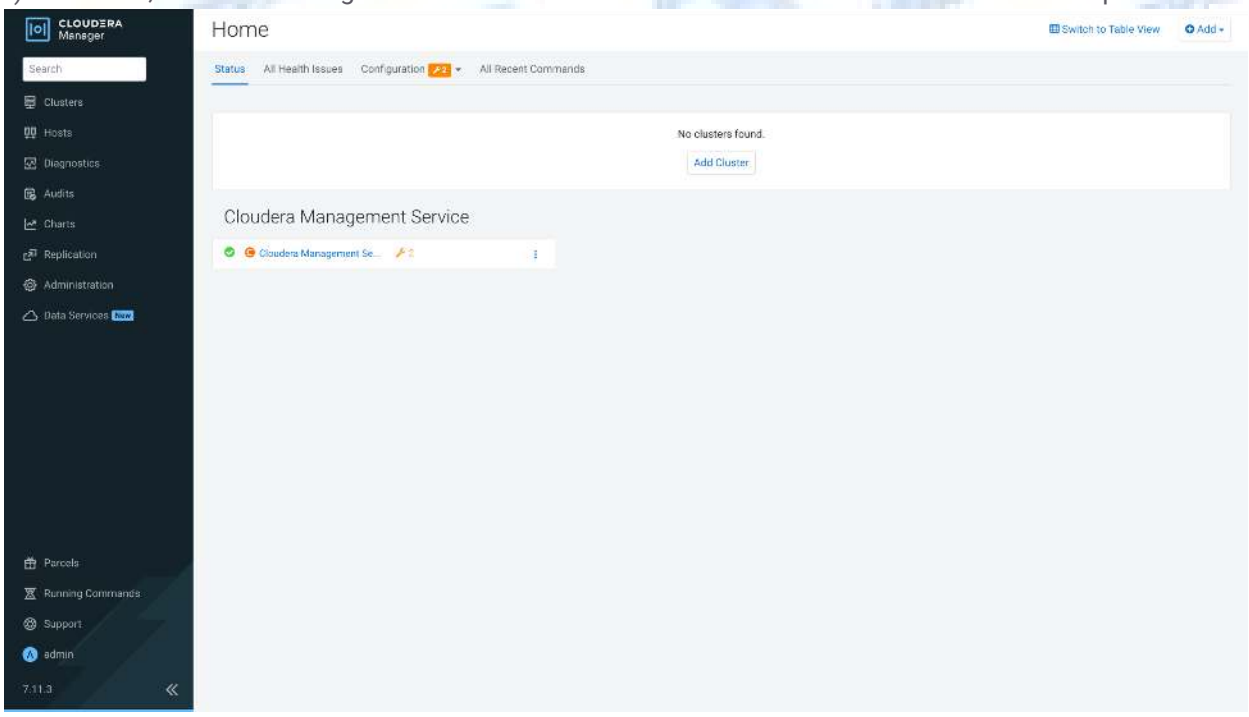
c) Setup Report Manager Database integration by providing the ***DBHostname***, ***DBNAME***, ***DBUser*** and ***DBPassword***. After entering the details, click on ***Test Connection.*** After the successful connection test, click ***Continue.***



d) ***Summary page*** will come up. Click on ***Finish***.



e) Now, Cloudera Management Service is visible as installed and status of different components is also visible.



****

Configure Cloudera Manager for external authentication using LDAP (LDAP integration):

An LDAP-compliant identity/directory service, such as OpenLDAP/FreeIPA, provides different options for enabling Cloudera Manager to look-up user accounts and groups in the directory:

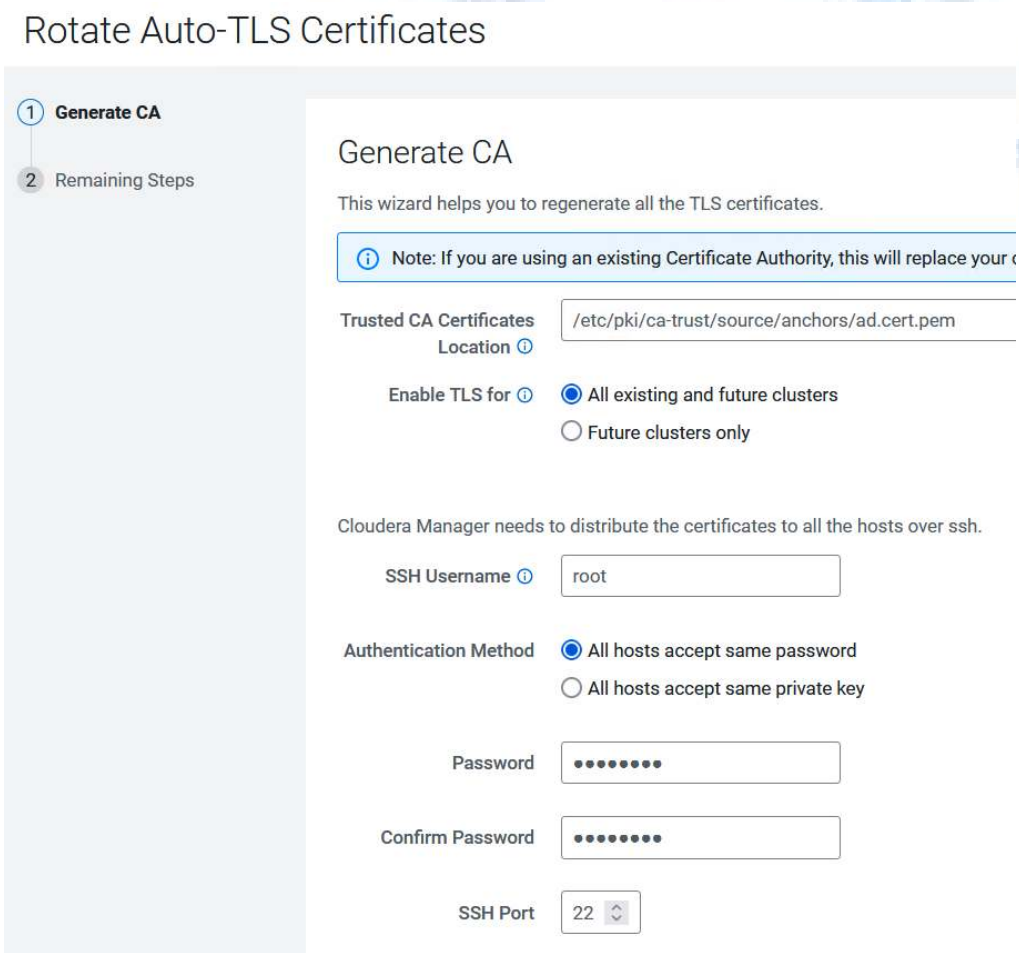
- Use a single Distinguished Name (DN) as a base for matching usernames in the directory, or
- Search filter options let you search for a particular user based on somewhat broader search criteria – for example Cloudera Manager users could be members of different groups or organizational units (OUs), so a single pattern does not find all those users. Search filter options also let you find all the groups to which a user belongs, to help determine if that user should have login or admin access.

Note: The ***LDAP Distinguished Name Pattern*** property is deprecated. Leave this field empty while configuring authentication using LDAP in Cloudera Manager.

Step1. Obtain CA certificate from active directory and copy it as for example. **(Only applied, in case you are going with an AD based setup)**

```
# cp ad.cert.cer /etc/pki/ca-trust/source/anchors/ad.cert.pem
# update-ca-trust force-enable
# update-ca-trust extract
# update-ca-trust check
```

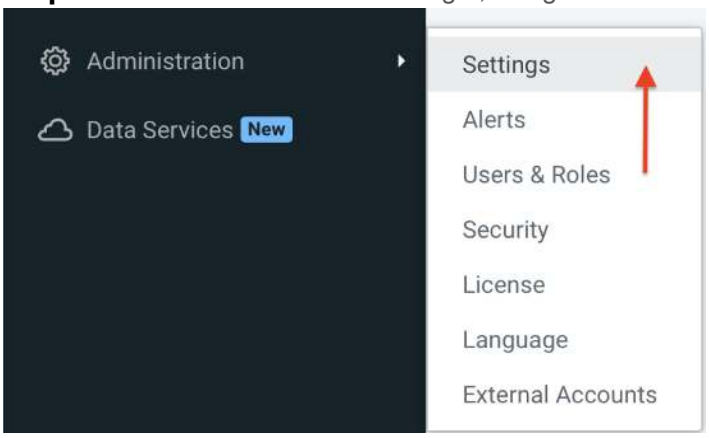
Step2. Update AutoTLS configuration by rotating Auto-TLS certificate with new CA certificate obtained. **(Only applied, in case you are going with an AD based setup).**



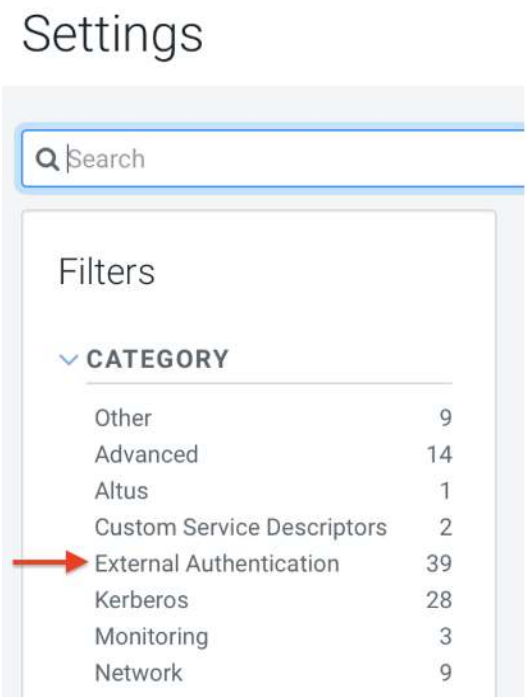
Step3. Restart clouder server configuration and restart cluster role/services and deploy client configuration.
(Only applied, in case you are going with AD based setup)

```
# systemctl restart cloudera-scm-server
# systemctl status cloudera-scm-server.service -l
# tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log
```

Step4. From Cloudera Manager, navigate to Administration→Settings.



Step5. In the filters section, click on External Authentication as shown in the screenshot below:



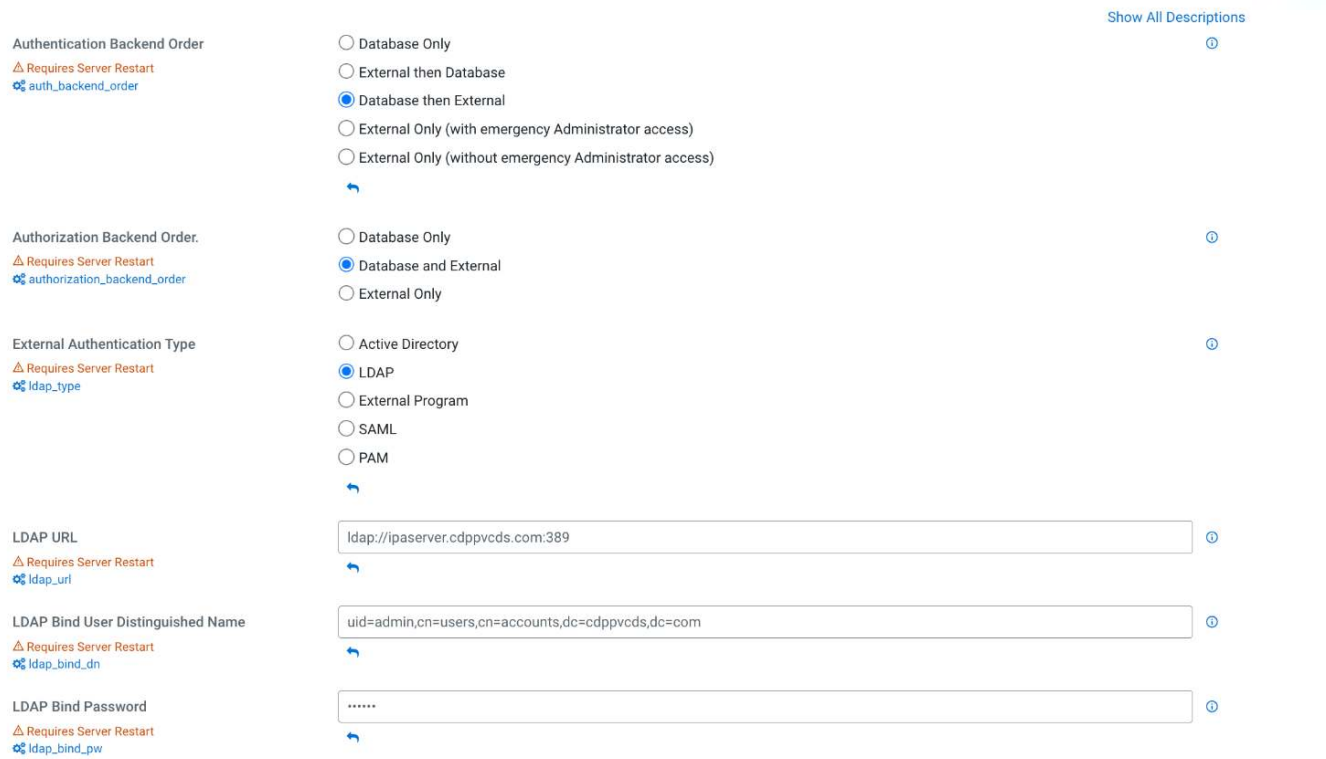
Step6. Search for “ldap” and enter values for ldap authentication, please refer to the sample values as mentioned in the below table and update according to your actual setup.

Table 7. LDAP Integration

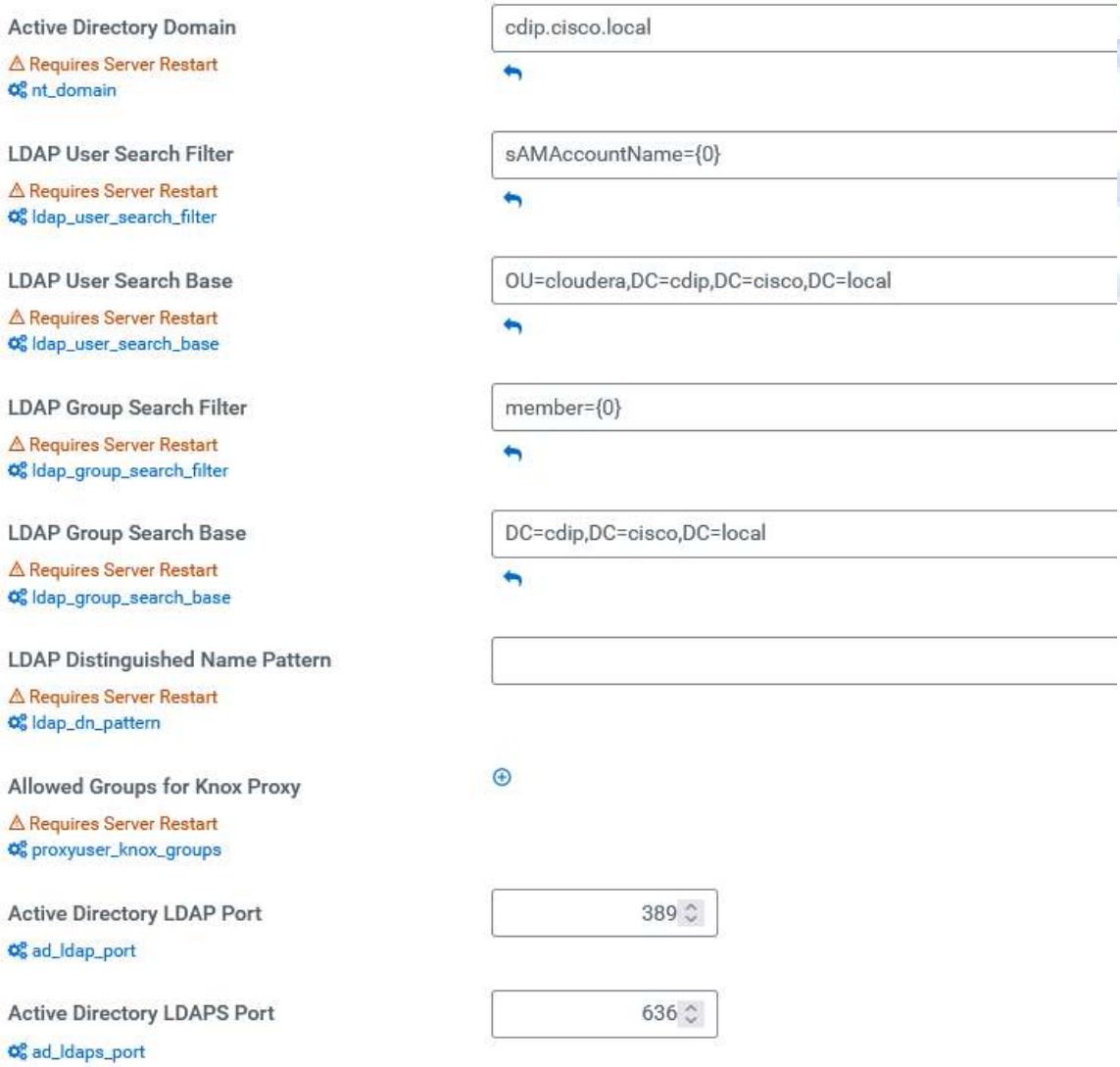
Component	Value
Authentication Backend Order:	Database then EXTERNAL

Component	Value
Authorization Backend Order:	Database and EXTERNAL
External Authentication Type:	LDAP
LDAP URL:	ldap://ipaserver.cldrsetup.local:389
LDAP Bind User Distinguished Name:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Bind Password:	*<cloudera123>* (password for KDC admin, configured earlier)
Active Directory Domain: **(For AD Based LDAP)**	<AD DOMAIN> (e.g CLDRSETUP.LOCAL)
LDAP User Search filter: **(For Open LDAP Based)**	(&(uid={0})(objectClass=person))
LDAP User Search filter: **(For AD Based)**	sAMAccountName={0}
LDAP User Search Base:	cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Group Search filter: **(For Open LDAP Based)**	(&(member={1})(objectClass=posixgroup))
LDAP Group Search filter: **(For AD Based)**	member={0}
LDAP Group Search Base:	cn=groups,cn=accounts,dc=cldrsetup,dc=local
LDAP DistName Pattern:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local

Step7. Below page will get open, select the appropriate options as mentioned below in the screenshot:



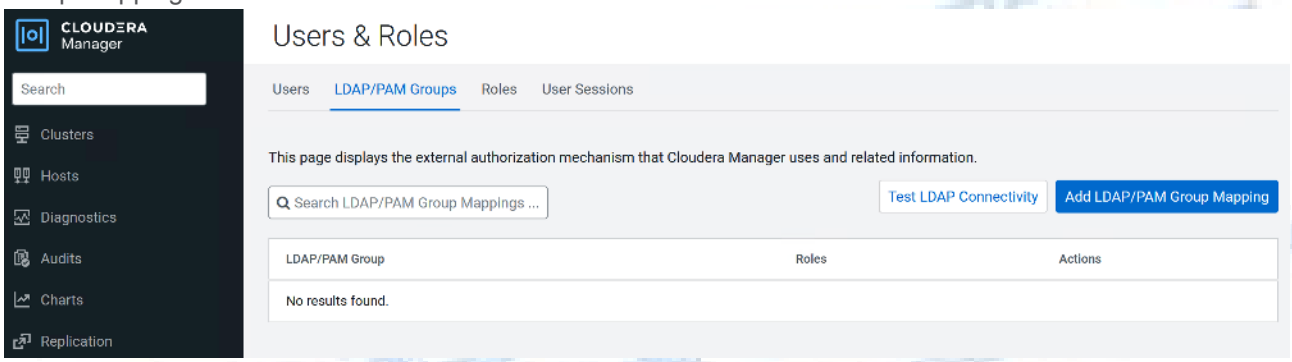
Additional Parameters for AD Based Setup:



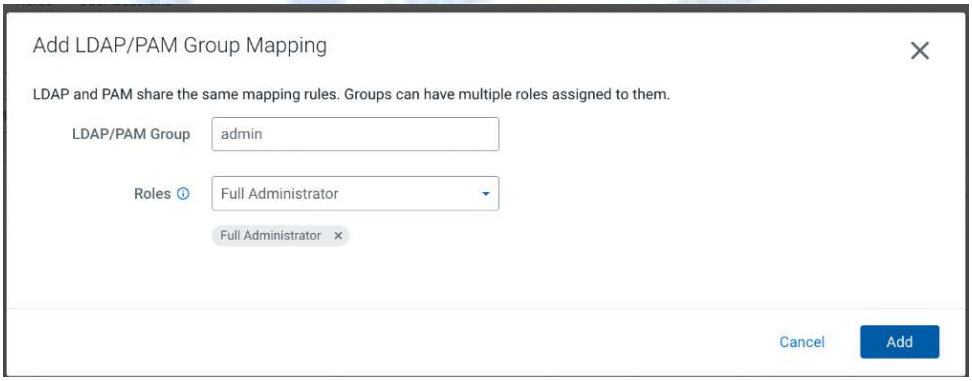
Step8. Click ***Save***. Once you click on the Save button, it will tell you to restart the CM Server, in order to bring the changes in effect. When you restart the CM Server from Backend. You will see the below entries in the logs.

```
[root@cldr-mngr ~]# systemctl restart cloudera-scm-server
[root@cldr-mngr ~]# systemctl status cloudera-scm-server -l

##### Run the below command to check the logs of cloudera-scm-server starting up. Wait until you see the
Started Jetty server message on the screen.
[root@cldr-mngr ~]# sudo tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log

2024-08-21 03:14:10,007 INFO WebServerImpl:com.cloudera.server.cmf.ExternalAuthenticationHelper: Using
LDAP authentication with properties: DN pattern=(uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local)
user search base=(cn=users,cn=accounts,dc=cldrsetup,dc=local) user search
filter=((&(uid={0})(objectClass=person))) group search base=(cn=groups,cn=accounts,dc=cldrsetup,dc=local)
group search filter=((&(member={1})(objectClass=posixgroup)))

2024-08-21 03:14:10,009 INFO WebServerImpl:com.cloudera.server.cmf.WebServerImpl: Authenticating against
database, then LDAP
```

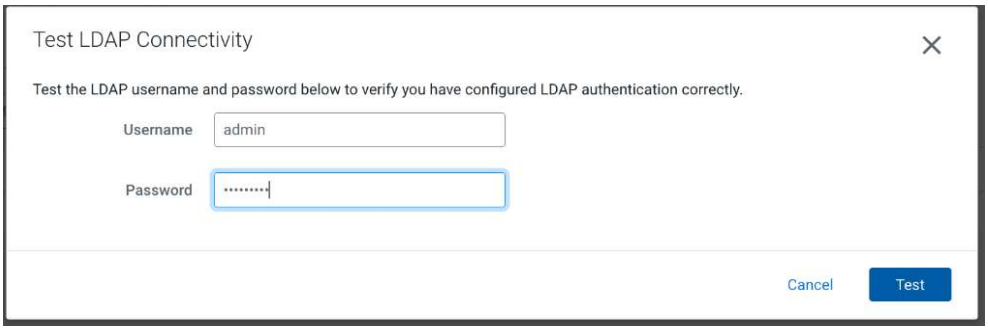
Step9. Login again to CM-UI and in ***Administration > Users & Roles > LDAP/PAM Groups***, add LDAP/PAM Group mapping.



Step10. ***Add LDAP/PAM Group mapping*** value and Roles to assign. (***LDAP/PAM Group: admin, Roles: Full Administrator***)



Step11. Click on ***Test LDAP Connectivity.*** Provide a username and password for an LDAP user to test whether that user can be authenticated. ***(username: admin, password: <cloudera123>)***

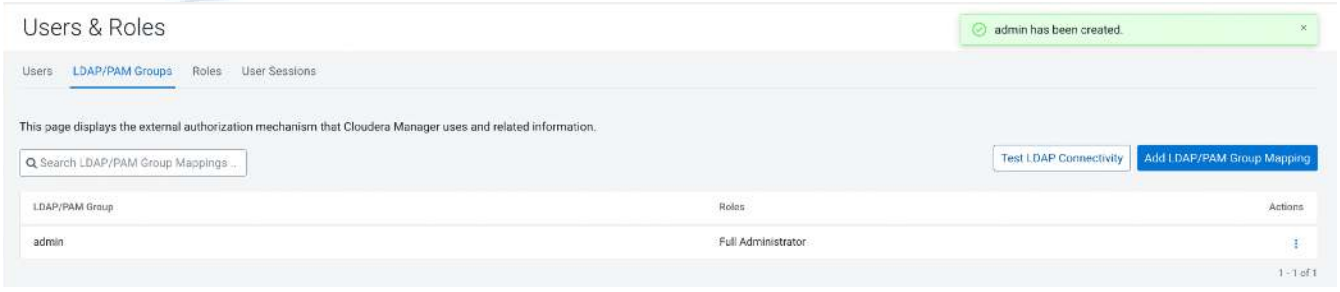


Step12. Click on ***Test*** to verify LDAP configuration is set up and working fine, as expected.

Step13. Login to the cldr-mngr server at backend and restart the Cloudera Manager Server.

```
[root@cldr-mngr ~]# systemctl restart cloudera-scm-server
[root@cldr-mngr ~]# systemctl status cloudera-scm-server -l
[root@cldr-mngr ~]# sudo tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log
2024-03-19 00:56:15,620 INFO
pool-7-thread-1:com.cloudera.server.cmf.components.CmServerStateSynchronizer: (30 skipped) Synced
up
2024-03-19 00:46:49,935 INFO LDAP login monitor thread.:
org.springframework.security.ldap.DefaultSpringSecurityContextSource:  URL
'ldap://ipaserver.cldrsetup.local:389/cn=users,cn=accounts,dc=cldrsetup,dc=local', root DN is
'cn=users,cn=accounts,dc=cldrsetup,dc=local'
2024-03-19 00:56:13,528 INFO LDAP login monitor thread.:
org.springframework.ldap.core.support.AbstractContextSource: Property 'password' not set - blank
password will be used
2024-03-19 00:56:14,269 INFO CommandPusher-1:com.cloudera.server.cmf.CommandPusherThread: Acquired
lease lock on DbCommand:1546344098
2024-03-19 00:56:14,620 INFO
pool-7-thread-1:com.cloudera.server.cmf.components.CmServerStateSynchronizer: (30 skipped) Cleaned
up
[root@cldr-mngr ~]#
```

Step14. Login to Cloudera Manager WebUI and assign Roles for new users. ***(Only, If used a different user than admin, else skip this step)***



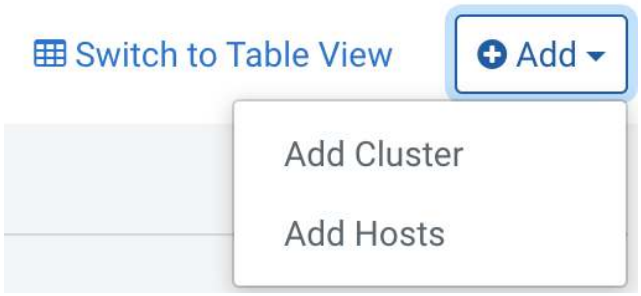
****

Setup Cloudera on premises (PvC) Base Cluster

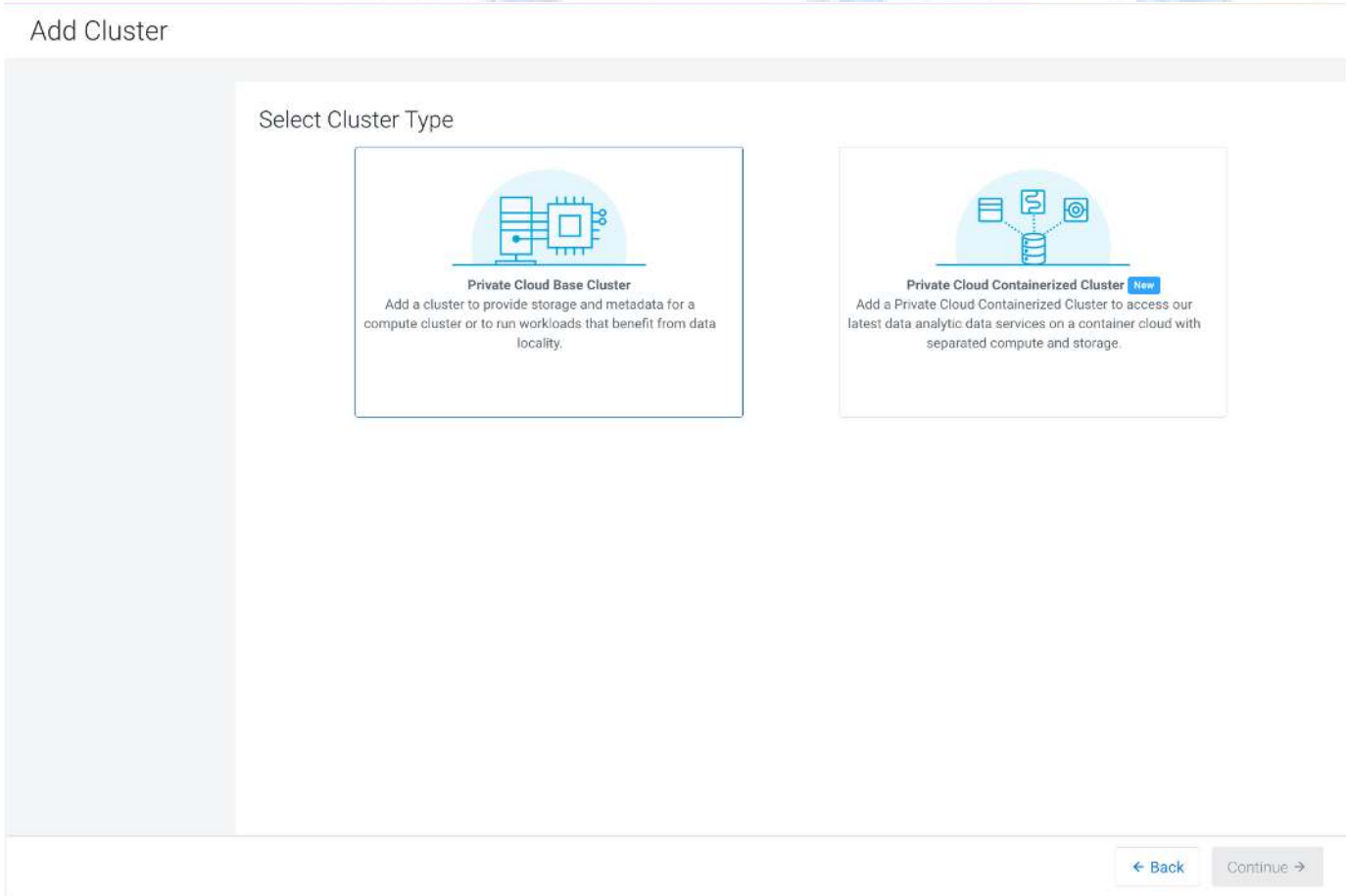
In this step, we will setup the Base cluster which will serve as the DataLake for the CDP Data Services that need the SDK capabilities for the cluster wide features like lineage, governance, security etc..,

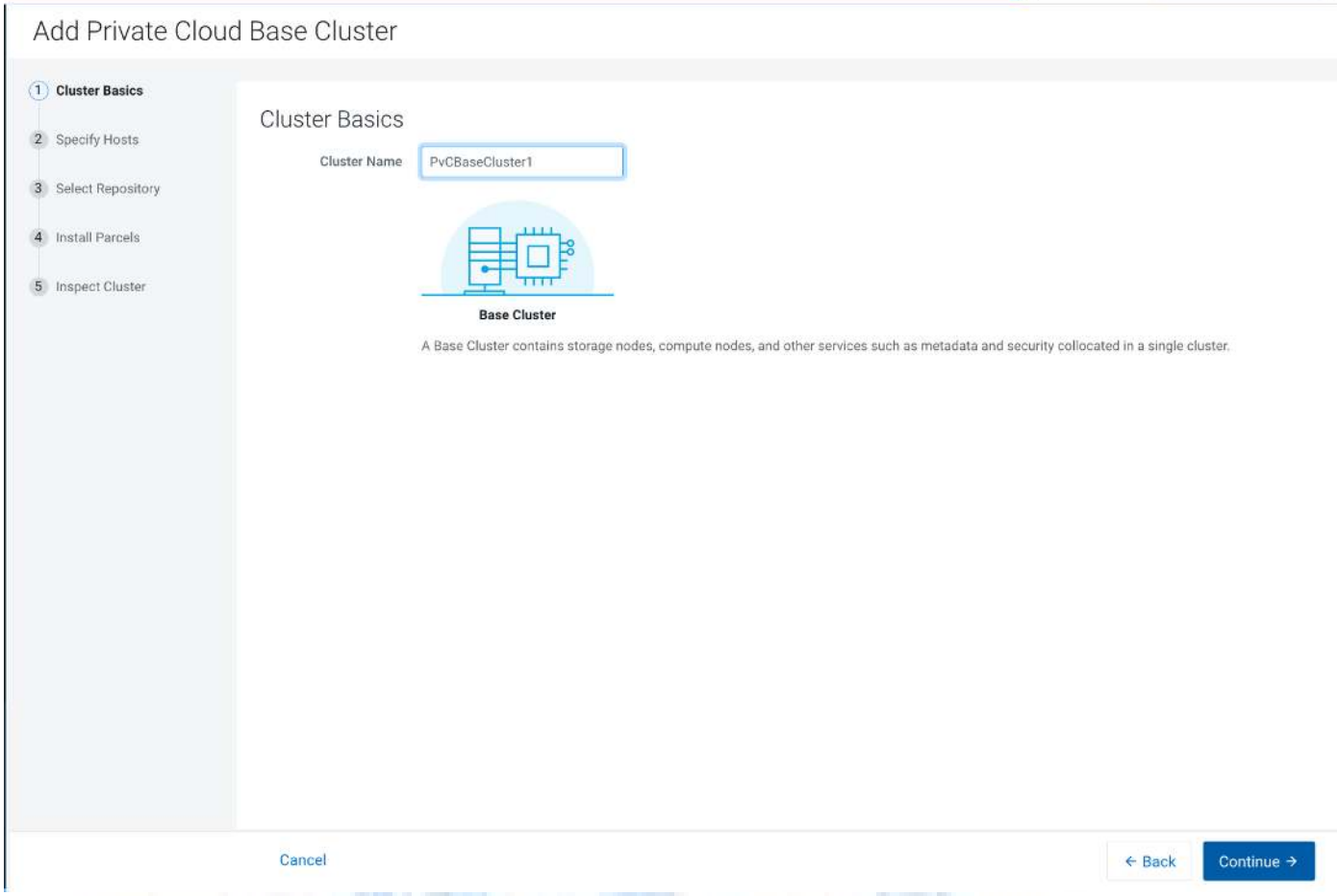
Procedure 9. Install Cloudera on premises Base using the Cloudera Manager WebUI

Step 1. In Cloudera Manager, on the top right corner, click ***(+) Add > Add Cluster***. The Select Cluster Type page appears.



Step 2. On the ***Select Cluster Type*** page, select the cluster type as ***Cloudera on premises Base Cluster*** (first option) and enter a cluster name. Click ***Continue***.



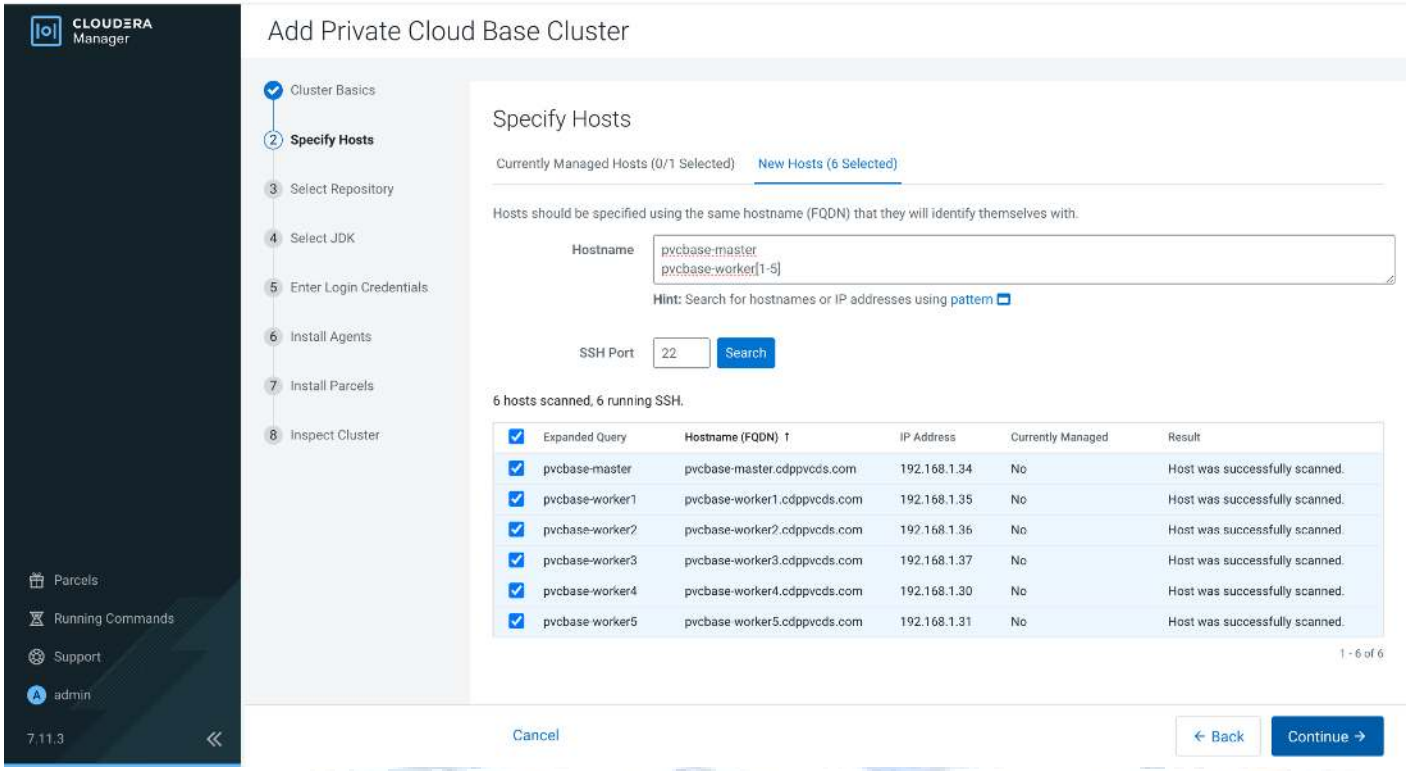


Step 3. Enter the cluster host names or IP addresses in the Hostnames field. You can Provide the host pattern pvcbase-master, pvcbase-worker[1-3] or pvcbase-worker[1-3].cldrsetup.local etc separated with a new line and Click on ***Search***.

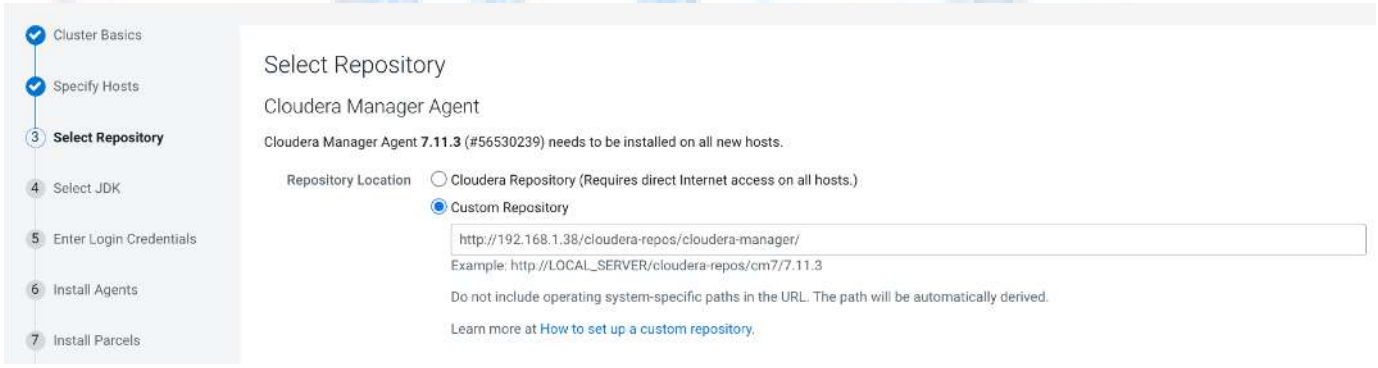
Note: Host names must be in lowercase. If you use uppercase letters in any host name, the cluster services will not start after enabling Kerberos.

Step 4. Specify the hosts that are part of the cluster using their IP addresses or hostname. The figure below shows a pattern that specifies the IP addresses range. Cloudera Manager will "discover" the hosts based on matching the pattern provided by you to add in the cluster. Verify that all desired nodes have been found and ***selected for installation***. Verify host entries, ***deselect*** any that you do not want to install services on, and click ***Continue***.

```
pvcbase-master.cldrsetup.local
pvcbase-worker[1-3].cldrsetup.local
```

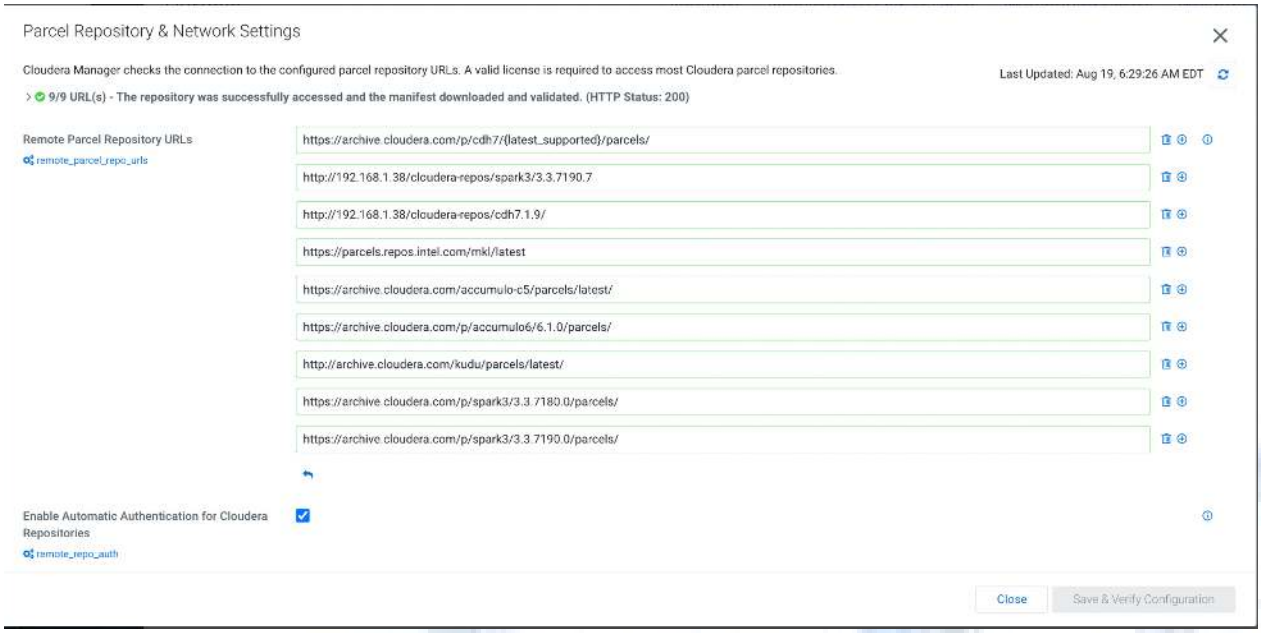


Step 5. The Select Repository section appears. Select Cloudera Repository option as mentioned. Enter ***Custom Repository*** or Cloudera Repository to install Cloudera Manager Agent on all nodes in the cluster. We have earlier configured the private yum repository on ***cldr-mngr*** node. Please provide the path here:
i.e. http://13.251.65.11/cloudera-repos/cloudera-manager/ and click on ***Continue***.

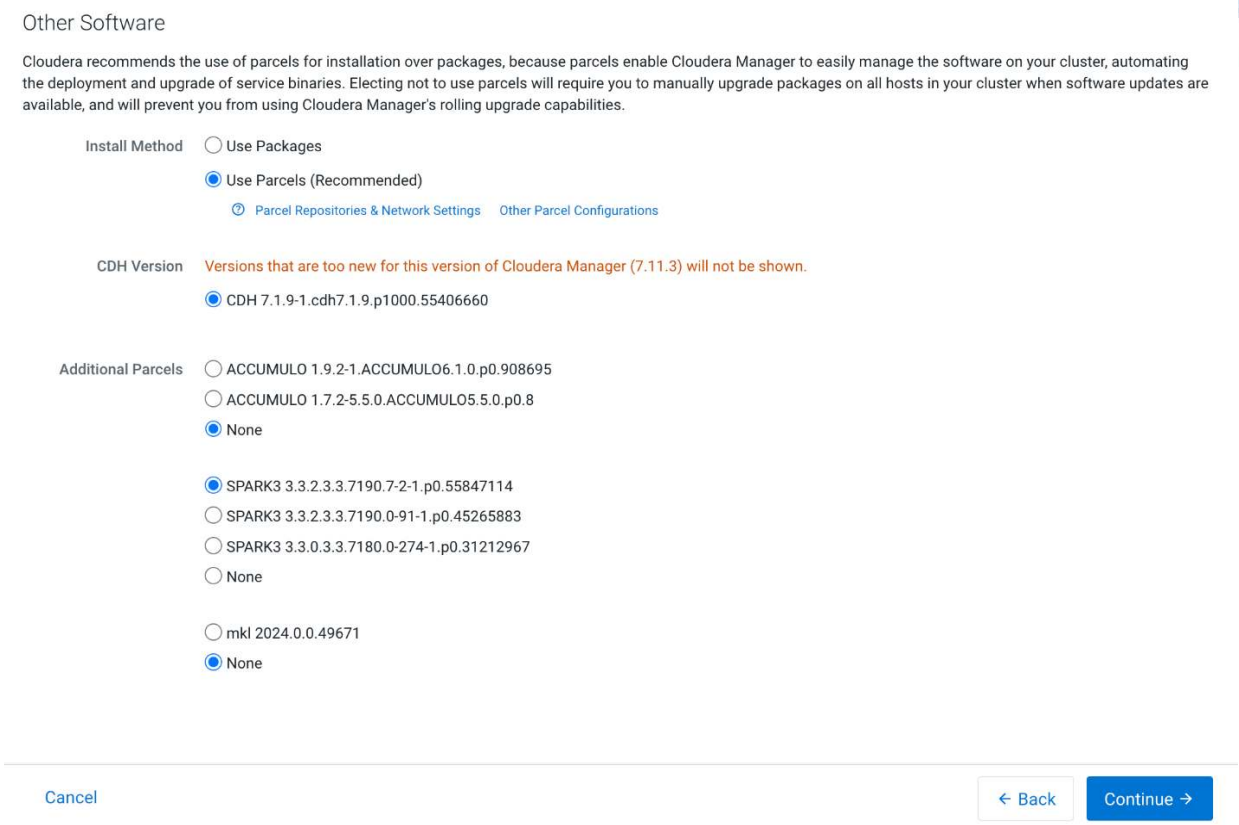


Step 6. In the other software section, select ***Use Parcels (Recommended)*** and click ***Parcel Repository & Network Settings*** to provide a custom Parcels location to be installed ***(in a new tab in the same browser window)***.

Step 7. Enter custom repository URL for CDH7 and CDS 3.3 parcels. Click on ***Save and Verify Configuration***. Close the Parcel Repository & Network Settings wizard.
i.e. http://13.251.65.11/cloudera-repos/spark3/3.3.7191000.4/
http://13.251.65.11/cloudera-repos/cdh7.3.1/

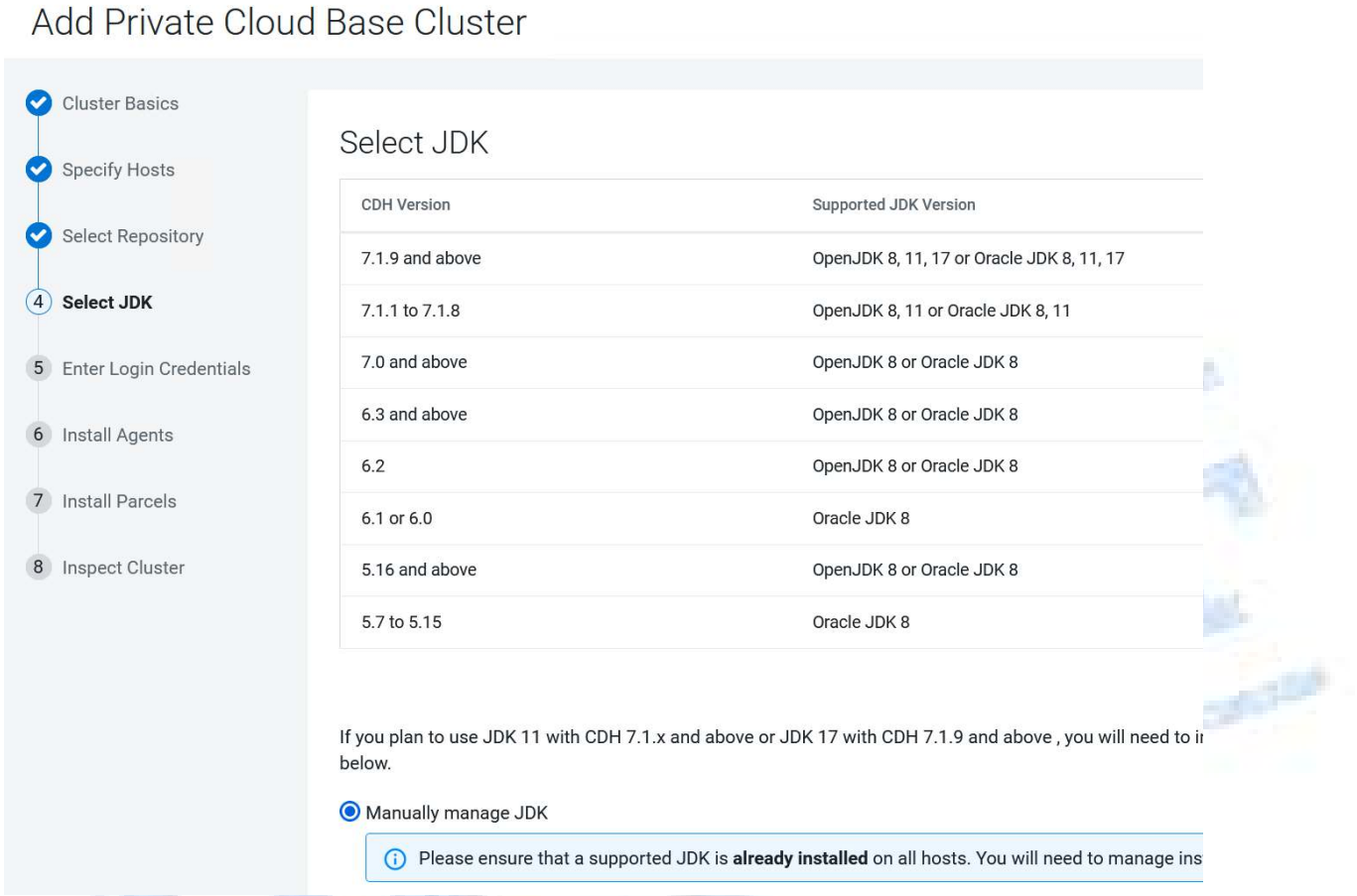


Step 8. Select the parcels for installation.



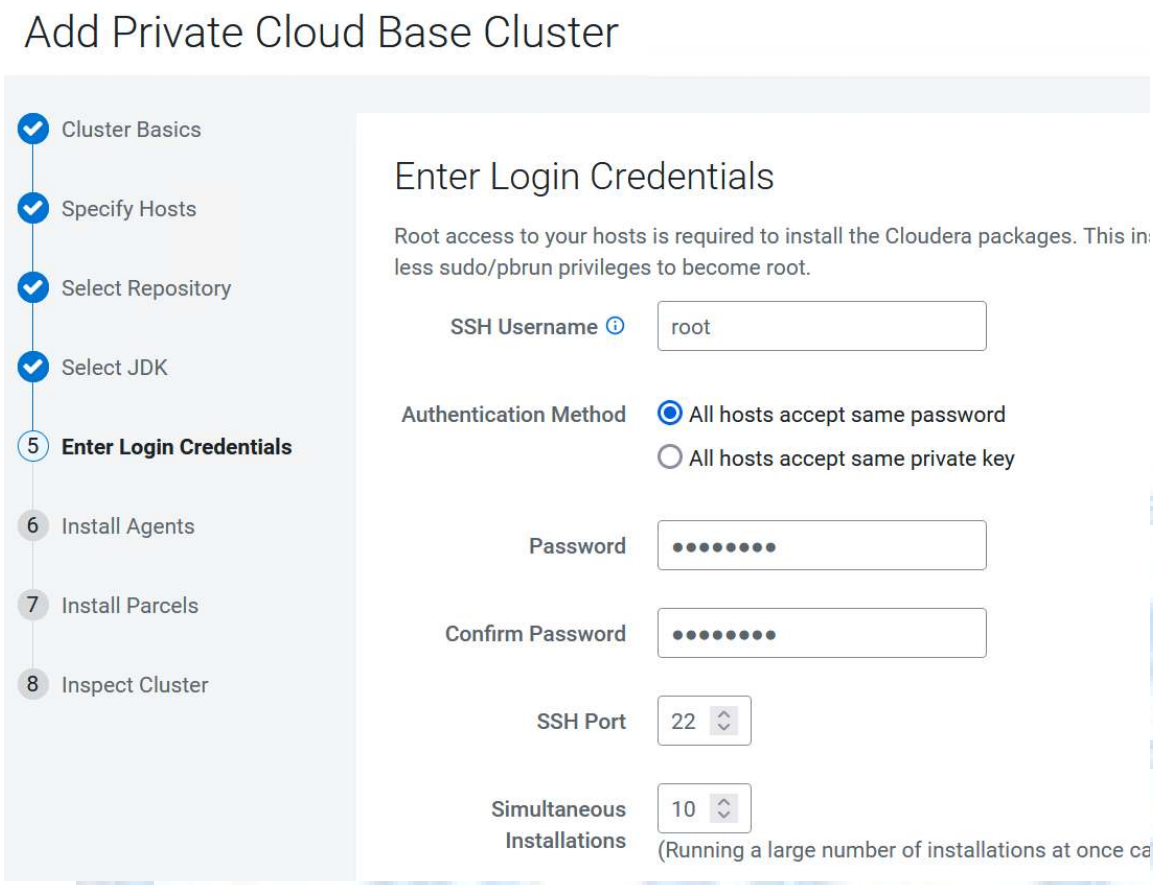
Step 9. Click *Continue*.

Step 10. Select the appropriate option for JDK. (manual installation for JDK11 with CDH 7.1.x and JDK17 with 7.3.1 and above): (Select ***Manually manage JDK*** here, as we have already installed a System-provided version of OpenJDK11 manually on all servers. Click on ***Continue***.

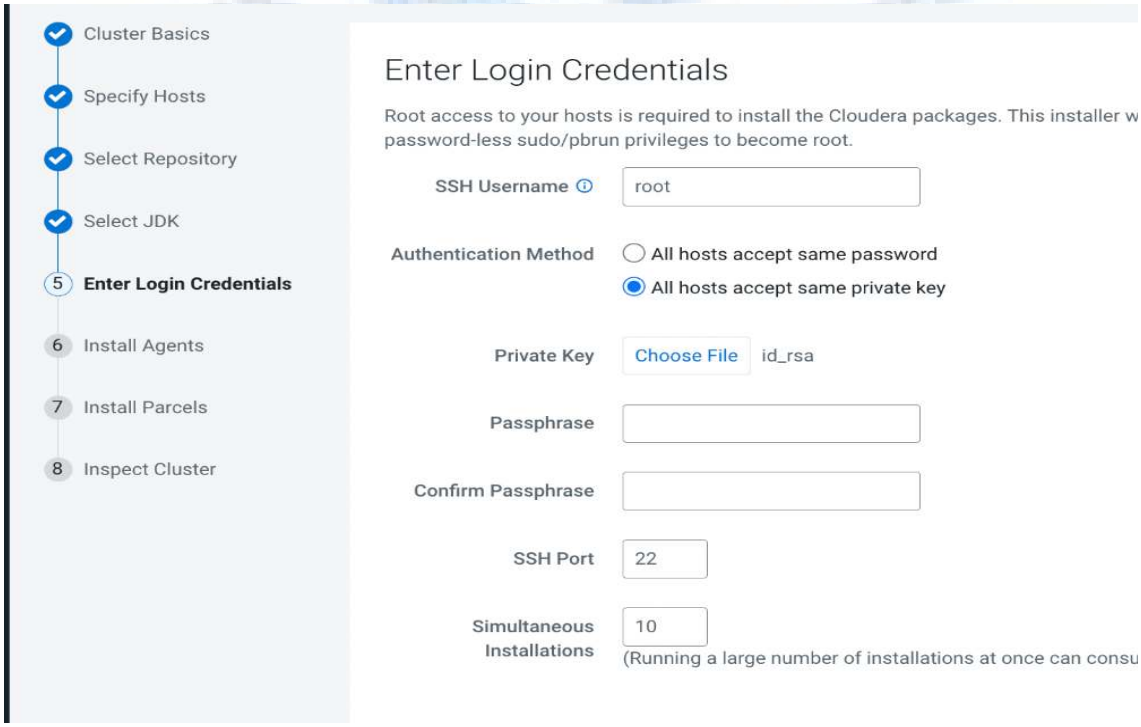


Step 11. The ***Enter Login Credentials section*** appears. On this page, provide the required ***details*** as mentioned ***in the below table***, for the hosts to install Cloudera packages. Click ***Continue***.

Component	Value
SSH Username	root
Authentication Method	All hosts accept same password / All hosts accept same private key
Password (If selected password based auth)	<password_for_vm_root_user> ***(e.g. cloudera123)***
Confirm Password (If selected password based auth)	<password_for_vm_root_user> ***(e.g. cloudera123) (again)***
Private Key (If selected private key based auth)	Upload the private key e.g. ***id_rsa*** generated in earlier steps
Passphrase (If selected private key based auth)	If Applicable
Repeat Passphrase (If selected private key based auth)	If Applicable

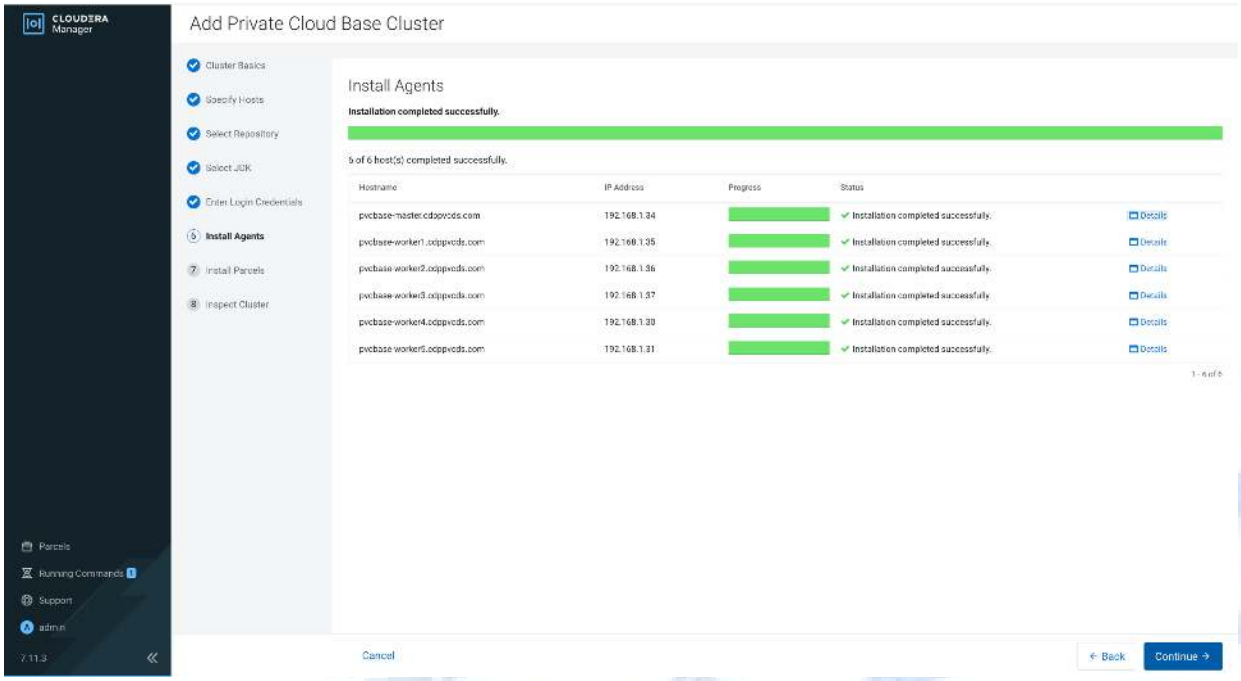


a. If Selected the ***Authentication method*** as ***All hosts accept same password***



b. If Selected the ***Authentication method*** as ***All hosts accept the same private key***.

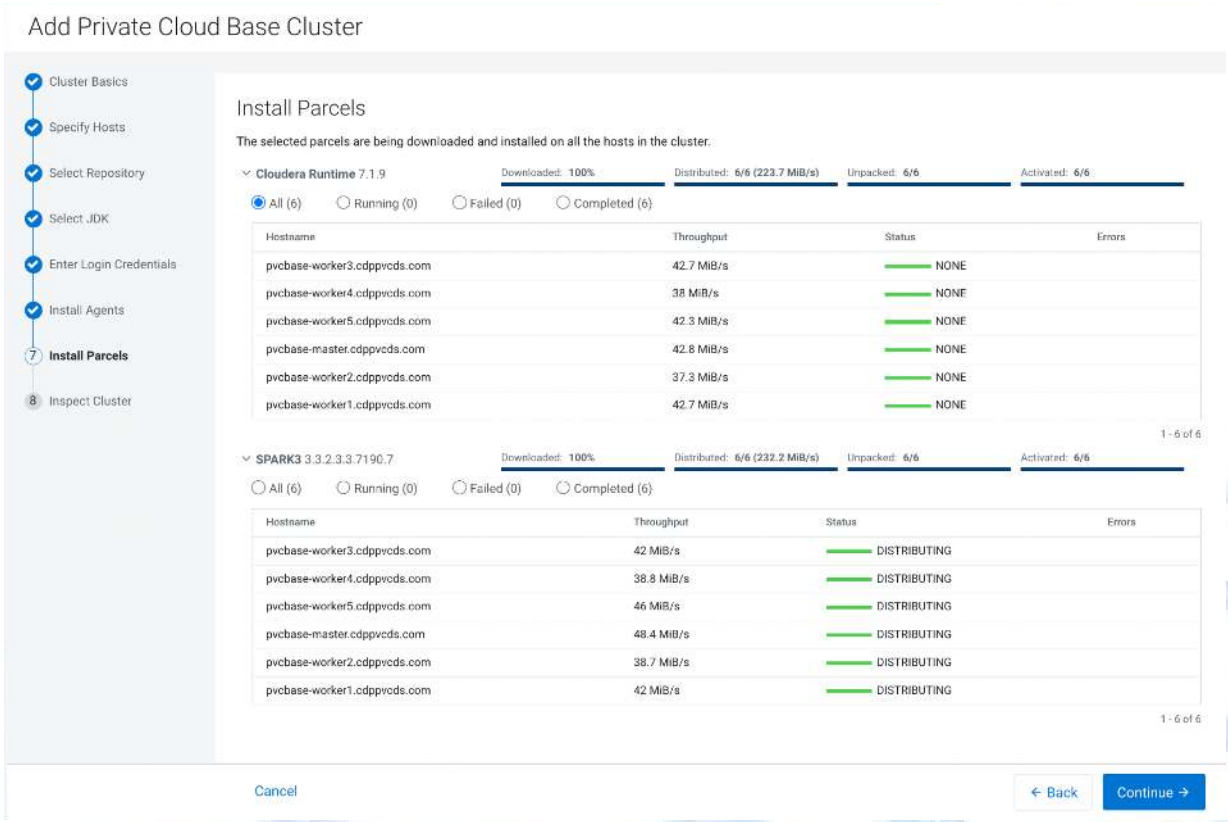
Step 12. The ***Install Agents*** section appears showing the progress of the installation. It will check for and install the ***JDK*** (If not already there) and ***cloudera-scm-agent*** software on all the Base Cluster nodes. Click ***Continue*** after the Cloudera Agent ***Installation completed successfully*** on all hosts.



Step 13. Stop at this stage. Create a directory on base cluster nodes (for handling of a bug associated with p1000)

```
[root@cldr-mngr ~]# ansible namenodes,datanodes -m shell -a "mkdir /var/lib/hadoop-hdfs"
[root@cldr-mngr ~]# ansible namenodes,datanodes -m shell -a "ls -lart /var/lib/hadoop-hdfs"
```

Step 14. After the agents are installed, the ***Install Parcels*** section appears showing the progress of the parcels distribution, activation and installation on all hosts part of the cluster creation. Once the parcels are installed successfully for all hosts, click on ***Continue***.



Step 15. Stop at this stage. On base cluster nodes (for handling of a bug associated with p1000). Once Parcels are activated, follow below steps: **(Failed to execute command Install YARN MapReduce Framework JARs on service YARN)**

```
[root@cldr-mngr ~]# ansible namenodes,datanodes -m shell -a "chown hdfs:hadoop /var/lib/hadoop-hdfs"
[root@cldr-mngr ~]# ansible namenodes,datanodes -m shell -a "ls -lart /var/lib/hadoop-hdfs"

##### Verify if links are good for the CM version for hadoop-hdfs filesystem JAR.

[root@cldr-mngr ~]# ansible namenodes,datanodes -m shell -a "namei -om
/var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar"
If jar not found on the host, you can download using the link below:
wget
https://repository.cloudera.com/repository/cloudera-repos/org/apache/hadoop/hadoop-ozone-filesystem-had
oop3/1.1.0.7.2.15.0-147/hadoop-ozone-filesystem-hadoop3-1.1.0.7.2.15.0-147.jar

If links are not working fine, deactivate and activate the parcel from CM Parcel Manager. Then Perform
Below steps:
[root@cldr-mngr ~]# ansible namenodes,datanodes -m shell -a "unlink
/etc/alternatives/ozone-filesystem-hadoop3.jar; unlink
/var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar"

[Notes]
Here are the notes from product/engineering team on the issue:

Verify if /var/lib/hadoop-hdfs exists on all nodes.
Check if hdfs user exists on all nodes. (grep hdfs /etc/passwd; grep hdfs /etc/group)

Create the /var/lib/hadoop-hdfs directory on all nodes.

Deactivate and activate the Cloudera Runtime parcel.
Alternatively, if there is a separate Ozone parcel installed on the cluster, deactivate and activate
the Ozone parcel instead.

Ensure that the directory has at least the following permissions:
/var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar
```

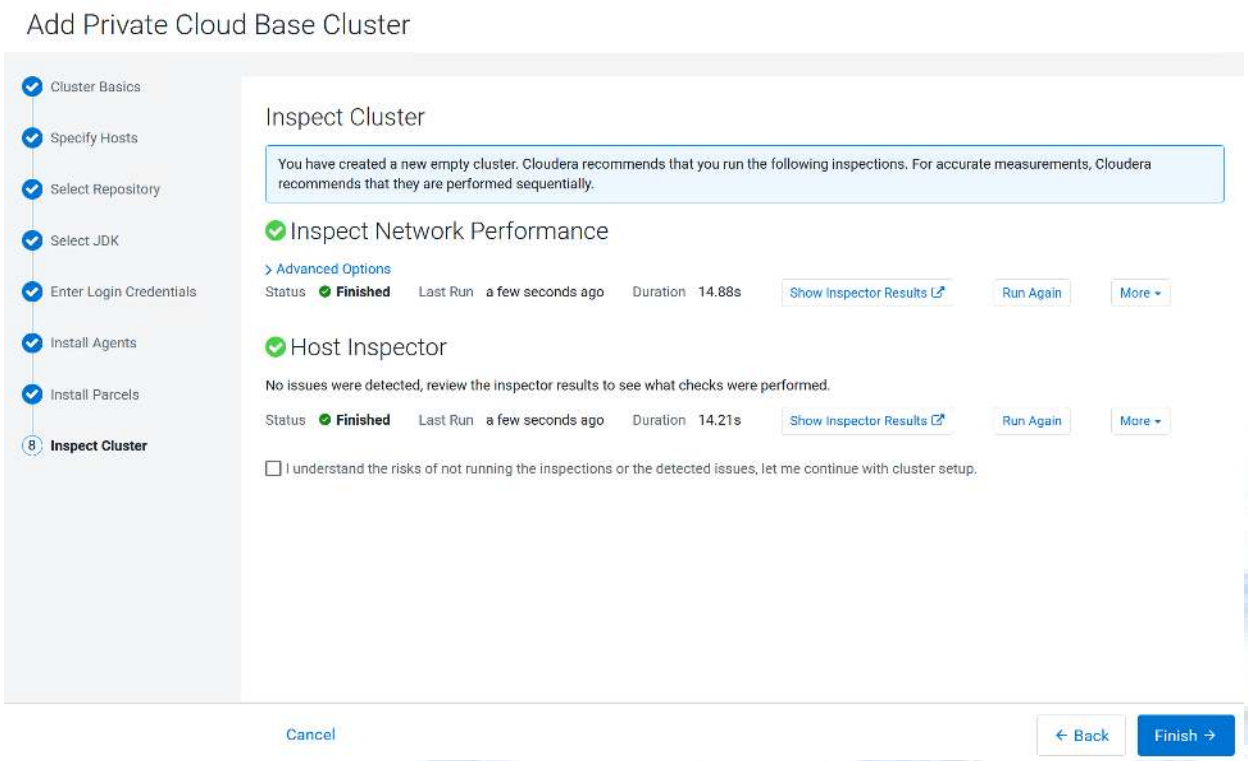
```
lrwxrwxrwx [owner: root, group: root]
/var/lib/hadoop-hdfs
drwxr-xr-x [owner: hdfs, group: hadoop]

Output for permissions should look similar to:
$ namei -om /var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar

f: /var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar
 drwxr-xr-x root         root         /
 drwxr-xr-x root         root         var
 drwxr-xr-x root         root         lib
 drwxr-xr-x hdfs         hadoop       hadoop-hdfs
 lrwxrwxrwx root         root         ozone-filesystem-hadoop3.jar ->
/etc/alternatives/ozone-filesystem-hadoop3.jar

The issue can also be found in the Ozone Known issues documentation.

Create the alternatives link manually for CDP binaries.
This step would have failed during the activation of the CDP parcel due to the missing folder.
$ alternatives --install /var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar ozone-filesystem-hadoop3.jar
/opt/cloudera/parcels/CDH-<path_to_active_parcel>/lib/hadoop-ozone/ozone-filesystem-hadoop3.jar 5

If an Ozone parcel is installed, create the symbolic links for the Ozone parcel.
$ alternatives --install /var/lib/hadoop-hdfs/ozone-filesystem-hadoop3.jar
ozone-filesystem-hadoop3.jar /opt/cloudera/parcels/OZONE-<path_to_activated
Ozone_parcel>/lib/hadoop-ozone/ozone-filesystem-hadoop3.jar 10

Upgrade:
Upgrade to a fixed release of CDP/Cloudera Manager once available.

For the latest update on this issue see the corresponding Knowledge Article:
TSB 2024-775: FileNotFoundException for the Ozone FS JAR during or after installation or upgrade
```

Step 16. After the parcels are installed the Inspect Cluster section appears.

Inspect Cluster by running ***Inspect Network Performance***. After the network inspector completes, click ***Show Inspector Results*** to view the results in a new tab. Address any reported issues (If there), and click ***Run Again***.

Click Inspect Hosts. After the host inspector completes, click ***Show Inspector Results*** to view the results in a new tab. Address any reported issues (If there), and click ***Run Again***.

Both the inspection tests should run successfully. Review inspector summary. Click ***Finish***.



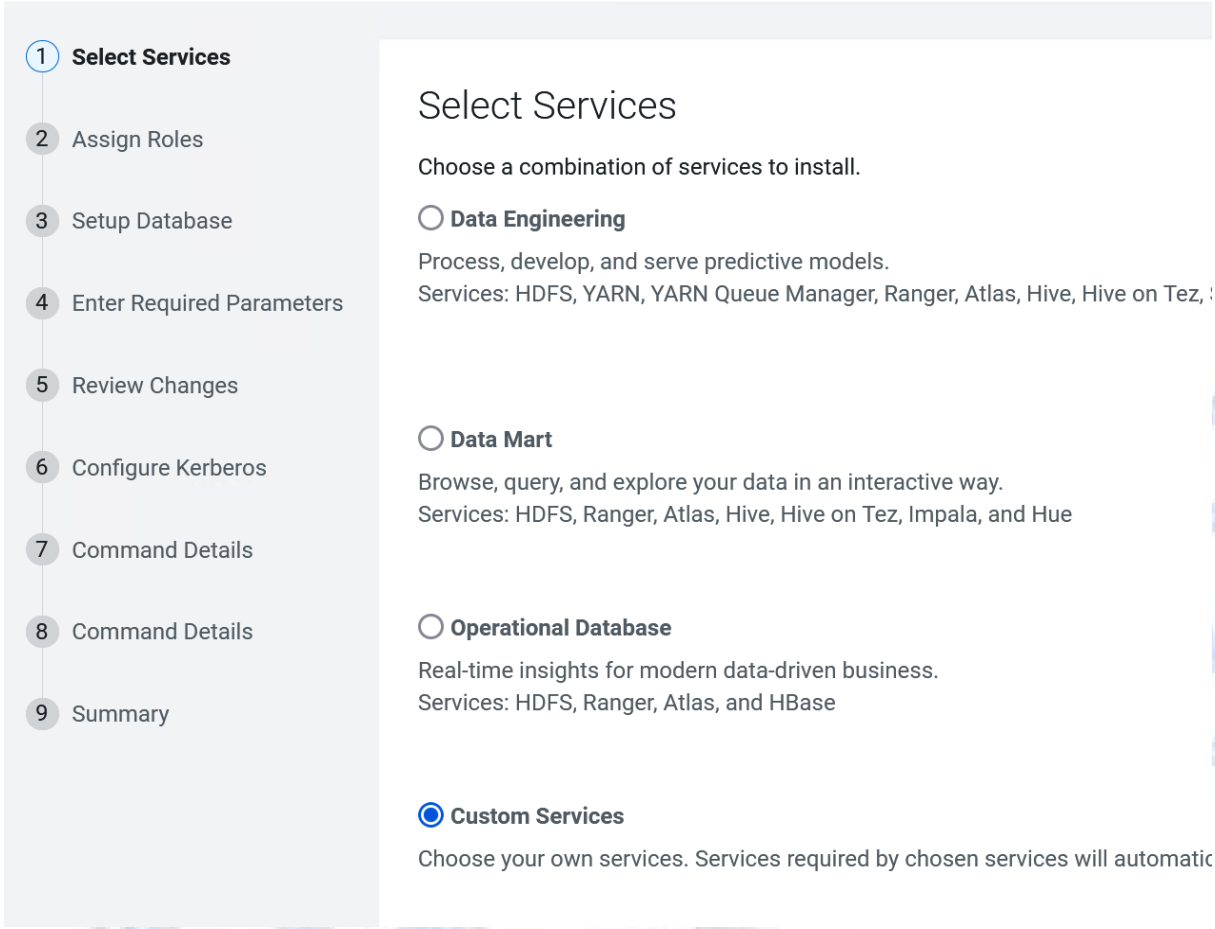

```
If java path mismatch error occurs in network/host inspection:
Take - Java Path from the host ( /usr/lib/jvm/java-17-openjdk-17.0.13.0.11-4.el9.x86_64/ ), goto CM,
open a new tab of CM-UI on browser and search for JAVA PATH in search bar present at left hand side:
# Override JAVA PATH in CM-UI> Search JAVA PATH> Override the value> Save Changes> Restart
Cloudera-SCM-Server
Re-run the inspect tools, this time all checks should be green.
```

Step 17. Click ***Continue***.

Cloudera on premises Base Cluster (Data Lake) Creation

Step 18. After the Cloudera on premises Base Cluster (runtime) setup is complete, if successful, it will automatically move to add services ***(Add Cluster -Configuration)*** wizard. It will ask to ***Select Services*** i.e. (a) Data Engineering, (b) Data Warehouse/ Data Mart, © Operational Database specific or (d) Custom control plane/base cluster services i.e. HDFS, YARN, Hive, Tez, HiveOnTez, Ozone, Zookeeper, Kafka, SOLR, Ranger, Atlas, HBase, Phoenix, Impala, HUE, Spark2, Spark3, YARN Queue Manager etc. Choose from a combination or services defined or select custom services. Services required based on selection will be automatically added.

Step 19. Select ***Custom Services*** option to install.



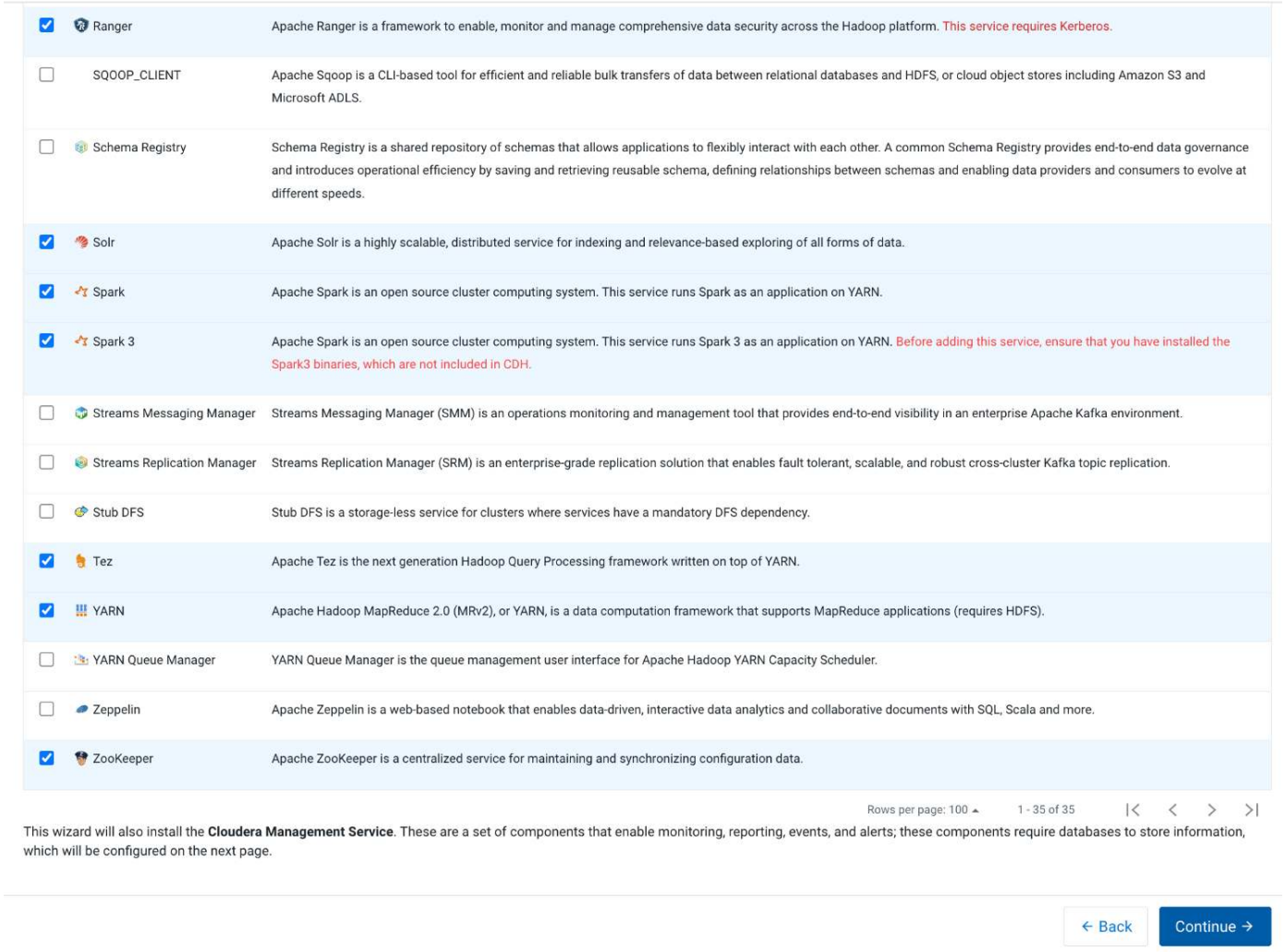
Note: It is important to select host(s) to deploy services based on the role intended for it. For detailed information, go to: [Runtime Cluster Hosts and Role Assignments]

Step 20. Under the ***Custom Services***, select the below custom Cloudera DataLake/Control Plane management services to be installed on the cluster. The Selection would look similar to below, Select services and Click ***Continue***.

Custom Services

Choose your own services. Services required by chosen services will automatically be included.

	Service Type	Description
☑	Atlas	Apache Atlas provides a set of metadata management and governance services that enable you to find, organize, and manage data assets. This service requires Kerberos.
☐	Cruise Control	Cruise Control simplifies the operation of Kafka clusters automating workload rebalancing and self-healing.
☑	HBase	Apache HBase is a highly scalable, highly resilient NoSQL OLTP database that enables applications to leverage big data.
☑	HDFS	Apache Hadoop Distributed File System (HDFS) is the primary storage system used by Hadoop applications. HDFS creates multiple replicas of data blocks and distributes them on compute hosts throughout a cluster to enable reliable, extremely rapid computations.
☑	Hive	Apache Hive is a SQL based data warehouse system. In CDH 6 and earlier, this service includes Hive Metastore and HiveServer2. In Cloudera Runtime 7.0 and later, this service only includes the Hive Metastore; HiveServer2 and other components of the Hive execution engines are part of the Hive on Tez service.
☑	Hive on Tez	Hive on Tez is a SQL query engine using Apache Tez.
☑	Hue	Hue is the leading SQL Workbench for optimized, interactive query design and data exploration.
☑	Iceberg Replication	Iceberg Replication facilitates the replication of Iceberg tables across clusters.
☑	Impala	Apache Impala provides a real-time SQL query interface for data stored in HDFS and HBase. Impala requires the Hive service and shares the Hive Metastore with Hue.
☑	Kafka	Apache Kafka is publish-subscribe messaging rethought as a highly scalable distributed commit log.
☐	Key-Value Store Indexer	Key-Value Store Indexer listens for changes in data inside tables contained in HBase and indexes them using Solr.
☑	Knox	The Apache Knox Gateway is an Application Gateway for interacting with the REST APIs and UIs of Apache Hadoop deployments. This service requires Kerberos.
☐	Kudu	Apache Kudu is a data store that enables real-time analytics on fast changing data.
☐	Livy	Apache Livy is a REST service for deploying Spark applications.
☐	Livy for Spark 3	Apache Livy for Spark 3 is a REST service used for deploying Spark3 applications Before adding this service, ensure that you have installed the Spark3 binaries, which are not included in CDH.
☑	Ozone	Apache Ozone is a Scalable, S3 Compatible, Distributed object store for Big Data.
☑	Phoenix	Apache Phoenix is a scale-out relational database that supports OLTP workloads and provides secondary indexes, materialized views, star schema support, and common HBase optimizations. Phoenix uses Apache HBase as the underlying data store.
☐	Query Processor	Query Processor indexes Hive & Tez events and provides APIs to access them



Step 21. Select host assignment for different services in the ***Add cluster - configuration*** wizard. You need to assign hosts to different roles across all the selected services. Use the below table as a reference to assign the roles.

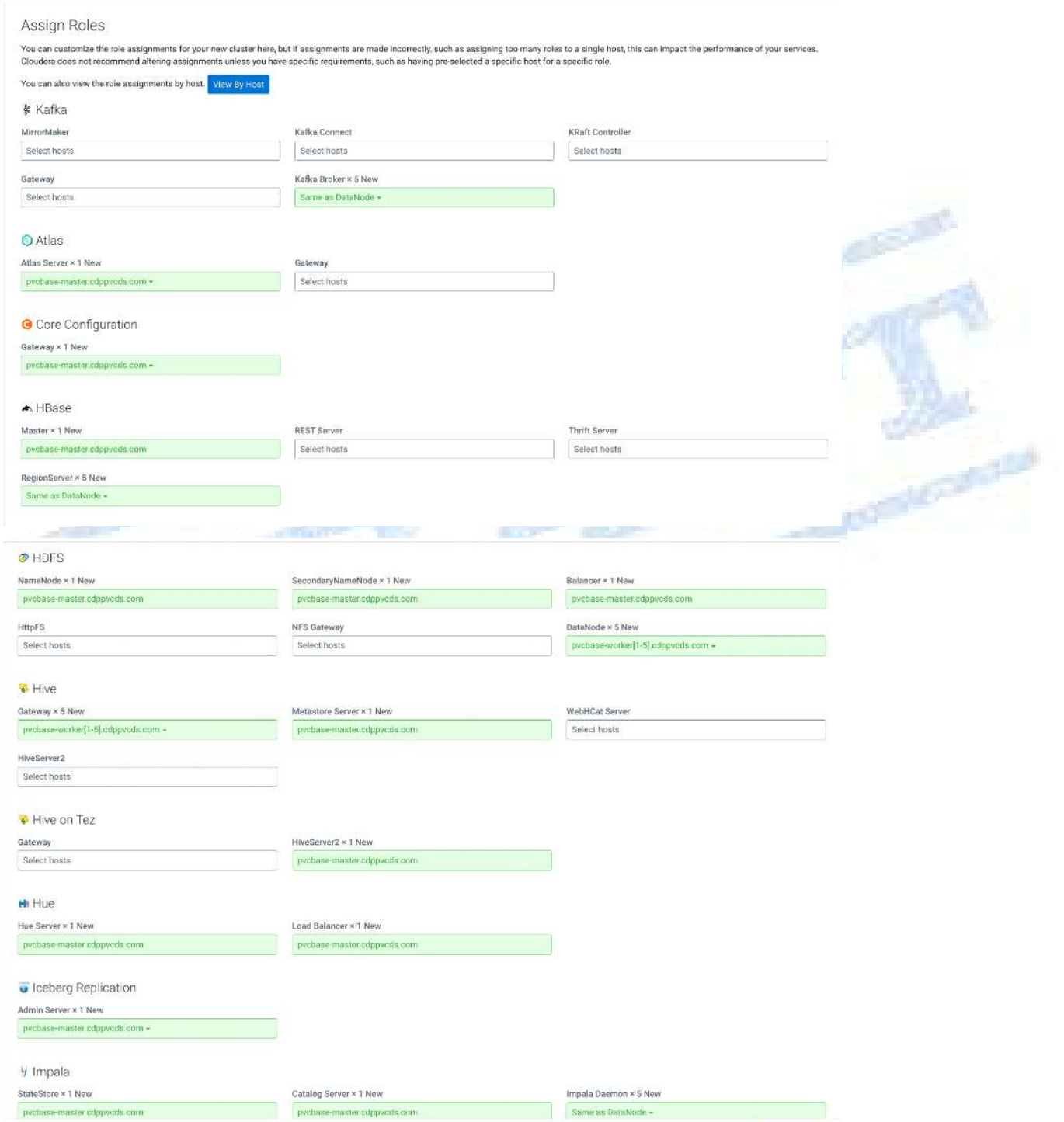
Table: Cloudera Data Platform Cloudera on premises Base host and Role assignment example

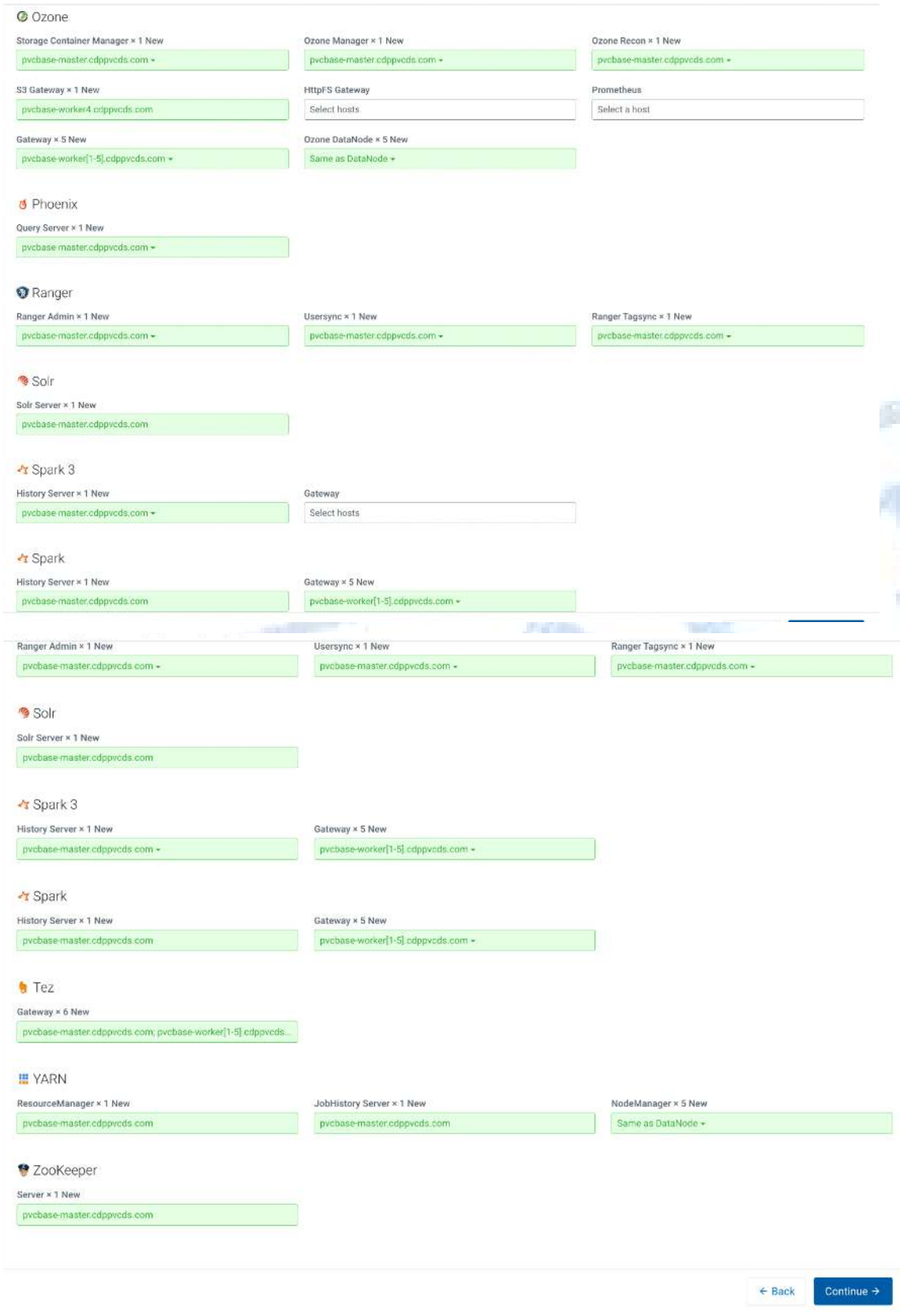
Service Name (Role Instance)	Host
HDFS	**NameNode** : pvcbase-master **SecondaryNameNode** : pvcbase-master **Balancer** : pvcbase-master **DataNode** : pvcbase-worker[1-5]
YARN	**ResourceManager** : pvcbase-master **NodeManager** : pvcbase-worker[1-5] **(Same as DataNode)** **JobHistoryServer** : pvcbase-master
Core Configuration	**Gateway** : pvcbase-master
Iceberg Replication	**Gateway** : pvcbase-master

Service Name (Role Instance)	Host
Knox	**Gateway** : pvcbase-master
~~Cloudera Management Service~~ This is already configured in previous steps	~~**Service Monitor :** cldr-mngr~~ ~~**Host Monitor :** cldr-mngr~~ ~~**Reports Manager :** cldr-mngr~~ ~~**Event Server :** cldr-mngr~~ ~~**Alert Publisher :** cldr-mngr~~
Spark2 (i.e. SparkOnYARN)/ Spark3	**Spark History Server** : pvcbase-master **Spark Gateway** : pvcbase-worker[1-5]
Hive	**Hive Metastore Server (HMS)** : pvcbase-master **Gateway** : pvcbase-worker[1-5]
Tez	**Gateway** : pvcbase-worker[1-5]
Hive on Tez	**HiveServer2** : pvcbase-master **Gateway** : pvcbase-worker[1-5]
Impala	**Impala Catalog Server** : pvcbase-master **Impala State Store** : pvcbase-master **Impala Daemon** : pvcbase-worker[1-5] **(Same as DataNode)**
HUE	**HUE Server** : pvcbase-master **LoadBalancer** : pvcbase-master
HBase	**HBase Master** : pvcbase-master **RegionServer** : pvcbase-worker[1-5] **(Same as DataNode)**
Phoenix	**Query Server** : pvcbase-master
Ozone	**Storage Container Manager** : pvcbase-master **Ozone Manager** : pvcbase-master **Ozone Recon** : pvcbase-master **S3 Gateway** : pvcbase-master **Gateway** : pvcbase-worker[1-5] **OzoneDataNode** : pvcbase-worker[1-5] **(Same as DataNode)**
CDP-INFRA-SOLR	**Solr Server** : pvcbase-master (can be installed on all hosts if needed if there is a search use case)
Kafka	**Kafka Broker** : base-master, pvcbaseworker[1-5] **(Same as DataNode)**
ZooKeeper **(must be >3)**	**Zookeeper Server** : pvcbase-master, pvcbase-worker[1-2]
Ranger	**Ranger Admin** : pvcbase-master **UserSync** : pvcbase-master **Ranger Tagsync** : pvcbase-master
Atlas	**Atlas Server** : pvcbase-master
Oozie Server (Optional)	pvcbase-master

Service Name (Role Instance)	Host
YARN Queue Manager (Optional)	

Step 22. Assign roles as updated above and shown as below.

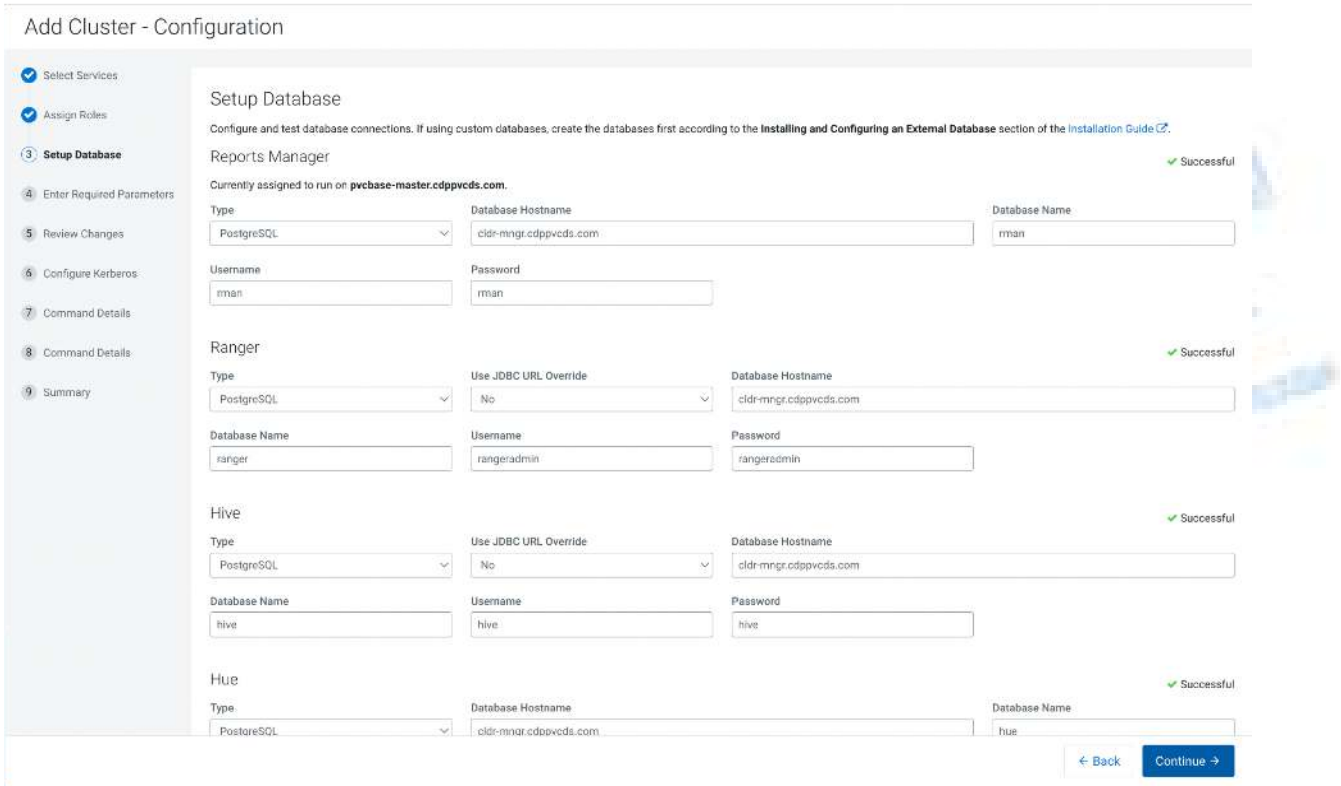




Step 23. Click ***Continue***. When you're doing the set-up for the first time on this CM Sever. ***Cloudera Management Services*** will be installed only once along with other control plane services.

Step 24. On the ***Setup Databases*** page, Select ***database type*** as ***Use Custom Database***. Provide database hostname, username, and password (created in earlier steps) for different services and click on ***Test Connection***. After a successful connection test, click ***Continue*** to install, configure and start services sequentially.

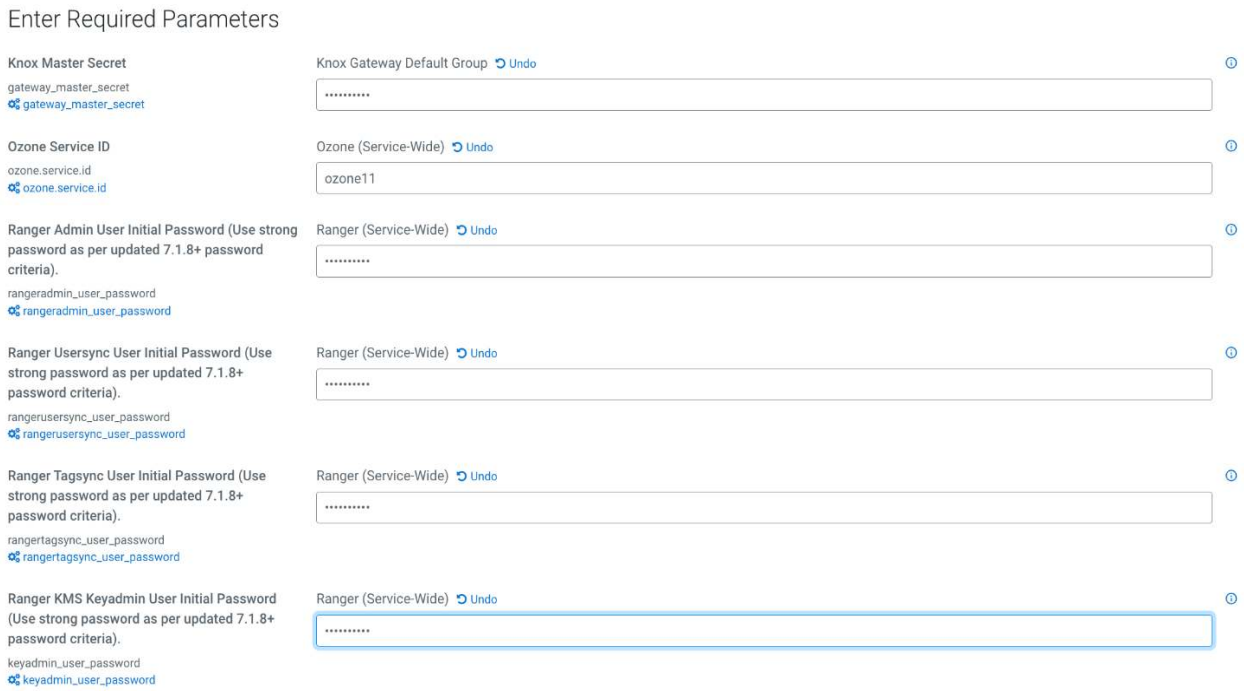
- Reports Manager (For Cloudera Management Service)
- Oozie Server (If selected to install, as optional)
- Ranger
- Hive
- YARN Queue Manager
- Hue



Step 25. Next, the ***Enter Required Parameters*** page appears.

Step 26. Wait at the parameter screen, enter the required parameters. For all the remaining parameters, set a common password so that it becomes easier while using those services. This password must have 1 lowercase, 1 uppercase, and 1 numeric value. Failing to adhere to this, the final step in the cluster setup would fail. Please see the required inputs below:

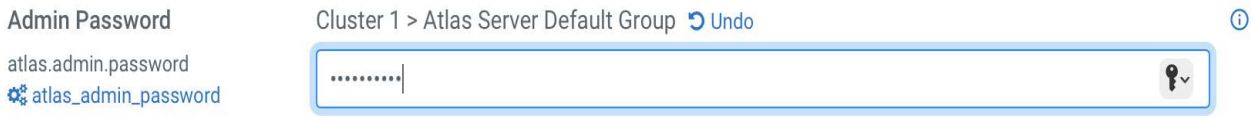
- *Knox Master Secret:* ***Cloudera@123***
- *Knox IDBroker Master Secret:* ***Cloudera@123***
- *Enter a suitable name for Ozone Service ID. i.e.:* ***ozone11***
- Ranger Admin: ***Cloudera@123***
- Ranger Usersync: ***Cloudera@123***
- Ranger Tagsync: ***Cloudera@123***
- Ranger KMS Keyadmin: ***Cloudera@123***



Step 27. Click ***Continue***.

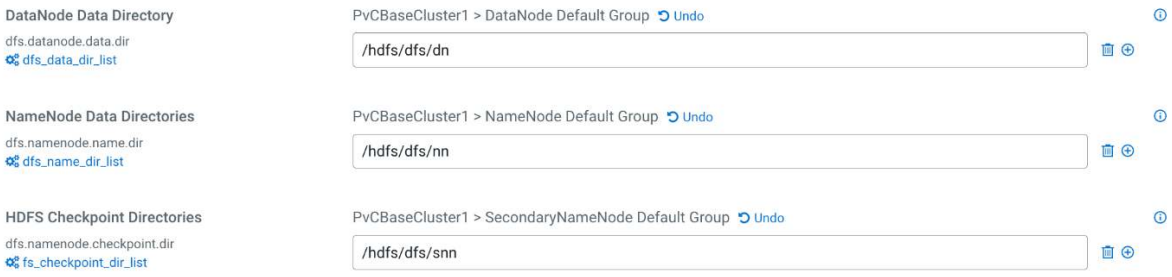
Step 28. The ***Review Changes*** page appears. Set a ***password*** for ***Atlas***. You can set it to the same value set for Ranger above.

- Atlas Admin password: ***Cloudera@123***



Step 29. Scroll down and verify/update the HDFS disks configuration according to the below.

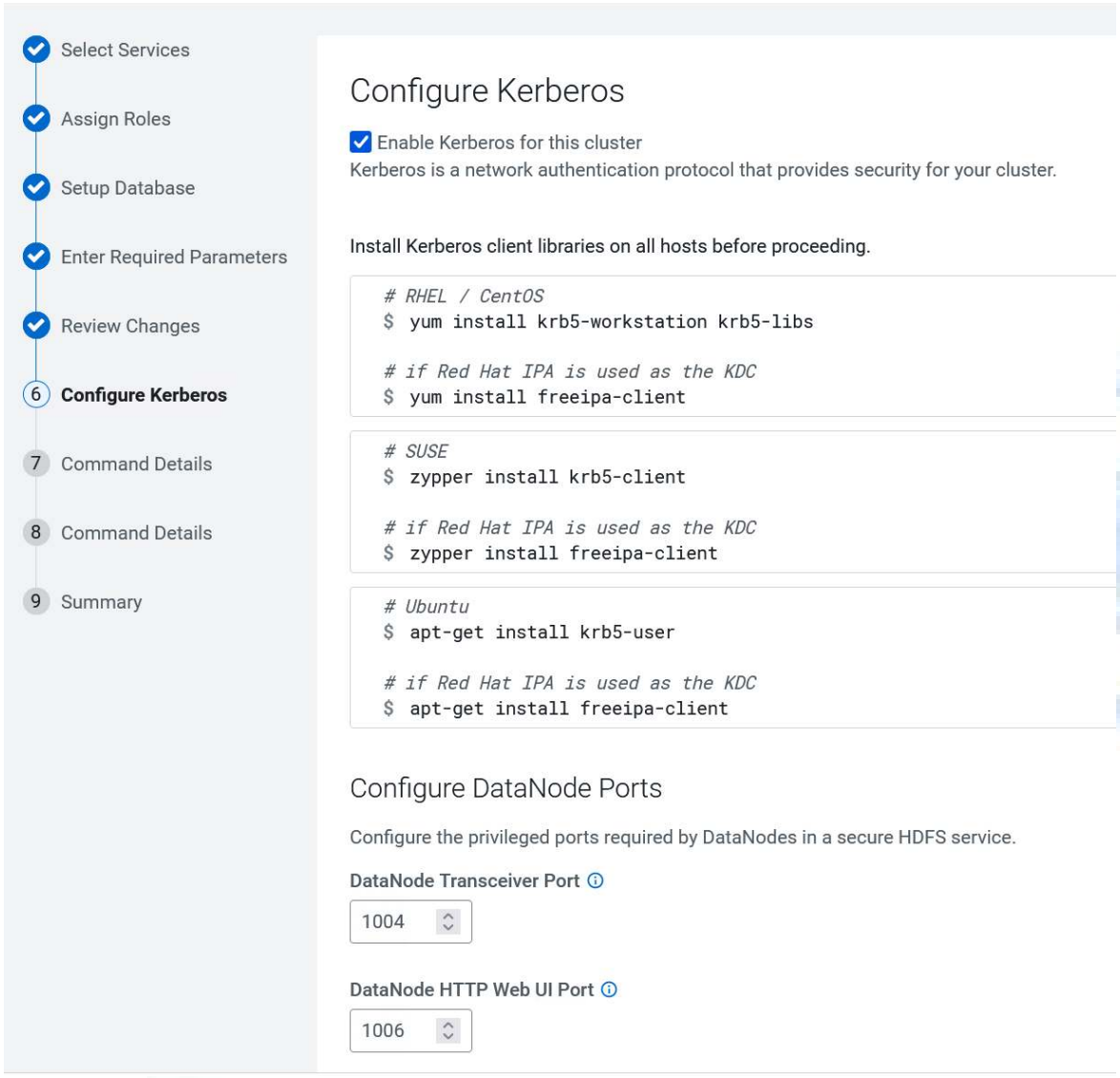
- DataNode Data Directory: ***/hdfs/dfs/dn***
- NameNode Data Directories: ***/hdfs/dfs/nn***
- HDFS Checkpoint Directories: ***/hdfs/dfs/snn***



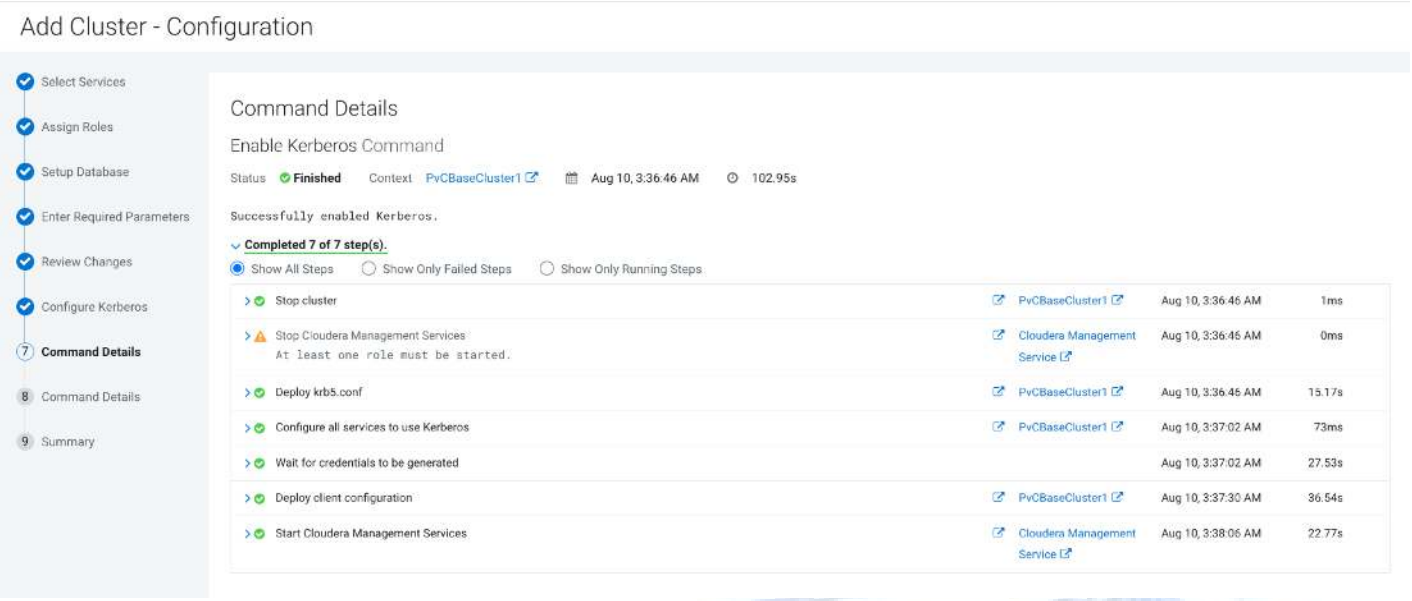
Step 30. Review the changes for all the services on the ***Review Changes*** page and verify/edit the configuration parameters as per your requirements. Click ***Continue***.

Step 31. A few sets of commands are running in the background. Wait for them to get executed successfully. Once done, Click ***Continue***.

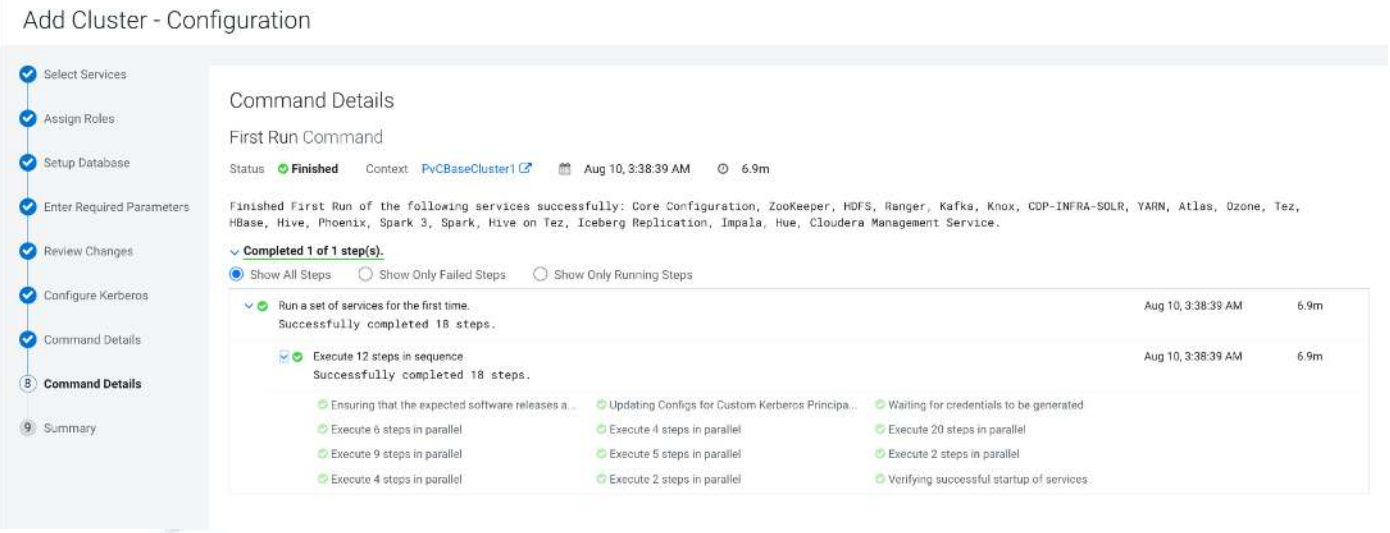
Step 32. Configure Kerberos and Keep Review and customize the configuration changes based on your requirements. Check the box for *Enable Kerberos for this cluster*. **Required libraries i.e. krb5-workstation, krb5-libs and freeipa-client are already installed on all servers in prior steps.**



Step 33. Click ***Continue*** after Cloudera Manager successfully runs the ***Enable Kerberos*** command.



Step 34. Installation wizard will run the first command to start cluster roles and services. Click ***Continue***.



Step 35. If you still face any issue while making the services up or during the installation or start of any Control Plane services please refer to troubleshooting PvC Base Cluster part at the end of this document. Though, some of the major issues during installation, their cause and their resolution is listed as below:

Zookeeper SASL error:
Solution:resolved by regenerate key tab

==========

Kafka error:
Solution:
Update clusterid and broker id from Role logs in meta.properties
rm -vf /var/local/kafka/data/meta.properties
rm -vf /tmp/kafka-logs/*
https://gautambangalore.medium.com/resolved-error-fatal-error-during-kafkaserver-startup-37f638c2c00c
ansible datanodes -m shell -a "sed -i 's#aHRUaTqoQzOpGf8qA5bXlQ#rxsV4DwNRvWtIcl5ejG-IA#g'

==========

Service NodeManager failed in state INITED
org.apache.hadoop.yarn.exceptions.YarnRuntimeException: Failed NodeManager login
at org.apache.hadoop.yarn.server.nodemanager.NodeManager.serviceInit(NodeManager.java:511)
at org.apache.hadoop.service.AbstractService.init(AbstractService.java:164)
at
org.apache.hadoop.yarn.server.nodemanager.NodeManager.initAndStartNodeManager(NodeManager.java:974)
at org.apache.hadoop.yarn.server.nodemanager.NodeManager.main(NodeManager.java:1054)
Caused by: org.apache.hadoop.security.KerberosAuthException: failure to login: for principal:
yarn/pvcbase-worker3.cldrsetup.local@CLDRSETUP.LOCAL from keytab yarn.keytab
javax.security.auth.login.LoginException: Client not found in Kerberos database (6) - CLIENT_NOT_FOUND
Solution:
Regenerate kerberos creds from administration>security for yarn

==========

YARN issue
Solution:
https://community.cloudera.com/t5/Support-Questions/Error-CM-Server-guid-updated-CDH-5-9-0/m-p/47221
https://community.cloudera.com/t5/Support-Questions/CDH-6-1-Installation-Issues-Unable-to-obtain-CM-release/td-p/88238
Fixed it by deleting /var/lib/cloudera-scm-agent/cm_guid on each node.

==========

Chrony issue on during ipaserver/ipaclient installation
Solution:
Stop chrnoyd and remove chrony from all hosts, then install ipa-server and then ipa-client. It will work.

==========

Rman db error
Exception while executing ddl scripts.
org.postgresql.util.PSQLException: ERROR: relation "rman_usergrouphistory_seq" already exists
at org.postgresql.core.v3.QueryExecutorImpl.receiveErrorResponse(QueryExecutorImpl.java:2725)
at org.postgresql.core.v3.QueryExecutorImpl.processResults(QueryExecutorImpl.java:2412)
Solution:
drop and recreate db solved issues.

==========

Ozone Error:
Found SOLR_SERVICE: ''
solution:
ozone install error
include solr in service dependency and restart services

==========

Issue: IPASERVER failed to resolve DNS ipaservices not working
Solution:
Port 53 was not open on AWS SecGrp: ipaserver was not working on aws due to it,
updated secgrp added 53 rule for dns fixed issue.

==========

Ranger Error:
Repo cm_kafka already exists ->
Solution:
delete cm_kafka from ranger

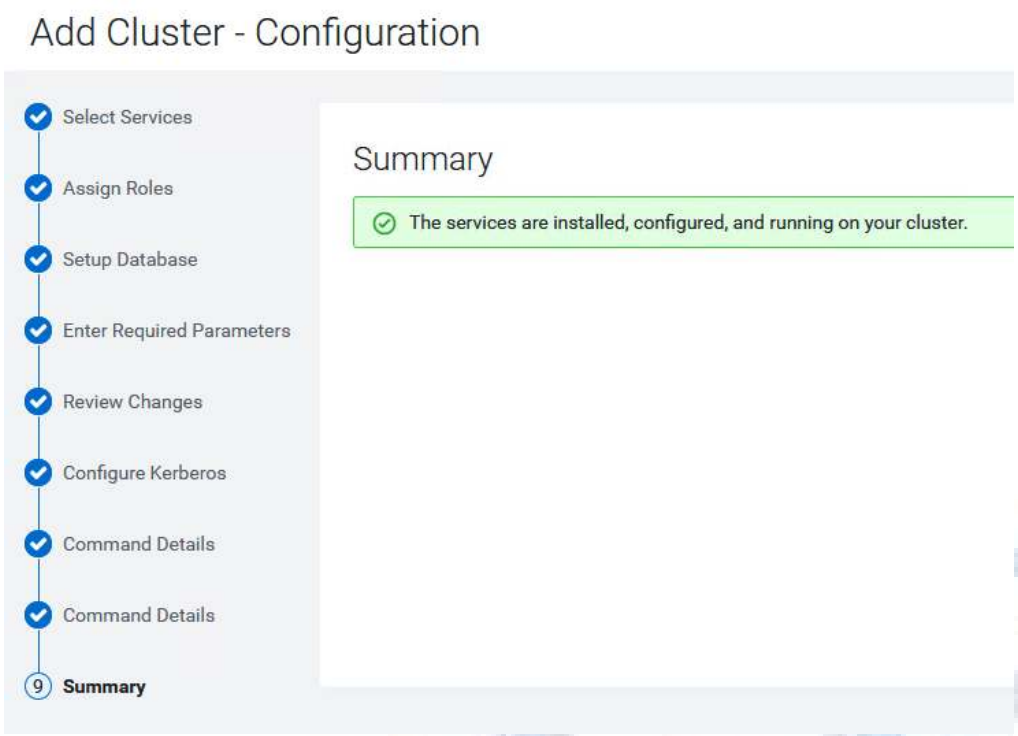
==========

CM Not able to login
2024-05-14 04:41:02,375 INFO CommandPusher-1:com.cloudera.server.cmf.CommandPusherThread: Acquired lease
lock on DbCommand:1546336335
2024-05-14 04:41:02,378 INFO CommandPusher-1:com.cloudera.cmf.service.AbstractOneOffHostCommand:
Unsuccessful 'RepMgrTestDatabaseConnection'
2024-05-14 04:41:02,379 INFO CommandPusher-1:com.cloudera.cmf.service.AbstractDbConnectionTestCommand:
Command exited with code: 1

```
2024-05-14 04:41:02,379 INFO CommandPusher-1:com.cloudera.cmf.service.AbstractDbConnectionTestCommand: +
MGMT_JAVA_OPTS='-Djava.net.preferIPv4Stack=true '
+ exec /usr/lib/jvm/java-17-openjdk-17.0.11.0.9-2.el9.x86_64/bin/java -Djava.net.preferIPv4Stack=true
-Djava.security.egd=file:///dev/urandom -cp
'/run/cloudera-scm-agent/process/1546336334-MGMT.REPORTSMANAGER-test-db-connection:/usr/share/java/mysql-c
onnector-java.jar:/usr/share/java/postgresql-connector-java.jar:/usr/share/java/oracle-connector-java.jar:
/opt/cloudera/cm/lib/*' com.cloudera.enterprise.dbutil.DbCommandExecutor db.properties
Exception in thread "main" java.lang.NoClassDefFoundError:
com/ongres/scram/common/stringprep/StringPreparation
        at org.postgresql.core.v3.ConnectionFactoryImpl.doAuthentication(ConnectionFactoryImpl.java:759)
        at org.postgresql.core.v3.ConnectionFactoryImpl.tryConnect(ConnectionFactoryImpl.java:161)
        at org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl.java:213)
        at org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:51)
        at org.postgresql.jdbc.PgConnection.<init>(PgConnection.java:225)
        at org.postgresql.Driver.makeConnection(Driver.java:465)
        at org.postgresql.Driver.connect(Driver.java:264)
        at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:681)
        at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:229)
        at com.cloudera.enterprise.dbutil.DbCommandExecutor.testDbConnection(DbCommandExecutor.java:265)
        at com.cloudera.enterprise.dbutil.DbCommandExecutor.main(DbCommandExecutor.java:140)
Caused by: java.lang.ClassNotFoundException: com.ongres.scram.common.stringprep.StringPreparation
        at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:641)
        at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:188)
        at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:525)
        ... 11 more
2024-05-14 04:41:02,379 ERROR CommandPusher-1:com.cloudera.cmf.model.DbCommand: Command
1546336335(RepMgrTestDatabaseConnection) has completed. finalstate:FINISHED, success:false, msg:Unexpected
error. Unable to verify database connection.
Caused by: java.lang.ClassNotFoundException: com.ongres.scram.common.stringprep.StringPreparation
        at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:641)
        at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:188)
        at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:525)
        ... 11 more
2024-05-14 04:50:38,519 ERROR CommandPusher-1:com.cloudera.cmf.model.DbCommand: Command
1546336734(OozieTestDatabaseConnection) has completed. finalstate:FINISHED, success:false, msg:Unexpected
error. Unable to verify database connection.
2024-05-14 04:50:38,519 INFO CommandPusher-1:com.cloudera.cmf.command.components.CommandStorage: Invoked
delete temp files for command:DbCommand{id=1546336734, name=OozieTestDatabaseConnection,
host=pvcbase-master.cldrsetup.local} at dir:/var/lib/cloudera-scm-server/temp/commands/1546336734
Solution:
Caused after change in hostssl parameter for Postgres (suspected)
Delete scm db and recreated db and restart scm server fixed the issue. This lead to reinstall entire base
and ecs clusters as metadata deleted from scm db

================================================================================================================

Other:
Configure Ozone with other data services before env creation, else it will lead to CDE installation error
Configure thrift server role in hbase for hue
Knox and Atlas works with local Linux Users and Password credentials i.e. PAM
For accessing the WebUI and fixing issues for web based authentication is not working for some of the
services including Knox, Atlas, HDFS (Namenode UI), Yarn (History server UI) etc.Disable Kerberos
Authentication for WebUI under each service configuration section which are showing 403 or 401 error.
In case of Cleanup and re-installation (end-to-end), make sure cleanup steps are performed properly and no
control plane services left user and groups created in /etc/passwd and /etc/group on all nodes of the cluster
include cldr-mngr.
```

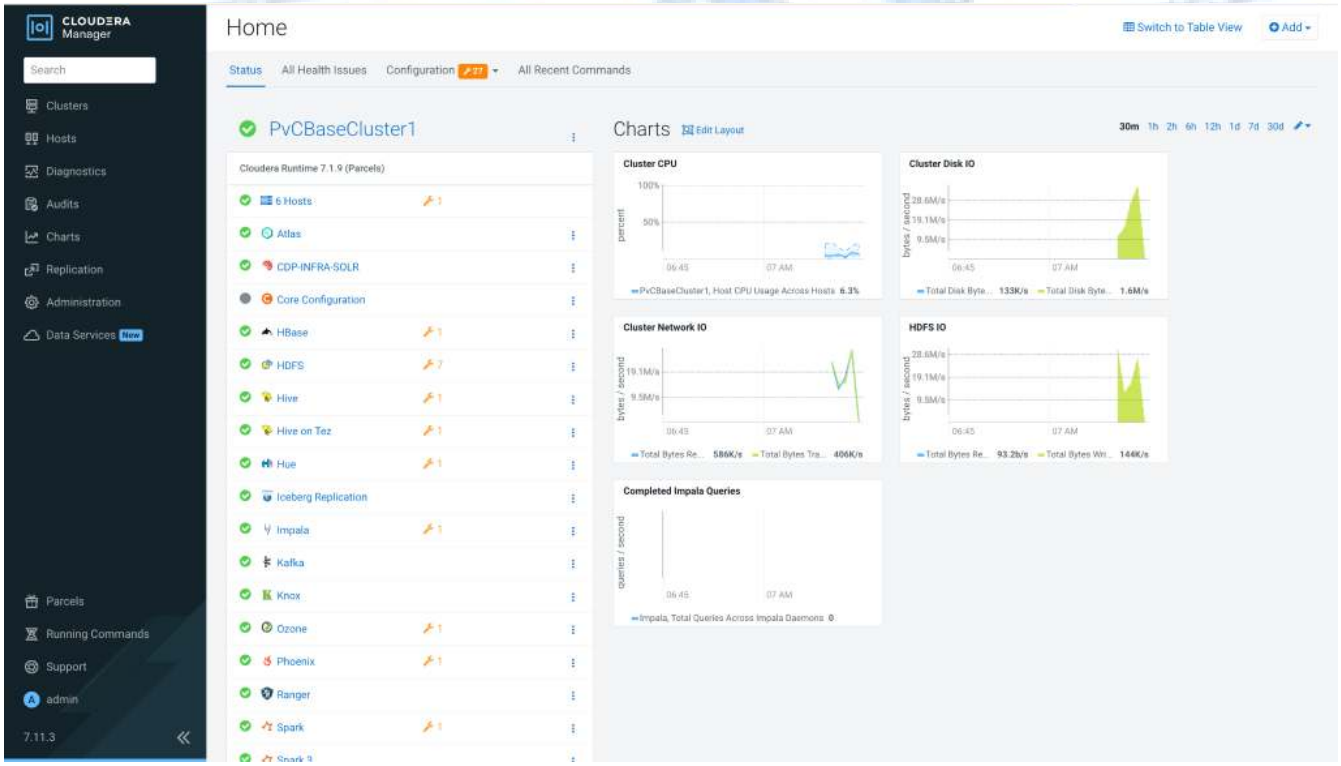
Step 36. Next, all the Cloudera services would get started and their prerequisite operations would also be run. These processes will run in a combination of serial and parallel processes. Wait for them to complete. Once completed, you will see a Green tick next to all the steps, as shown in the screenshot above.

Step 37. Once completed above step, click ***Continue***. You will see a summary page like below.



Step 38. Click ***Finish*** on the Summary page.

Step 39. This will navigate to the main page where you can see all the services installed.



Step 40. All the services should be in Healthy state. If there are any instances in Bad Health, troubleshoot the same and fix it.

Step 41. This completes the *Cloudera on premises Base cluster* setup.

Note: You might need to adjust configuration parameters of the cluster after successful first run command execution. Apply the changes and restart the cluster.

Step 42. We will perform the adjustment of the configuration parameters of the cluster to fix some of the issues with the services on the base cluster, in the next steps below.

Step 43. Optionally, we can also perform the prerequisites and compatibility tests for Hardware using the script provided below by Cloudera.

https://github.com/cloudera-labs/toolkits/tree/main/data_services-toolkit/DS_Pre-Install_Check

****

Additional requirements and details for Cloudera on premises Base Cluster services:

Note: Common Data Lake Services' URLs:

CM-UI:
HTTP: http://cldr-mngr.cldrsetup.local:7180/cmf/
HTTPS: https://cldr-mngr.cldrsetup.local:7183/cmf/

HDFS:
HDFS-NAMENODE UI: https://pvcbase-master.cldrsetup.local:9871/dfshealth.html

YARN:
HDFS-YARN JobHistory UI: https://pvcbase-master.cldrsetup.local:19890/jobhistory
YARN RM UI: https://pvcbase-master.cldrsetup.local:8090/ui2/#/cluster-overview

Ranger: https://pvcbase-master.cldrsetup.local:6182/index.html#/policymanager/resource

Atlas: https://pvcbase-master.cldrsetup.local:31443/login.jsp
(Login works with PAM-Linux Server Local User Credentials)

Knox:
https://pvcbase-master.cldrsetup.local:8443/gateway/knoxsso/knoxauth/login.html?originalUrl=https://pvcbase-master.cldrsetup.local:8443/gateway/homepage/home/?profile=token
(Login works with PAM-Linux Server Local User Credentials)

HiveServer2 UI: https://pvcbase-master.cldrsetup.local:10002/

HUE: https://pvcbase-master.cldrsetup.local:8889/hue/editor/?type=hive

HBASE: https://pvcbase-master.cldrsetup.local:16010/master-status

Ozone:
Ozone Recon: https://pvcbase-master.cldrsetup.local:9889/#/Overview
Ozone SCM: https://pvcbase-master.cldrsetup.local:9877/#!/
Ozone Manager: https://pvcbase-master.cldrsetup.local:
S3 Gateway: https://pvcbase-master.cldrsetup.local:
Gateway: https://pvcbase-workter1.cldrsetup.local:
OzoneDataNode: https://pvcbase-worker1.cldrsetup.local:

Spark JobHistory Server:
Spark2: https://pvcbase-master.cldrsetup.local:18488/
Spark3: https://pvcbase-master.cldrsetup.local:18489/

Impala:
Impala Catalog: https://pvcbase-master.cldrsetup.local:25020/
Impala Statestore: https://pvcbase-master.cldrsetup.local:25010/

Job History Server: https://pvcbase-master.cldrsetup.local:9991/

Step 1. Port requirements for different services on PvC Base Cluster/Data Services (ECS) Cluster:

Please whitelist the below ports or make sure , firewall is disabled in the internal network. (Not required in on-premise Private datacenter based network)

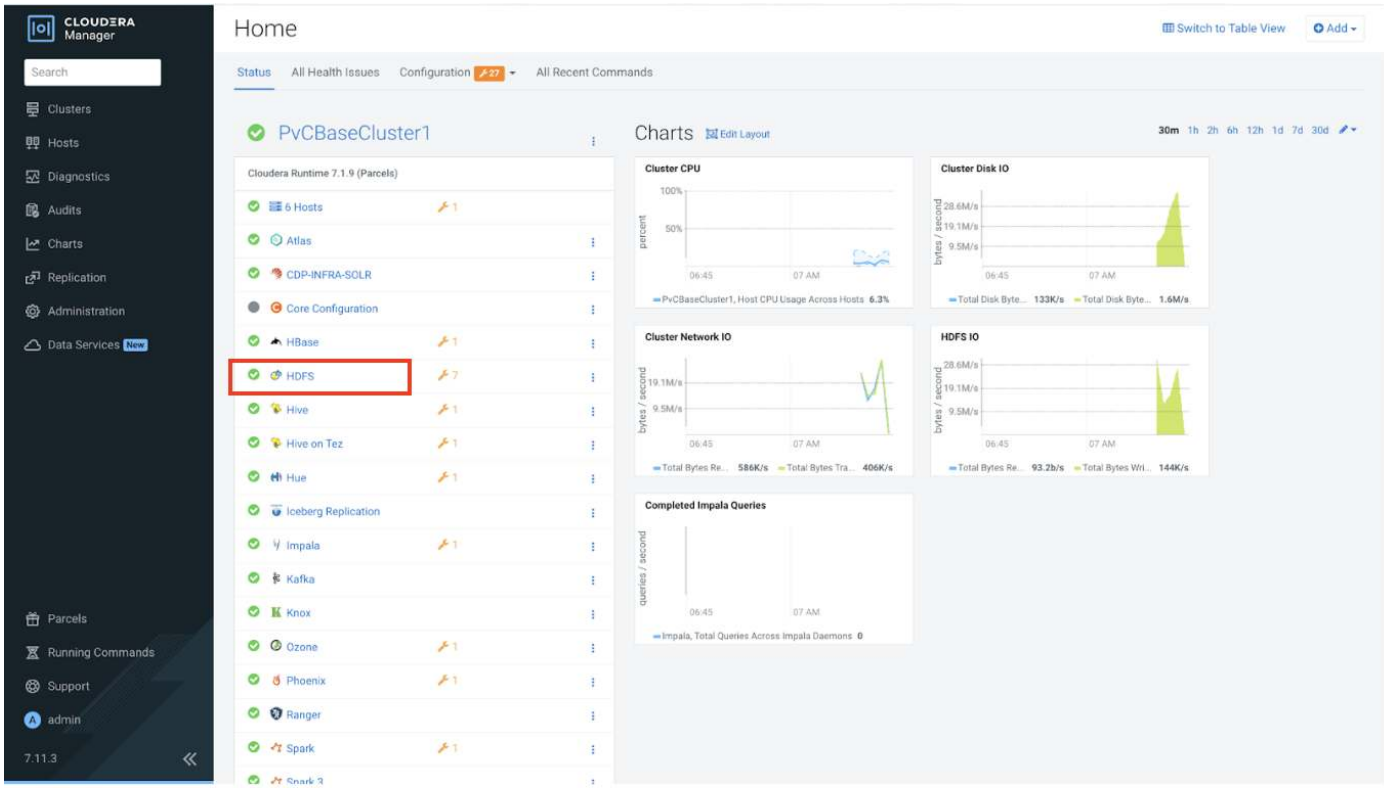
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-ports-used-by-runtime.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-ports-third-party-components.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-ports-used-by-cm.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-service-dependencies.html
https://docs.cloudera.com/cdp-private-cloud-data-services/latest/installation-ecs/topics/cdppvc-installation-ecs-steps.html

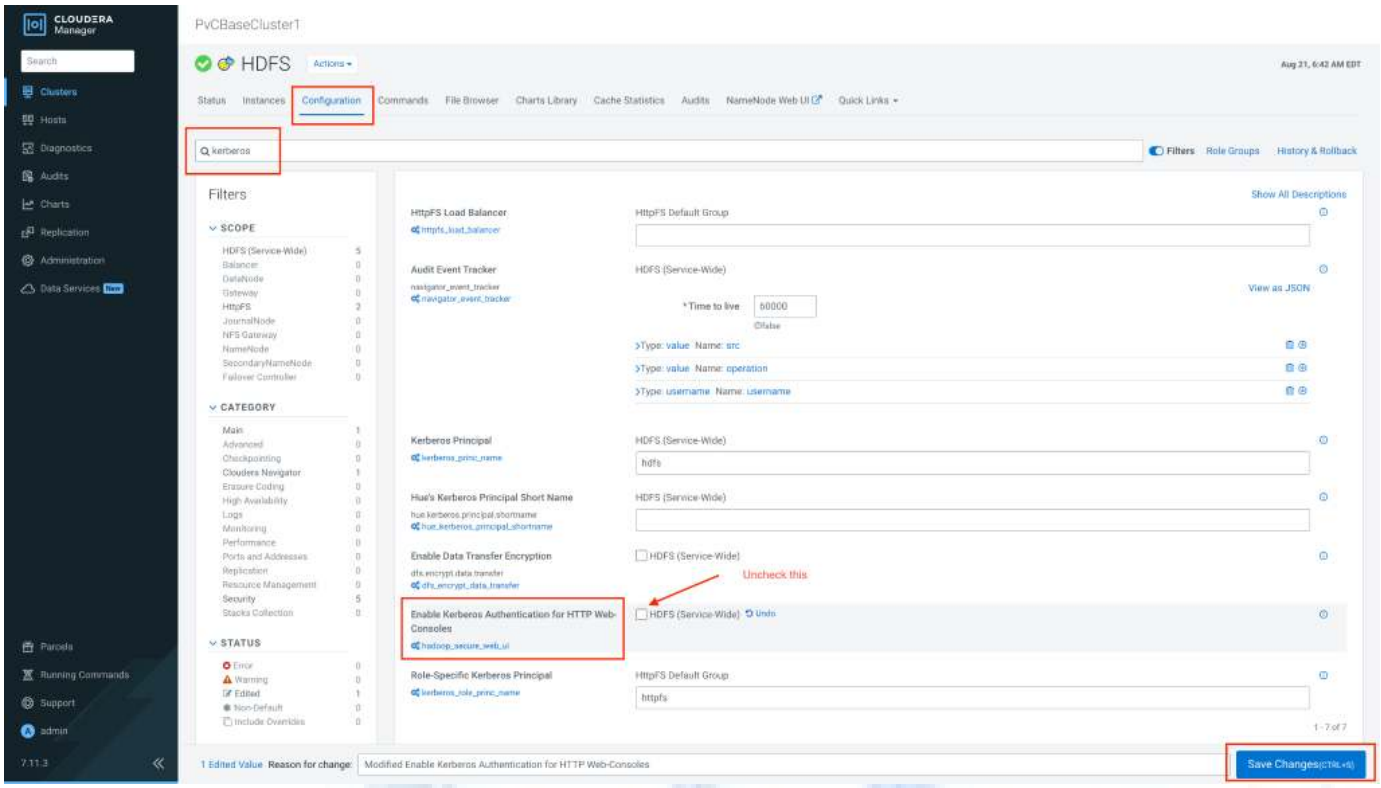
Step 2. Disable Kerberized Web-UI for YARN, Spark, HBase, HDFS, etc. from configuration:

While accessing Web UIs from web browsers, for HDFS-Namenode, YARN services, HiveServer2, Impala Services, etc. If gets 401 unauthorized error: We need to enable non-kerberized webUI for those services, for this we need to:

- Login to CM-UI> Go to individual services> Go to the Configuration section for individual services> Search for ***Kerberos***.
- Disable the option, by unchecking the checkbox, for Kerberos authentication for WebUI. Save changes. Restart the Stale Services to update the backend configuration files.

Example Screenshots for HDFS. You can follow the same for other services, which requires WebUI access.





1. For Impala, Hive, Hive on Tez edit value for
Ranger Plugin URL Auth Filesystem Schemes - file:,wasb:,adl:

2. Disable ~~Enable~~ Kerberos Authentication for HTTP Web-Consoles - HBase (Service-Wide), YARN, Spark2, Spark3, HiveServer2, Impala, HDFS, etc. ~~Click on Generate missing credentials for Kerberos.~~

3. For TLS/SSL enabled HDFS configuration you might see a warning as *"DataNode configuration is valid, but not recommended. There are two recommended configurations:*
(1) DataNode Transceiver Port and Secure DataNode Web UI Port (TLS/SSL) both >= 1024, DataNode Data Transfer Protection set, Hadoop TLS/SSL enabled;
(2) DataNode Transceiver Port and DataNode HTTP Web UI Port both < 1024, DataNode Data Transfer Protection not set, Hadoop TLS/SSL disabled."

DataNode Transceiver Port (dfs_datanode_port)- 9866
DataNode HTTP Web UI Port (dfs.datanode.http.address) - 9864
DataNode Data Transfer Protection (dfs.data.transfer.protection) - Authentication

4. Install Ranger Plugin for services, add service dependency i.e. Hive etc. and restart the cluster.
Ensure that the Ranger Solr and Ranger HDFS plugins are enabled. See Additional Steps for Apache Ranger for more details on Configuration Steps.

5. Make Sure HDFS, Ozone services are Installed and Running Successfully (critical services - if not working properly, then no other service will work properly)
1.HDFS 2.Zookeeper 3. …….

6. Atlas and Knox work with PAM authentication i.e. Local (Non-LDAP) users created on the base-master node where your Atlas server is running, unless Atlas is explicitly configured (integrated) to use LDAP. So you may need to create a local user on the base-master node, if it does not already exist.

7. Atlas is having dependencies on some additional services i.e. HBase, SOLR and Kafka

Step 3. Optionally, Update the /etc/hosts file on your working machine/JumpHost where you are trying to access your CM-UI to work with the URLs smoothly:

Open ***C:\Windows\System32\drivers\etc*** (on Windows) or ***/etc/hosts*** (on MAC/Linux), with sudo privileges.

```
ksahu@Kuldeeps-MacBook-Air ~ % sudo vi /etc/hosts
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost

52.221.202.246 pvcecs-master.cldrsetup.local
52.221.202.246 hue-kd-hive-vw1.apps.cldrsetup.local
52.221.202.246 cml-task-bo6klv.kuldeep-cml.apps.cldrsetup.local
52.221.202.246 kuldeep-cml.apps.cldrsetup.local
18.139.222.78 pvcbase-master.cldrsetup.local
13.251.65.11 cldr-mngr.cldrsetup.local

# ECS Links
52.221.202.246 console-cdp.apps.cldrsetup.local prometheus-cp.apps.cldrsetup.local
infra-prometheus.apps.cldrsetup.local validation-cdp.apps.cldrsetup.local
kube-dashboard.apps.cldrsetup.local longhorn.apps.cldrsetup.local fluent-console-cdp.apps.cldrsetup.local
vault.localhost.localdomain

# PvC Base Cluster Nodes
18.139.222.78 pvcbase-master.cldrsetup.local pvcbase-master
13.215.202.164 pvcbase-worker1.cldrsetup.local pvcbase-worker1
172.31.23.0   pvcbase-worker2.cldrsetup.local pvcbase-worker2
18.141.13.157 pvcbase-worker3.cldrsetup.local pvcbase-worker3

# PvC Data Services Cluster Nodes
172.31.30.239 pvcecs-master.cldrsetup.local pvcecs-master
172.31.22.43  pvcecs-worker1.cldrsetup.local pvcecs-worker1
172.31.30.249 pvcecs-worker2.cldrsetup.local pvcecs-worker2
172.31.26.24  pvcecs-worker3.cldrsetup.local pvcecs-worker3
172.31.24.198 pvcecs-worker4.cldrsetup.local pvcecs-worker4
172.31.24.53  pvcecs-worker5.cldrsetup.local pvcecs-worker5
```

Step 4. Optionally, Update the /etc/hosts file on your working machine/JumpHost (The /etc/hosts entries required for ECS Data Services)
https://docs.cloudera.com/management-console/1.5.5/private-cloud-security-overview/mc-private-cloud-security-overview.pdf

```
Embedded Container Service (ECS) :
• console-cdp.apps.APPDOMAIN
• prometheus-cp.apps.APPDOMAIN
• infra-prometheus.apps.APPDOMAIN
• validation-cdp.apps.APPDOMAIN
• kube-dashboard.apps.APPDOMAIN
• longhorn.apps.APPDOMAIN
• fluent-console-cdp.apps.APPDOMAIN

Entries required by CDW
Let APPDOMAIN be the base app domain for the ECS cluster. For example, if your console URL is
"console-cdp.apps.cldrsetup.local", then the APPDOMAIN is "cldrsetup.local". Let VWHNAME be the name of
the CDW Virtual Warehouse. This must match the name the user provides when creating a new Virtual
Warehouse (VW).

Endpoints of Hive VW:
• hue-VWHNAME.apps.APPDOMAIN
• hs2-VWHNAME.apps.APPDOMAIN

Endpoints of Impala VW:
• hue-VWHNAME.apps.APPDOMAIN
• coordinator-VWHNAME.apps.APPDOMAIN
• admissiond-web-VWHNAME.apps.APPDOMAIN
• catalogd-web-VWHNAME.apps.APPDOMAIN
• coordinator-web-VWHNAME.apps.APPDOMAIN
• statestored-web-VWHNAME.apps.APPDOMAIN
• impala-proxy-VWHNAME.apps.APPDOMAIN
```

- impala-autoscaler-web-**VWHNAME**.apps.**APPDOMAIN**

Endpoints of Viz:
- viz-VWHNAME.apps.APPDOMAIN

****

Configure Ranger with SSL/TLS enabled PostgreSQL Database

Login to Cloudera Manager Web Console. Go to ***Ranger > Configuration.***

Note: Make sure that:

- The database and database user for Ranger service are created in the required postgreSQL.
- A database server certificate is issued by a trusted certificate authority.
- The server host name matches the host name in the database server certificate.

From CDPDC-7.1.5 onwards, Ranger service requires postgres JDBC driver ***version >= 42.2.5***. The Ranger code also constructs the JDBC connection string to have ***sslmode=verify-full***, if Ranger Database SSL configurations are set in case of postgresql database type.

For more details:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-enable-ssl-tls-ranger-postgres-db.html

Copy the database server certificate to ***/var/lib/ranger/*** path, or use any custom path.

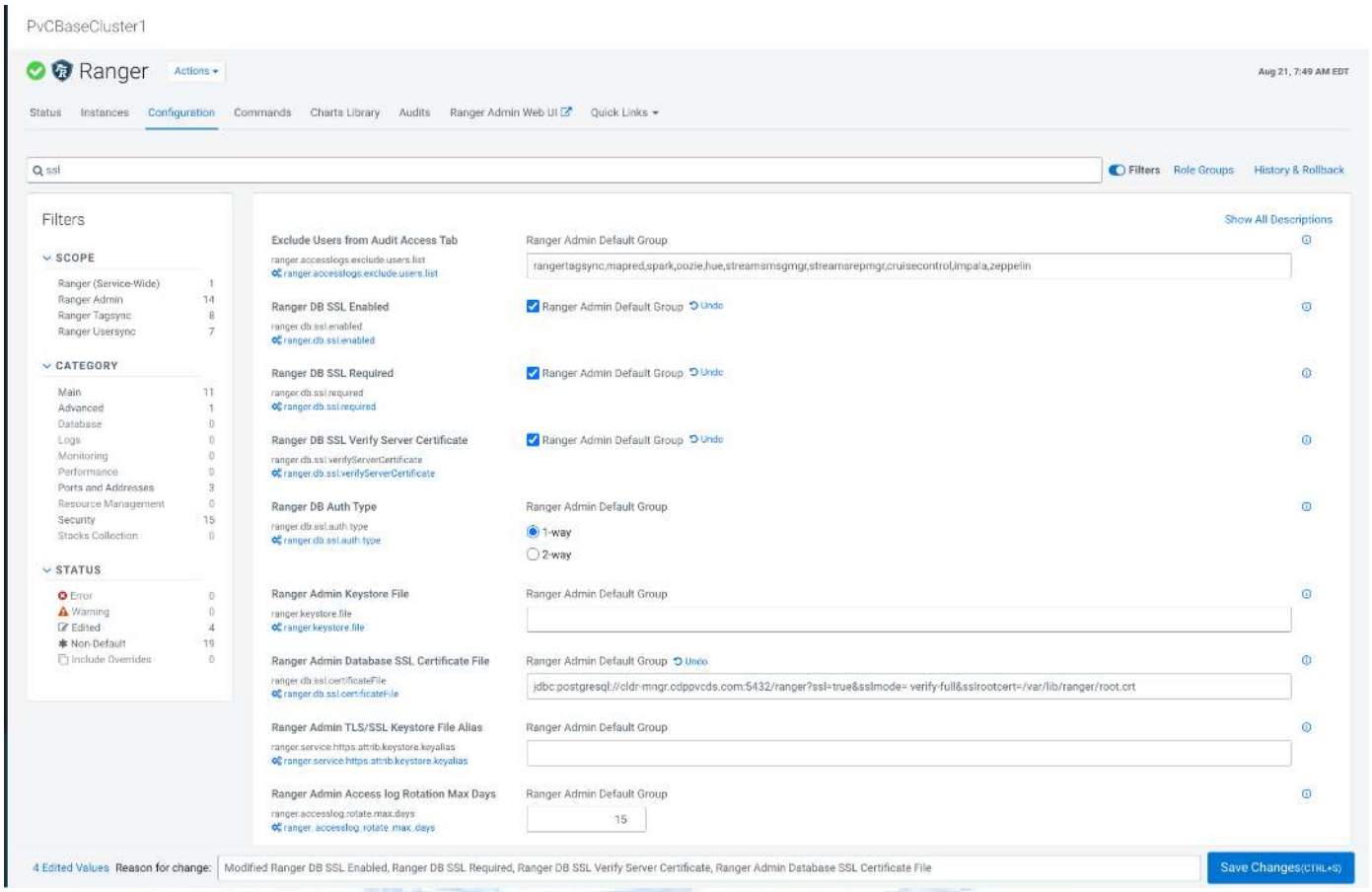
```
[root@pvcbase-master ~]# cp -rv /root/.postgresql/root.crt /var/lib/ranger/root.crt
```

- **In Review Config, search for SSL and update the following configurations:**

Component	Value
Ranger DB SSL Enabled: (ranger.db.ssl.enabled)	true (Checked)
Ranger DB SSL Required: (ranger.db.ssl.required)	true (Checked)
Ranger DB SSL Verify Server Certificate: (ranger.db.ssl.verifyServerCertificate)	true (Checked)
Ranger DB Auth Type: (ranger.db.ssl.auth.type)	1-way
Ranger Admin Database SSL Certificate File: (ranger.db.ssl.certificateFile)	<path-to-db-server-certificate>: ***/var/lib/ranger/root.crt*** or custom path ***/var/lib/cloudera-scm-server/.postgresql/root.crt***
Ranger Database JDBC URL Override:	jdbc:postgresql://<db_host>:<db_port>/<db_name>?sslmode=verify-full& sslrootcert=<server_certificate_path> ***jdbc:postgresql://cldr-mngr.cldrsetup.local:5432/ranger?ssl=true&sslmode=verify-full& cert=/var/lib/ranger/root.crt***
Set Load Balancer Address (Optional)	http://<ranger_host>:6080 http://pvcbase-master.cldrsetup.local:6080 https://<ranger_host>:6182 https://pvcbase-master.cldrsetup.local:6182

- After updating the configurations, click on ***Save Changes.***
- Ranger Service restart is required for rangeradmin after updating Ranger configuration.
- Click on ***Actions -> Restart*** under Ranger Service.
- Run below command on pvcbase-master node to check the Ranger logs, during restart.

```
[root@pvcbase-master ~]# tail -f /var/log/ranger/admin/catalina.out
```



****

Configure Hive metastore with SSL/TLS enabled PostgreSQL Database ***(Mandatory Step for CDW)***

In the Cloudera Manager Web console; go to ***Hive > Configuration > Hive Metastore Database JDBC URL Override***.

Copy the database server certificate to ***/var/lib/hive/*** path, or use any custom path.

```
[root@pvcbase-master ~]# cp -rv /root/.postgresql/root.crt /var/lib/hive/root.crt
```

Edit value as:

jdbc:postgresql://<db_host>:<db_port>/<db_name>?sslmode=verify-full&sslrootcert=<server_certificate_path>

jdbc:postgresql://cldr-mngr.cldrsetup.local:5432/hive?ssl=true&sslmode=verify-full&sslrootcert=/var/lib/hive/root.crt

Note: Restart required for Hive Metastore Server and HiveServer2 after updating Hive configuration.

Note: Click on ***Actions -> Restart*** under ***Hive*** and ***Hive-on-Tez*** Services.

****

Scale the Cluster (Optional– Skip this step)

The role assignment recommendation above is for clusters with at least 64 servers and in High Availability. For smaller clusters running without High Availability the recommendation is to dedicate one server for Name Node and a second server for secondary name node and YARN Resource Manager. For larger clusters larger than 16 nodes the recommendation is to dedicate one server each for name node, YARN Resource Manager and one more for running both Name Node (High Availability) and Resource Manager (High Availability) as in the table (no Secondary Name Node when in High Availability).

Note: For production clusters, it is recommended to set up Name Node and Resource manager in High Availability mode.

This implies that there will be at least 3 master nodes, running the Name Node, YARN Resource manager, the failover counterpart being designated to run on another node and a third node that would have similar capacity as the other two nodes.

All the three nodes will also need to run zookeeper and quorum journal node services. It is also recommended to have a minimum of 8 Data Nodes in a cluster. Please refer to the next section for details on how to enable HA.

Enable High Availability (Optional– Skip this step)

Note: Setting up High Availability is done after the Cloudera Installation is completed.
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/managing-clusters/topics/cm-high-availability.html

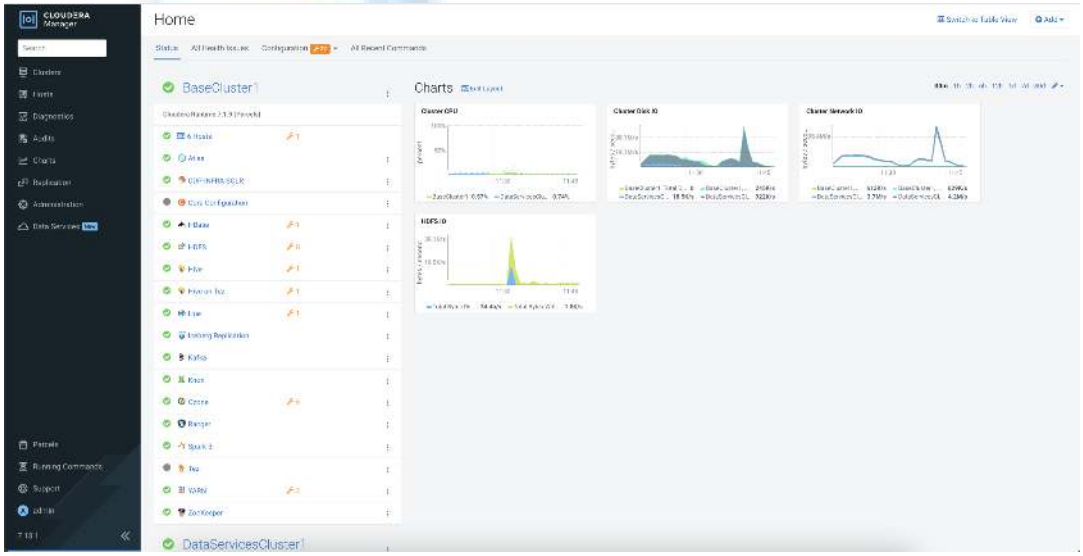
Configure Browsers for Kerberos Authentication

Note: To enable specific web browsers to use SPNEGO to negotiate Kerberos authentication, please visit:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/security-how-to-guides/topics/cm-security-browser-access-kerberos-protected-url.html

Cloudera on premises Base checklist

Cloudera support matrix lists the supported software for the Cloudera on premises Base cluster and the Cloudera on premises Data Services containerized cluster.

Please review Cloudera on premises Base Checklist:
https://docs.cloudera.com/cdp-private-cloud-data-services/latest/installation/topics/cdppvc-installation-pvcbase-checklist.html



****

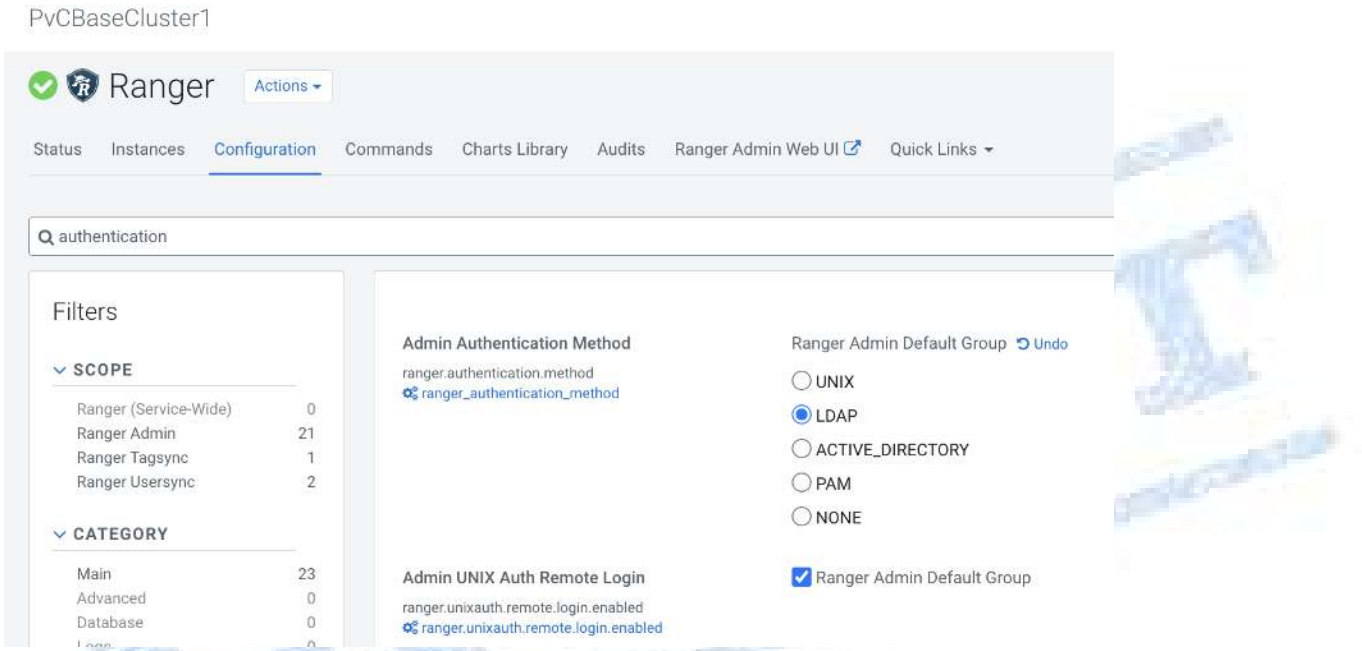
Configure Ranger authentication for LDAP (Optional– Skip this Step)

Follow steps below to configure Ranger for LDAP authentication.

1. In Cloudera Manager, select ***Ranger***, then click the ***Configuration*** tab.

2. To display the authentication settings, type "***authentication***" in the Search box. Scroll down to see all of the ***LDAP*** settings.

3. Select LDAP for "Admin Authentication Method".



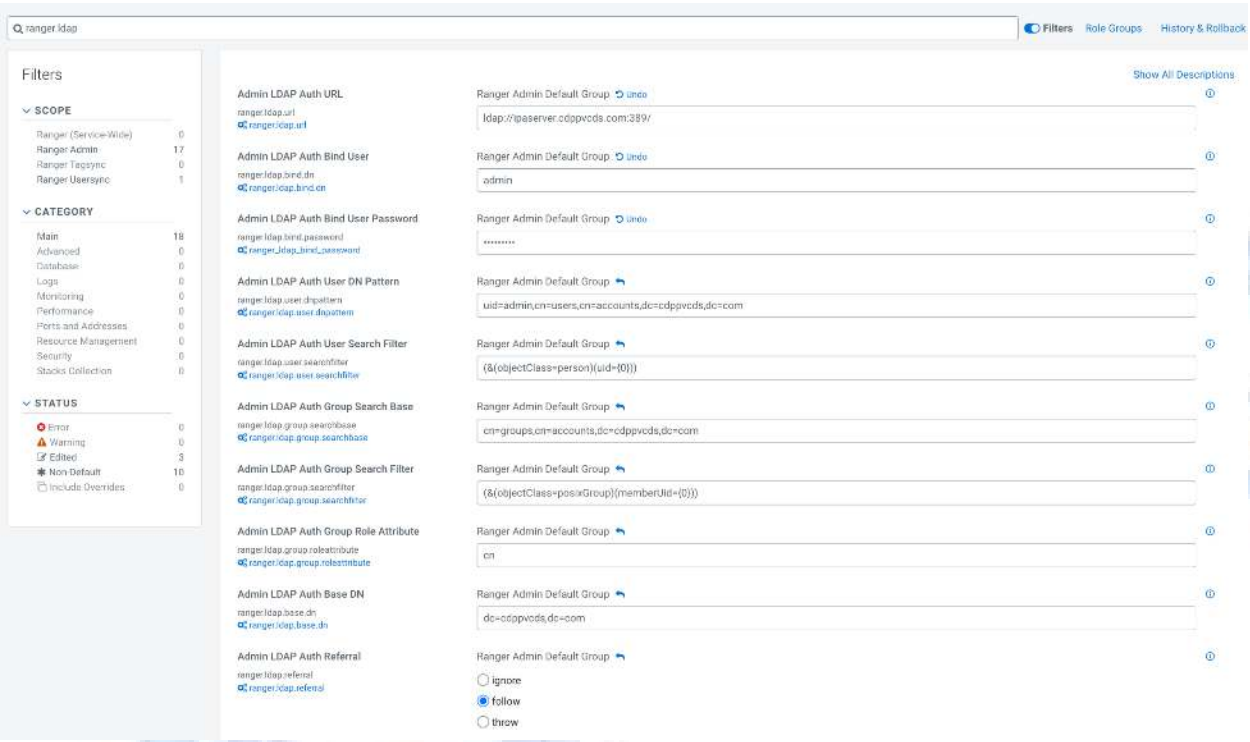
4. Configure the following settings for LDAP authentication as shown below, the details depends on your configuration, based on existing LDAP/AD setup:

Table 8. User LDAP Integration

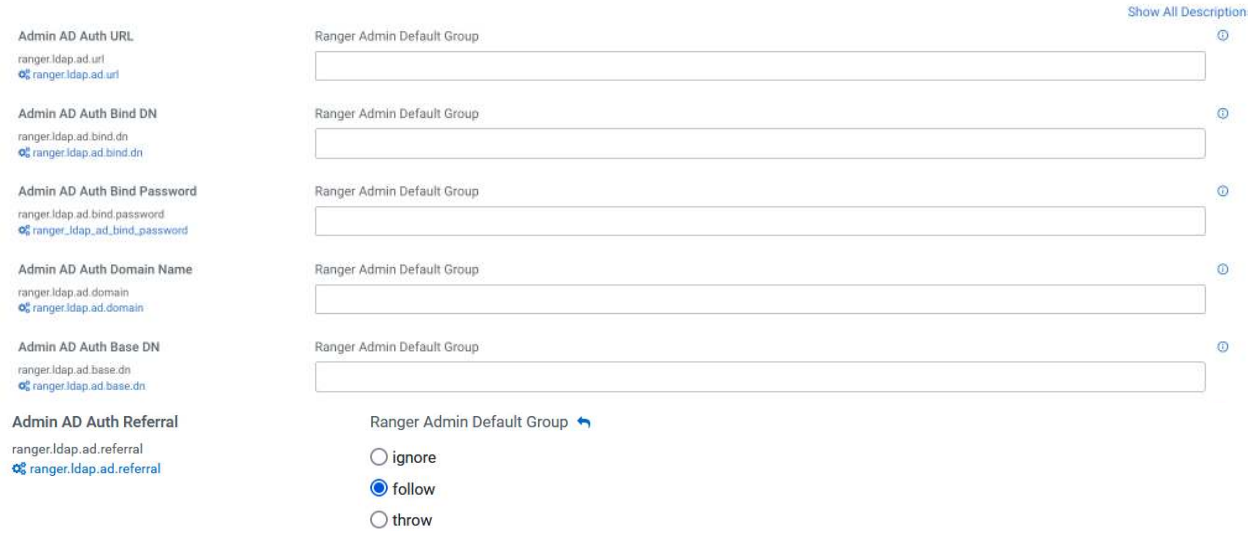
Component	Value
Admin **LDAP**/AD Auth URL:	ldap://ipaserver.cldrsetup.local:389/ ***(Give LDAP or AD Server LDAP ADDR)***
Admin **LDAP**/AD Auth **Bind User**/ Bind DN:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
Admin **LDAP**/AD Auth Bind User Password:	***<cloudera123>*** (password for KDC admin, configured earlier)
Admin **LDAP**/AD Auth User DN Pattern:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
Admin **LDAP**/AD Auth User Search Filter:	**For AD:** (&(objectClass=user)(sAMAccountName={0})) **For LDAP**: (&(objectClass=person)(uid={0}))
Admin **LDAP**/AD Auth Group Search Base:	cn=groups,cn=accounts,dc=cldrsetup,dc=local
Admin **LDAP**/AD Auth Group Search Filter:	**For AD:** (&(objectClass=group)(member={0})) **For LDAP**: (&(objectClass=posixGroup)(memberUid={0}))

Component	Value
Admin **LDAP**/AD Auth Group Role Attribute:	cn
Admin **LDAP**/AD Auth Base DN:	dc=cldrsetup,dc=local
Admin **LDAP**/AD Auth Referral:	follow
Admin AD Auth Domain Name: (For AD Setup)	cldrsetup.local

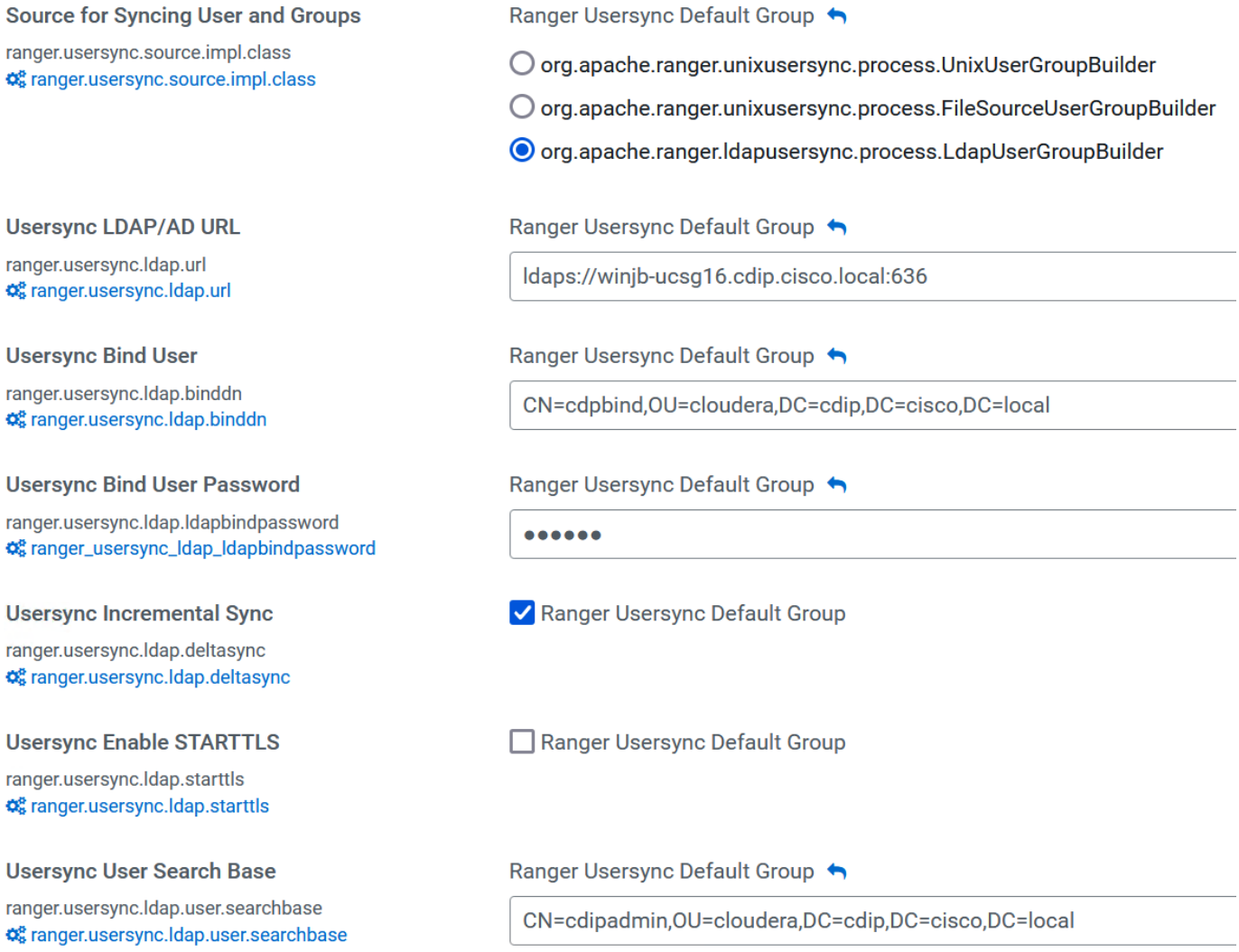
(Search under Configuration for- *ranger.ldap*)



Additional parameters required for AD Based integration: (Search under Configuration for- *ranger.ldap.ad*)



5. Edit Usersync configuration. Example values set are shown in the screenshot below:



Usersync User Search Scope
ranger.usersync.ldap.user.searchscope
ranger.usersync.ldap.user.searchscope
Ranger Usersync Default Group
- (•) sub
- ( ) base
- ( ) one

Usersync User Object Class
ranger.usersync.ldap.user.objectclass
ranger.usersync.ldap.user.objectclass
Ranger Usersync Default Group
user

Usersync User Search Filter
ranger.usersync.ldap.user.searchfilter
ranger.usersync.ldap.user.searchfilter
Ranger Usersync Default Group
(&(objectClass=user)(sAMAccountName={0}))

Usersync User Name Attribute
ranger.usersync.ldap.user.nameattribute
ranger.usersync.ldap.user.nameattribute
Ranger Usersync Default Group
sAMAccountName

Usersync Referral
ranger.usersync.ldap.referral
ranger.usersync.ldap.referral
Ranger Usersync Default Group
- ( ) ignore
- (•) follow
- ( ) throw

Usersync Username Case Conversion
ranger.usersync.ldap.username.caseconversion
ranger.usersync.ldap.username.caseconversion
Ranger Usersync Default Group
- ( ) none
- (•) lower
- ( ) upper

Usersync Groupname Case Conversion
ranger.usersync.ldap.groupname.caseconversion
ranger.usersync.ldap.groupname.caseconversion
Ranger Usersync Default Group
- ( ) none
- (•) lower
- ( ) upper

Usersync Enable User Search
ranger.usersync.user.searchenabled
ranger.usersync.user.searchenabled
[x] Ranger Usersync Default Group

Usersync Group Search Base
ranger.usersync.group.searchbase
ranger.usersync.group.searchbase
Ranger Usersync Default Group
CN=cdipadmin,OU=cloudera,DC=cdip,DC=cisco,DC=local

Usersync Group Object Class
ranger.usersync.group.objectclass
ranger.usersync.group.objectclass
Ranger Usersync Default Group
group

Usersync Group Name Attribute
ranger.usersync.group.nameattribute
ranger.usersync.group.nameattribute
Ranger Usersync Default Group
cn

Usersync Group Member Attribute
ranger.usersync.group.memberattributename
ranger.usersync.group.memberattributename
Ranger Usersync Default Group
member

Usersync Group Hierarchy Levels
ranger.usersync.ldap.grouphierarchylevels
ranger.usersync.ldap.grouphierarchylevels
Ranger Usersync Default Group
0

Usersync Ldap Group Names
ranger.usersync.ldap.groupnames
ranger.usersync.ldap.groupnames
Ranger Usersync Default Group
⊕

Table 9. UserSync LDAP Integration

Component	Value
Source for Syncing User and Groups:	org.apache.ranger.ldapusersync.process.LdapUserGroupBuilder
Ranger Usersync Unix Backend:	nss
Usersync LDAP/AD URL:	ldap://ipaserver.cldrsetup.local:389/
Usersync Bind User:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
Usersync Bind User Password:	*<cloudera123>* (password for KDC admin, configured earlier)
Usersync User Search Base:	cn=users,cn=accounts,dc=cldrsetup,dc=local
Usersync User Search Scope:	sub
Usersync User Object Class:	person
Usersync User Search Filter:	uid=*
Usersync User Name Attribute:	uid
Usersync Referral:	follow
Usersync Username Case Conversion:	none
Usersync Groupname Case Conversion:	none
Usersync Enable User Search:	Ranger Usersync Default Group
Usersync Group Search Base:	cn=groups,cn=accounts,dc=cldrsetup,dc=local
Usersync Group Search Scope:	sub
Usersync Group Object Class:	ipausergroup
Usersync Group Name Attribute:	cn
Usersync Group Member Attribute:	member

6. Click on save changes.
7. Restart Ranger service.
8. Login to Ranger Admin WebUI with ldap authentication

For more details:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/security-ranger-authentication-unix-ldap-ad/topics/security-ranger-authentication-ldap-settings.html

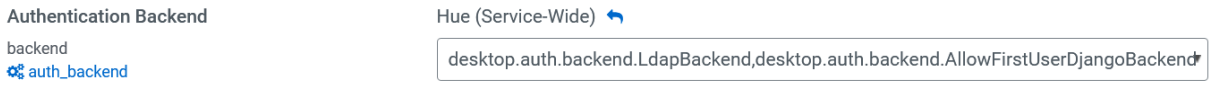
****

Configure Hue for LDAP Authentication (Optional– Skip this Step)

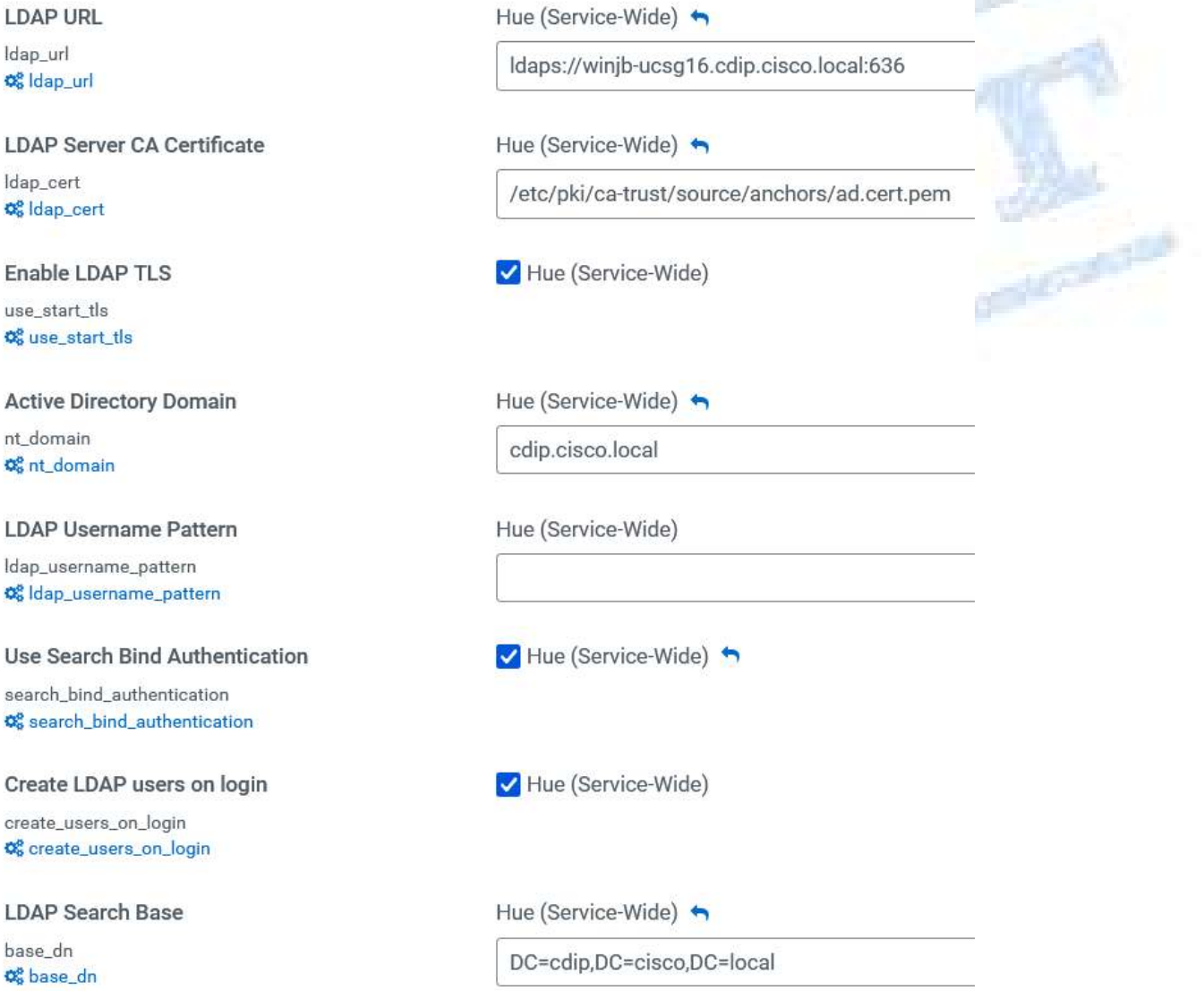
Configuring Hue for Lightweight Directory Access Protocol (LDAP) enables you to import users and groups from a directory service, synchronize group membership manually or automatically at login, and authenticate with an LDAP server. Hue supports Microsoft Active Directory (AD) and open standard LDAP such as OpenLDAP and Forgerock OpenDJ Directory Services.

1. Login to ***Cloudera Manager***. Go to ***Cluster > Hue > Configuration***.

2. Change value for Authentication Backend – desktop.auth.backend.LdapBeckend,desktop.auth.backend.AllowFirstUserDjangoBackend



3. Edit value for LDAP configuration. Example values set are shown in the screenshot below:



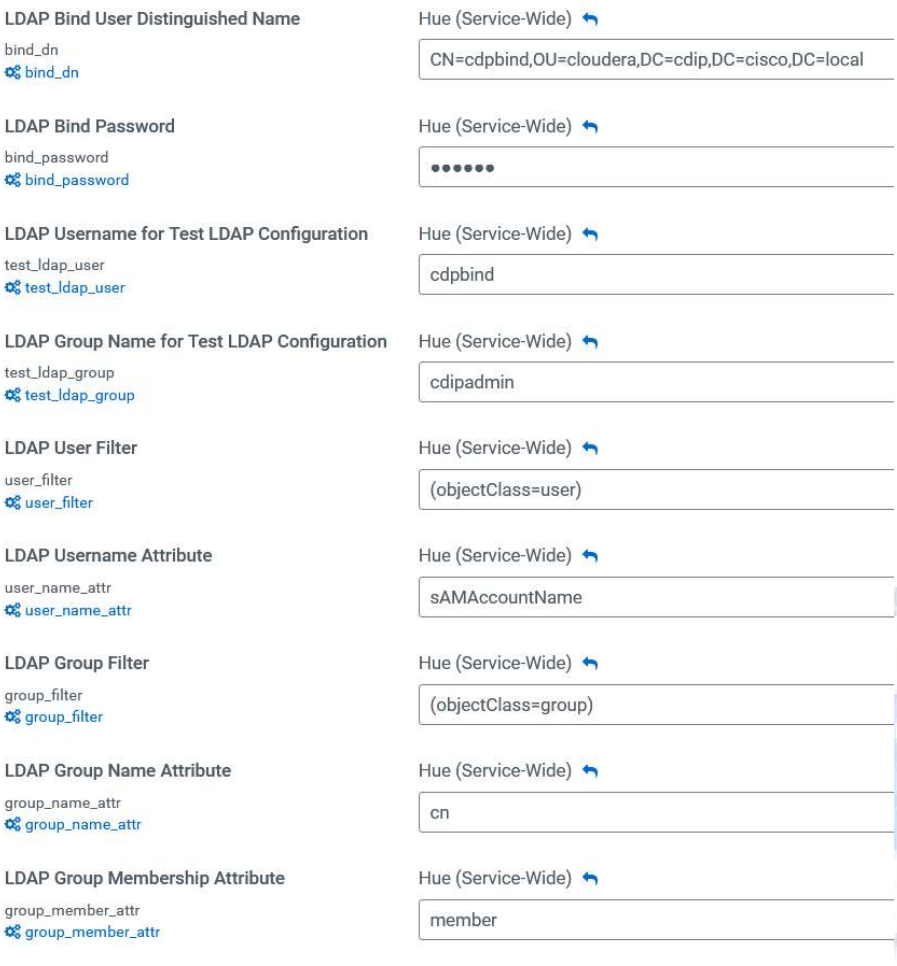


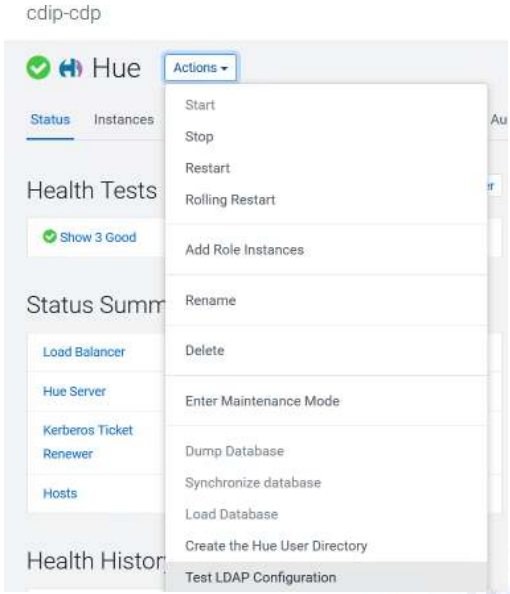
Table: **LDAP Integration**

Component	Value
LDAP URL:	ldap://ipaserver.cldrsetup.local:389/
LDAP Server CA Certificate (Optional):	/root/cacert.p12
Enable LDAP TLS (Hue):	True (Checked)
LDAP Search Base:	dc=cldrsetup,dc=local
LDAP Bind User Distinguished Name:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Bind Password:	***<cloudera123>*** (password for KDC admin, configured earlier)
LDAP Username for Test LDAP Config:	admin
LDAP Group Name for Test LDAP Config:	users
LDAP User filter:	(&(uid={0})(objectClass=person))
LDAP Group filter:	(&(member={1})(objectClass=posixgroup))
LDAP Group Name Attribute:	cn
LDAP Group Membership Attribute:	member

4. Click on save changes

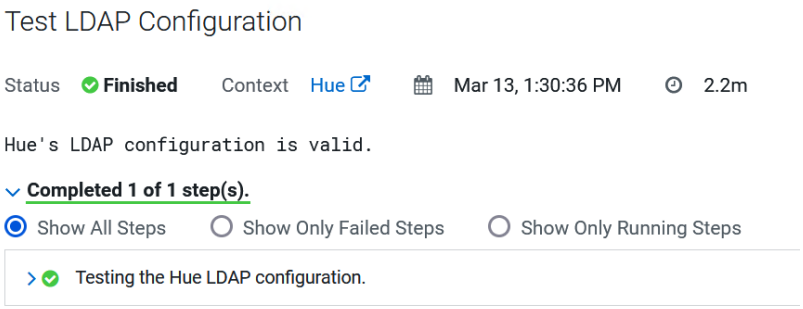
5. Restart HUE service.

6. Click on Actions next to Hue. Click on Test LDAP Configuration.



7. Click on Test LDAP Configuration.

8. Click on Finish.



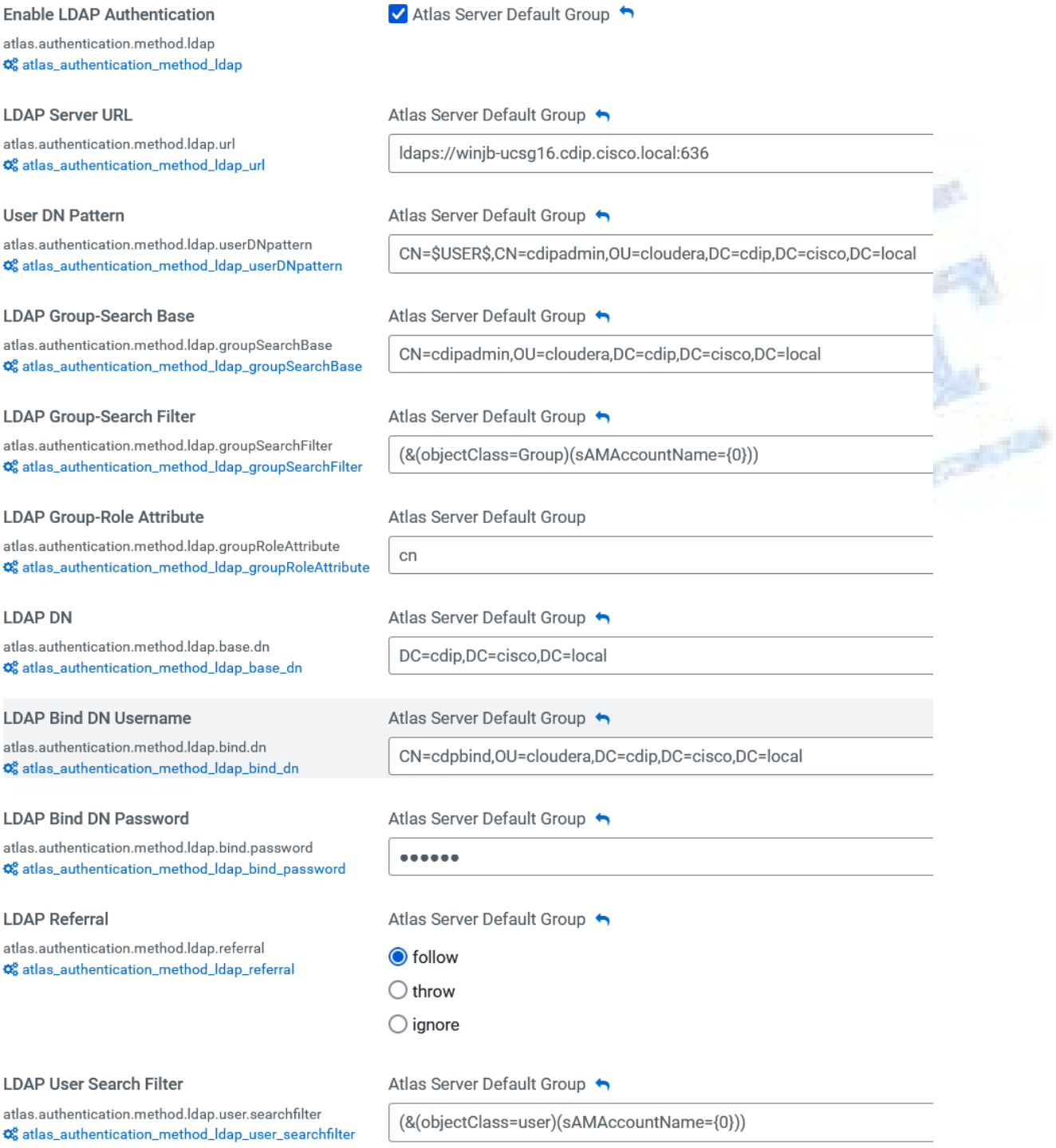
For more details:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/securing-hue/topics/hue-authenticate-users-with-ldap.html

****

Configure Atlas for LDAP authentication (Optional– Skip this Step)

Follow steps below to configure Atlas authentication for LDAP. (***If LDAP is not integrated for Atlas, we need to use local OS users present on the node where Atlas server is installed i.e. base-master)***

1. Login to Cloudera Manager WebUI. Go to Cluster > Atlas > Configuration.
2. Edit LDAP configuration. Sample configuration is shown in the screenshot below:



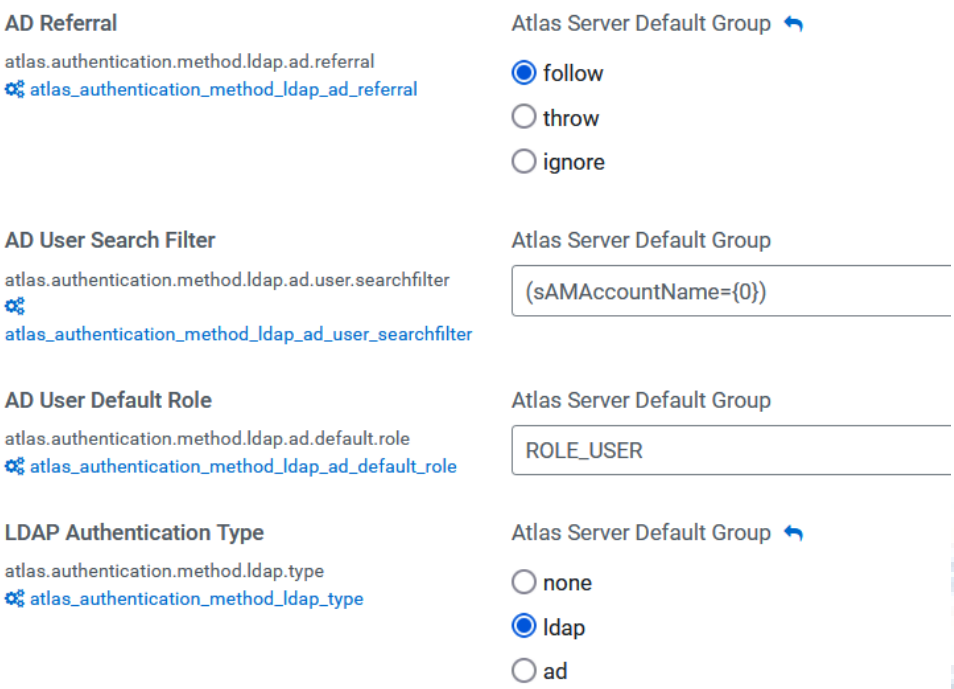


Table 10. Atlas LDAP Integration

Component	Value
Enable LDAP Authentication (Atlas):	True (Checked)
LDAP Server URL:	ldap://ipaserver.cldrsetup.local:389/
User DN Pattern:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Group Search filter:	(&(member={1})(objectClass=posixgroup))
LDAP Group Search Base:	cn=groups,cn=accounts,dc=cldrsetup,dc=local
LDAP Group-Role Attribute:	cn
LDAP DN:	dc=cldrsetup,dc=local
LDAP Bind DN Username:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Bind DN Password:	*<cloudera123>* (password for KDC admin, configured earlier)
LDAP Referral:	follow
LDAP User filter:	(&(uid={0})(objectClass=person))
LDAP Authentication Type:	LDAP
AD Referral: **(Only for AD Setup)**	follow
AD User Search Filter: **(Only for AD Setup)**	(sAMAccountName={0})
AD User Default Role: **(Only for AD Setup)**	ROLE_USER

3. Click on save changes.

4. Restart Atlas service.

For more details:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/atlas-securing/topics/atlas-configure-ldap-authentication.html

****

Configure Hive for LDAP Authentication (Optional– Skip this Step)

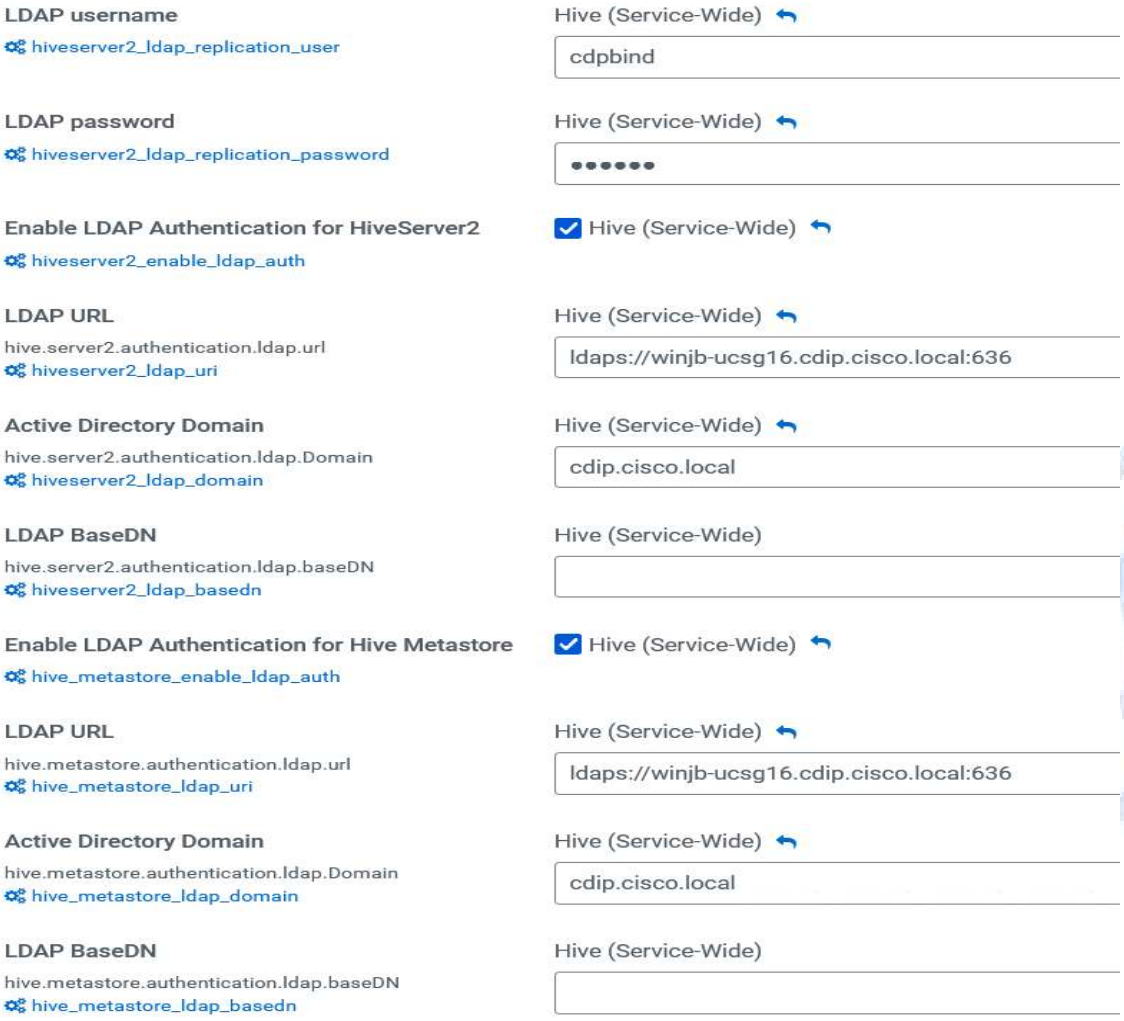


Table 11. LDAP Integration-Hive

Component	Value
LDAP Username:	admin
LDAP Password:	*<cloudera123>* (password for KDC admin, configured earlier)
Enable LDAP Authentication for HiveS2:	True (Checked)
LDAP URL:	ldap://ipaserver.cldrsetup.local:389/
LDAP Base DN:	dc=cldrsetup,dc=local
Enable LDAP Authentication for HMS:	True (Checked)
LDAP URL:	ldap://ipaserver.cldrsetup.local:389/
LDAP Base DN:	dc=cldrsetup,dc=local

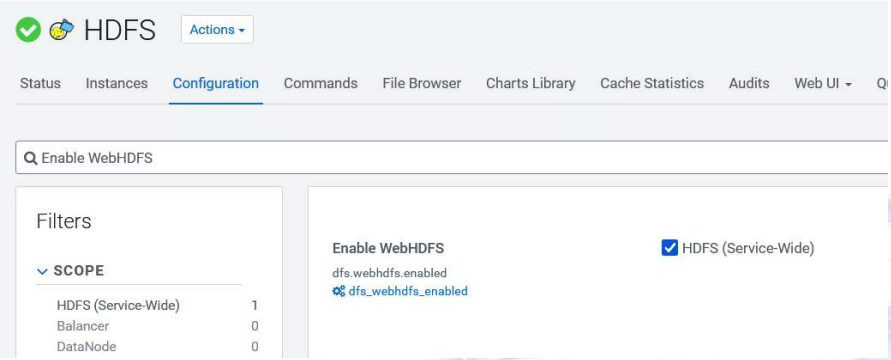
****

Configure HDFS properties to optimize log collection (Optional– Skip this Step)

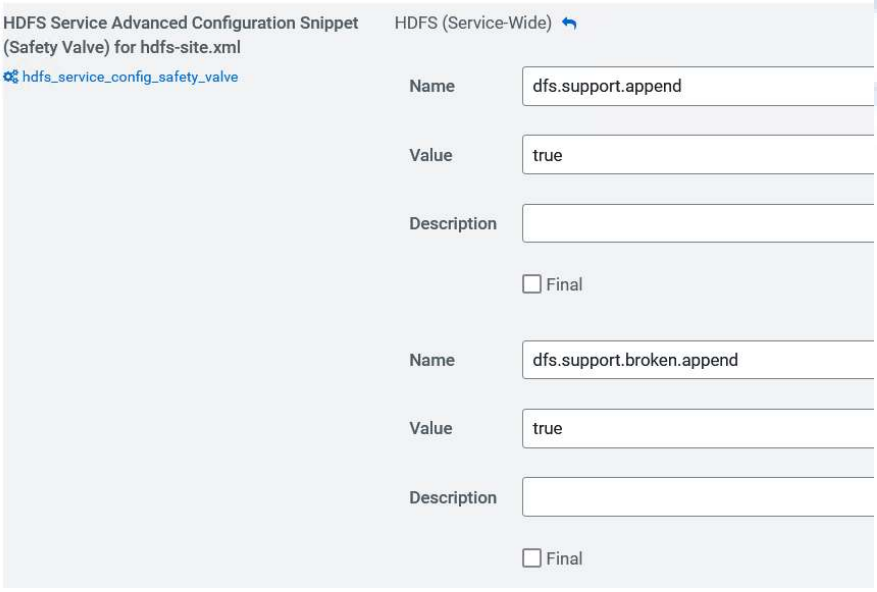
CDP uses “out_webhdfs” Fluentd output plugin to write records into HDFS, in the form of log files, which are then used by different Data Services to generate diagnostic bundles. Over time, these log files can grow in size. To optimize the size of logs that are captured and stored on HDFS, you must update certain HDFS configurations in the hdfs-site.xml file using Cloudera Manager.

1. Login to ***Cloudera Manager WebUI***.
2. Go to ***Cluster*** > ***HDFS Service*** > ***Configuration.***
3. Enable ***WebHDFS.***

cdip-cdp



4. Edit value for HDFS Service Advanced Configuration Snippet (Safety Valve) for hdfs-site.xml as shown in the screenshot below:



5. Click Save Changes.
6. Restart the HDFS service.
7. Restart Cloudera on premises Base cluster.

****

Cloudera on premises (PvC) Data Services (DS) Installation

Cloudera on premises Data Services lets you deploy and use the Cloudera Data Warehouse (CDW), Cloudera AI a.k.a. Cloudera Machine Learning (CML/CAI), and Cloudera Data Engineering (CDE) Data Services.

This section summarizes Cloudera on premises Data Science v1.5.5 installation through Embedded Container Service on Cloudera on premises Base 7.3.1.

A Cloudera on premises Data Services deployment includes an Environment, a Data Lake, the Management Console, and Data Services (Data Warehouse, Machine Learning, Data Engineering). Other tools and utilities include Replication Manager, Data Recovery Service, CDP CLI, and monitoring using Grafana.

To deploy Cloudera on premises Data Services you need a Cloudera on premises Base cluster, along with container-based clusters that run the Data Services. You can either use a dedicated ***RedHat OpenShift container cluster (OCP)*** or deploy an ***Embedded Container Service (ECS)*** container cluster.

The Cloudera on premises deployment process involves configuring Management Console, registering an environment by providing details of the Data Lake configured on the Base cluster, and then creating the workloads.

Platform Managers and Administrators can rapidly provision and deploy the data services through the Management Console, and easily scale them up or down as required.

Cloudera on premises Base provides the following components and services that are used by Cloudera on premises Data Services:

- SDX Data Lake cluster for security, metadata, and governance
- HDFS and Ozone for storage
- Powerful and open-source Cloudera Runtime services such as Ranger, Atlas, Hive Metastore (HMS), etc.
- Networking infrastructure that supports network traffic between storage and compute environments.

Embedded Container Service (ECS) checklist

Use the checklist for Embedded Container Service (ECS) for Cloudera on premises Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation-ecs/topics/cdppvc-installation-ds-checklist.html

Cloudera on premises Data Services software requirements

1. You must have a minimum of one agent node for ECS.
2. Enable TLS on the Cloudera Manager cluster for communication with components and services.
3. Set up Kerberos on these clusters.
4. Ensure that all of the hosts in the ECS cluster have more than ***300 GiB*** of free space in the each ***/var/lib*** and ***/docker*** directories, on each host in a Cloudera on premises Containerized Cluster, at the time of installation. The default docker service uses ***/docker*** folder. Whether you wish to retain ***/docker*** or override ***/docker*** with any other folder, you ***must have a minimum of 300 GiB free space***. The hosts in a Cloudera on premises Containerized Cluster that have GPUs are required to have nVidia Drivers and nvidia-container-runtime installed.

5. ***Python 3.8<*** is required for Cloudera Manager version 7.11.3.0 and higher versions. Cloudera Manager agents will not start unless Python 3.8 is installed on the cluster nodes.

6. The cluster generates multiple hosts and host-based routing is used in the cluster in order to route it to the right service. You must decide on a domain for the services which Cloudera Manager by default points to one of the host names on the cluster. However, during the installation, you should check the default domain and override the default domain (only if necessary) with what you plan to use as the domain. The default domain must have a ***wildcard DNS entry***. For example, "****.apps.cldrsetup.local***".

7. It is recommended that you leave IPv6 enabled at the OS level on all ECS nodes.

8. Take care of enough disk space is available on each host in ECS cluster.

9. On each of the ECS hosts, create three partitions of the attached 2T EBS volume (non-root) and mount those partitions as volumes ***/lhdata /cdwdata /docker***. (***/docker*** is used for docker cache, ***/lhdata*** for LongHorn NFS Volume storage, and ***/cdwdata*** for local storage and CDW)

10. ECS requires JDK, krb5-workstation, krb5-libs, NTP, iptables packages to be installed on all hosts.

11. Enable cgroup v1 in Red Hat Enterprise Linux 9. ***(optional, skip this step)***

```
# In RHEL 9 cgroup-v2 is enabled by default, follow steps below to enable cgroup v1:

# Check if the cgroup-v2 is mounted currently as default.
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "mount | grep cgroup"

# Add the kernel command line parameter systemd.unified_cgroup_hierarchy=0 &
systemd.legacy_systemd_cgroup_controller.

[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a 'grubby --update-kernel=/boot/vmlinuz-$(uname -r)
--args="systemd.unified_cgroup_hierarchy=0 systemd.legacy_systemd_cgroup_controller"'

# Reboot the system for changes to take effect.
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "systemctl reboot"

# Verify the changes after reboot:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "cat /proc/cmdline"
BOOT_IMAGE=(hd0,gpt2)/vmlinuz-5.14.0-162.23.1.el9_1.x86_64 root=/dev/mapper/rhel-root ro
crashkernel=1G-4G:192M,4G-64G:256M,64G-:512M resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root
rd.lvm.lv=rhel/swap rhgb quiet systemd.unified_cgroup_hierarchy=0 systemd.legacy_systemd_cgroup_controller

# Mount shows legacy cgroup-v1 mounted now.
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "mount | grep cgroup"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "ll /sys/fs/cgroup"
```

12. For CAI, you must install ***nfs-utils*** in order to mount longhorn-nfs provisioned mounts. The ***nfs-utils*** package is required on every node of the ECS cluster. Run this command ***"dnf install nfs-utils"*** to install ***nfs-utils***.

```
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "dnf install -y nfs-utils iscsi-initiator-utils"
```

13. Check ***iptables*** is installed and working fine.

```
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "iptables -L"
```

14. If ***iptables*** is not working as expected or giving command not found error then remove the package on all ecs nodes and re-install.

```
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "dnf remove -y iptables && dnf install -y iptables
&& iptables -L"
```

15. If ***iptables*** is not present on ecs cluster nodes, then Install ***iptables***, and verify if it is working as expected.

```
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "dnf install -y iptables && iptables -L"
```

16. For nodes with NVIDIA GPU ***(if applicable)***, ensure that the GPU hosts have nVidia Drivers and ***nvidia-container-runtime*** installed. You must confirm that drivers are properly loaded on the host by executing the command nvidia-smi. You must also install the nvidia-container-toolkit package. ***(skip this step)***

17. You must install ***nvidia-container-toolkit (if applicable)***. (***nvidia-container-runtime*** is migrated to ***nvidia-container-toolkit***, see Migration Notice.) The steps for this are shown in the NVIDIA Installation Guide. If using Red Hat Enterprise Linux (RHEL), use dnf to install the package. See Installing the NVIDIA Container Toolkit. ***(skip this step)***

```
##### Verify linux version:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "uname -m && cat /etc/*release"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "uname -a"

##### Verify GCC installation and version
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "gcc --version"

##### Verify nodes with NVIDIA GPU installed:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "lspci -nnv | grep -i nvidia"

##### Set subscription to RHEL9.x
[root@ipaserver ~]# ansible all -m shell -a "subscription-manager release --set=9.4"
[root@ipaserver ~]# ansible all -m shell -a "subscription-manager release --show"
[root@ipaserver ~]# ansible all -m shell -a "sudo dnf clean all"
[root@ipaserver ~]# ansible all -m shell -a "sudo rm -rvf /var/cache/dnf"

##### (Optional if not completed prior) Enable optional repos - On RHEL 9 Linux only
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "subscription-manager repos
--enable=rhel-9-for-x86_64-appstream-rpms"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "subscription-manager repos
--enable=rhel-9-for-x86_64-baseos-rpms"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "subscription-manager repos
--enable=codeready-builder-for-rhel-9-x86_64-rpms"

##### Install kernel headers and development packages for the currently running kernel
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo dnf install kernel-devel-$(uname -r)
kernel-headers-$(uname -r)"

##### Remove outdated Signing Key:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo rpm --erase gpg-pubkey-7fa2af80*"

##### Download and Install NVIDIA CUDA Toolkit [This exercise documented with CUDA 12.3.2 for RHEL 9
rpm(local) installation]
[root@ipaserver ~]# wget
https://developer.download.nvidia.com/compute/cuda/12.2.2/local_installers/cuda-repo-rhel9-12-2-local-12.2
.2_535.104.05-1.x86_64.rpm
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m copy -a
"src=/root/cuda-repo-rhel9-12-2-local-12.2.2_535.104.05-1.x86_64.rpm
dest=/root/cuda-repo-rhel9-12-2-local-12.2.2_535.104.05-1.x86_64.rpm"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo rpm --install
cuda-repo-rhel9-12-2-local-12.2.2_535.104.05-1.x86_64.rpm"

##### From the NVIDIA Driver Downloads page, download NVIDIA Driver
https://www.nvidia.com/download/index.aspx as per the GPU, OS and NVIDIA CUDA version.
[root@ipaserver ~]# wget
https://nvidia.github.io/libnvidia-container/stable/rpm/nvidia-container-toolkit.repo

[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m copy -a
"src=/root/nvidia-driver-local-repo-rhel9-535.161.07-1.0-1.x86_64.rpm
dest=/root/nvidia-driver-local-repo-rhel9-535.161.07-1.0-1.x86_64.rpm"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo rpm --install
nvidia-driver-local-repo-rhel9-535.161.07-1.0-1.x86_64.rpm"

[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo dnf clean all"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo dnf -y module install
nvidia-driver:latest-dkms"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo dnf -y install cuda"

##### Enable nvidia-persistenced services:
```

```
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo systemctl enable
nvidia-persistenced.service"

##### Reboot the machine:
[root@ipaserver ~]#  ansible ecsmasternodes,ecsnodes -m shell -a "sudo reboot"

##### After the machine boots, verify that the NVIDIA drivers are installed properly:
[root@ipaserver ~]#  ansible ecsmasternodes,ecsnodes -m shell -a "sudo nvidia-smi"

##### Installing with dnf
##### Configure the production repository:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m copy -a "src=/root/nvidia-container-toolkit.repo
dest=/etc/yum.repos.d/nvidia-container-toolkit.repo"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo dnf config-manager --enable
nvidia-container-toolkit-experimental"
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m shell -a "sudo dnf install -y
nvidia-container-toolkit"
```

If you are installing ECS on RHEL 8 or RHEL 9: ***(Not Required for Cloud Based Servers, perform only on Datacenter/Bare-metal environments for RKE Based ECS clusters)***

```
##### Run the following command to check to see if the nm-cloud-setup.service and nm-cloud-setup.timer
services are enabled:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m copy -a "systemctl status nm-cloud-setup.service
nm-cloud-setup.timer"
#####  If the nm-cloud-setup.service and nm-cloud-setup.timer services are enabled, disable them by
running the following command on each host you added:
[root@ipaserver ~]# ansible ecsmasternodes,ecsnodes -m copy -a "systemctl disable nm-cloud-setup.service
nm-cloud-setup.timer && systemctl stop nm-cloud-setup.service nm-cloud-setup.timer"
##### For more information, see Known issues and limitations.
If you disabled the nm-cloud-setup.service and nm-cloud-setup.timer services, reboot the added hosts.
```

Note: For more information, see: Known issues and limitations.

Note: Prepare Cloudera on premises Base for the Cloudera on premises Data Services installation: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation-ecs/topics/cdppvc-installation-ecs-prepare-cdp-private-cloud-base.html

Note: Use this checklist to ensure that your Cloudera on premises Base is configured and ready for installing Cloudera on premises Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation-ecs/topics/cdppvc-installation-pvcbase-checklist.html

Note: Use this checklist to ensure that your Embedded Container Service (ECS) is configured and ready for installing Cloudera on premises Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation-ecs/topics/cdppvc-installation-ds-checklist.html

Checklist Item	Details
Docs Links	Docs - ECS Checklist Docs - ECS System Requirements Support Matrix

ClouderaManager version	7.11.3 - CHF6 (minimum)
LDAP Configuration	CM is configured for LDAP, and you have a copy of the CA trust chain that signed your LDAP server's cert (i.e. root cert & intermediates if any).
LDAP Bind User Account	You have the account and password for LDAP Bind user account.
Base Cluster & CM Agents TLS	TLS: AutoTLS? Yes/No TLS: AutoTLS, signed with customer CA? Yes/No TLS: Manual, signed with customer CA? Yes/No TLS: Manual self signed (not recommended) xxx
Kerberos Admin Account	CM is setup with Kerberos admin account (cloudera-scm account & password).
ECS Installer Prerequisites Checker	IMPORTANT! CM ECS installer now has an Enforcing prerequisites checker, install will fail if your infra does not meet requirements outlined in docs. See item 14, in doc.
Runtime Version	7.3.1 (CHF6+) is installed & configured. See support matrix
New Changes with 1.5.5	
Minimum Required Components	Zookeeper, HDFS, Ozone, HBase, Hive Metastore, Kafka, Solr-Infra, Ranger, Atlas, YARN (optional for CDW, but required for Spark pushdown in CDE, CAI).
Base Components TLS	All Base components configured with TLS.
Base Components Kerberos	All Base components configured with Kerberos.
CDW Base Cluster HMS	CDW - Base HMS, mTLS can be used instead of user/password. If using CDW, the Base Cluster HMS is configured to allow TLS connections (set to allow TLS, but not forced to require it).

Wildcard DNS Record	CRITICAL: A wildcard subdomain (called app_domain in cm) has been created in DNS. For AD this usually means a subdomain folder + an "A record" with name = "*" pointing to the ECS host and a CNAME record. The "A record" does not need a reverse PTR.* ***AD Example:** If corp domain is "company.com", then create a new subdomain folder called "ecs-dev", then within that create AD another called "apps", then create a "A record" inside that folder, its name is "*", its IP will be the IP of the host you will install ECS Server onto. Verify with `dig` utility e.g., `$ dig foobar.apps.cdppvc.com`. Look for "ANSWER" section in the result.
DNS Resolver	DNS needs to be the primary resolver, do not use entries in /etc/hosts (except for localhost).
ECS Nodes File System	RHEL 8 & RHEL 9 - disk partition that backs "/var" should be xfs file system (using the default ftype = 1), this is usually the root partition /.
RHEL/Centos 7.x Support	**New in 1.5.5:** RH/Centos 7.x no longer supported. If running RHEL 7.x, you must upgrade to a higher version before installing Cloudera on premises Data Services.
Hostname Configuration	Hostname must = FQDN, use `hostnamectl set-hostname <fqdn>`.
/etc/resolv.conf Entries	/etc/resolv.conf must NOT have more than 3 "nameserver" entries.
MTU Size	Ensure that net interfaces are configured to a maximum transmission unit size (MTU) not smaller than 1450, (typical MTU is 1500), required by the Calico CNI plugin.
Storage Devices	Validate that Storage devices meet Sys Requirements (see docs) and are mounted using a consistent naming convention. Devices are expected to be SCSI block devices (NFS mounts do not work). Devices for CDW Cache and Block should be separate mounts. Use a Logical Volume Manager if you have multiple devices per type.
Disk Mounts	All disk mounts should be xfs(type1), including root partition. The partition backing /var/lib **must have at least 300GB free** after install of CM agent. If 300GB free is not available, then must symlink "/var/lib/docker" & "/var/lib/rancher" from another partition (must be xfs).

Required Mounts	3 mounts required per host, name them any you like, DO NOT USE symbolic links. **See**: https://longhorn.io/docs/1.5.5/volumes-and-nodes/multidisk/#use-an-alternative-path-for-a-disk-on-the-node 1) A xfs mount for Docker storage (used for install staging, CAI model building, and embedded repo storage) at least 300GB 2) A xfs mount for CDW Cache, this is a physical NVMe device, sizing dependent on use case, consumed in 600GB chunks per executor & coordinator 3) A xfs mount for Longhorn Block storage which provides k8s Persistent Volumes (PVs), min 1 TB per node, vendor recommends this mount should be a Logical Volume (from LVM2) (for future expansion needs)
Passwordless Sudo User	You have credentials for a passwordless sudo user for each host. If you don't have these credentials, then you will have to install CMAgents manually (not using wizard) and configure them for TLS manually.
CM Agents Configuration	Nodes are configured with requirements to run CM Agents.
iptables Configuration	Do NOT disable iptables, but ensure iptables is set to defaults. Any customer-set chains/rules must be un-set. (firewalld not supported, nftables not yet certified). Check if `/etc/sysconfig/iptables` exists and inform the customer to remove it. Ensure no system management software like puppet, ansible, etc., will recreate it on reboot. ECS will attempt to install iptables-services if it's not present.
RHEL8 - iptables	Install iptables like: `# yum --setopt=tsflags=noscripts install -y iptables`. See docs: ECS Installation Steps.
RHEL8 - Net Manager	Services `nm-cloud-setup.service` and `nm-cloud-setup.timer` must be stopped (if they exist) and disabled. This is seen on AWS RHEL images, unlikely to see this on-prem. (**Risky, as we found instances can get corrupt, due to network connectivity may lost from AWS side)**
RHEL8 - VMware	Certain versions of VMware virtual network interface (vmxnet3) can cause what seems like intra-pod packet drops. This is due to the Calico Issue. Workaround: See case Cloudera Support.
RHEL9 - iptables	Check iptables configuration as described for RHEL8.

Time Service	Host has chronyd or NTP running.
DNS Resolving	Each ECS host is forward and reverse DNS resolvable.
DNS Forward and Reverse	Each ECS host can forward and reverse DNS resolve each base host.
SELinux and System Settings	SELinux off or set to Permissive, vm.swappiness = 1, Transparent Huge Pages Off (1.3.4+ support SELinux, see docs).
NFS Requirements	Linux package "nfs-utils" must be installed for CAI & CDE.
GPU Requirements & Config	See docs for GPU requirements & config for CAI.
3rd Party Software	Certain 3rd party network monitors/firewalls may seriously interfere with ECS cluster traffic. Illumio... Illumio runs an agent on each host, uses iptables and can block traffic if it is running in Primary Firewall Mode. That mode must be set to = off. See Illumio Docs. VMware NSX-T can create blocking firewall rules. See NSX-T Jira.
Air Gapped Customers	If a customer is Air Gapped, they must download all the ECS bits and stage them behind an HTTP server. Make sure they do this right away as this is 150GB of content (circa 1.5.0).
License Key	In every case, Customer (or Cloudera employee) MUST have a valid license key that includes an entitlement for PrivateCloud!
Ingress Certificates	It is usually better to have a customer CA signed ECS Ingress domain cert, but not required as ECS will sign using an RKE CA cert (the RKE CA will have to be pre-trusted on a user's machine, or the browser will show "insecure" TLS connection). This ECS generated CA is not managed by CM's auto-TLS, so for JDBC/impala-shell/beeline/other connections, you must add this to a client truststore manually. The ECS generated CA will have a CN=rke2-server-ca@xxxxxxxxxx and is located at /var/lib/rancher/rke2/server/tls/server-ca.crt.

Customer CA Signed Certs	For customer CA signed certs: This cert must include 2 SubjectAltNames. If wildcard format is not allowed, let the installer use the RKE CA. Example: DNS.1 = *.apps.cdppvc.com DNS.2 = apps.cdppvc.com
New in 1.5:	
No Wildcard TLS	There is an option to deploy when wildcard SAN entries are not allowed. See doc for method & limits: No wildcard TLS.
CAI Workbench Certificates	If installing in a network domain that requires strict host checking, HSTS, you will need to use TLS for CAI Workbenchs. Fun fact: "cloudera.com" requires HSTS, which is usually not seen with customers but is emerging. This cert must include SubjectAltNames. DNS.1 = *.xxxx-xxxx.apps.cdppvc.com (xxxx-xxxx = the CAI workspace ID)
CDE Virtual Cluster Certificates	Same as CAI for HSTS. CDE includes a utility to generate RKE signed certs for each virtual cluster. If you make your own cert for a CDE virtual cluster, this cert must include SubjectAltNames. DNS.1 = *.xxxx-xxxx.apps.cdppvc.com (xxxx-xxxx = the CDE virtual cluster ID)
CDW HMS Configuration	CDW can be configured to allow non-TLS connections for HMS db. See CDW Documentation.
FreeIPA Support	FreeIPA is now supported. "LDAP Group Search Filter" in Mgmt Console must include **(!(cn=admins))**, or group sync will break e.g. **(&(member={0})(objectclass=posixgroup)(!(cn=admins)))**. Must alter the /etc/krb5.conf - comment out includedir directives. In CM7.10.1, CM generated "local to auth" krb5 rules include embedded "\Q" and "\E" chars, this is legal for base but causes a parsing problem for CDW. Contact for workaround (Should be patched in 1.5.1 - CHF1).
Fixed in 1.5.0:	

/etc/resolv.conf Entries	/etc/resolv.conf must NOT have more than 2 "search" entries, causes a side-effect in Impala Statestore pod. Per Jira.
External DS Metadata DB	New: External DS metadata db is deprecated. This will be a problem for "legacy" customers as we have no migrate from external to embedded db.
Container Repos	Able to use customer-owned container repos is now supported e.g., Artifactory, Nexus, Harbor, Docker dist.
Iceberg Tables	Support for Iceberg Tables.
CDSW to CAI Migration	Tech Preview: CDSW to CAI migration. See release notes.
New in 1.4.1:	
/var/lib Storage Requirements	/var/lib must have 100GB.
Docker Storage Requirements	Docker storage must have 200GB.
vCores Requirements	Min 16 vcores required (on Masters too? Yes).
iptables-save	iptables-save to /var/lib/ecs/iptables.save (we don't need to flush IPtables upon reinstall anymore).
Oracle HMS	Oracle HMS is now GA. This can get complicated as many Oracle customers use TLS + Client cert auth, which we don't support. We support TLS + user password auth only. Contact for workaround.
Data Viz	Data Viz is GA.
Ozone for CDW	Ozone for CDW is Tech Preview.

Impala Custom Pod Sizes	Impala Custom pod sizes now GA. See Impala Pod Configs.
CDE Resource Limits	CDE Resource Limits - Tech Preview.
/etc/resolv.conf Entries (2)	/etc/resolv.conf must NOT have more than 2 "search" entries, causes a side-effect in Impala Statestore pod. Per Jira.
New In 1.3.4:	
Cloudera Manager	Cloudera Manager now prevents ECS Server hosts from running workloads. ECS Servers are masters only. See Cloudera Manager Release Notes.
ECS Hosts Workloads	ECS hosts can now be configured to reserve hosts for workloads that require GPU drivers. See GPU Node Setup.
SELinux Support	SELinux is now supported for ECS clusters. See SELinux Documentation.

****

Installing Cloudera on premises Data Services using ECS

Follow the steps here to install Cloudera on premises Data Services with the ***Embedded Container Service (ECS)***.

Follow the steps outlined below to add hosts to be part of the Cloudera on premises Data Services cluster and the install ECS (embedded container service) through either internet or air gapped method.

https://docs.cloudera.com/cdp-private-cloud-data-services/latest/installation-ecs/topics/cdppvc-installation-ecs-steps.html
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/index.html

Note: We will be installing Cloudera on premises Data Services via the internet method.

Note: For more details on dedicating ECS node for specific workload type please visit: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/managing-ecs/topics/cm-managing-ecs-dedicating-nodes-for-workloads.html

Note: If you do not have entitlements to access https://archive.cloudera.com/p/cdp-pvc-ds/latest/, contact your Cloudera account team to get the necessary entitlements.

Latest ECS Supported Version Of Cloudera-Manager is *7.11.3 CHF11*

****

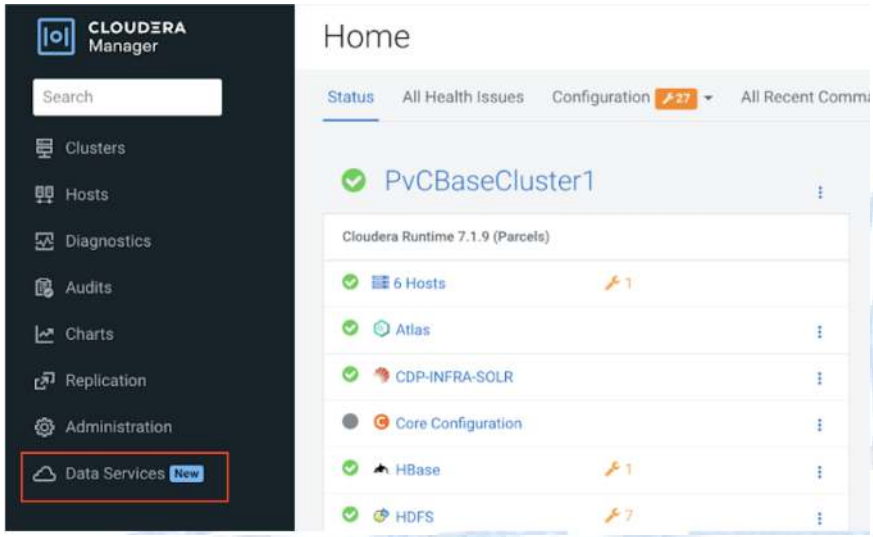
Installing ECS Cluster

Follow the steps in this topic to install Cloudera on premises Data Services with the Embedded Container Service (ECS).

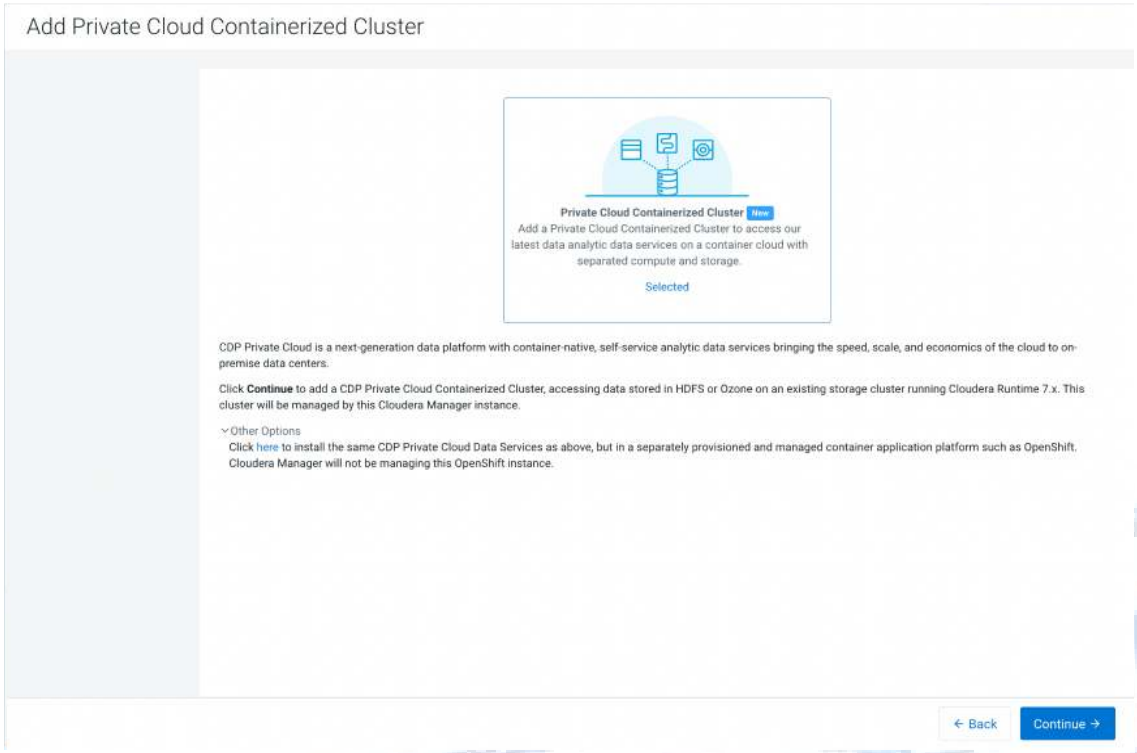
Important:

RHEL 7.x support on ECS has been dropped in Cloudera on premises Data Services 1.5.5 and higher versions. If you are running RHEL 7.x, you must upgrade to a higher version before installing Cloudera on premises Data Services.

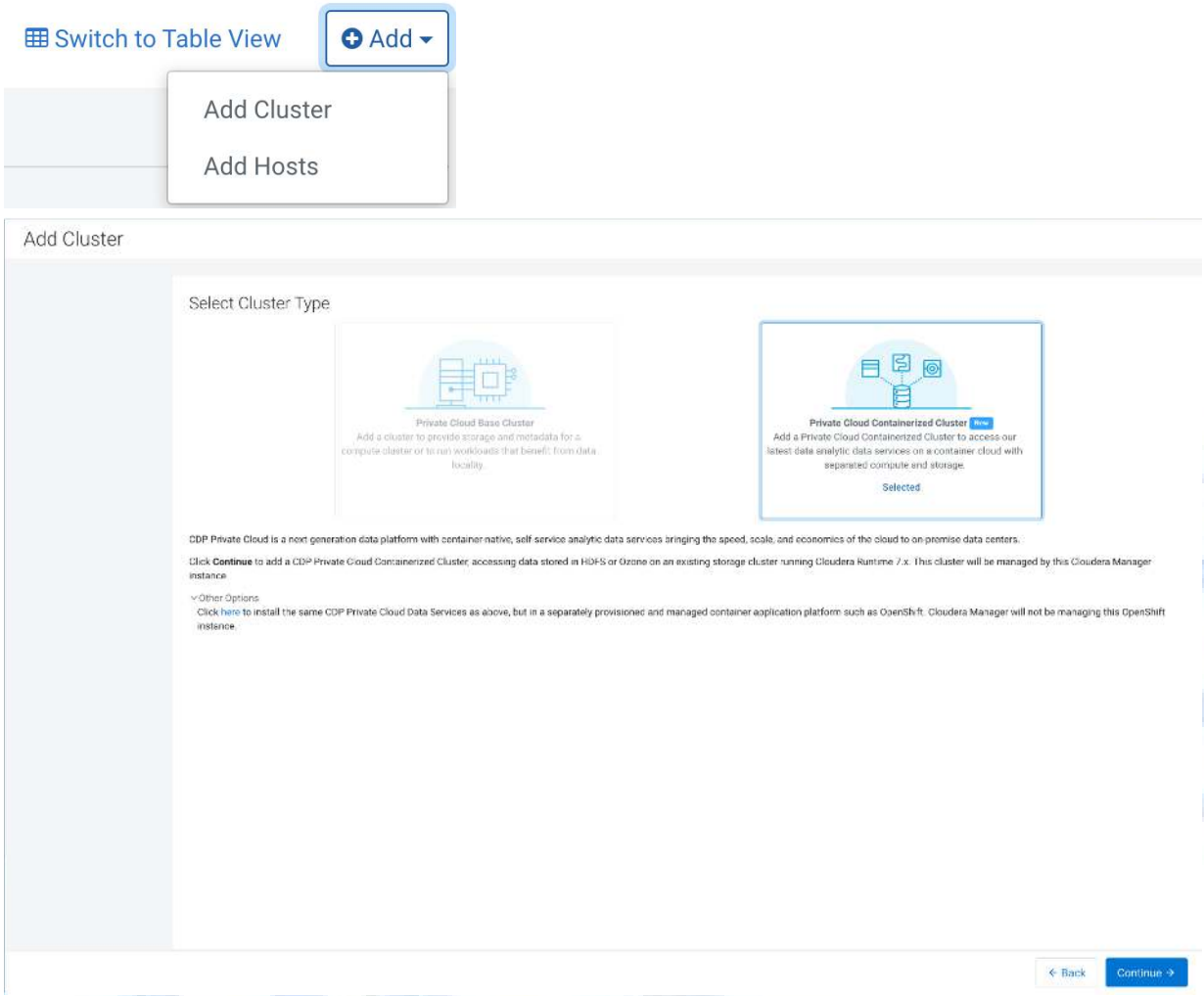
1. In the ***Cloudera Manager WebUI*** console, go to the ***Data Services*** page by clicking on the ***Data Services*** link on the Pane located at the Left Hand side of the browser window. Alternatively, you can also click **(+)** ***Add > Add Cluster*** at the top right in Cloudera Manager, then select ***Private Cloud Containerized Cluster*** as the cluster type.



2. The ***Add Private Cloud Containerized Cluster*** page appears. Click ***Continue*** on the page.



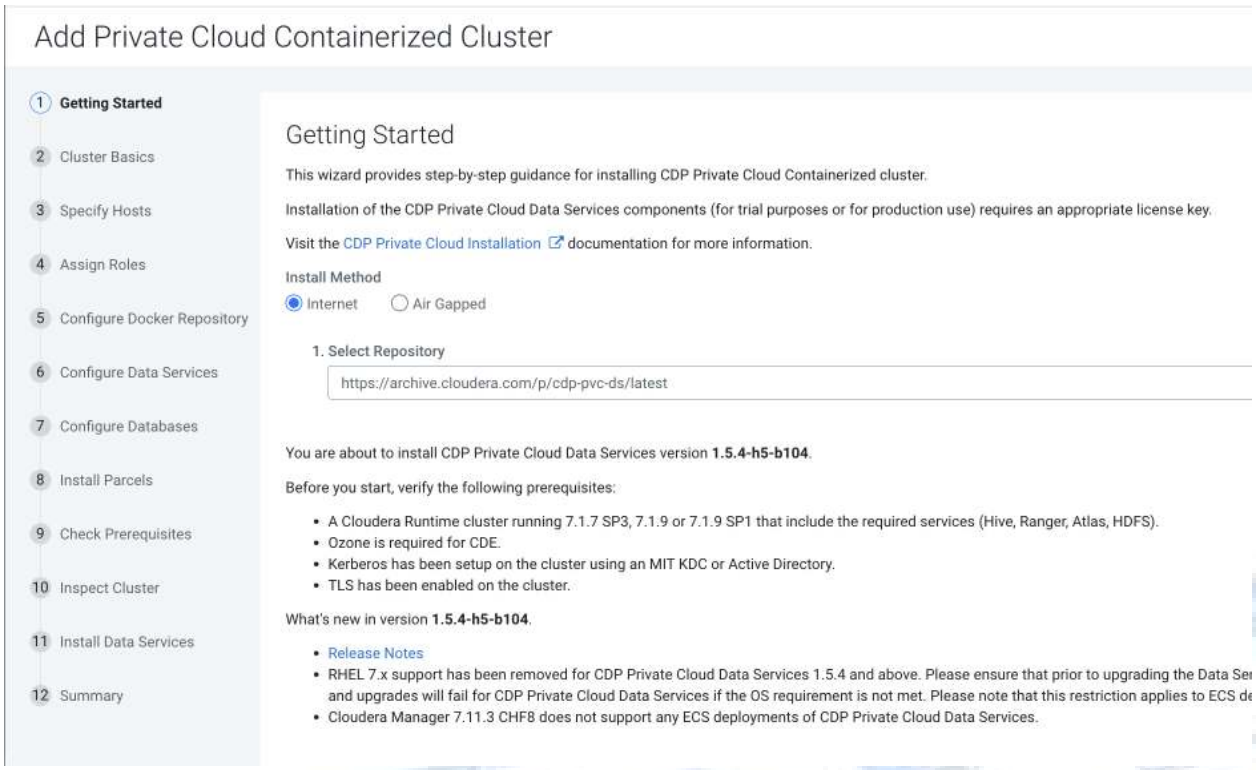
Note: Alternatively, you can also click ***(+) Add > Add Cluster*** at the ***top right*** in ***Cloudera Manager***, then select ***Private Cloud Containerized Cluster*** as the ***cluster type***, then click **Continue**.



Step. Getting started

3. On the getting started page of the installation wizard, select either ***Internet*** or ***Air Gapped*** as the Install Method. ***(We are going here with Internet Installation method only)***

4. If you select the ***Internet Install*** Method option on the ***Getting Started*** page, images are copied over the internet from the Cloudera repository. For this deployment, we will select the ***Internet*** as the install method. ***Select Repository.*** (To use a custom repository link provided to you by Cloudera, click ***Custom Repository, instructions for this are mentioned in later steps***):



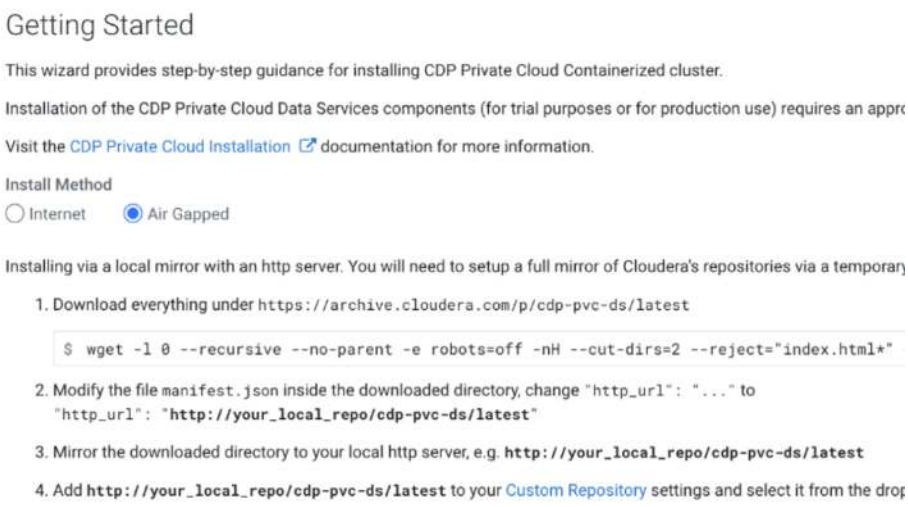
Note: Verify if the version shown below is the same as the version that we are willing to install i.e. ***1.5.5-h5*** for current setup.

If not (as shown in below screenshot), then we need to add-up an additional custom URL and configure that to use our specific version.

Add the additional URL by **(+)** and click on ***Save Changes.***

5. When you select the ***Air Gapped*** install option, extra steps are displayed. Follow these steps on the ***cldr-mngr*** node (our bits server), to download and mirror the Cloudera archive URL using a local HTTP server:

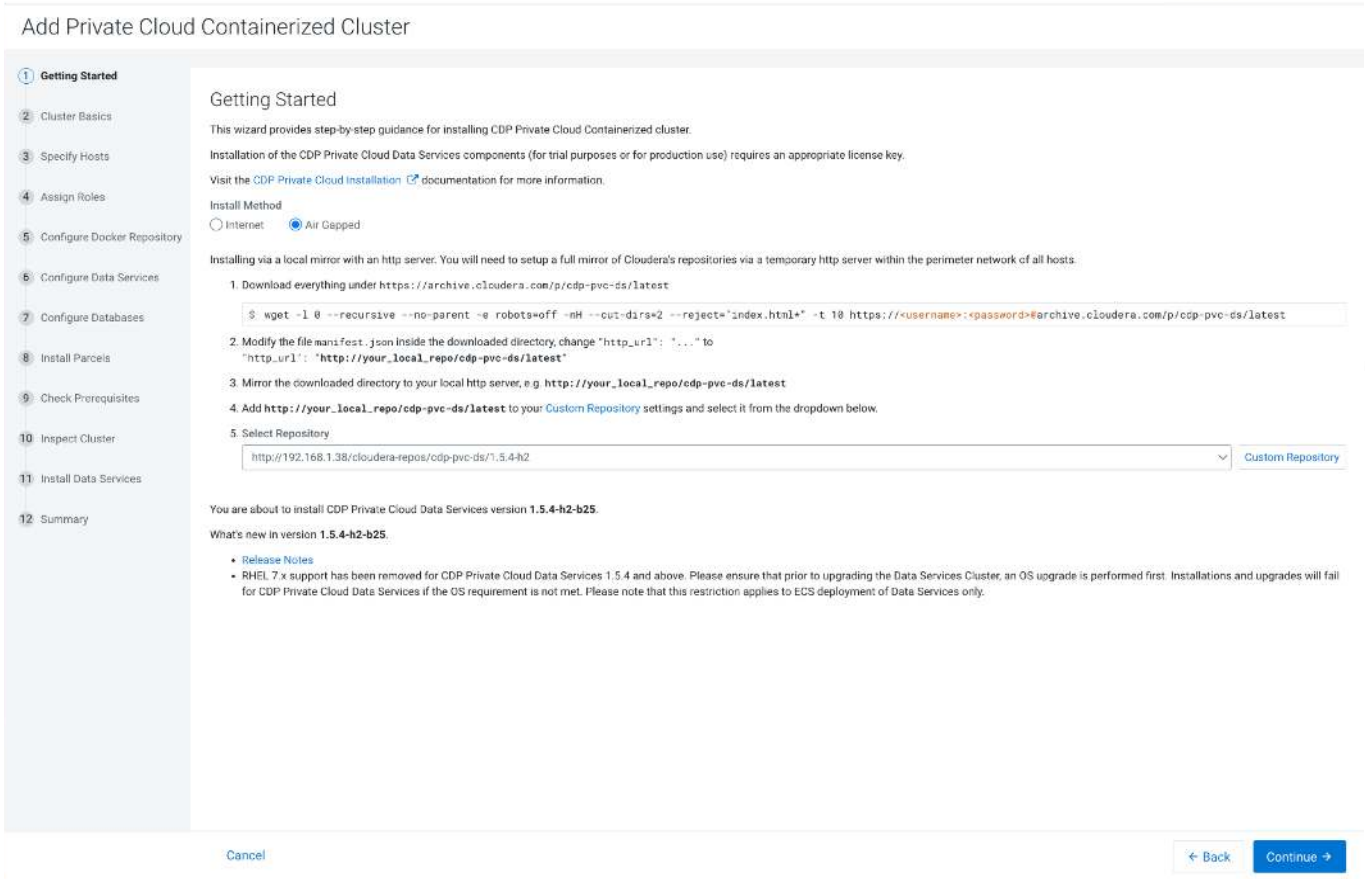
6. (For installing via a local mirror with an http server. You will need to set up a full mirror of Cloudera's repositories via a temporary HTTP server within the perimeter network of all hosts.): **(Skip this step, as we will choose Internet method in next steps to Install)**




```
[root@cldr-mngr ~]# mkdir -p /var/www/html/cloudera-repos/cdp-pvc-ds/
[root@cldr-mngr ~]# cd /var/www/html/cloudera-repos/cdp-pvc-ds/

##### Download everything under https://archive.cloudera.com/p/cdp-pvc-ds/latest/ to your local
httpserver, e.g. http://your_local_repo/cdp-pvc-ds/latest/ using the below command

[root@cldr-mngr cdp-pvc-ds]# wget -l 0 --recursive --no-parent -e robots=off -nH --cut-dirs=2
--reject="index.html*" -t 10 https://<username>:<password>@archive.cloudera.com/p/cdp-pvc-ds/latest/
[root@cldr-mngr cdp-pvc-ds]#

[root@cldr-mngr cdp-pvc-ds]# ls -lt 1.5.5-h2/
total 116300
-rw-r--r-- 1 root root    284820 Mar 15 10:13 manifest.json
-rw-r--r-- 1 root root 118747085 Mar 15 10:13 cdp-private-1.5.5-h2-b10.tgz
drwxr-xr-x 2 root root      4096 Mar 15 10:13 parcels
drwxr-xr-x 2 root root      4096 Mar 15 10:12 manifests
drwxr-xr-x 2 root root     32768 Mar 15 10:12 images
[root@cldr-mngr ~]#

##### Modify the manifest.json file inside the downloaded directory. Change "http_url": "..." to
"http_url": "http://your_local_repo/cloudera-repos/cdp-pvc-ds/1.5.5-h2/"

[root@cldr-mngr ~]# vi manifest.json
"http_url": "http://192.168.1.38/cloudera-repos/cdp-pvc-ds/1.5.5-h2/"
[root@cldr-mngr ~]#
```

7. Click ***Custom Repository***. Add ***http://your_local_repo/cloudera-repos/cdp-pvc-ds/1.5.5-h2/*** as a custom repository. Click on ***Save Changes***. **(Skip this step, as we have chosen Internet method steps to Install)**

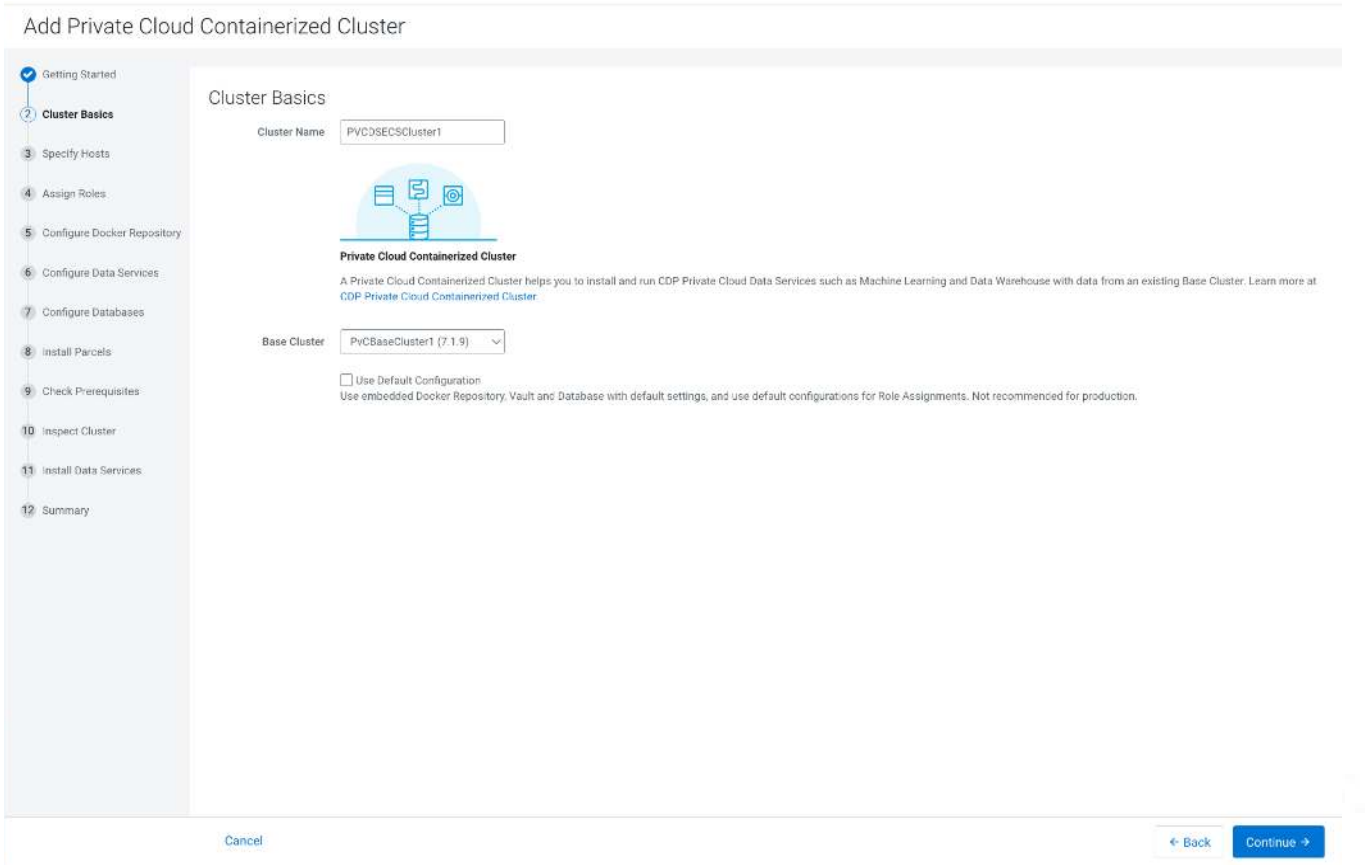
8. Click the ***Select Repository*** drop-down and select ***http://your_local_repo/cloudera-repos/cdp-pvc-ds/1.5.5-h2/*** **(Skip this step, as we will chosen Internet method steps to Install)**



9. Click ***Continue***.

Step. Cluster Basics

10. On the ***Cluster Basics- Add Private Cloud Containerized Cluster*** page, enter the ***Cluster Name*** for the ***Cloudera on premises Data Services (ECS) cluster*** that you want to create in the ***Cluster Name*** field. From the ***Base Cluster*** drop-down list, select the ***Cloudera on premises Base Cluster*** (which is created earlier i.e. ***PvCBaseCluster1***), that has the storage and SDX services that you want this new Cloudera on premises Data Services instance to connect with. Click ***Continue***.



Step. Specify Hosts

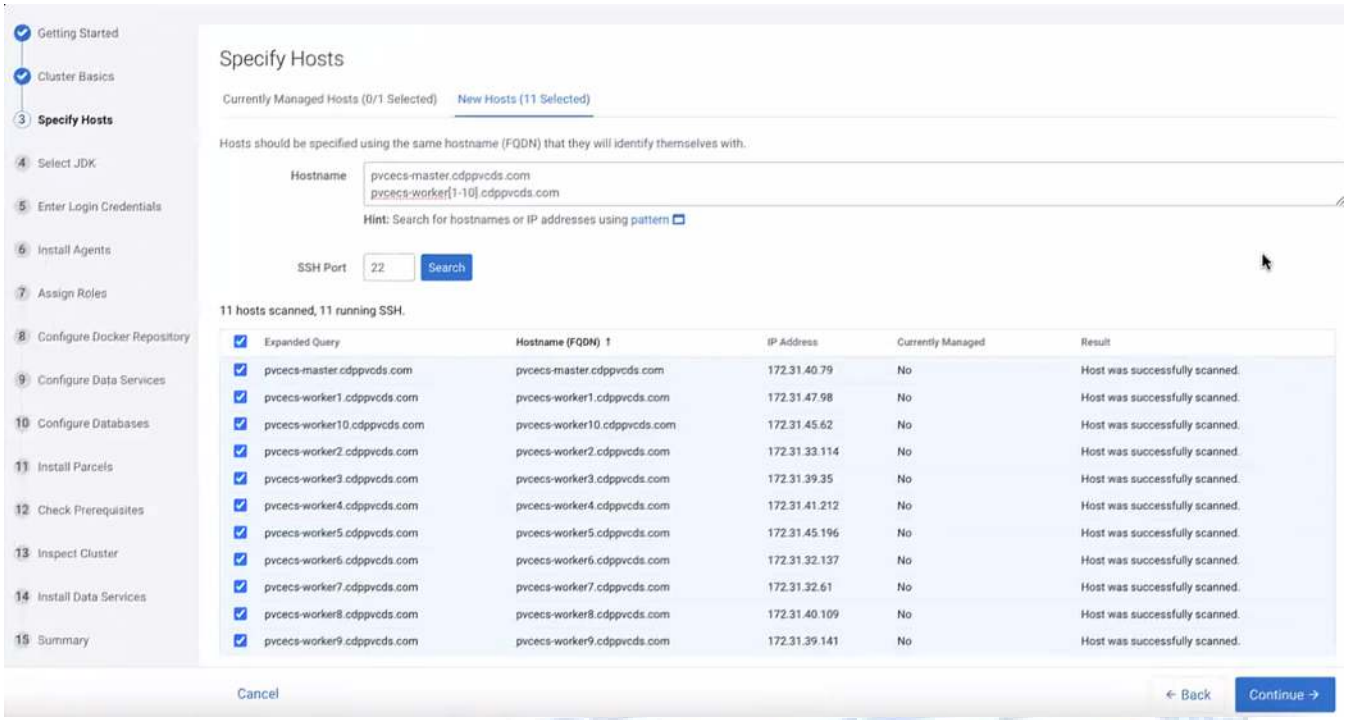
11. On the ***Specify Hosts*** page, hosts that have already been added to Cloudera Manager are listed on the ***Currently Managed Hosts*** tab. If the intended ECS nodes are not added previously (in case of a fresh setup) and you can't see them listed under ***Currently Managed Hosts*** tab, Click on ***New Hosts*** tab.

Note: To specify the hosts that are part of the cluster, enter Fully Qualified Domain Names (FQDNs)/hostnames or IP addresses in the Hostname field, or provide a list of search patterns/ IP address range of available matching ECS hosts. Host names must be in lowercase. If you use uppercase letters in any host name, the cluster services will not start after enabling Kerberos.

Note: You can Provide host pattern pvcbase-master, pvcbase-worker[1-5] or pvcbase-worker[1-5].cldrsetup.local etc separated with a new line and Click on ***Search***. Cloudera Manager will "discover" the hosts based on matching the pattern provided by you to add in the cluster. Verify that all desired nodes have been found and ***selected for installation***. Verify host entries, ***deselect*** any that you do not want to install services on. Select and/or deselect one or more of these hosts to add to the ECS cluster and click ***Continue***.

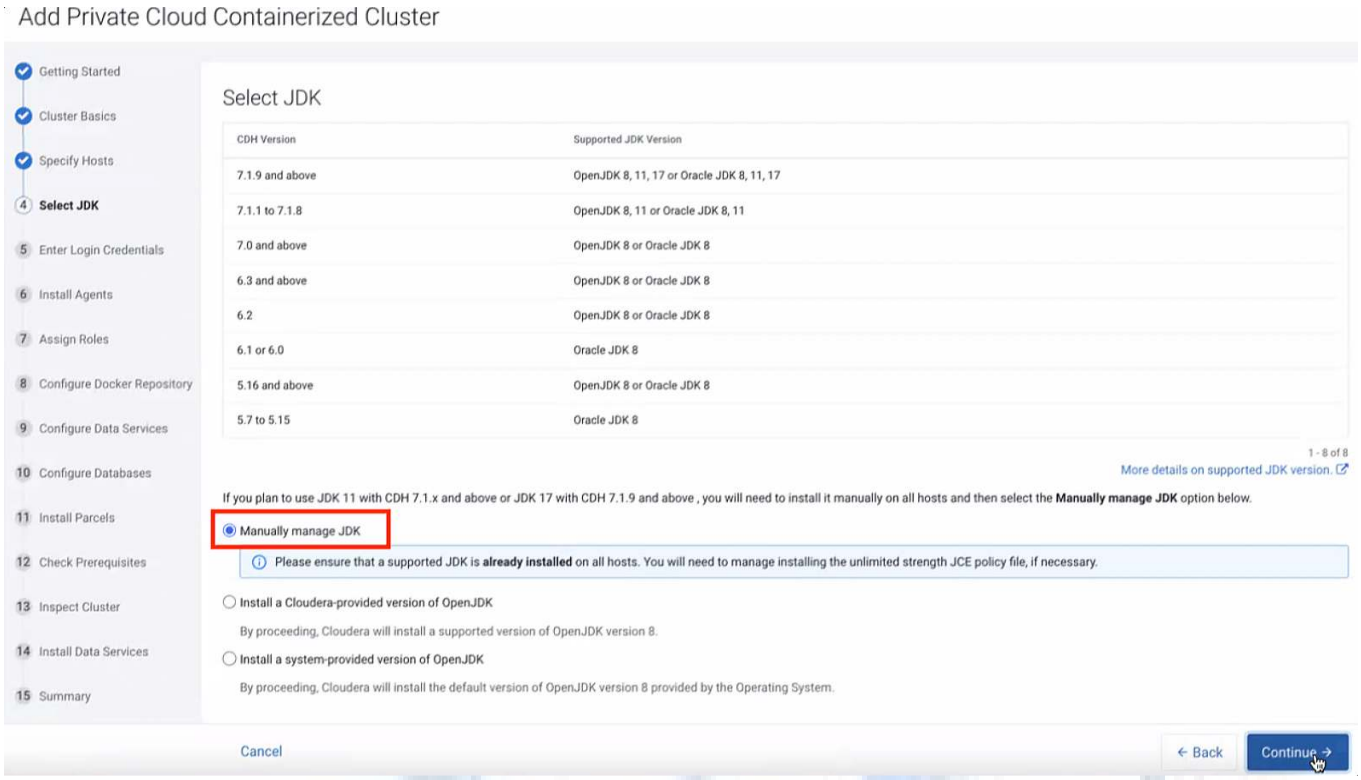
```
pvcecs-master.cldrsetup.local
pvcecs-worker[1-10].cldrsetup.local
```

Note: Click the pattern link under the Hostname box to display more information about allowed **FQDN** patterns.



Step. Select JDK

12. On the ***Select JDK*** page, select any one from the below options:
    - **Manually manage JDK (Select this option)** (manual installation of JDK11/17 with CDH 7.3.1+).
    - Install a Cloudera-provided version of OpenJDK
    - Install a system-provided version of OpenJDK

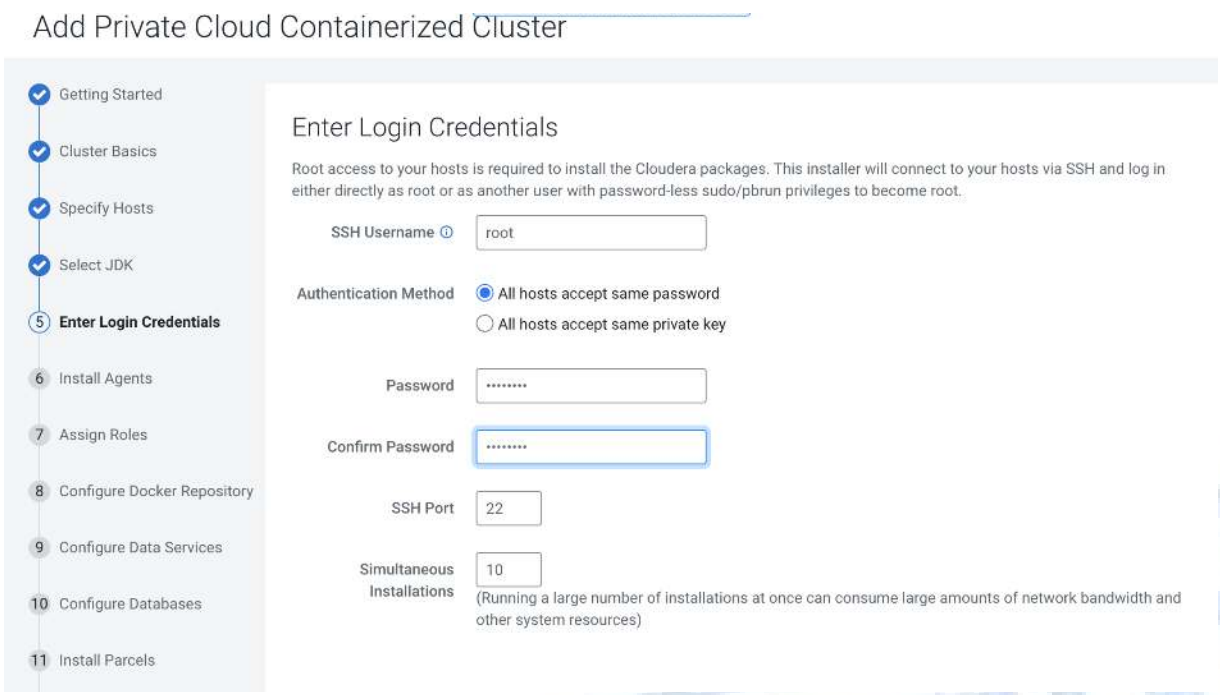


Step. Enter Login Credentials

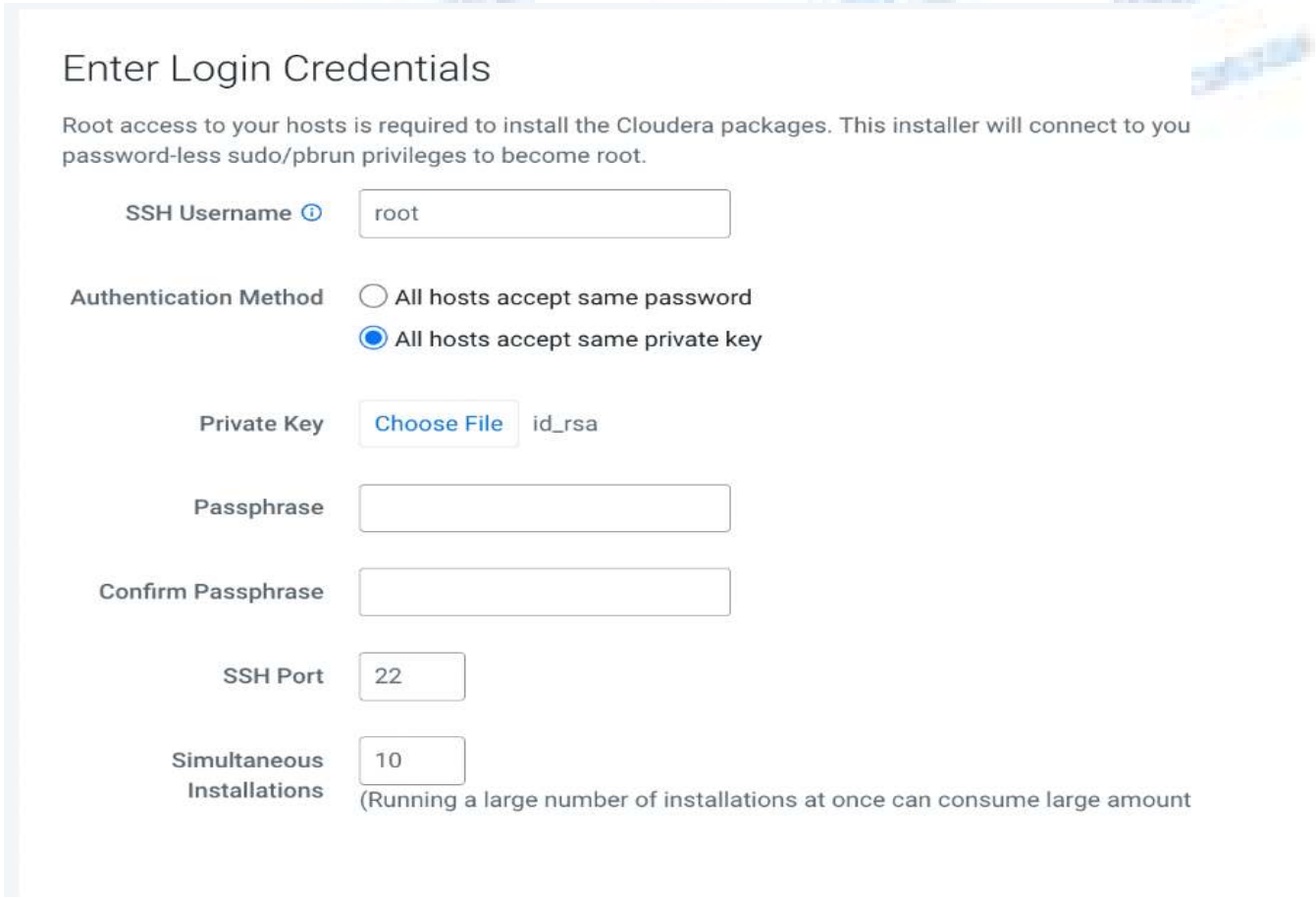
13. On the ***Enter Login Credentials*** page, '***All hosts accept the same password'*** is selected by default. Enter the user name in the SSH Username box, and type in and confirm the password. You can also select the ***All hosts accept the same private key*** option and provide the Private Key generated in previous steps and passphrase (If applicable).

Enter the values for the parameters as shown below. (**We will be using the private key approach**, you can use password option as well, both options should considerably work)

Component	Value
Enable TLS for	All existing and future clusters
SSH username	***root***
Authentication method	***All hosts accept same private key*** / All hosts accept same password
Private Key (If using Key approach)	Choose the private key created and downloaded in earlier section
Password (If using Password approach)	Enter VM's root users' password
Confirm Password	Enter VM's root users' password (again)



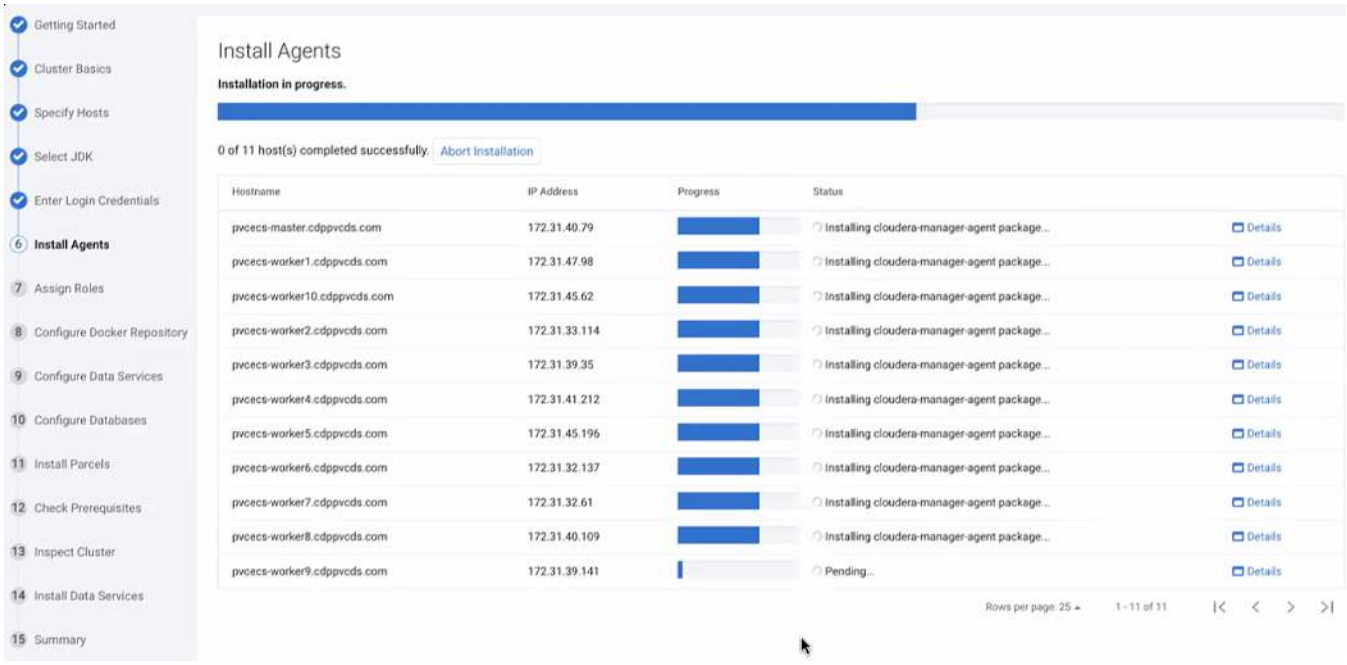
a. Screenshot for using the Password based authentication method.

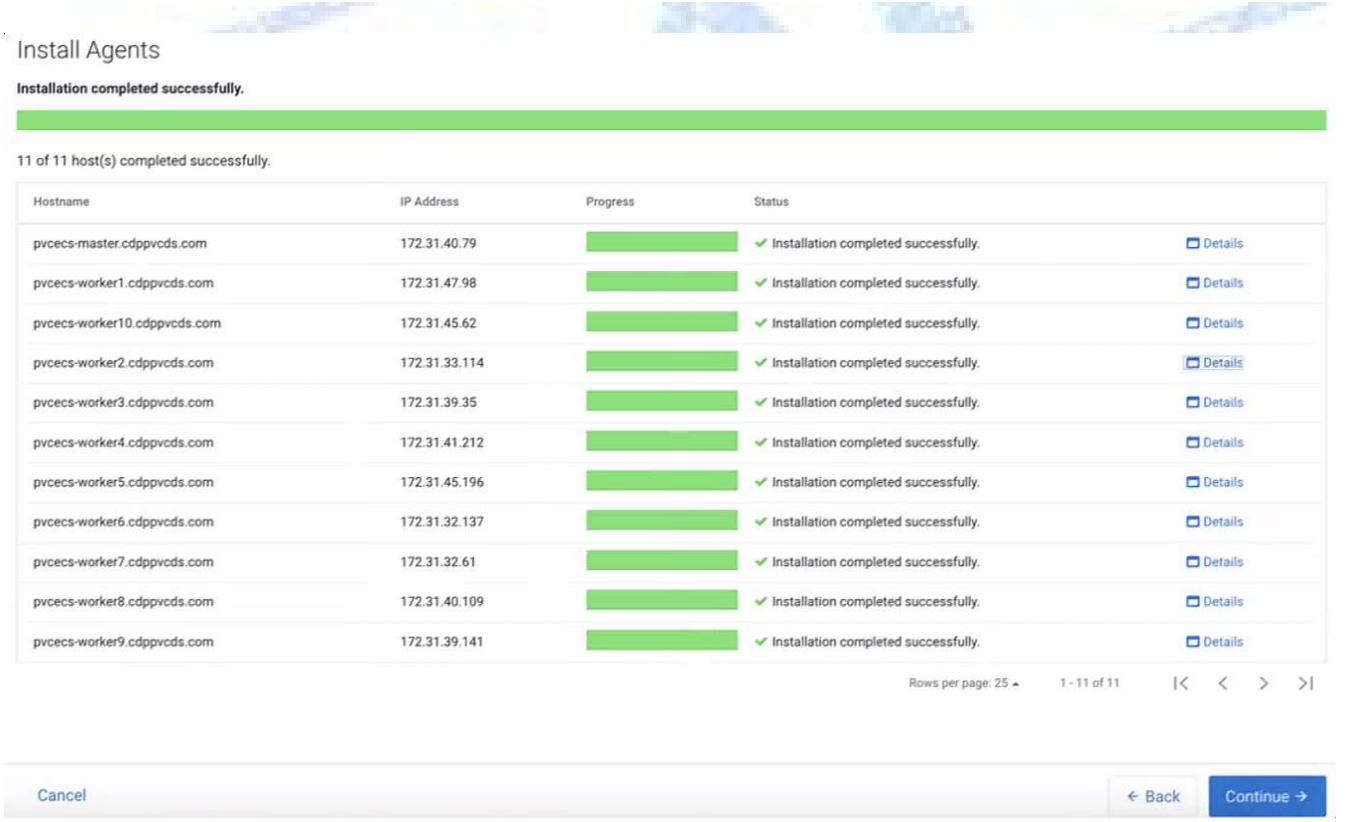


b. Screenshot for using the Private Key based authentication method.

Step. Install Agents

14. The ***Install Agents*** page appears and displays a progress indicator showing the agent packages getting installed. Click on ***Continue*** after successful agent installation on hosts to be added in Cloudera Manager.





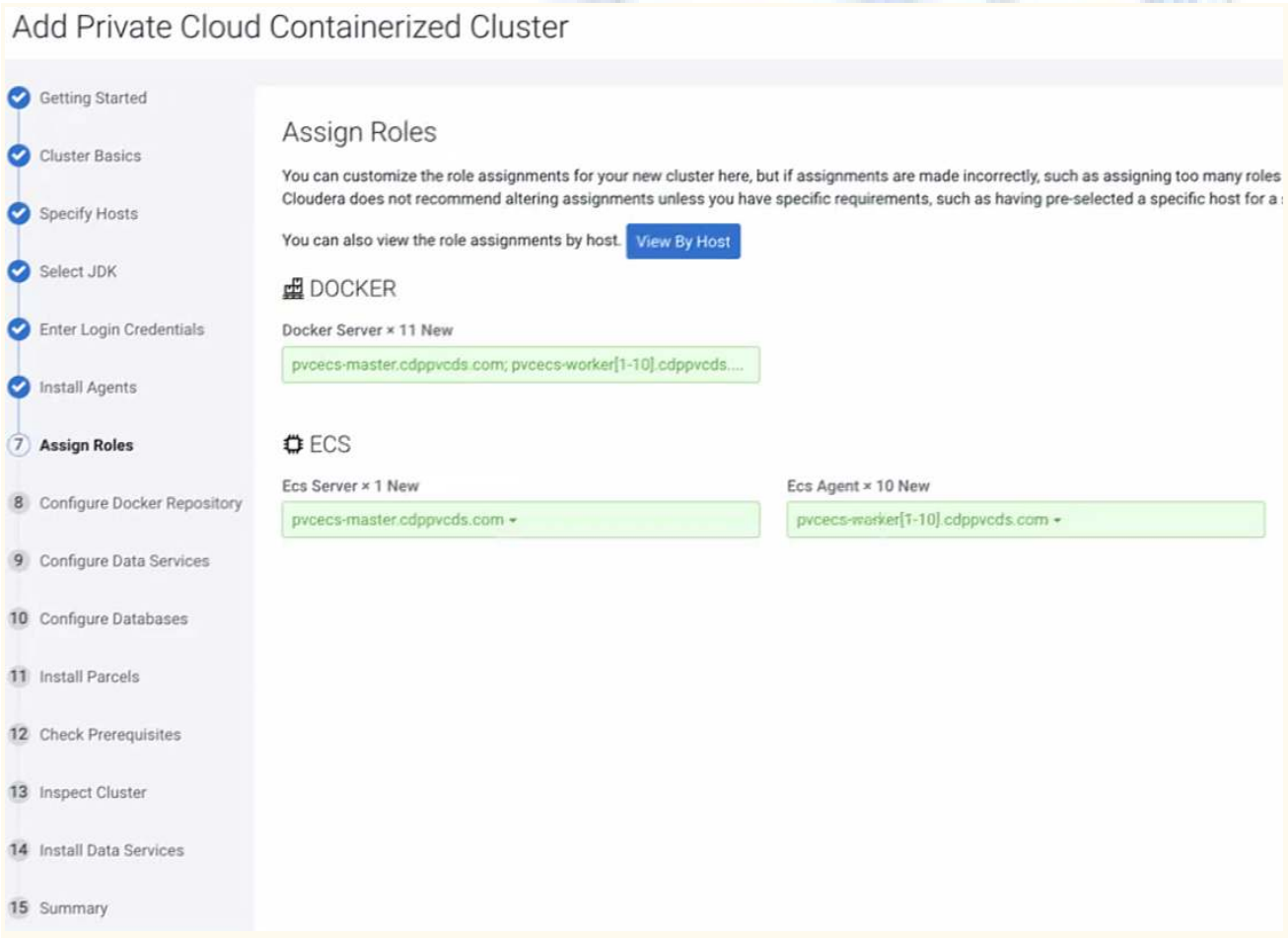
Step. Assign Roles

15. Next on the ***Assign Roles*** page, ensure that the roles assignment for your new ***Cloudera on premises Containerized cluster*** is as follows. You can customize the role assignment for your cluster. But, Cloudera does not recommend altering assignments unless you have specific requirements such as having selected a specific host for a specific role.

Note: Single node ECS installation is supported, but is only intended to enable CDSW to CAI migration. If you are installing ECS on a single node, only the Docker and ECS Server roles are assigned. The ECS Agent role is not required for single node installation.

Note: With 1 mgmt node and 12 worker node for CDP Data Services ECS cluster we select host role assignment as:

Role	ECS Host
Docker Server	All ECS Hosts (i.e., ECS master and worker nodes) pvcecs-master, pvcecs-worker[1-11]
ECS Server	ECS Master Nodes only (pvcecs-master)
ECS Agent	ECS Worker Nodes only (pvcecs-worker[1-11])



16. Click **Continue**.

Step. Configure Docker Repository

17. On the ***Configure Docker Repository*** page, select ***Cloudera default Docker Repository*** or (***Use an Embedded Docker Repository*** option. Then select Default in the below section. There are several options for configuring a Docker Repository. For more information about these options, see Docker repository access.)

The following ports must be opened and allowed no matter which Docker repository option you choose.

- Ports required for Cloudera Manager/Cloudera Manager agent (port 5000 is required for CAI):

Protocol	Port
TCP	7180-7192
TCP	19001
TCP	5000
TCP	9000

- Inbound rules for ECS Server nodes (Kubernetes/RKE2):

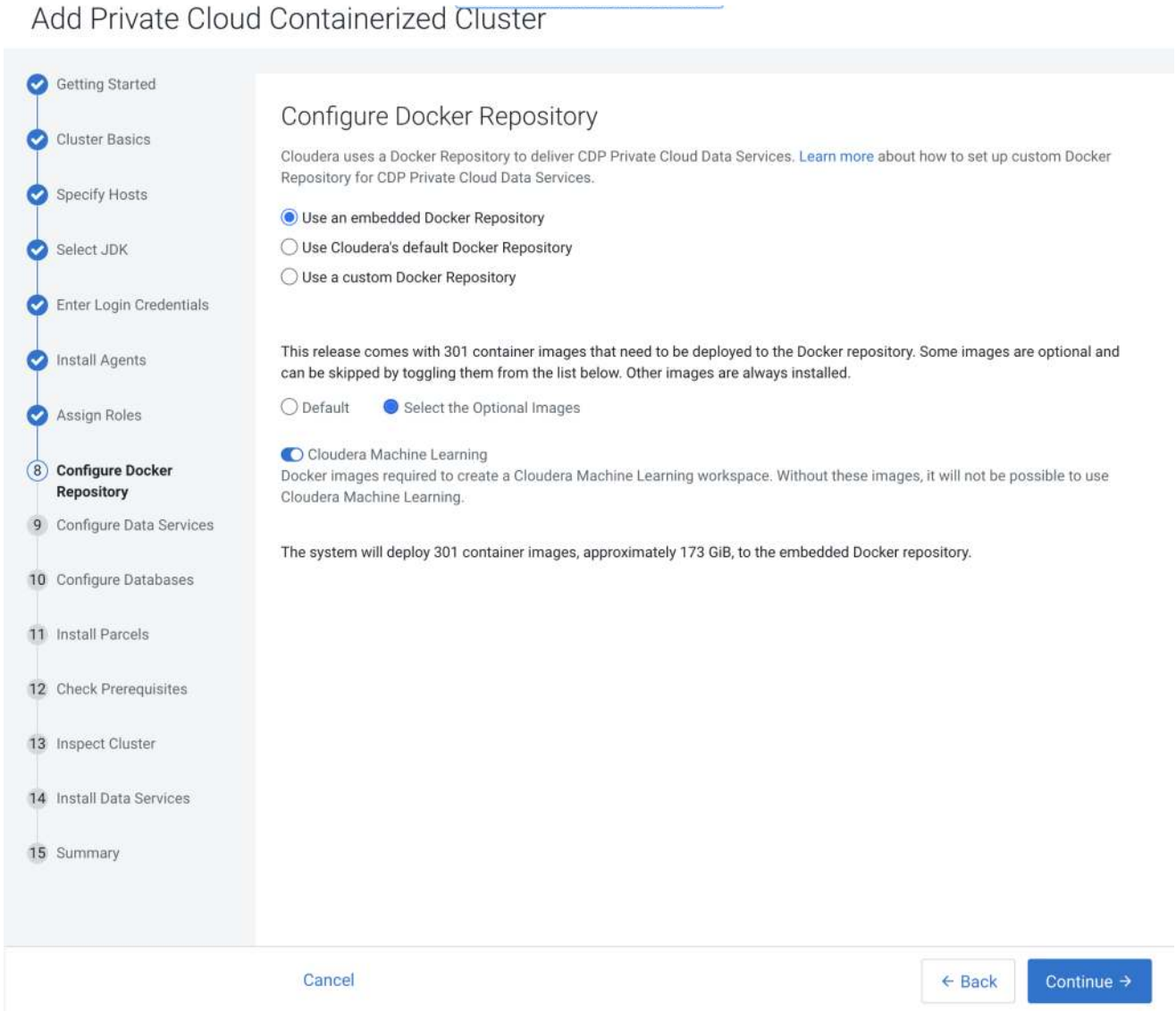
Protocol	Port
TCP	9345
TCP	6443
UDP	8472
TCP	10250
TCP	2379
TCP	2380
TCP	30000-32767

- Inbound Rules for the ECS Agent (Kubernetes/RKE2):

Protocol	Port
UDP	4789

Embedded Docker Repository:

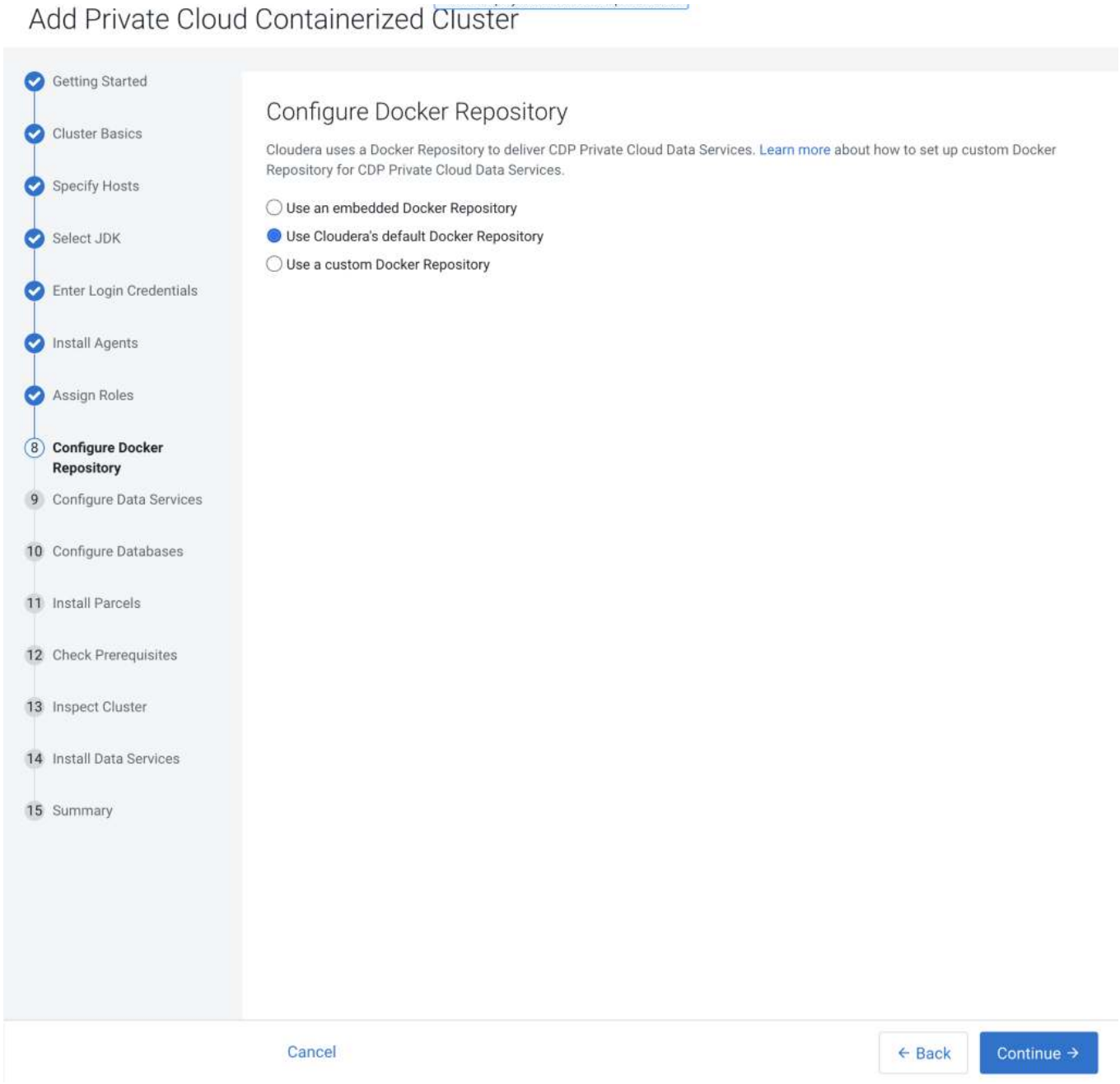
Proceed with default selection to deploy all of the default Docker images to the repository, or select ***Select the Optional Images*** to choose which images to deploy. If you will be deploying Cloudera AI (CAI) a.k.a. Cloudera Machine Learning (CAI), toggle the ***Cloudera Machine Learning*** switch on to copy the images for CAI.



a. Screenshot for Use an embedded Docker Repository Option

Cloudera default Docker Repository: **(We will setup this way)**

This option requires that cluster hosts have access to the internet and you have selected Internet as the install method.

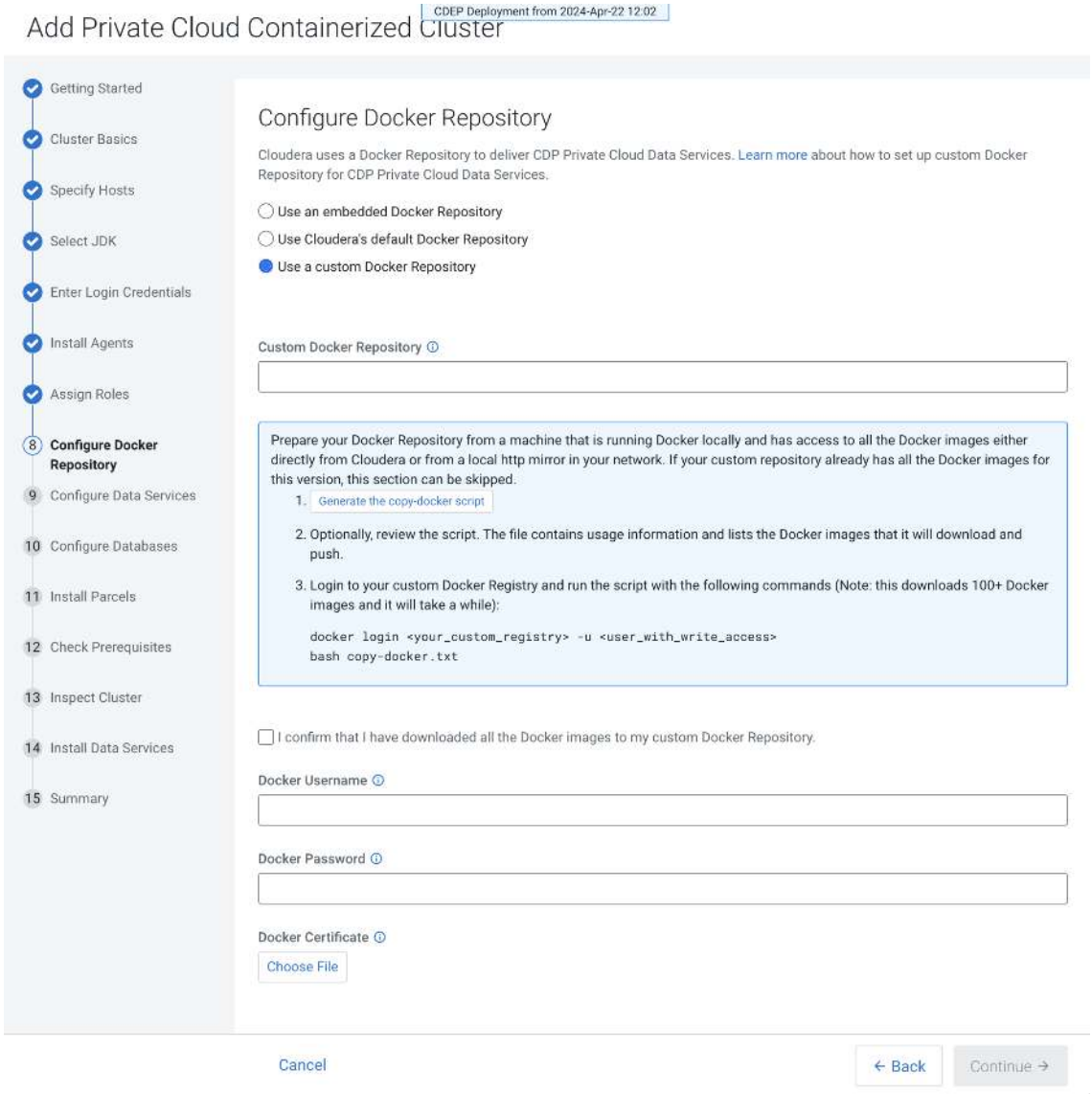


b. Screenshot for Use Cloudera's Docker Repository Option

Custom Docker Repository:

This option requires that you set up a Docker Repository in your environment and that all cluster hosts have connectivity to the repository.

Note: If you are installing ECS on a single node, you should select the Use a Custom Docker Repository option. Single node ECS installation is supported, but is only intended to enable CDSW to CAI migration.



c. Screenshot for Use Cloudera's Docker Repository Option

You must enter the following options:

Option	Value
Custom Docker Repository:	Enter the URL for your Docker Repository
Docker Username:	Enter the username for the Docker Repository
Docker Password:	Enter the password for the Docker Repository

Important: Do not use the $ character for this password.

Docker Certificate – Click the ***Choose File*** button to upload a TLS certificate to secure communications with the Docker Repository.
Click the ***Generate the copy-docker script*** button to generate and download a script that copies the Docker images from Cloudera, or (for air-gapped installation) from a local http mirror in your network.
Run the script from a machine that is running Docker locally and has access to the Docker images using the following commands:

```
docker login [***URL for Docker Repository***] -u [***username of user with write access***]
bash copy-docker.txt
```

The copying operation may take 4 - 5 hours.

Note: Embedded Repository can be a single point of failure. If the node that runs the Docker Repository fails or becomes unavailable, some cluster functionalities might become unavailable. Moving the Docker Repository to another node is a complex process and will require engaging Cloudera Professional Services.

Note: Cloudera Repository option is best suited for proof-of-concept, non-production deployments or deployments that do not have security requirements that disallow internet access. This option requires that cluster hosts have access to the internet, and an installation method selected as Internet.

Step. Configure Data Services

18. On the ***Configure Data Services*** page, modify configuration as appropriate and modify the storage related parameters. Edit Application domain to match “app.example.com”. For example in this solution we configure AD Domain Services with “CLDRSETUP.LOCAL” as domain name. Created a wildcard entry “*.apps.cldrsetup.local”. Click ***Continue***.

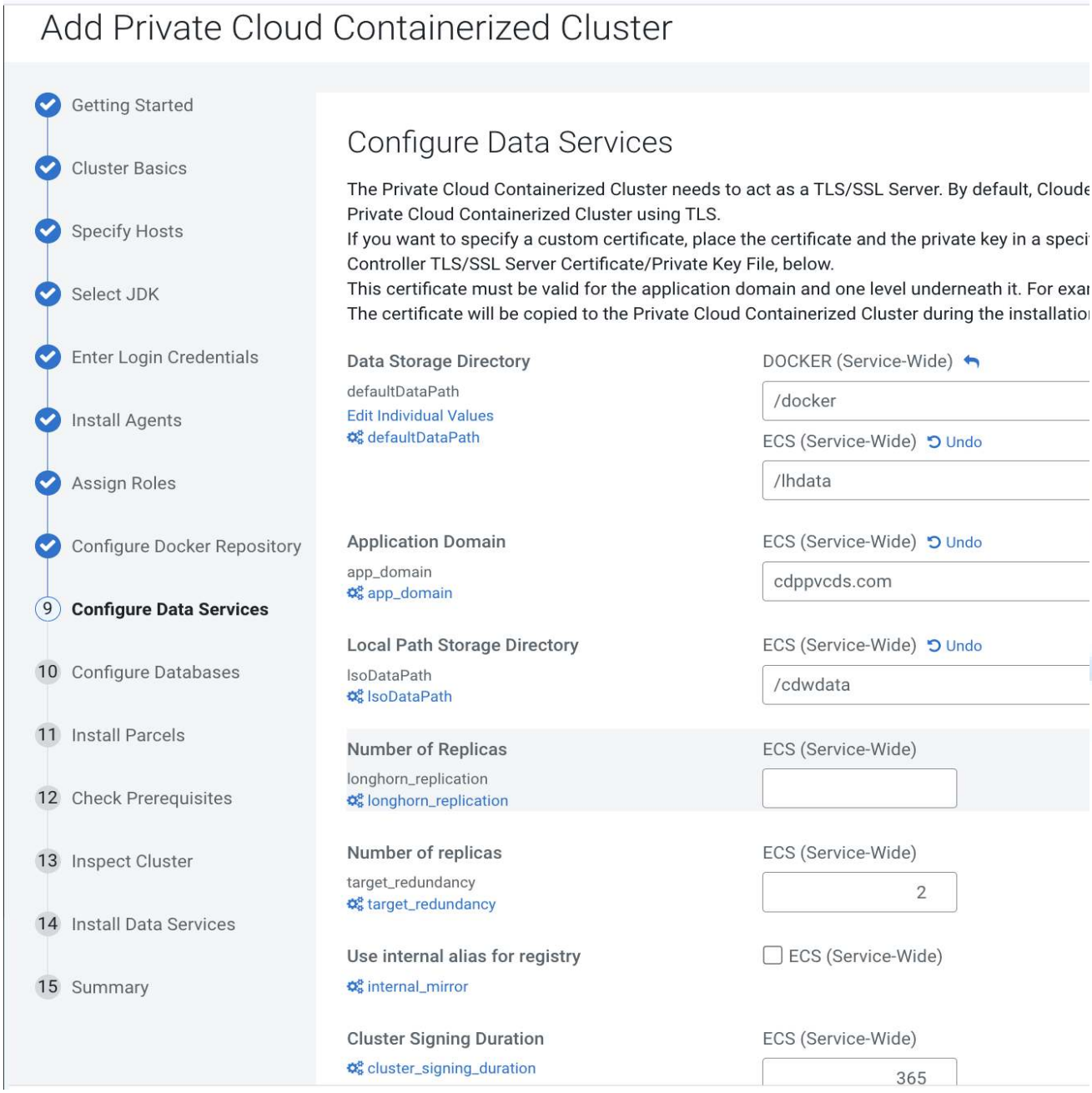
Role	ECS Host
Data Storage Directory:	/docker
ECS (Service Wide):	/lhdata
Application Domain:	cldrsetup.local
Local Path Storage Directory:	/cdwdata

On the ***Configure Data Services*** page, you can modify configuration settings such as the data storage directory, number of replicas, and so on. If you want to specify a custom certificate, place the certificate and the private key in a specific location on the Cloudera Manager server host and specify the paths in the input boxes labeled as Ingress Controller TLS/SSL Server Certificate/Private Key File below. This certificate will be copied to the Control Plane during the installation process.

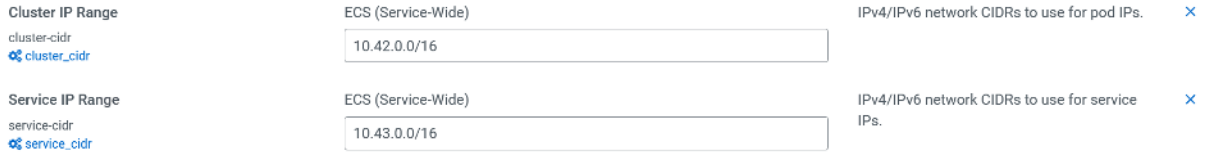
Note: The "Ingress Controller TLS/SSL Server Certificate File (PEM Format)" must only contain -----BEGIN CERTIFICATE----- through -----END CERTIFICATE----- (inclusive) for the server and CA certs. It cannot include any preamble text and, and must not include a private key.
The "Ingress Controller TLS/SSL Server Private Key File (PEM Format)" must only contain the unencrypted key, and only the header through the footer, with no preamble text.

Both of these files must be readable by the "cloudera-scm" account.

For information on the required entries that must be present in DNS and TLS certificates when not using wildcards, refer to 'No Wildcard DNS/TLS Setup'



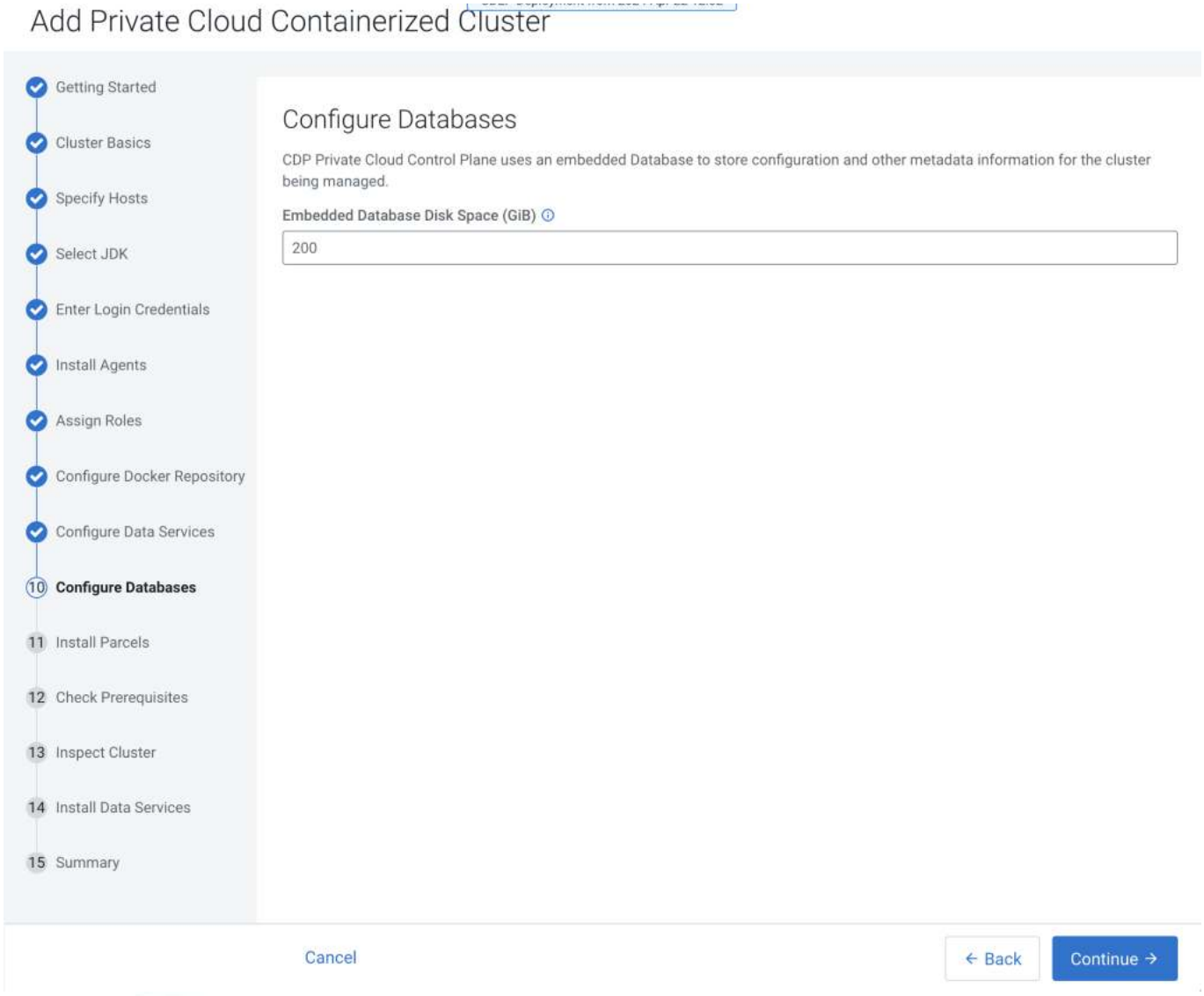
Note: Please review the range of cluster IP and service IP as part of the ECS installation. It might conflict existing network configuration. Please adjust the range of IPs to be configured. Consult with the network team to avoid potential conflict.



19. Click *Continue*.

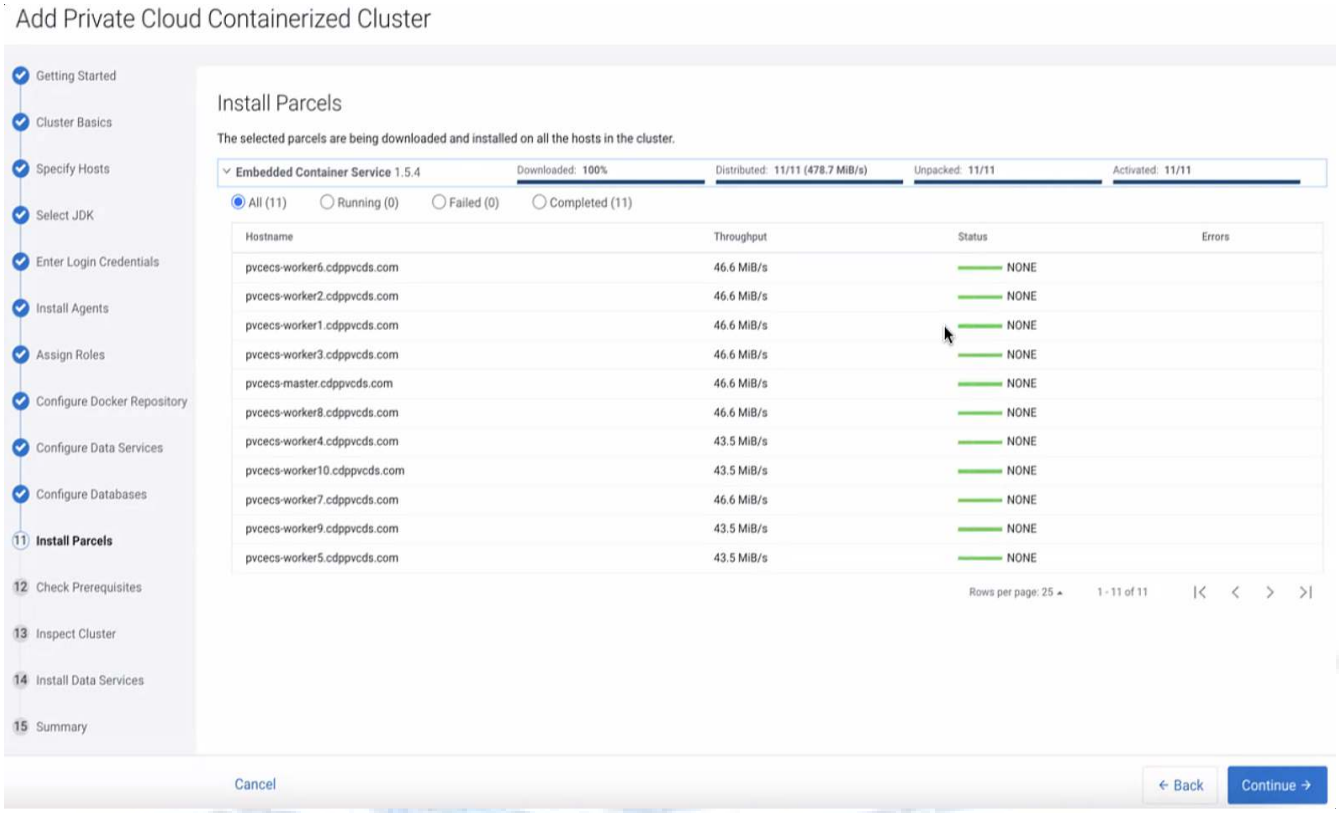
Step. Configure Databases

20. On the ***Configure Databases*** page, edit size for the ***Embedded Database Disk Space***. Click ***Continue***.



Step. Install Parcels

21. On the ***Install Parcels*** page, the selected parcel is downloaded to the Cloudera Manager server host, distributed, unpacked, and activated on the ECS cluster hosts. Click ***Continue***.



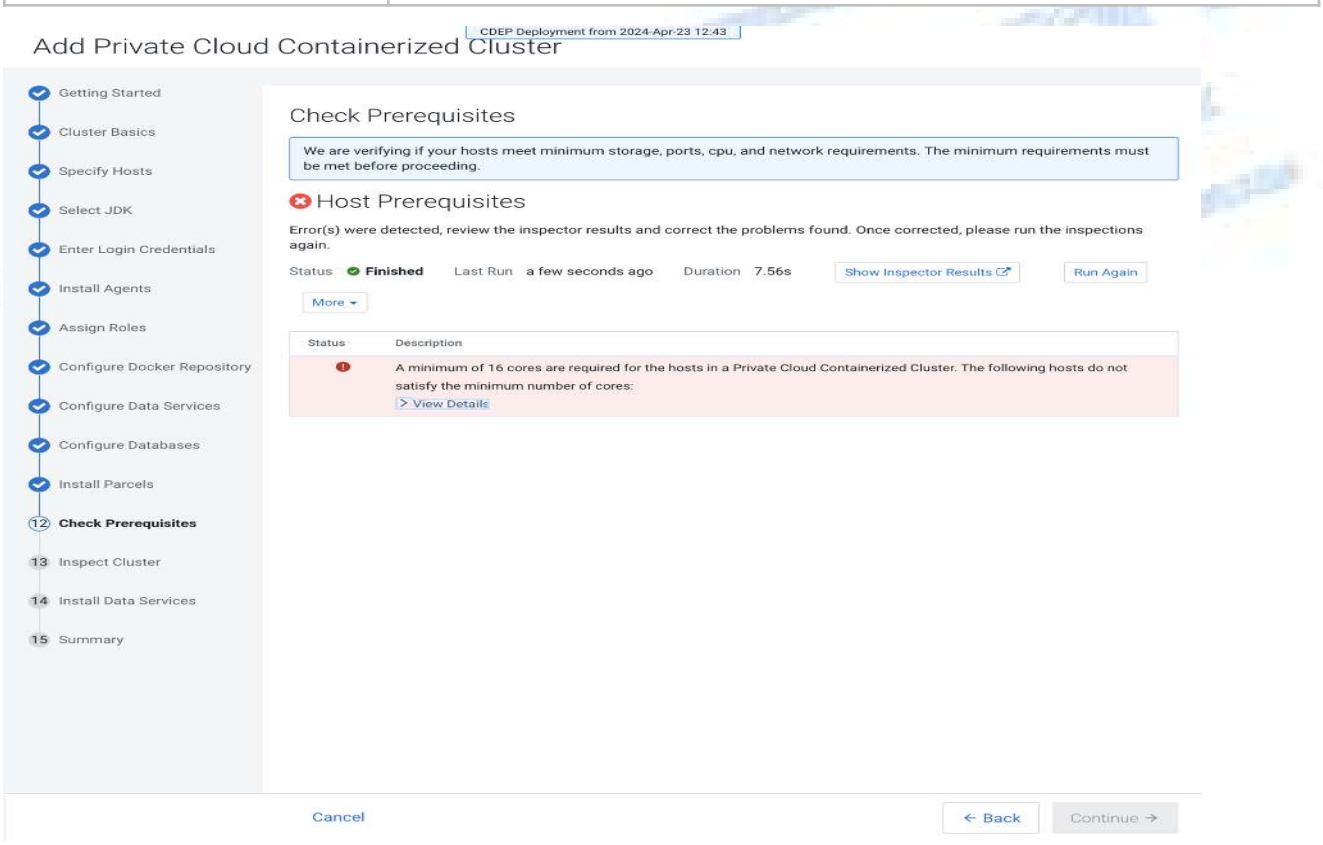
Step. Check Prerequisites

22. If the hosts do not meet the prerequisites, the ***Check Prerequisites*** page displays the applicable issues. Correct the issues, then click ***Run Again***. After all of the issues have been resolved, click ***Continue***. Prerequisites checks are included in the new ECS release version 1.5.5.

The following prerequisites are checked:

Host Prerequisite Inspection	Validation
StorageInspection:	Checks for a minimum of 300GiB space in the /var/lib and docker data directories. Checks if /var/lib/longhorn or its parent directories are symlinked. If they are, this inspection will fail.
CPUInspection:	Checks to make sure the hosts have 16 virtual cores.
PortsInspection:	Checks for the availability of ports 443 and 80.

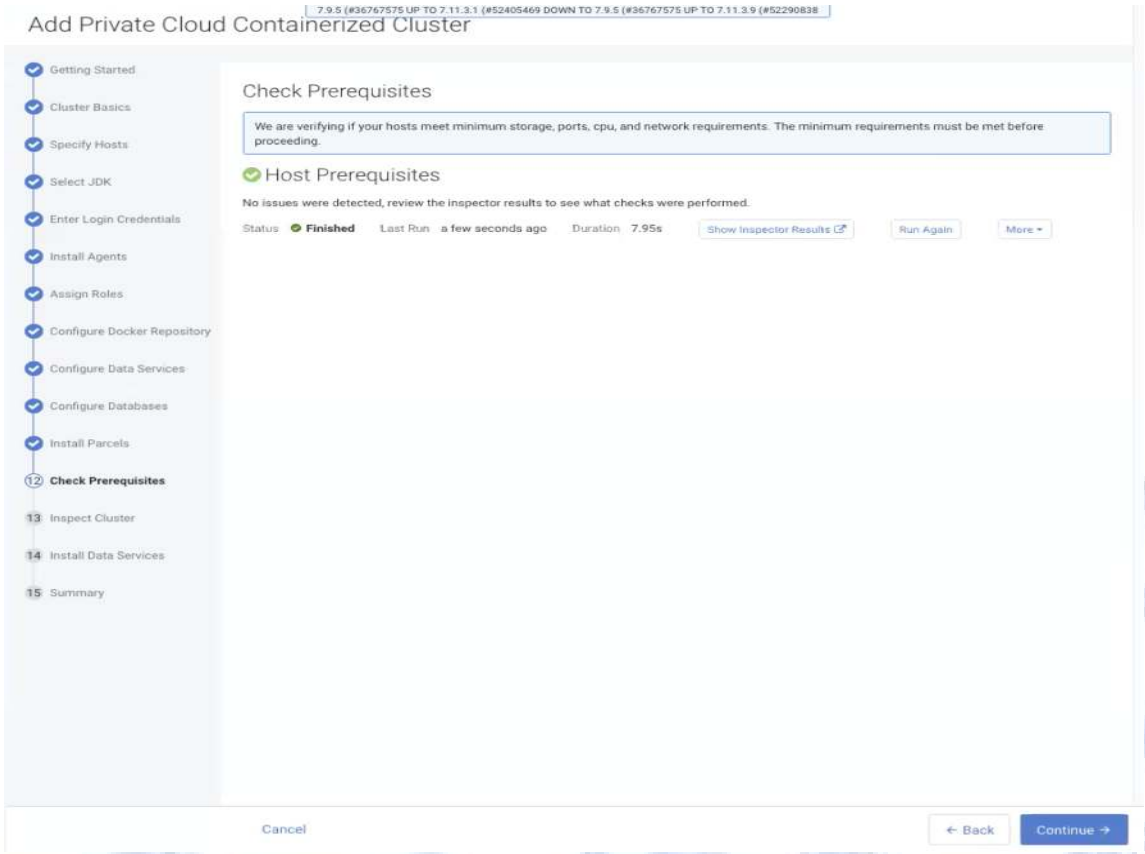
Host Prerequisite Inspection	Validation
EcsHostDnsInspection:	Checks to make sure there are less than 3 nameserver entries in the /etc/resolv.conf file, and checks the connections to the Cloudera Manager cluster and the CDP console. It also checks to see if vault.localhost.localdomain's ping can be resolved. If not, it is likely that the host /etc/nsswitch.conf file is misconfigured. If this inspection fails: ● Check the /etc/resolv.conf and /etc/nsswitch.conf files and ensure that /etc/resolv.conf does not contain 3 or more nameservers, and that /etc/nsswitch.conf does not contain myhostname under the hosts field. ● Check to see if the connections were resolved correctly. If connection to the CDP console fails, check to see if your DNS wildcard is configured properly.
VersionInspection:	Checks that Java is installed and consistent among all ECS hosts.
IPTablesInspection:	Checks that if the iptables command exists, rules are cleared. If the iptables command does not exist, iptables gets installed during FirstRun so this inspection passes. If iptables are installed and the rules are not cleared, this inspection will fail.
EcsCleanUpHostInspection:	Checks to make sure that the /var/lib/rancher and docker data directories do not contain any files.



Note: If the prerequisite check fails due to any reason or you need to skip this step, try the workaround by right-clicking on the screen, select ***Inspect***, go to the ***Console*** tab, and run the below command:

```
document.querySelector('.btn.next').removeAttribute('disabled');
```

This will enable the **Continue** button.

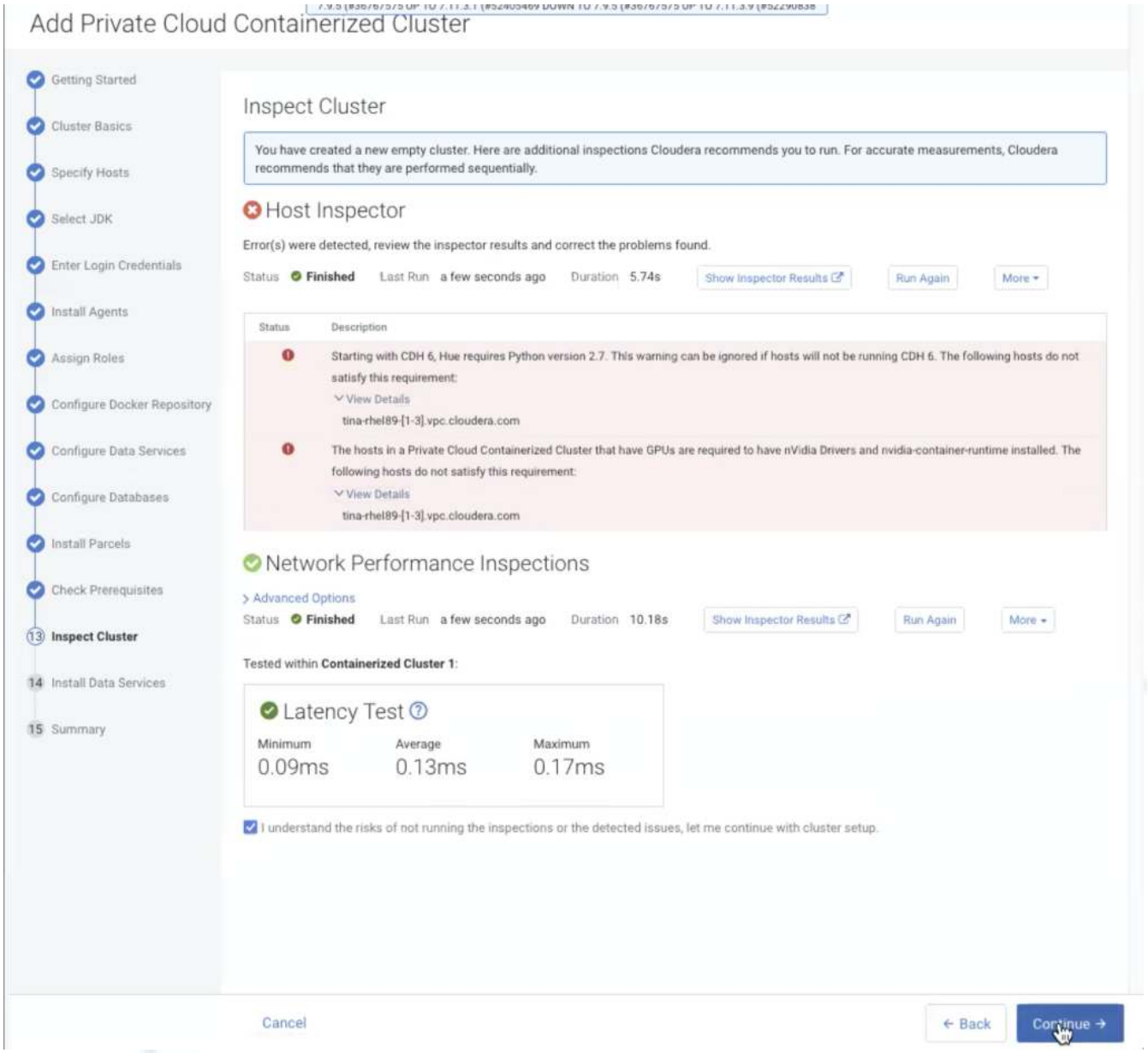


Step. Inspect Cluster

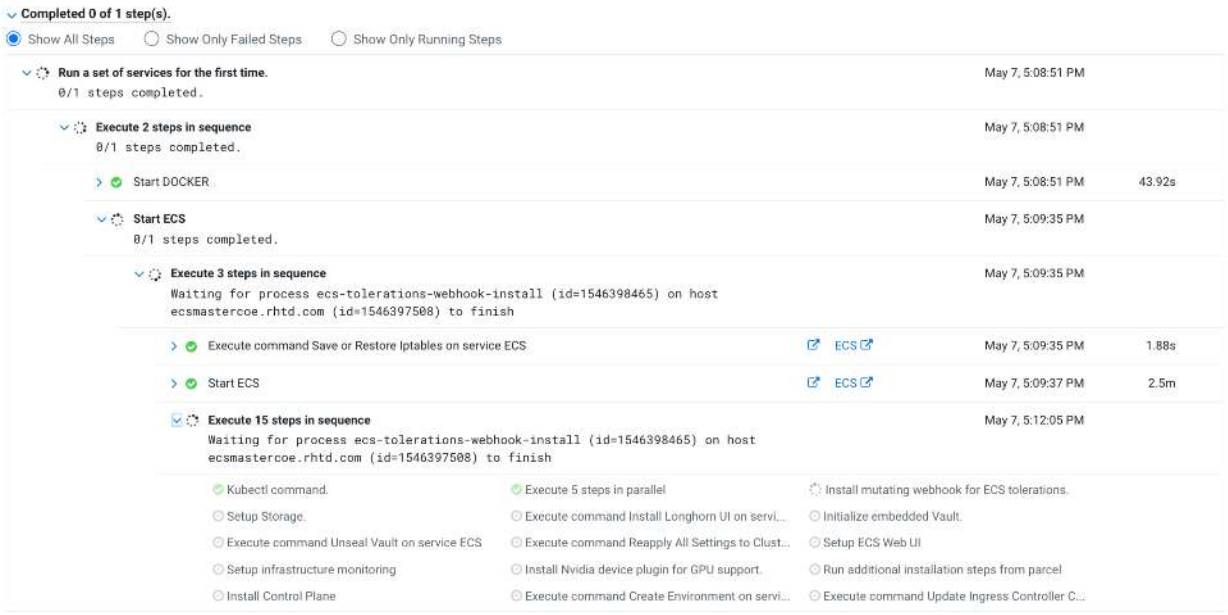
23. On the ***Inspect Cluster*** page, you can ***Inspect your Network Performance and Hosts***. Click on the ***Show Inspector Results***. If the inspect tool displays any issues, you can fix those issues and click on ***Run Again*** to rerun the inspect tool. After all of the issues have been resolved, click on ***Continue***.

Note: These inspections are more comprehensive host and network tests that you can ***optionally run***. To ***skip these tests***, select the ***I understand the risks of not running the inspections or the detected issues, let me continue with cluster setup*** checkbox.

Note: Safe to ignore unrelated errors in the host inspector result. For example, the hosts in a Cloudera on premises Containerized Cluster that have GPUs are required to have NVidia Drivers and NVidia-container-runtime installed. The following hosts do not satisfy this requirement: pvcecs-worker[1-12].cldrsetup.local
Since all hosts part of the ECS installation might not have NVIDIA GPU installed and NVidia driver and NVidia container-runtime is not installed on non-GPU node(s). It is safe to ignore the warning and click on the checkbox to continue with ECS installation.



Step. Install Data Services



24. While the other steps for installing data services cluster (ECS) is running, login to the pvcecs-master node and create kubeconfig file by copying the rke2.yaml file on ecs master node, in order to be able to run the kubectl commands from master node:

```
[root@pvcecs-master ~]# rm -rvf ~/.kube && mkdir -p ~/.kube && cp /etc/rancher/rke2/rke2.yaml
~/.kube/config

##### The kubectl binary path may vary, do whereis kubectl or use find or locate commands to find the
exact path of the kubectl binary on the node.
[root@pvcecs-master ~]# export PATH=/var/lib/rancher/rke2/data/v1.30.12-rke2r1-328c510931ca/bin/:$PATH
[root@pvcecs-master ~]# kubectl get pods
No resources found in the default namespace.
[root@pvcecs-master ~]# kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.43.0.1    <none>        443/TCP   13h
[root@pvcecs-master ~]#

##### Create alias on node, to run the kubectl command
[root@pvcecs-master ~]# echo "export
PATH=/var/lib/rancher/rke2/data/v1.30.12-rke2r1-328c510931ca/bin/:$PATH" >> ~/.bashrc && source ~/.bashrc
[root@pvcecs-master ~]# echo "alias k=kubectl" >> ~/.bashrc && source ~/.bashrc
#### You can copy the config file and set the above ~/.bashrc paths on any of ECS nodes to run kubectl
commands from there.
[root@pvcecs-master ~]#
```

Note: Run # kubectl get pods -A to review all pods and their status as either running or completed.

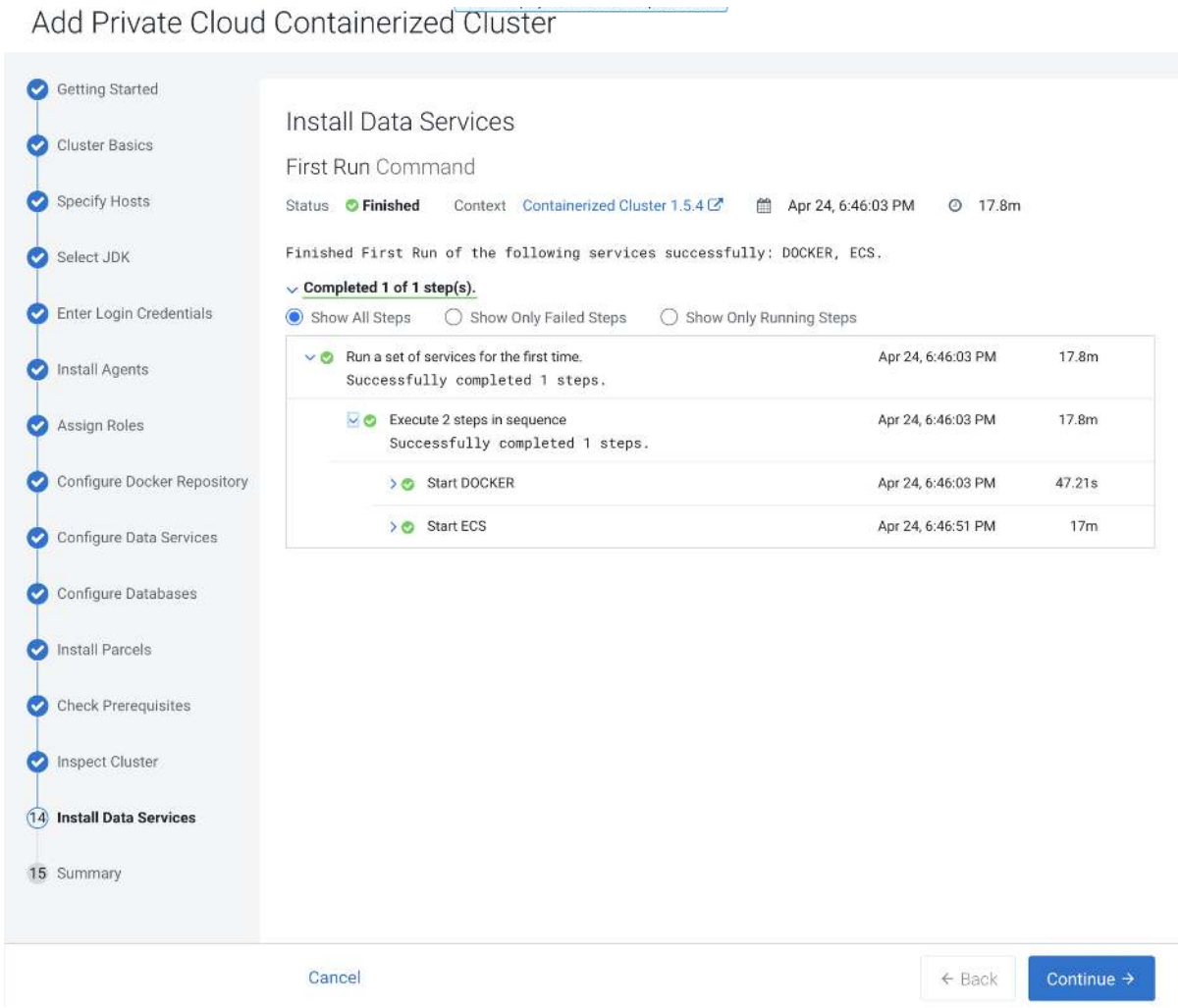
25. After the **RKE2 installation step is completed**, **keep the remaining steps running but immediately log in** to ***pvcecs-master*** node and ***update the config for coredns to point to private DNS*** (ipaserver). The coredns config should look like similar to below one:

```
##### Test the nslookup and dig from ubuntu image if it is able to resolve the IP from Private DNS
[root@pvcecs-master ~]# kubectl run -it ubuntu1 --image=ubuntu bash

##### If not able to resolve the IP from Private DNS, we need to point coredns to PrivateDNS/ FreeIPA server
[root@pvcecs-master ~]# kubectl get configmap -n kube-system | grep coredns
```

```
[root@pvcecs-master ~]# kubectl edit configmap rke2-coredns-rke2-coredns -n kube-system
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  Corefile: |
    .:53 {
        errors
        health {
            lameduck 5s
        }
        ready
        kubernetes cluster.local cluster.local in-addr.arpa ip6.arpa {
            pods insecure
            fallthrough in-addr.arpa ip6.arpa
            ttl 30
        }
        prometheus :9153
        forward . 172.31.24.240 {
            max_concurrent 1000
        }
        cache 30
        loop
        reload
        loadbalance
    }
kind: ConfigMap
metadata:
  annotations:
    meta.helm.sh/release-name: rke2-coredns
    meta.helm.sh/release-namespace: kube-system
  creationTimestamp: "2024-05-17T07:12:19Z"
  labels:
    app.kubernetes.io/instance: rke2-coredns
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: rke2-coredns
    helm.sh/chart: rke2-coredns-1.24.006
    k8s-app: kube-dns
    kubernetes.io/cluster-service: "true"
    kubernetes.io/name: CoreDNS
  name: rke2-coredns-rke2-coredns
  namespace: kube-system
  resourceVersion: "39715"
  uid: dcb8f600-e107-4c1b-9db7-701060fe063a

[root@pvcecs-master ~]# kubectl rollout restart deployment rke2-coredns-rke2-coredns -n kube-system

##### Verify the status that deployment of coredns is updated with configmap changes
[root@pvcecs-master ~]# kubectl get pod -A |grep dns

##### Test the nslookup and dig from ubuntu image if it is actually able to resolve the IP from Private DNS
[root@pvcecs-master ~]# kubectl run -it ubuntu1 --image=ubuntu bash
root@ubuntu2:/# apt-get update && apt-get install net-tools dnsutils -y

root@ubuntu2:/# nslookup console-cdp.apps.cldr.internal
```

26. ***Install Data Services*** step will run a set of first run commands and report status on Data Services installation. The installation progress is displayed on the ***Install Data Services*** page. This step will take ***nearly an hour to complete***. When the installation is complete, click ***Continue***.

Note: Installing Data Services can take several hours. The copying operation for Docker repository may take 4 - 5 hours.



27. If you still face any issue while making the services up or during the installation or start of any ECS services please refer to troubleshooting PvC Data Services Cluster part at the end of this document. Though, some of the major issues during installation, their cause and their resolution is listed as below:

```
ksahu@Kuldeeps-MacBook-Air ~ %

Name resolution not working inside pod for console-cdp leading cli pod to fail and env not getting created
Cluster name CM url not able to connect from ECS cluster
Solution:
Update coredns pod by edit and redeploy and mention private dns server ip in config
make sure wildcard dns is set up properly.

After issue is fixed and env get created for ECS: it will show like below:
Fri May 17 01:23:55 AM PDT 2024
Running on: pvcecs-master.cldrsetup.local (172.31.30.239)
Fetching session token...
Creating environment cldrsetup
secret "cm.args" deleted
secret/cm.args created
job.batch "cli" deleted
job.batch/cli created
job.batch/cli condition met
/opt/app-root/lib64/python3.9/site-packages/urllib3/connectionpool.py:1103: InsecureRequestWarning:
Unverified HTTPS request is being made to host 'console-cdp.apps.cldrsetup.local'. Adding certificate
verification is strongly advised. See:
https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
```

```
  warnings.warn(
{
    "environment": {
        "environmentName": "cldrsetup",
        "crn":
"crn:altus:environments:us-west-1:d750ff5a-791d-46c8-a95a-f1efaf6185ef:environment:cldrsetup/ad47b514-a135-47
dc-a8dd-8d11f33145f7",
        "cloudPlatform": "standard"
    }
}
Waiting for environment cldrsetup to be available
The http response code is 200
...
The http response code is 200
...
The http response code is 200
...
The http response code is 200
...
The http response code is 200
Environment was successfully created
```

==========

Failed to reconcile with temporary etcd:
Solution:
bootstrap data already found and encrypted with different token rke error
rm -vf /var/lib/rancher/k3s/server/node-token

==========

Update k8s cordons and rollout
Solution:
https://www.reddit.com/r/k3s/comments/p3rdap/lost_bootstrap_data_already_found_and_encrypted/
[root@pvcecs-master bin]# rm -vf /var/lib/rancher/k3s/server/node-token
rm -rvf /var/lib/rancher/ /etc/rancher

==========

X509 error-
Solution:
do cleanup for ECS properly

==========

Unable to connect 6443
Solution:
mkdir -p ~/.kube && cp /etc/rancher/rke2/rke2.yaml ~/.kube/config

==========

Aws_key_id error in cde cluster creation
Error Logs:
Moving from CadenceInstallCompleted to ClusterChartInstallationFailed
{"StartState":"CadenceInstallCompleted","EndState":"ClusterChartInstallationFailed","Attempt":3}
Chart installation failed, dex base overrides: unable to retrieve aws_key_id from the env service
Error fetching fluent log config: unable to retrieve aws_key_id from the env service
Generating Webhook cert and key
Setting up TGT generator
Installing dex-base charts for CDE 1.24.0
{"StartState":"CadenceInstallCompleted","EndState":"ClusterChartInstallationFailed","Attempt":2}
Chart installation failed, dex base overrides: unable to retrieve aws_key_id from the env service
Error fetching fluent log config: unable to retrieve aws_key_id from the env service

Cause:
Configure ozone with other base cluster services before env creation
Solution:
Install ozone on base cluster, restart base cluster services after applying the stale configurations and recreate the ECS environment.
https://community.cloudera.com/t5/Support-Questions/Unable-to-create-Data-Engineering-Cluster-in-CDP-Private/m-p/377969

Note: Ozone is a necessary service if you want to install CDE
This error typically occurs when required services like Ozone are not present in the base cluster at the time of CDE environment creation. In such cases, the cdp-services namespace remains empty and lacks critical components like the s3proxy, which is essential for Fluent Bit to forward logs to Ozone. The s3proxy deployment is triggered automatically during the environment creation process and cannot be manually deployed via Helm or other means. Therefore, if Ozone is added after the environment has already been created, the only solution is to delete and recreate the entire Data Services installation. This ensures that the s3proxy is properly deployed, allowing the environment to function correctly and resolving the AWS credential-related error during CDE Virtual Cluster setup.

==========

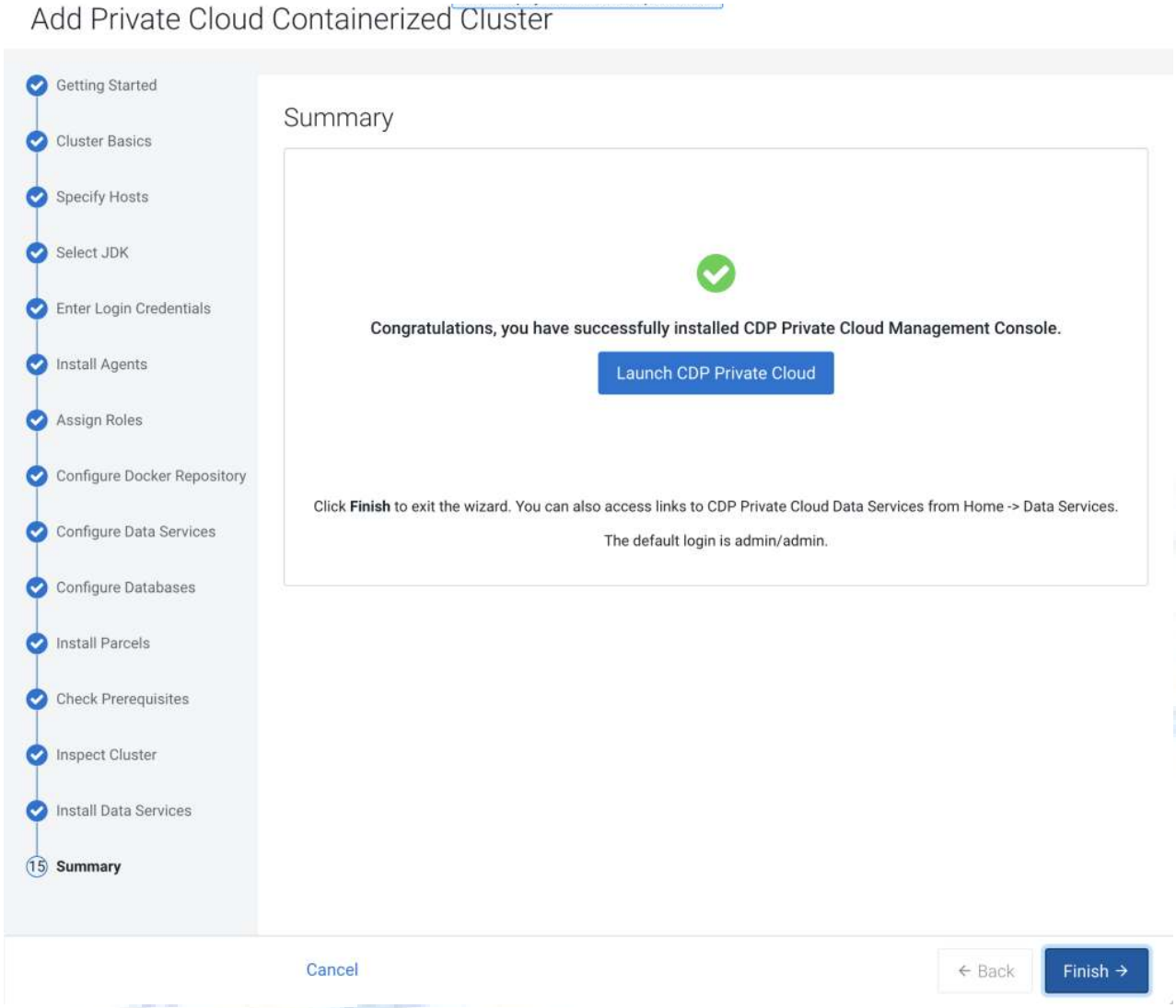
Other:
https://repost.aws/knowledge-center/create-lv-on-ebs-partition

==========

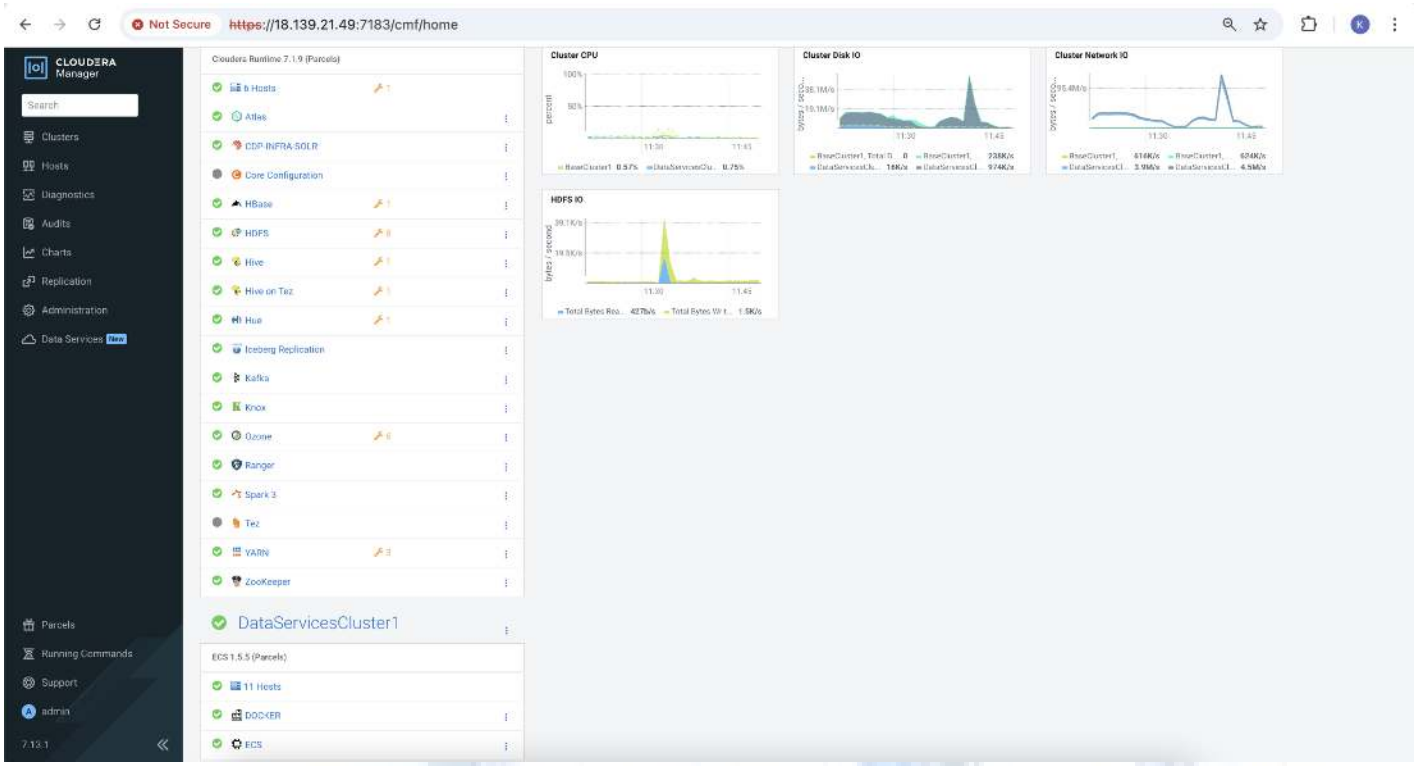
UNRESOLVED: CDE Error

```
[root@pvcecs-master ~]# k get po -A|grep dex | grep -v -E 'Run|Comp'
dex-base-j95kjx9f                        cdp-cde-embedded-db-0
0/1     Pending                  0               8m54s
dex-base-j95kjx9f                        dex-base-data-connectors-746fdcddc8-jvbsx
0/1     CrashLoopBackOff         6 (3m22s ago)   8m54s
dex-base-j95kjx9f                        dex-base-management-api-6c5d48fd96-dgrnt
0/1     CrashLoopBackOff         6 (2m21s ago)   8m54s

[root@pvcecs-master ~]# kubectl logs -n dex-base-j95kjx9f dex-base-data-connectors-746fdcddc8-jvbsx
Defaulted container "data-connectors" out of: data-connectors, k8tz (init)
{"level":"INFO","timestamp":"2024-05-27T01:37:23.834-0700","caller":"cmd/data-connectors.go:92","message":
"Configuration file used: /etc/dc/data-connectors.yaml"}
{"level":"INFO","timestamp":"2024-05-27T01:37:23.834-0700","caller":"cmd/start.go:112","message":"Data
store init","store type":"sql"}

2024/05/27 01:37:23
/grid/0/jenkins/workspace/workspace/App_builds/SOURCES/data-connectors/pkg/store/sqlstore/utils.go:79
[error] failed to initialize database, got error dial tcp 10.43.139.205:3306: connect: connection refused
{"level":"FATAL","timestamp":"2024-05-27T01:37:23.840-0700","caller":"cmd/start.go:115","message":"store
initialisation failure","error":"dcObject persistent store init failure: DB session creation error: error
creating DB session for 'mysql', error: dial tcp 10.43.139.205:3306: connect: connection
refused","stacktrace":"github.infra.cloudera.com/CDH/data-connectors/pkg/cmd.startServer\n\t/grid/0/jenkin
s/workspace/workspace/App_builds/SOURCES/data-connectors/pkg/cmd/start.go:115\ngithub.infra.cloudera.com/C
DH/data-connectors/pkg/cmd.glob..func3\n\t/grid/0/jenkins/workspace/workspace/App_builds/SOURCES/data-conn
ectors/pkg/cmd/start.go:68\ngithub.com/spf13/cobra.(*Command).execute\n\t/grid/0/jenkins/.asdf/installs/go
lang/1.17.6/packages/pkg/mod/github.com/spf13/cobra@v1.1.3/command.go:856\ngithub.com/spf13/cobra.(*Comman
d).ExecuteC\n\t/grid/0/jenkins/.asdf/installs/golang/1.17.6/packages/pkg/mod/github.com/spf13/cobra@v1.1.3
/command.go:960\ngithub.com/spf13/cobra.(*Command).Execute\n\t/grid/0/jenkins/.asdf/installs/golang/1.17.6
/packages/pkg/mod/github.com/spf13/cobra@v1.1.3/command.go:897\ngithub.infra.cloudera.com/CDH/data-connect
ors/pkg/cmd.Execute\n\t/grid/0/jenkins/workspace/workspace/App_builds/SOURCES/data-connectors/pkg/cmd/data
-connectors.go:105\nmain.main\n\t/grid/0/jenkins/workspace/workspace/App_builds/SOURCES/data-connectors/cm
d/data-connectors/main.go:32\nruntime.main\n\t/grid/0/jenkins/.asdf/installs/golang/1.17.6/go/src/runtime/
proc.go:255"}
[root@pvcecs-master ~]# kubectl logs -n dex-base-j95kjx9f dex-base-management-api-6c5d48fd96-dgrnt
Defaulted container "dex-base-management-api" out of: dex-base-management-api, k8tz (init)
No Envoy proxy, skip waiting for its readiness
Skip fetching DB certificates
Running main binary: /dex/bin/runtime-management-server
2024/05/27 01:38:24 INFO config.go:61 Loaded config from: /etc/dex/conf/dex.yaml
2024/05/27 01:38:24 INFO db.go:204 Opening database connection, driver: mysql
2024/05/27 01:38:24 FATAL db.go:226 dbConn.BeginTx failed: dial tcp 10.43.139.205:3306: connect:
connection refused

[root@pvcecs-master ~]# kubectl logs -n dex-base-j95kjx9f cdp-cde-embedded-db-0
Defaulted container "cdp-cde-embedded-db" out of: cdp-cde-embedded-db, k8tz (init)
[root@pvcecs-master ~]#
```

28. When the installation is complete, the ***Summary page*** appears. Click ***Launch Cloudera on premises***.



29. Alternatively, you can also click ***Finish*** and then ***access the Private Cloud Data Services*** instance from ***Cloudera Manager***. Click ***Data Services***, then click ***Open Private Cloud Data Services*** for the applicable Data Services cluster.

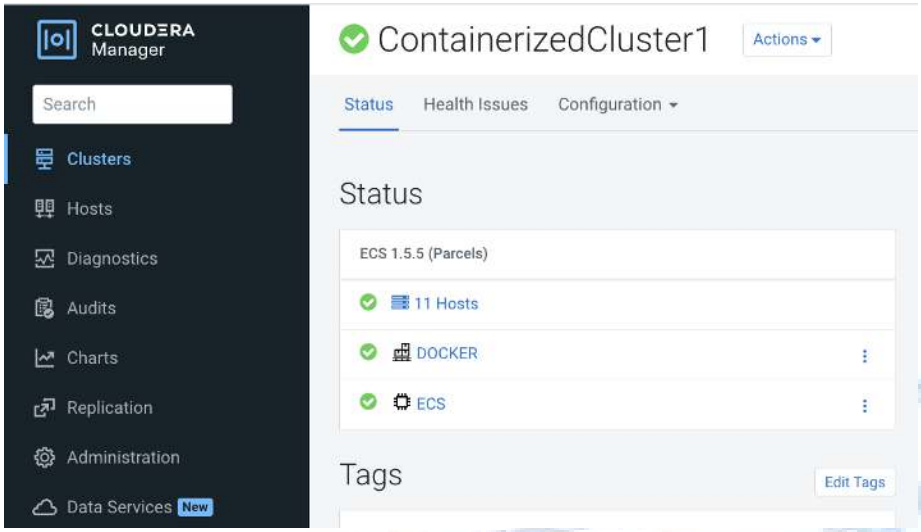


****

Additional Steps for ECS Cluster Setup: ***(Optional, Skip this step)***

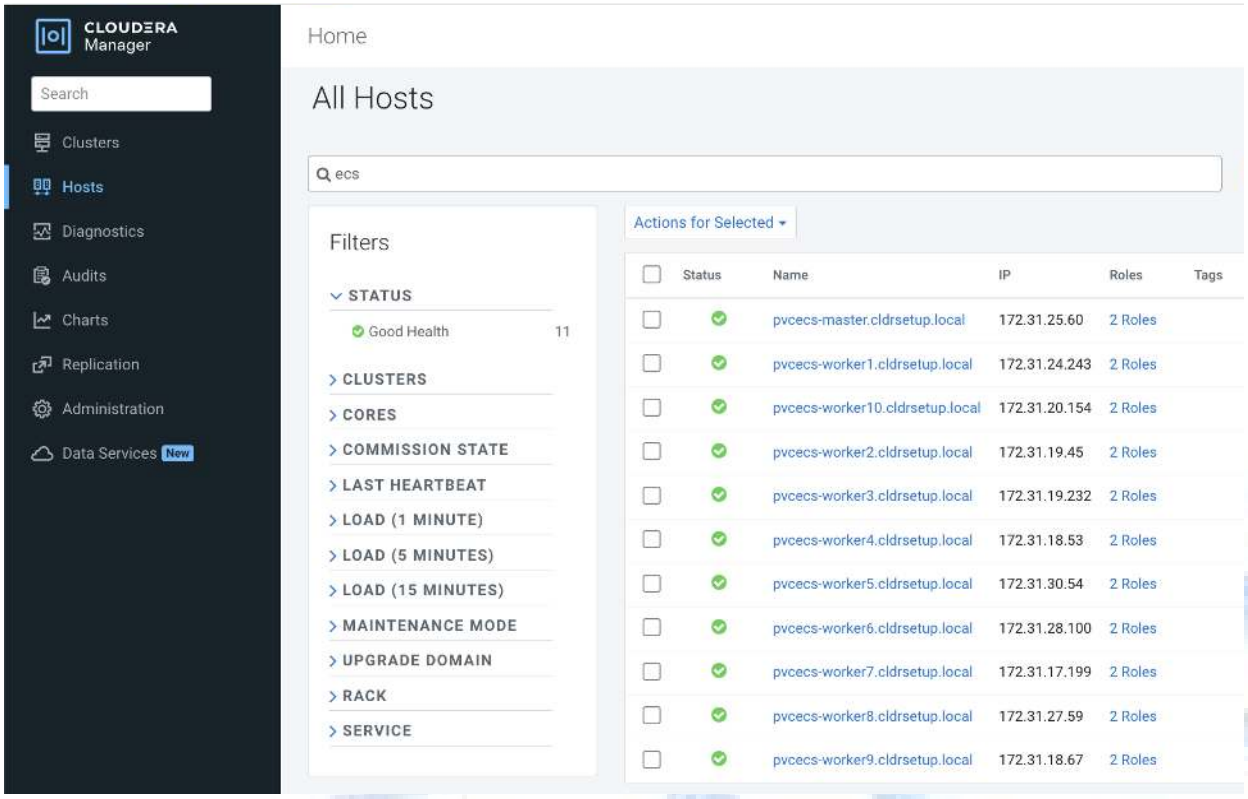
Note: If nvgfd-gpu-feature-discovery-xxxx pods remain in crashlookbackoff please apply patch to fix the issue.

```
[root@pvcecs-master ~]# kubectl patch clusterrolebinding gpu-feature-discovery -p
'{"subjects":[{"kind":"ServiceAccount","name":"gpu-feature-discovery","namespace":"kube-system"}]}'
```

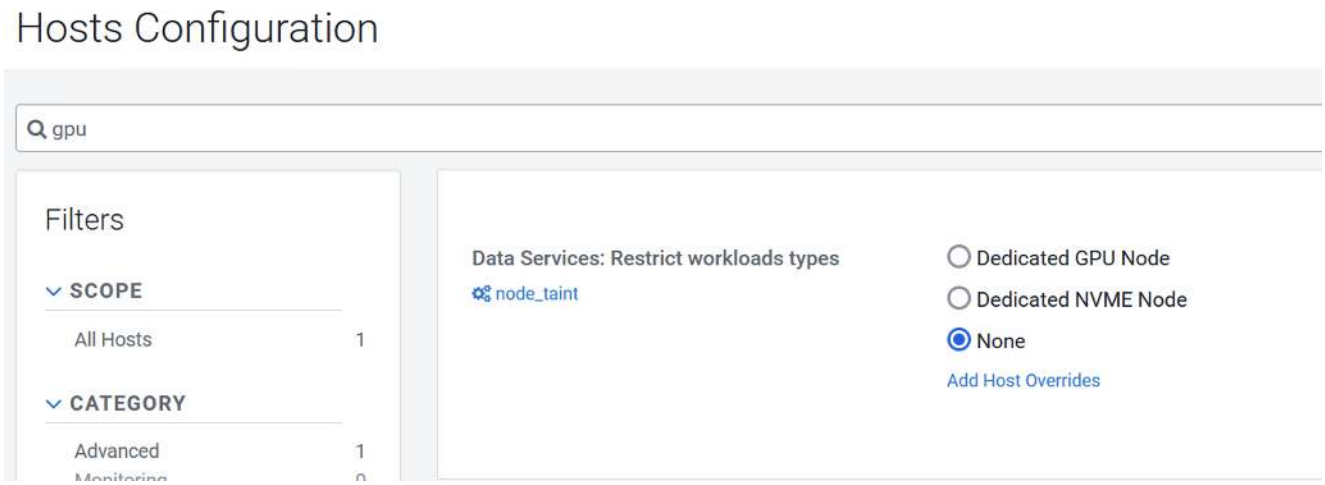


To reserve a GPU node in Cloudera on premises Data Services ECS cluster, assign a taint to the node. Set the node taint “nvidia.com/gpu: true:NoSchedule” For more details on setting up GPU node: https://docs.cloudera.com/machine-learning/1.5.5/private-cloud-requirements/topics/ml-gpu-node-setup.html

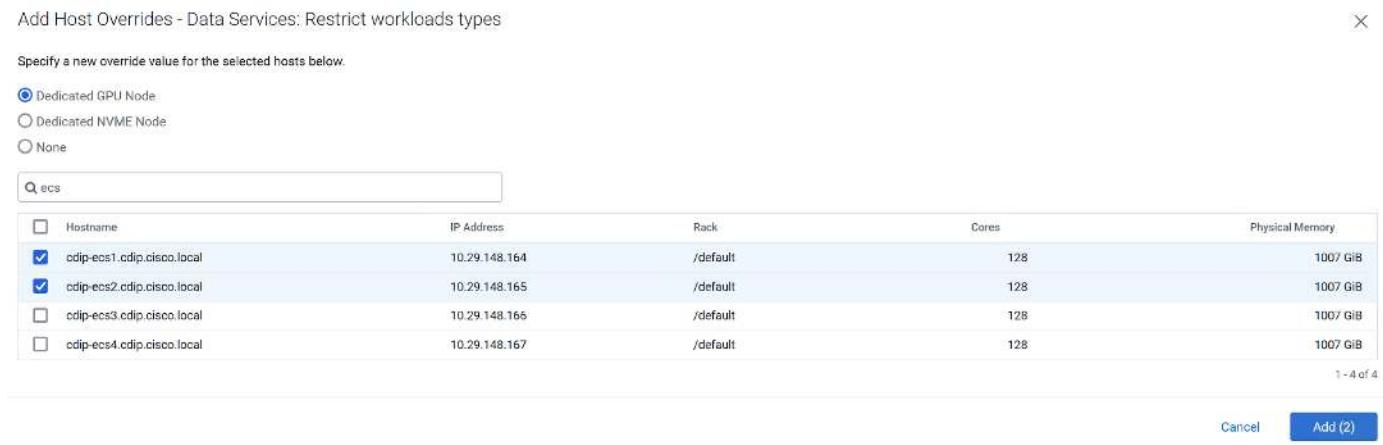
Step 1. To ***set up a GPU node*** for ECS, go to ***Hosts > Configuration***.



Step 2. Edit value for Data Services: Restrict workload types (node_taint) by clicking on Add Host Overrides.



Step 3. Add Host Overrides for the ECS nodes as per the requirement. For example, we selected two of the four nodes as Dedicated GPU Nodes.



Step 4. Click on ***Add*** and click ***Save Changes***.

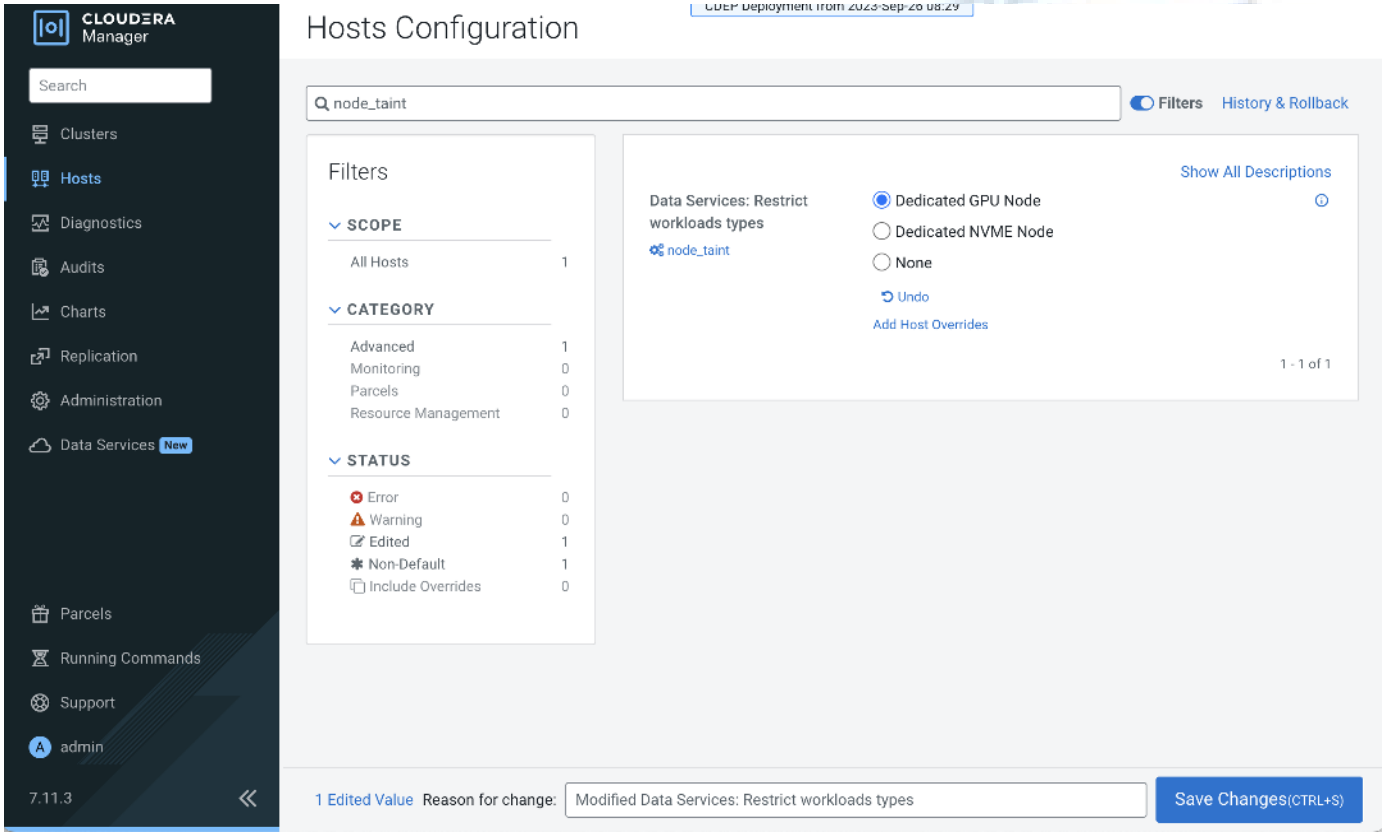
****

Dedicating ECS nodes for specific workloads *(Optional, Skip this step)*

You use Cloudera Manager to dedicate Embedded Container Service (ECS) cluster nodes for specific workloads. You can dedicate GPU nodes for CAI workloads, and NVME nodes for CDW workloads.

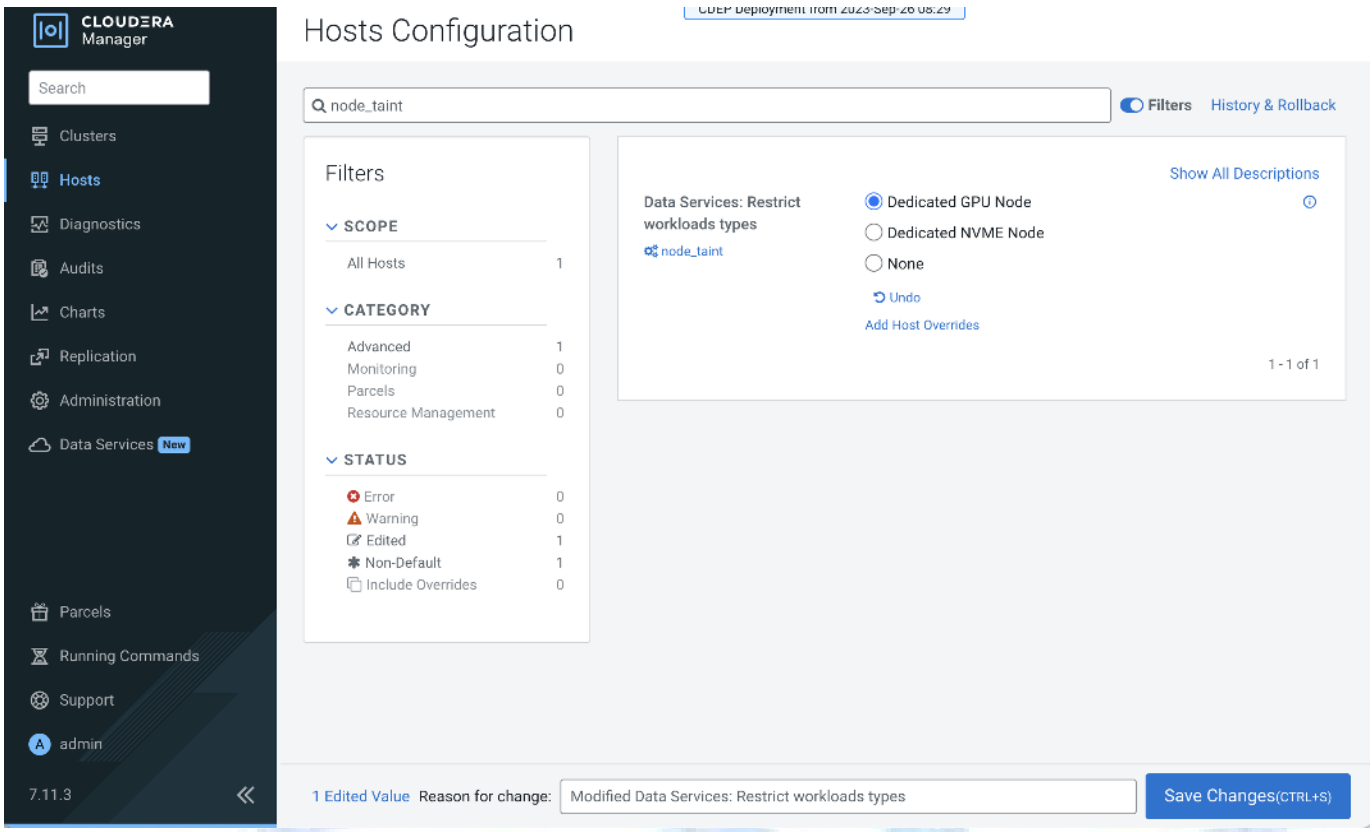
Dedicating ECS nodes when creating a new cluster

1. Check the ECS installation requirements.
2. Add the new hosts to ***Cloudera Manager.***
3. In Cloudera Manager, click ***Hosts > All Hosts***, then select one or more of the ***new ECS hosts***.
4. Click the ***Configuration*** tab, then use the ***Search box*** to locate the ***node_taint*** configuration property.
5. ***Select Dedicated GPU Node*** to dedicate the node for CAI workloads, or select ***Dedicated NVME node*** to dedicate the node for ***CDW*** workloads. When either of these options are selected, no other workload pods will be allowed to run on the dedicated node.
6. Click ***Save Changes***.
7. ***Repeat the previous steps*** to add the ***other ECS hosts*** to ***Cloudera Manager*** and ***assign*** workload types.
8. Follow the ECS installation procedure. When you reach the ***Specify Hosts*** page in the installation wizard, the hosts you added to ***Cloudera Manager*** appear. Select the hosts, click ***Continue***, then proceed through the rest of the installation wizard.
9. After the installation is complete, the applicable workloads will only run on the specified dedicated nodes.



Dedicating ECS nodes in an existing cluster

1. Open the ***Cloudera Manager Admin Console***.
2. On the ***Home page***, click the ***ECS Cluster***.
3. Click ***Hosts***, select one or more of the ***ECS hosts***, then click the ***Configuration*** tab.
4. Click the ***Configuration*** tab, then use the ***Search box*** to locate the ***node_taint*** configuration property.
5. Select ***Dedicated GPU Node*** to dedicate the node for ***CAI*** workloads, or select ***Dedicated NVME node*** to dedicate the node for ***CDW*** workloads. When either of these options are selected, no other workload pods will be allowed to run on the dedicated node.

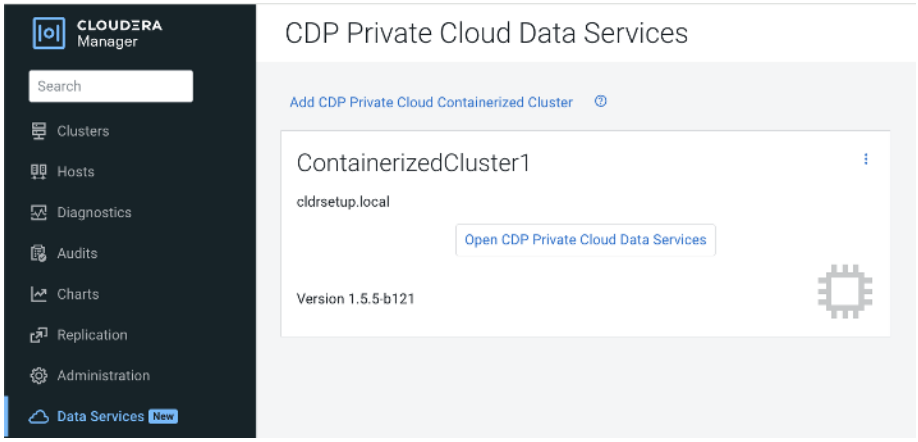


6. Click ***Save Changes***.
7. ***Repeat the previous steps*** to assign workload types to the other ***ECS*** hosts.
8. On the ***ECS Cluster*** landing page, click ***Actions > Refresh Cluster.***
9. After the Refresh is complete, click ***Actions > Rolling Restart***.

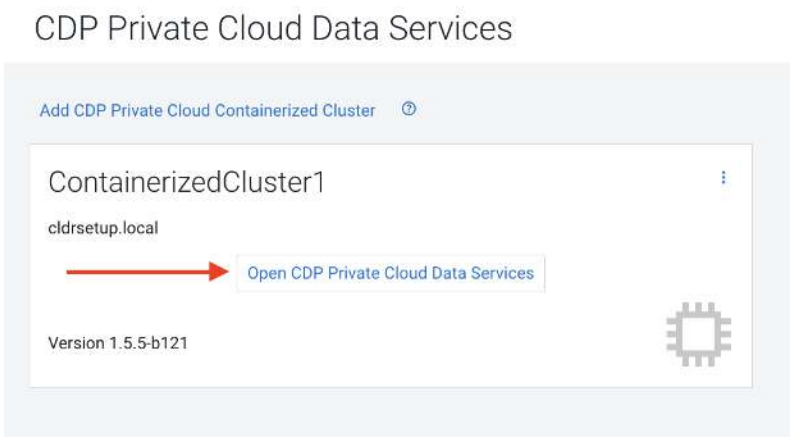
****

Accessing Cloudera on premises

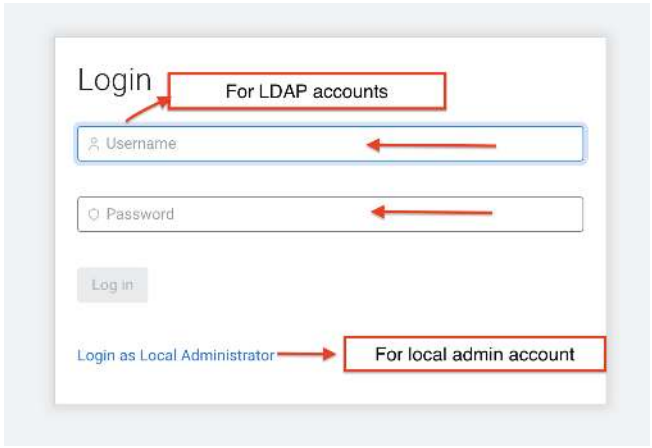
Step 1. From the ***Cloudera Manager*** screen, click on ***Data Services(New)*** in the left pane.



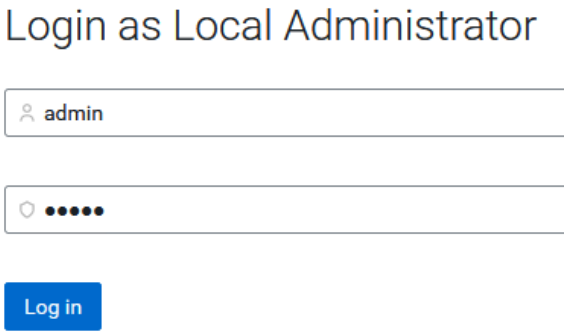
Step 2. On the ***CDP Private Cloud Containerized services*** page, click on the ***Open CDP Private Cloud Data Services*** button. This will open the ***Cloudera on premises authentication*** page.



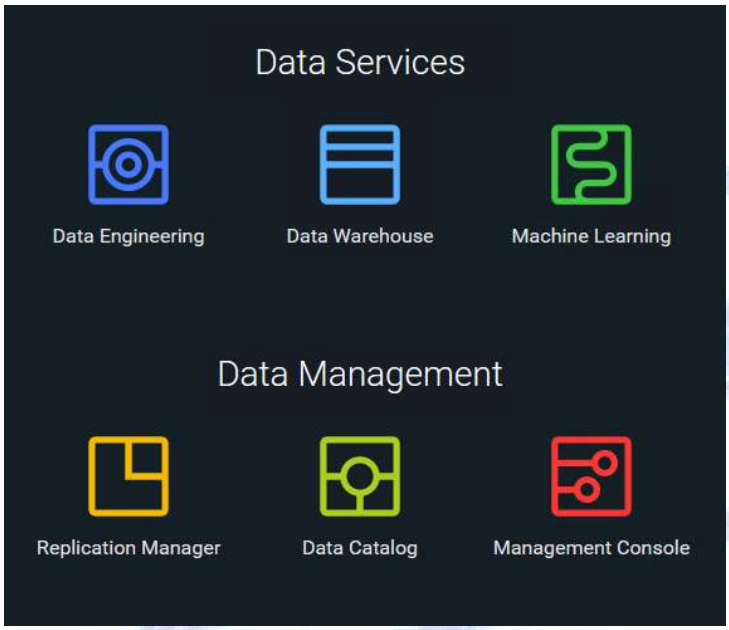
Step 3. On the PvC DS Authentication Page, if you have ***LDAP account*** credentials, enter its username and password and then click on ***Login***. Else, you can click on ***Login as Local Administrator***, and enter the default credentials. ***(admin/admin)***



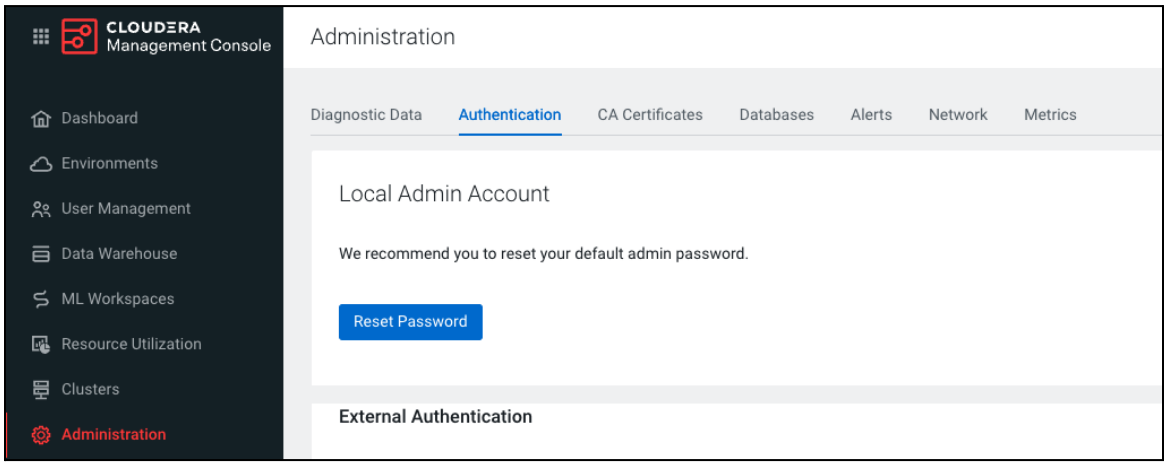
Step 4. *Login* to *Cloudera on premises Data Services* as *local administrator*. *(admin/admin)*

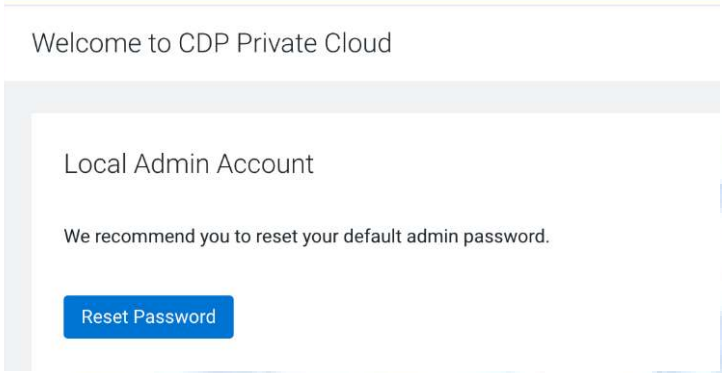


Step 5. After authenticating successfully, you will land at the *CDP console/Data Services* page. From this page, you can navigate to different data services and the management services. Click on the *Management Console*.



Step 6. On the *Welcome to Cloudera on premises* page, click *Reset Password* to change the *Local Administrator Account password*. (OR) On the *Management Console* page, navigate to *Administration > Authentication*, and then click *Reset Password* to change the *Local Administrator Account password*.



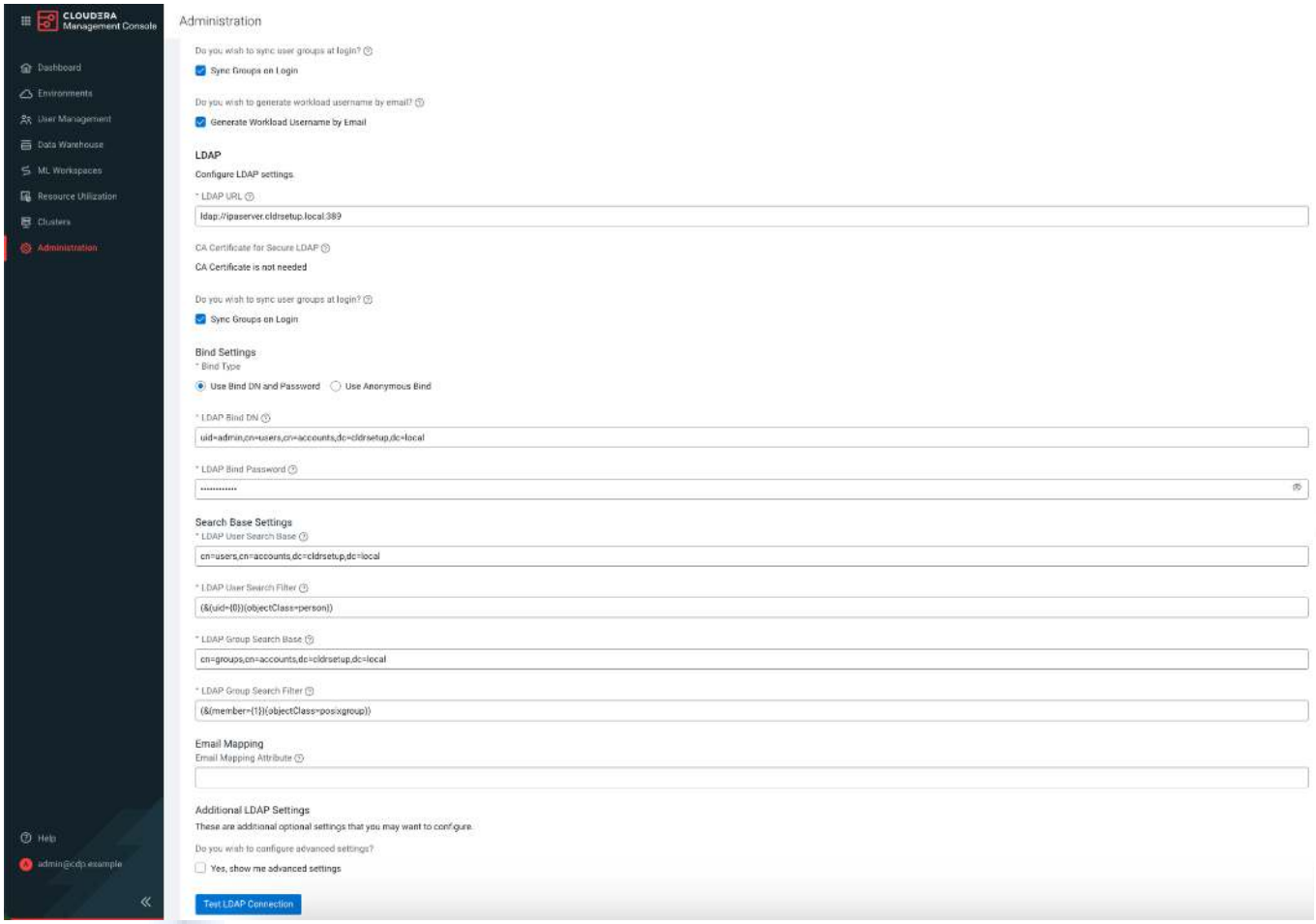


Step 7. ***Set up external authentication*** using the URL of the LDAP server and a CA certificate (If using prior existing LDAP server and not proceeding with FreeIPA setup) of your secure LDAP ***(e.g. ldap://<ipa_or_ldap_server_fqdn>:389/)***. `Learn more about` `LDAP user authentication for Cloudera on premises.` Enter values for ldap authentication, as mentioned in the below table.

Table 12. LDAP Integration

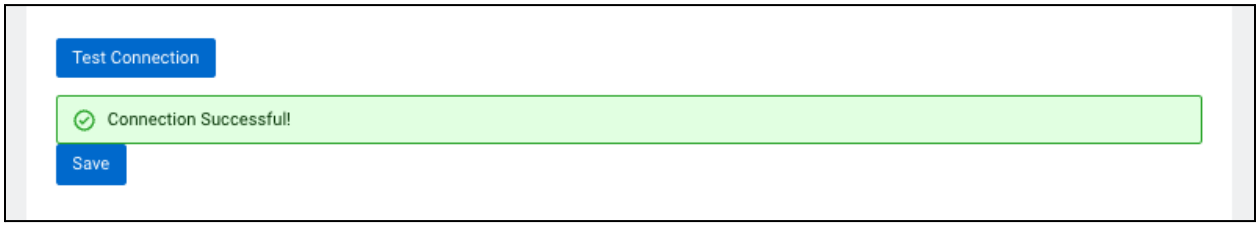
Component	Value
Authentication Backend Order:	Database then EXTERNAL
Authorization Backend Order:	Database and EXTERNAL
External Authentication Type:	LDAP
LDAP URL:	ldap://ipaserver.cldrsetup.local:389/
LDAP Bind User Distinguished Name:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Bind Password:	***<cloudera123>*** (password for KDC admin, configured earlier)
Active Directory Domain: **(For AD Based LDAP)**	<AD DOMAIN>
LDAP User Search filter:	(&(uid={0})(objectClass=person))
LDAP User Search Base:	cn=users,cn=accounts,dc=cldrsetup,dc=local
LDAP Group Search filter:	(&(member={1})(objectClass=posixgroup))

Component	Value
LDAP Group Search Base:	cn=groups,cn=accounts,dc=cldrsetup,dc=local
LDAP DistName Pattern:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local

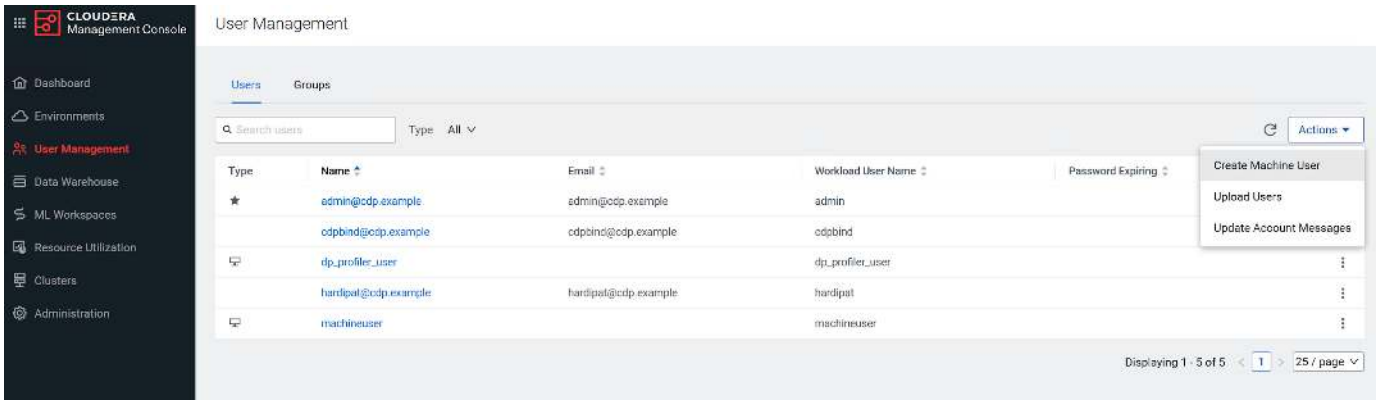


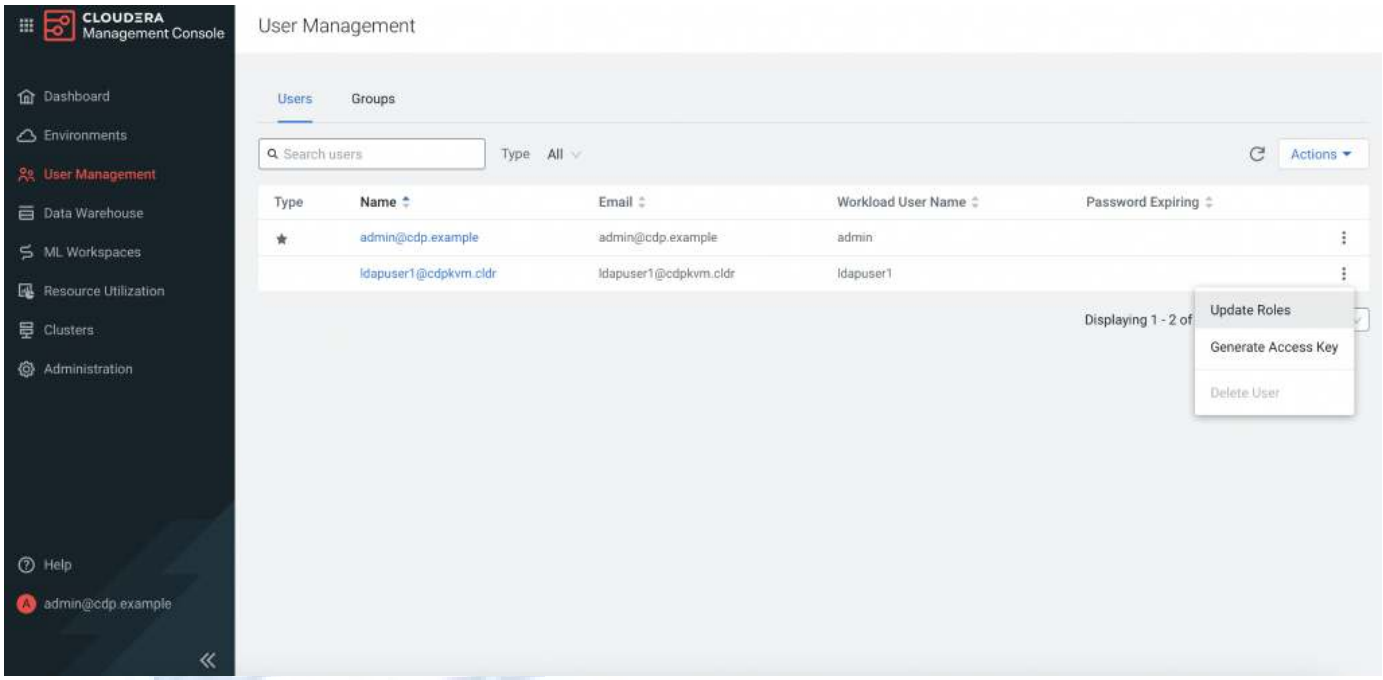
Step 8. Follow the instructions on the ***Welcome to CDP Private Cloud page*** to complete this step.

Step 9. Click ***Test Connection*** to ensure that you are able to connect to the ***configured LDAP server***.



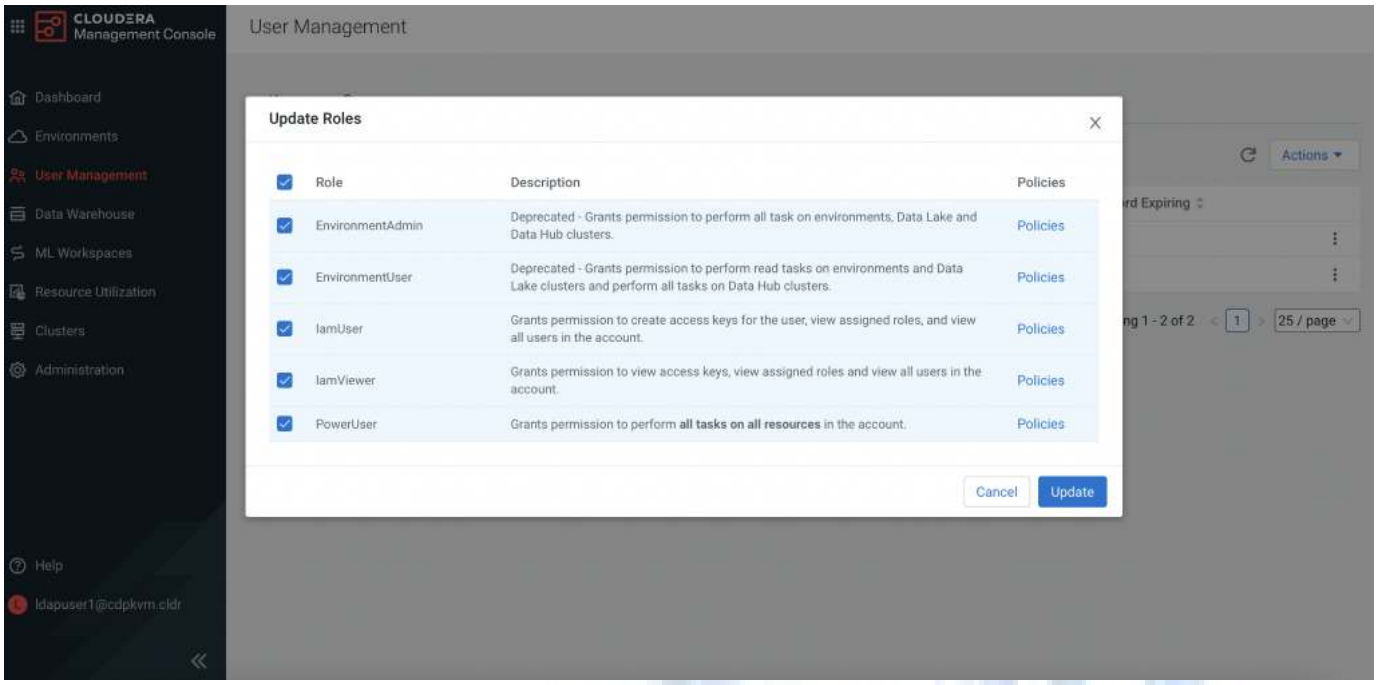
Step 10. The ***User Management*** tab allows users to add or update roles on existing users. ***Groups*** tab allows users to sync user groups from the active directory to access ***CDP Data Services***.





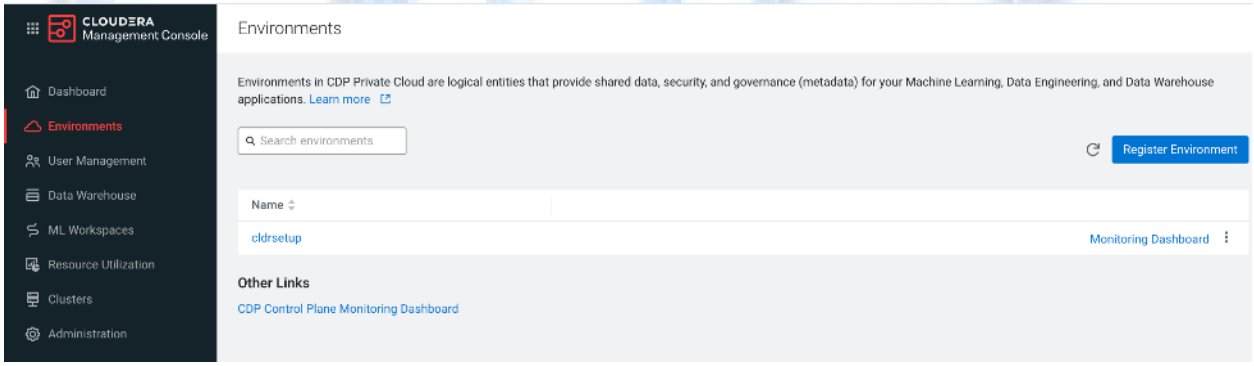
For more details on Cloudera on premises Management console please visit:

https://docs.cloudera.com/management-console/1.5.5/index.html

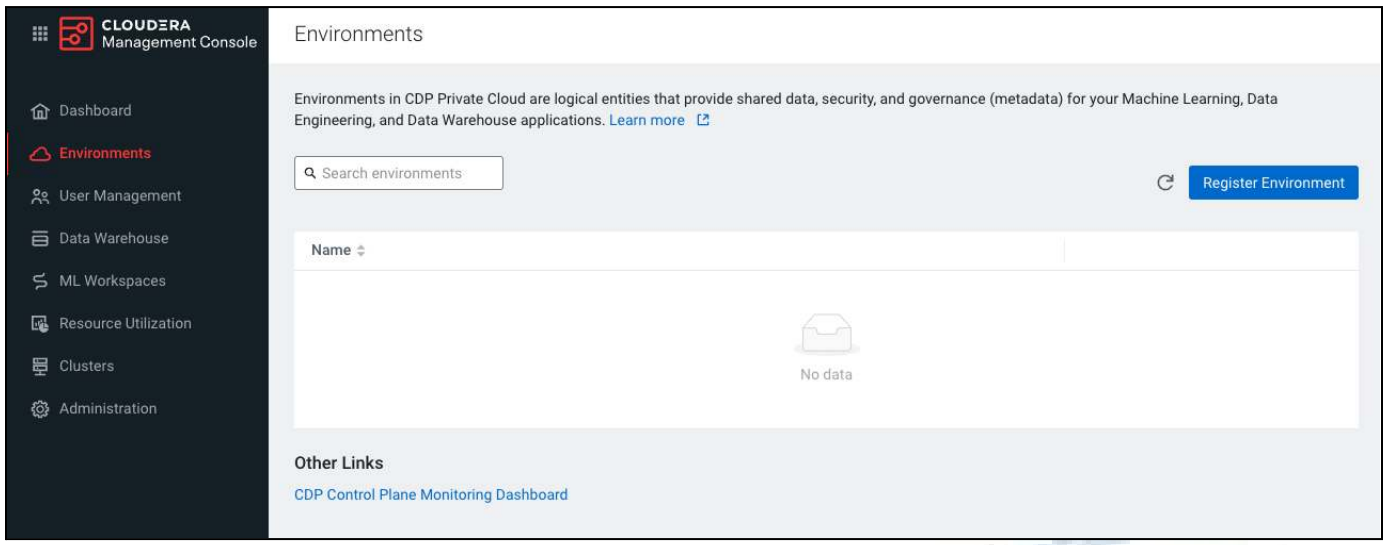


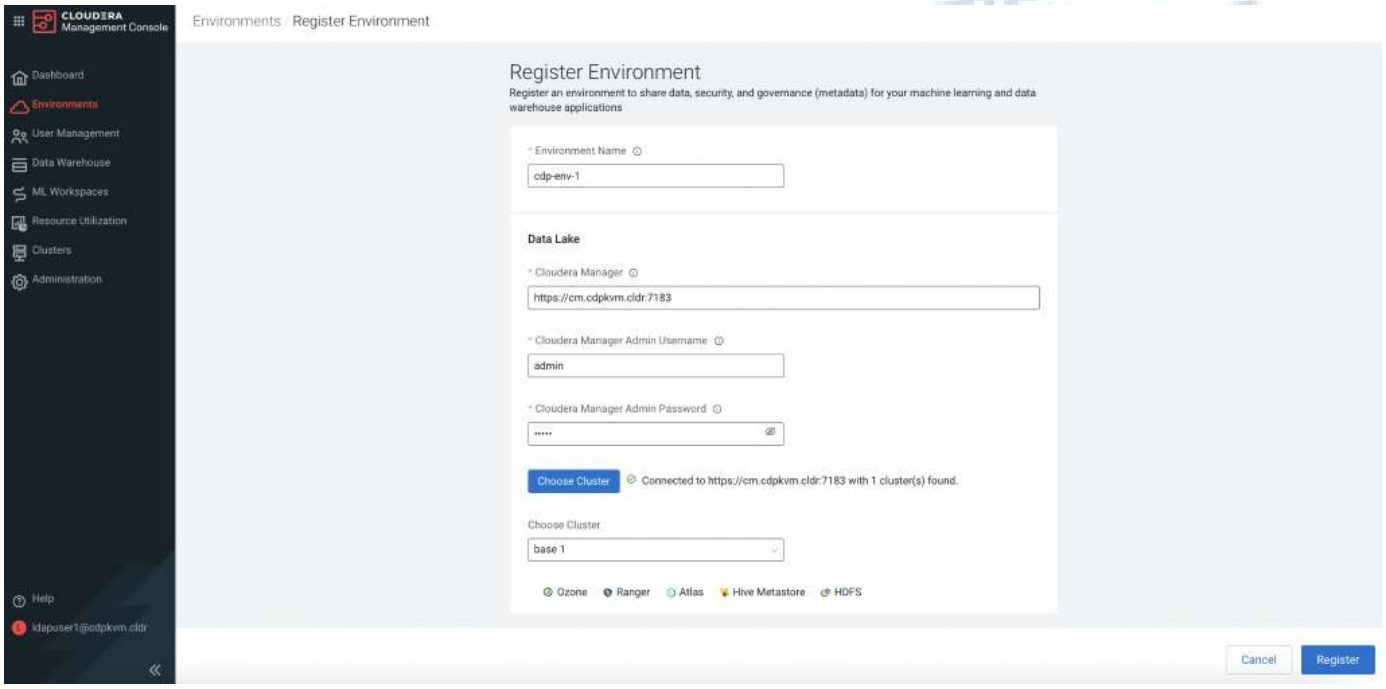
Step 11. After successfully configuring and testing the setup for ***LDAP integration***, the page will auto-redirect for the Environments Page, if not, navigate to the `Environments` page, by clicking into the menu in the left pane of your screen.

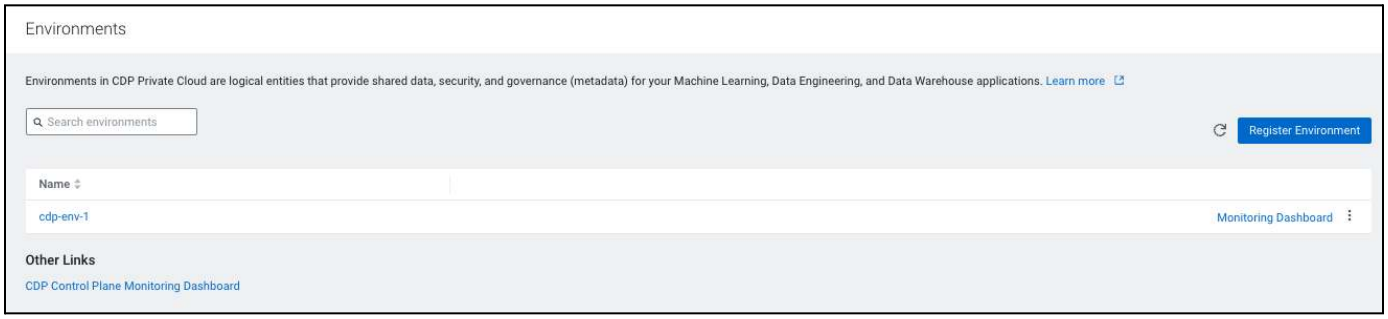
Step 12. If everything was correctly setup previously, then you will be able to see the environment registered by default for your cluster as shown below:

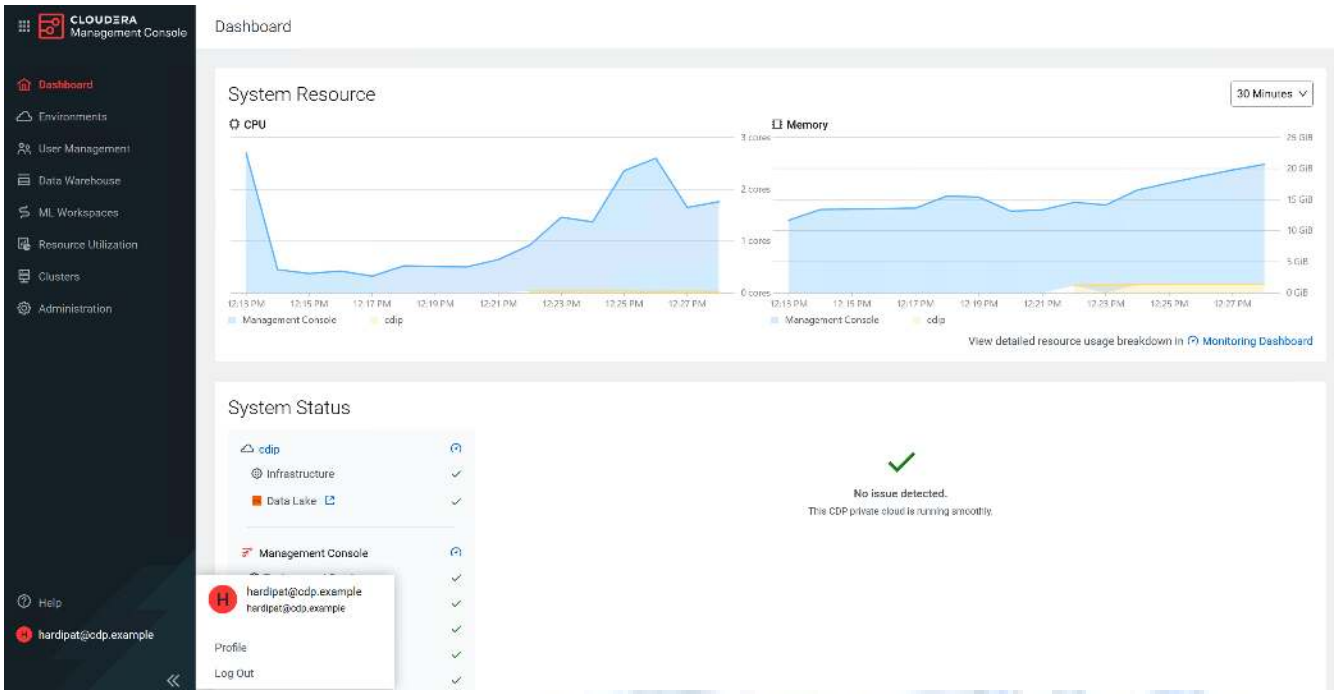


Step 13. If environment is not created somehow due to some issue during the installation, page will auto-redirect for ***Register Environment*** Page, if not, navigate to the `Environments` page, and select ***Register Environment*** where you will provide the Cloudera Manager details and credentials to register the ***PvC Base Cluster DataLake/Control Plane environment*** to DS. Click ***Choose Cluster,*** select the PvC Base cluster from the populated list and click on ***Register***.









Step 14. To come to this section further, from the ***Cloudera Manager*** screen, click on ***Data Services(New)*** in the left pane and then click on ***Open Private Cloud Data Services*** to launch your ***CDP Private Cloud Data Services instance***. Log in using the default username and password ***admin***.

- Click ***Launch CDP*** to launch your CDP Cloudera on premises.
- Log in using the default username and password admin.
- In the ***Welcome to CDP Private Cloud*** page, click ***Change Password*** to change the Local Administrator Account password.

- Set up external authentication using the URL of the LDAP server and a CA certificate of your secure LDAP. Follow the instructions on the ***Welcome to CDP Private Cloud*** page to complete this step.
- Click ***Test Connection*** to ensure that you are able to connect to the configured LDAP server.

- Register a CDP Cloudera on premises environment
- Create your first Virtual Warehouse in the CDW Data Service
- Provision an ML Workspace in the CAI Data Service
- Add a CDE service in the CDE Data Service

****

Cloudera on premises Machine Learning (CAI)

Please review Cloudera on premises Machine Learning for more details.

Please review requirements page for ECS and get started with CAI on Cloudera on premises:
https://docs.cloudera.com/machine-learning/1.5.5/private-cloud-requirements/topics/ml-pvc-intro.html

For more details on CAI workspace and how to steps, visit:
https://docs.cloudera.com/machine-learning/cloud/workspaces/topics/ml-provision-workspaces.html

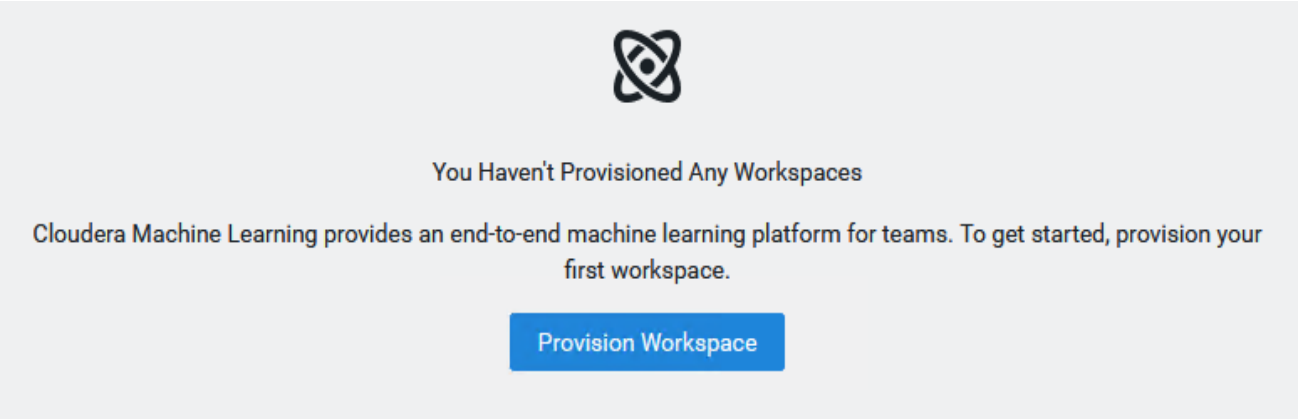
https://docs.cloudera.com/machine-learning/1.5.5/workspaces-privatecloud/topics/ml-pvc-provision-ml-workspace.html

ML Workspace Creation:

To get started with CAI follow steps below:

1. On the ***Cloudera Private Cloud Data Services console***, click on Cloudera ***AI***.

2. First time login requires provision of a workspace. Since this will be the first time you open CAI, there will be no CAI workspace. You will see the screen below. Click on ***Provision Workspace*** on the same page.

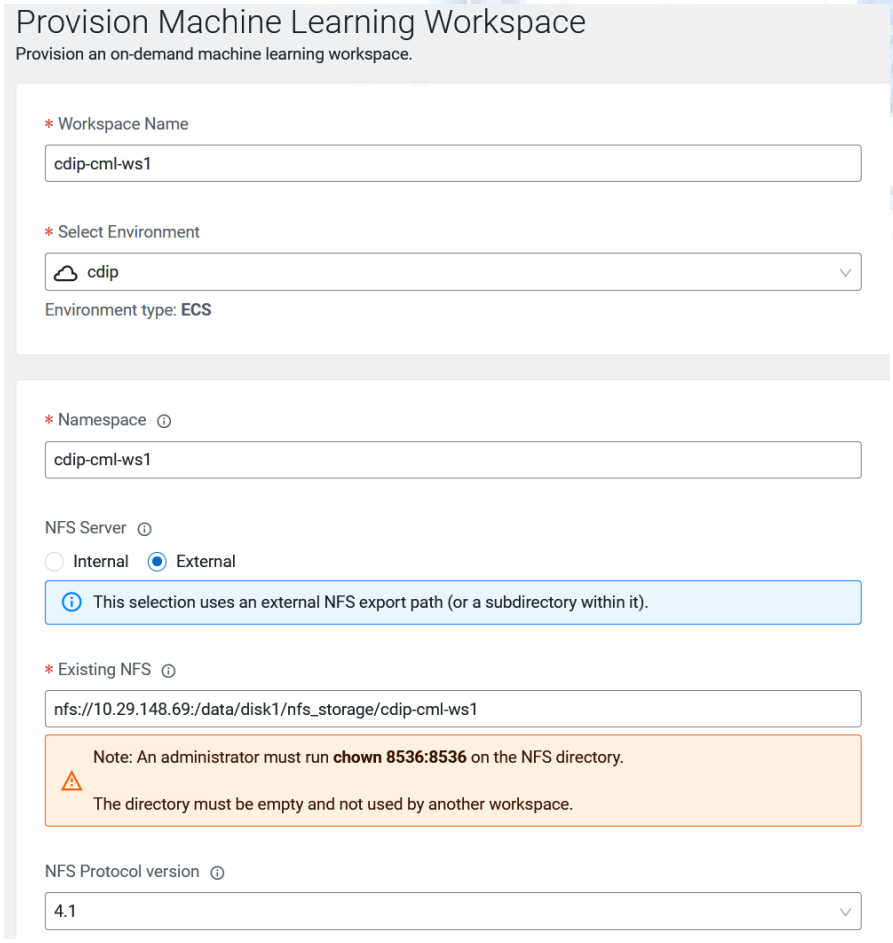


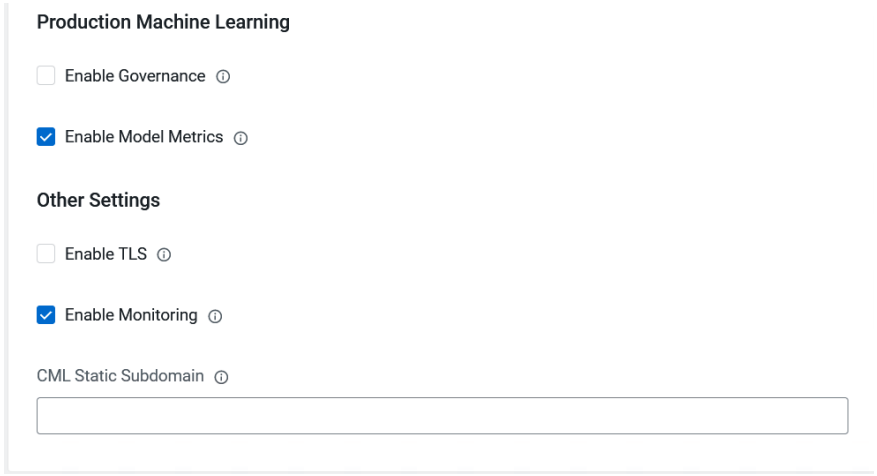
3. Provide input required to provision machine learning workspace. Click on provision workspace.

- Enter the configuration values for the workspace as described below.
  - **Workspace name**: A suitable name for the workspace.
  - **Environment**: Select the default environment from the drop down.
  - **Namespace**: This will be the kubernetes namespace under which the pods would be spinned up. By default, it is set to CAI. You can change it if you wish to.
  - **NFS server**: Select ***Internal***.
  - If you choose ***External NFS Server,*** perform on ***all ECS nodes***. **(Skip this, as we are not using it in current setup)**

```
#### nfs://172.31.30.239:/lhdata/nfs_storage/kuldeep-test-cml-w1
[root@pvcecs-master ~]# mkdir -p /lhdata/nfs_storage/kuldeep-test-cml-w1
[root@pvcecs-master ~]# chown 8536:8536 /lhdata/nfs_storage/kuldeep-test-cml-w1
```

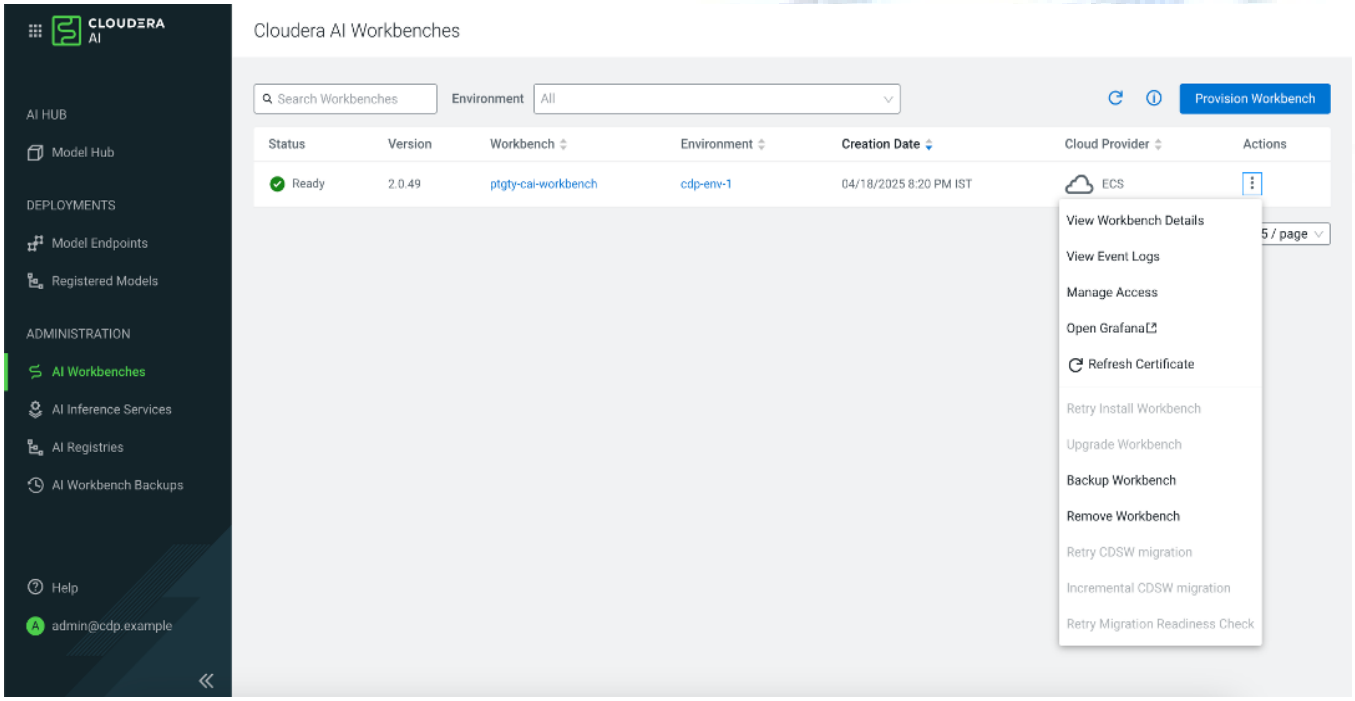
- Under Production Learning, the below parameters need to be updated.
  - **Enable Governance**: This provided advanced lineage and governance features. For simple demos or POCs, you may choose to disable it.
  - **Enable Model Metrics**: Keep it enabled. It provides you with the metrics.
  - **Enable TLS**: You can keep it disabled.
  - **Enable Monitoring**: This helps in monitoring the resource usage for the provisioned workspace. Enable it.
  - **CAI Static Subdomain**: Enter any short name for this parameter that helps in monitoring the resource usage for the provisioned workspace.



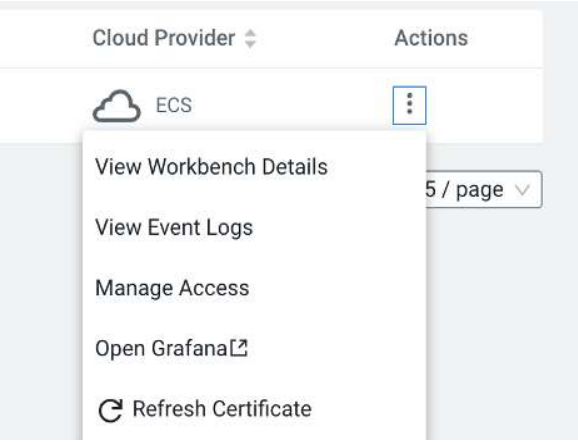


Note: Click on i icon to get more information on the field.

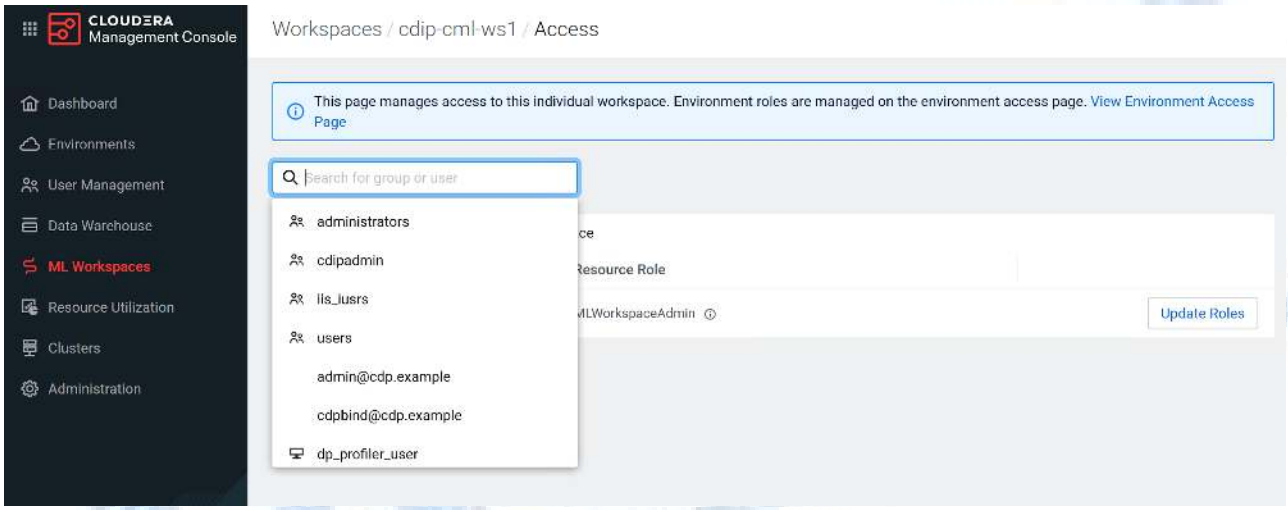
4. When provisioning of the workbench is completed the status reports as ***Ready***. Once it is created, it appears on the AI Workbenches page as shown below.



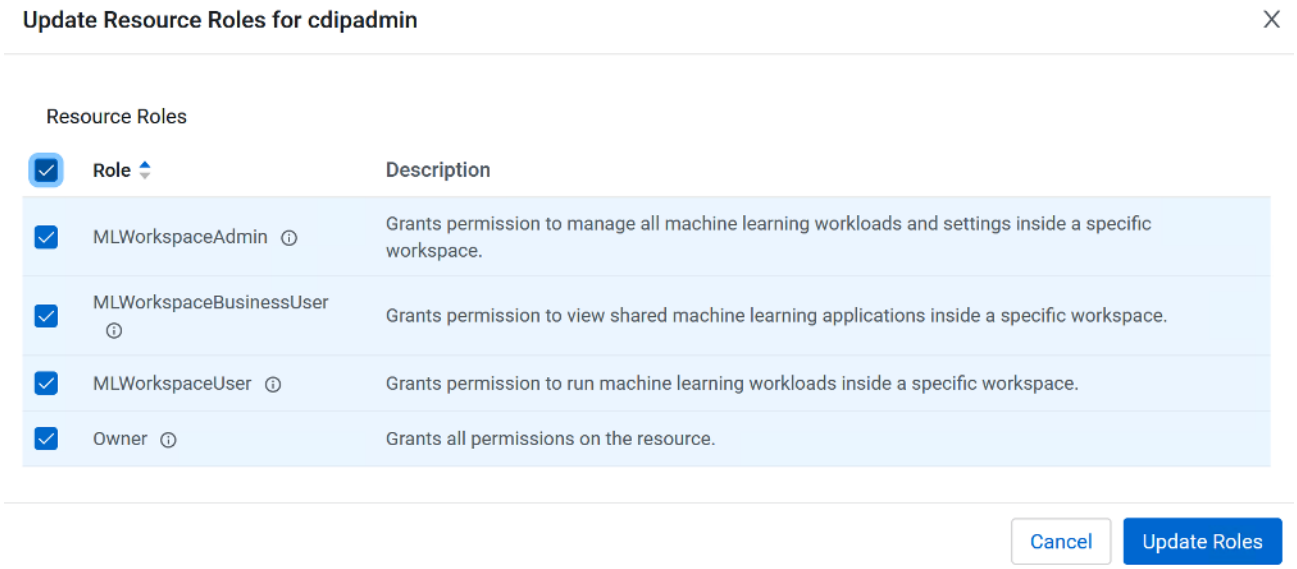
5. Click on Manage Access.



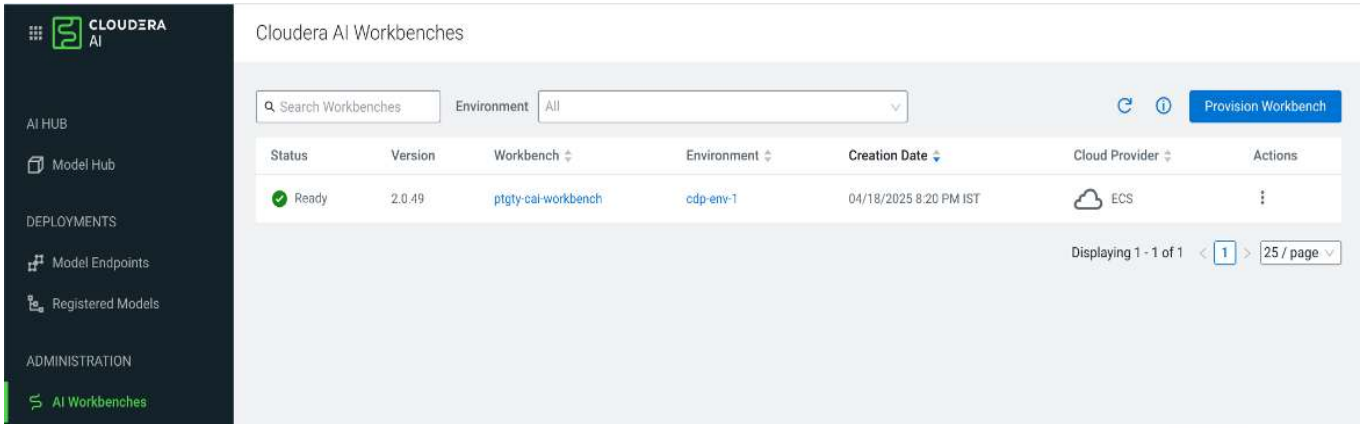
6. In the search field search for a user or group to be able to access AI Workbench.



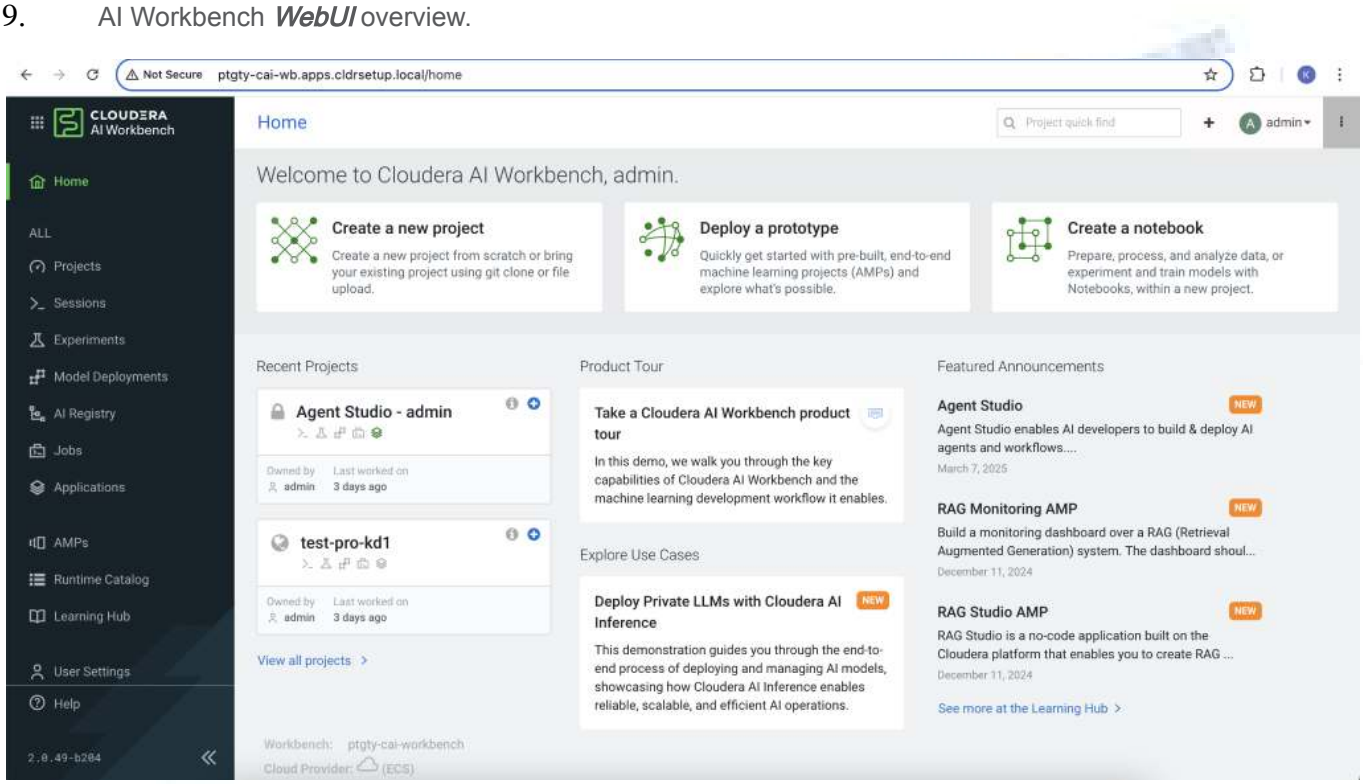
7. Update Resource role for user or group selected to manage access to workbench provisioned in Cloudera AI.



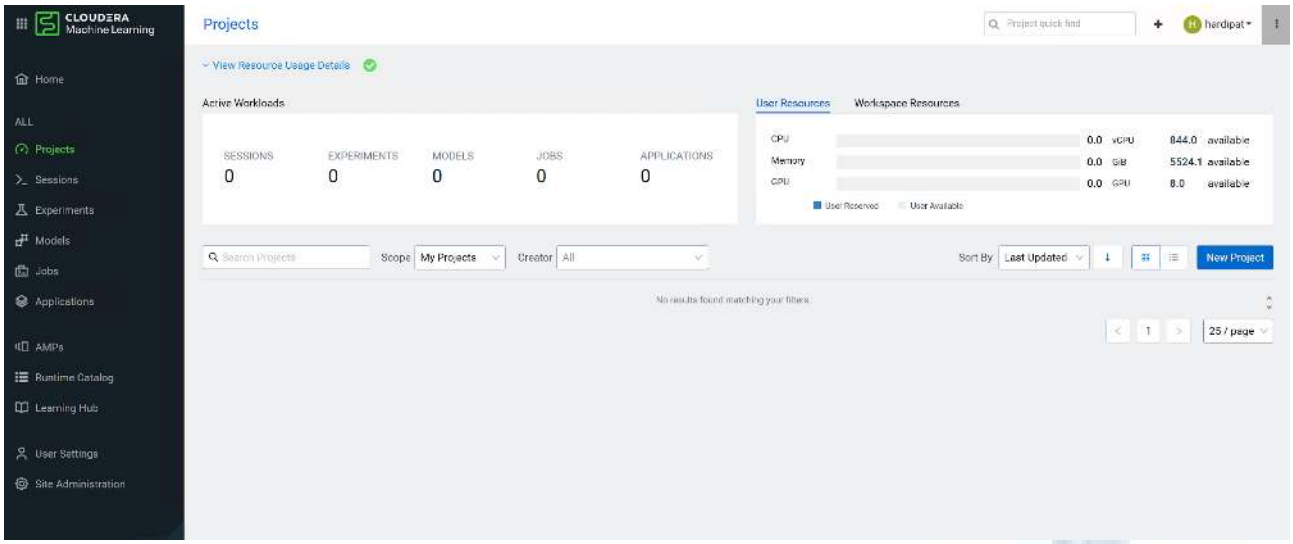
8. Click on the workbench name created.



9. AI Workbench ***WebUI*** overview.



10. Click on Projects tab, expand View Resource Usage Details to review available resources.



11. For more details and how to review projects section in ML workspace: https://docs.cloudera.com/machine-learning/cloud/projects/index.html

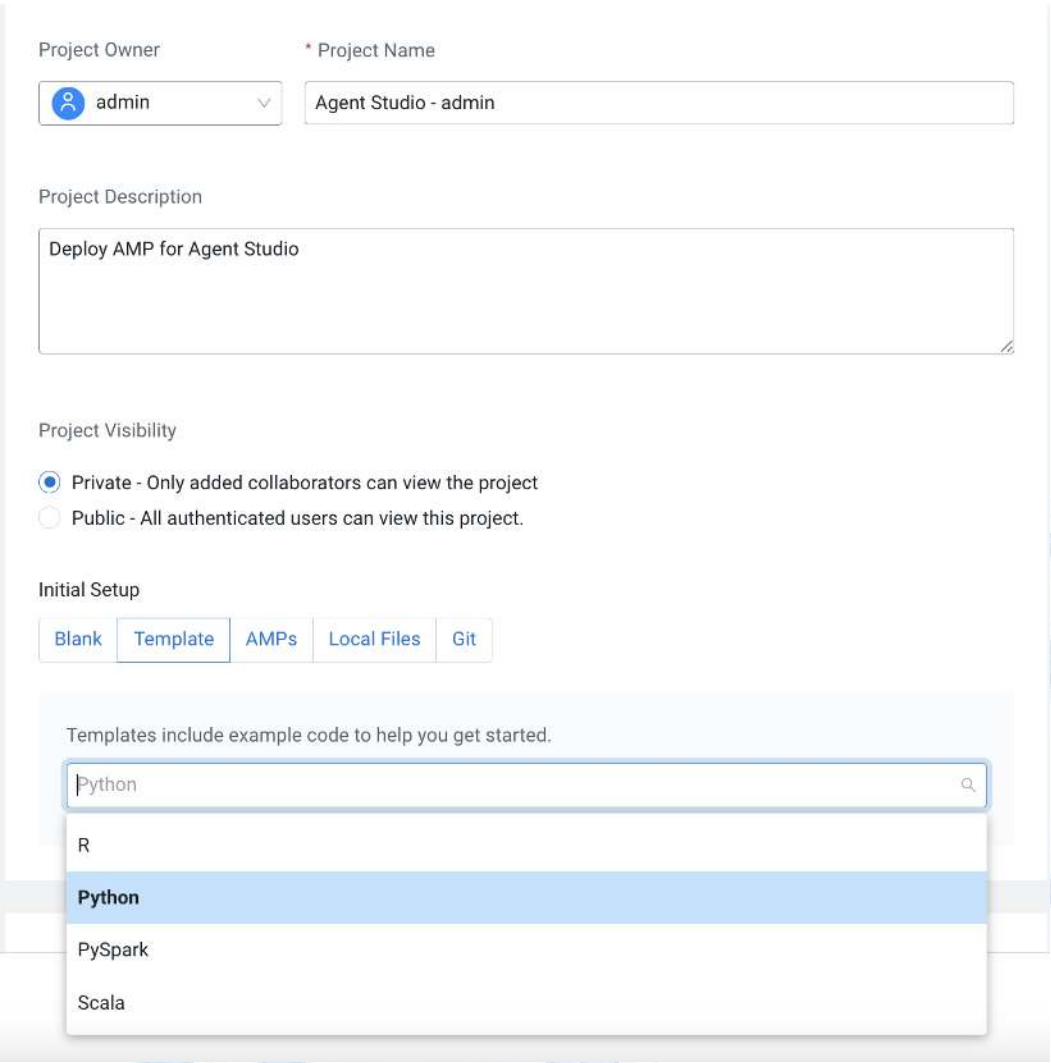
Creation of Project in AI Workbench:

12. At the middle right, you will find the New Project button. Click on it. New Project page appears. Enter the details as described below. Enter project name and select type of initial setup.

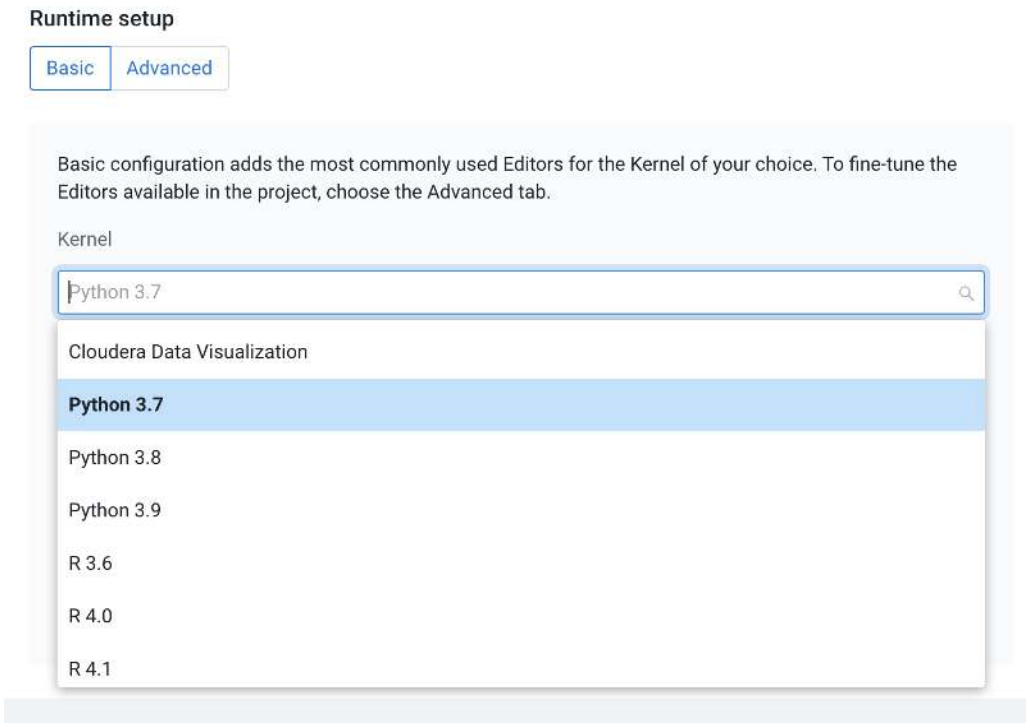
- **Project Name**: Enter a suitable name for your project.
- **Project Description**: Enter a description for the project.
- **Project Visibility**: Keep it Public for any demos or PoC's. If you are creating this in a multi-tenant environment, choose Private.

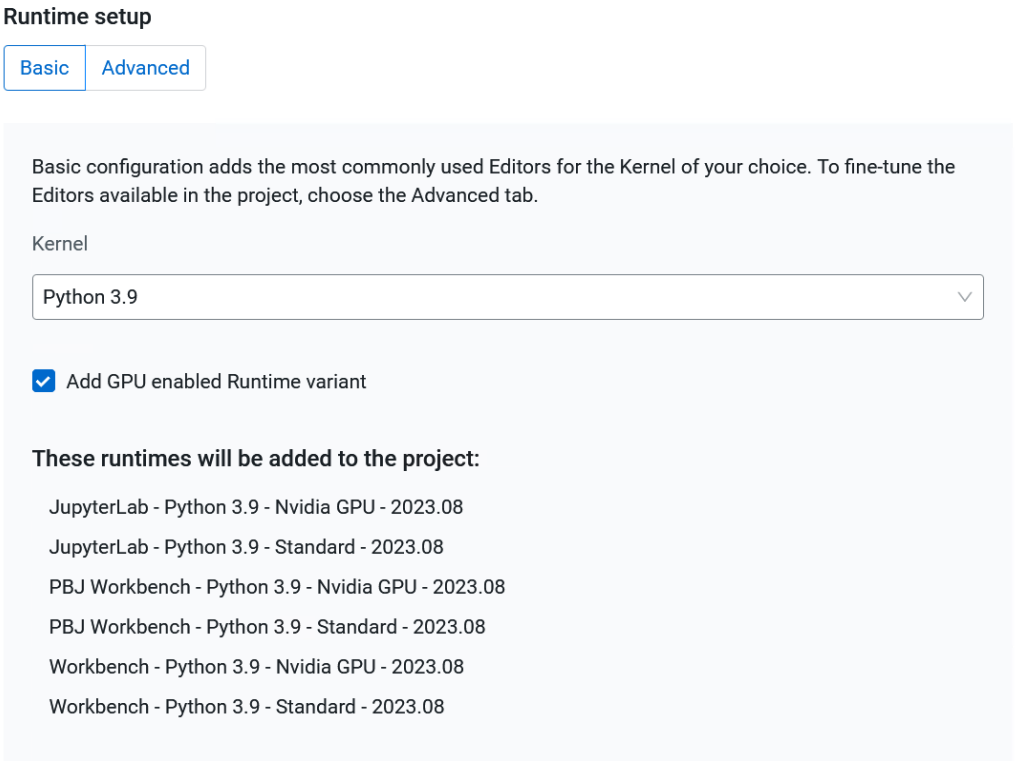
The Initial Setup Section for the New Project has five options as described below. Choose any of these based on your requirement.

- **Blank**: Choose this if you want to start from scratch.
- **Template**: Template projects contain example code that can help you get started with Cloudera AI. They are available in R, Python, PySpark, and Scala. Using a template project is not required, but it helps you start using Cloudera AI right away.
- **AMPs**: Applied ML Prototypes provide components to create a complete project. They may include jobs, models and experiments.
- **Local Files**: Choose this if you have all the necessary files in a folder or in a zip.
- **Git**: Choose this if all the resources are stored in a github project.



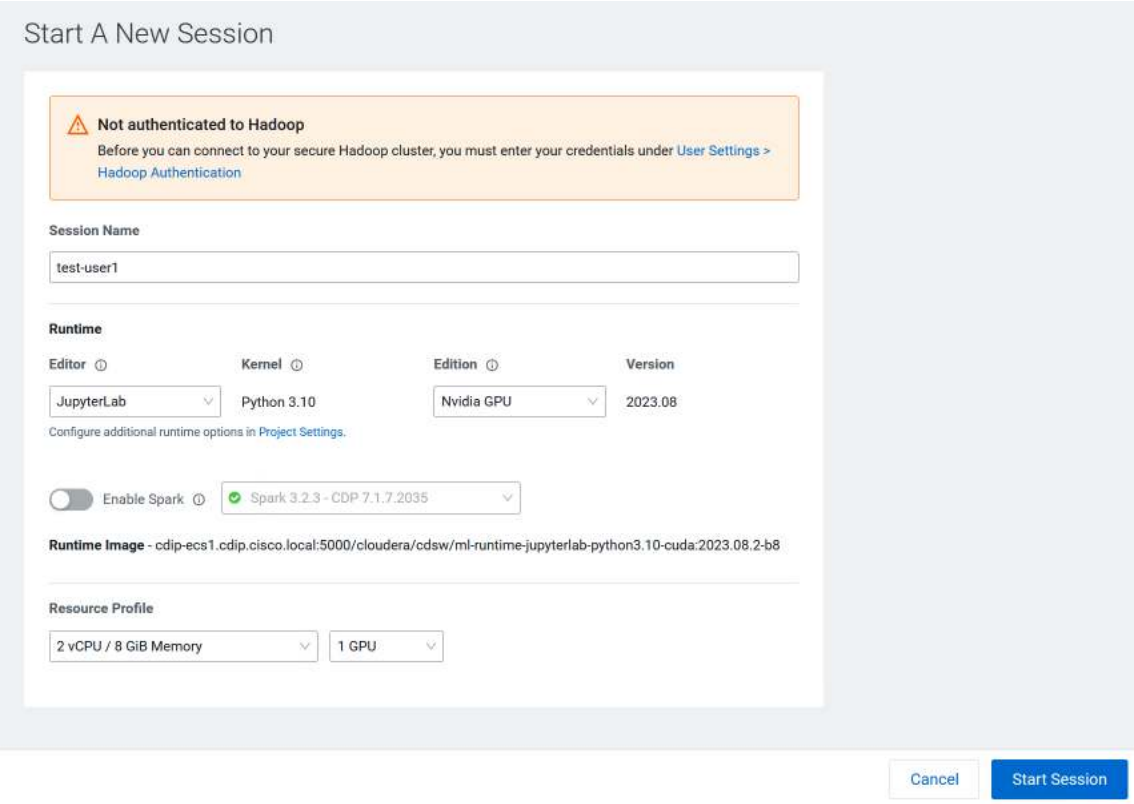
13. Select ***Runtime setup***. For initial exploration, select Basic and keep the kernel to ***Python3.9*** (enable checkbox to add GPU enabled Runtime variant, if applicable — **We are not using GPU in our current setup**).





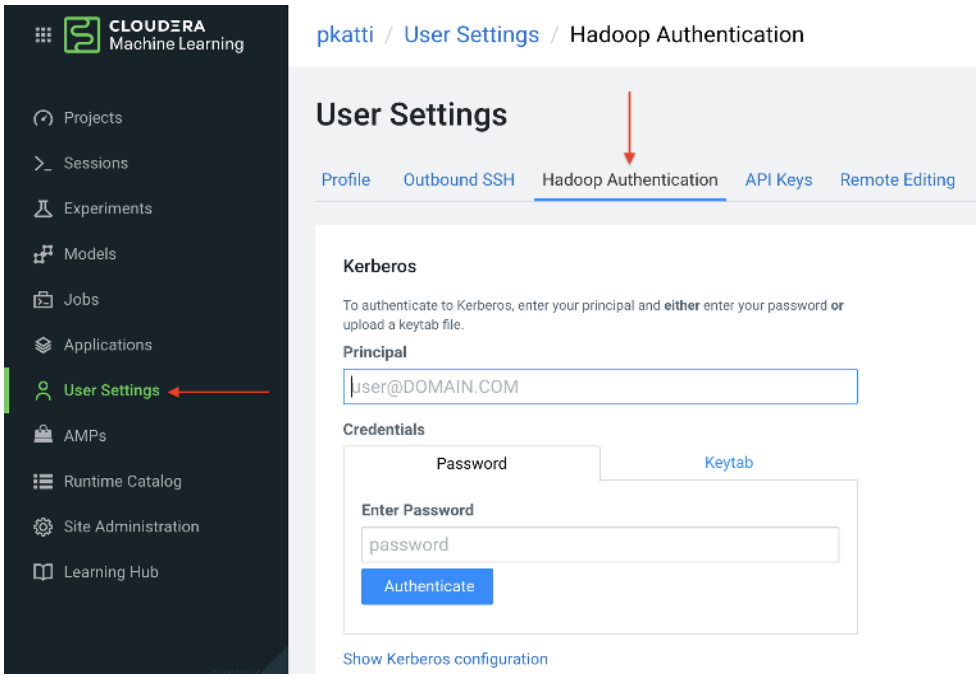
14. Click on ***Create Project***. After some time, a new project will be created and will be available on the Projects page.

15. Click on the sessions tab and enter details for the new session. You will see a warning like below:

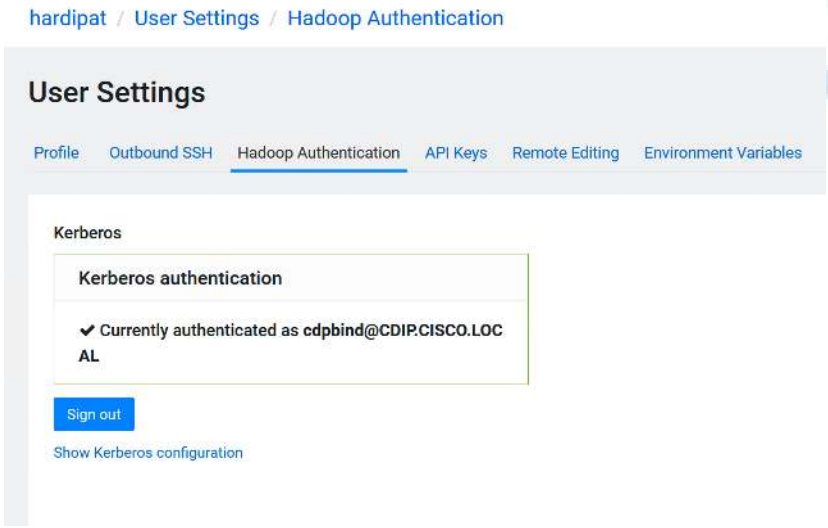


16. Before starting any new session in the recently created Project, you must complete the hadoop authentication part as the cluster setup is kerberized.

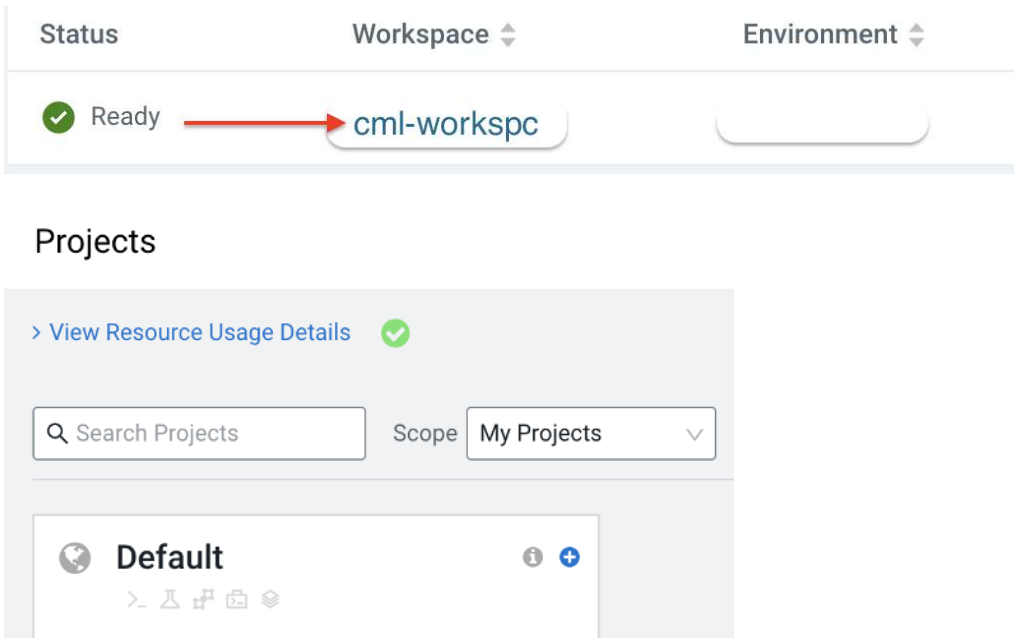
So, to be able to access data from Hadoop clusters ***go to user > user settings > Hadoop authentication.***
Open a ***new tab in the same browser*** window, by duplicating the existing tab. Go to the ***CAI home page*** and click on ***User Settings*** in the left pane. Click on ***Hadoop Authentication***.



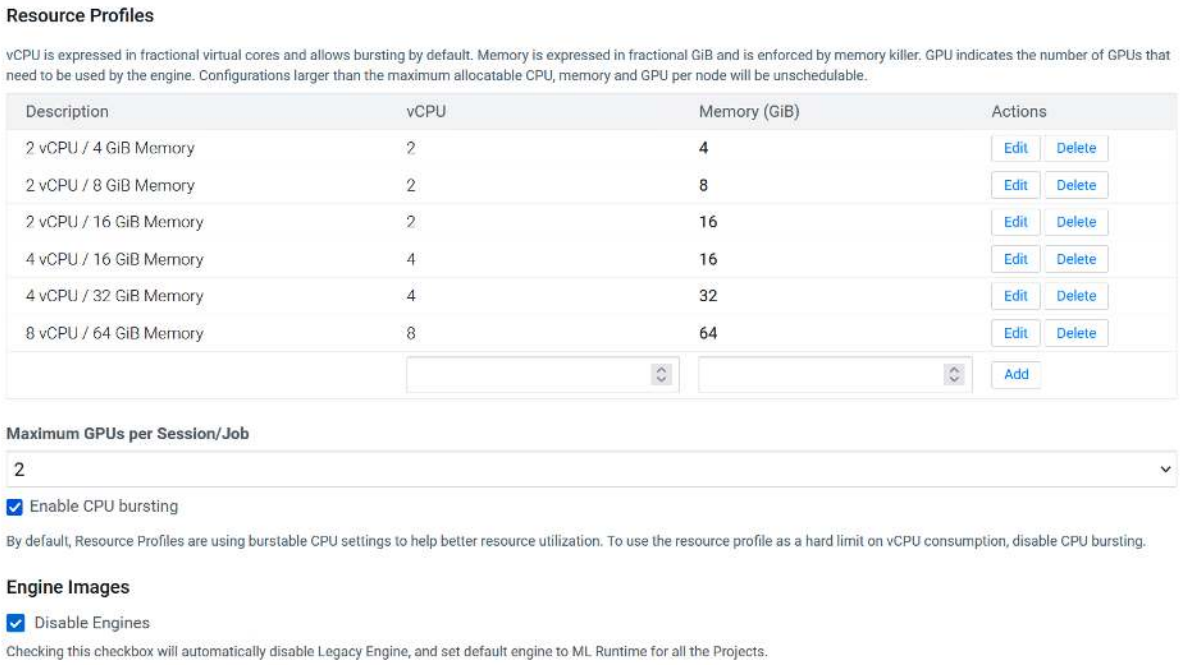
- Enter ***Principal*** e.g. username@DOMAIN.LOCAL i.e. ***admin@CLDRSETUP.LOCAL***
- Under the ***Credentials*** and password (i.e. ***cloudera123***) or keytab details of the LDAP user and click on Authenticate.
- Once the authentication is successful, proceed to the next step. You will see the output similar to below screenshot, after the successful authentication and integration to Kerberized Hadoop Cluster.



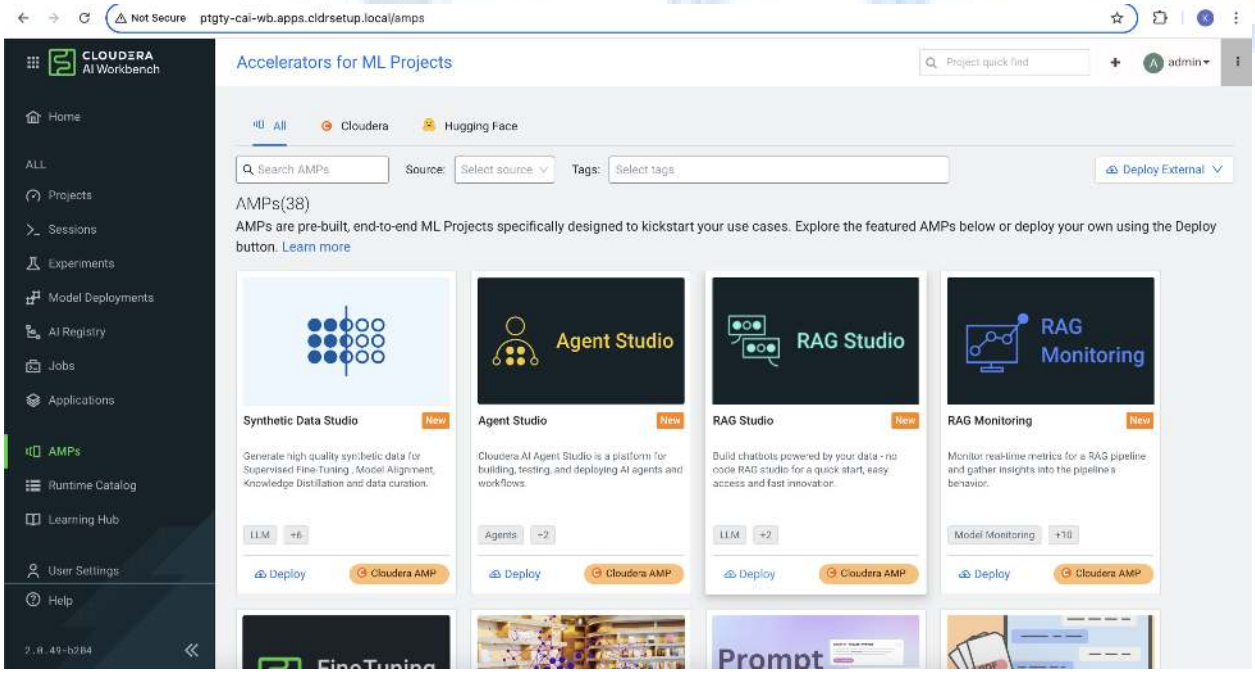
17. Now, to explore the CAI IDEs, click on the newly created workspace. It will open the Projects screen of CAI. Go to the ***Project page*** and click on the newly created project and then click on the ***New Session*** button on the top right. Explore CAI by running the jobs with different IDEs like Jupyterlab and Workbench.



18. Go to ***Site Administration*** to edit ***Resource profile*** and ***GPU per session/ Job***.

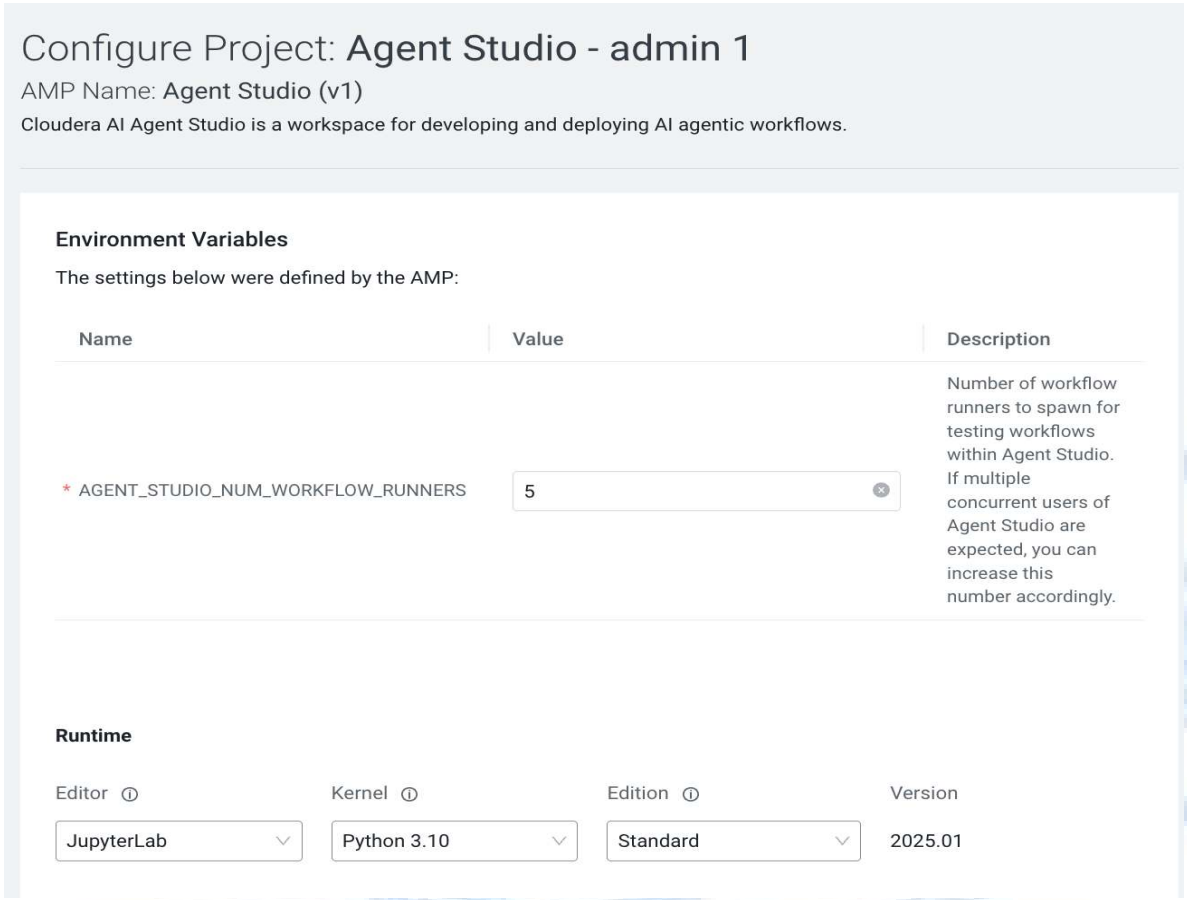


19. Go to the ***AMPs*** tab to get started with pre-built models.

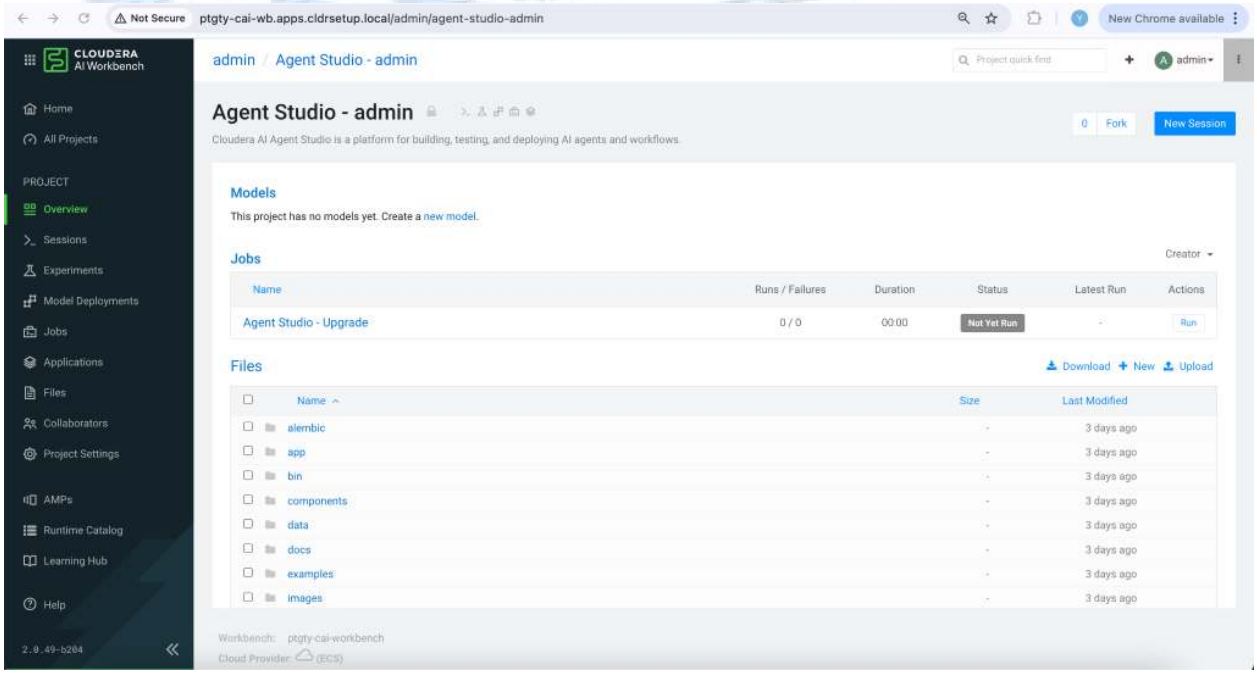


20. Select ***AMP*** and click on ***Configure Project***.

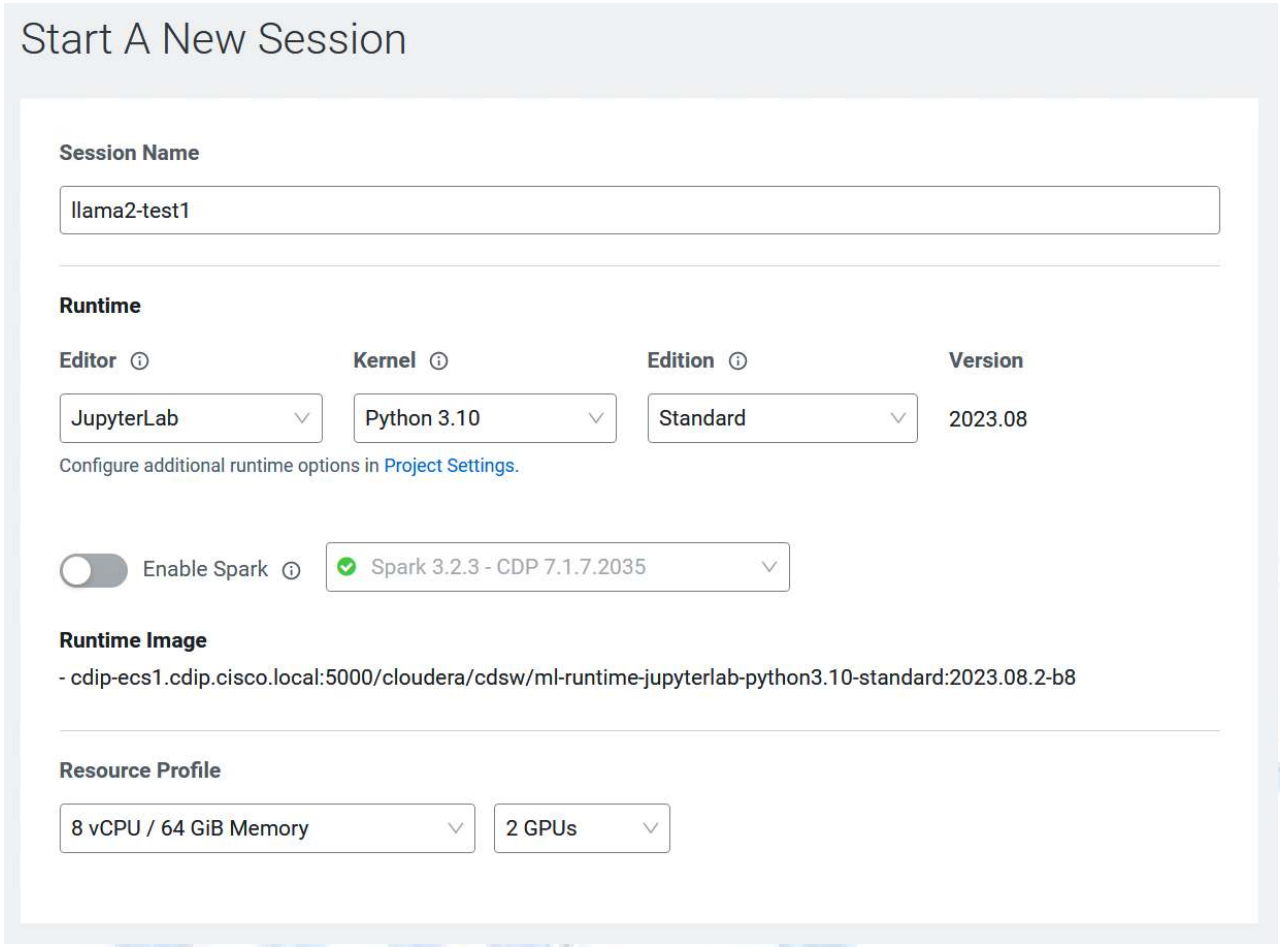
21. After editing the ***Runtime field*** for the new project, click on ***Launch Project.***



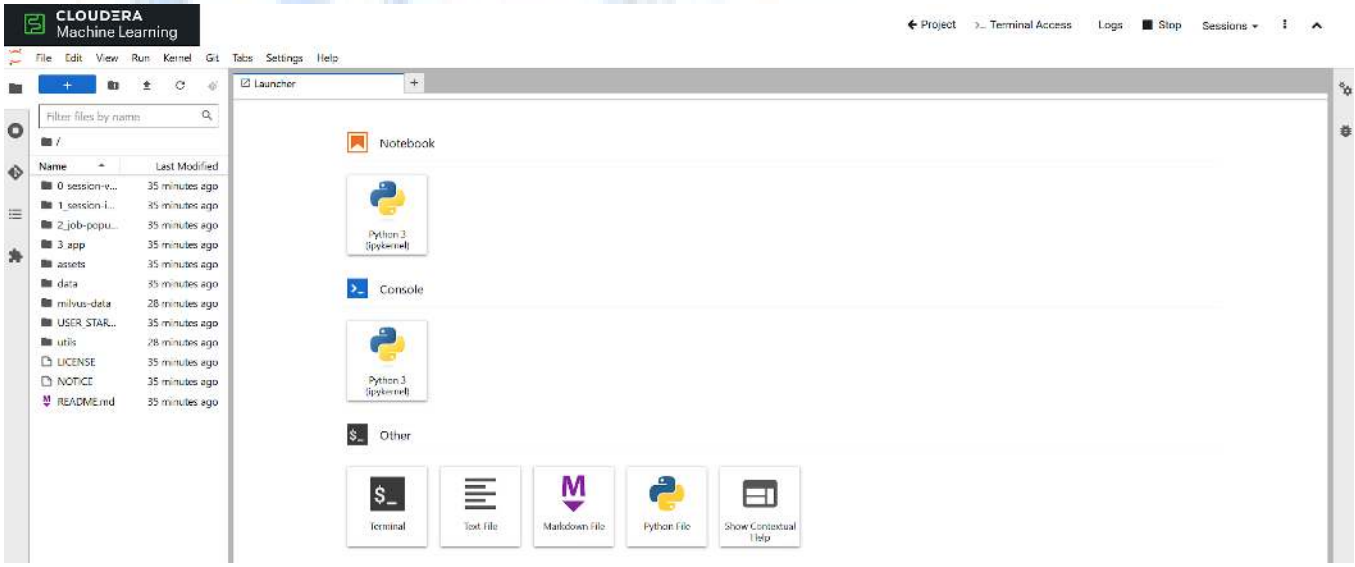
22. ***Agent Studio - admin – AMP*** project overview.



23. Create a new session by ***Start A New Session*** with desired resources, editor, kernel, and number of GPUs.

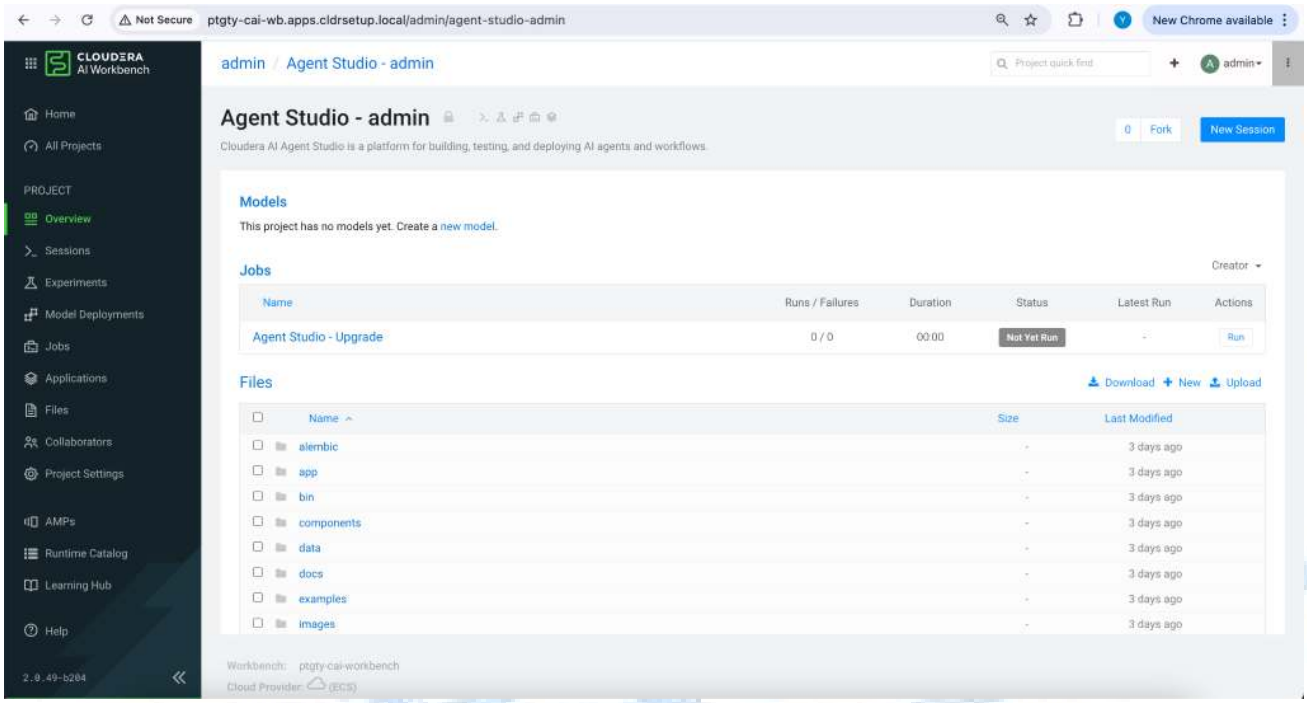


24. ***Jupyter notebook*** session in ***CAI***.

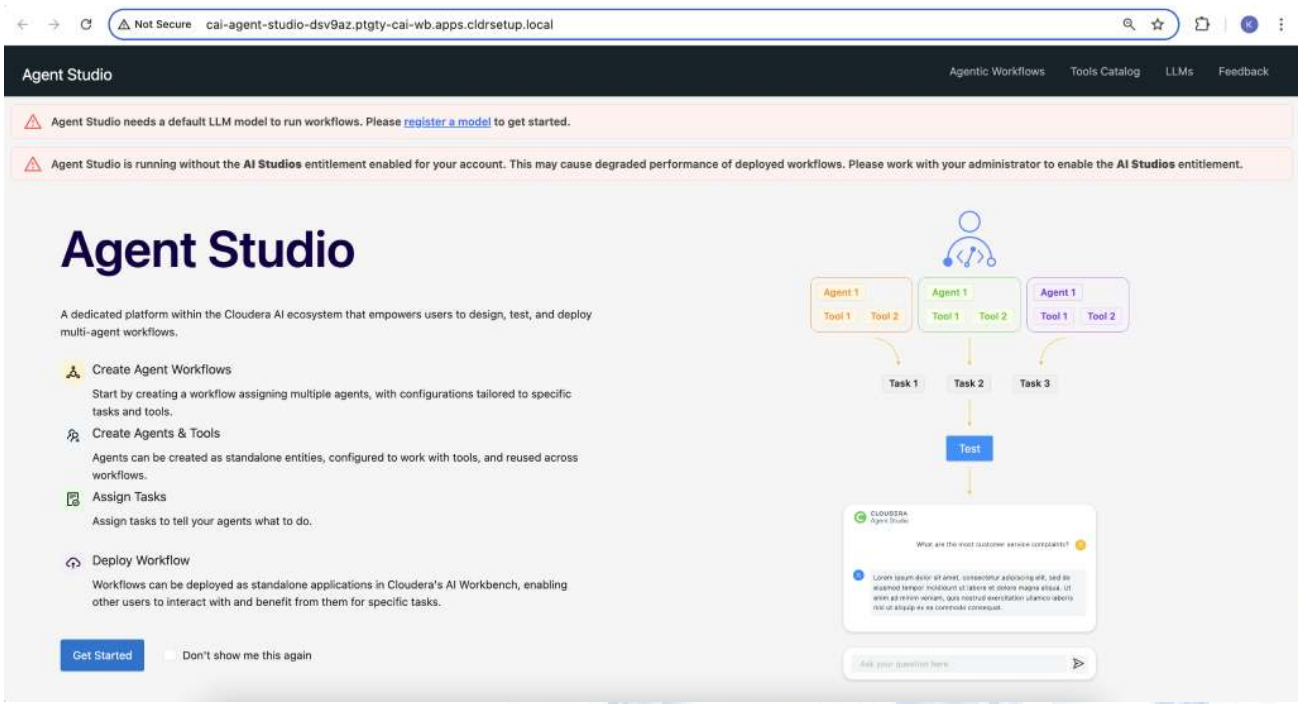


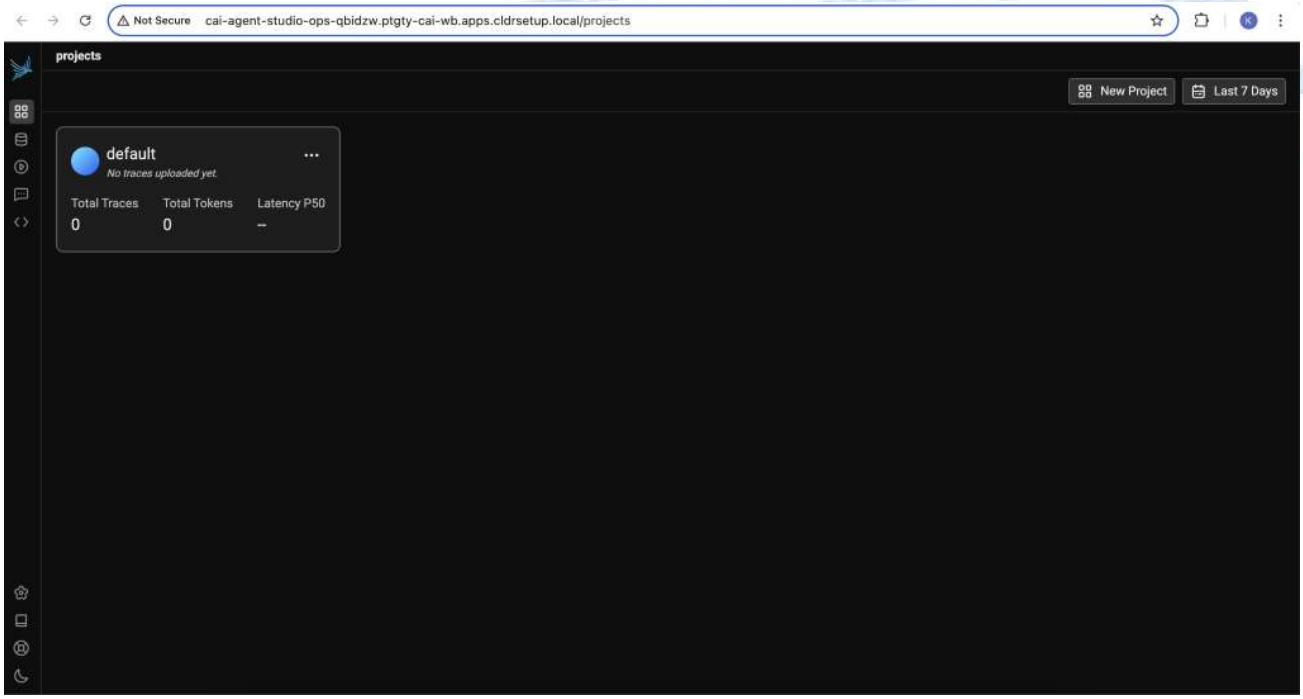
25. Open ***AMP*** created project. To access ***WebUI*** click on ***Open***.

Note: If you come across any new URLs for accessing a service's Web UI, make sure to add a hostname-to-IP mapping in your laptop's **`/etc/hosts`** file. Use the ECS master's IP if you're mapping for Data Services. For example: **`ptgty-cai-wb.apps.cldrsetup.local`** should map to the ECS master IP to be accessible via your browser.

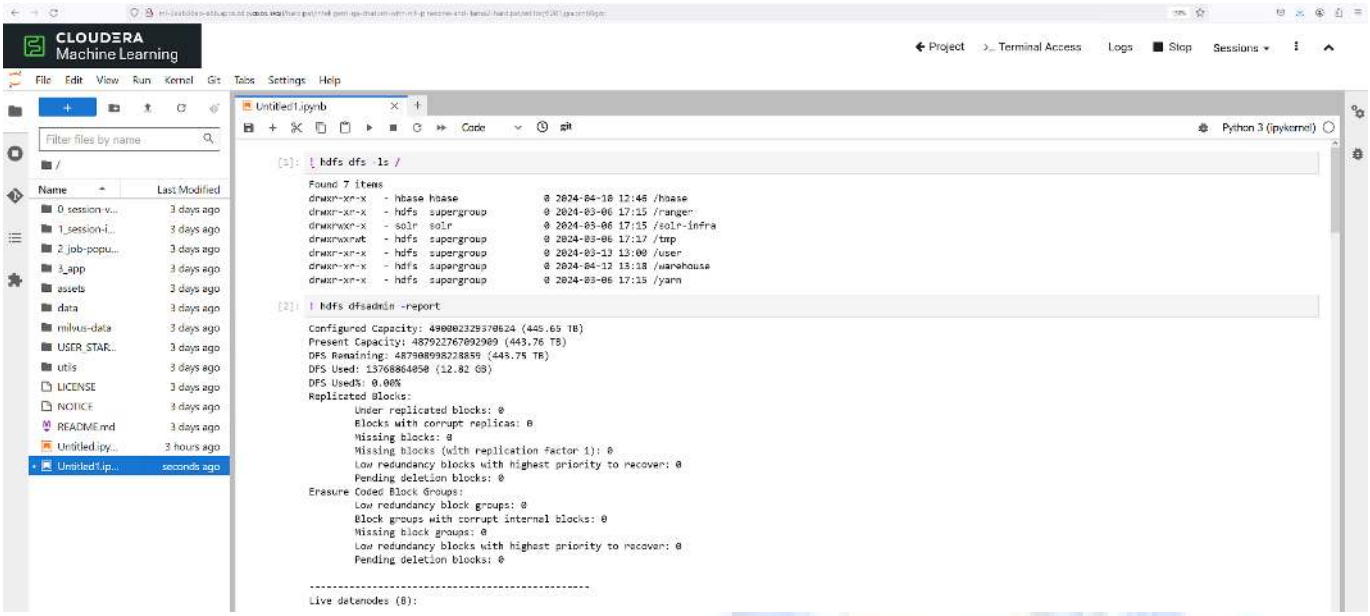


26. WebUI for ***Agent Studio - admin*** with pre-trained data is now available. ***Change settings*** on the WebUI or use them out of the box. For example, we questioned "***what is Cloudera Data Platform?***".





27. ***Access HDFS data*** from the ***jupyter notebook session*** in CAI.



Note: Deploying and documentation of every aspect of ***AI Workbench, project, and user management*** is not covered here. Please refer to the related ***Cloudera documentation*** on ***Cloudera AI How to section*** for more details:

https://docs.cloudera.com/machine-learning/cloud/product/topics/ml-product-overview.html

****

Cloudera on premises Data Warehouse (CDW)

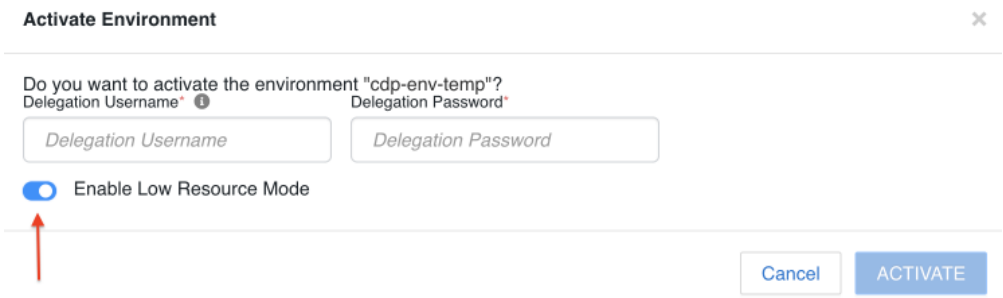
https://docs.cloudera.com/data-warehouse/1.5.5/private-cloud-getting-started/topics/dw-private-cloud-create-virtual-warehouse-openshift-overview.html

Enable CDW environment and creation of Database Catalog

- Open ***CDP Data Services*** page.
- Click on ***Data Warehouse***.

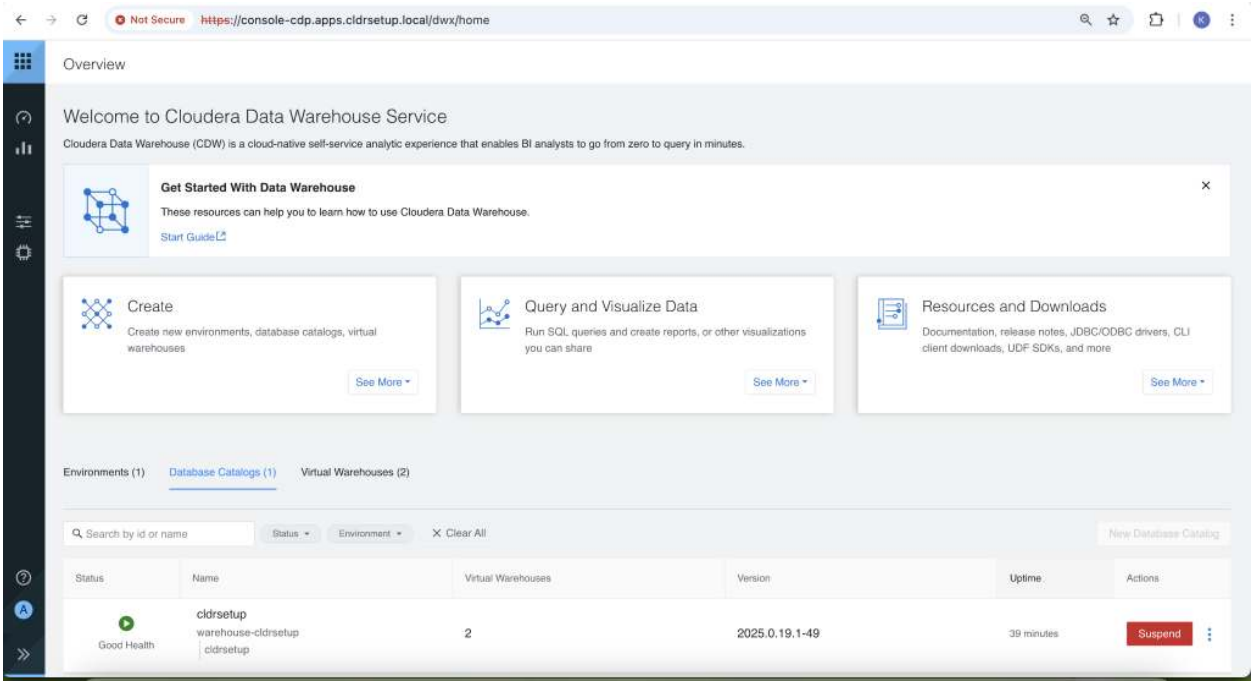
- On the ***Overview*** page, click on the ***Activate*** icon as shown below.

- On the ***Activate Environment*** page, enter the ***LDAP username and password***. Enable ***Low resource mode*** and click on ***Activate***.



Create Virtual Warehouse

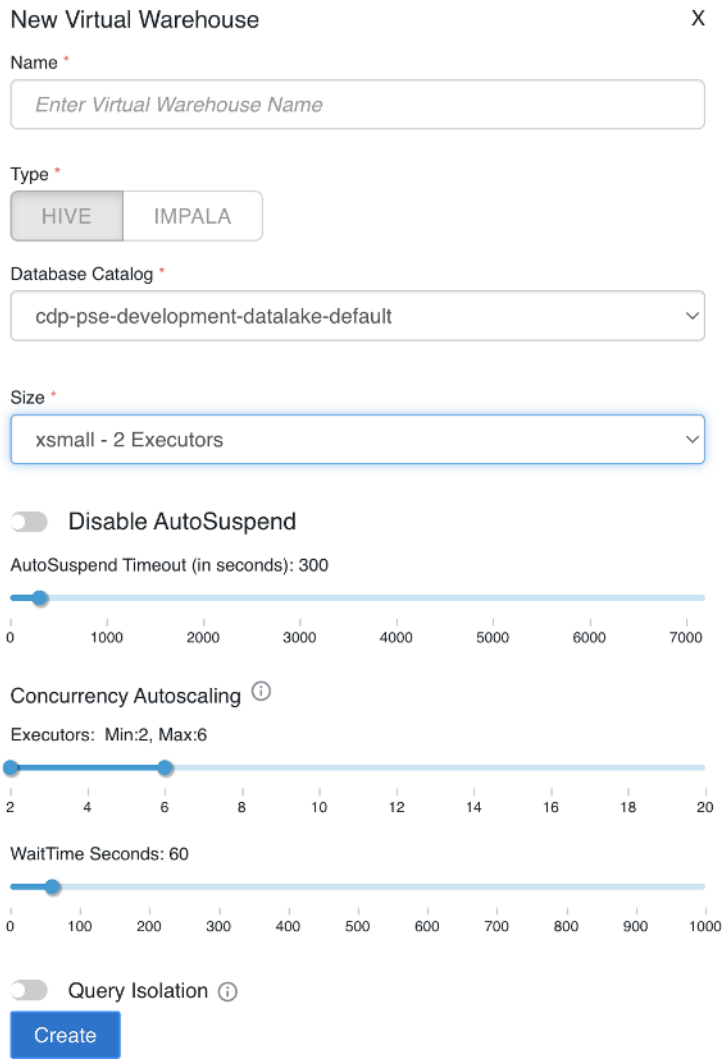
- Once the ***environment*** is ***activated***, a ***default Database Catalog*** gets created automatically.



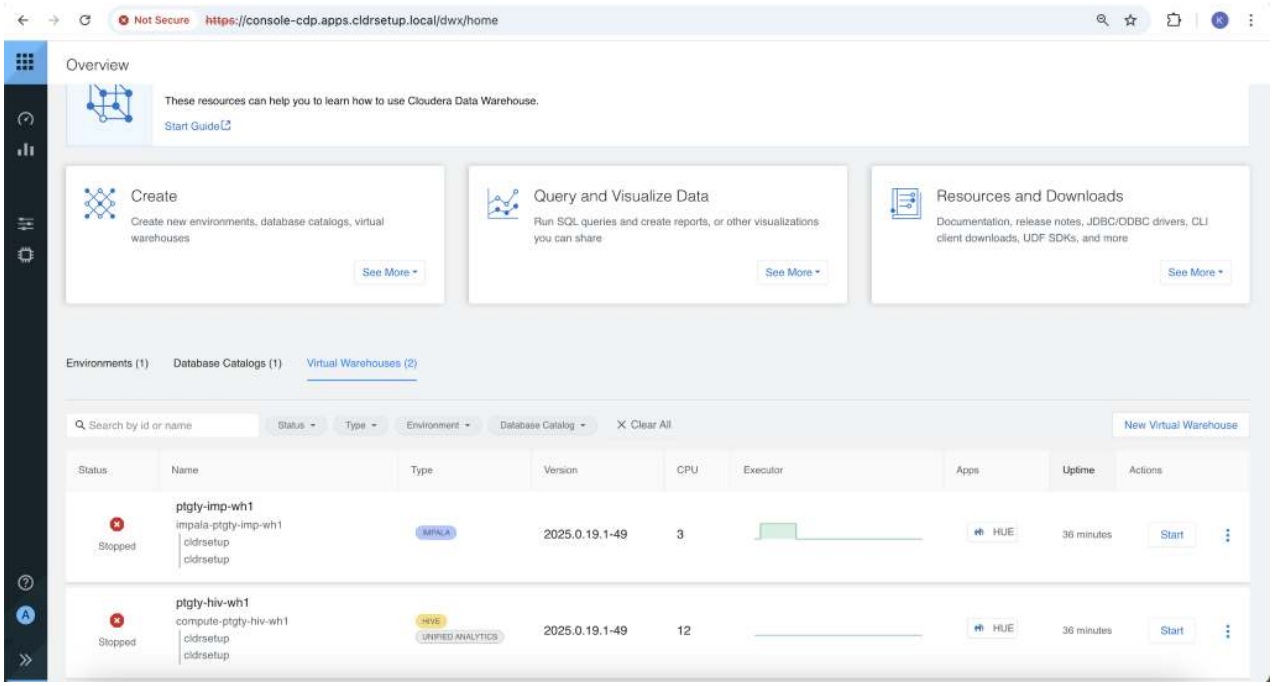
- Once the ***database catalog*** is created, click on **+** icon next to ***Virtual Warehouses***.

- A ***New Virtual Warehouse*** tab appears on the same page.
- Enter the ***name*** for the ***new virtual warehouse(***VW).
- Choose the ***type*** of VW, i.e. ***Hive*** or ***Impala***.
- Choose the ***default Database catalog*** that appears in the dropdown.
- Choose ***Size*** as ***xsmall-2 Executors***.
- ***AutoSuspend***: If you want the VW to keep running all the time, you can ***Disable*** it.

- Keep the remaining parameters ***default*** and click on ***Create***.



- A new ***Virtual Warehouse*** will be created. You can use ***Hue*** to submit queries to the underlying engine of the Virtual Warehouse.



****

Cloudera on premises Data Engineering (CDE)

https://docs.cloudera.com/data-engineering/1.5.5/enable-data-engineering/topics/cde-private-cloud-add-cde-service.html

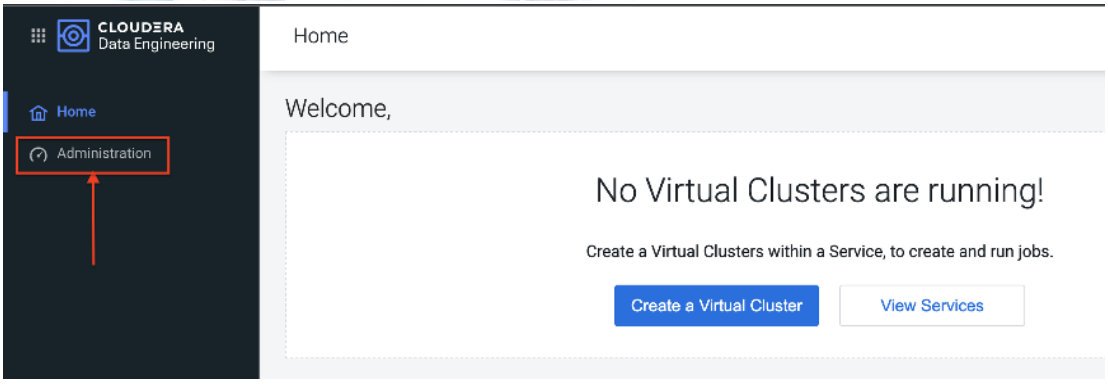
CDP Base cluster requirements:

The ***Cloudera Data Engineering (CDE)*** service requires proper configuration of ***Ozone*** service in the Base cluster. Ensure that ***Ozone*** is running properly otherwise you will end up with issues while enabling CDE.

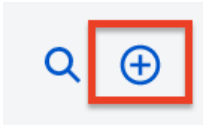
Enabling CDE Service:

- From the ***CDP console page***, click on ***Data Engineering***.

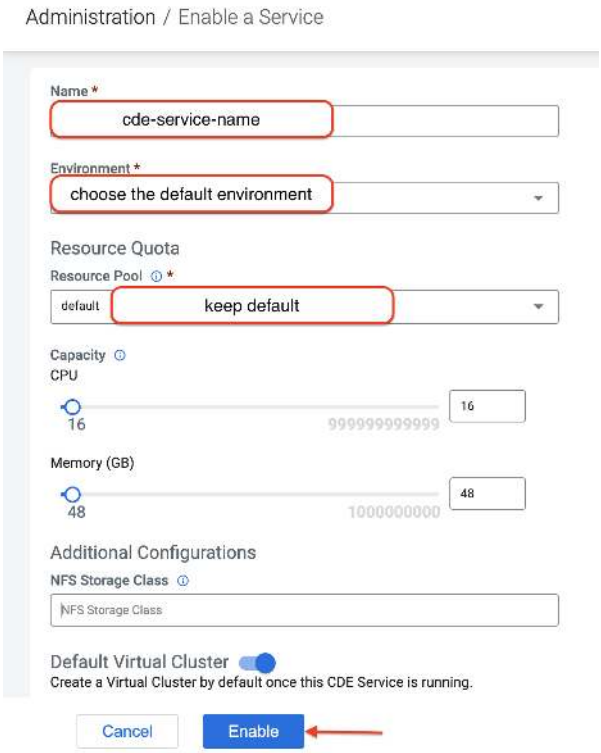
- This will open the ***CDE home page***. Since this will be the first time you will be opening CDE, you will not see any virtual clusters. Click on ***Administration*** in the left pane.



- Click on **+** icon as shown below which will allow you to ***enable CDE service*** post which you can ***create Virtual Clusters***.

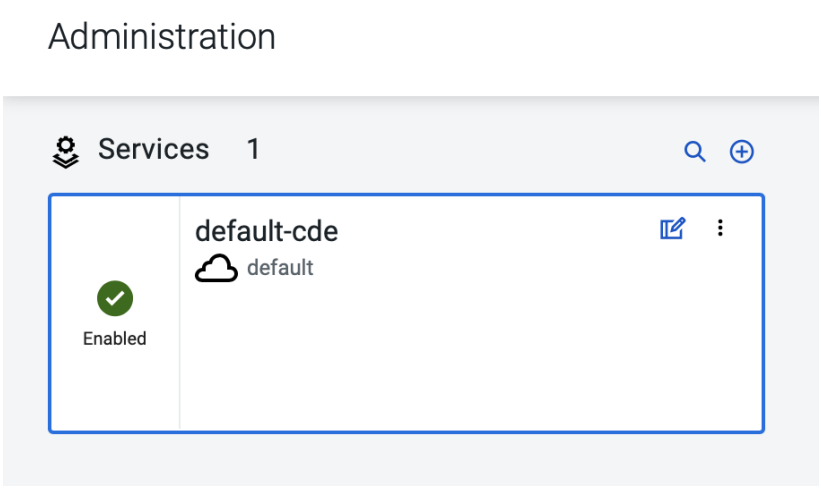


- On the ***Enable a Service*** page, enter the ***values*** as shown below and then click on ***Enable***.



Please note that the cpu and memory config chosen here are ***minimum values***. You can choose to increase it.

- This will take ***approximately 30 mins*** after which you will be able to see a ***CDE service*** on the ***CDE*** home page.

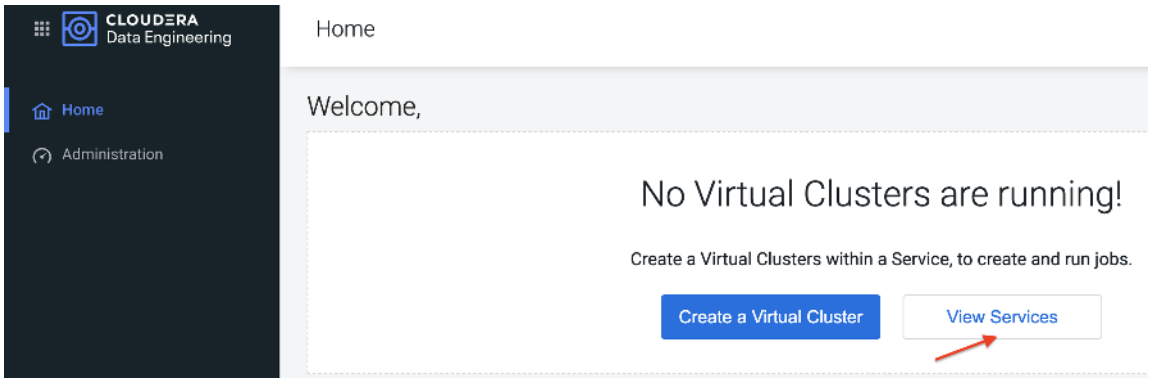


(*default **CDE** is the name given as an example. You will see as per the value you entered in the previous step.*)

- The ***CDE Home page*** displays the status of the ***CDE*** service initialization. You can view logs for the service by clicking on the service ***vertical ellipsis (three dots) menu***, and then clicking ***View Logs***.

https://docs.cloudera.com/management-console/1.5.5/private-cloud-environments/topics/mc-private-cloud-environment-register-ui.html

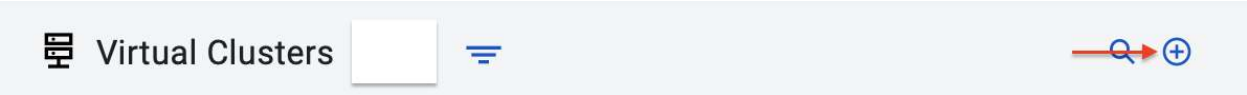
If you are unable to see the service, then the chances are that the default virtual cluster would not have been created properly. In this case, click on the ***View Services*** button and then you will be able to see the ***CDE*** service ***enabled***.



Create Virtual Cluster:

When you ***enable CDE service***, by default a new ***Virtual cluster with Spark2.4*** will be created. If you have not enabled this option earlier, then you need to create a virtual cluster again.

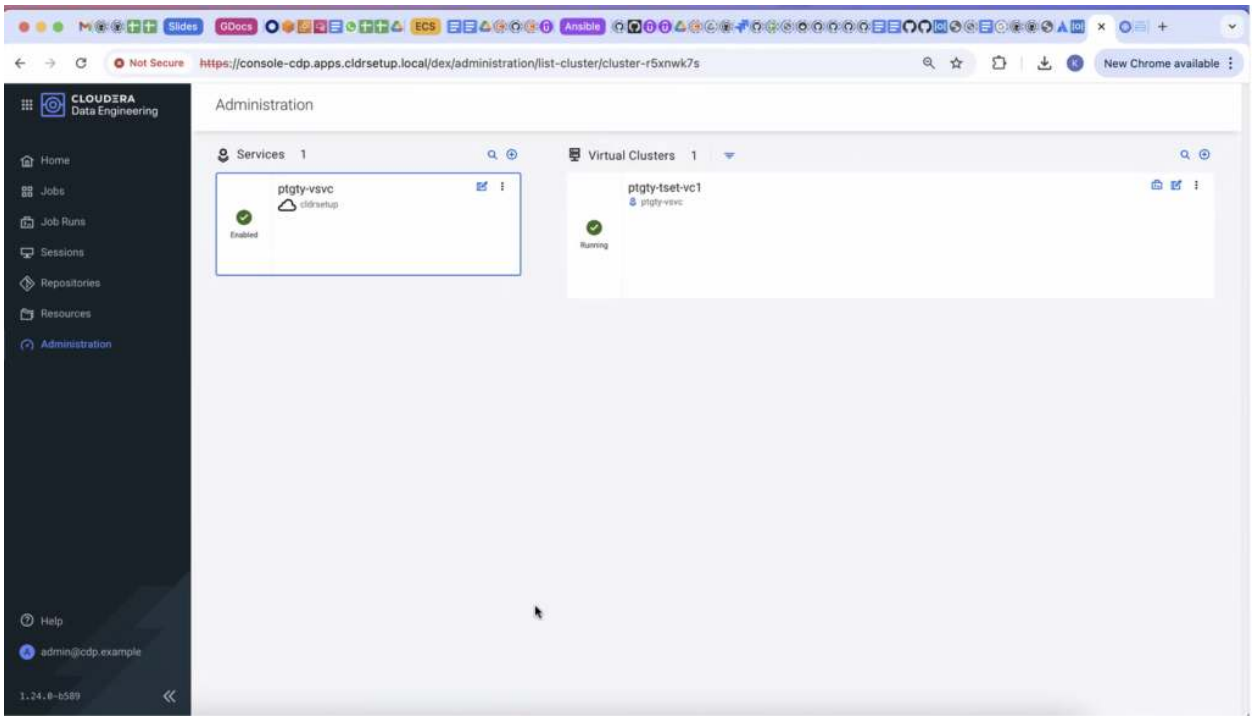
- On the ***CDE Home page***, click on the **+** icon next to ***Virtual clusters*** as shown below.

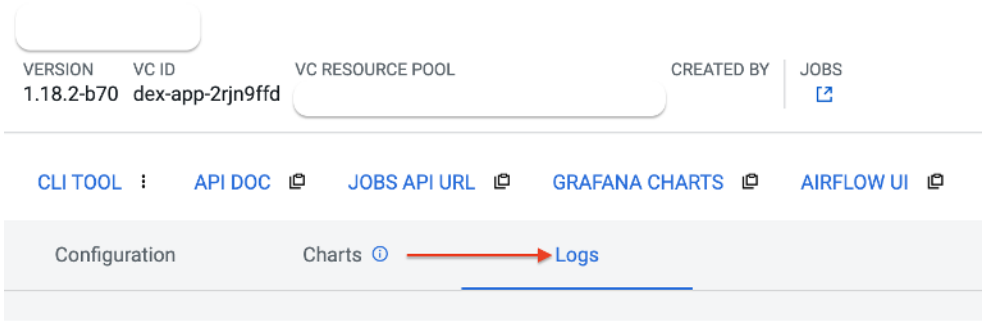


- On the ***Create a Virtual Cluster*** page, enter the below ***values*** and click on ***Create***.
  - ***Cluster Name:*** Cluster Name should adhere to the below conditions.
    - Begin with a letter
    - Be between 3 and 30 characters (inclusive)
    - Contain only alphanumeric characters and hyphens
  - ***Service***: Select the CDE service created earlier.
  - ***Spark Version***: Select the Spark version as per your requirement. If you need both ***Spark2.4*** and ***Spark3.7***, you can create two virtual clusters provided you have sufficient resources.

This will take approximately 20 minutes.

- You can check the logs of the cluster creation by clicking on the ***pencil*** icon and selecting the ***Logs*** section on the cluster page as shown below.





Initializing Virtual Cluster

Every time a ***new virtual cluster*** is created, there are a few ***manual steps*** that must be performed.

- Log in to the ***ECS master*** and run the next set of ***commands*** as per the instructions.
- Run the below command to create a temporary directory and navigate to the same.

```
[root@pvcecs-master ~]# mkdir -p /tmp/cde-latest && cd /tmp/cde-latest
```

- Download the script cdp-cde-utils using wget.

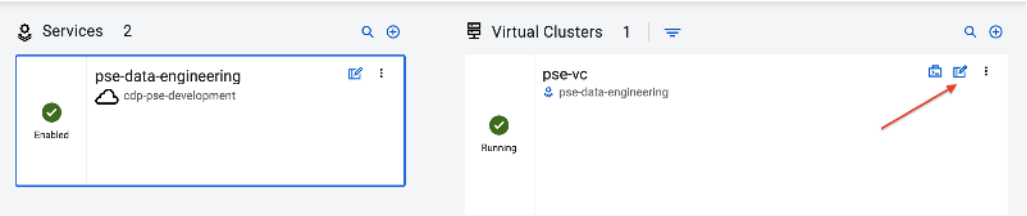
```
[root@pvcecs-master ~]# wget https://docs.cloudera.com/data-engineering/1.5.5/cdp-cde-utils.sh
```

- Add execute permission to this script.

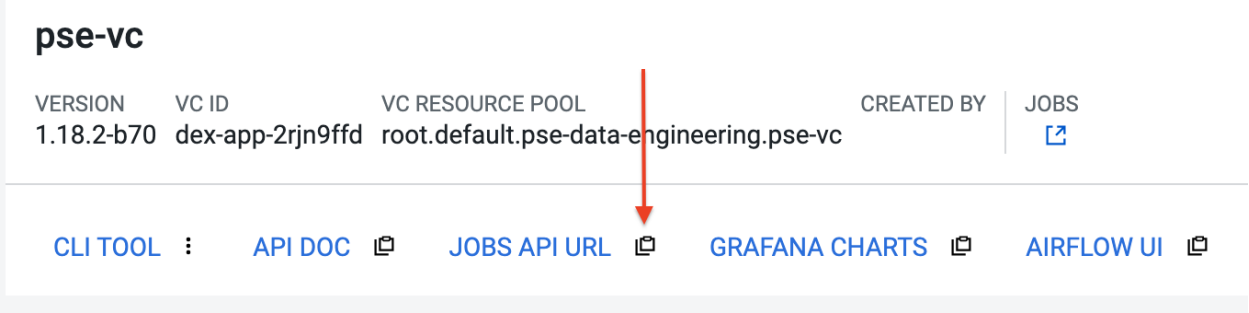
```
[root@pvcecs-master ~]# chmod +x /tmp/cde-latest/cdp-cde-utils.sh
```

- Identify the ***virtual cluster endpoint***:

On the ***CDE homepage***, select the ***CDE service*** in which the ***virtual cluster*** is created. Click on the ***pencil*** icon on the virtual cluster to be configured.



- Click JOBS API URL to copy the URL to your clipboard.



- Paste the URL into a text editor to identify the endpoint host. For example, if the URL is similar to the following:

```
http://dfdj6kgx.cde-2cdxw5x5.ecs-demo.example.com/dex/api/v1
```

Then the endpoint will then be as shown below.

```
dfdj6kgx.cde-2cdxw5x5.ecs-demo.example.com
```

- Once you get the endpoint of the virtual cluster, ***login to the ECS master*** and navigate to ***/tmp/cde-latest*** directory where the ***cdp-cde-utils.sh*** script is present.

```
[root@pvcecs-master ~]# cd /tmp/cde-latest
```

- Generate a self-signed certificate with the below command. Replace the endpoint_host with the endpoint of your virtual cluster that you got from the previous step.

```
[root@pvcecs-master ~]# ./cdp-cde-utils.sh init-virtual-cluster -h <endpoint_host> -a
```

For the example host we used above, this command will be as below.

```
[root@pvcecs-master ~]# ./cdp-cde-utils.sh init-virtual-cluster -h
dfdj6kgx.cde-2cdxw5x5.ecs-demo.example.com -a
```

- These steps must be performed for each virtual cluster you create.

Configuring LDAP Users on CDE

This step is required to submit the jobs to CDE from the LDAP users.

- Log in to the ECS master host and navigate to the directory /tmp/cde-latest.

```
[root@pvcecs-master ~]# cd /tmp/cde-latest
```

- Install krb5-workstation package using dnf.

```
[root@pvcecs-master ~]# dnf install krb5-workstation krb5-libs -y
```

- Create a file named ***<username>.principal*** containing the user principal. As an example, we will consider admin as the username. Here ***CLDRSETUP.LOCAL*** is the realm provided during IPA setup. You need to replace it with the realm you configured.

```
[root@pvcecs-master ~]# cat>> admin.principal
cdpuser@CLDRSETUP.LOCAL
```

- Generate a keytab named ***<username>.keytab*** for the user using ***ktutil***:

```
[root@pvcecs-master ~]# cat>> admin.keytab
[root@pvcecs-master ~]# sudo ktutil
ktutil:  addent -password -p admin@CLDRSETUP.LOCAL -k 1 -e aes256-cts
Password for admin@CLDRSETUP.LOCAL:
ktutil:  addent -password -p admin@CLDRSETUP.LOCAL -k 2 -e aes128-cts
Password for admin@CLDRSETUP.LOCAL:
ktutil:  wkt admin.keytab
ktutil:  q
```

- Validate the keytab using ***klist***. This command should use the principals created with two encryptions provided above, namely aes256-cts and aes128-cts.

```
[root@pvcecs-master ~]# klist -ekt admin.keytab
```

- Validate the ***keytab*** using ***kinit***. This command should get executed successfully.

```
[root@pvcecs-master ~]# kinit -kt admin.keytab admin@CLDRSETUP.LOCAL
```

- Make sure that the ***keytab*** is valid before continuing. If the ***kinit*** command fails, the user will not be able to run jobs in the ***virtual cluster***. After verifying that the ***kinit*** command succeeds, you can ***destroy*** the Kerberos ticket by running ***kdestroy***.
- Use the ***cdp-cde-utils.sh*** script to copy the user ***keytab*** to the virtual cluster hosts.

```
[root@pvcecs-master ~]# ./cdp-cde-utils.sh init-user-in-virtual-cluster -h <endpoint_host> -u <user> -p
<principal_file> -k <keytab_file>
```

For the above example, the command would be below.

```
[root@pvcecs-master ~]# ./cdp-cde-utils.sh init-user-in-virtual-cluster -h
dfdj6kgx.cde-2cdxw5x5.ecs-demo.example.com -u cdpuser -p cdpuser.principal -k cdpuser.keytab
```

- Repeat these steps for all users that need to submit jobs to the virtual cluster.

****

Appendix

This appendix contains the following:

Appendix A – References Used in Guide

Cloudera on premises Base Getting Started Guide:

https://docs.cloudera.com/cdp-private-cloud/latest/index.html

Cloudera on premises Data Services Getting Started Guide:

https://docs.cloudera.com/cdp-private-cloud-data-services/latest/index.html

Cloudera on premises Machine Learning Overview:

https://docs.cloudera.com/machine-learning/1.5.5/index.html

Cloudera on premises Data Engineering Overview:

https://docs.cloudera.com/data-engineering/1.5.5/index.html

Cloudera on premises Data Warehouse Overview:

https://docs.cloudera.com/data-warehouse/1.5.5/index.html

Appendix B – Glossary of Terms

This glossary addresses some terms used in this document, for the purposes of aiding understanding. This is not a complete list of all multi cloud terminology.

Ansible	An infrastructure automation tool, used to implement processes for instantiating and configuring IT service components, such as VMs on an IaaS platform. Supports the consistent execution of processes defined in YAML "playbooks" at scale, across multiple targets. Because the Ansible artifacts (playbooks) are text-based, they can be stored in a Source Code Management (SCM) system, such as GitHub. This allows for software development like processes to be applied to infrastructure automation, such as, Infrastructure-as-code (see IaC below). https://www.ansible.com
AWS (Amazon Web Services)	Provider of IaaS and PaaS. https://aws.amazon.com
Azure	Microsoft IaaS and PaaS. https://azure.microsoft.com/en-gb/
Containers (Docker)	A (Docker) container is a means to create a package of code for an application and its dependencies, such that the application can run on different platforms which support the Docker environment. In the context of aaS, microservices are typically packaged within Linux containers orchestrated by Kubernetes (K8s). https://www.docker.com https://www.cisco.com/c/en/us/products/cloud-systems-management/containerplatform/index.html
DevOps	The underlying principle of DevOps is that the application development and operations teams should work closely together, ideally within the context of a toolchain that automates the stages of development, test, deployment, monitoring, and issue handling. DevOps is closely aligned with IaC, continuous integration and deployment (CI/CD), and Agile software development practices. https://en.wikipedia.org/wiki/DevOps https://en.wikipedia.org/wiki/CI/CD

IaaS (Infrastructure as-a-Service)	Infrastructure components provided aaS, located in data centers operated by a provider, typically accessed over the public Internet. IaaS provides a base platform for the deployment of workloads, typically with containers and Kubernetes (K8s).
IaC (Infrastructure as-Code)	Given the ability to automate aaS via APIs, the implementation of the automation is typically via Python code, Ansible playbooks, and similar. These automation artifacts are programming code that define how the services are consumed. As such, they can be subject to the same code management and software development regimes as any other body of code. This means that infrastructure automation can be subject to all of the quality and consistency benefits, CI/CD, traceability, automated testing, compliance checking, and so on, that could be applied to any coding project. https://en.wikipedia.org/wiki/Infrastructure_as_code
IAM (Identity and Access Managemer	IAM is the means to control access to IT resources so that only those explicitly authorized to access given resources can do so. IAM is an essential foundation to a secure multi cloud environment. https://en.wikipedia.org/wiki/Identity_management
GCP (Google Cloud Platform)	Google IaaS and PaaS. https://cloud.google.com/gcp
Kubernetes (K8s)	Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications. https://kubernetes.io
Microservices	A microservices architecture is characterized by processes implementing fine-grained services, typically exposed via REST APIs and which can be composed into systems. The processes are often container-based, and the instantiation of the services is often managed with Kubernetes. Microservices managed in this way are intrinsically well suited for deployment into IaaS environments, and as such, are the basis of a cloud native architecture. https://en.wikipedia.org/wiki/Microservices
PaaS (Platform-as-a-Service)	PaaS is a layer of value-add services, typically for application development, deployment, monitoring, and general lifecycle management. The use of IaC with IaaS and PaaS is very closely associated with DevOps practices.
Private on-premises data center	A data center infrastructure housed within an environment owned by a given enterprise is distinguished from other forms of data center, with the implication that the private data center is more secure, given that access is restricted to those authorized by the enterprise. Thus, circumstances can arise where very sensitive IT assets are only deployed in a private data center, in contrast to using public IaaS. For many intents and purposes, the underlying technology can be identical, allowing for hybrid deployments where some IT assets are privately deployed but also accessible to other assets in public IaaS. IAM, VPNs, firewalls, and similar are key technologies needed to underpin the security of such an arrangement.
REST API	Representational State Transfer (REST) APIs is a generic term for APIs accessed over HTTP(S), typically transporting data encoded in JSON or XML. REST APIs have the advantage that they support distributed systems, communicating over HTTP, which is a well-understood protocol from a security management perspective. REST APIs are another element of a cloud-native applications architecture, alongside microservices. https://en.wikipedia.org/wiki/Representational_state_transfer
SaaS (Software-as-a-Service)	End-user applications provided “aaS” over the public Internet, with the underlying software systems and infrastructure owned and managed by the provider.
SAML (Security Assertion Markup Language	Used in the context of Single-Sign-On (SSO) for exchanging authentication and authorization data between an identity provider, typically an IAM system, and a service provider (some form of SaaS). The SAML protocol exchanges XML documents that contain security assertions used by the aaS for access control decisions. https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language

Terraform	An open-source IaC software tool for cloud services, based on declarative configuration files. https://www.terraform.io

Appendix C – Glossary of Acronyms

ACL—Access-Control List

AD—Microsoft Active Directory

API—Application Programming Interface

CDP – Cloudera Data Platform

Cloudera on premises – Cloudera Data Platform Private Cloud

Cloudera on premises DS – Cloudera Data Platform Private Cloud Data Services

CDW – Cloudera Data Warehouse

CAI – Cloudera AI a.k.a. Cloudera Machine Learning

CDE – Cloudera Data Engineering

CPU—Central Processing Unit

DC—Data Center

DHCP—Dynamic Host Configuration Protocol

DNS—Domain Name System

HA—High-Availability

ICMP— Internet Control Message Protocol

LAN—Local Area Network

MAC—Media Access Control Address (OSI Layer 2 Address)

MTU—Maximum Transmission Unit

NAT—Network Address Translation

OSI—Open Systems Interconnection model

RHEL – Red Hat Enterprise Linux

Syslog—System Logging Protocol

TCP—Transmission Control Protocol (OSI Layer 4)

UDP—User Datagram Protocol (OSI Layer 4)

URL—Uniform Resource Locator

VM—Virtual Machine

VPN—Virtual Private Network

Cloudera Data Platform Cloudera on premises latest release note, go to:
https://docs.cloudera.com/cdp-private-cloud-upgrade/latest/release-guide/topics/cdpdc-release-notes-links.html

Cloudera Data Platform Cloudera on premises Base Requirements and Supported Versions, go to:
https://docs.cloudera.com/cdp-private-cloud-upgrade/latest/release-guide/topics/cdpdc-requirements-supported-versions.html

Cloudera Data Platform Cloudera on premises Data Services installation on Embedded Container Service requirements and supported versions, go to:
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/index.html

****

FreeIPA Reference

https://www.devopszones.com/2020/03/how-to-add-freeipa-user-in-cli-and-web.html

Add users on FreeIPA

- Log in to the IPA server and run kinit with admin and enter the password: ***kinit admin***
- Run the below command to create a user. Enter the password in the password prompt.

ipa user-add --password <User Name>

```
[root@pvcbasemaster ~]# ipa
ipa: ERROR: Ticket expired
[root@pvcbasemaster ~]# kinit admin && klist -e
Password for admin@CLDRSETUP.LOCAL:
[root@pvcbasemaster ~]# ipa user-add --password kdsahu
First name: Kuldeep
  Last name: Sahu
User login [kdsahu]: kdsahu
Password:
Enter Password again to verify:
--------------------
Added user "kdsahu"
--------------------
  User login: kdsahu
  First name: Kuldeep
  Last name: Sahu
  Full name: Kuldeep Sahu
  Display name: Kuldeep Sahu
  Initials: KS
  Home directory: /home/kdsahu
  GECOS: Kuldeep Sahu
  Login shell: /bin/sh
  Principal name: kdsahu@CLDRSETUP.LOCAL
  Principal alias: kdsahu@CLDRSETUP.LOCAL
  User password expiration: 20240321113054Z
  Email address: kdsahu@cldrsetup.local
  UID: 971200008
  GID: 971200008
  Password: True
  Member of groups: ipausers
  Kerberos keys available: True
[root@pvcbasemaster ~]# su - kdsahu
Last login: Thu Mar 21 01:00:40 PDT 2024 on pts/0

# 1 Add the user 'cdptest'
ipa user-add cdptest \
    --first=cdp \
    --last=test \
    --password
# (You'll be prompted to set and confirm a password for cdptest.)

# 2 Authenticate as an IPA admin
kinit admin

# 3 Generate a keytab for cdptest@CLDRSETUP.LOCAL
ipa-getkeytab \
  --principal=cdptest@CLDRSETUP.LOCAL \
  --keytab=/tmp/cdptest.keytab

# 4 Verify the keytab contents
klist -k /tmp/cdptest.keytab
```

If instead you need a service principal named
cdptest (for example, to run as an SPN on a host), you would:
```
bash
# Create a service principal
ipa service-add cdptest/your-host.cldrsetup.local

# Pull its keytab
ipa-getkeytab \
  --principal=cdptest/your-host.cldrsetup.local@CLDRSETUP.LOCAL \
  --keytab=/tmp/cdptest.keytab

# Verify
klist -k /tmp/cdptest.keytab
```

command to get the principal file and key tab file with help of that
Here are the commands to list the principals in a keytab and extract individual keytab entries:
```
bash
# 🔍 1. List all principals in the keytab
klist -k /tmp/cdptest.keytab

# Example output:
# Keytab name: FILE:/tmp/cdptest.keytab
# KVNO Principal
# ---- --------------------------------------------------------
#    1 cdptest@CLDRSETUP.LOCAL

# 🎯 2. Extract a specific principal's keys into a new keytab
#     (useful if you want to split out one principal from a multi-principal keytab)
ktutil <<EOF
  rkt /tmp/cdptest.keytab
  l
  wkt /tmp/cdptest-single.keytab Principal:cdptest@CLDRSETUP.LOCAL
  quit
EOF

# 🔐 3. Verify the newly created single-principal keytab
klist -k /tmp/cdptest-single.keytab
klist -k
 shows you all principals stored in a keytab file.
ktutil
 can read (
rkt
) an existing keytab, list (
l
) its entries, and write (
wkt
) only the entries matching a given principal into a new keytab.
This way you can both inspect and slice your keytab as needed.
```

Free-IPA Command Reference:

```
kinit admin
ipa dnsrecord-add 16.172.in-addr.arpa. 231.31 --ptr-rec console-cdp.apps.pvcecsmaster.cldrsetup.local.
ipa dnsrecord-del 16.172.in-addr.arpa. 231.31 --ptr-rec console-cdp.apps.pvcecsmaster.cldrsetup.local
ipa dnsrecord-find 16.172.in-addr.arpa.
ipa dnsrecord-add cldrsetup.local *.apps
ipa dnszone-list
ipa user-del cmadmin
ipa user-show
ipa status | start | stop
ipactl status | stop | start | restart
```

```
[kuldeep@pvcbasemaster ~]$ kinit kdsahu
Password for kdsahu@CLDRSETUP.LOCAL:
Password expired.  You must change it now.
Enter new password:
```

```
Enter it again:
[kuldeep@pvcbasemaster ~]$
```

```
[root@ipaserver ~]# ldapsearch  -H ldap://ipaserver.cldrsetup.local:389 -D
"uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local" -w 'cloudera123' -b
"cn=users,cn=accounts,dc=cldrsetup,dc=local" '(&(uid=admin))' | grep -v "#"

dn: uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
objectClass: top
objectClass: person
objectClass: posixaccount
objectClass: krbprincipalaux
objectClass: krbticketpolicyaux
objectClass: inetuser
objectClass: ipaobject
objectClass: ipasshuser
objectClass: ipaSshGroupOfPubKeys
uid: admin
krbPrincipalName: admin@CLDRSETUP.LOCAL
cn: Administrator
sn: Administrator
uidNumber: 971200000
gidNumber: 971200000
homeDirectory: /home/admin
loginShell: /bin/bash
gecos: Administrator
ipaUniqueID: e42d6b54-e094-11ee-9c71-0050568db389
memberOf: cn=admins,cn=groups,cn=accounts,dc=cldrsetup,dc=local
memberOf: cn=Replication Administrators,cn=privileges,cn=pbac,dc=cldrsetup,dc=c
 om
memberOf: cn=Add Replication Agreements,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=Read Replication Agreements,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=Modify DNA Range,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=Read LDBM Database Configuration,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=Host Enrollment,cn=privileges,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=System: Add krbPrincipalName to a Host,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=System: Enroll a Host,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=System: Manage Host Enrollment Password,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=System: Manage Host Keytab,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=System: Manage Host Principals,cn=permissions,cn=pbac,dc=cldrsetup,dc=local
memberOf: cn=trust admins,cn=groups,cn=accounts,dc=cldrsetup,dc=local
krbLastPwdChange: 20240312172405Z
krbPasswordExpiration: 20240610172405Z
krbExtraData:: AAK1j/Blcm9vdC9hZG1pbkBDRFBQVkNEUy5DT00A
krbLoginFailedCount: 0
krbLastFailedAuth: 20240325064805Z

search: 2
result: 0 Success
[root@ipaserver ~]#
```

```
 [root@pvcecsmaster ~]# ipa dnsrecord-find 16.172.in-addr.arpa.
  Record name: @
  NS record: ipaserver.cldrsetup.local.
  Record name: 226.31
  PTR record: ipaserver.cldrsetup.local.
  Record name: 227.31
  PTR record: pvcbase-master.cldrsetup.local.
  Record name: 228.31
  PTR record: pvcbase-worker1.cldrsetup.local.
  Record name: 231.31
  PTR record: pvcecs-master.cldrsetup.local.
  Record name: 232.31
  PTR record: pvcecs-worker1.cldrsetup.local.
----------------------------
Number of entries returned 12
----------------------------
```

****

Perform the PvC Base Cluster Validation:

https://training-team.gitbook.io/setting-up-cloudera-data-platform-cdp/hive-validation
https://www.quora.com/How-do-you-load-data-into-a-Hive-external-table
https://stackoverflow.com/questions/17425492/hive-insert-query-like-sql
https://github.com/mionisation/BI_BigData_2_HiveDatasetAnalysis/blob/master/createMovieLensTables.hql
https://grouplens.org/datasets/movielens/20m/

Validation:

```
[root@pvcbase-master ~]# dnf install -y wget unzip
[root@pvcbase-master ~]# wget https://files.grouplens.org/datasets/movielens/ml-20m.zip
[root@pvcbase-master ~]# unzip ml-20m.zip
[root@pvcbase-master ~]# cd ml-20m
[root@pvcbase-master ml-20m]# sed -i 1d *
[root@pvcbase-master ml-20m]#
```

```
[root@pvcbase-master ml-20m]# hdfs dfs -ls /
24/03/21 04:32:28 WARN ipc.Client: Exception encountered while connecting to the server :
org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
ls: DestHost:destPort pvcbasemaster.cldrsetup.local:8020 , LocalHost:localPort
pvcbasemaster.cldrsetup.local/172.16.31.227:0. Failed on local exception: java.io.IOException:
org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
[root@pvcbase-master ml-20m]#
```

```
# Locate the HDFS keytab
[root@pvcbase-master ml-20m]# find / -name hive.keytab
/run/cloudera-scm-agent/process/1546340988-hive_on_tez-HIVESERVER2/hive.keytab
/run/cloudera-scm-agent/process/1546340973-hive-HIVEMETASTORE/hive.keytab

# List its contents
[root@pvcbase-master ml-20m]# klist -kt
/run/cloudera-scm-agent/process/1546340973-hive-HIVEMETASTORE/hive.keytab
Keytab name: FILE:/run/cloudera-scm-agent/process/1546340973-hive-HIVEMETASTORE/hive.keytab
KVNO Timestamp         Principal
---- ----------------- --------------------------------------------------------
   1 06/10/25 11:29:34 hive/pvcbase-master.redhat.local@REDHAT.LOCAL

# Obtain a Kerberos ticket for the HDFS principal
[root@pvcbase-master ml-20m]# kinit -kt
/run/cloudera-scm-agent/process/1546340973-hive-HIVEMETASTORE/hive.keytab
hive/pvcbase-master.redhat.local@REDHAT.LOCAL

# Verify your ticket cache
[root@pvcbase-master ml-20m]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: hive/pvcbase-master.redhat.local@REDHAT.LOCAL

Valid starting     Expires            Service principal
07/01/25 06:17:08  07/02/25 05:17:24  krbtgt/REDHAT.LOCAL@REDHAT.LOCAL
        renew until 07/08/25 06:17:08

[root@pvcbase-master ml-20m]#  hdfs dfs -mkdir /tmp/movielens
[root@pvcbase-master ml-20m]#  hdfs dfs -put * /tmp/movielens/
[root@pvcbase-master ml-20m]#  hdfs dfs -chown -R hdfs:supergroup /tmp/movielens
[root@pvcbase-master ml-20m]#  hdfs dfs -ls /tmp/movielens/
[root@pvcbase-master ml-20m]#  hive

CREATE DATABASE movielens;
use movielens;
CREATE TABLE IF NOT EXISTS ratings ( userId int, movieId int, rating double, ts bigint)
COMMENT "Movie Ratings"
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\054'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
LOAD DATA INPATH '/tmp/movielens/movies.csv' overwrite INTO TABLE movies;
LOAD DATA INPATH '/tmp/movielens/tags.csv' overwrite INTO TABLE tags;
LOAD DATA INPATH '/tmp/movielens/ratings.csv' overwrite INTO TABLE ratings;
LOAD DATA INPATH '/tmp/movielens/genome-tags.csv' overwrite INTO TABLE genome_tags;
LOAD DATA INPATH '/tmp/movielens/genome-scores.csv' overwrite INTO TABLE genome_scores;
```

Run the queries from HUE for create db, create table.
Upload data from Hive cli.
Run select query to fetch operations from HUE.

****

OZONE Validation:

```
[root@pvcbase-master ~]# ozone sh bucket list ozone11
24/05/26 07:32:42 WARN ipc.Client: Exception encountered while connecting to the server :
org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
24/05/26 07:32:42 WARN ipc.Client: Exception encountered while connecting to the server :
org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
24/05/26 07:32:42 WARN ipc.Client: Exception encountered while connecting to the server :
org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
24/05/26 07:32:42 ERROR client.OzoneClientFactory: Couldn't create RpcClient protocol exception:
        ... 42 more
org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]

[root@pvcbase-master ~]# klist -e
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: admin@CLDRSETUP.LOCAL

Valid starting       Expires              Service principal
05/15/2024 21:24:12  05/16/2024 20:35:44  krbtgt/CLDRSETUP.LOCAL@CLDRSETUP.LOCAL
        renew until 05/22/2024 21:24:09, Etype (skey, tkt): aes256-cts-hmac-sha1-96,
aes256-cts-hmac-sha384-192

[root@pvcbase-master ~]# find / -name ozone.keytab
/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab
/run/cloudera-scm-agent/process/1546347511-ozone-OZONE_RECON/ozone.keytab
/run/cloudera-scm-agent/process/1546347517-ozone-STORAGE_CONTAINER_MANAGER/ozone.keytab
/run/cloudera-scm-agent/process/1546347273-ozone-STORAGE_CONTAINER_MANAGER/ozone.keytab
/run/cloudera-scm-agent/process/1546347267-ozone-OZONE_RECON/ozone.keytab
/run/cloudera-scm-agent/process/1546347277-ozone-S3_GATEWAY/ozone.keytab
/run/cloudera-scm-agent/process/1546344328-ozone-OZONE_RECON/ozone.keytab
/run/cloudera-scm-agent/process/1546344334-ozone-STORAGE_CONTAINER_MANAGER/ozone.keytab
/run/cloudera-scm-agent/process/1546344338-ozone-S3_GATEWAY/ozone.keytab
/run/cloudera-scm-agent/process/1546344034-ozone-STORAGE_CONTAINER_MANAGER/ozone.keytab
/run/cloudera-scm-agent/process/1546344028-ozone-OZONE_RECON/ozone.keytab
/run/cloudera-scm-agent/process/1546344038-ozone-S3_GATEWAY/ozone.keytab

[root@pvcbase-master ~]# klist -kt
/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab
Keytab name: FILE:/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 05/19/2024 22:11:49 HTTP/pvcbase-master.cldrsetup.local@CLDRSETUP.LOCAL
   2 05/19/2024 22:11:49 s3g/pvcbase-master.cldrsetup.local@CLDRSETUP.LOCAL

[root@pvcbase-master ~]# kinit -kt
/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab
s3g/pvcbase-master.cldrsetup.local@CLDRSETUP.LOCAL

[root@pvcbase-master ~]# klist -kt
/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab

Keytab name: FILE:/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab
KVNO Timestamp           Principal
```

```
---- ------------------- ------------------------------------------------------
   2 05/19/2024 22:11:49 HTTP/pvcbase-master.cldrsetup.local@CLDRSETUP.LOCAL
   2 05/19/2024 22:11:49 s3g/pvcbase-master.cldrsetup.local@CLDRSETUP.LOCAL

[root@pvcbase-master ~]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: s3g/pvcbase-master.cldrsetup.local@CLDRSETUP.LOCAL

Valid starting       Expires              Service principal
05/26/2024 07:36:13  05/27/2024 07:11:26  krbtgt/CLDRSETUP.LOCAL@CLDRSETUP.LOCAL
        renew until 06/02/2024 07:36:13

================================================

[root@pvcbase-master ~]# ozone sh volume list
[ ]

[root@pvcbase-master ~]# ozone sh volume create ozone11
24/05/26 07:46:20 INFO rpc.RpcClient: Creating Volume: ozone11, with s3g as owner and space quota
set to -1 bytes, counts quota set to -1

[root@pvcbase-master ~]# ozone sh volume list
[ {
  "metadata" : { },
  "name" : "ozone11",
  "admin" : "s3g",
  "owner" : "s3g",
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "usedNamespace" : 0,
  "creationTime" : "2024-05-26T14:46:20.912Z",
  "modificationTime" : "2024-05-26T14:46:20.912Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "s3g",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  } ],
  "refCount" : 0
} ]

[root@pvcbase-master ~]# ozone sh bucket create ozone11/testkdbkt1
24/05/26 07:47:19 INFO rpc.RpcClient: Creating Bucket: ozone11/testkdbkt1, with server-side default
bucket layout, s3g as owner, Versioning false, Storage Type set to DISK and Encryption set to
false, Replication Type set to server-side default replication type, Namespace Quota set to -1,
Space Quota set to -1

[root@pvcbase-master ~]# ozone sh bucket list ozone11
[ {
  "metadata" : { },
  "volumeName" : "ozone11",
  "name" : "testkdbkt1",
  "storageType" : "DISK",
  "versioning" : false,
  "usedBytes" : 0,
  "usedNamespace" : 0,
  "creationTime" : "2024-05-26T14:47:19.692Z",
  "modificationTime" : "2024-05-26T14:47:19.692Z",
  "sourcePathExist" : true,
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "bucketLayout" : "FILE_SYSTEM_OPTIMIZED",
  "owner" : "s3g",
  "link" : false
} ]
[root@pvcbase-master ~]#
```

****

Cleanup Cloudera on premises Base Cluster:

https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-uninstallation.html

UnInstall and Cleanup Steps (If Installation fails and not-able to resolve the issues)

Stop all Services

Delete the Cluster

On the Home page, Click the drop-down list next to the cluster you want to delete and select Delete.

Uninstall the Cloudera Manager Server

```
#### Cleanup DB
systemctl status cloudera-scm-server
#cd /etc/yum.repos.d
#rm -rvf cloudera-manager.repo
date
systemctl stop postgresql-17
#dnf remove -y postgresql-contrib postgresql-17-contrib postgresql-server postgresql-17-server
#userdel postgres

systemctl stop cloudera-scm-server cloudera-scm-agent cloudera-scm-server-db
cloudera-manager-server-db
dnf remove -y cloudera-manager-daemons cloudera-manager-agent cloudera-manager-server
cloudera-manager-server-db

systemctl daemon-reload
#rm -rvf /var/lib/pgsql/17/data/
mv -v /var/lib/pgsql/17/data/ /var/lib/pgsql/14/data_bkp_$(date +%Y%m%d)
```

Cleanup CDP-CM, Base Master and Worker nodes

```
#/opt/cloudera/installer/uninstall-cloudera-manager.sh
systemctl stop cloudera-scm-server cloudera-scm-agent cloudera-scm-server-db supervisord;
dnf remove -y cloudera-manager-server cloudera-manager-server-db-2 cloudera-management-agent
cloudera-management-daemon cloudera-manager-*; dnf clean all; systemctl daemon-reload;
for u in cloudera-scm* flume hadoop hdfs hbase hive httpfs hue impala llama mapred oozie solr spark
sqoop sqoop2 yarn zookeeper; do sudo kill $(ps -u $u -o pid=); done
sudo umount cm_processes
```

Cleanup CDP-CM, Base Master and Worker nodes

```
sudo rm -rvf /usr/share/cmf /var/lib/cloudera* /var/cache/yum/cloudera* /var/log/cloudera*
/var/run/cloudera* /etc/cloudera-scm-server /opt/cloudera /etc/cloudera-scm-agent
/var/lib/cloudera-scm-agent/cm_guid* /tmp/.scm_prepare_node.lock
sudo rm -rvf /tmp/kafka-logs
sudo rm -rvf /var/lib/flume-ng /var/lib/hadoop* /var/lib/hue /var/lib/navigator /var/lib/oozie
/var/lib/solr /var/lib/sqoop* /var/lib/zookeeper /hadoop-ozone /impala /hadoop-ozone
/var/local/kafka/data/meta.properties

sudo rm -rvf /hdfs/* /dfs* /hdfs/mapred/* /hdfs/yarn/* /var/lib/had*ozon* /yarn*
/etc/{*atlas*,*hadoop*,ranger,hue,impala,knox,hbase,*hive*,hbase-solr,hadoop-kms,*ozone*,*kafka*,*z
eppelin*,*spark*,sqoop*,schemaregistry,*solr*,hive-hcatalog,hive-webhcat,hue,*hbase*,*kudu*,*knox*,
zookeeper,*tez*,streams*} /tmp/kafka-logs/* /var/local/kafka/data/meta.properties
/var/lib/cloudera-scm-agent/cm_guid

##### Only If you are doing end-to-end cleanup, including cloudera-manager and postgres DB, run on all
for user in hdfs httpfs sqoop kafka yarn hbase streamsrepmgr streamsmsgmgr livy kms atlas
schemaregistry hue zookeeper accumulo phoenix mapred druid ranger zeppelin oozie kudu knox superset
solr hive cruisecontrol impala rangerraz ozone tez dpprofiler flume nifi nifiregistry nifitoolkit
spark flink rangerrms omid hadoop kraft; do userdel -r "$user" 2>/dev/null; done

for group in hdfs hue httpfs sqoop kafka yarn hbase streamsrepmgr streamsmsgmgr livy kms atlas
schemaregistry hue zookeeper accumulo phoenix mapred druid ranger zeppelin oozie kudu knox superset
```

```
solr hive cruisecontrol impala rangerraz ozone tez dpprofiler flume nifi nifiregistry nifitoolkit
spark flink rangerrms omid hadoop kraft; do groupdel "$group" 2>/dev/null; done

java -version
python3 -V
```

****

Cleanup Cloudera on premises Data Services-ECS Cluster:

https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation-ecs/topics/cdppvc-installation-ecs-uninstall-pvc.html

CLEANUP ECS

```
# On each host in the cluster:
/opt/cloudera/parcels/ECS/docker/docker container stop registry
/opt/cloudera/parcels/ECS/docker/docker container rm -v registry
/opt/cloudera/parcels/ECS/docker/docker image rm registry:2

# Stop the ECS cluster in Cloudera Manager

# On each host:
cd /opt/cloudera/parcels/ECS/bin
./rke2-killall.sh # usually 2 times is sufficient

# Use umount to unmount all NFS disks.
# umount /docker /lhdata /cdwdata  #not needed in our case
./rke2-uninstall.sh
systemctl daemon-reload
rm -rvf /ecs/* # assumes the default defaultDataPath and lsoDataPath
rm -rvf /var/lib/docker_server/* # deletes the auth and certs
rm -rvf /etc/docker/certs.d/ /etc/docker/* # delete the ca.crt
rm -rvf /var/lib/docker/*
rm -rvf /etc/rancher /var/lib/rancher /var/log/rancher /var/lib/rancher/k3s/server/node-token
rm -rvf /run/k3s /opt/containerd /opt/cni
rm -rvf /docker/* /lhdata/* /cdwdata/*
rm -rvf ~/.kube ~/.cache
systemctl daemon-reload

# Delete the ECS cluster in Cloudera Manager. To delete, follow the steps as below:
# In Cloudera Manager, navigate to Cloudera on premises Data Services and click . Click Delete.
# The Delete Cluster wizard appears. Click Delete.

#Clean IPtables on each host:
echo "Reset iptables to ACCEPT all, then flush and delete all other chains";
declare -A chains=( [filter]=INPUT:FORWARD:OUTPUT [raw]=PREROUTING:OUTPUT
[mangle]=PREROUTING:INPUT:FORWARD:OUTPUT:POSTROUTING [security]=INPUT:FORWARD:OUTPUT
[nat]=PREROUTING:INPUT:OUTPUT:POSTROUTING );
for table in "${!chains[@]}";
      do
      echo "${chains[$table]}" | tr : $"\n" | while IFS= read -r;
      do
      sudo iptables -t "$table" -P "$REPLY" ACCEPT
      done
      sudo iptables -t "$table" -F
      sudo iptables -t "$table" -X
done
iptables -F; iptables -L
rm -rvf /etc/rancher /var/lib/rancher /var/log/rancher

# remove agents from all hosts in CM UI (Optional)
systemctl stop cloudera-scm-agent
dnf remove -y cloudera-manager-agent cloudera-manager-daemons
rm -rvf /opt/cloudera/cm-agent/
rm -rf /opt/cloudera/ /var/lib/cloudera-scm-agent

# Remove the hosts from CM-UI
```

Alternatively, an experimental script is available. This script combines steps three through five. The script is available here:
https://github.com/cloudera-labs/snippets/blob/main/private-cloud/kill-2-rke.sh

 # Reboot the host(s).
 # Before you install ECS again, ensure that the IP tables list is empty by executing the following command:

```
iptables -L
```

****

```
alternatives --set python /usr/bin/python2

openssl s_client -connect cldr-mngr.cldrsetup.local:8443 < /dev/null | sed -ne '/-BEGIN
CERTIFICATE-/,/-END CERTIFICATE-/p' > knoxssoAmbari.crt
```

If you are using PostgreSQL, turn off Readline support by using the -n option. (history-passwd)
Start the Cloudera Management Service when the Reports Manager role is ready. See Starting the Cloudera Management Service.

Ensure that the Ranger Solr and Ranger HDFS plugins are enabled.

Important
Ensure you complete the following tasks before you start performing the steps to configure TLS 1.2 on the Reports Manager for communicating with the database:

On the Cloudera Manager UI, navigate to Clusters > Cloudera Management Service.
Select the Configuration tab and search for reportsmanager_db_safety_valve.
Based on your database type you must override headlamp.db.properties file with JDBC URL properties. Enter the appropriate values in the following format to override the connection to use TLS 1.2.

```
PostgreSQL
com.cloudera.headlamp.orm.hibernate.connection.url=jdbc:postgresql://<DB-HOST>:<DB-PORT>/<DB_NAME>?
useSSL=true&trustCertificateKeyStoreUrl=<PATH_TO_TRUSTSTORE_FILE>&trustCertificateKeyStoreType=<TRU
STSTORE_TYPE>&trustCertificateKeyStorePassword=<TRUSTSTORE_PASSWORD>
com.cloudera.headlamp.db.type=postgresql
com.cloudera.headlamp.db.host=<DB-HOST>:<DB-PORT>
com.cloudera.headlamp.db.name=<DB_NAME>
```

```
[15/Mar/2024 04:59:00 -0700] 10184 MainThread agent  ERROR Heartbeating to localhost:7182 failed.
Traceback (most recent call last):
  File "/opt/cloudera/cm-agent/lib/python3.8/site-packages/cmf/agent.py", line 1588, in
_send_heartbeat
    transceiver = cmf.https.HTTPSTransceiver(
  File "/opt/cloudera/cm-agent/lib/python3.8/site-packages/cmf/https.py", line 245, in __init__
    self.conn.connect()
  File "/opt/cloudera/cm-agent/lib/python3.8/site-packages/M2Crypto/httpslib.py", line 74, in
connect
    sock.connect((self.host, self.port))
  File "/opt/cloudera/cm-agent/lib/python3.8/site-packages/M2Crypto/SSL/Connection.py", line 337,
in connect
    if not check(self.get_peer_cert(),
  File "/opt/cloudera/cm-agent/lib/python3.8/site-packages/M2Crypto/SSL/Checker.py", line 122, in
__call__
    raise WrongHost(expectedHost=self.host,
M2Crypto.SSL.Checker.WrongHost: Peer certificate subjectAltName does not match host, expected
localhost, got DNS:pvcbasemaster.cldrsetup.local
```

```
pvcecs[1-5].cldrsetup.local; pvcecsmaster.cldrsetup.local: IOException thrown while collecting data
from host: Received fatal alert: internal_error
```

Solution:

```
openssl s_client -connect pvcbasemaster.cldrsetup.local:7183

[root@pvcbasemaster cloudera-scm-agent]# cat /etc/cloudera-scm-agent/config.ini|grep server
change hostname to dnsname in place of localhost and restart all agents (heartbeat issue resolved)

/opt/cloudera/cm-agent/bin/supervisorctl -c /var/run/cloudera-scm-agent/supervisor/supervisord.conf
restart status_server

grep -v -e '^[[:space:]]*$' -e '^#' /etc/cloudera-scm-agent/config.ini
```

```
grep -v -e '^[[:space:]]*$' -e '^#' /etc/cloudera-scm-agent/config.ini2024-03-16 03:04:54,710 ERROR
pool-7-thread-1:com.cloudera.server.cmf.components.CmServerStateSynchronizer: Failed during cleanup
: null
```

Solution:
Set java_home by searching java in configuration on the CM console.
Install Postgres and CDP base same day all together otherwise may cause ssl issue (observation)

Stale service status require restart of cluster

Ozone client config issue while deploy krb - known issue
It appears that you might have a proxy setup for the Administration Console. Specify the proxy url as the Frontend url or disable the HTTP Referer Check option.

Ranger, Atlas not running
Due to kafka issue and SOLR issue

SOLR error:
Initialize SOLR and create HDFS home dir from actions and start service will fix issue

---

Kafka error:
Kafka and SOLR depends on Ozone, (SOLR depends on Kafka as well) install this first

---

Tez error
tez -> action -> upload tez file to hdfs
CM > Hive > Action > Create hive dir

YARN queue manager error:

```
[root@pvcbase-master ~]# sudo mkdir /var/lib/hadoop-yarn/
[root@pvcbase-master ~]# sudo chmod +077 /var/lib/hadoop-yarn/
[root@pvcbase-master ~]# sudo chown yarn:hadoop /var/lib/hadoop-yarn/
```

Kafka error:

Solution: delete /var/local/kafka/data/meta.properties

Enable thrift server for hbase-hue

```
set wal property codec-hbase
```

HBASE master bad health:

The problem lies in Cloudera Management Monitor Service, not in Hbase itself. What I did is to restart Cloudera Management Monitor Service, and then restart HBase. After that everything seems to be fine.

Ozone error - Could not find or is not a file

Make sure that hdfs_service is enabled in the Ozone configuration. By having this enabled, the CM agent will put the core-site.xml into the process directory and that error will be gone.
Cleanup ozone directories before redeployment.

HDDS error Ozone

```
[root@pvcbase-master ~]# systemctl restart cloudera-scm-supervisord
```

Ozone ERROR datanode fail to start:

Solution: Perform Proper Cleanup on namenode and datanodes.

```
[root@pvcbase-master ~]# rm -rvf /hdfs/*had*oz* /var/lib/had*oz* /etc/had*oz*
```

```
[root@pvcbase-master ~]# sudo -u postgres psql -U postgres -p 5432 -h $(hostname)
Password for user postgres:
psql (14.11)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression:
off)
Type "help" for help.

postgres=# \q

[root@pvcbasemaster data]# echo -n 'Sahu@123{admin}' | md5sum
c94251c29cd07ed2daf0b6edcf843362  -
```

```
[root@pvcbasemaster data]# sudo -u postgres psql -U postgres -p 5432 -h $(hostname)
Password for user postgres:
psql (14.11)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression:
off)
Type "help" for help.

postgres=# \c ranger
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression:
off)
You are now connected to database "ranger" as user "postgres".
ranger=# update x_portal_user set password = '9746e519adb14ec3ffbf4aff051f104d' where login_id =
'admin';
UPDATE 1
ranger=#
ranger=# select * from x_portal_user where login_id = 'admin';
 id |        create_time         |       update_time       | added_by_id | upd_by_id | first_name |
last_name | pub_scr_name | login_id |
    password             | email | status | user_src | notes | other_attributes | sync_source |
old_passwords | password_updated_time
----+----------------------------+-------------------------+-------------+-----------+------------+
-----------+--------------+----------+----------------------------------+-------+--------+---------
-+-------+------------------+-------------+---------------+----------------------
  1 | 2024-03-17 12:15:27.272988 | 2024-03-17 19:15:43.854 |             |           | Admin      |
| Admin        | admin    | c94251c29cd07ed2daf0b6edcf843362 |       |      1 |        0 |       |
|              |               |
(1 row)
ranger=# \q
-------------------------------------------------------------------------------------------------------
----

[root@pvcbasemaster data]# sudo -u postgres psql -U rangeradmin -p 5432 -d ranger -h $(hostname)
Password for user rangeradmin:
psql (14.11)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression:
off)
Type "help" for help.
ranger=> \q
```

```
[root@pvcbase-master ~]# tail -f
/var/log/ranger/admin/access_log-pvcbasemaster.cldrsetup.local-2024-03-21.log
```

```
2024-03-19 23:59:28.861 PDT [7434] LOG:  could not accept SSL connection: EOF detected
2024-03-19 23:59:28.861 PDT [7427] LOG:  could not accept SSL connection: EOF detected
2024-03-19 23:59:28.861 PDT [7539] LOG:  could not accept SSL connection: EOF detected
```

Caused by: java.io.FileNotFoundException: /var/lib/cloudera-scm-server/.postgresql/root.crt (Permission denied)

Solution:

Ranger UI error SSL issue : issue was with permission on cloudera-scm-server directory where root.crt was stored.

Postgres Connection Limit exceeded:

```
Operation error. response=VXResponse={org.apache.ranger.view.VXResponse@2ca9483cstatusCode={1}
msgDesc={RangerKRBAuthenticationFilter Failed : Exception [EclipseLink-4002] (Eclipse Persistence
Services - 2.7.7.v20200504-69f2c2b80d): org.eclipse.persistence.exceptions.DatabaseException
Internal Exception: java.sql.SQLException: Connections could not be acquired from the underlying
database!
Error Code: 0} messageList={null} }
javax.ws.rs.WebApplicationException
      at org.apache.ranger.common.RESTErrorUtil.createRESTException(RESTErrorUtil.java:56)

--------------------------------------------------------------------------------------------------

Request failed. loginId=null, logMessage=RangerKRBAuthenticationFilter Failed : Exception
[EclipseLink-4002] (Eclipse Persistence Services - 2.7.7.v20200504-69f2c2b80d):
org.eclipse.persistence.exceptions.DatabaseException
Internal Exception: java.sql.SQLException: Connections could not be acquired from the underlying
database!
Error Code: 0
javax.ws.rs.WebApplicationException
      at org.apache.ranger.common.RESTErrorUtil.createRESTException(RESTErrorUtil.java:56)

--------------------------------------------------------------------------------------------------

2024-03-19 00:55:23,767 WARN
C3P0PooledConnectionPoolManager[identityToken->1bqot7nb2ns2s4q1pyen82|14b0e127]-HelperThread-#0:com
.mchange.v2.resourcepool.BasicResourcePool:
com.mchange.v2.resourcepool.BasicResourcePool$ScatteredAcquireTask@1ab83b08 -- Acquisition Attempt
Failed!!! Clearing pending acquires. While trying to acquire a needed new resource, we failed to
succeed more than the maximum number of allowed acquisition attempts (5). Last acquisition attempt
exception:
org.postgresql.util.PSQLException: FATAL: sorry, too many clients already
        at
org.postgresql.core.v3.ConnectionFactoryImpl.doAuthentication(ConnectionFactoryImpl.java:698)

--------------------------------------------------------------------------------------------------
```

standard_conforming_strings=off

```
2024-03-19 23:16:26.279 PDT [26314] WARNING:  nonstandard use of escape in a string literal at
character 261
2024-03-19 23:16:26.279 PDT [26314] HINT:  Use the escape string syntax for escapes, e.g., E'\r\n'.
2024-03-19 23:16:33.719 PDT [24593] FATAL:  sorry, too many clients already
```

Solution:

Increase max_connections to 1000 on postgresql.conf file

```
2024-03-26 03:35:19,203 ERROR - [main:] ~ GraphBackedSearchIndexer.initialize() failed
(GraphBackedSearchIndexer:386)
org.apache.solr.client.solrj.impl.HttpSolrClient$RemoteSolrException: Error from server at
https://pvcbasemaster.cldrsetup.local:8995/solr: Can not find the specified config set:
vertex_index
```

https://community.cloudera.com/t5/Support-Questions/atlas-webui-is-not-accessible/td-p/324743

stop atlas> initialize atlas> start atlas

```
[root@pvcbase-master ~]# klist -kt
/run/cloudera-scm-agent/process/1546342867-SolrServerGracefulShutDown/solr.keytab
[root@pvcbase-master ~]# kinit -kt
/run/cloudera-scm-agent/process/1546342867-SolrServerGracefulShutDown/solr.keytab
solr/pvcbasemaster.cldrsetup.local@CLDRSETUP.LOCAL
```

```
[root@pvcbase-master ~]# /opt/cloudera/parcels/CDH/bin/zookeeper-client
```

Chart not showing--> install mgmt service.

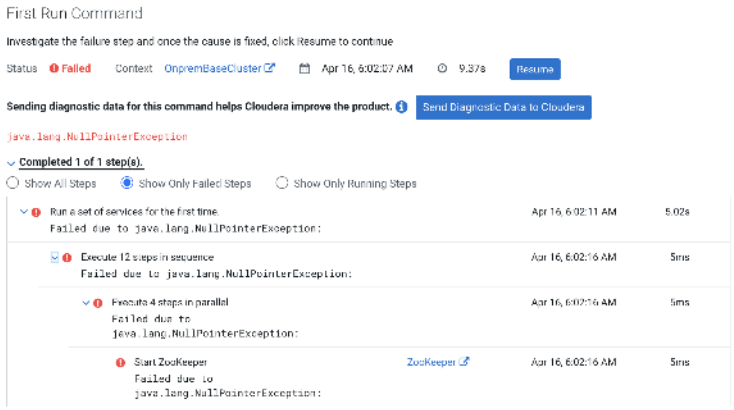
```
logfile=/var/log/cloudera-scm-agent/supervisord.log
```

****

NullPointerException while starting zookeeper

```
Failed due to com.cloudera.cmf.command.CmdExecException: java.lang.NullPointerException
```

Solution: Zookeeper instances should be 3.



Solr Error:

```
java.nio.file.NoSuchFileException:
/opt/cloudera/parcels/CDH-7.3.1-1.cdh7.3.1.p1032.62597146/lib/solr/server/solr-webapp/webapp/WEB-INF/lib/ozone-filesystem-hadoop3-1.4.0.7.3.1.1032-3.jar
```

Solution: Find the correct gbn specific to cdh version installed.
URL for jar file:
https://cloudera-build-us-west-1.vpc.cloudera.com/s3/build/52717809/cdh/7.x/maven-repository/org/[…]6/ozone-filesystem-hadoop3-1.4.0.7.3.1.1-246.jar

---

Yarn Error:

```
Failed to execute command Install YARN MapReduce Framework JARs on service YARN
```

Solution: Find the correct gbn specific to cdh version installed.

Ranger admin UI not opening:

Solution: The issue is with the time synchronization, run "chronyc -a makestep" to sync the time.

Atlas & Knox UI not able to login:

Solution: need to update permission in order to access atlas and knox ui with default pam authentication enabled, use command.

```
chmod 444 /etc/shadow
```

Imp Links:
https://community.cloudera.com/t5/Support-Questions/atlas-webui-is-not-accessible/td-p/324743
https://community.cloudera.com/t5/Support-Questions/how-to-change-default-Atlas-UI-admin-password/td-p/177312
https://community.cloudera.com/t5/Support-Questions/Knox-authentication-with-PAM/m-p/339556

Cloudera on premises Data Services ECS Cluster Error Handling:

https://spacelift.io/blog/kubectl-port-forward
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/day-two-operations/cdppvc-data-services-day-two-operations.pdf
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.1/managing-ecs/cm-manage-ecs.pdf
https://docs.cloudera.com/documentation/other/reference-architecture/PDF/cloudera_ref_arch_cdp_dc.pdf
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/upgrade-ecs/cdppvc-upgrade-ecs.pdf

Could not get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:8080: connect: connection refused

Solution: kubeconfig error

```
[root@pvcecs-master ~]# journalctl -xeu kubelet
[root@pvcecs-master ~]# mkdir -p ~/.kube && cp /etc/rancher/rke2/rke2.yaml ~/.kube/config
[root@pvcecs-master ~]# mkdir -p ~/.kube && cp /etc/rancher/rke2/rke2.yaml ~/.kube/config
[root@pvcecs-master ~]# kubectl get pods
-bash: kubectl: command not found
[root@pvcecs-master ~]# export
PATH=/var/lib/rancher/rke2/data/v1.26.10-rke2r1-e58e49f33617/bin/:$PATH
[root@pvcecs-master ~]# kubectl get pods
No resources found in the default namespace.
[root@pvcecs-master ~]# kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.43.0.1    <none>        443/TCP   13h
[root@pvcecs-master ~]#
```

Error in commands

```
kubectl -n kubernetes-dashboard get sa/admin-user -o 'jsonpath={.secrets[0].name}'
```

Solution:

```
kubectl -n kubernetes-dashboard get secret/admin-user-secret  -o 'jsonpath={.metadata.name}'
(working)
```

```
+ /var/lib/rancher/rke2/bin/kubectl --kubeconfig /etc/rancher/rke2/rke2.yaml -n
kubernetes-dashboard get secret -o 'go-template={{.data.token | base64decode}}'
error: error executing template "{{.data.token | base64decode}}": template: output:1:16: executing
"output" at <base64decode>: invalid value; expected string
```

Solution:

```
kubectl get secret -n kubernetes-dashboard admin-user-secret -o 'go-template={{.data.token |
base64decode}}' (working command)
```

Error: INSTALLATION FAILED: cannot re-use a name that is still in use

```
++ K8S_DASHBOARD_CHART_FILE_NAME=kubernetes-dashboard-5.10.0.tgz
++
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/../installer/ins
tall/bin/linux/helm install kubernetes-dashboard
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-5.10.0.tgz -n kubernetes-dashboard --create-namespace -f
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-overrides.yaml
```

Solution:

```
kubectl delete ns kubernetes-dashboard
```

Install python3.8 for ecs
Always mount correct volumes in config, created previously

Bootstrap error kubectl- remove /var/lib/rancher directory

```
/var/lib/rancher/rke2/agent/logs/kubelet.log
```

```
[root@pvcecsmaster ~]# kubectl get pods
Unable to connect to the server: tls: failed to verify certificate: x509: certificate signed by
unknown authority
```

Solution:
After proper cleanup and reinstall issue will solved.

```
[root@pvcecs-master ~]# mkdir -p ~/.kube
[root@pvcecsmaster opt]# cp /etc/rancher/rke2/rke2.yaml ~/.kube/config
cp: overwrite '/root/.kube/config'? Yes

[root@pvcecsmaster opt]# /var/lib/rancher/rke2/data/v1.26.10-rke2r1-e58e49f33617/bin/kubectl get
nodes
NAME                          STATUS   ROLES                       AGE   VERSION
pvcecsmaster.cldrsetup.local   Ready    control-plane,etcd,master   61s   v1.26.10+rke2r1
[root@pvcecsmaster opt]# cp /tmp/ecs_util.sh /opt/cloudera/cm-agent/service/ecs/ecs_util.sh
```

Solution: cleanup was not done properly,after proper cleanup by script, it started running

---

Error: INSTALLATION FAILED: cannot re-use a name that is still in use

```
++
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/../installer/ins
tall/bin/linux/helm install kubernetes-dashboard
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-5.10.0.tgz -n kubernetes-dashboard --create-namespace -f
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-overrides.yaml
```

Solution:

```
[root@pvcecs-master ~]# k get pods -A|grep dashboard
kubernetes-dashboard   kubernetes-dashboard-68876b7cb8-fl58c                  1/1     Running
0             4m13s
[root@pvcecsmaster ~]# k delete pod kubernetes-dashboard
Error from server (NotFound): pods "kubernetes-dashboard" not found
[root@pvcecsmaster ~]# k delete pod kubernetes-dashboard-68876b7cb8-fl58c -n kubernetes-dashboard
pod "kubernetes-dashboard-68876b7cb8-fl58c" deleted
[root@pvcecsmaster ~]#
[root@pvcecsmaster ~]# k delete ns kubernetes-dashboard
```

```
[root@pvcecs-master ~]# k logs cli-v2m7w -n cdp
```

```
HTTPSConnectionPool(host='console-cdp.apps.cldrsetup.local', port=443): Max retries exceeded with
url: /api/v1/environments2/createPrivateEnvironment (Caused by
NameResolutionError("<cdpcli.cdprequest.CdpHTTPSConnection object at 0x7fbb901047f0>: Failed to
resolve 'console-cdp.apps.cldrsetup.local' ([Errno -2] Name or service not known)"))
```

Solution: Make sure to setup wildcard DNS in advance before proceeding with ECS Setup steps

---

```
k edit cm rke2-coredns-rke2-coredns  -n kube-system -o yaml
apiVersion: v1
data:
  Corefile: |
    .:53 {
      errors
      health {
          lameduck 5s
      }
      ready
      kubernetes cluster.local cluster.local in-addr.arpa ip6.arpa {
          pods insecure
          fallthrough in-addr.arpa ip6.arpa
          ttl 30
      }
      prometheus 0.0.0.0:9153
      #forward . /etc/resolv.conf
      forward . 172.16.31.226
      cache 30
      loop
      reload
      loadbalance
    }
```

Possible Cause and Solution:
Wildcard dns was not set up properly. Before proceeding with ECS installation, set up the wildcard DNS.

Map docker dependency if localhost:6443 apiserver error come in ECS

Bootstrap error

```
[time="2024-03-19T11:10:29-07:00" level=fatal msg="Failed to reconcile with temporary etcd:
bootstrap data already found and encrypted with different token"
```

Solution:
Cleanup not done properly.

****

Error in Initializing embedded Vault phase while installing data services cluster

```
failed to create secret namespaces "istio-ingress" not found
Error from server (NotFound): pods "vault-0" not found

Warning: deleting cluster-scoped resources, not scoped to the provided namespace
++ kubectl delete secret ingress-default-cert -n kube-system
++ kubectl delete secret ingress-default-cert -n istio-ingress
Error from server (NotFound): secrets "ingress-default-cert" not found
++ exit_code=1
++ kubectl delete namespace vault-system
++ kubectl create namespace vault-system
```

```
++ kubectl apply -f /opt/cloudera/parcels/ECS-1.5.5-b82-ecs-1.5.5-b82.p0.65068258/vault/vault.yaml
Warning: Detected changes to resource vault which is currently being deleted.
++ TMPDIR=logs
++ openssl genrsa -out logs/vault.key 2048
++ cat /opt/cloudera/parcels/ECS-1.5.5-b82-ecs-1.5.5-b82.p0.65068258/vault/csr.conf
++ envsubst
++ openssl req -new -key logs/vault.key -subj
'/CN=system:node:vault.vault-system.svc;/O=system:nodes' -out logs/server.csr -config logs/csr.conf
++ cat
+++ cat logs/server.csr
+++ base64
+++ tr -d '\n'
++ kubectl create -f logs/csr.yaml
++ kubectl certificate approve vault-csr
++ '[' -z ']'
++ sleep 1
+++ kubectl get csr vault-csr -o 'jsonpath={.status.certificate}'
```

Solution: To resolve the issue, manually create the istio-ingress namespace with the below command, once it becomes active, delete the vault-0 pod in the terminating state, then simply resume the installation process again from the UI, and it should proceed correctly from that point.

```
create ns istio-ingress
```

Issue while activating the CDW environment:

```
failed to create secret namespaces "istio-ingress" not found
Error from server (NotFound): pods "vault-0" not found

unable to create cluster initial state: unable to create the CDW resource pool: non-retriable
error: failed to update the resource pool 'root.cdp-env-1.cdw', because: 'Request type:
'UpdateResourcePool', state: 'FAILED', last updated: '2025-04-21T07:35:46.134Z', finished: 'true',
error: 'policy violation: child quota larger than parent quota'', request ID:
'rp-req-e235fed3-757b-4cb9-91ad-f4570d1e2a25'
```

Solution: There was some issue with the cdw environment due to which it was not getting activated, we tried a workaround to create another environment to activate cdw service and it worked.

Commands:

```
 kubectl get po -A|grep access
 kubectl delete pod -n cdp cdp-release-cluster-access-manager-67c7b5d8cc-5gcrh
 kubectl get po -A|grep pool
 kubectl delete pod -n cdp cdp-release-resource-pool-manager-7dd8c84569-42d6c
```

Error: Resource Pool Manager Fails to Start After Upgrade

```
{"level":"ERROR","timestamp":"2025-04-29T10:09:01.448Z","caller":"cam/driver.go:265","message":"Err
or updating queueConfig","grpc error":"rpc error: code = InvalidArgument desc = queue config update
failed"}
...
"pool manager startup failure","error":"cam operation failed, returned state 'UNSET', info: '',
grpc error: rpc error: code = InvalidArgument desc = queue config update failed"
```

Solution: This issue was observed in the monitoring namespace under CDW.

This can be resolved by restarting the Cluster Access Manager followed by the Resource Pool Manager services in sequence.

Error: Vault Pod Not Running – Upgrade Process Stuck

```
Vault pod not in running state.
Upgrade checks hanging on "Vault unseal".
```

Solution: The Vault pod was sealed due to node issues during the earlier stages of the upgrade. To resolve this, manually unseal it using the command below, then verify the pod status before continuing with the upgrade.

```
vault operator unseal
```

Error: Upgrade Hook Job Stuck – Pod Deleted But Not Recreated

```
kubectl get job cdp-release-pre-upgrade-hook-job1 -n cdp
NAME                                 STATUS    COMPLETIONS   DURATION   AGE
cdp-release-pre-upgrade-hook-job1    Running   0/1           12s        12s
```

Solution: If the pod was deleted, manually reapply the job YAML. Monitor the job's progress or troubleshoot it using below command. If necessary, delete and recreate the job.

```
kubectl logs job/cdp-release-pre-upgrade-hook-job1 -n cdp
```

Commands to delete and recreate the job.

```
kubectl get job -A | grep cdp-release-pre-upgrade-hook
kubectl get job -o yaml -n cdp cdp-release-pre-upgrade-hook-job1 > upgrade.yml
cat upgrade.yml
kubectl get job -A | grep cdp-release-pre-upgrade-hook
kubectl delete job cdp-release-pre-upgrade-hook-job1 -n cdp
kubectl get job -A | grep cdp-release-pre-upgrade-hook
kubectl apply -f upgrade.yml -n cdp
```

CDW Data Catalog Creation Issue

```
Initializing the schema to: 3.1.3000.2025.0.19.1-49
Metastore connection URL:
jdbc:postgresql://postgres-service-default-warehouse:5432/hive?createDatabaseIfNotExist=true&sslmod
e=verify-ca&sslrootcert=/mnt/certs/hms-root.crt
Metastore connection Driver: org.postgresql.Driver
Metastore connection User: hive
Failed to get schema version.
Underlying cause: org.postgresql.util.PSQLException : SSL error: PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path
to requested target

| warehouse-cldrsetup   metastore-0                    •   0/1    Init:0/4           0    0    0
0     n/a       0       0 n/a    pvcecs-worker10.cldrsetup.local    20s      |
```

Solution: The CDW Data Catalog creation failed due to a certificate validation error between the Hive metastore and the PostgreSQL backend. This was caused by an invalid or missing truststore certificate (hms-root.crt). To fix this, extract the correct certificate, validate it, and reapply the secret before restarting the metastore pod.

```
Steps:
# Backup current file if needed
kubectl get secret pg-hms-db-certs -n warehouse-cldrsetup -o yaml > pg-hms-db-certs-backup.yaml

# Recreate the secret with the expected key
kubectl delete secret pg-hms-db-certs -n warehouse-cldrsetup
```

```
kubectl create secret generic pg-hms-db-certs \
  --from-file=TRUSTSTORE_PEM=root.crt \
  -n warehouse-cldrsetup
```

Kubernetes Command Reference:

```
export PATH=$PATH:/var/lib/rancher/rke2/bin
echo "alias
helm='/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/installer/install/bin/linux/hel
m'" >> ~/.bashrc
echo "alias docker='/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/docker/docker'"
>> ~/.bashrc
echo "alias
helm='/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/installer/install/bin/linux/hel
m'" >> ~/.bashrc
echo "alias kubectl="sudo -E /var/lib/rancher/rke2/data/v1.26.10-rke2r1-e58e49f33617/bin/kubectl
--kubeconfig /etc/rancher/rke2/rke2.yaml'" >> ~/.bashrc
alias k=kubectl
. ~/.bashrc

kubectl      get|describe|delete|create
all|pods|nodes|ns|namespaces|svc|service|pv|pvc|rb|rolebinding|sa|roles|csr|secret|hpa|netpol|state
fulset|replicaset|crd              -n vault-system | -A
kubectl get namespace vault-system -o json|yaml > tmp.json
helm list -n vault-system
kubectl api-resources  --namespaced=true -o name | xargs -n 1 kubectl get -n vault-system
k delete ns vault-system --force
k get pods -A -o wide |grep dash
k get sa kubernetes-dashboard -n kubernetes-dashboard -o yaml
kubectl delete pods -n kube-system -l k8s-app=kube-dns
kubectl port-forward deployment.apps/kubernetes-dashboard 8443:https -n kubernetes-dashboard
kubectl port-forward deployment.apps/kubernetes-dashboard 8443:443 -n kubernetes-dashboard
k logs -n cdp cdp-release-dssapp-6b5b68bcfd-b9rdd
k apply -f secret.yaml
k create token default
kubectl api-resources --verbs=list --namespaced -o name | xargs -n 1 kubectl get -n vault-system
--kubeconfig /etc/rancher/rke2/rke2.yaml
helm list -n kubernetes-dashboard
k get|delete|describe|edit  helmcharts.helm.cattle.io    -n vault-system

helm list -n kubernetes-dashboard
helm list -A
helm upgrade --install kubernetes-dashboard
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-5.10.0.tgz -n kubernetes-dashboard  -f
/opt/cloudera/parcels/ECS-1.5.5-b10-ecs-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-overrides.yaml --dry-run | grep admin
find / -name kubernetes-dashboard*.tgz
netstat -lntup|grep 6443
openssl s_client -connect pvcbasemaster.cldrsetup.local:5432 -debug -msg
cp /etc/rancher/rke2/rke2.yaml .kube/config
envsubst
```

****

Acknowledgements

For their support and contribution to the design, validation, and creation of this Validated Design, the author would like to thank:

- Kuldeep Sahu, Partner Solutions Engineer, Cloudera Inc.
- Venkatesh Sellappa, Director, Partner Solutions Engineering, Cloudera Inc.

****