Cloudera on premises / CDP Private Cloud (PvC)

Installation & Setup

:: Cloudera Deployment Guide ::

Published: December 2025

In partnership with:

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 Platform on Red Hat OpenShift Virt VMs.

****

Table of Contents

****

Author History

Name	Version	Date
Kuldeep Sahu	1.0	23-May-2024

****

Cloudera on premises Installation and Configuration on Red Hat 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/1.5.5/installation/topics/cdppvc-installation-airgap.html
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-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/topics/cdppvc-installation-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:

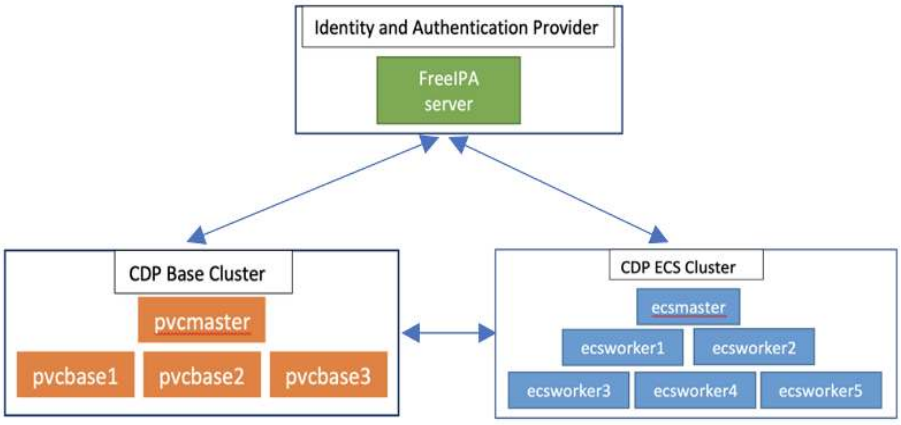
- Cloudera on premises Base Cluster
- Cloudera on premises Data Services Cluster

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 **Red Hat OCP**.

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 OCP) 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

- **Red Hat OCP Data Services Cluster:-** This is the **Red Hat OpenShift Container Platform (OCP)** based Kubernetes cluster that forms the basis for all the containerized workloads of CDP Data Services. It consists of OCP masters and OCP workers.

This document doesn't involve the setup of the OCP cluster. This document assumes that the OCP cluster with required hardware specifications and topology (number of master, worker and infra nodes) is ready in advance. Minimum requirements should be satisfied i.e. network connectivity, forward and reverse DNS resolution, wildcard DNS configuration between the PvC Base and PvC Data Services i.e. OCP cluster nodes.

- **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

Service	Dependencies
Kafka	ZooKeeper
Livy	• Spark • Impala
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 Red Hat Openshift Virt 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 OCP 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 OCP 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.

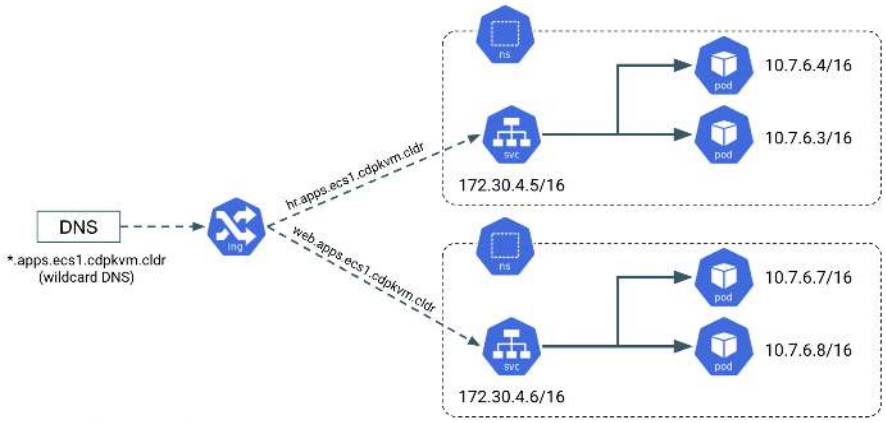
Count	CDP Role
1	FreeIPA Server (Will be used for FreeIPA, Kerberos, Private DNS, LDAP, NTP, KDC, and will be used as an ansible controller node for automation purpose)
1	Cloudera-Manager Server (with external PostgreSQL Database server, will be used for downloading bits as well)
4	CDP Base Cluster (1 Master and 3 Worker Nodes)
6-13	CDP OpenShift Data Services Cluster (3 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.redhat.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.

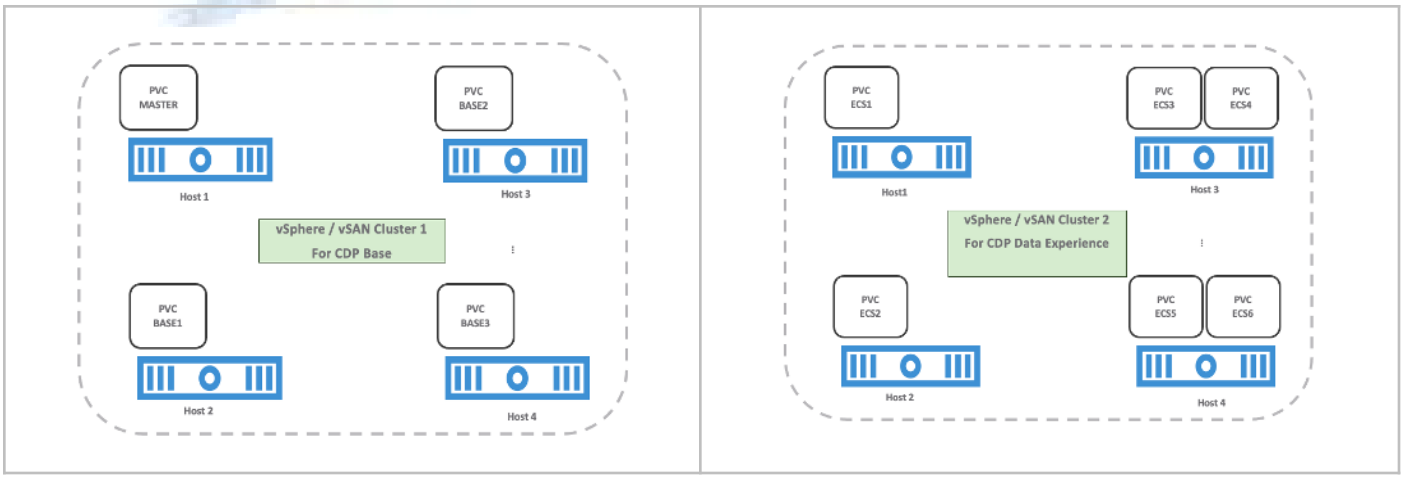
The total number of CA certificates must not exceed 10. Otherwise, pods will be evicted during initialization due to limited memory (1Gi) to process the configmap file.

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/7.1.9/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 /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/7.1.9/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 (OCP) 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	Note
OCP DS CLUSTER	**CDW**				
OCP-Master[1-3]	pvcocp-master[1-3]	16	96GB	root partition + OCP reqs + 500GB	**Accumulated Specs
OCP-Worker	pvcocp-worker1	32	256GB	root partition + OCP reqs + 600GB	
OCP-Worker	pvcocp-worker2	32	256GB	root partition + OCP reqs + 600GB	
OCP-Worker	pvcocp-worker3	32	256GB	root partition + OCP reqs + 600GB	

PvC Data Service (OCP) 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	Note
OCP DS CLUSTER	**CML**				
OCP-Master[1-3]	pvcocp-master[1-3]	16	96GB	root partition + OCP reqs + 500GB **+ 1000GB NFS**	**Accumulated Specs
OCP-Worker	pvcocp-worker1	32	128GB	root partition + OCP reqs + 250GB	
OCP-Worker	pvcocp-worker2	32	128GB	root partition + OCP reqs + 250GB	

PvC Data Service (OCP) 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	note
OCP DS CLUSTER	**CDE**				
OCP-Master[1-3]	pvcocp-master[1-3]	16	96GB	root partition + OCP reqs + 500GB **+ 500GB NFS**	**Accumulated Specs
OCP-Worker	pvcocp-worker1	32	128GB	root partition + OCP reqs + 70GB	
OCP-Worker	pvcocp-worker2	32	128GB	root partition + OCP reqs + 70GB	
OCP-Worker	pvcocp-worker3	32	128GB	root partition + OCP reqs + 70GB	
OCP-Worker	pvcocp-worker4	32	128GB	root partition + OCP reqs + 70GB	

PvC Data Service(OCP) 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	Note
OCP DS CLUSTER	**CML+CDW+CDE**				
OCP-Master[1-3]	pvcocp-master[1-3]	16	96GB	root partition + OCP reqs + 500GB **+ 1500GB NFS**	**Accumulated Specs
OCP-Worker	pvcocp-worker1	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker2	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker3	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker4	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker5	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker6	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker7	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker8	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker9	32	128GB	root partition + OCP reqs + 400GB	
OCP-Worker	pvcocp-worker10	32	128GB	root partition + OCP reqs + 400GB	

****

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)	9.5 **(Verify with SupportMatrix first)**	All Servers
OpenJDK	17.0.14.0.7-1 >=	All Servers
Python3	3.9.22 >=	All Servers
PostgreSQL DB	16 >=	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)	
OCP DB Configuration	Embedded	OCP Only
Vault	Embedded	OCP Only
Docker Registry Type	Embedded/ Cloudera Default	OCP Only
NFS for CAI	Embedded i.e Internal	OCP 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
pvcocp-master	OCP Master Node
pvcocp-worker1 to pvcocp-worker10	OCP Worker Nodes
redhat.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.

****

Prerequisites

Before getting into the actual installation of CDP Private Cloud 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.

Need some Prerequisites from Red Hat team before proceeding with the actual setup:

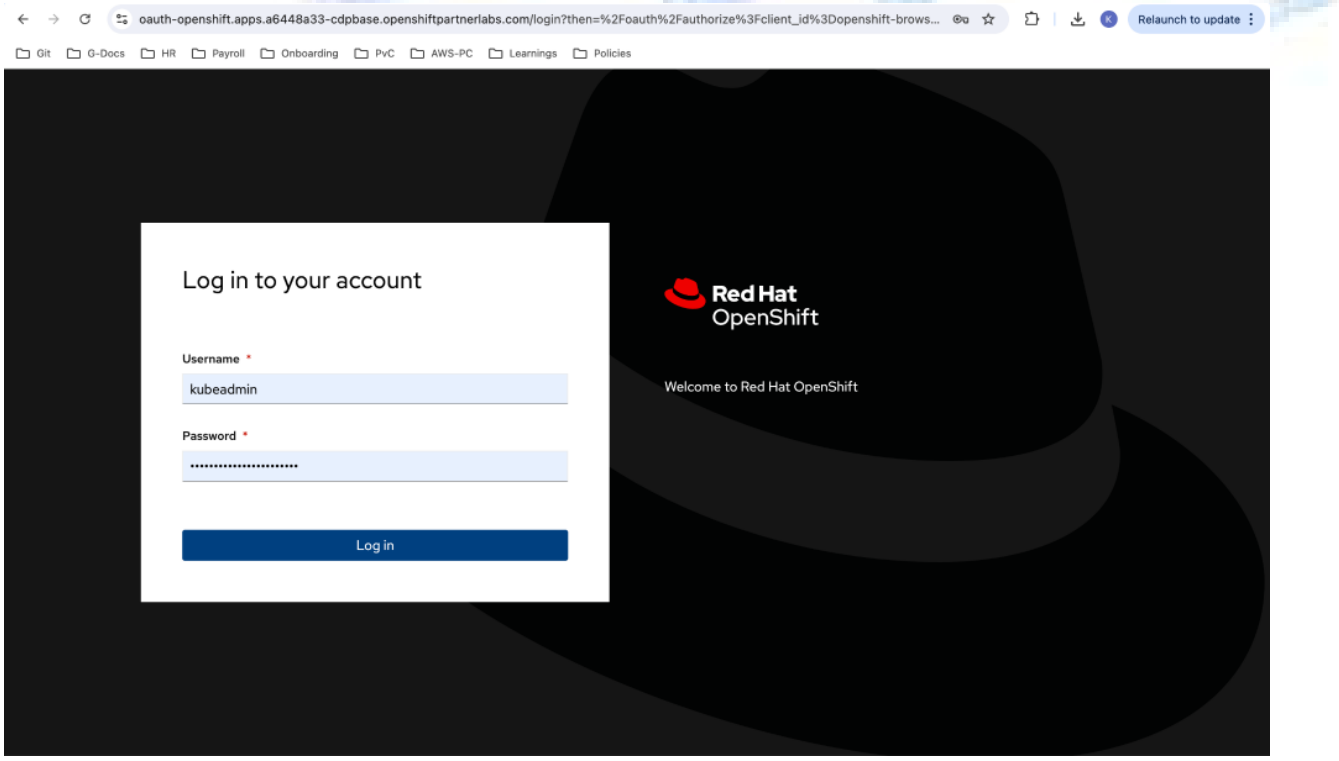
- Access to OCP Partner Lab environment.
- A secondary L2 Network is created and the adapter is attached to the VMs in order to resolve the IP locally.
- VMs created for specified RAM/CPU/Disk configurations for PvC Base Cluster.
- An OCP cluster created with required hardware configurations for PvC Data Services Installation purpose.
- VMs should be able to authenticate directly for root users from external, either via password or KeyPairs(Pub-Priv Keys)--identical key/password for all machines.
- A way to expose the ports for UI access for different VMs e.g. CM-UI, Nexus, Vault, HUE etc.
- We will be installing a Private DNS service and Kerberos/KDC with the help of FreeIPA, HashiCorp Vault, Nexus3 Repository Manager for Private Docker Repository Hosting, and External PostgreSQL DB.
- PvC Base Cluster VMs and OCP cluster nodes should be able to communicate with each other and perform forward and reverse lookup for Domain Names <-> IP Addresses.

Some Notes and helpful commands/instructions for OCP access environment setup on Local Machine:

Login to OpenShift Lab Environment from Command Line:

Open the URL in the web browser and authenticate using your Lab Credentials shared by Red Hat Team:

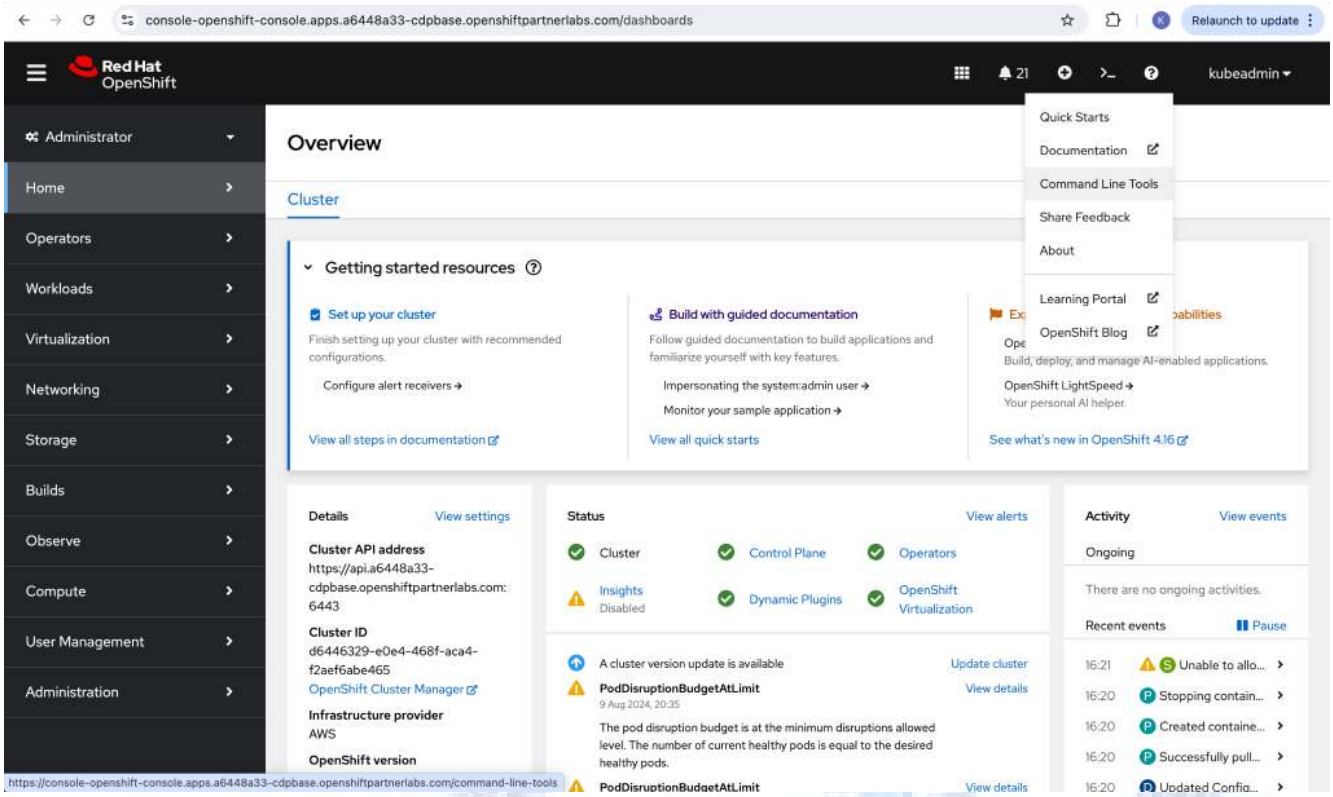
https://oauth-openshift.apps.a6448a33-cdpbase.openshiftpartnerlabs.com/



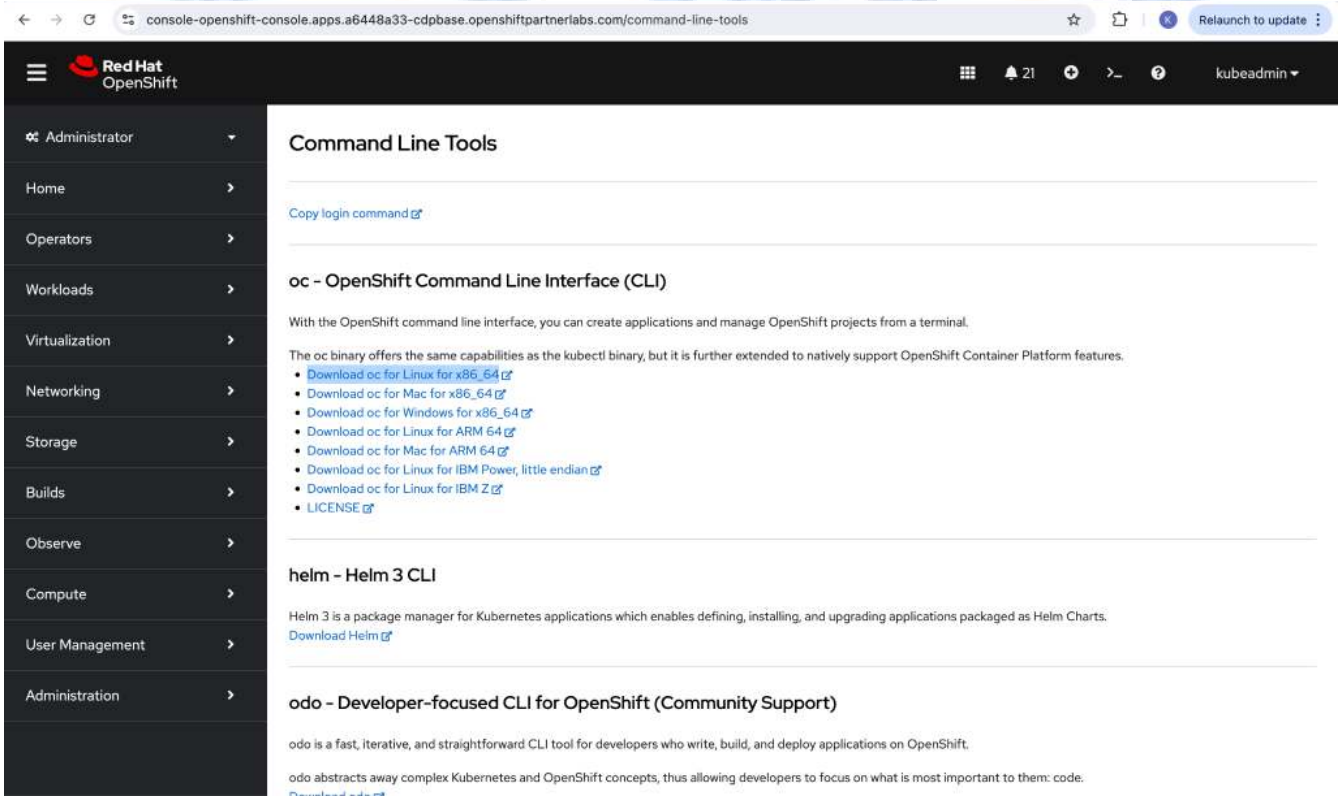
Install OpenShift Client (OC Command Line)

```
ksahu@Kuldeeps-MacBook-Air ~ % mkdir ~/ocp && cd ~/ocp
```

Click Help (?) on Top Bar > Command Line Tools >



Below view will open, copy the Link by Right Click of Mouse, we will use this link with curl command in below steps. On the same page you can copy the Login Command as well.



Linux:

```
[root@ipaserver ocp]# curl -LO
https://mirror.openshift.com/pub/openshift-v4/clients/ocp/latest/openshift-client-linux.tar.gz
[root@ipaserver ocp]# tar -xvf openshift-client-linux.tar.gz
[root@ipaserver ocp]# cd openshift-client-linux
```

Mac:

```
ksahu@Kuldeeps-MacBook-Air ocp % curl -LO
https://mirror.openshift.com/pub/openshift-v4/clients/ocp/latest/openshift-client-mac.tar.gz
ksahu@Kuldeeps-MacBook-Air ocp % tar -xvf openshift-client-mac.tar.gz
ksahu@Kuldeeps-MacBook-Air ocp % cd openshift-client-mac
```

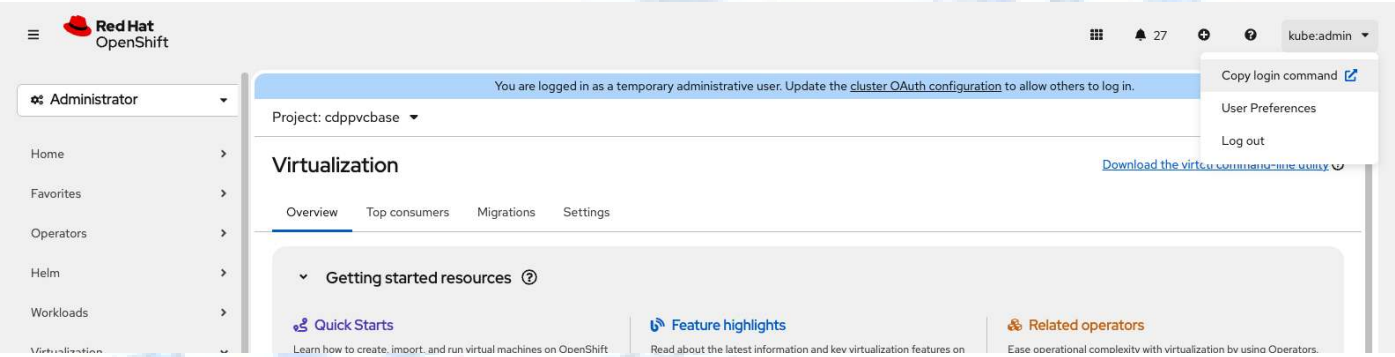
Common steps after Untar:

```
ksahu@Kuldeeps-MacBook-Air openshift-client-mac % chmod +x oc kubectl
ksahu@Kuldeeps-MacBook-Air openshift-client-mac % sudo mv -v oc kubectl /usr/local/bin/
ksahu@Kuldeeps-MacBook-Air openshift-client-mac % oc version
ksahu@Kuldeeps-MacBook-Air openshift-client-mac % kubectl -version
ksahu@Kuldeeps-MacBook-Air openshift-client-mac % cd ~/ocp
ksahu@Kuldeeps-MacBook-Air ocp %
```

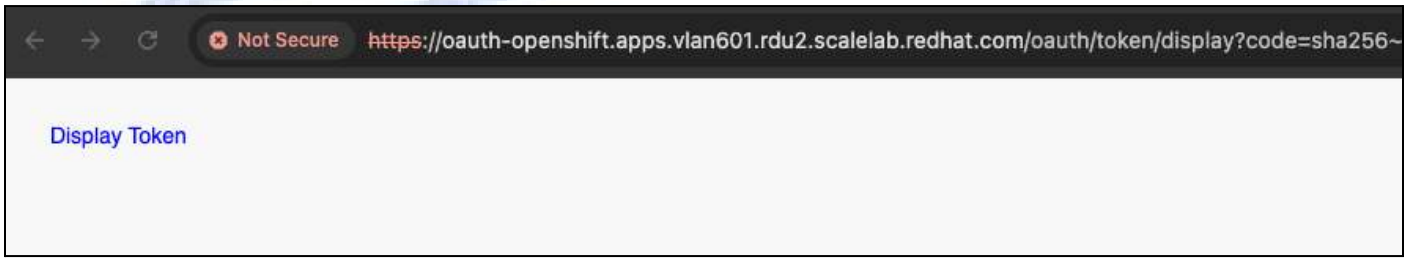
Generate Login Token:

Login to OpenShift Lab Environment from Command Line:

From the OpenShift console, in the upper right side of the screen, click on the username that you're currently logged in as, then select "***Copy login command***".



Then, click on "***Display Token***":



Copy the Login Command from ***Log in with this token*** to run on CLI.

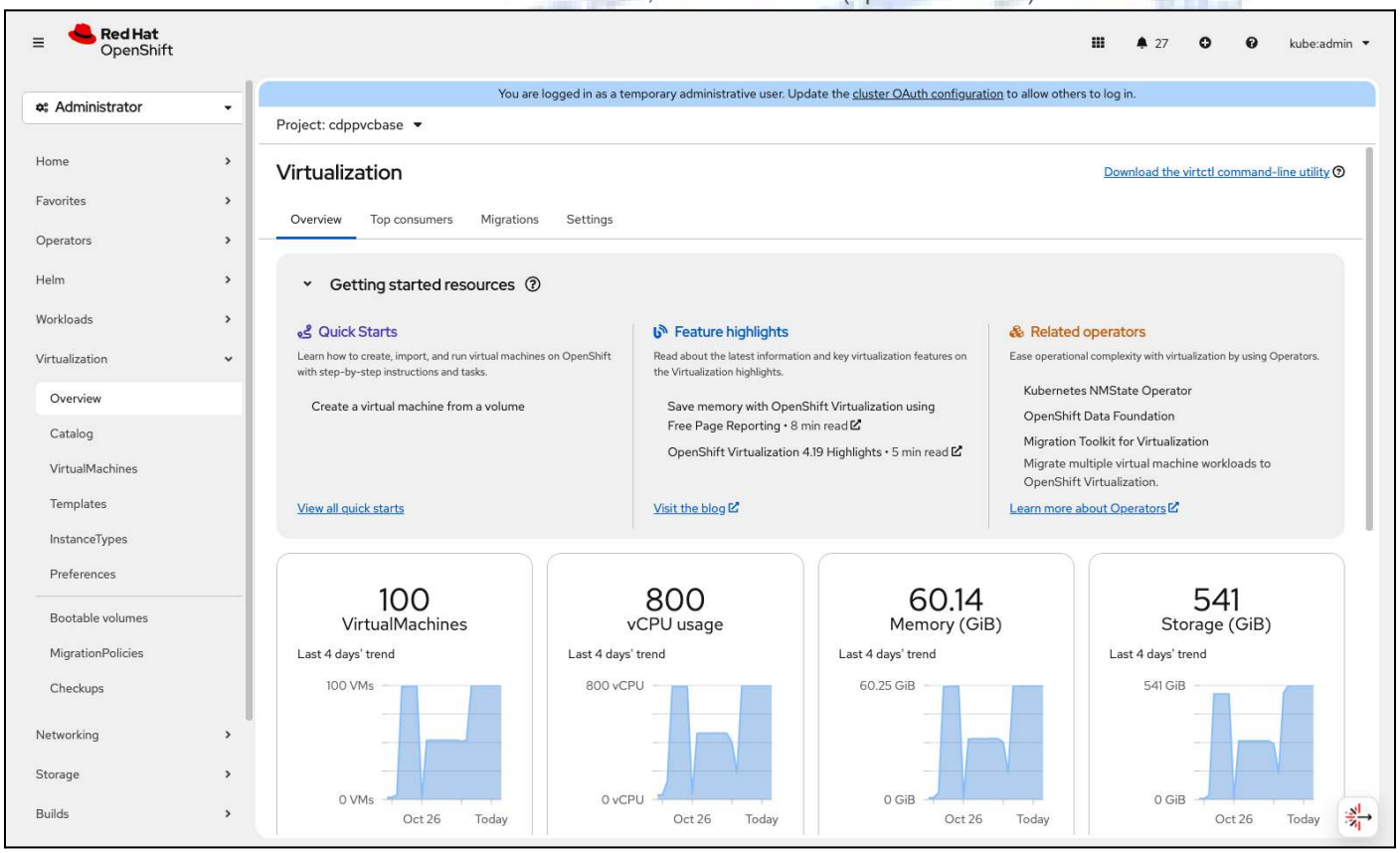
This will provide you both a token and a command (similar to above) on the browser itself, to be run on CLI to authenticate. Just copy the command and run on your computer's CLI:

```
ksahu@Kuldeeps-MacBook-Air ocp % oc login --token=sha256~nBsF-2uzRAnU04t8aC8Gkip2bjcUZ7eBvjvumKMbXUw
--server=https://api.vlan601.rdu2.scalelab.redhat.com:6443
```

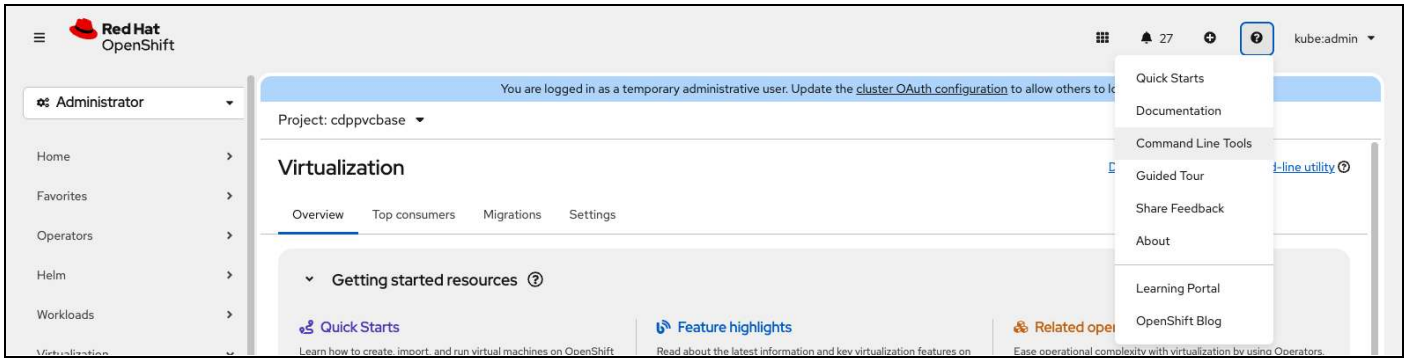
Once Login through CLI, you will be able to run the OCP commands.

Install VirtCTL for interacting with OCP VMs (OCP Virtualization)

From *OCP Console UI*, under *Virtualization> Overview* Section, click on "***Download the virtctl command-line utility***", to download **`virtctl`** and the other command line utilities, such as the **`oc`** (openshift client).



The same download page can be selected by clicking on the "?" icon in the upper right side of the screen, and then selecting "***Command Line Tools***".




```
ksahu@Kuldeeps-MacBook-Air ocp % wget
https://hyperconverged-cluster-cli-download-openshift-cnv.apps.a6448a33-cdpbase.openshiftpartnerlabs.com/
amd64/mac/virtctl.zip
ksahu@Kuldeeps-MacBook-Air ocp % unzip virtctl.zip
ksahu@Kuldeeps-MacBook-Air ocp % cd virtctl
ksahu@Kuldeeps-MacBook-Air virtctl % chmod +x virtctl
ksahu@Kuldeeps-MacBook-Air virtctl % sudo mv -v virtctl /usr/local/bin
ksahu@Kuldeeps-MacBook-Air virtctl % virtctl –version
ksahu@Kuldeeps-MacBook-Air virtctl % cd ~/ocp
ksahu@Kuldeeps-MacBook-Air ocp %
```

Switch the Default OCP Project:

```
ksahu@Kuldeeps-MacBook-Air ocp % oc project cdppvcbase
ksahu@Kuldeeps-MacBook-Air ocp % oc get svc
```

OpenShift commands:

```
ksahu@Kuldeeps-MacBook-Air ocp % oc projects
ksahu@Kuldeeps-MacBook-Air ocp % oc new-project my-new-project
ksahu@Kuldeeps-MacBook-Air ocp % oc new-app my-new-app
ksahu@Kuldeeps-MacBook-Air ocp % oc logs -f bc/my-new-app
ksahu@Kuldeeps-MacBook-Air ocp % oc status
ksahu@Kuldeeps-MacBook-Air ocp % oc rsh dc/postgresql.          (ssh into container)
ksahu@Kuldeeps-MacBook-Air ocp % oc new-app -L (list of toolchain like help command)
ksahu@Kuldeeps-MacBook-Air ocp % oc --help
ksahu@Kuldeeps-MacBook-Air ocp % oc get route
ksahu@Kuldeeps-MacBook-Air ocp % oc expose svc name
ksahu@Kuldeeps-MacBook-Air ocp % oc get pods | deployments | svc
ksahu@Kuldeeps-MacBook-Air ocp % oc create -f ./nginx.yml
```

K8s Commands:

```
ksahu@Kuldeeps-MacBook-Air ocp % kubectl config current-context
ksahu@Kuldeeps-MacBook-Air ocp % kubectl config get-contexts
ksahu@Kuldeeps-MacBook-Air ocp % kubectl config use-context <context-name>
```

Preliminary Work

Ensure the prior steps were completed previously:
Create Project for CDP PVC Base VMs
Create SSH Keys in Project
Auto Register RHEL VMs
Create VMs on OpenShift Virtualization

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. Ensure that the hostname for each machine/host is set so we can refer to them with names instead of IP addresses for simplicity and ease of identification, also follow **Step 2** while logging in to each host. **Replace your ORG DOMAIN**

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 root@10.1.49.1
```

PW: 100yard-

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.redhat.local ipaserver

# Cloudera Manager Server
172.31.1.38 cldr-mngr.redhat.local cldr-mngr
172.31.1.38 postgresdb.redhat.local postgresdb
172.31.1.38 nexus.redhat.local nexus
172.31.1.38 vault.redhat.local vault

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

127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.1.49.1  ipaserver.cdp.rdu2.scalelab.redhat.com  ipaserver

# Cloudera Manager & Utility Server
10.1.49.100 cldr-mngr cldr-mngr.cdp.rdu2.scalelab.redhat.com
10.1.49.100 postgresdb postgresdb.cdp.rdu2.scalelab.redhat.com
10.1.49.100 nexus nexus.cdp.rdu2.scalelab.redhat.com
10.1.49.100 vault vault.cdp.rdu2.scalelab.redhat.com
10.1.49.101 cldr-utility cldr-utility.cdp.rdu2.scalelab.redhat.com

# PvC Base Cluster Nodes
10.1.49.102 pvcbase-master01 pvcbase-master01.cdp.rdu2.scalelab.redhat.com
10.1.49.103 pvcbase-master02 pvcbase-master02.cdp.rdu2.scalelab.redhat.com
10.1.49.104 pvcbase-master03 pvcbase-master03.cdp.rdu2.scalelab.redhat.com
10.1.49.105 pvcbase-worker01 pvcbase-worker01.cdp.rdu2.scalelab.redhat.com
10.1.49.106 pvcbase-worker02
10.1.49.107 pvcbase-worker03
10.1.49.108 pvcbase-worker04
10.1.49.109 pvcbase-worker05
10.1.49.110 pvcbase-worker06
10.1.49.111 pvcbase-worker07
10.1.49.112 pvcbase-worker08
10.1.49.113 pvcbase-worker09
10.1.49.114 pvcbase-worker10
10.1.49.115 pvcbase-worker11
10.1.49.116 pvcbase-worker12
10.1.49.117 pvcbase-worker13
10.1.49.118 pvcbase-worker14
10.1.49.119 pvcbase-worker15
10.1.49.120 pvcbase-worker16
10.1.49.121 pvcbase-worker17
10.1.49.122 pvcbase-worker18
10.1.49.123 pvcbase-worker19
10.1.49.124 pvcbase-worker20
10.1.49.125 pvcbase-worker21
10.1.49.126 pvcbase-worker22
10.1.49.127 pvcbase-worker23
10.1.49.128 pvcbase-worker24
10.1.49.129 pvcbase-worker25
10.1.49.130 pvcbase-worker26
10.1.49.131 pvcbase-worker27
10.1.49.132 pvcbase-worker28
```

```
10.1.49.133 pvcbase-worker29
10.1.49.134 pvcbase-worker30
10.1.49.135 pvcbase-worker31
10.1.49.136 pvcbase-worker32
10.1.49.137 pvcbase-worker33
10.1.49.138 pvcbase-worker34
10.1.49.139 pvcbase-worker35
10.1.49.140 pvcbase-worker36
10.1.49.141 pvcbase-worker37
10.1.49.142 pvcbase-worker38
10.1.49.143 pvcbase-worker39
10.1.49.144 pvcbase-worker40
10.1.49.145 pvcbase-worker41
10.1.49.146 pvcbase-worker42
10.1.49.147 pvcbase-worker43
10.1.49.148 pvcbase-worker44
10.1.49.149 pvcbase-worker45
10.1.49.150 pvcbase-worker46
10.1.49.151 pvcbase-worker47
10.1.49.152 pvcbase-worker48
10.1.49.153 pvcbase-worker49
10.1.49.154 pvcbase-worker50
10.1.49.155 pvcbase-worker51
10.1.49.156 pvcbase-worker52
10.1.49.157 pvcbase-worker53
10.1.49.158 pvcbase-worker54
10.1.49.159 pvcbase-worker55
10.1.49.160 pvcbase-worker56
10.1.49.161 pvcbase-worker57
10.1.49.162 pvcbase-worker58
10.1.49.163 pvcbase-worker59
10.1.49.164 pvcbase-worker60
10.1.49.165 pvcbase-worker61
10.1.49.166 pvcbase-worker62
10.1.49.167 pvcbase-worker63
10.1.49.168 pvcbase-worker64
10.1.49.169 pvcbase-worker65
10.1.49.170 pvcbase-worker66
10.1.49.171 pvcbase-worker67
10.1.49.172 pvcbase-worker68
10.1.49.173 pvcbase-worker69
10.1.49.174 pvcbase-worker70
10.1.49.175 pvcbase-worker71
10.1.49.176 pvcbase-worker72
10.1.49.177 pvcbase-worker73
10.1.49.178 pvcbase-worker74
10.1.49.179 pvcbase-worker75
10.1.49.180 pvcbase-worker76
10.1.49.181 pvcbase-worker77
10.1.49.182 pvcbase-worker78
10.1.49.183 pvcbase-worker79
10.1.49.184 pvcbase-worker80
10.1.49.185 pvcbase-worker81
10.1.49.186 pvcbase-worker82
10.1.49.187 pvcbase-worker83
10.1.49.188 pvcbase-worker84
10.1.49.189 pvcbase-worker85
10.1.49.190 pvcbase-worker86
10.1.49.191 pvcbase-worker87
10.1.49.192 pvcbase-worker88
10.1.49.193 pvcbase-worker89
10.1.49.194 pvcbase-worker90
10.1.49.195 pvcbase-worker91
10.1.49.196 pvcbase-worker92
10.1.49.197 pvcbase-worker93
10.1.49.198 pvcbase-worker94
10.1.49.199 pvcbase-worker95
```

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.cdp.rdu2.scalelab.redhat.com

[root@ipaserver ~]# hostname -i
10.1.49.1

[root@ipaserver ~]# cat /etc/hostname
ipaserver.cdp.rdu2.scalelab.redhat.com

[root@ipaserver ~]# ip addr show eno2np1 | grep -e inet
   inet 10.1.49.1/23 brd 10.1.49.255 scope global noprefixroute eno2np1

[root@ipaserver ~]# ip addr show |grep $(hostname -i)
   inet 10.1.49.1/23 brd 10.1.49.255 scope global noprefixroute eno2np1

[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.cdp.rdu2.scalelab.redhat.com.     1200 IN A 10.1.49.1

[root@ipaserver ~]#
[root@ipaserver ~]# uname -a
Linux ipaserver.cdp.rdu2.scalelab.redhat.com 5.14.0-427.13.1.el9_4.x86_64 #1 SMP PREEMPT_DYNAMIC Wed Apr
10 10:29:16 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.redhat.local]:	ipaserver.redhat.local
Please confirm the domain name [redhat.local]:	REDHAT.LOCAL
Please provide a realm name [redhat.local]:	REDHAT.LOCAL
Directory Manager password:	<Password For Directory Manager> ***(redhat123)***
Password (confirm):	<Confirm Password> ***(redhat123)***
IPA admin password:	<Password For IPA Admin> ***(redhat123)***
Password (confirm):	<Confirm Password> ***(redhat123)***
Do you want to configure DNS forwarders? [yes]:	<ENTER>

Parameter	Value
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>
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 4.12.2

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: master.example.com


Server host name [ipaserver.cdp.rdu2.scalelab.redhat.com]: <ENTER>

Warning: skipping DNS resolution of host ipaserver.cdp.rdu2.scalelab.redhat.com
The domain name has been determined based on the host name.

Please confirm the domain name [cdp.rdu2.scalelab.redhat.com]: <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 [CDP.RDU2.SCALELAB.REDHAT.COM]: <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: <redhat123><ENTER>
Password (confirm): <redhat123><ENTER>

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

IPA admin password: <redhat123><ENTER>
Password (confirm): <redhat123><ENTER>

Checking DNS domain cdp.rdu2.scalelab.redhat.com., please wait ...
Do you want to configure DNS forwarders? [yes]: <ENTER>
Following DNS servers are configured in /etc/resolv.conf: 10.1.37.190, 10.1.36.2
Do you want to configure these servers as DNS forwarders? [yes]: <ENTER>
All detected DNS servers were added. You can enter additional addresses now:
Enter an IP address for a DNS forwarder, or press Enter to skip:
DNS forwarders: 10.1.37.190, 10.1.36.2
Do you want to search for missing reverse zones? [yes]: no <ENTER>
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 [CDP]: <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: foreman.rdu2.scalelab.redhat.com
Enter a NTP source pool address, or press Enter to skip: foreman.rdu2.scalelab.redhat.com <ENTER>

The IPA Master Server will be configured with:
Hostname:       ipaserver.cdp.rdu2.scalelab.redhat.com
IP address(es): 10.1.49.1
Domain name:    cdp.rdu2.scalelab.redhat.com
Realm name:     CDP.RDU2.SCALELAB.REDHAT.COM

The CA will be configured with:
Subject DN:   CN=Certificate Authority,O=CDP.RDU2.SCALELAB.REDHAT.COM
Subject base: O=CDP.RDU2.SCALELAB.REDHAT.COM
Chaining:     self-signed

BIND DNS server will be configured to serve IPA domain with:
Forwarders:       10.1.37.190, 10.1.36.2
Forward policy:   only
Reverse zone(s):  No reverse zone

NTP server:     foreman.rdu2.scalelab.redhat.com
NTP pool:       foreman.rdu2.scalelab.redhat.com
Continue to configure the system with these values? [no]: yes <ENTER>

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

Disabled p11-kit-proxy
Synchronizing time
Configuration of chrony was changed by installer.
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=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com ...
Perform post-installation tasks ...
  [2/43]: tune ldbm plugin
  [3/43]: adding default schema
  [4/43]: enabling memberof plugin
  [5/43]: enabling winsync plugin
  [6/43]: configure password logging
  [7/43]: configuring replication version plugin
  [8/43]: enabling IPA enrollment plugin
  [9/43]: configuring uniqueness plugin
  [10/43]: configuring uuid plugin
  [11/43]: configuring modrdn plugin
  [12/43]: configuring DNS plugin
  [13/43]: enabling entryUSN plugin
  [14/43]: configuring lockout plugin
  [15/43]: configuring graceperiod plugin
  [16/43]: configuring topology plugin
  [17/43]: creating indices
  [18/43]: enabling referential integrity plugin
  [19/43]: configuring certmap.conf
  [20/43]: configure new location for managed entries
  [21/43]: configure dirsrv ccache and keytab
  [22/43]: enabling SASL mapping fallback
  [23/43]: restarting directory server
  [24/43]: adding sasl mappings to the directory
  [25/43]: adding default layout
  [26/43]: adding delegation layout
  [27/43]: creating container for managed entries
  [28/43]: configuring user private groups
  [29/43]: configuring netgroups from hostgroups
  [30/43]: creating default Sudo bind user
  [31/43]: creating default Auto Member layout
  [32/43]: adding range check plugin
  [33/43]: creating default HBAC rule allow_all
  [34/43]: adding entries for topology management
  [35/43]: initializing group membership
  [36/43]: adding master entry
  [37/43]: initializing domain level
  [38/43]: configuring Posix uid/gid generation
  [39/43]: adding replication acis
  [40/43]: activating sidgen plugin
  [41/43]: activating extdom plugin
  [42/43]: configuring directory to start on boot
  [43/43]: restarting directory server
Done configuring directory server (dirsrv).
Configuring Kerberos KDC (krb5kdc)
  [1/11]: adding kerberos container to the directory
  [2/11]: configuring KDC
  [3/11]: initialize kerberos container
  [4/11]: adding default ACIs
  [5/11]: creating a keytab for the directory
  [6/11]: creating a keytab for the machine
  [7/11]: adding the password extension to the directory
  [8/11]: creating anonymous principal
  [9/11]: starting the KDC
  [10/11]: configuring KDC to start on boot
  [11/11]: enable PAC ticket signature support
```

```
Done configuring Kerberos KDC (krb5kdc).
Configuring kadmin
  [1/2]: starting kadmin
  [2/2]: configuring kadmin to start on boot
Done configuring kadmin.
Configuring ipa-custodia
  [1/5]: Making sure custodia container exists
  [2/5]: Generating ipa-custodia config file
  [3/5]: Generating ipa-custodia keys
  [4/5]: starting ipa-custodia
  [5/5]: configuring ipa-custodia to start on boot
Done configuring ipa-custodia.
Configuring certificate server (pki-tomcatd). Estimated time: 3 minutes
  [1/32]: configuring certificate server instance
  [2/32]: stopping certificate server instance to update CS.cfg
  [3/32]: backing up CS.cfg
  [4/32]: Add ipa-pki-wait-running
Set start up timeout of pki-tomcatd service to 90 seconds
  [5/32]: secure AJP connector
  [6/32]: reindex attributes
  [7/32]: exporting Dogtag certificate store pin
  [8/32]: disabling nonces
  [9/32]: set up CRL publishing
  [10/32]: enable PKIX certificate path discovery and validation
  [11/32]: authorizing RA to modify profiles
  [12/32]: authorizing RA to manage lightweight CAs
  [13/32]: Ensure lightweight CAs container exists
  [14/32]: Enable lightweight CA monitor
  [15/32]: Ensuring backward compatibility
  [16/32]: updating IPA configuration
  [17/32]: starting certificate server instance
  [18/32]: configure certmonger for renewals
  [19/32]: requesting RA certificate from CA
  [20/32]: publishing the CA certificate
  [21/32]: adding RA agent as a trusted user
  [22/32]: configure certificate renewals
  [23/32]: Configure HTTP to proxy connections
  [24/32]: enabling CA instance
  [25/32]: importing IPA certificate profiles
  [26/32]: migrating certificate profiles to LDAP
  [27/32]: adding default CA ACL
  [28/32]: adding 'ipa' CA entry
  [29/32]: Recording random serial number state
  [30/32]: Recording HSM configuration state
  [31/32]: configuring certmonger renewal for lightweight CAs
  [32/32]: deploying ACME service
Done configuring certificate server (pki-tomcatd).
Configuring directory server (dirsrv)
  [1/3]: configuring TLS for DS instance
  [2/3]: adding CA certificate entry
  [3/3]: restarting directory server
Done configuring directory server (dirsrv).
Configuring ipa-otpd
  [1/2]: starting ipa-otpd
  [2/2]: configuring ipa-otpd to start on boot
Done configuring ipa-otpd.
Configuring the web interface (httpd)
  [1/22]: stopping httpd
  [2/22]: backing up ssl.conf
  [3/22]: disabling nss.conf
  [4/22]: configuring mod_ssl certificate paths
  [5/22]: setting mod_ssl protocol list
```

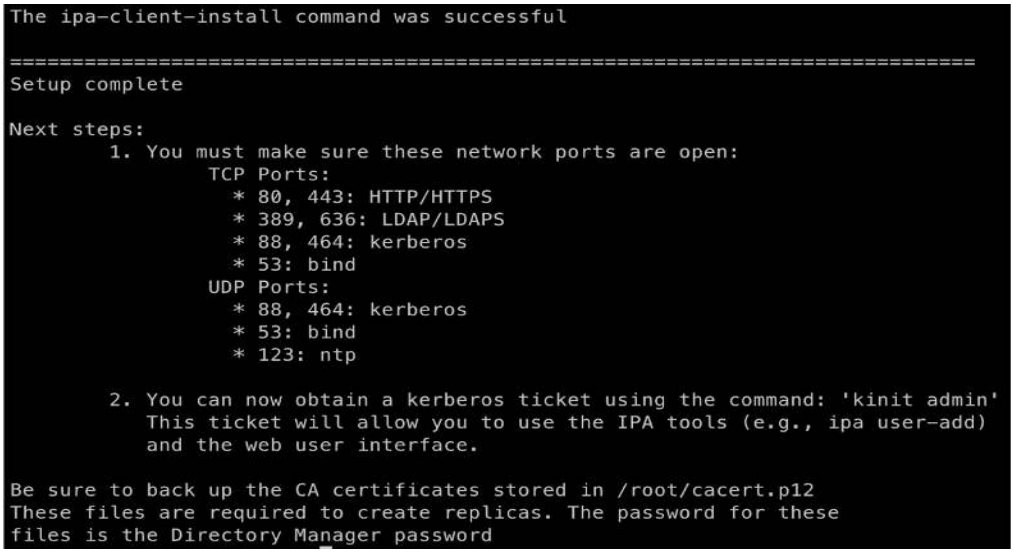
```
  [6/22]: configuring mod_ssl log directory
  [7/22]: disabling mod_ssl OCSP
  [8/22]: adding URL rewriting rules
  [9/22]: configuring httpd
Nothing to do for configure_httpd_wsgi_conf
  [10/22]: setting up httpd keytab
  [11/22]: configuring Gssproxy
  [12/22]: setting up ssl
  [13/22]: configure certmonger for renewals
  [14/22]: publish CA cert
  [15/22]: clean up any existing httpd ccaches
  [16/22]: enable ccache sweep
  [17/22]: configuring SELinux for httpd
  [18/22]: create KDC proxy config
  [19/22]: enable KDC proxy
  [20/22]: starting httpd
  [21/22]: configuring httpd to start on boot
  [22/22]: enabling oddjobd
Done configuring the web interface (httpd).
Configuring Kerberos KDC (krb5kdc)
  [1/1]: installing X509 Certificate for PKINIT
Done configuring Kerberos KDC (krb5kdc).
Applying LDAP updates
Upgrading IPA:. Estimated time: 1 minute 30 seconds
  [1/10]: stopping directory server
  [2/10]: saving configuration
  [3/10]: disabling listeners
  [4/10]: enabling DS global lock
  [5/10]: disabling Schema Compat
  [6/10]: starting directory server
  [7/10]: upgrading server
  [8/10]: stopping directory server
  [9/10]: restoring configuration
  [10/10]: starting directory server
Done.
Restarting the KDC
dnssec-validation no
Configuring DNS (named)
  [1/12]: generating rndc key file
  [2/12]: adding DNS container
  [3/12]: setting up our zone
  [4/12]: setting up our own record
  [5/12]: setting up records for other masters
  [6/12]: adding NS record to the zones
  [7/12]: setting up kerberos principal
  [8/12]: setting up LDAPI autobind
  [9/12]: setting up named.conf
created new /etc/named.conf
created named user config '/etc/named/ipa-ext.conf'
created named user config '/etc/named/ipa-options-ext.conf'
created named user config '/etc/named/ipa-logging-ext.conf'
  [10/12]: setting up server configuration
  [11/12]: configuring named to start on boot
  [12/12]: changing resolv.conf to point to ourselves
Done configuring DNS (named).
Restarting the web server to pick up resolv.conf changes
Configuring DNS key synchronization service (ipa-dnskeysyncd)
  [1/7]: checking status
  [2/7]: setting up bind-dyndb-ldap working directory
  [3/7]: setting up kerberos principal
  [4/7]: setting up SoftHSM
  [5/7]: adding DNSSEC containers
```

```
  [6/7]: creating replica keys
  [7/7]: configuring ipa-dnskeysyncd to start on boot
Done configuring DNS key synchronization service (ipa-dnskeysyncd).
Restarting ipa-dnskeysyncd
Restarting named
Updating DNS system records
Configuring SID generation
  [1/8]: adding RID bases
  [2/8]: creating samba domain object
  [3/8]: adding admin(group) SIDs
  [4/8]: updating Kerberos config
'dns_lookup_kdc' already set to 'true', nothing to do.
  [5/8]: activating sidgen task
  [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 considerable amount of time, please wait..
Done.
Configuring client side components
This program will set up IPA client.
Version 4.12.2

Using existing certificate '/etc/ipa/ca.crt'.
Client hostname: ipaserver.cdp.rdu2.scalelab.redhat.com
Realm: CDP.RDU2.SCALELAB.REDHAT.COM
DNS Domain: cdp.rdu2.scalelab.redhat.com
IPA Server: ipaserver.cdp.rdu2.scalelab.redhat.com
BaseDN: dc=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com

Configured /etc/sssd/sssd.conf
Systemwide CA database updated.
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ed25519_key.pub
Adding SSH public key from /etc/ssh/ssh_host_rsa_key.pub
SSSD enabled
Configured /etc/openldap/ldap.conf
Configured /etc/ssh/ssh_config
Configured /etc/ssh/sshd_config.d/04-ipa.conf
Configuring cdp.rdu2.scalelab.redhat.com 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 ~]#

##### 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@CDP.VLAN601.RDU2.SCALELAB.REDHAT.COM:<redhat123>

[root@ipaserver ~]# klist -e
Ticket cache: KCM:0
Default principal: admin@CDP.RDU2.SCALELAB.REDHAT.COM

Valid starting       Expires              Service principal
10/26/2025 00:36:36  10/26/2025 23:47:11
krbtgt/CDP.RDU2.SCALELAB.REDHAT.COM@CDP.RDU2.SCALELAB.REDHAT.COM
        Etype (skey, tkt): aes256-cts-hmac-sha384-192, aes256-cts-hmac-sha384-192

##### try kinit admin@redhat.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:

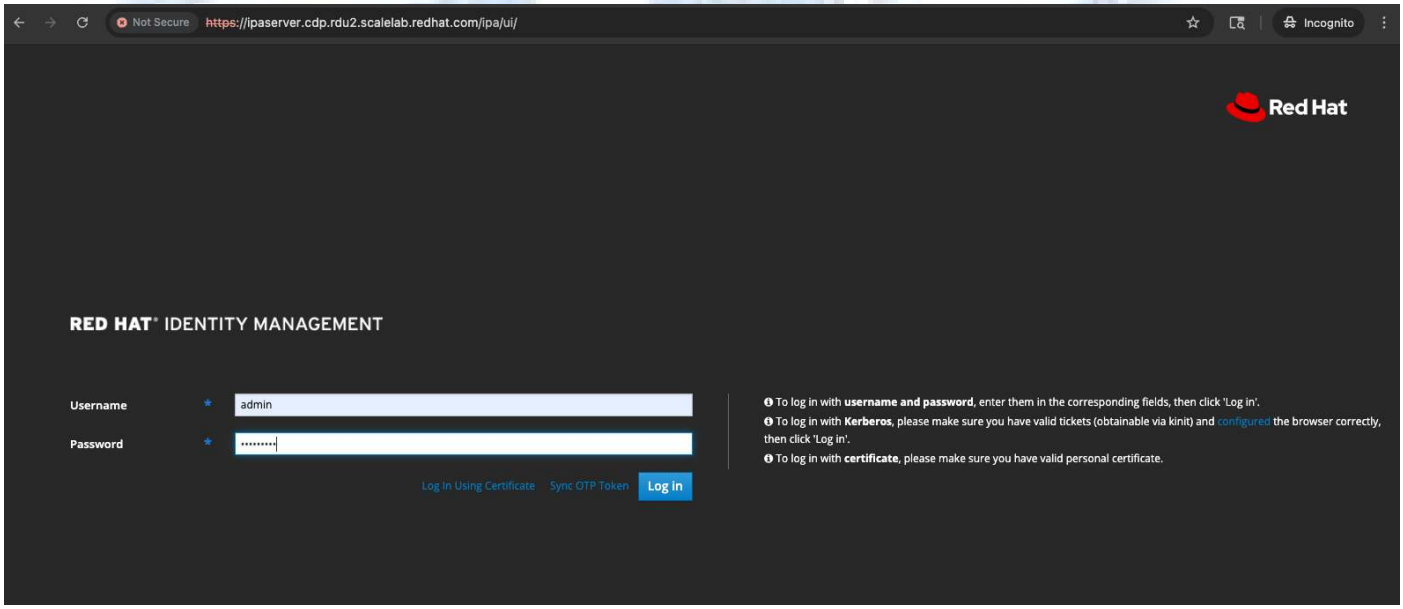
```
[root@ipaserver centos]# kinit admin
Password for admin@CDPPVCDS.COM:
[root@ipaserver centos]# klist -e
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: admin@CDPPVCDS.COM

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:

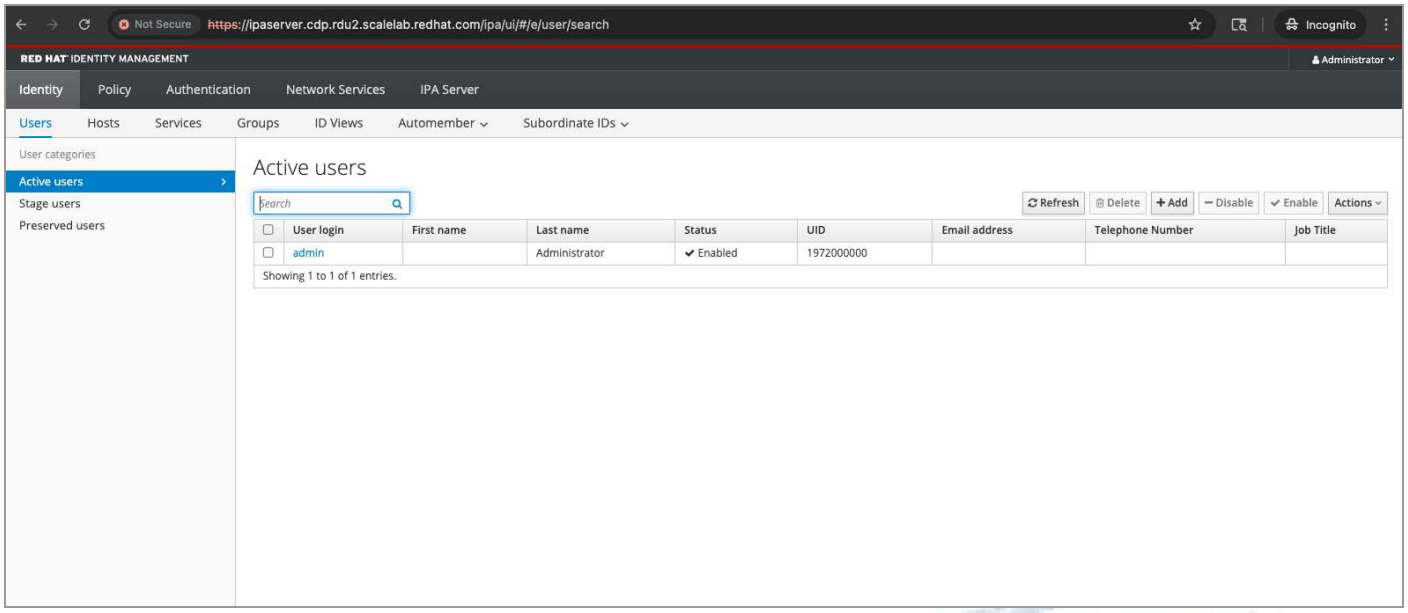
Note: Once the FreeIPA server is successfully installed, the FreeIPA Web UI is automatically set up as part of the installation process. You can access the Web UI directly by putting either the hostname or the IP address of the IPA server into your browser (You'll need to add the ***hostname<->IP*** 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
10.1.49.1   ipaserver.cdp.rdu2.scalelab.redhat.com   ipaserver


##### Access the below URL on browser, and the IPA Admin console will open.
https://ipaserver.cdp.rdu2.scalelab.redhat.com/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/redhat123)***



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



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

The RHEL IdM/IPA server was set up on a separate VM, running outside from OpenShift Virtualization. Since OpenShift requires a DNS server as a prerequisite to installation, the ipa server was set up prior to installing OpenShift.

DNS & IP Address assignments:

Hostname	IP Address
api.ocp.redhat.local	192.168.2.183
*.apps.ocp.redhat.local	192.168.2.184
c240m4-01.redhat.local (master/worker)	192.168.2.170
c240m4-02.redhat.local (master/worker)	192.168.2.171
c240m4-03.redhat.local (master/worker)	192.168.2.172
c220m4-01.redhat.local (worker)	192.168.2.180
c220m4-02.redhat.local (worker)	192.168.2.181
c220m4-03.redhat.local (worker)	192.168.2.182

Metallb IPAddressPool	192.168.2.20-192.168.2.25

For additional information, refer to the Red Hat Identity Management (IdM) documentation:

- RHEL 9: Installing Identity Management
  - Installing an IdM server: With integrated DNS, without a CA
    - Certificates required to install an IdM server without a CA
- RHEL 9: Planning Identity Management
- RHEL 9: Managing IdM users, groups, hosts, and access control rules
- RHEL 9: Managing certificates in IdM
- RHEL 9: Working with DNS in Identity Management
- RHEL 9: Using Ansible to install and manage Identity Management
- RHEL 9: Accessing Identity Management services

- How can we create a CA-less Identity Managment setup?

Install ipa server dependencies packages through dnf using the following command.

[root@idm ~]# **dnf install -y ipa-server bind bind-dyndb-ldap ipa-server-dns firewalld**

Run the below command to uninstall the previous installation that used an integrated CA.

[root@idm ~]# ***ipa-server-install --uninstall***

```
This is a NON REVERSIBLE operation and will delete all data and configuration!
It is highly recommended to take a backup of existing data and configuration using ipa-backup utility before
proceeding.

Are you sure you want to continue with the uninstall procedure? [no]: yes
Updating DNS system records
Forcing removal of idm.redhat.local
----------------------------------------------------------------------
Deleted IPA server "idm.redhat.local"
----------------------------------------------------------------------
Unconfiguring CA
Shutting down all IPA services
Unconfiguring named
Unconfiguring ipa-dnskeysyncd
Unconfiguring web server
Unconfiguring krb5kdc
Unconfiguring kadmin
Unconfiguring directory server
Unconfiguring ipa-custodia
Unconfiguring ipa-otpd
Removing IPA client configuration
Removing Kerberos service principals from /etc/krb5.keytab
Disabling client Kerberos and LDAP configurations
Redundant SSSD configuration file /etc/sssd/sssd.conf was moved to /etc/sssd/sssd.conf.deleted
Restoring client configuration files
Restoring rdu3.labs.perfscale.redhat.com as NIS domain.
```

```
nscd daemon is not installed, skip configuration
nslcd daemon is not installed, skip configuration
Systemwide CA database updated.
Client uninstall complete.
The ipa-client-install command was successful
The ipa-server-install command was successful


# Configure ipa-server with integrated DNS by using the following command:

[root@idm ~]# 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 4.12.2

This includes:
  * 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

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: master.example.com

Server host name [idm.redhat.local]: <ENTER>

Warning: skipping DNS resolution of host
ipaserver.cdppvcds.rdu3.labs.perfscale.redhat.com
The domain name has been determined based on the host name.

Please confirm the domain name [redhat.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 [REDHAT.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: <redhat123><ENTER>
Password (confirm): <redhat123><ENTER>

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

IPA admin password: <redhat123><ENTER>
Password (confirm): <redhat123><ENTER>

Checking DNS domain redhat.local., please wait ...
Do you want to configure DNS forwarders? [yes]: <ENTER>
Following DNS servers are configured in /etc/resolv.conf: 8.8.8.8
Do you want to configure these servers as DNS forwarders? [yes]: <ENTER>
All detected DNS servers were added. You can enter additional addresses now:
Enter an IP address for a DNS forwarder, or press Enter to skip: <ENTER>

DNS forwarders: 8.8.8.8
Checking DNS forwarders, please wait ...
DNS server 8.8.8.8 does not support DNSSEC: answer to query '. SOA' is missing DNSSEC
signatures (no RRSIG data)
Please fix forwarder configuration to enable DNSSEC support.

DNS server 8.8.8.8: answer to query '. SOA' is missing DNSSEC signatures (no RRSIG
data)
Please fix forwarder configuration to enable DNSSEC support.

DNS server 8.8.8.8: answer to query '. SOA' is missing DNSSEC signatures (no RRSIG
data)
Please fix forwarder configuration to enable DNSSEC support.
WARNING: DNSSEC validation will be disabled

Do you want to search for missing reverse zones? [yes]: <no> <ENTER>
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 [CDPPVCDS]: <ENTER>

Do you want to configure chrony with NTP server or pool address? [no]: <yes> <ENTER>
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:       idm.redhat.local
IP address(es): 192.168.1.210
Domain name:    redhat.local
Realm name:     REDHAT.LOCAL

BIND DNS server will be configured to serve IPA domain with:
Forwarders:       8.8.8.8
Forward policy:   only
Reverse zone(s):  No reverse zone

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

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

Disabled p11-kit-proxy
Synchronizing time
Configuration of chrony was changed by installer.
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=redhat,dc=local ...
Perform post-installation tasks ...
  [2/43]: tune ldbm plugin
  [3/43]: adding default schema
  [4/43]: enabling memberof plugin
  [5/43]: enabling winsync plugin
  [6/43]: configure password logging
  [7/43]: configuring replication version plugin
  [8/43]: enabling IPA enrollment plugin
  [9/43]: configuring uniqueness plugin
  [10/43]: configuring uuid plugin
  [11/43]: configuring modrdn plugin
  [12/43]: configuring DNS plugin
  [13/43]: enabling entryUSN plugin
  [14/43]: configuring lockout plugin
  [15/43]: configuring graceperiod plugin
  [16/43]: configuring topology plugin
  [17/43]: creating indices
```

```
  [18/43]: enabling referential integrity plugin
  [19/43]: configuring certmap.conf
  [20/43]: configure new location for managed entries
  [21/43]: configure dirsrv ccache and keytab
  [22/43]: enabling SASL mapping fallback
  [23/43]: restarting directory server
  [24/43]: adding sasl mappings to the directory
  [25/43]: adding default layout
  [26/43]: adding delegation layout
  [27/43]: creating container for managed entries
  [28/43]: configuring user private groups
  [29/43]: configuring netgroups from hostgroups
  [30/43]: creating default Sudo bind user
  [31/43]: creating default Auto Member layout
  [32/43]: adding range check plugin
  [33/43]: creating default HBAC rule allow_all
  [34/43]: adding entries for topology management
  [35/43]: initializing group membership
  [36/43]: adding master entry
  [37/43]: initializing domain level
  [38/43]: configuring Posix uid/gid generation
  [39/43]: adding replication acis
  [40/43]: activating sidgen plugin
  [41/43]: activating extdom plugin
  [42/43]: configuring directory to start on boot
  [43/43]: restarting directory server
Done configuring directory server (dirsrv).
Configuring Kerberos KDC (krb5kdc)
  [1/11]: adding kerberos container to the directory
  [2/11]: configuring KDC
  [3/11]: initialize kerberos container
  [4/11]: adding default ACIs
  [5/11]: creating a keytab for the directory
  [6/11]: creating a keytab for the machine
  [7/11]: adding the password extension to the directory
  [8/11]: creating anonymous principal
  [9/11]: starting the KDC
  [10/11]: configuring KDC to start on boot
  [11/11]: enable PAC ticket signature support
Done configuring Kerberos KDC (krb5kdc).
Configuring kadmin
  [1/2]: starting kadmin
  [2/2]: configuring kadmin to start on boot
Done configuring kadmin.
Configuring ipa-custodia
```

```
  [1/5]: Making sure custodia container exists
  [2/5]: Generating ipa-custodia config file
  [3/5]: Generating ipa-custodia keys
  [4/5]: starting ipa-custodia
  [5/5]: configuring ipa-custodia to start on boot
Done configuring ipa-custodia.
Configuring directory server (dirsrv)
  [1/3]: configuring TLS for DS instance
  [2/3]: adding CA certificate entry
  [3/3]: restarting directory server
Done configuring directory server (dirsrv).
Configuring ipa-otpd
  [1/2]: starting ipa-otpd
  [2/2]: configuring ipa-otpd to start on boot
Done configuring ipa-otpd.
Configuring the web interface (httpd)
  [1/21]: stopping httpd
  [2/21]: backing up ssl.conf
  [3/21]: disabling nss.conf
  [4/21]: configuring mod_ssl certificate paths
  [5/21]: setting mod_ssl protocol list
  [6/21]: configuring mod_ssl log directory
  [7/21]: disabling mod_ssl OCSP
  [8/21]: adding URL rewriting rules
  [9/21]: configuring httpd
Nothing to do for configure_httpd_wsgi_conf
  [10/21]: setting up httpd keytab
  [11/21]: configuring Gssproxy
  [12/21]: setting up ssl
  [13/21]: publish CA cert
  [14/21]: clean up any existing httpd ccaches
  [15/21]: enable ccache sweep
  [16/21]: configuring SELinux for httpd
  [17/21]: create KDC proxy config
  [18/21]: enable KDC proxy
  [19/21]: starting httpd
  [20/21]: configuring httpd to start on boot
  [21/21]: enabling oddjobd
Done configuring the web interface (httpd).
Applying LDAP updates
Upgrading IPA:. Estimated time: 1 minute 30 seconds
  [1/10]: stopping directory server
  [2/10]: saving configuration
  [3/10]: disabling listeners
  [4/10]: enabling DS global lock
```

```
  [5/10]: disabling Schema Compat
  [6/10]: starting directory server
  [7/10]: upgrading server
Could not get dnaHostname entries in 60 seconds
Could not get dnaHostname entries in 60 seconds
  [8/10]: stopping directory server
  [9/10]: restoring configuration
  [10/10]: starting directory server
Done.
Restarting the KDC
dnssec-validation no
Configuring DNS (named)
  [1/12]: generating rndc key file
  [2/12]: adding DNS container
  [3/12]: setting up our zone
  [4/12]: setting up our own record
  [5/12]: setting up records for other masters
  [6/12]: adding NS record to the zones
  [7/12]: setting up kerberos principal
  [8/12]: setting up LDAPI autobind
  [9/12]: setting up named.conf
created new /etc/named.conf
created named user config '/etc/named/ipa-ext.conf'
created named user config '/etc/named/ipa-options-ext.conf'
named user config '/etc/named/ipa-logging-ext.conf' already exists
  [10/12]: setting up server configuration
  [11/12]: configuring named to start on boot
  [12/12]: changing resolv.conf to point to ourselves
Done configuring DNS (named).
Restarting the web server to pick up resolv.conf changes
Configuring DNS key synchronization service (ipa-dnskeysyncd)
  [1/7]: checking status
  [2/7]: setting up bind-dyndb-ldap working directory
  [3/7]: setting up kerberos principal
  [4/7]: setting up SoftHSM
  [5/7]: adding DNSSEC containers
  [6/7]: creating replica keys
  [7/7]: configuring ipa-dnskeysyncd to start on boot
Done configuring DNS key synchronization service (ipa-dnskeysyncd).
Restarting ipa-dnskeysyncd
Restarting named
Updating DNS system records
Configuring SID generation
  [1/8]: adding RID bases
  [2/8]: creating samba domain object
```

```
  [3/8]: adding admin(group) SIDs
  [4/8]: updating Kerberos config
'dns_lookup_kdc' already set to 'true', nothing to do.
  [5/8]: activating sidgen task
  [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 considerable amount of time, please wait..
Done.
Configuring client side components
This program will set up IPA client.
Version 4.12.2

Using existing certificate '/etc/ipa/ca.crt'.
Client hostname: idm.redhat.local
Realm: REDHAT.LOCAL
DNS Domain: redhat.local
IPA Server: idm.redhat.local
BaseDN: dc=redhat,dc=local

Configured /etc/sssd/sssd.conf
Systemwide CA database updated.
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ed25519_key.pub
Adding SSH public key from /etc/ssh/ssh_host_rsa_key.pub
SSSD enabled
Configured /etc/openldap/ldap.conf
Configured /etc/ssh/ssh_config
Configured /etc/ssh/sshd_config.d/04-ipa.conf
Configuring redhat.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.

The ipa-server-install command was successful
```

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

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

```
##### 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@idm ~]# kinit admin
Password for admin@REDHAT.LOCAL: <redhat123>
```

```
[root@idm ~]# klist -e
Ticket cache: KCM:0
Default principal: admin@REDHAT.LOCAL

Valid starting       Expires              Service principal
05/19/2025 14:49:13  05/20/2025 14:38:23  krbtgt/REDHAT.LOCAL@REDHAT.LOCAL
       Etype (skey, tkt): aes256-cts-hmac-sha384-192, aes256-cts-hmac-sha384-192
```

```
##### Verify the status of ipa services installed
[root@idm ~]# 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

Access the IPA Admin console: https://idm.redhat.local/ipa/ui/

You will see below WebUI for IPAServer Administration. Enter the same admin credentials used for CLI authentication: (i.e. *admin/redhat123*)

Procedure 3. Set Up Password-less Login

In OpenShift Virtualization, a public SSH key was set to automatically apply to any new VirtualMachines created within the ***cdpvcbase*** project:
Create SSH Keys in Project

In addition, when the VMs were created, they applied a cloud-init script that created a user account: ***cloud-user*** with a password of *"r3dh4t!"*.
Create VMs on OpenShift Virtualization

However, the "authorized_keys" file for the root account does not allow login as root. So, we will have to login as cloud-user and use sudo command to edit it to allow this.

Edit /root/.ssh/authorized_keys to remove the first part of line that does not allow login as root:

cat /root/.ssh/authorized_keys

```
no-port-forwarding,no-agent-forwarding,no-X11-forwarding,command="echo 'Please login as the user \"cloud-user\" rather than the user \"root\".';echo;sleep 10;exit 142" ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQC5f46fvSfwLSXxOzCC52qFa21AS0kvJoolggoGL+CUWGdq6Kf4ExWCJ2d1iDkAcPt
6y7jGshujhfUR6DE1kuFpozTOgqBjwZl7ICtA5lmkhb7sxHuYYNFAZNli6D38IbdMSfth23I2ypBpcQAOWweCzfPnjsjani
LG5EfTWw3K75QQOMxLvE4W0uKbA/EKg9F8Eq2ImKLHZekB8nCFoi2mVgwEYGTSMSD5zAz56tuSbuppbkXfgf+4FFb/LFp3T
JyCPQvu396ZRbEF64EWvbMC01aAYNM3NpRqSqsdGWWJoTnRzXSRKb096tOxWmYTyRIfOs/nuRAlSb3kU0EN4qC5YQs9wobp
nLQ5T3Um8QP4pQxWhWIHEAW3SWMOlIfeBjT8TPS8OCG3EfLRVxJFdNZc/IfD3CGnevW8ovlSek//HfspwMMMauXQfY57OfX
hn9BxVtnC6ymGlDBv9hoa82qtvigfkMU1uOxD9Ep7rl14vprcQ5PFl8ypO9DPsKj4Etc=
root@ipaserver.cdp.rdu2.scalelab.redhat.com
```

vi /root/.ssh/authorized_keys

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQC5f46fvSfwLSXxOzCC52qFa21AS0kvJoolggoGL+CUWGdq6Kf4ExWCJ2d1iDkAcPt
6y7jGshujhfUR6DE1kuFpozTOgqBjwZl7ICtA5lmkhb7sxHuYYNFAZNli6D38IbdMSfth23I2ypBpcQAOWweCzfPnjsjani
```

```
LG5EfTWw3K75QQOMxLvE4W0uKbA/EKg9F8Eq2ImKLHZekB8nCFoi2mVgwEYGTSMSD5zAz56tuSbuppbkXfgf+4FFb/LFp3T
JyCPQvu396ZRbEF64EWvbMC01aAYNM3NpRqSqsdGWWJoTnRzXSRKb096tOxWmYTyRIfOs/nuRAlSb3kU0EN4qC5YQs9wobp
nLQ5T3Um8QP4pQxWhWIHEAW3SWMOlIfeBjT8TPS8OCG3EfLRVxJFdNZc/IfD3CGnevW8ovlSek//HfspwMMMauXQfY57OfX
hn9BxVtnC6ymGlDBv9hoa82qtvigfkMU1uOxD9Ep7rl14vprcQ5PFl8ypO9DPsKj4Etc=
root@ipaserver.cdp.rdu2.scalelab.redhat.com
```

Alternatively, instead of editing the above authorized_keys file, simply copy it from the cloud-user:

```
[root@ipaserver ~]# ssh cloud-user@cldr-mngr.redhat.local
[root@ipaserver ~]# sudo cp .ssh/authorized_keys /root/.ssh/authorized_keys
```

The following command was used to copy the authorized_keys file to the root user from the cloud-user user for all 100 of the VMs running in the scale lab:

```
[root@ipaserver ~]# for i in {100..199}; do ssh -t -o StrictHostKeyChecking=accept-new
cloud-user@"10.1.49.$i" sudo -- "sh -c 'cp .ssh/authorized_keys /root/.ssh/authorized_keys'";
done

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

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 2.14.18]
  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 = /usr/bin/ansible
  python version = 3.9.18 (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.cdp.rdu2.scalelab.redhat.com

[ipaserver]
ipaserver.cdp.rdu2.scalelab.redhat.com

[cldr-mngr]
cldr-mngr.cdp.rdu2.scalelab.redhat.com

[cldr-utility]
cldr-utility.cdp.rdu2.scalelab.redhat.com

[namenodes]
```

```
pvcbase-master01.cdp.rdu2.scalelab.redhat.com
pvcbase-master02.cdp.rdu2.scalelab.redhat.com
pvcbase-master03.cdp.rdu2.scalelab.redhat.com

[ocpmasternodes]
pvcbase-master01.cdp.rdu2.scalelab.redhat.com
pvcbase-master02.cdp.rdu2.scalelab.redhat.com
pvcbase-master03.cdp.rdu2.scalelab.redhat.com

[datanodes]
pvcbase-worker01.cdp.rdu2.scalelab.redhat.com
pvcbase-worker02.cdp.rdu2.scalelab.redhat.com
pvcbase-worker03.cdp.rdu2.scalelab.redhat.com
pvcbase-worker04.cdp.rdu2.scalelab.redhat.com
pvcbase-worker05.cdp.rdu2.scalelab.redhat.com
pvcbase-worker06.cdp.rdu2.scalelab.redhat.com
pvcbase-worker07.cdp.rdu2.scalelab.redhat.com
pvcbase-worker08.cdp.rdu2.scalelab.redhat.com
pvcbase-worker09.cdp.rdu2.scalelab.redhat.com
pvcbase-worker10.cdp.rdu2.scalelab.redhat.com
pvcbase-worker11.cdp.rdu2.scalelab.redhat.com
pvcbase-worker12.cdp.rdu2.scalelab.redhat.com
pvcbase-worker13.cdp.rdu2.scalelab.redhat.com
pvcbase-worker14.cdp.rdu2.scalelab.redhat.com
pvcbase-worker15.cdp.rdu2.scalelab.redhat.com
pvcbase-worker16.cdp.rdu2.scalelab.redhat.com
pvcbase-worker17.cdp.rdu2.scalelab.redhat.com
pvcbase-worker18.cdp.rdu2.scalelab.redhat.com
pvcbase-worker19.cdp.rdu2.scalelab.redhat.com
pvcbase-worker20.cdp.rdu2.scalelab.redhat.com
pvcbase-worker21.cdp.rdu2.scalelab.redhat.com
pvcbase-worker22.cdp.rdu2.scalelab.redhat.com
pvcbase-worker23.cdp.rdu2.scalelab.redhat.com
pvcbase-worker24.cdp.rdu2.scalelab.redhat.com
pvcbase-worker25.cdp.rdu2.scalelab.redhat.com
pvcbase-worker26.cdp.rdu2.scalelab.redhat.com
pvcbase-worker27.cdp.rdu2.scalelab.redhat.com
pvcbase-worker28.cdp.rdu2.scalelab.redhat.com
pvcbase-worker29.cdp.rdu2.scalelab.redhat.com
pvcbase-worker30.cdp.rdu2.scalelab.redhat.com
pvcbase-worker31.cdp.rdu2.scalelab.redhat.com
pvcbase-worker32.cdp.rdu2.scalelab.redhat.com
pvcbase-worker33.cdp.rdu2.scalelab.redhat.com
pvcbase-worker34.cdp.rdu2.scalelab.redhat.com
pvcbase-worker35.cdp.rdu2.scalelab.redhat.com
pvcbase-worker36.cdp.rdu2.scalelab.redhat.com
pvcbase-worker37.cdp.rdu2.scalelab.redhat.com
pvcbase-worker38.cdp.rdu2.scalelab.redhat.com
pvcbase-worker39.cdp.rdu2.scalelab.redhat.com
pvcbase-worker40.cdp.rdu2.scalelab.redhat.com
pvcbase-worker41.cdp.rdu2.scalelab.redhat.com
pvcbase-worker42.cdp.rdu2.scalelab.redhat.com
pvcbase-worker43.cdp.rdu2.scalelab.redhat.com
pvcbase-worker44.cdp.rdu2.scalelab.redhat.com
pvcbase-worker45.cdp.rdu2.scalelab.redhat.com
pvcbase-worker46.cdp.rdu2.scalelab.redhat.com
pvcbase-worker47.cdp.rdu2.scalelab.redhat.com
pvcbase-worker48.cdp.rdu2.scalelab.redhat.com
pvcbase-worker49.cdp.rdu2.scalelab.redhat.com
pvcbase-worker50.cdp.rdu2.scalelab.redhat.com
pvcbase-worker51.cdp.rdu2.scalelab.redhat.com
pvcbase-worker52.cdp.rdu2.scalelab.redhat.com
pvcbase-worker53.cdp.rdu2.scalelab.redhat.com
pvcbase-worker54.cdp.rdu2.scalelab.redhat.com
pvcbase-worker55.cdp.rdu2.scalelab.redhat.com
pvcbase-worker56.cdp.rdu2.scalelab.redhat.com
pvcbase-worker57.cdp.rdu2.scalelab.redhat.com
pvcbase-worker58.cdp.rdu2.scalelab.redhat.com
pvcbase-worker59.cdp.rdu2.scalelab.redhat.com
pvcbase-worker60.cdp.rdu2.scalelab.redhat.com
pvcbase-worker61.cdp.rdu2.scalelab.redhat.com
pvcbase-worker62.cdp.rdu2.scalelab.redhat.com
pvcbase-worker63.cdp.rdu2.scalelab.redhat.com
pvcbase-worker64.cdp.rdu2.scalelab.redhat.com
```

```
pvcbase-worker65.cdp.rdu2.scalelab.redhat.com
pvcbase-worker66.cdp.rdu2.scalelab.redhat.com
pvcbase-worker67.cdp.rdu2.scalelab.redhat.com
pvcbase-worker68.cdp.rdu2.scalelab.redhat.com
pvcbase-worker69.cdp.rdu2.scalelab.redhat.com
pvcbase-worker70.cdp.rdu2.scalelab.redhat.com
pvcbase-worker71.cdp.rdu2.scalelab.redhat.com
pvcbase-worker72.cdp.rdu2.scalelab.redhat.com
pvcbase-worker73.cdp.rdu2.scalelab.redhat.com
pvcbase-worker74.cdp.rdu2.scalelab.redhat.com
pvcbase-worker75.cdp.rdu2.scalelab.redhat.com
pvcbase-worker76.cdp.rdu2.scalelab.redhat.com
pvcbase-worker77.cdp.rdu2.scalelab.redhat.com
pvcbase-worker78.cdp.rdu2.scalelab.redhat.com
pvcbase-worker79.cdp.rdu2.scalelab.redhat.com
pvcbase-worker80.cdp.rdu2.scalelab.redhat.com
pvcbase-worker81.cdp.rdu2.scalelab.redhat.com
pvcbase-worker82.cdp.rdu2.scalelab.redhat.com
pvcbase-worker83.cdp.rdu2.scalelab.redhat.com
pvcbase-worker84.cdp.rdu2.scalelab.redhat.com
pvcbase-worker85.cdp.rdu2.scalelab.redhat.com
pvcbase-worker86.cdp.rdu2.scalelab.redhat.com
pvcbase-worker87.cdp.rdu2.scalelab.redhat.com
pvcbase-worker88.cdp.rdu2.scalelab.redhat.com
pvcbase-worker89.cdp.rdu2.scalelab.redhat.com
pvcbase-worker90.cdp.rdu2.scalelab.redhat.com
pvcbase-worker91.cdp.rdu2.scalelab.redhat.com
pvcbase-worker92.cdp.rdu2.scalelab.redhat.com
pvcbase-worker93.cdp.rdu2.scalelab.redhat.com
pvcbase-worker94.cdp.rdu2.scalelab.redhat.com
pvcbase-worker95.cdp.rdu2.scalelab.redhat.com
```

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

```
[root@ipaserver ~]# ansible datanodes -m ping
[WARNING]: Invalid characters were found in group names but not replaced, use -vvvv to see details
pvcbase-worker02.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker04.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker05.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker01.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker03.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

```
pvcbase-worker06.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker07.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker08.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker09.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker10.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker11.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker14.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker13.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker12.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker15.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker16.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
```

```
    "changed": false,
    "ping": "pong"
}
pvcbase-worker17.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker19.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker20.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker18.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker21.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker22.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker23.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker24.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker25.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker26.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker27.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
```

```
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker28.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker29.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker31.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker32.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker33.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker30.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker34.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker35.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker36.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker37.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
```

```
    "ping": "pong"
}
pvcbase-worker39.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker38.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker40.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker41.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker42.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker43.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker44.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker46.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker45.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker47.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker49.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
```

```
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker48.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker50.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker51.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker53.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker54.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker52.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker55.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker56.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker57.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker58.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
```

```
}
pvcbase-worker59.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker61.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker62.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker63.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker60.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker64.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker66.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker65.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker69.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker67.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker68.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
```

```
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker70.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker71.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker72.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker73.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker74.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker76.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker75.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker77.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker78.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker79.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

```
pvcbase-worker80.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker81.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker82.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker83.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker84.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker85.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker86.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker87.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker89.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker88.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker91.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
```

```
    "changed": false,
    "ping": "pong"
}
pvcbase-worker90.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker92.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker93.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker94.cdp.rdu2.scalelab.redhat.com | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
pvcbase-worker95.cdp.rdu2.scalelab.redhat.com | 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. 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@idm ~]# kinit admin
Password for admin@CDP.RDU2.SCALELAB.REDHAT.COM: <redhat123>

[root@ipaserver ~]# ipa dnszone-add --name-from-ip=10.1.48.0/23 --skip-overlap-check
Zone name [1.10.in-addr.arpa.]:
  Zone name: 1.10.in-addr.arpa..
  Active zone: True
  Authoritative nameserver: ipaserver.cdp.rdu2.scalelab.redhat.com.
  Administrator e-mail address: hostmaster
  SOA serial: 1715598489
  SOA refresh: 3600
  SOA retry: 900
  SOA expire: 1209600
  SOA minimum: 3600
  BIND update policy: grant CDP.RDU2.SCALELAB.REDHAT.COM krb5-subdomain 1.10.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 2. 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 3. 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 OCP 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.redhat.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.
```

ADD CDP BASE NODES DNS:

Generate DNS records for RH Scalelab VMs

```
[root@ipaserver ~]# vms=(cldr-mngr cldr-utility pvcbase-master01 pvcbase-master02 pvcbase-master03
pvcbase-worker01 pvcbase-worker02 pvcbase-worker03 pvcbase-worker04 pvcbase-worker05 pvcbase-worker06
pvcbase-worker07 pvcbase-worker08 pvcbase-worker09 pvcbase-worker10 pvcbase-worker11 pvcbase-worker12
pvcbase-worker13 pvcbase-worker14 pvcbase-worker15 pvcbase-worker16 pvcbase-worker17 pvcbase-worker18
pvcbase-worker19 pvcbase-worker20 pvcbase-worker21 pvcbase-worker22 pvcbase-worker23 pvcbase-worker24
pvcbase-worker25 pvcbase-worker26 pvcbase-worker27 pvcbase-worker28 pvcbase-worker29 pvcbase-worker30
pvcbase-worker31 pvcbase-worker32 pvcbase-worker33 pvcbase-worker34 pvcbase-worker35 pvcbase-worker36
pvcbase-worker37 pvcbase-worker38 pvcbase-worker39 pvcbase-worker40 pvcbase-worker41 pvcbase-worker42
pvcbase-worker43 pvcbase-worker44 pvcbase-worker45 pvcbase-worker46 pvcbase-worker47 pvcbase-worker48
pvcbase-worker49 pvcbase-worker50 pvcbase-worker51 pvcbase-worker52 pvcbase-worker53 pvcbase-worker54
pvcbase-worker55 pvcbase-worker56 pvcbase-worker57 pvcbase-worker58 pvcbase-worker59 pvcbase-worker60
pvcbase-worker61 pvcbase-worker62 pvcbase-worker63 pvcbase-worker64 pvcbase-worker65 pvcbase-worker66
pvcbase-worker67 pvcbase-worker68 pvcbase-worker69 pvcbase-worker70 pvcbase-worker71 pvcbase-worker72
pvcbase-worker73 pvcbase-worker74 pvcbase-worker75 pvcbase-worker76 pvcbase-worker77 pvcbase-worker78
pvcbase-worker79 pvcbase-worker80 pvcbase-worker81 pvcbase-worker82 pvcbase-worker83 pvcbase-worker84
pvcbase-worker85 pvcbase-worker86 pvcbase-worker87 pvcbase-worker88 pvcbase-worker89 pvcbase-worker90
pvcbase-worker91 pvcbase-worker92 pvcbase-worker93 pvcbase-worker94 pvcbase-worker95)

[root@ipaserver ~]# ips=(100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147
148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174
175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199)

[root@ipaserver ~]# for (( i=0; i<${#vms[@]}; i++ )); do ipa dnsrecord-add
cdp.rdu2.scalelab.redhat.com ${vms[i]} --a-rec 10.1.49.${ips[i]} --a-create-reverse;
done


[root@ipaserver ~]# ipa dnsrecord-find 49.1.10.in-addr.arpa.
ipa: WARNING: Search result has been truncated: Configured size limit exceeded
  Record name: @
  NS record: ipaserver.cdp.rdu2.scalelab.redhat.com.

  Record name: 1
  PTR record: ipaserver.cdp.rdu2.scalelab.redhat.com.

  Record name: 100
  PTR record: cldr-mngr.cdp.rdu2.scalelab.redhat.com.

  Record name: 101
  PTR record: cldr-utility.cdp.rdu2.scalelab.redhat.com.

  Record name: 102
  PTR record: pvcbase-master01.cdp.rdu2.scalelab.redhat.com.
```

```
Record name: 103
PTR record: pvcbase-master02.cdp.rdu2.scalelab.redhat.com.

Record name: 104
PTR record: pvcbase-master03.cdp.rdu2.scalelab.redhat.com.

Record name: 105
PTR record: pvcbase-worker01.cdp.rdu2.scalelab.redhat.com.

Record name: 106
PTR record: pvcbase-worker02.cdp.rdu2.scalelab.redhat.com.

Record name: 107
PTR record: pvcbase-worker03.cdp.rdu2.scalelab.redhat.com.

Record name: 108
PTR record: pvcbase-worker04.cdp.rdu2.scalelab.redhat.com.

Record name: 109
PTR record: pvcbase-worker05.cdp.rdu2.scalelab.redhat.com.

Record name: 110
PTR record: pvcbase-worker06.cdp.rdu2.scalelab.redhat.com.

Record name: 111
PTR record: pvcbase-worker07.cdp.rdu2.scalelab.redhat.com.

Record name: 112
PTR record: pvcbase-worker08.cdp.rdu2.scalelab.redhat.com.

Record name: 113
PTR record: pvcbase-worker09.cdp.rdu2.scalelab.redhat.com.

Record name: 114
PTR record: pvcbase-worker10.cdp.rdu2.scalelab.redhat.com.

Record name: 115
PTR record: pvcbase-worker11.cdp.rdu2.scalelab.redhat.com.

Record name: 116
PTR record: pvcbase-worker12.cdp.rdu2.scalelab.redhat.com.

Record name: 117
PTR record: pvcbase-worker13.cdp.rdu2.scalelab.redhat.com.

Record name: 118
PTR record: pvcbase-worker14.cdp.rdu2.scalelab.redhat.com.

Record name: 119
PTR record: pvcbase-worker15.cdp.rdu2.scalelab.redhat.com.

Record name: 120
PTR record: pvcbase-worker16.cdp.rdu2.scalelab.redhat.com.

Record name: 121
PTR record: pvcbase-worker17.cdp.rdu2.scalelab.redhat.com.

Record name: 122
PTR record: pvcbase-worker18.cdp.rdu2.scalelab.redhat.com.

Record name: 123
PTR record: pvcbase-worker19.cdp.rdu2.scalelab.redhat.com.

Record name: 124
PTR record: pvcbase-worker20.cdp.rdu2.scalelab.redhat.com.

Record name: 125
PTR record: pvcbase-worker21.cdp.rdu2.scalelab.redhat.com.

Record name: 126
PTR record: pvcbase-worker22.cdp.rdu2.scalelab.redhat.com.

Record name: 127
PTR record: pvcbase-worker23.cdp.rdu2.scalelab.redhat.com.

Record name: 128
PTR record: pvcbase-worker24.cdp.rdu2.scalelab.redhat.com.

Record name: 129
PTR record: pvcbase-worker25.cdp.rdu2.scalelab.redhat.com.

Record name: 130
PTR record: pvcbase-worker26.cdp.rdu2.scalelab.redhat.com.
```

```
Record name: 131
PTR record: pvcbase-worker27.cdp.rdu2.scalelab.redhat.com.

Record name: 132
PTR record: pvcbase-worker28.cdp.rdu2.scalelab.redhat.com.

Record name: 133
PTR record: pvcbase-worker29.cdp.rdu2.scalelab.redhat.com.

Record name: 134
PTR record: pvcbase-worker30.cdp.rdu2.scalelab.redhat.com.

Record name: 135
PTR record: pvcbase-worker31.cdp.rdu2.scalelab.redhat.com.

Record name: 136
PTR record: pvcbase-worker32.cdp.rdu2.scalelab.redhat.com.

Record name: 137
PTR record: pvcbase-worker33.cdp.rdu2.scalelab.redhat.com.

Record name: 138
PTR record: pvcbase-worker34.cdp.rdu2.scalelab.redhat.com.

Record name: 139
PTR record: pvcbase-worker35.cdp.rdu2.scalelab.redhat.com.

Record name: 140
PTR record: pvcbase-worker36.cdp.rdu2.scalelab.redhat.com.

Record name: 141
PTR record: pvcbase-worker37.cdp.rdu2.scalelab.redhat.com.

Record name: 142
PTR record: pvcbase-worker38.cdp.rdu2.scalelab.redhat.com.

Record name: 143
PTR record: pvcbase-worker39.cdp.rdu2.scalelab.redhat.com.

Record name: 144
PTR record: pvcbase-worker40.cdp.rdu2.scalelab.redhat.com.

Record name: 145
PTR record: pvcbase-worker41.cdp.rdu2.scalelab.redhat.com.

Record name: 146
PTR record: pvcbase-worker42.cdp.rdu2.scalelab.redhat.com.

Record name: 147
PTR record: pvcbase-worker43.cdp.rdu2.scalelab.redhat.com.

Record name: 148
PTR record: pvcbase-worker44.cdp.rdu2.scalelab.redhat.com.

Record name: 149
PTR record: pvcbase-worker45.cdp.rdu2.scalelab.redhat.com.

Record name: 150
PTR record: pvcbase-worker46.cdp.rdu2.scalelab.redhat.com.

Record name: 151
PTR record: pvcbase-worker47.cdp.rdu2.scalelab.redhat.com.

Record name: 152
PTR record: pvcbase-worker48.cdp.rdu2.scalelab.redhat.com.

Record name: 153
PTR record: pvcbase-worker49.cdp.rdu2.scalelab.redhat.com.

Record name: 154
PTR record: pvcbase-worker50.cdp.rdu2.scalelab.redhat.com.

Record name: 155
PTR record: pvcbase-worker51.cdp.rdu2.scalelab.redhat.com.

Record name: 156
PTR record: pvcbase-worker52.cdp.rdu2.scalelab.redhat.com.

Record name: 157
PTR record: pvcbase-worker53.cdp.rdu2.scalelab.redhat.com.

Record name: 158
PTR record: pvcbase-worker54.cdp.rdu2.scalelab.redhat.com.
```

```
Record name: 159
PTR record: pvcbase-worker55.cdp.rdu2.scalelab.redhat.com.

Record name: 160
PTR record: pvcbase-worker56.cdp.rdu2.scalelab.redhat.com.

Record name: 161
PTR record: pvcbase-worker57.cdp.rdu2.scalelab.redhat.com.

Record name: 162
PTR record: pvcbase-worker58.cdp.rdu2.scalelab.redhat.com.

Record name: 163
PTR record: pvcbase-worker59.cdp.rdu2.scalelab.redhat.com.

Record name: 164
PTR record: pvcbase-worker60.cdp.rdu2.scalelab.redhat.com.

Record name: 165
PTR record: pvcbase-worker61.cdp.rdu2.scalelab.redhat.com.

Record name: 166
PTR record: pvcbase-worker62.cdp.rdu2.scalelab.redhat.com.

Record name: 167
PTR record: pvcbase-worker63.cdp.rdu2.scalelab.redhat.com.

Record name: 168
PTR record: pvcbase-worker64.cdp.rdu2.scalelab.redhat.com.

Record name: 169
PTR record: pvcbase-worker65.cdp.rdu2.scalelab.redhat.com.

Record name: 170
PTR record: pvcbase-worker66.cdp.rdu2.scalelab.redhat.com.

Record name: 171
PTR record: pvcbase-worker67.cdp.rdu2.scalelab.redhat.com.

Record name: 172
PTR record: pvcbase-worker68.cdp.rdu2.scalelab.redhat.com.

Record name: 173
PTR record: pvcbase-worker69.cdp.rdu2.scalelab.redhat.com.

Record name: 174
PTR record: pvcbase-worker70.cdp.rdu2.scalelab.redhat.com.

Record name: 175
PTR record: pvcbase-worker71.cdp.rdu2.scalelab.redhat.com.

Record name: 176
PTR record: pvcbase-worker72.cdp.rdu2.scalelab.redhat.com.

Record name: 177
PTR record: pvcbase-worker73.cdp.rdu2.scalelab.redhat.com.

Record name: 178
PTR record: pvcbase-worker74.cdp.rdu2.scalelab.redhat.com.

Record name: 179
PTR record: pvcbase-worker75.cdp.rdu2.scalelab.redhat.com.

Record name: 180
PTR record: pvcbase-worker76.cdp.rdu2.scalelab.redhat.com.

Record name: 181
PTR record: pvcbase-worker77.cdp.rdu2.scalelab.redhat.com.

Record name: 182
PTR record: pvcbase-worker78.cdp.rdu2.scalelab.redhat.com.

Record name: 183
PTR record: pvcbase-worker79.cdp.rdu2.scalelab.redhat.com.

Record name: 184
PTR record: pvcbase-worker80.cdp.rdu2.scalelab.redhat.com.

Record name: 185
PTR record: pvcbase-worker81.cdp.rdu2.scalelab.redhat.com.

Record name: 186
PTR record: pvcbase-worker82.cdp.rdu2.scalelab.redhat.com.
```

```
  Record name: 187
  PTR record: pvcbase-worker83.cdp.rdu2.scalelab.redhat.com.

  Record name: 188
  PTR record: pvcbase-worker84.cdp.rdu2.scalelab.redhat.com.

  Record name: 189
  PTR record: pvcbase-worker85.cdp.rdu2.scalelab.redhat.com.

  Record name: 190
  PTR record: pvcbase-worker86.cdp.rdu2.scalelab.redhat.com.

  Record name: 191
  PTR record: pvcbase-worker87.cdp.rdu2.scalelab.redhat.com.

  Record name: 192
  PTR record: pvcbase-worker88.cdp.rdu2.scalelab.redhat.com.

  Record name: 193
  PTR record: pvcbase-worker89.cdp.rdu2.scalelab.redhat.com.

  Record name: 194
  PTR record: pvcbase-worker90.cdp.rdu2.scalelab.redhat.com.

  Record name: 195
  PTR record: pvcbase-worker91.cdp.rdu2.scalelab.redhat.com.

  Record name: 196
  PTR record: pvcbase-worker92.cdp.rdu2.scalelab.redhat.com.

  Record name: 197
  PTR record: pvcbase-worker93.cdp.rdu2.scalelab.redhat.com.
----------------------------
Number of entries returned 100
----------------------------
```

ADD OPENSHIFT NODES, API & WILDCARD DNS:

Create DNS records for the OpenShift management cluster:

```
ipa dnsrecord-add vlan601.rdu2.scalelab.redhat.com api --a-rec 10.1.48.3 --a-create-reverse
 Record name: api
  A record: 10.1.48.3

ipa dnsrecord-add vlan601.rdu2.scalelab.redhat.com *.apps --a-rec 10.1.48.4 --a-create-reverse
 Record name: *.apps
  A record: 10.1.48.4

nslookup api.vlan601.rdu2.scalelab.redhat.com
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   api.vlan601.rdu2.scalelab.redhat.com
Address: 10.1.48.3

nslookup test.apps.vlan601.rdu2.scalelab.redhat.com
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   test.apps.vlan601.rdu2.scalelab.redhat.com
Address: 10.1.48.4
```

ADD OPENSHIFT NODES, API & WILDCARD DNS (for CDP Data Services running on VMs):

Create DNS records for the OpenShift guest cluster:

```
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com api --a-rec 10.1.49.3 --a-create-reverse
 Record name: api
  A record: 10.1.49.3

ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com *.apps --a-rec 10.1.49.4 --a-create-reverse
```

```
 Record name: *.apps
  A record: 10.1.49.4


nslookup api.cdp.rdu2.scalelab.redhat.com
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   api.cdp.rdu2.scalelab.redhat.com
Address: 10.1.49.3

nslookup test.apps.cdp.rdu2.scalelab.redhat.com
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   test.apps.cdp.rdu2.scalelab.redhat.com
Address: 10.1.49.

Add OpenShift Nodes:
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-master01 --a-rec 10.1.49.200 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-master02 --a-rec 10.1.49.201 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-master03 --a-rec 10.1.49.202 --a-create-reverse

ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker01 --a-rec 10.1.49.203 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker02 --a-rec 10.1.49.204 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker03 --a-rec 10.1.49.205 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker04 --a-rec 10.1.49.206 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker05 --a-rec 10.1.49.207 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker06 --a-rec 10.1.49.208 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker07 --a-rec 10.1.49.209 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker08 --a-rec 10.1.49.210 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker09 --a-rec 10.1.49.211 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-worker10 --a-rec 10.1.49.212 --a-create-reverse

ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-infra01 --a-rec 10.1.49.213 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-infra02 --a-rec 10.1.49.214 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com pvcocp-infra03 --a-rec 10.1.49.215 --a-create-reverse


ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com api.pvcocp --a-rec 10.1.49.5 --a-create-reverse
 Record name: api
  A record: 10.1.49.5

ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com *.apps.pvcocp --a-rec 10.1.49.6 --a-create-reverse
 Record name: *.apps
  A record: 10.1.49.6

ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com f36-h17-000-r640 --a-rec 10.1.49.216 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com f36-h18-000-r640 --a-rec 10.1.49.217 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com f36-h19-000-r640 --a-rec 10.1.49.218 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com d29-h11-000-r750 --a-rec 10.1.49.219 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com d29-h13-000-r750 --a-rec 10.1.49.220 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com d30-h05-000-r750 --a-rec 10.1.49.221 --a-create-reverse
ipa dnsrecord-add cdp.rdu2.scalelab.redhat.com d30-h07-000-r750 --a-rec 10.1.49.222 --a-create-reverse
```

Step 4. Verify the DNS records have been added successfully:

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

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

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

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

  Record name: 185.16
  PTR record: pvcbase-worker1.redhat.local.
```

```
  Record name: 0.23
  PTR record: pvcbase-worker2.redhat.local.

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

  Record name: 239.30
  PTR record: pvcocp-master.redhat.local.

  Record name: 43.22
  PTR record: pvcocp-worker1.redhat.local.

  Record name: 249.30
  PTR record: pvcocp-worker2.redhat.local.

  Record name: 198.24
  PTR record: pvcocp-worker3.redhat.local.

  Record name: 53.24
  PTR record: pvcocp-worker4.redhat.local.

  Record name: 24.26
  PTR record: pvcocp-worker5.redhat.local.

  Record name: 43.22
  PTR record: pvcocp-worker6.redhat.local.

  Record name: 0.23
  PTR record: pvcocp-worker7.redhat.local.

  Record name: 198.24
  PTR record: pvcocp-worker8.redhat.local.

  Record name: 53.24
  PTR record: pvcocp-worker9.redhat.local.

  Record name: 24.26
  PTR record: pvcocp-worker10.redhat.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> (***redhat123***)

```
[root@pvcbase-master ~]# ipa-client-install --force-join
This program will set up IPA client.
Version 4.12.2

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.redhat.local
Realm: REDHAT.LOCAL
DNS Domain: redhat.local
IPA Server: ipaserver.redhat.local
BaseDN: dc=redhat,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@REDHAT.LOCAL: <redhat123>
Successfully retrieved CA cert
    Subject:     CN=Certificate Authority,O=REDHAT.LOCAL
    Issuer:      CN=Certificate Authority,O=REDHAT.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 REDHAT.LOCAL
Created /etc/ipa/default.conf
Configured /etc/sssd/sssd.conf
Systemwide CA database updated.
Hostname (pvcbase-master.redhat.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 redhat.local as NIS domain.
Configured /etc/krb5.conf for IPA realm REDHAT.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@REDHAT.LOCAL: <redhat123>

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

Valid starting       Expires              Service principal
05/19/2025 14:50:23  05/20/2025 14:38:23  HTTP/ipaserver.redhat.local@REDHAT.LOCAL
        Etype (skey, tkt): aes256-cts-hmac-sha384-192, aes256-cts-hmac-sha384-192
05/19/2025 14:49:13  05/20/2025 14:38:23  krbtgt/REDHAT.LOCAL@REDHAT.LOCAL
        Etype (skey, tkt): aes256-cts-hmac-sha384-192, aes256-cts-hmac-sha384-192

##### try kinit admin@REDHAT.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 OCP 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-openshift-cluster-environments.html

```
[root@ipaserver ~]# cat /etc/resolv.conf
search cdpbase.svc.cluster.local svc.cluster.local cluster.local us-west-2.compute.internal
my-vms.svc.cluster.local redhat.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=redhat.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.redhat.local
Server:         172.31.24.240
Address:        172.31.24.240#53

Name:   cldr-mngr.redhat.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.redhat.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.redhat.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.redhat.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 OCP data service cluster (if not set properly, chances of OCP installation getting corrupt):

```
[root@ipaserver ~]#  ipa dnsrecord-add redhat.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 ocp-master node
  Record name: *.apps
  A record: 172.31.30.239

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

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

[root@ipaserver ~]# dig console-cdp.apps.redhat.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.redhat.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     pvcocp-master.redhat.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 (Ignore the errors if firewall is not installed)

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

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 ~]#
```

```
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# 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
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag
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
```

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:
# pvcocp-worker4.redhat.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"

##### Reboot the namenodes & datanodes machines:
[root@ipaserver ~]# ansible namenodes,datanodes -m shell -a "sudo reboot"
```

Procedure 17. 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.redhat.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.redhat.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
```

****

Sonatype Nexus3 Repository Manager Setup

This article describes the steps to deploy the external Docker registry in the designated Nexus server.

(Completely Optional) -- Required only, when you want to perform the setup with ***External Docker Repository*** (which requires additional steps). We will proceed for the current setup with the ***Embedded Docker Repository.***

Procedure 1. Setup Nexus3

Step 1. From a host connected to the Internet, download and configure the Nexus3 Tar as shown below. We will directly login to ***cldr-mngr*** and perform below steps:

```
[root@ipaserver ~]# ssh root@cldr-mngr
[root@cldr-mngr ~]# cd /opt
[root@cldr-mngr opt]# sudo wget https://download.sonatype.com/nexus/3/nexus-3.70.1-02-java11-unix.tar.gz
[root@cldr-mngr opt]# tar xvf nexus-3.70.1-02-java11-unix.tar.gz
[root@cldr-mngr opt]# sudo mv -v nexus-3.70.1-02 nexus
[root@cldr-mngr opt]# sudo adduser -m nexus
[root@cldr-mngr opt]# sudo chown -R nexus:nexus /opt/nexus
[root@cldr-mngr opt]# sudo chown -R nexus:nexus /opt/sonatype-work
[root@cldr-mngr opt]# rm -vf nexus-3.70.1-02-java11-unix.tar.gz
```

Step 2. Install Docker: **https://docs.docker.com/engine/install/rhel/**

```
[root@cldr-mngr opt]# sudo yum install -y yum-utils
[root@cldr-mngr opt]# sudo yum-config-manager --add-repo https://download.docker.com/linux/rhel/docker-ce.repo
[root@cldr-mngr opt]# sudo yum install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

Step 3. Update Configuration Files:

```
[root@cldr-mngr opt]# sudo cat> /opt/nexus/bin/nexus.rc
run_as_user="nexus"
[root@cldr-mngr opt]# sudo vi /opt/nexus/bin/nexus.vmoptions
-Xms512m
-Xmx512m
-XX:MaxDirectMemorySize=512m
[root@cldr-mngr opt]#
```

Step 4. Create nexus service.

```
[root@cldr-mngr opt]# cat>/etc/systemd/system/nexus.service
[Unit]
Description=Nexus Repository Manager
After=network.target
[Service]
Type=forking
LimitNOFILE=65536
User=nexus
Group=nexus
ExecStart=/opt/nexus/bin/nexus start
ExecStop=/opt/nexus/bin/nexus stop
User=nexus
Restart=on-abort
[Install]
WantedBy=multi-user.target
[root@cldr-mngr opt]#
```

Step 5. Enable and start Nexus:

```
[root@cldr-mngr opt]# sudo ln -s /opt/nexus/bin/nexus /etc/init.d/nexus
[root@cldr-mngr opt]# sudo chkconfig --add nexus
[root@cldr-mngr opt]# sudo chkconfig --levels 345 nexus on

[root@cldr-mngr opt]# sudo service nexus start
```

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

Step 6. Create the SSL certificates on the server. The following nexus.crt certificate will be used during CDP PvC Data Services installation.

```
[root@cldr-mngr opt]# cd /opt/nexus

[root@cldr-mngr nexus]# keytool -genkeypair -keystore keystore.jks -storepass password -keypass password
-alias jetty -keyalg RSA -keysize 2048 -validity 5000 -dname "CN=nexus.redhat.local, OU=CLDR, O=Red Hat
Inc, L=Raleigh, ST=North Carolina, C=US" -ext "SAN=DNS:nexus.redhat.local,IP:$(hostname -i)" -ext
"BC=ca:true"

[root@cldr-mngr nexus]# keytool -exportcert -keystore keystore.jks -storepass password -alias jetty -rfc >
nexus.crt

##### Copy the JKS file to /opt/nexus/etc/ssl/ directory.
[root@cldr-mngr nexus]# cp keystore.jks /opt/nexus/etc/ssl/
[root@cldr-mngr nexus]# cp nexus.crt etc/jetty/
[root@cldr-mngr nexus]# cp nexus.crt etc/ssl/

##### Check that the hostname is correctly defined in the certificate.
[root@cldr-mngr nexus]# openssl x509 -noout -text -in nexus.crt | grep -A1 X509v3
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                96:5F:A2:EB:CF:E4:B2:00:06:75:F9:FD:11:5F:A5:2A:20:55:F3:59
            X509v3 Subject Alternative Name:
                DNS:nexus.redhat.local
[root@cldr-mngr nexus]#
##### Download the nexus.crt file (or copy) to your local Laptop machine's ~/ocp directory, it will be
required later during the data service installation.
```

Step 7. Configure the /opt/sonatype/sonatype-work/nexus3/etc/nexus.properties file.

```
[root@cldr-mngr nexus]# cat> /opt/nexus/etc/nexus-default.properties
## DO NOT EDIT - CUSTOMIZATIONS BELONG IN $data-dir/etc/nexus.properties
##
# Jetty section
application-port=8081
application-host=0.0.0.0
#nexus-args=${jetty.etc}/jetty.xml,${jetty.etc}/jetty-http.xml,${jetty.etc}/jetty-requestlog.xml
nexus-args=${jetty.etc}/jetty.xml,${jetty.etc}/jetty-http.xml,${jetty.etc}/jetty-https.xml,${jetty.etc}/je
tty-requestlog.xml

nexus-context-path=/
#Nexus section
nexus-edition=nexus-pro-edition
nexus-features=\
  nexus-pro-feature
nexus.hazelcast.discovery.isEnabled=true

application-port-ssl=8443
ssl.etc=etc/ssl
[root@cldr-mngr nexus]#
```

Step 8. Verify and configure the /opt/nexus/etc/jetty/jetty-https.xml file as shown as follows.

```
[root@cldr-mngr nexus]# vi etc/jetty/jetty-https.xml
<New id="sslContextFactory" class="org.eclipse.jetty.util.ssl.SslContextFactory$Server">
     <Set name="certAlias">jetty</Set>
     <Set name="KeyStorePath"><Property name="ssl.etc"/>/keystore.jks</Set>
     <Set name="KeyStorePassword">password</Set>
     <Set name="KeyManagerPassword">password</Set>
     <Set name="TrustStorePath"><Property name="ssl.etc"/>/keystore.jks</Set>
     <Set name="TrustStorePassword">password</Set>
     <Set name="EndpointIdentificationAlgorithm"></Set>
     <Set name="NeedClientAuth"><Property name="jetty.ssl.needClientAuth" default="false"/></Set>
     <Set name="WantClientAuth"><Property name="jetty.ssl.wantClientAuth" default="false"/></Set>
     <Set name="IncludeProtocols">
         <Array type="java.lang.String">
```

```
            <Item>TLSv1.2</Item>
        </Array>
    </Set>
 </New>
[root@cldr-mngr nexus]#
```

Step 9. Restart the Nexus service to get the SSL related changes in effect:

```
##### Restart the Nexus service.
[root@cldr-mngr opt]# sudo service nexus restart
[root@cldr-mngr opt]# tail -f /opt/sonatype-work/nexus3/log/nexus.log

[root@cldr-mngr opt]# sudo service nexus status

##### Note: Make sure below ports are open:
9998 9999 8081 8443

##### Note: Error Handling
The class file com.install4j.runtime.launcher.UnixLauncher is contained in the jar file
'[nexus-installation].install4j\i4jruntime.jar'.
The folder starts with a dot '.' - this means the folder is hidden.
Does the jar file exist in your filesystem and are the permissions correct?

'chown' with '-R' should honor the hidden folder '.install4j' but I am not quite sure about 'mv' (sudo mv
-v nexus*/* /opt/nexus).
```

~~Step 10.~~ ~~Create OCP Service to access the Nexus WebUI~~

```
ksahu@Kuldeeps-MacBook-Air ocp % virtctl expose vm clouderamanager --name nexus-webtlssvc --type LoadBalancer
--port 8443
Service nexus-webtlssvc successfully exposed for vm clouderamanager

##### (Optional) Create service to expose Docker Registry Web UI endpoint:
ksahu@Kuldeeps-MacBook-Air ocp % virtctl expose vm clouderamanager --name nexusdocker-webtlssvc --type
LoadBalancer --port 9999
Service nexusdocker-webtlssvc successfully exposed for vm clouderamanager
```

~~Step 11.~~ ~~List OCP Service Web URL to access the Nexus WebUI over TLS/SSL:~~

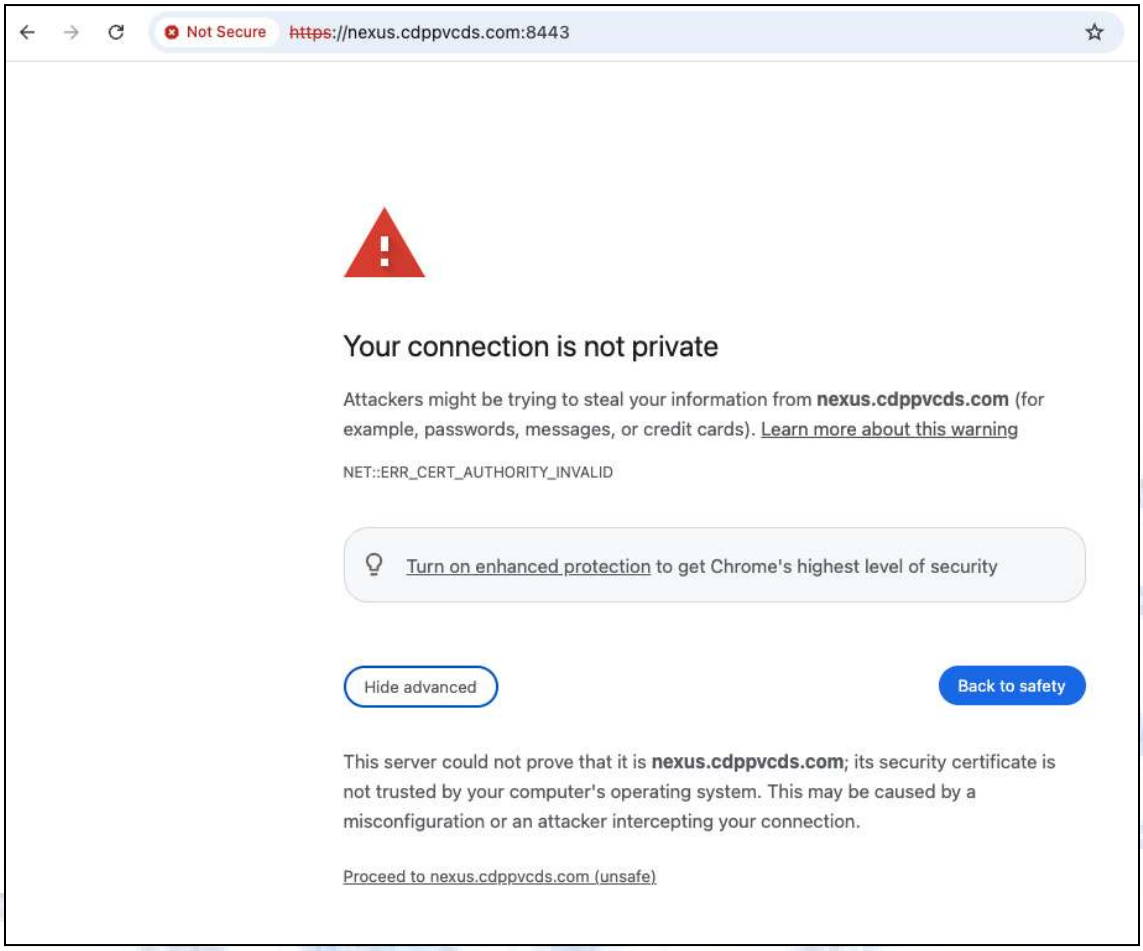
```
##### Take the password for admin user from below location to reset the admin password
ksahu@Kuldeeps-MacBook-Air ocp % oc get svc | grep nexus
nexus-webtlssvc                 LoadBalancer   172.30.216.54
a776572efd7764ac18429e4e52ec854f-2001921907.us-west-2.elb.amazonaws.com   8443:31992/TCP   4s

ksahu@Kuldeeps-MacBook-Air ocp % oc get svc | grep docker
nexusdocker-webtlssvc           LoadBalancer   172.30.101.135
a2426b40b0bc44d30b878573d5bc253c-367387264.us-west-2.elb.amazonaws.com   9999:31071/TCP   6d23h
```

Step 12. Update /etc/hosts and Login to WebUI https://<Nexus_IP_Addr_Or_LB_URL>:8443/

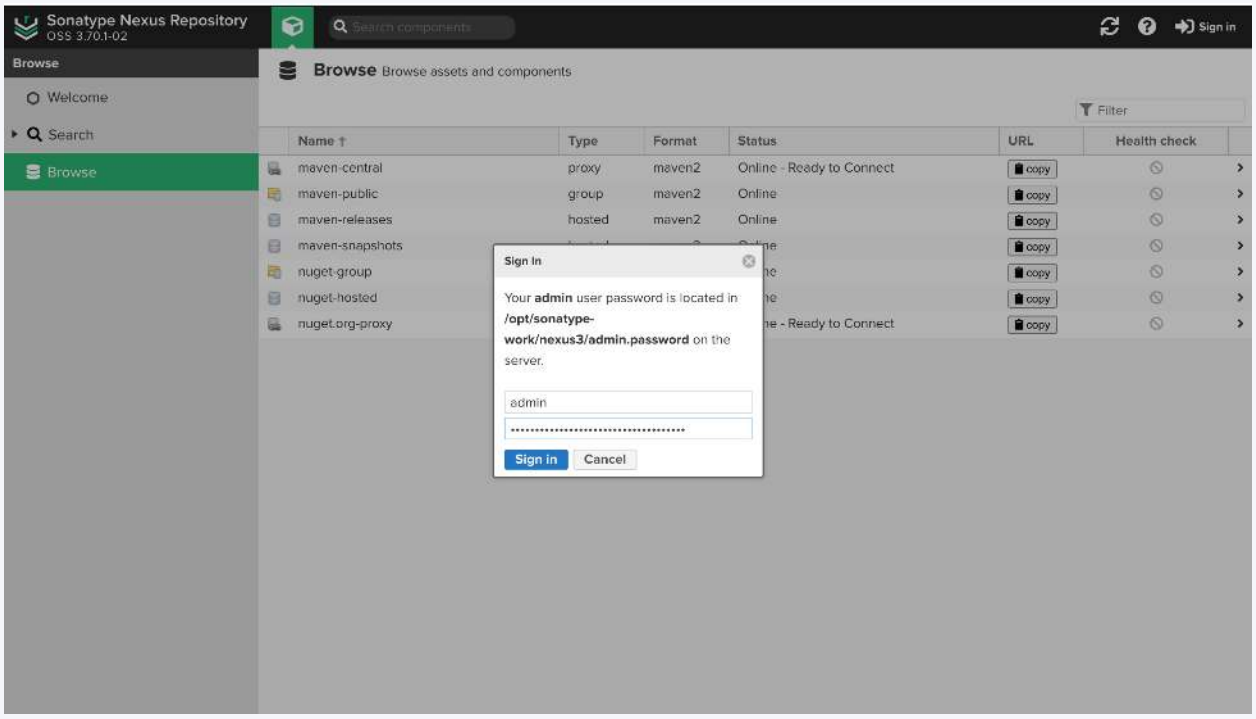
```
##### Take the password for admin user from below location to reset the admin password
[root@cldr-mngr nexus]# cat /opt/sonatype-work/nexus3/admin.password
[root@cldr-mngr nexus]# tail -f /opt/sonatype-work/nexus3/log/nexus.log
```

https://nexus.redhat.local:8443/
~~https://a8a54ef4bb25c49e992c3d68836ac04e-1031794758.us-west-2.elb.amazonaws.com:8443/~~

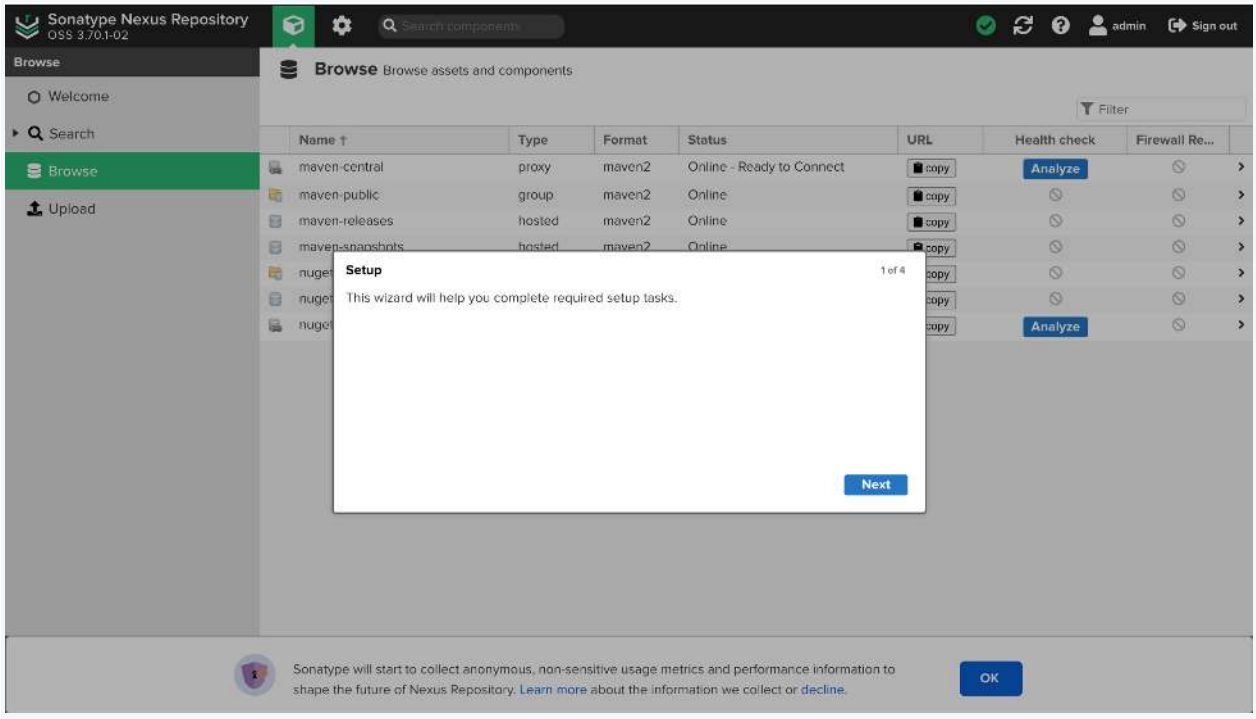
Step 13. You will see the below screen on the browser, when opening for the first time. Just click on the ***Advanced*** button and ***Proceed to the <URL> (Unsafe)***, as shown below.

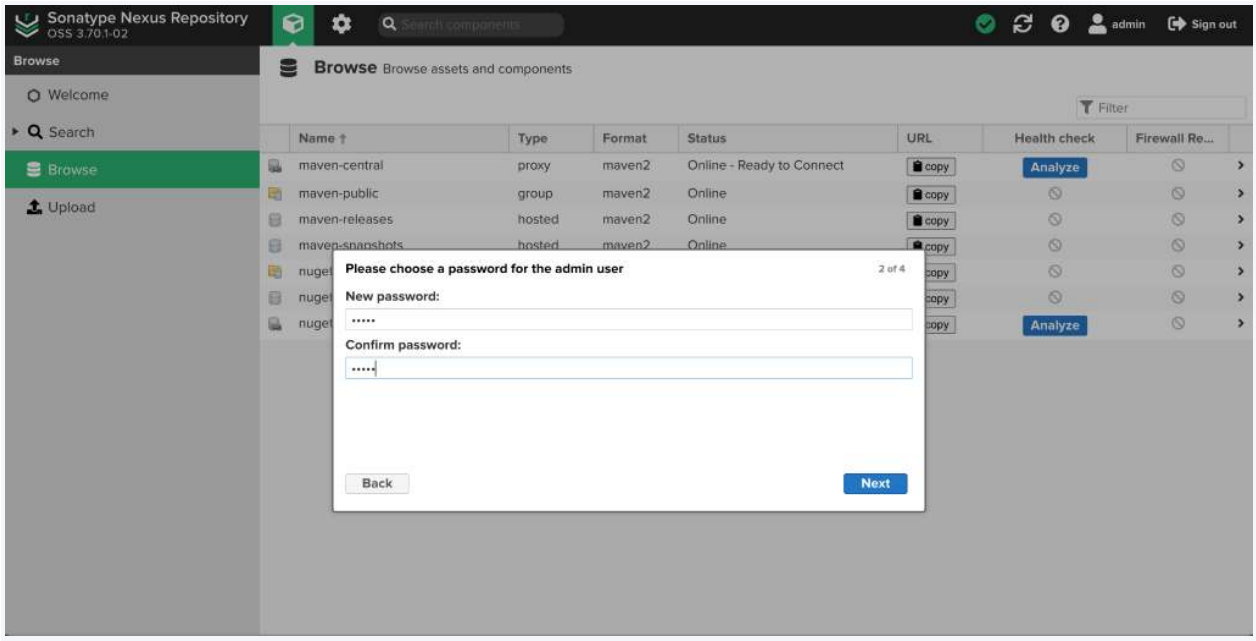


Step 14. Once the page is loaded, click on ***SignIn***. you will be asked to provide the ***username*** (admin) and ***password*** (Obtained from the password file from installation direction in the previous steps).

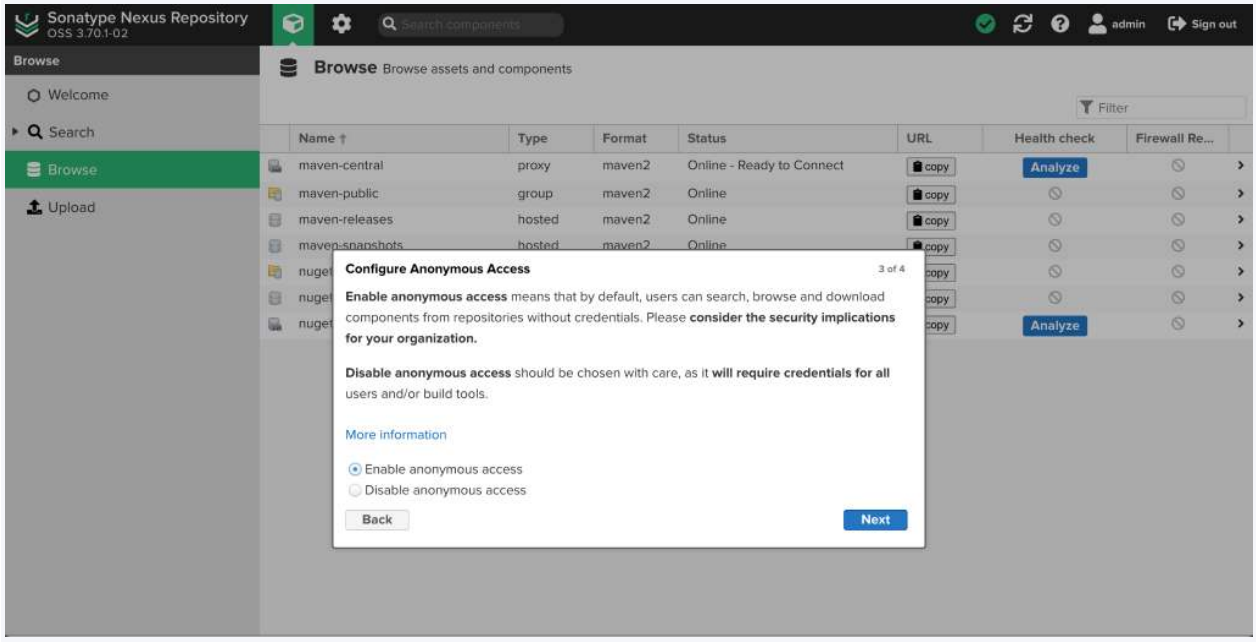


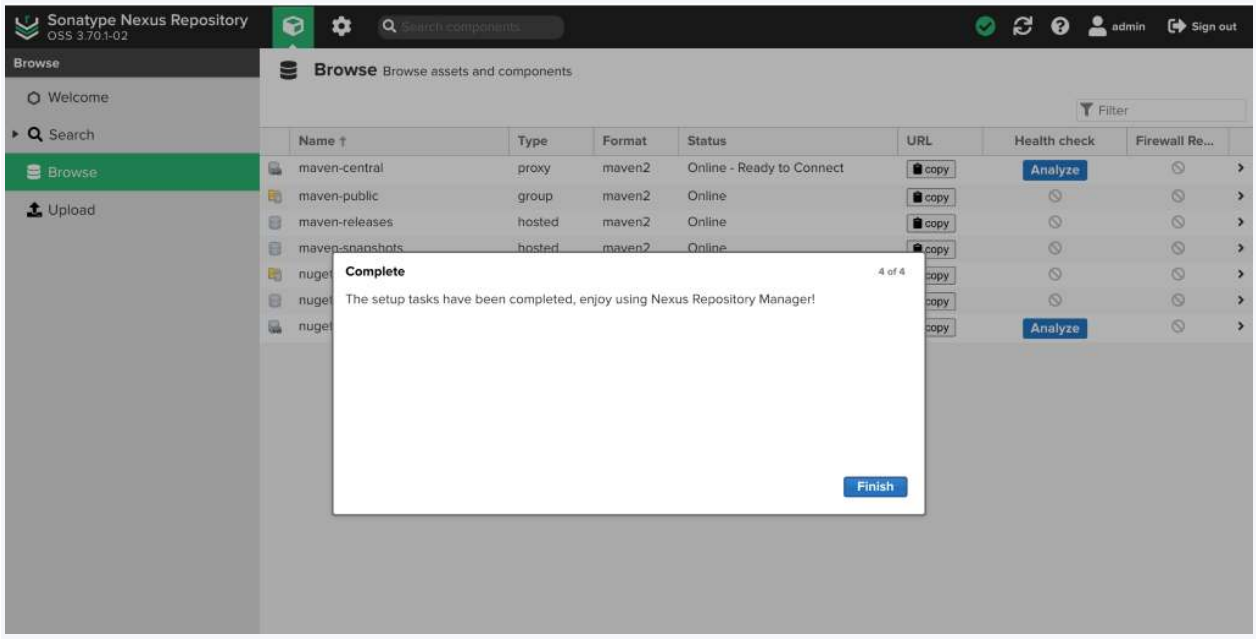
Step 15. Now, you have been asked to reset the initial admin password.



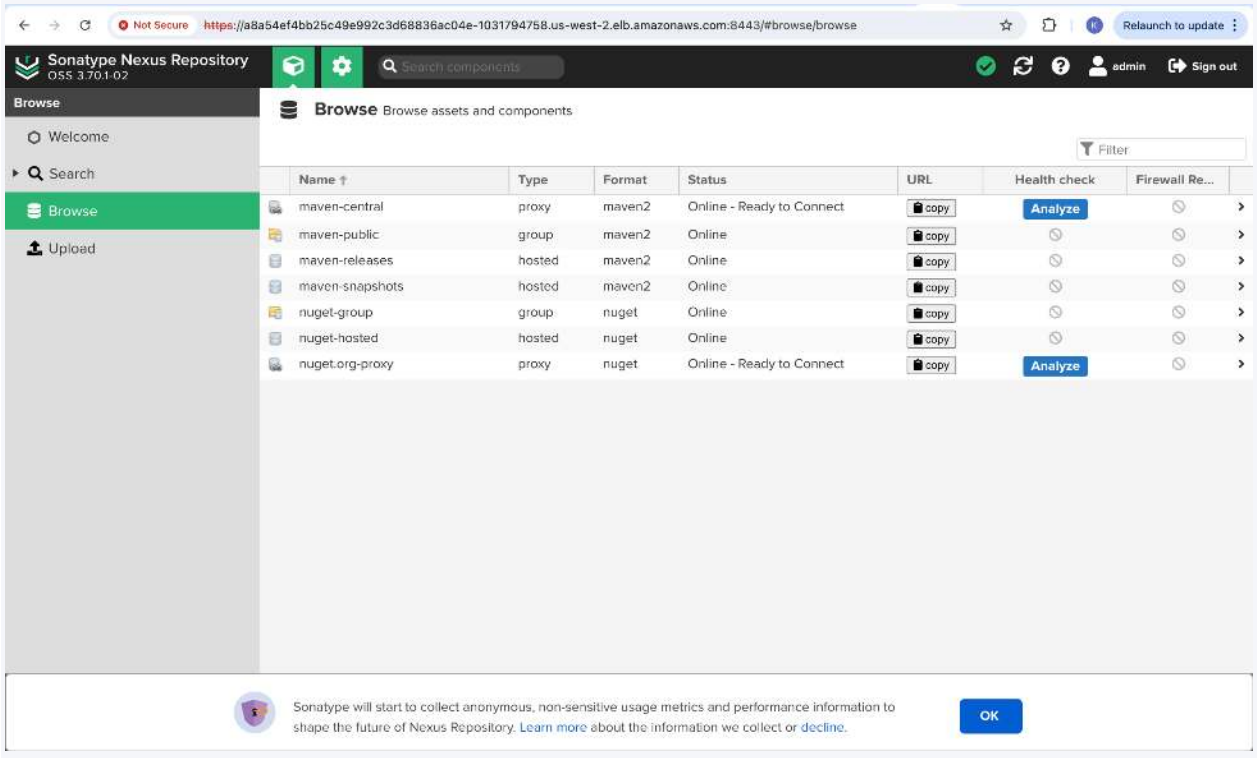


Step 16. Enable anonymous (READ) access.

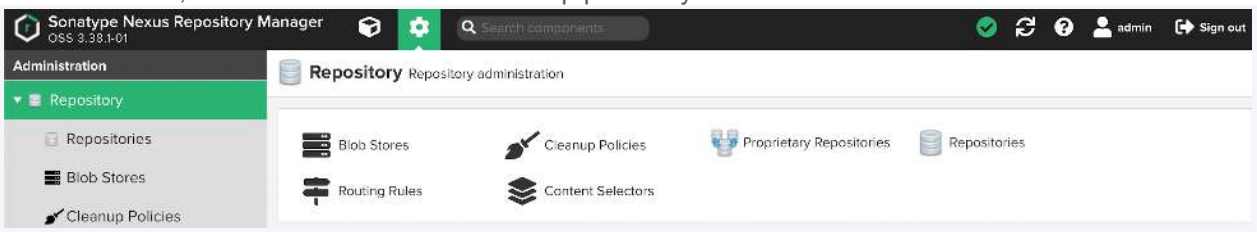




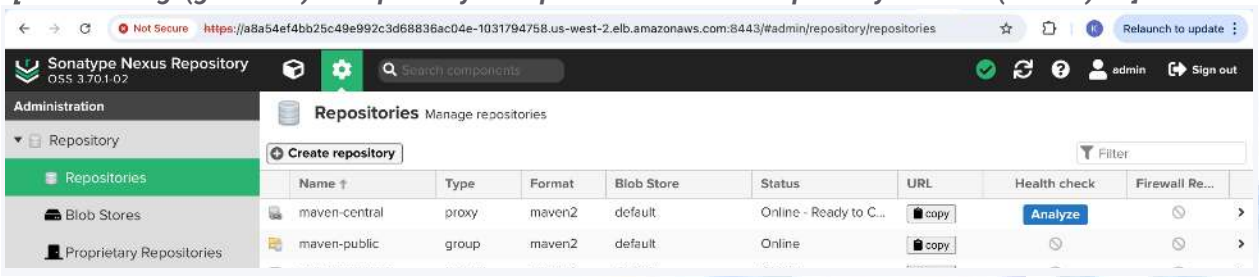
Step 17. Once you click on ***Finish*** as above, the Nexus Manager Home screen appears.



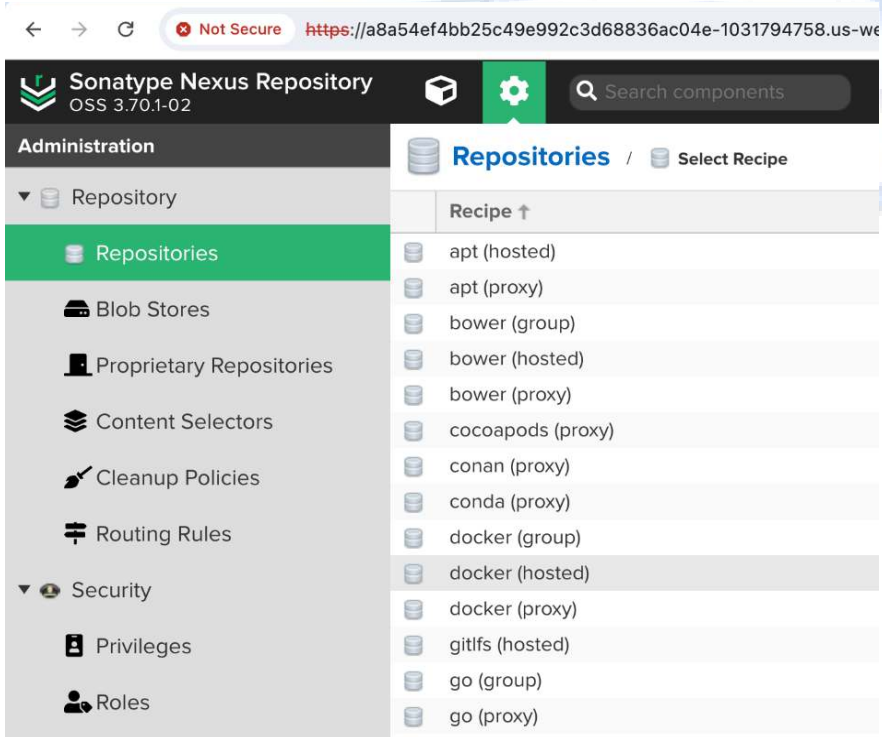
Step 18. Our Nexus is TLS Enabled (HTTPS). Now we will enable the Privately Hosted Docker Repository on Nexus. For this, click on the Gear icon on the top pane of your home screen.



Step 19. Now, click on ***Repository*** Option in the Left Pane and click on ***Create Repository*** button in the middle. ***[Go to Settings(gear icon) -> Repository -> Repositories ->Create repository->docker (hosted) ->.]***

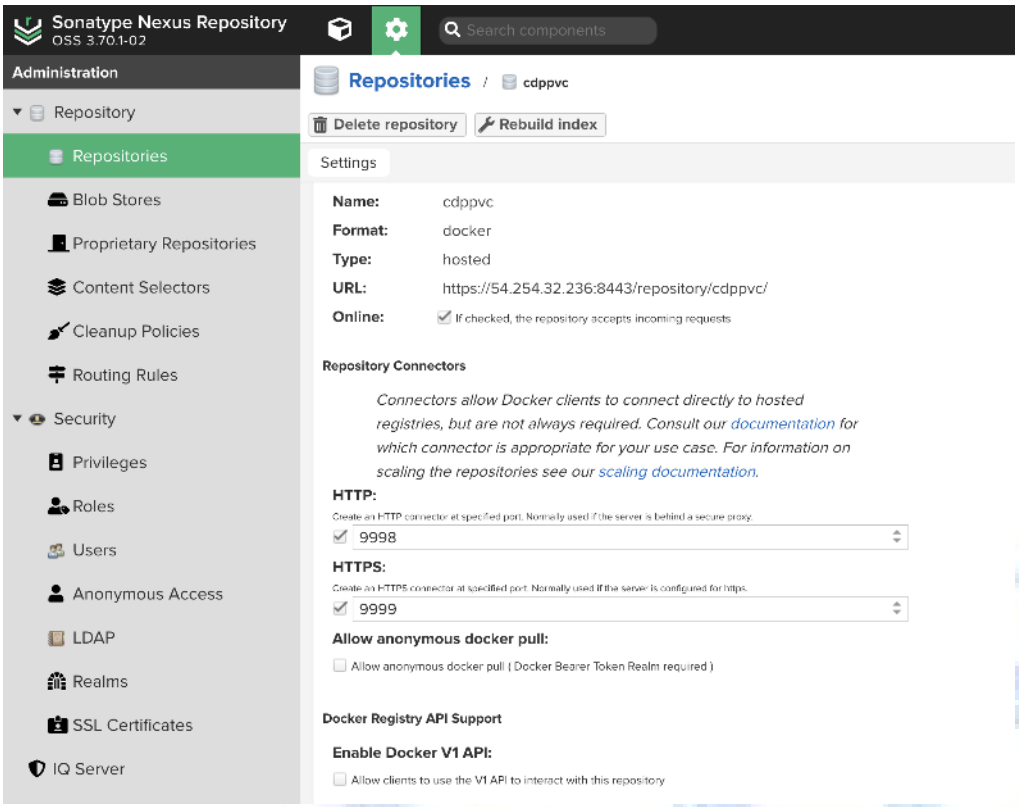


Step 20. Once you click on the Click Repository button, the page below will appear, choose docker(hosted) as the repository type.

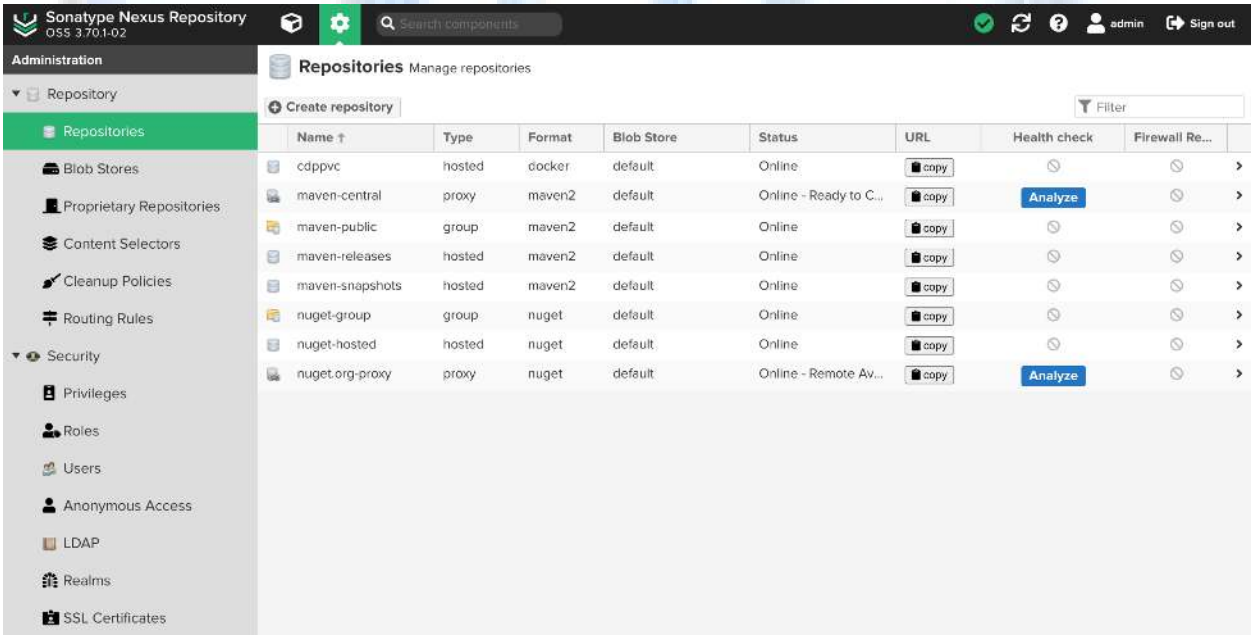


Setup the Docker repository with SSL port. In this setup, we will create the SSL-enabled docker repository with URL `https://nexus.redhat.local:9999/cdppvc`. Follow below steps to create Docker Hosted (private) repo on Nexus.

Step 21. Provide the name of Docker Repo, ports for ***HTTPS (9999)*** and ***HTTP (9998)*** access.



Step 22. Click on ***Create Repository***. Below page will come up with the repository created with the name you provided in the previous step.



Step 23. Ensure that the port SSL port 9999 is up and running, on the host, after Docker Repository is enabled.

```
[root@cldr-mngr opt]# netstat -an | grep 999
tcp        0      0 0.0.0.0:9999            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:9998            0.0.0.0:*               LISTEN
```

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

Step 24. Verify the access to Docker Repository using curl command.

```
[root@cldr-mngr opt]# curl -u admin:admin "https://nexus.redhat.local:9999/v2/_catalog" | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
curl: (60) SSL certificate problem: self-signed certificate
More details here: https://curl.se/docs/sslcerts.html

curl failed to verify the legitimacy of the server and therefore could not
establish a secure connection to it. To learn more about this situation and
how to fix it, please visit the web page mentioned above.
[root@cldr-mngr opt]#

##### curl will not work. to fix this, update the CA cert (nexus.crt) in the truststore of the server.

[root@cldr-mngr nexus]# sudo cp /opt/nexus/etc/ssl/nexus.crt /etc/pki/ca-trust/source/anchors/nexus.crt
[root@cldr-mngr nexus]# sudo update-ca-trust

##### Check that the CA cert has successfully been imported into the truststore of the server.

[root@cldr-mngr opt]# #openssl crl2pkcs7 -nocrl -certfile /etc/pki/tls/certs/ca-bundle.crt | openssl pkcs7
-print_certs | grep subject | grep nexus

##### You may now curl the SSL-enabled Docker URL successfully.
[root@cldr-mngr opt]# curl -u admin:admin "https://nexus.redhat.local:9999/v2/_catalog" -k | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    19  100    19    0     0    174      0 --:--:-- --:--:-- --:--:--   175
{
  "repositories": []
}
[root@cldr-mngr opt]# openssl s_client -connect nexus.redhat.local:9999
CONNECTED(00000003)
depth=0 C = US, ST = North Carolina, L = Raleigh, O = Red Hat Inc, OU = CLDR, CN = nexus.redhat.local
[root@cldr-mngr opt]#
```

Step 25. Curl command will work, but if you do docker pull, will give the error for unauthorized access, to fix this follow below steps:

```
##### Edit or create the file /etc/docker/daemon.json and add insecure-registries:
Make sure docker-ce is installed (not podman docker), if not, uninstall podman using dnf and reinstall
using official documentation https://docs.docker.com/engine/install/rhel/

[root@cldr-mngr opt]# vi /etc/docker/daemon.json
{
    "insecure-registries" : ["nexus.redhat.local:9999","192.168.1.38:9999"]
}

##### Restart docker daemon
[root@cldr-mngr opt]# systemctl restart docker
[root@cldr-mngr opt]# cd nexus
```

Step 26. Login to Private Nexus Docker Repository.

```
[root@cldr-mngr nexus]# curl -v --cacert nexus.crt https://nexus.redhat.local:9999/v2/_catalog
*   Trying 192.168.1.38:9999...
* Connected to nexus.redhat.local (192.168.1.38) port 9999 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
*  CAfile: nexus.crt
* TLSv1.0 (OUT), TLS header, Certificate Status (22):
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN), TLS header, Certificate Status (22):

##### Perform a test to ensure that you can login and pull the image from the Cloudera repository.
[root@cldr-mngr nexus]# docker login nexus.redhat.local:9999 -u admin -p admin
```

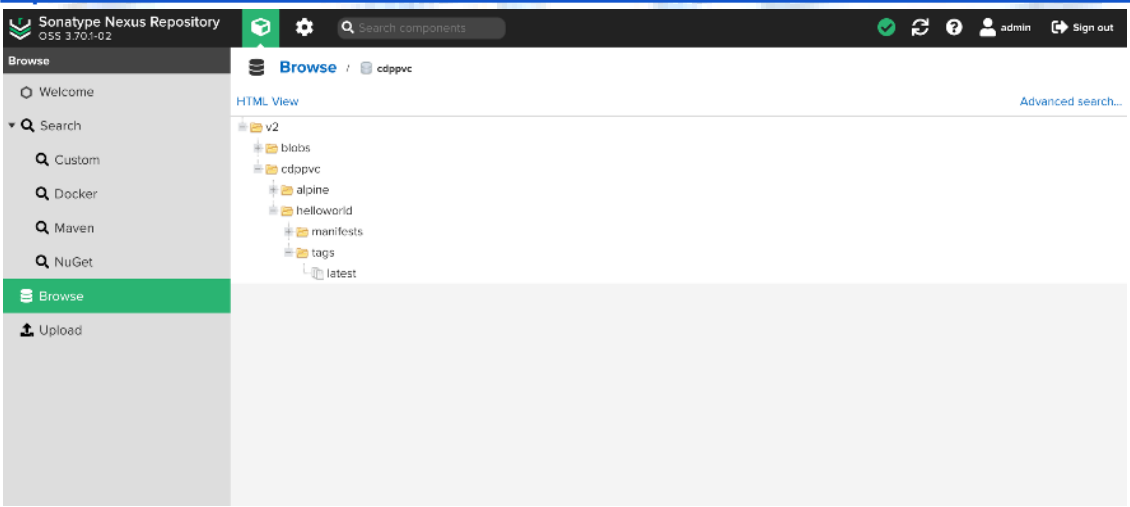
```
#docker login https://nexus.redhat.local:9999/cdppvc --username admin --password admin

WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
[root@cldr-mngr nexus]#
```

Step 27. Test Private Docker Repository is Working Fine:

```
[root@cldr-mngr nexus]# docker image ls
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE

[root@cldr-mngr nexus]# docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:1408fec50309afee38f3535383f5b09419e6dc0925bc69891e79d84cc4cdcec6
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest

[root@cldr-mngr nexus]# docker tag hello-world:latest nexus.redhat.local:9999/cdppvc/helloworld:latest

[root@cldr-mngr nexus]# docker push nexus.redhat.local:9999/cdppvc/helloworld:latest
The push refers to repository [nexus.redhat.local:9999/cdppvc/helloworld]
ac28800ec8bb: Pushed
latest: digest: sha256:d37ada95d47ad12224c205a938129df7a3e52345828b4fa27b03a98825d1e2e7 size: 524

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

Step 28. You can see the docker repo and image on Nexus UI.

https://a8a54ef4bb25c49e992c3d68836ac04e-1031794758.us-west-2.elb.amazonaws.com:8443/
https://a2426b40b0bc44d38b878573d5bc253c-367387264.us-west-2.elb.amazonaws.com:9999/



For CDP PvC Data Services on the OpenShift platform, import the CA certificate `nexus.crt` into the OpenShift platform using this method. This step is needed to prevent "x509: certificate signed by unknown authority" issue when the system attempts to pull the docker images from the above Docker registry in the process of provisioning the CDP Data Services Control Plane pods.

****

Hashicorp Vault Setup

This article describes the steps to deploy the external Hashicorp Vault.

(Completely Optional) -- Required only, when you want to perform the setup with ***External Vault Server*** (which requires additional steps). We will proceed for the current setup with the ***Embedded Vault Server.***

Procedure 1. Setup HashiCorp Vault

Step 1. Install the Vault package on *CLDR-MNGR* node:

```
##### Install the following software package.
[root@cldr-mngr ~]# yum install -y yum-utils
##### Add the repo.
[root@cldr-mngr ~]# yum-config-manager --add-repo https://rpm.releases.hashicorp.com/RHEL/hashicorp.repo
##### Install the Vault.
[root@cldr-mngr ~]# yum -y install vault

[root@cldr-mngr ~]# vault --version
[root@cldr-mngr ~]# sudo useradd --system --home /etc/vault --shell /bin/false vault
[root@cldr-mngr ~]# sudo mkdir -p /etc/vault/tls
[root@cldr-mngr ~]# cd /opt
```

Step 2. Generate the certificate for enabling TLS for vault:

```
[root@cldr-mngr opt]# export DOMAIN=vault.redhat.local

[root@cldr-mngr opt]# mkdir vault-certs
[root@cldr-mngr opt]# cd vault-certs

##### Create SSL certificate using the authorized (or self-signed) CA certificate.
[root@cldr-mngr vault-certs]# openssl req -x509 \
        -sha256 -days 356 \
        -nodes \
        -newkey rsa:2048 \
        -subj "/CN=${DOMAIN}/C=US/ST=North Carolina/L=Raleigh" \
        -keyout vault.key -out vault.crt

[root@cldr-mngr vault-certs]# openssl genrsa -out ${DOMAIN}.key 2048
[root@cldr-mngr vault-certs]#
##### Download the vault.crt file (or copy) to your local Laptop machine's ~/ocp directory, it will be
required later during the data service installation.
```

Step 3. Create the csr.conf:

```
[root@cldr-mngr vault-certs]# cat > csr.conf <<EOF
[ req ]
default_bits = 2048
prompt = no
default_md = sha256
req_extensions = req_ext
distinguished_name = dn
[ dn ]
C = US
ST = North Carolina
L = Raleigh
CN = ${DOMAIN}
[ req_ext ]
subjectAltName = @alt_names
[ alt_names ]
DNS.1 = ${DOMAIN}
IP.1 = 127.0.0.1
IP.2 = $(hostname -i)
EOF
[root@cldr-mngr vault-certs]#
```

Step 4. Generate the certificates.

```
[root@cldr-mngr vault-certs]# openssl req -new -key ${DOMAIN}.key -out ${DOMAIN}.csr -config csr.conf
[root@cldr-mngr vault-certs]# openssl x509 -req \
    -in ${DOMAIN}.csr \
    -CA vault.crt -CAkey vault.key \
    -CAcreateserial \
    -out ${DOMAIN}.crt \
    -days 365 -sha256 \
    -extfile csr.conf -extensions req_ext

[root@cldr-mngr vault-certs]# ls -l | grep crt
total 28
-rw-r--r-- 1 root root  297 Aug 12 05:53 csr.conf
-rw-r--r-- 1 root root 1326 Aug 12 05:53 vault.redhat.local.crt
-rw-r--r-- 1 root root 1062 Aug 12 05:53 vault.redhat.local.csr
-rw------- 1 root root 1704 Aug 12 05:52 vault.redhat.local.key
-rw-r--r-- 1 root root 1289 Aug 12 05:52 vault.crt
-rw------- 1 root root 1704 Aug 12 05:52 vault.key
-rw-r--r-- 1 root root   41 Aug 12 05:53 vault.srl
[root@cldr-mngr vault-certs]#

#### Copy the certificates to /opt/vault/tls directory and change the permission.
[root@cldr-mngr vault-certs]# chmod 0640 *.key
[root@cldr-mngr vault-certs]# chmod 0644 *.key
[root@cldr-mngr vault-certs]# chown root:root *.crt
[root@cldr-mngr vault-certs]# chown root:vault *.key
[root@cldr-mngr vault-certs]# cp *.crt /opt/vault/tls/
[root@cldr-mngr vault-certs]# cp *.key /opt/vault/tls/

[root@cldr-mngr vault-certs]# ll
total 28
-rw-r--r-- 1 root root   297 Aug 12 05:53 csr.conf
-rw-r--r-- 1 root root  1326 Aug 12 05:53 vault.redhat.local.crt
-rw-r--r-- 1 root root  1062 Aug 12 05:53 vault.redhat.local.csr
-rw-r--r-- 1 root vault 1704 Aug 12 05:52 vault.redhat.local.key
-rw-r--r-- 1 root root  1289 Aug 12 05:52 vault.crt
-rw-r--r-- 1 root vault 1704 Aug 12 05:52 vault.key
-rw-r--r-- 1 root root    41 Aug 12 05:53 vault.srl
[root@cldr-mngr vault-certs]#
```

Step 5. Update the vault.hcl config.

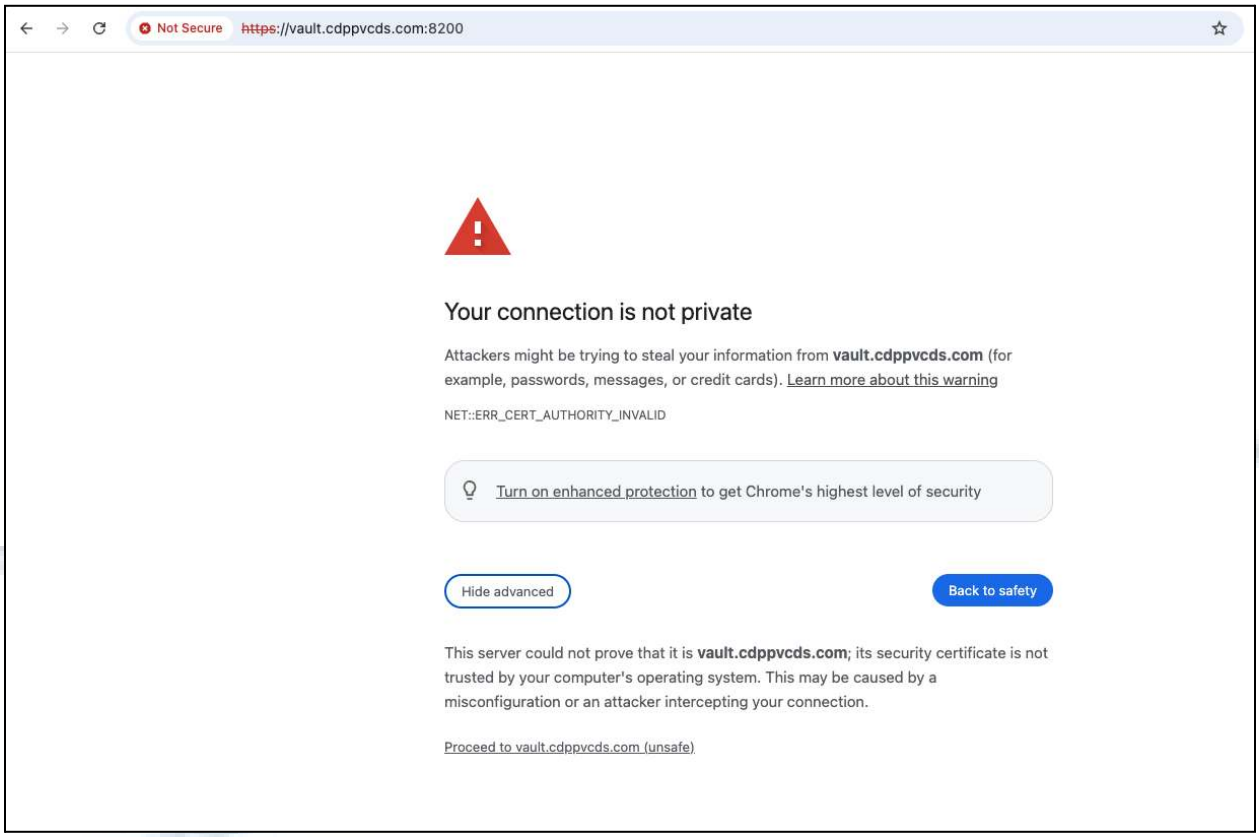
```
##### Backup the /etc/vault.d/vault.hcl file.
[root@cldr-mngr opt]# cp /etc/vault.d/vault.hcl /etc/vault.d/vault.hcl-orig

##### Edit the /etc/vault.d/vault.hcl file. Here, 192.168.1.38 is the IP address of the node where
vault-server will be installed
[root@cldr-mngr opt]# cat>/etc/vault.d/vault.hcl
ui = true
cluster_addr  = "https://192.168.1.38:8201"
api_addr      = "https://192.168.1.38:8200"
#mlock = true
disable_mlock = true

storage "file" {
  path = "/opt/vault/data"
}

# HTTPS listener
listener "tcp" {
  address       = "0.0.0.0:8200"
  tls_cert_file = "/opt/vault/tls/vault.redhat.local.crt"
  tls_key_file  = "/opt/vault/tls/vault.redhat.local.key"
  tls_client_ca_file = "/opt/vault/tls/vault.crt"
}
[root@cldr-mngr opt]#
```

Step 6. Enable and start the Vault server. Vault is now TLS enabled. Generate the unseal token and initialize the Vault.

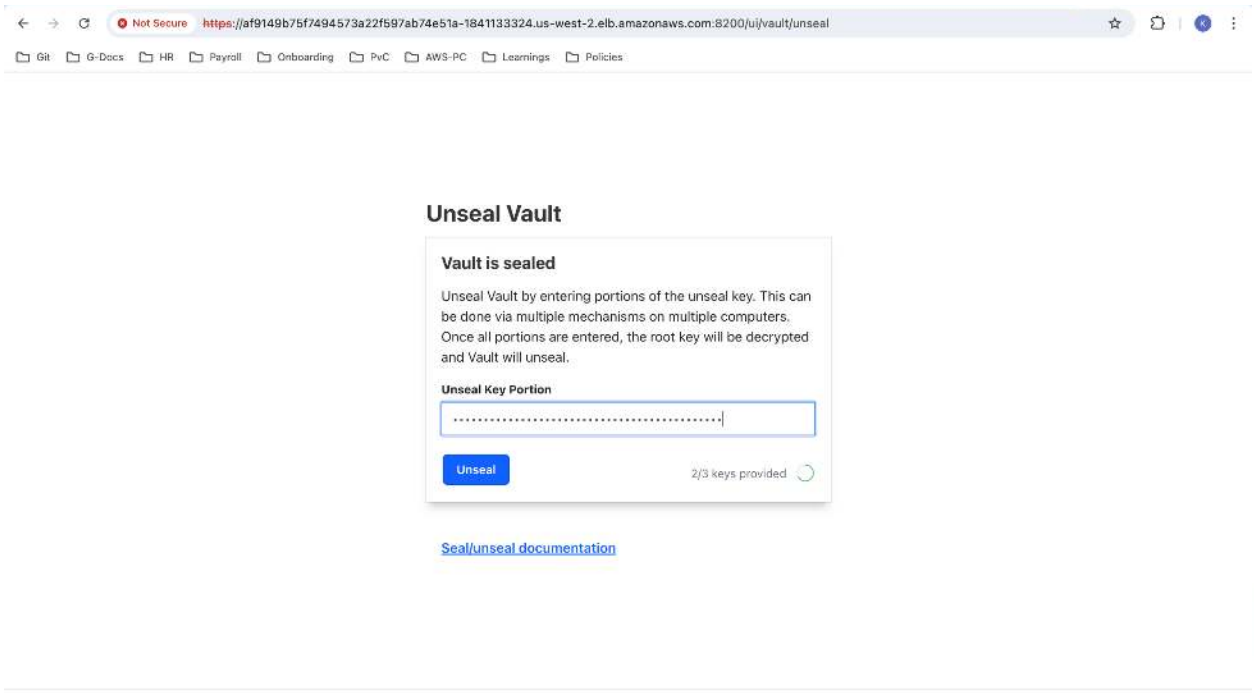
```
##### Enable and start the Vault service.
[root@cldr-mngr vault-certs]# systemctl enable vault.service
Created symlink /etc/systemd/system/multi-user.target.wants/vault.service →
/usr/lib/systemd/system/vault.service.
[root@cldr-mngr vault-certs]# systemctl start vault.service

##### Check that port 8200 is up and running.
[root@cldr-mngr opt]# netstat -an | grep -e 8200 -e 8201
tcp        0      0 0.0.0.0:8200            0.0.0.0:*               LISTEN

##### Check the status of the Vault. You might encounter the following errors and if yes, proceed with the
following procedures to fix the errors.
[root@cldr-mngr opt]# vault status
Error checking seal status: Get "https://127.0.0.1:8200/v1/sys/seal-status": x509: cannot validate
certificate for 127.0.0.1 because it doesn't contain any IP SANs

[root@cldr-mngr vault-certs]# echo "export VAULT_ADDR=https://$(hostname -i):8200" >> ~/.bashrc && source
~/.bashrc

[root@cldr-mngr opt]# vault status
Error checking seal status: Get "https://vault.redhat.local:8200/v1/sys/seal-status": x509: certificate
signed by unknown authority

[root@cldr-mngr opt]# cp vault.crt /etc/pki/ca-trust/source/anchors/
[root@cldr-mngr opt]# update-ca-trust extract
[root@cldr-mngr opt]# vault status
Key                Value
---                -----
Seal Type          shamir
Initialized        false
Sealed             true
Total Shares       0
Threshold          0
Unseal Progress    0/0
Unseal Nonce       n/a
Version            1.17.3
Build Date         2024-08-06T14:28:45Z
Storage Type       file
HA Enabled         false

##### Initialize the Vault operator. Jot down the produced root token and its associated Unseal key (output
of below command) to unseal the Vault when needed and store it to file in secure place: /opt/vault/secure.txt
[root@cldr-mngr vault-certs]# vault operator init | tee secure.txt
Unseal Key 1: 4ivPJFgY0726s087/mgsvaksrkE4Zg3GW0ykyNkgmpG0
Unseal Key 2: hsVjbN6T6sr83eY66uOnb+BVwBdcmPBUrpjRsszOMBDu
Unseal Key 3: mGzKoI8L/4IvKwSD6NJZlDT/BRcWmM8UhAJyGN/kRdD5
Unseal Key 4: CYlJwx5rKvpm/1jpZrxWsb/qa9zOWde3UPjlpKmmUuha
Unseal Key 5: mbwvrLmTs7ltqiwzAT5sk5gJX9Lg/DKbYiox+nO2AZEO

Initial Root Token: hvs.Kd1tgPb6fYY4M2BeXFhPYlsq

Vault initialized with 5 key shares and a key threshold of 3. Please securely
distribute the key shares printed above. When the Vault is re-sealed,
restarted, or stopped, you must supply at least 3 of these keys to unseal it
before it can start servicing requests.

Vault does not store the generated root key. Without at least 3 keys to
reconstruct the root key, Vault will remain permanently sealed!

It is possible to generate new unseal keys, provided you have a quorum of
existing unseal keys shares. See "vault operator rekey" for more information.

##### Create and expose the Vault service to be able to access the Vault WebUI.
ksahu@Kuldeeps-MacBook-Air ocp % virtctl expose vm clouderamanager --name vault-webtlssvc --type LoadBalancer
--port 8200
Service vault-webtlssvc successfully exposed for vm clouderamanager
```

```
##### List the Vault Service LoadBalancer URL.
ksahu@Kuldeeps-MacBook-Air ocp % oc get svc | grep vault
clouderamanager-vaultwebsvc    LoadBalancer   172.30.167.46
af9149b75f7494573a22f597ab74e51a-1841133324.us-west-2.elb.amazonaws.com   8200:30464/TCP   10d

#####
You may now log in to the SSL-enabled Vault portal https://vault.redhat.local:8200 WebUI
https://<Vault_IP_Addr>:8200/ui
https://af9149b75f7494573a22f597ab74e51a-1841133324.us-west-2.elb.amazonaws.com:8200/ui/
https://af9149b75f7494573a22f597ab74e51a-1841133324.us-west-2.elb.amazonaws.com:8200/ui/vault/dashboard
```

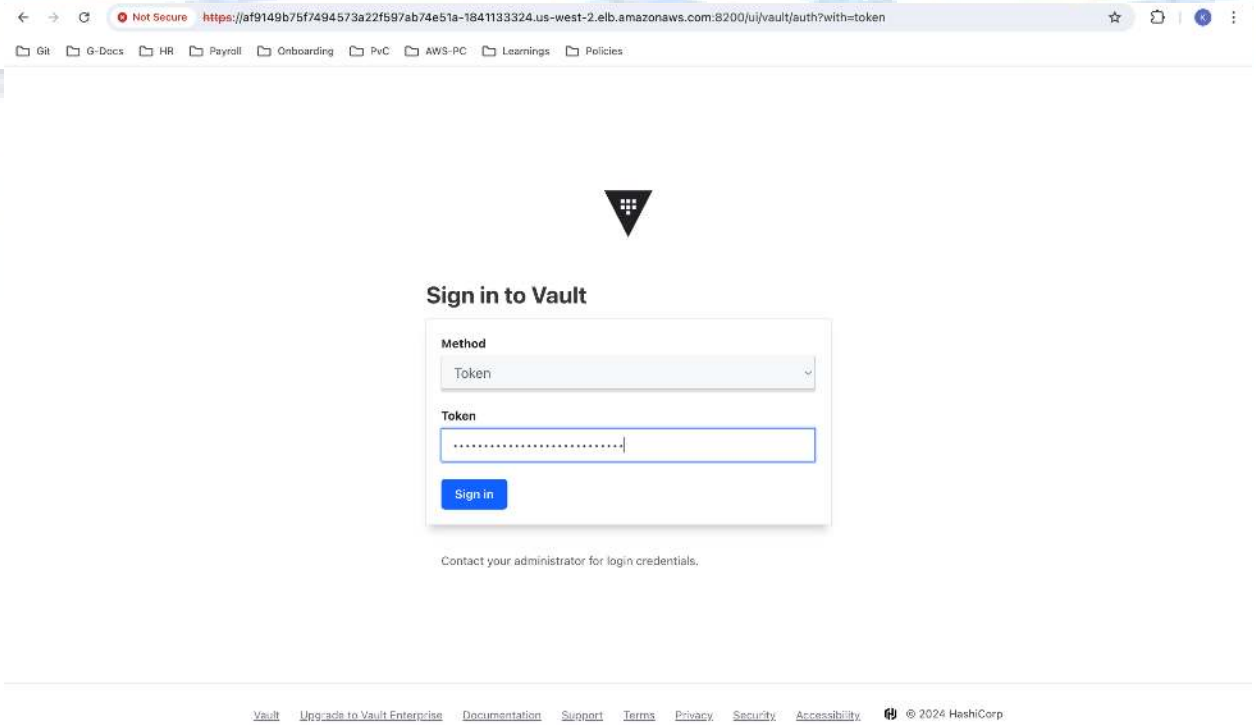
Step 7. You will see the below screen on the browser, when opening for the first time. Just click on the *Advanced* button and *Proceed to the URL (Unsafe)*, as shown below.



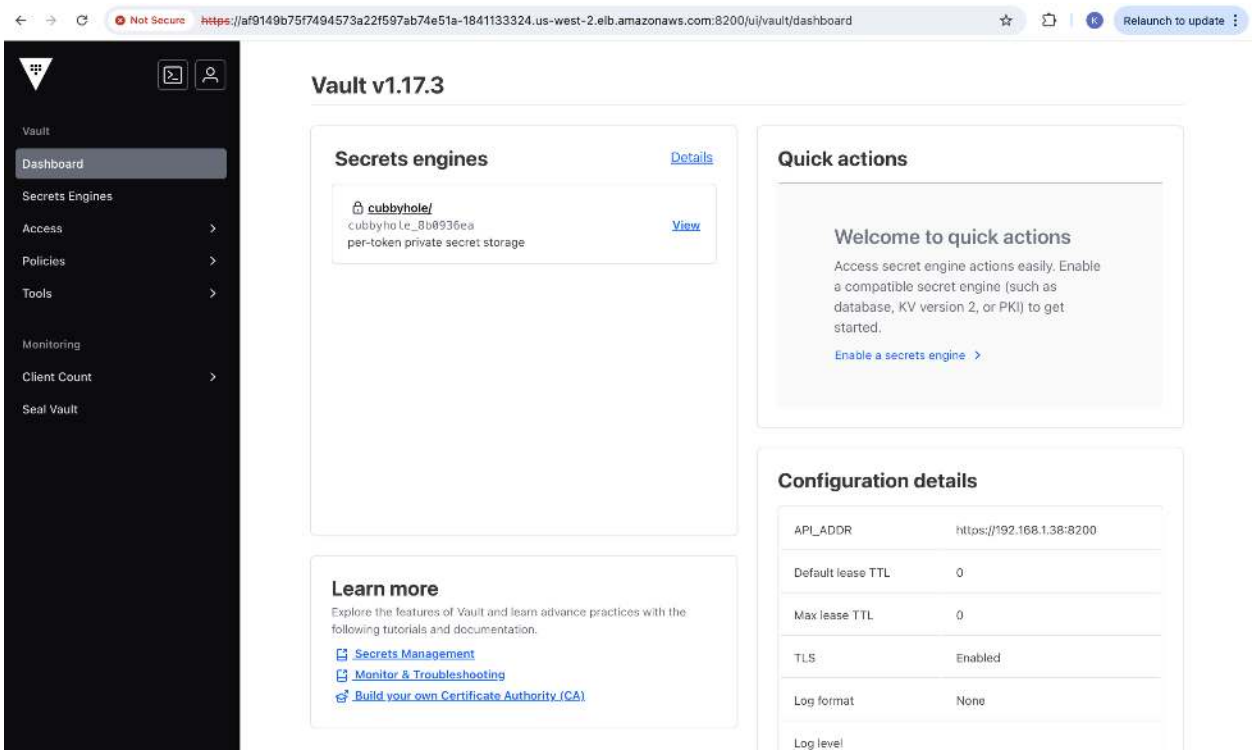
Step 8. You will see the below screen on the browser, Unseal the Vault, with the help of the unseal token, generated in the previous steps. You need to provide any 3 unseal tokens out of 5.



Step 9. You will see the below screen on the browser, input the root token, generated in the previous steps.



Step 10. Vault Home page is available.



****

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/OCP, 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_REPO_URL="https://${USERNAME}:${PASSWORD}@archive.cloudera.com/p/cm7/${CM_VERSION}"
BASE_PARCEL_URL="${BASE_REPO_URL}/redhat9/yum"

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

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

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

wget ${BASE_PARCEL_URL}/cloudera-manager.repo
```

```
wget ${BASE_PARCEL_URL}/cloudera-manager-trial.repo
wget ${BASE_PARCEL_URL}/RPM-GPG-KEY-cloudera
wget ${BASE_REPO_URL}/allkeys.asc
wget ${BASE_REPO_URL}/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 ${BASE_PARCEL_URL}/RPMS/x86_64/cloudera-manager-${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 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 ~]#
```

Note:

Step 2. Make sure that the data directory, which by default is /var/lib/pgsql/16/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 3. 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.redhat.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 4. 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/16/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 5. 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 6. To enable MD5 authentication, edit /var/lib/pgsql/16/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/16/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/16/data/pg_hba.conf ~

##### If you have the file backed up, copy it
[root@cldr-mngr ~]# cp /var/lib/pgsql/16/data/pg_hba.conf /var/lib/pgsql/16/data/pg_hba.conf_orig
[root@cldr-mngr ~]# cp ~/pg_hba.conf /var/lib/pgsql/16/data/pg_hba.conf
```

Step 7. Configure settings to ensure your system performs as expected. Update these settings in the /var/lib/pgsql/16/data/postgresql.conf file. Settings vary based on cluster size and resources as follows:

```
[root@cldr-mngr ~]# vi /var/lib/pgsql/16/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/16/data/postgresql.conf ~

##### If you have the file backed up, copy it
[root@cldr-mngr ~]# cp /var/lib/pgsql/16/data/postgresql.conf /var/lib/pgsql/16/data/postgresql.conf_orig
[root@cldr-mngr ~]# cp ~/postgresql.conf /var/lib/pgsql/16/data/postgresql.conf
```

Note: Settings vary based on cluster size and resources.

Step 8. 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 9. 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.redhat.local -d postgres -U postgres
Password for user postgres: <postgres>
psql (14.13)
Type "help" for help.

postgres=#\q
```

Step 10. 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/16/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.redhat.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.redhat.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.redhat.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.redhat.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.redhat.local
        Validity
            Not Before: May 14 06:33:09 2024 GMT
            Not After : May 12 06:33:09 2034 GMT
        Subject: CN = cldr-mngr.redhat.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/16/data/server.key

##### Edit Configuration file for PostgreSQL (postgresql.conf) to enable SSL
[root@cldr-mngr data]# cat <<EOF >> /var/lib/pgsql/16/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/16/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.redhat.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/16/data/root.crt to /root/.postgresql/root.crt
[root@cldr-mngr data]# mkdir -p /root/.postgresql/
[root@cldr-mngr data]# cp /var/lib/pgsql/16/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.redhat.local -p 5432 -U postgres "dbname=postgres
sslmode=verify-full"
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=#
postgres=# \q

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

Step 11. 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 5. 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.redhat.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/16/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/16/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.redhat.local:5432/scm?ssl=true&sslmode=verify-ca&sslrootcert=/var/lib/cloudera-sc
m-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.redhat.local --jdbc-url
"jdbc:postgresql://cldr-mngr.redhat.local:5432/scm?ssl=true&sslmode=verify-ca&sslrootcert=/var/lib/cloudera-s
cm-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.redhat.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.redhat.local:5432/scm?ssl=true&s
slmode=verify-ca&sslrootcert=/var/lib/cloudera-scm-server/.postgresql/root.crt
[root@cldr-mngr ~]#
```

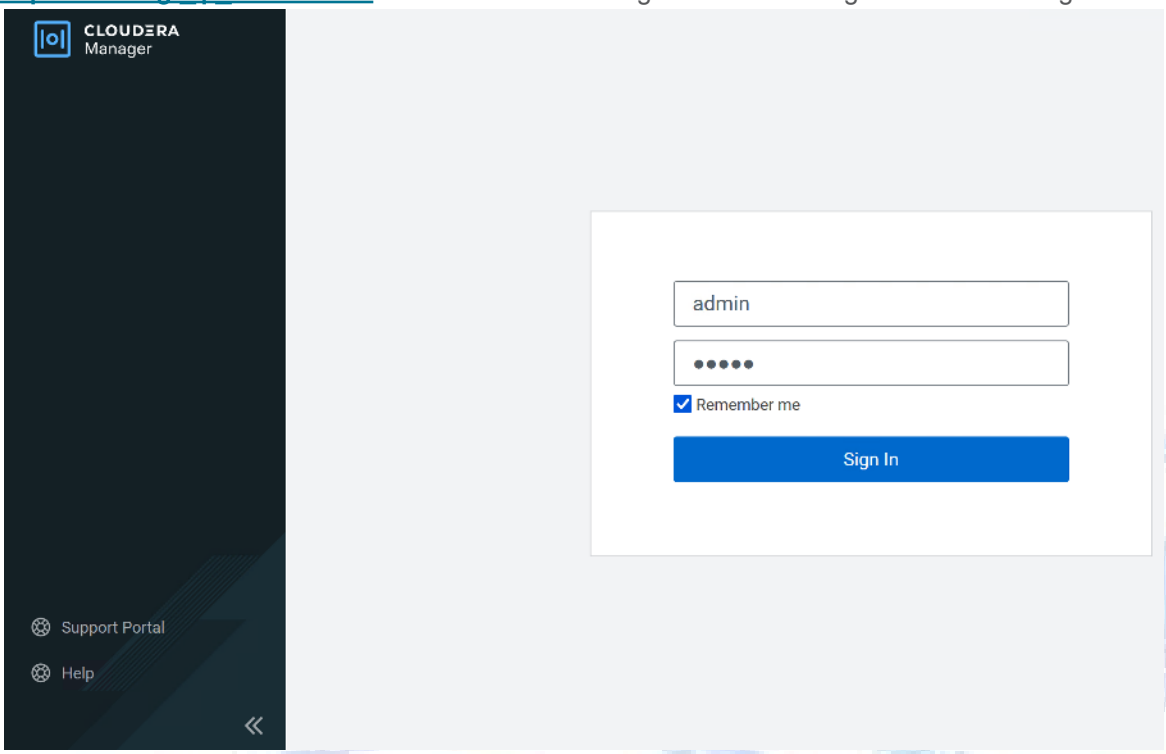
Step 8. Start the Cloudera Manager Server:

```
[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.

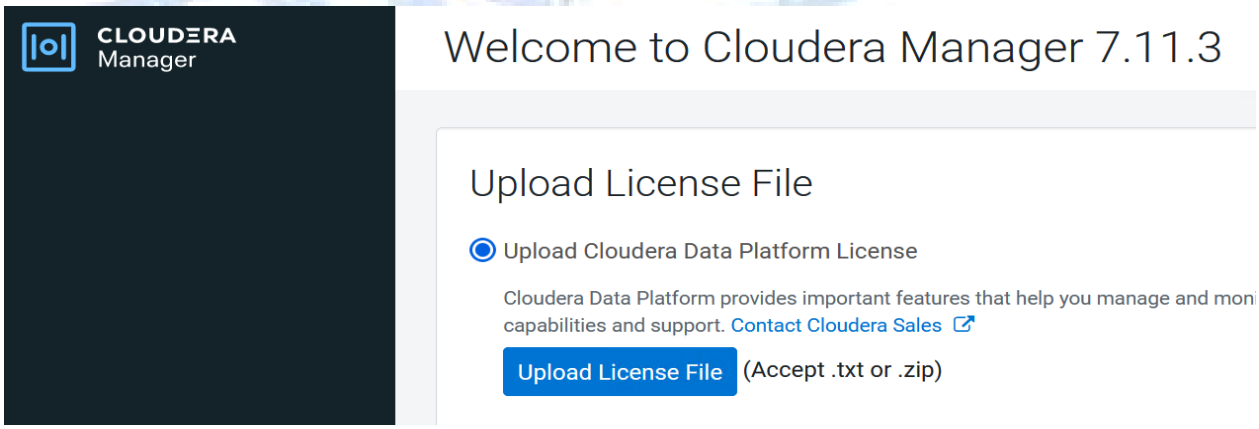
```
INFO WebServerImpl:org.eclipse.jetty.server.Server: Started @89711ms
INFO WebServerImpl:com.cloudera.server.cmf.WebServerImpl: Started Jetty server.
```

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.



Note: The default username and password for Cloudera Manager is ***admin/admin.***

Step 11. 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 12. Activate your license for Cloudera Data Platform by clicking the ***Continue*** button. Click Continue.

Step 13. The ***Add Private Cloud Base Cluster*** page appears. Next, we will enable AutoTLS for CM.

Step 14. 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
```

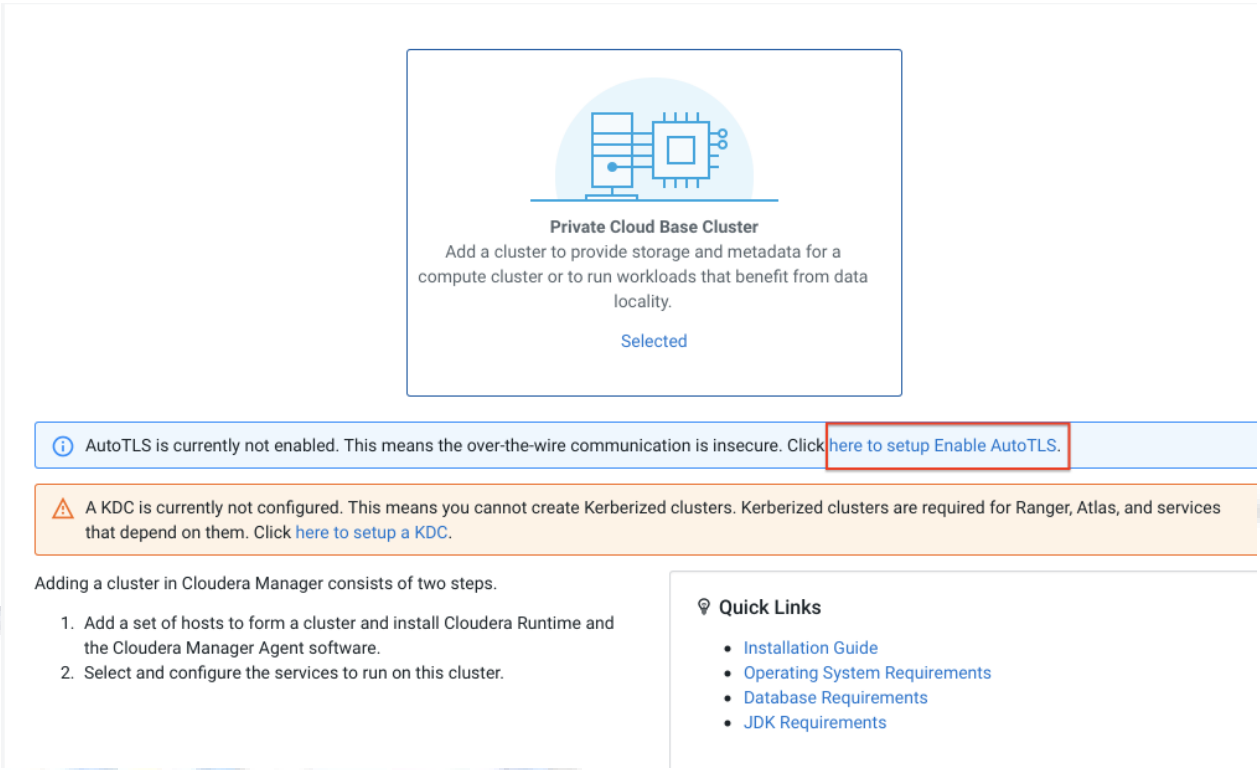
```
[root@cdpbase cloudera-scm-server]# systemctl status cloudera-scm-agent
● cloudera-scm-agent.service - Cloudera Manager Agent Service
   Loaded: loaded (/usr/lib/systemd/system/cloudera-scm-agent.service; enabled; vendor preset: disabled)
   Active: active (running) since Sat 2023-04-08 02:24:22 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 6. Enable AutoTLS

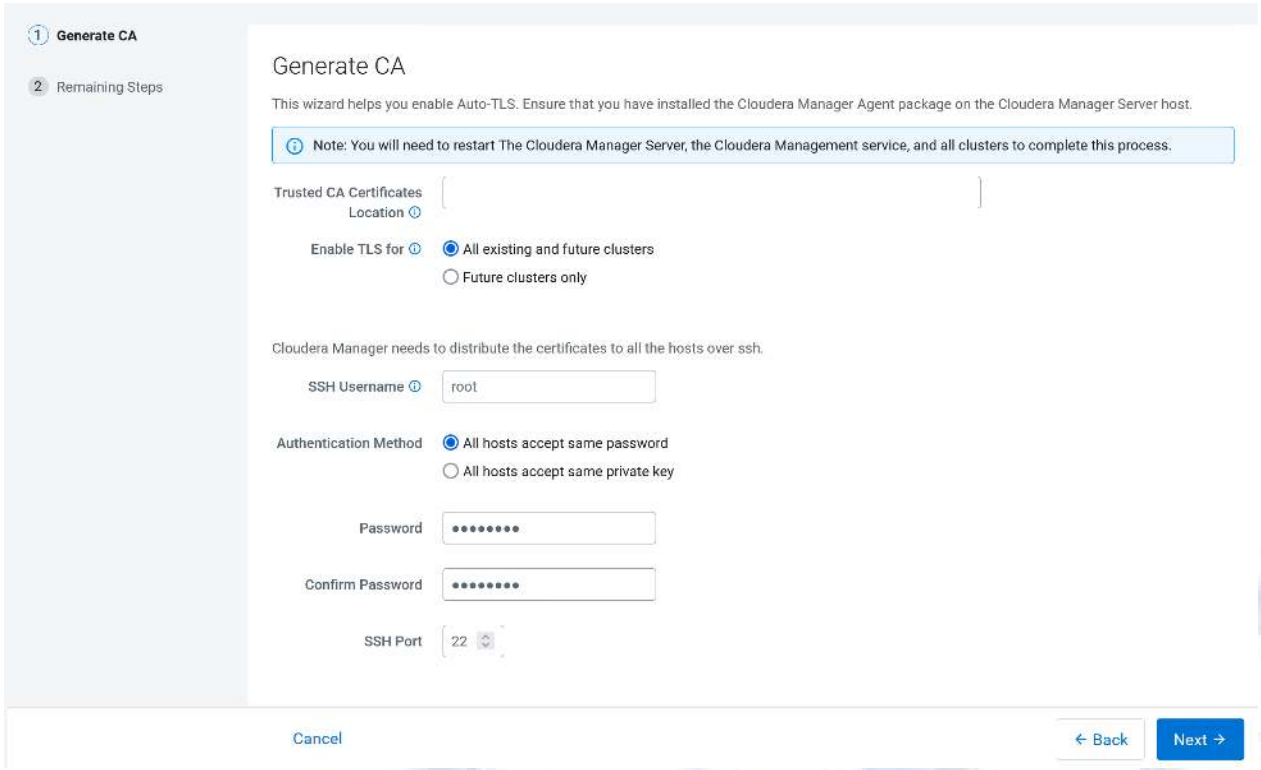
Auto-TLS is managed using the certmanager utility, which is included in the Cloudera Manager Agent software, and not the Cloudera Manager Server software. You must install the Cloudera Manager Agent software on the Cloudera Manager Server host to be able to use the utility. You can use certmanager to 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.

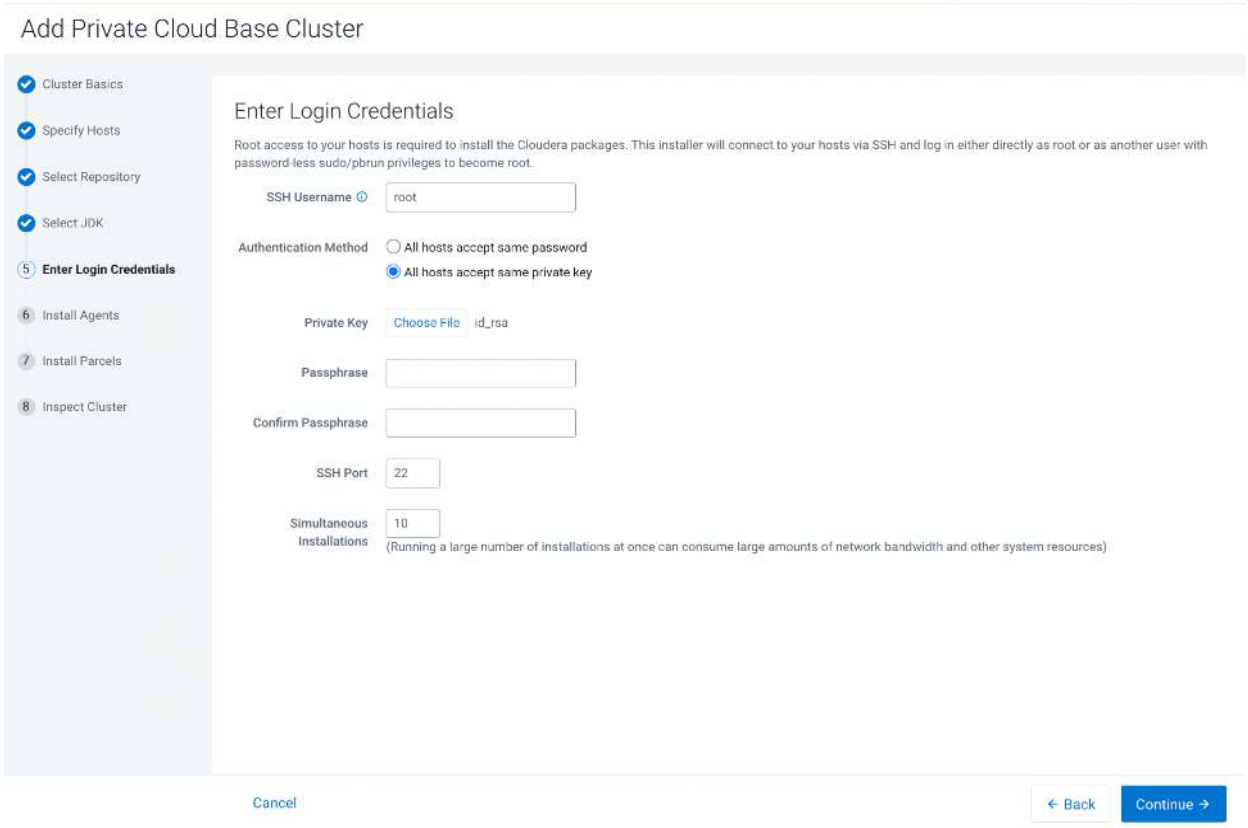


Step 2. Below screen will appear. Enter the values for the parameters as shown below. (**We will be using the private key approach**, you can use password option as well, both options should considerably work)

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.

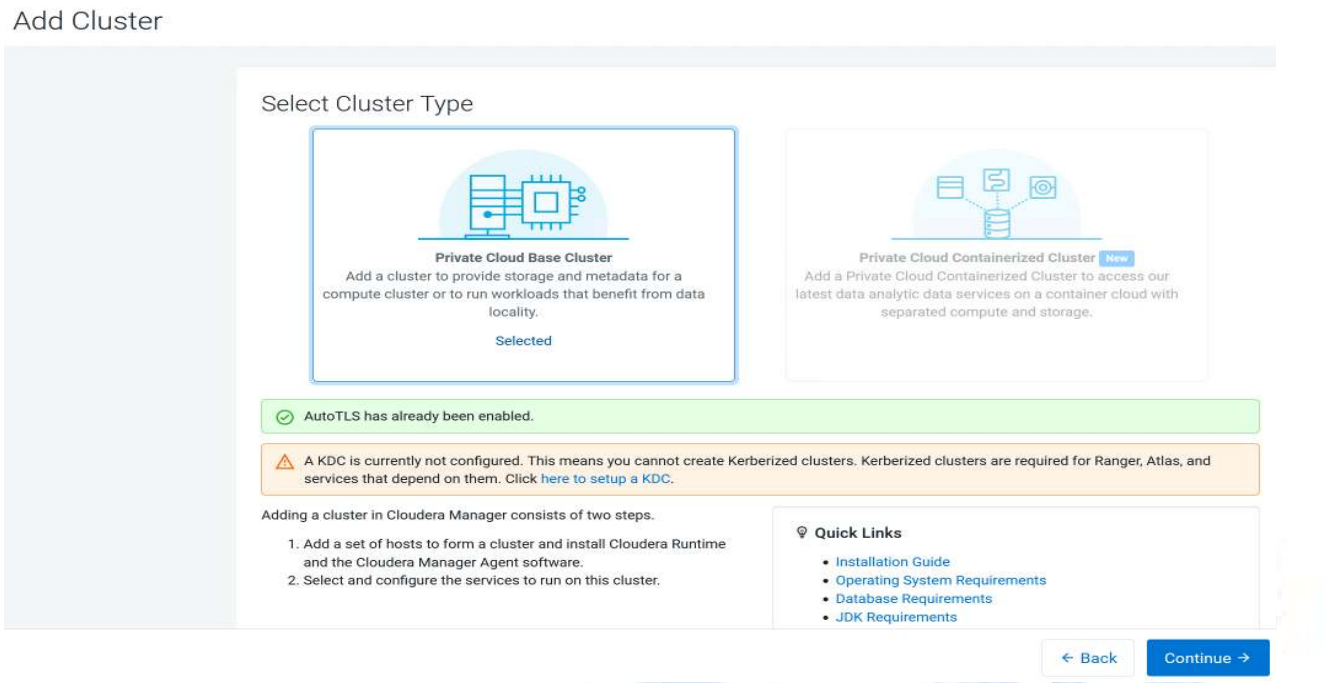
Step 5. After enabling the AutoTLS for the CM-UI through the browser, login to cldr-mngr node at backend as user root and restart Cloudera Manager Server, suggested in the previous screenshot.

```
[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 7. 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}
```

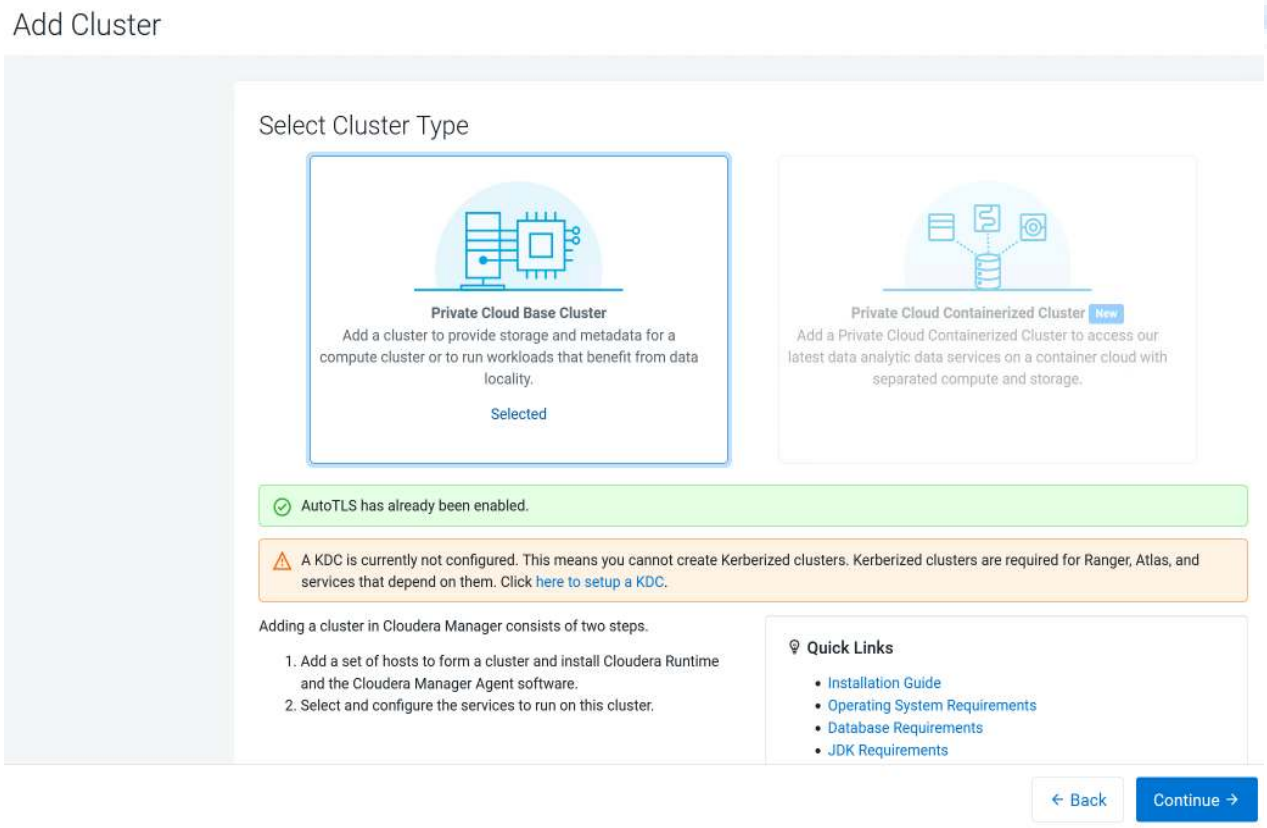
```
[libdefaults]
 default_realm = CDPPVCDS.COM
 dns_lookup_realm = false
 dns_lookup_kdc = true
 rdns = false
 ticket_lifetime = 24h
 forwardable = true
 udp_preference_limit = 0
# default_ccache_name = KEYRING:persistent:%{uid}
```

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.

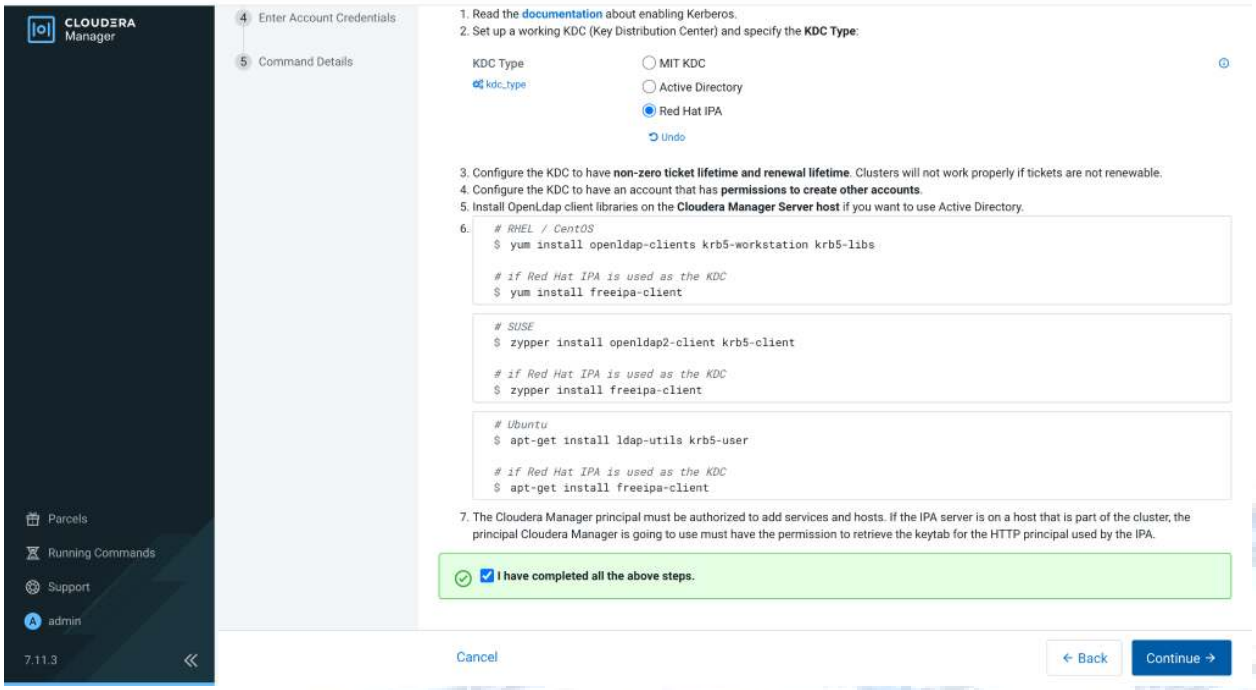
Step 3. In the Cloudera manager console click on ***here to set up a KDC***, on the same TLS page, to enable the kerberos authentication on the cluster.



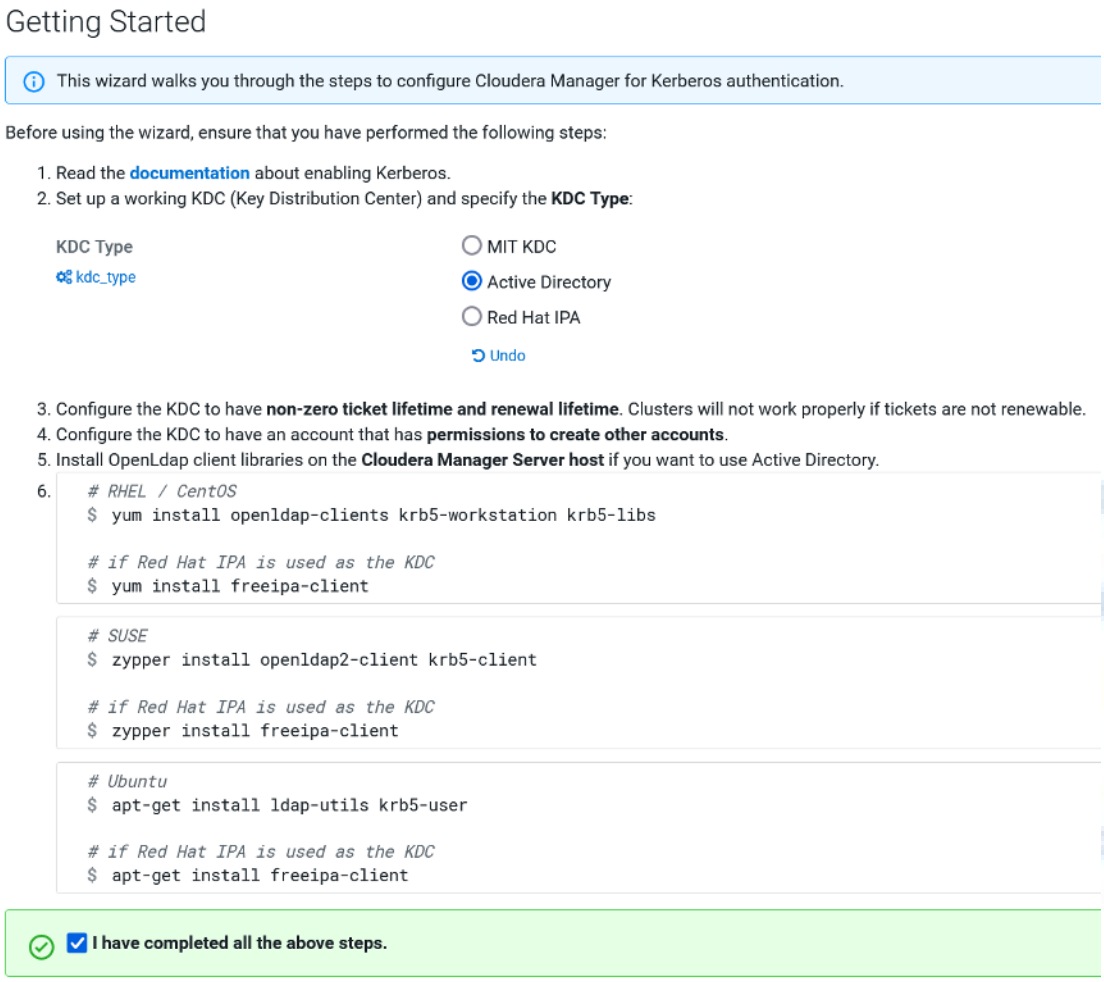
Step 4. Click Continue.

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"
```

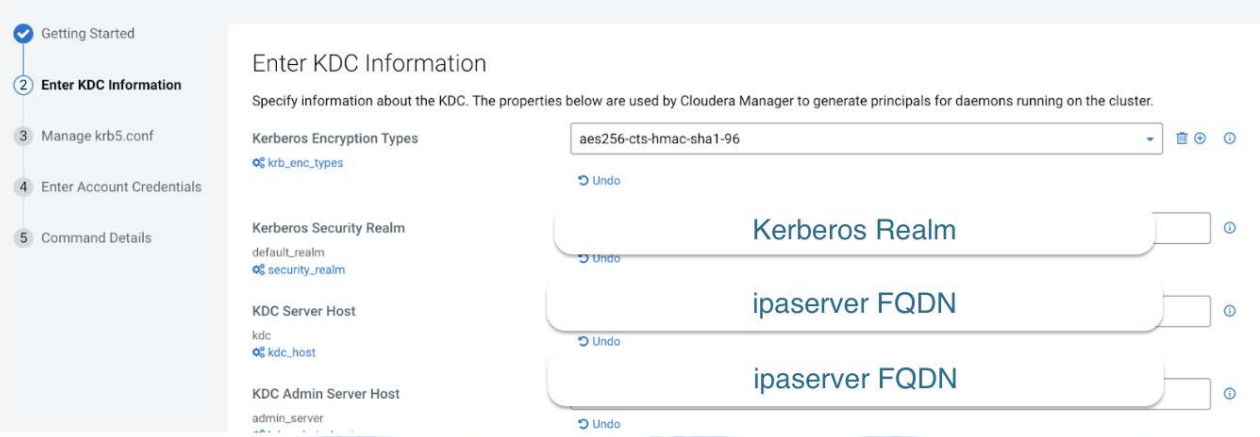
Step 8. For enabling Kerberos, under the Enter KDC Information page, provide below inputs for ***Enter KDC information*** for this Cloudera Manager. Use Table 6 as an example to fill-in the KDC setup information, provide below inputs and click Next.

Table 4. KDC Setup components and their corresponding value

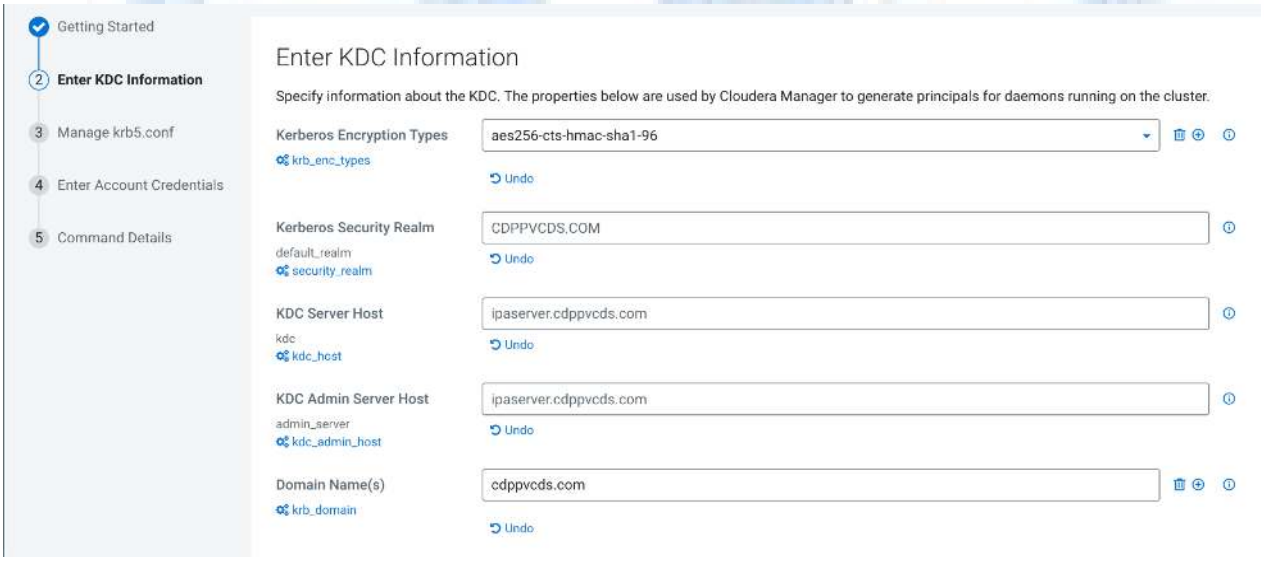
Component	Value
Kerberos Encryption Types	aes256-cts-hmac-sha1-96
Kerberos Security Realm	REDHAT.LOCAL
KDC Server Host	ipaserver.redhat.local
KDC Admin Server Host	ipaserver.redhat.local

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

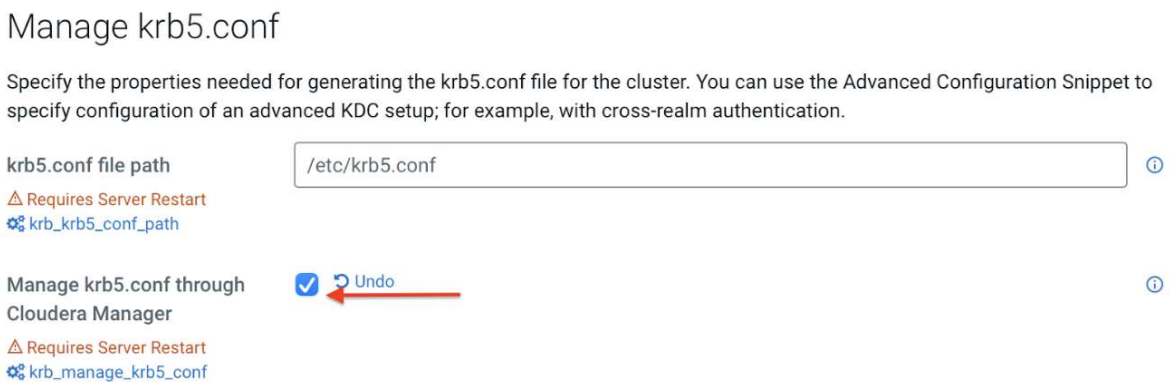
Check the picture below on where to populate the above mentioned fields:



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

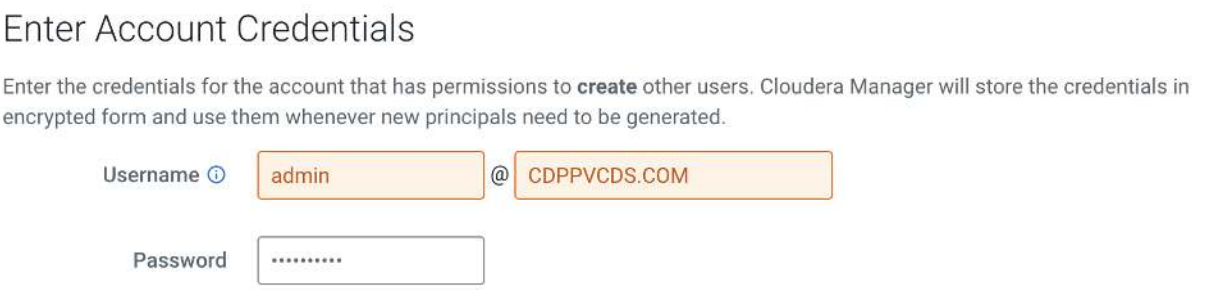


Step 9. On the Next Page, check the box for ***Manage "krb5.conf"*** to enable it through Cloudera Manager. This will install the krb5.conf file in all the hosts selected for the data lake.



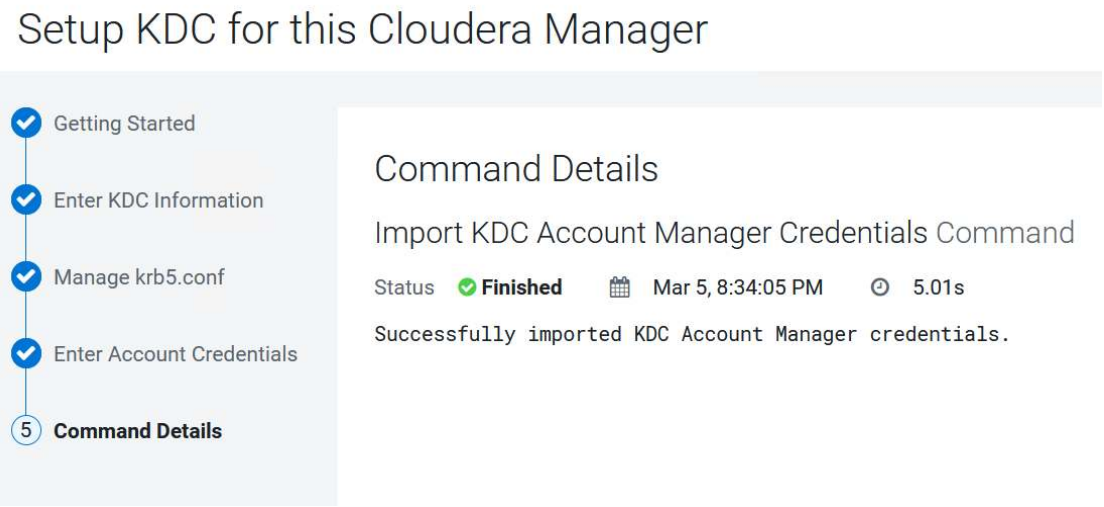
Step 10. Next, enter the details as per the configuration of FreeIPA you did before. i.e., provide the domain and password of the admin user configured earlier in the FreeIPA setup. Enter account credentials for the admin which you have created. This credential will be used to generate the keytabs. In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



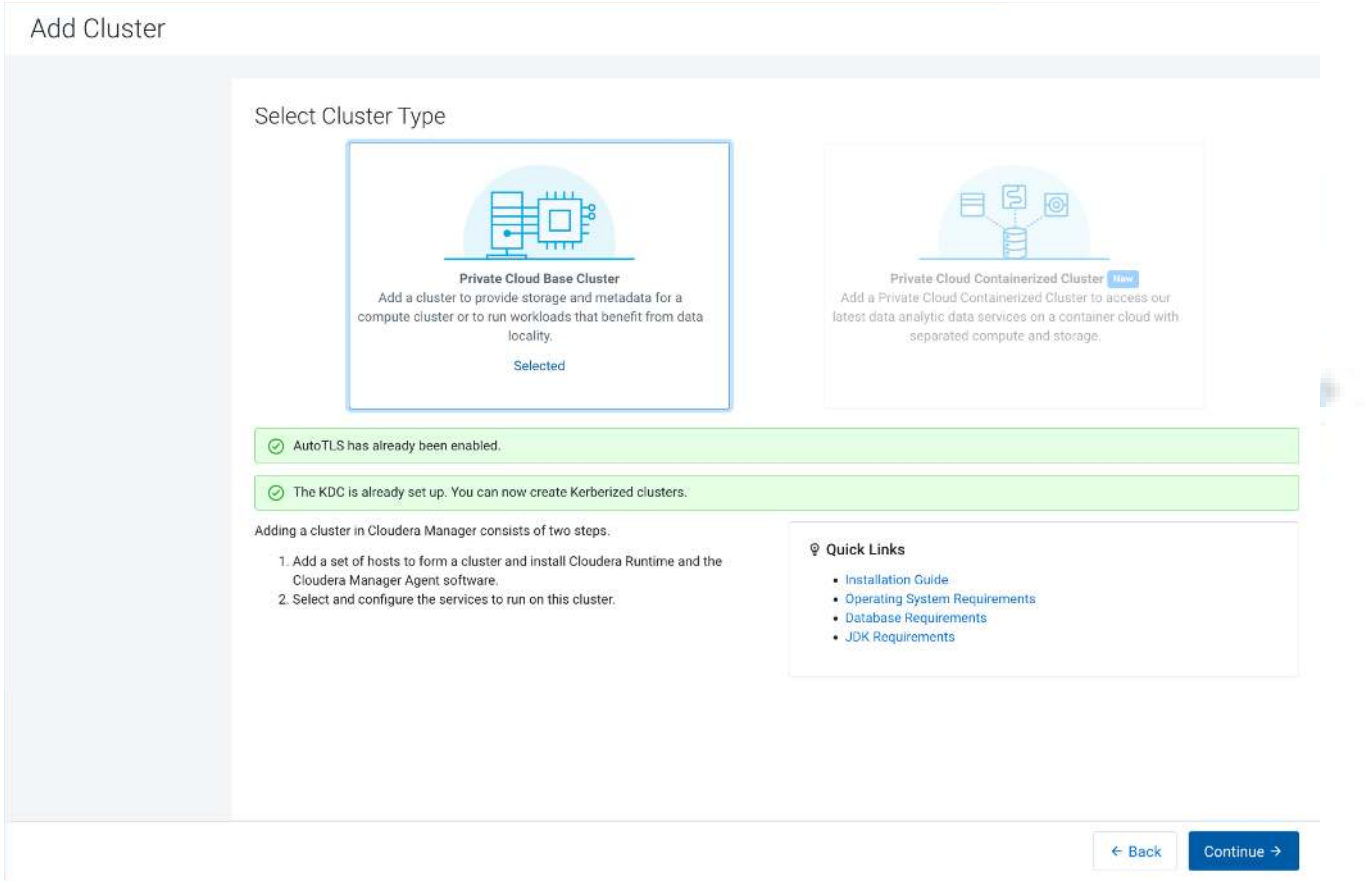
Step 11. Click Finish to complete the KDC setup.

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:



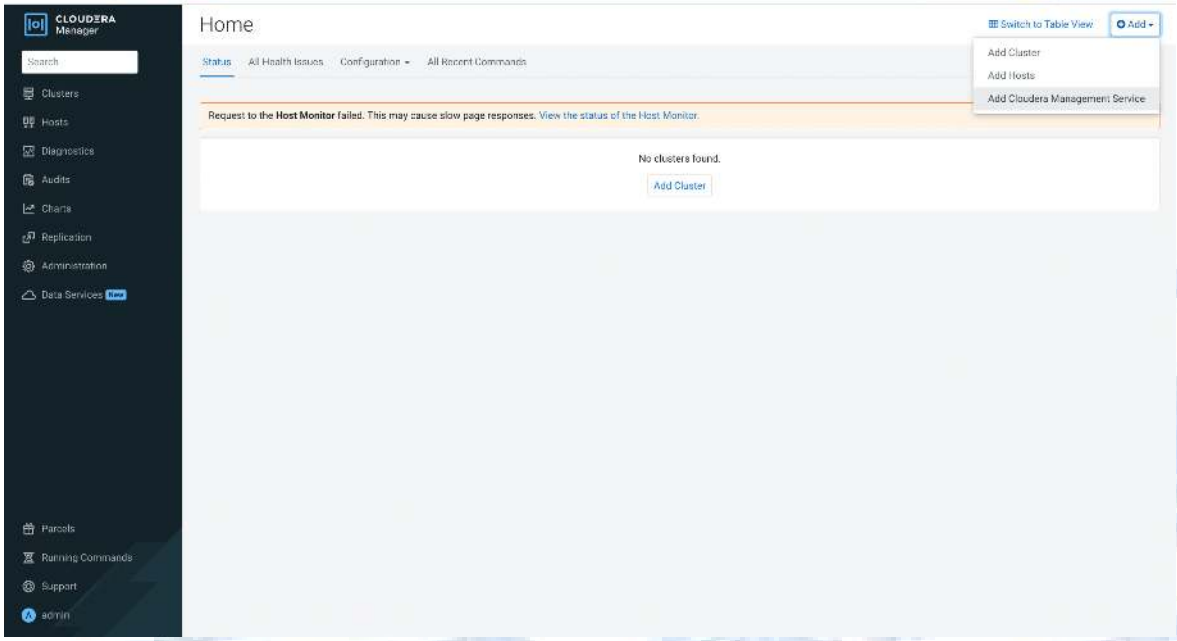
Step 14. Verify Kerberos configuration.

```
[root@ipaserver ~]# kinit admin@redhat.local
Password for admin@REDHAT.LOCAL: <redhat123>
[root@ipaserver ~]# klist -e
Ticket cache: KCM:0
Default principal: admin@REDHAT.LOCAL

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

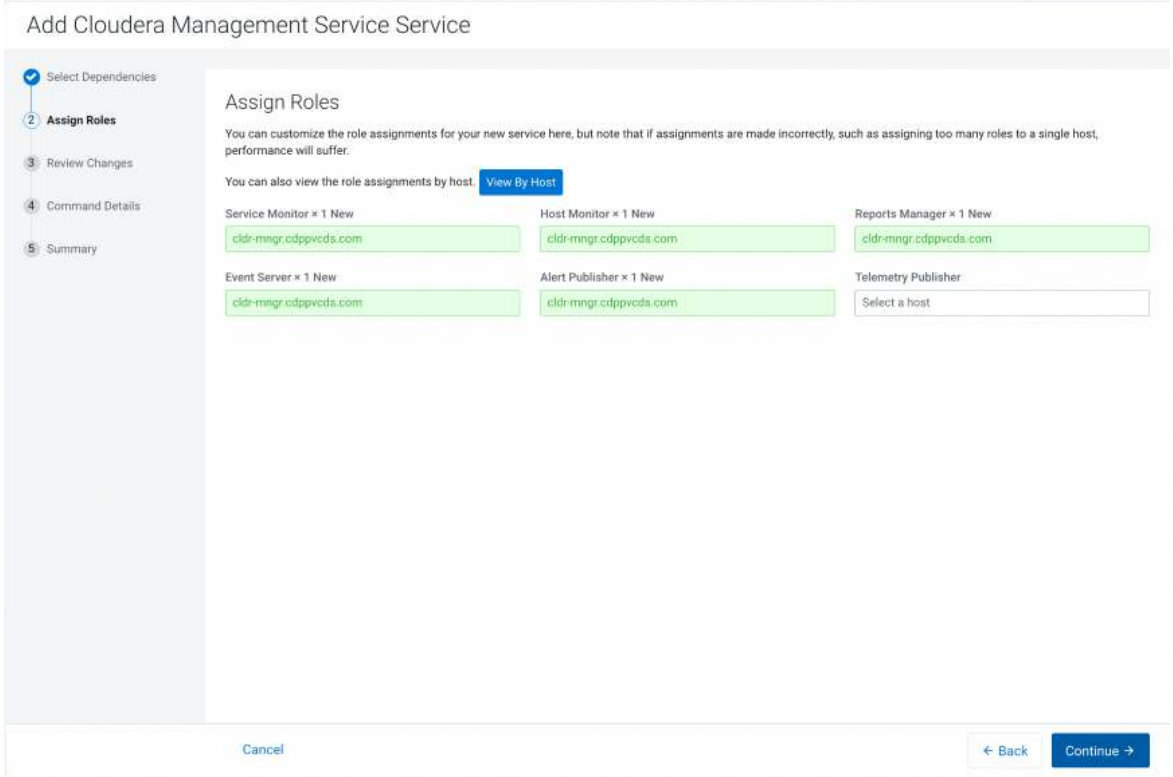
Step 15. Setup the *Cloudera Management Services*: **(Only if the status of hosts/services/charts are not visible / visible as (?) / or showing the errors on console)**

```
##### 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.
```

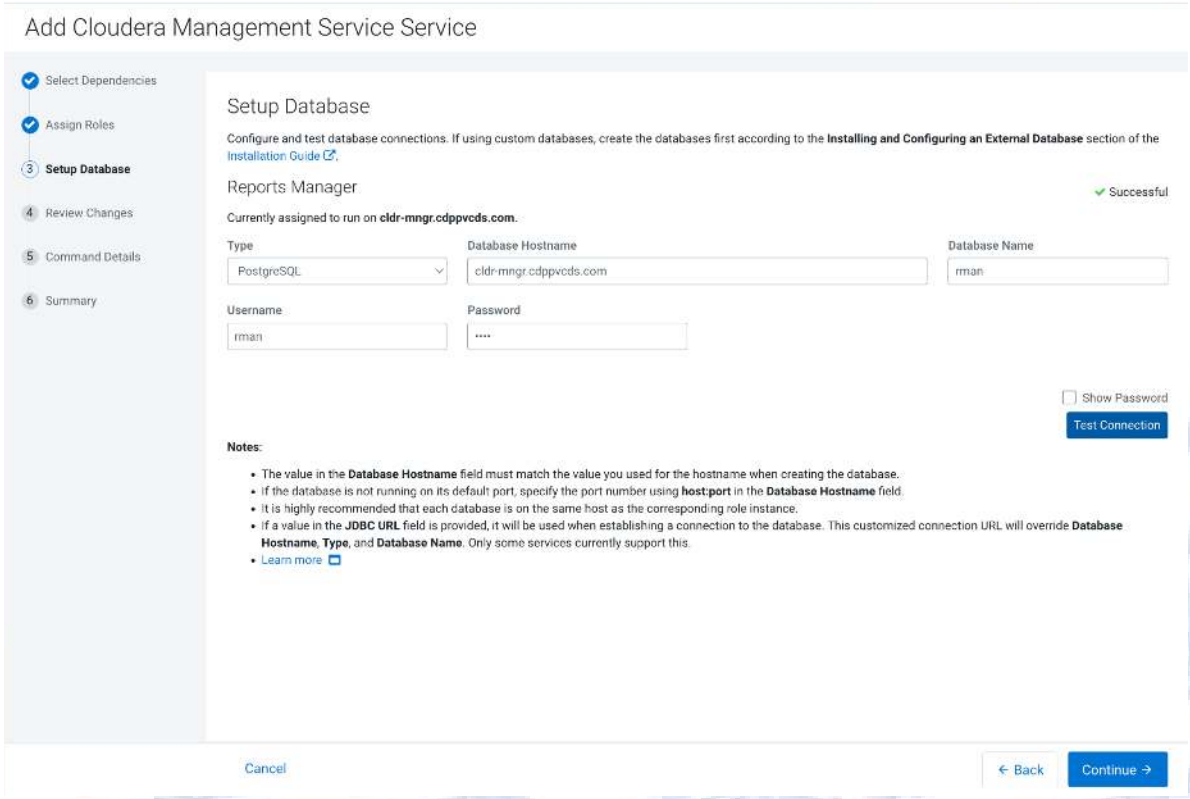
a) Go to —> WebUI -> Top Right Corner -> (+)Add -> Add Cloudera Management Service



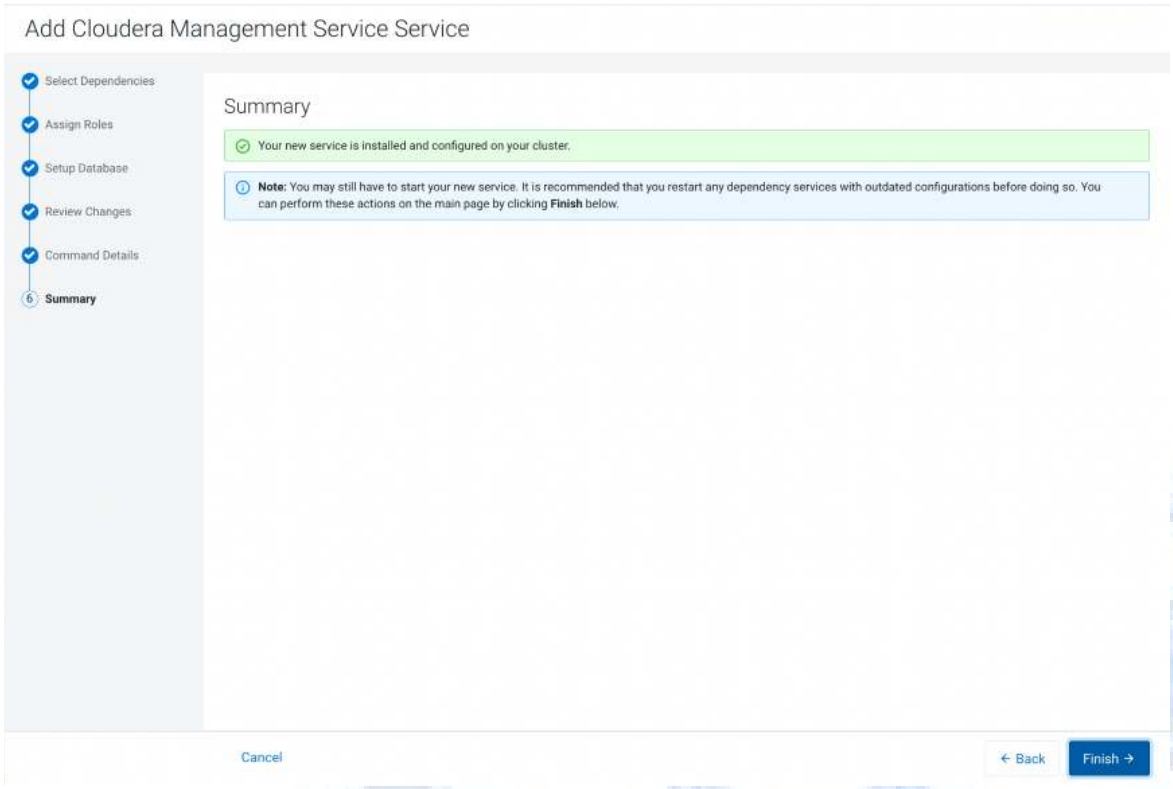
b) Assign Roles for different Cloudera Management Services to CLDR-MNGR Host (e.g. cldr-mngr.redhat.local)



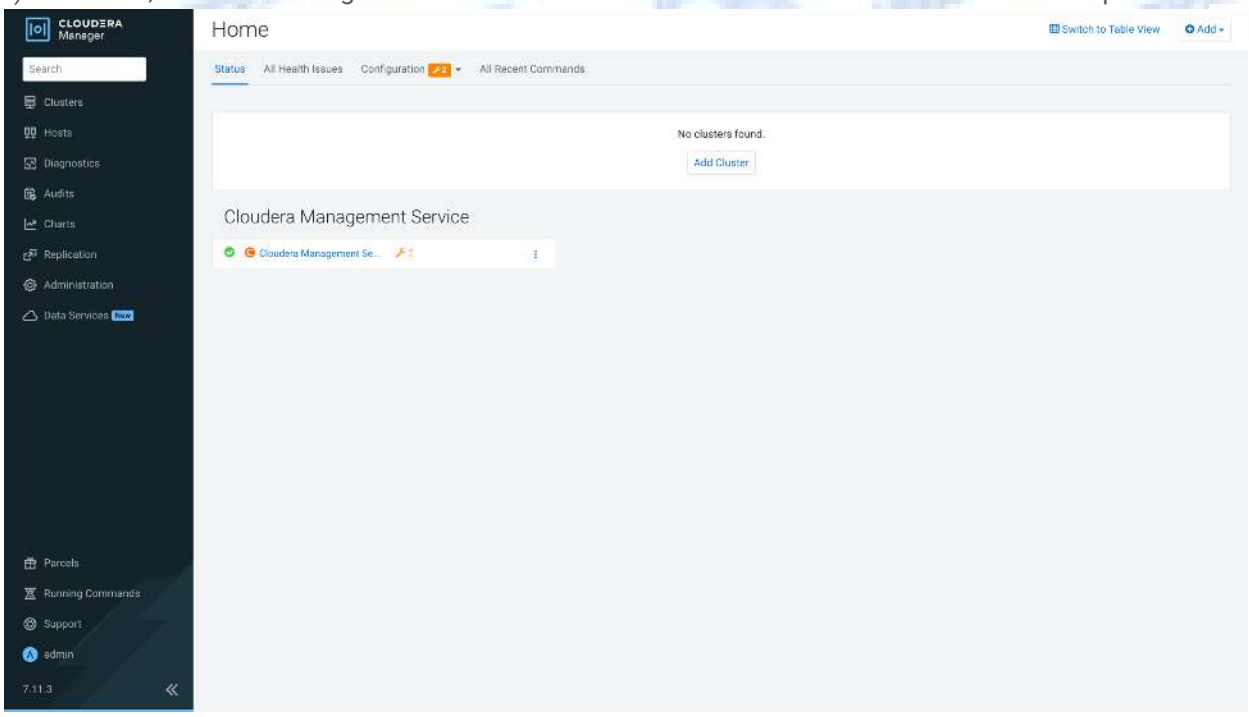
c) Setup Report Manager Database integration by providing the ***DBHostname***, ***DBNAME***, ***DBUser*** and ***DBPassword***. After entering the details, click on ***Test Connection.*** After the successful connection test, click ***Continue.***



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:

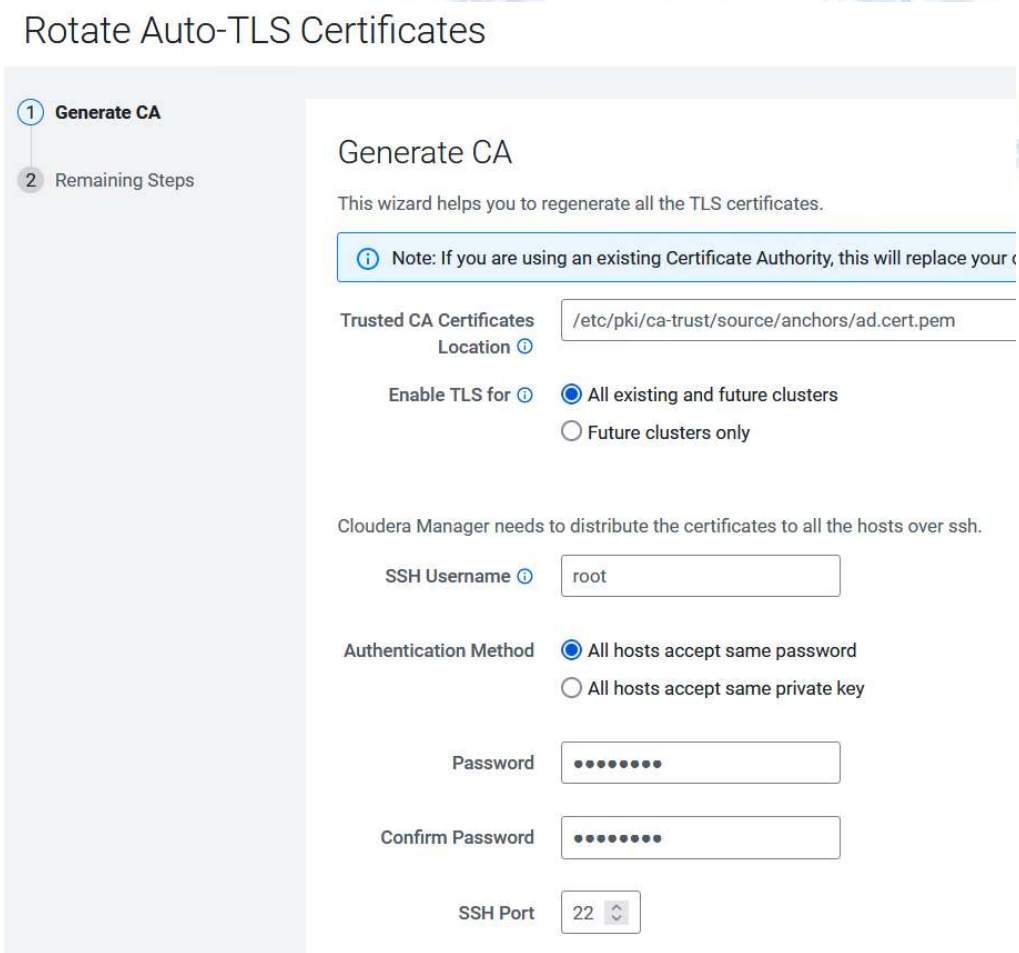
- Use a single Distinguished Name (DN) as a base for matching usernames in the directory, or
- Search filter options let you search for a particular user based on somewhat broader search criteria – for example Cloudera Manager users could be members of different groups or organizational units (OUs), 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
```

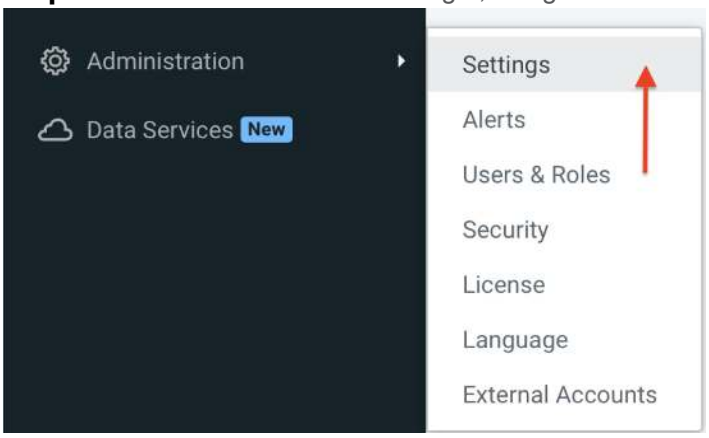
Step2. Update AutoTLS configuration by rotating Auto-TLS certificate with new CA certificate obtained. **(Only applied, in case you are going with an AD based setup).**



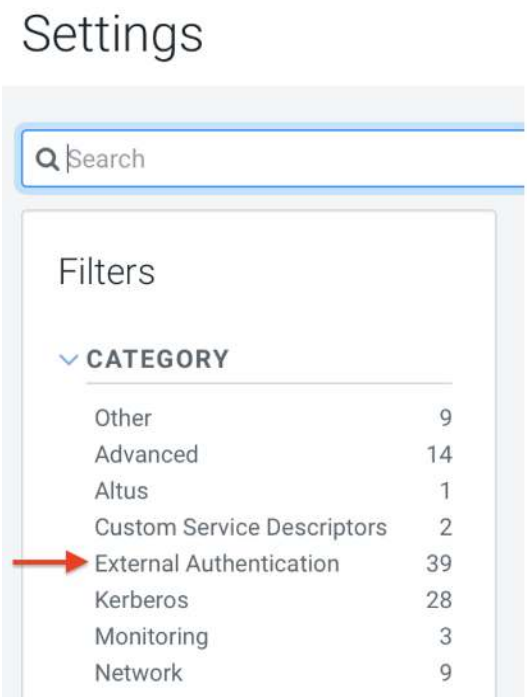
Step3. Restart clouder server configuration and restart cluster role/services and deploy client configuration.
(Only applied, in case you are going with AD based setup)

```
# 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:



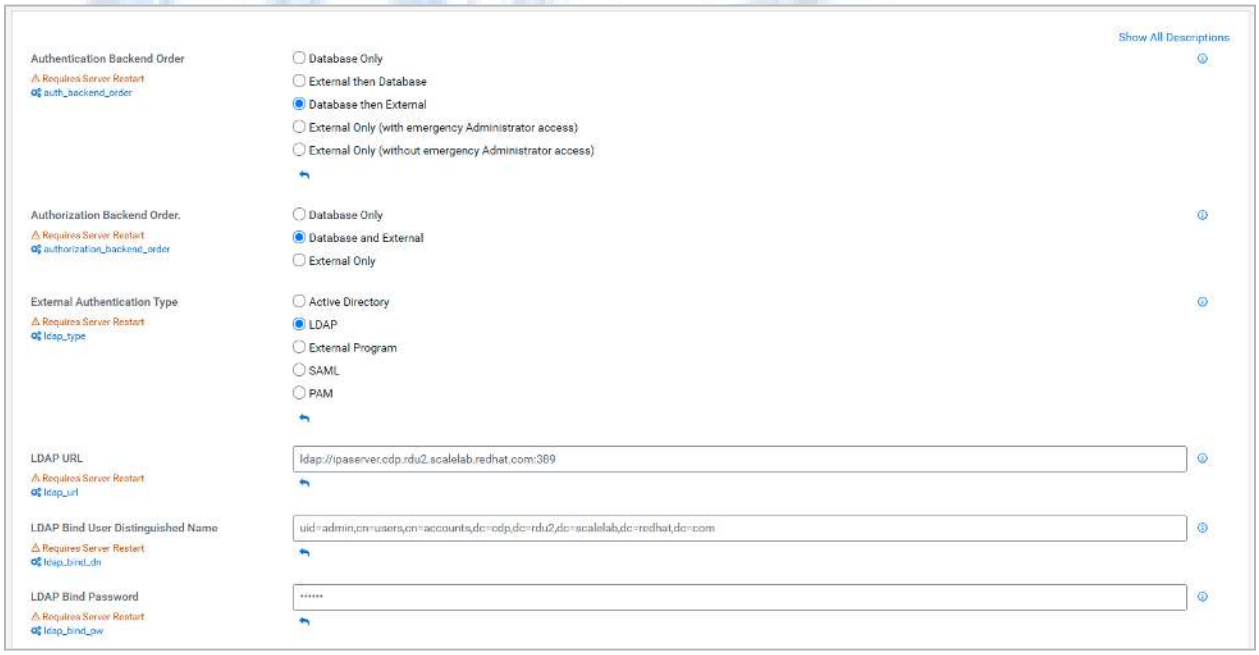
Step6. Search for “ldap” and enter values for ldap authentication, please refer to the sample values as mentioned in the below table and update according to your actual setup.

Table 7. LDAP Integration

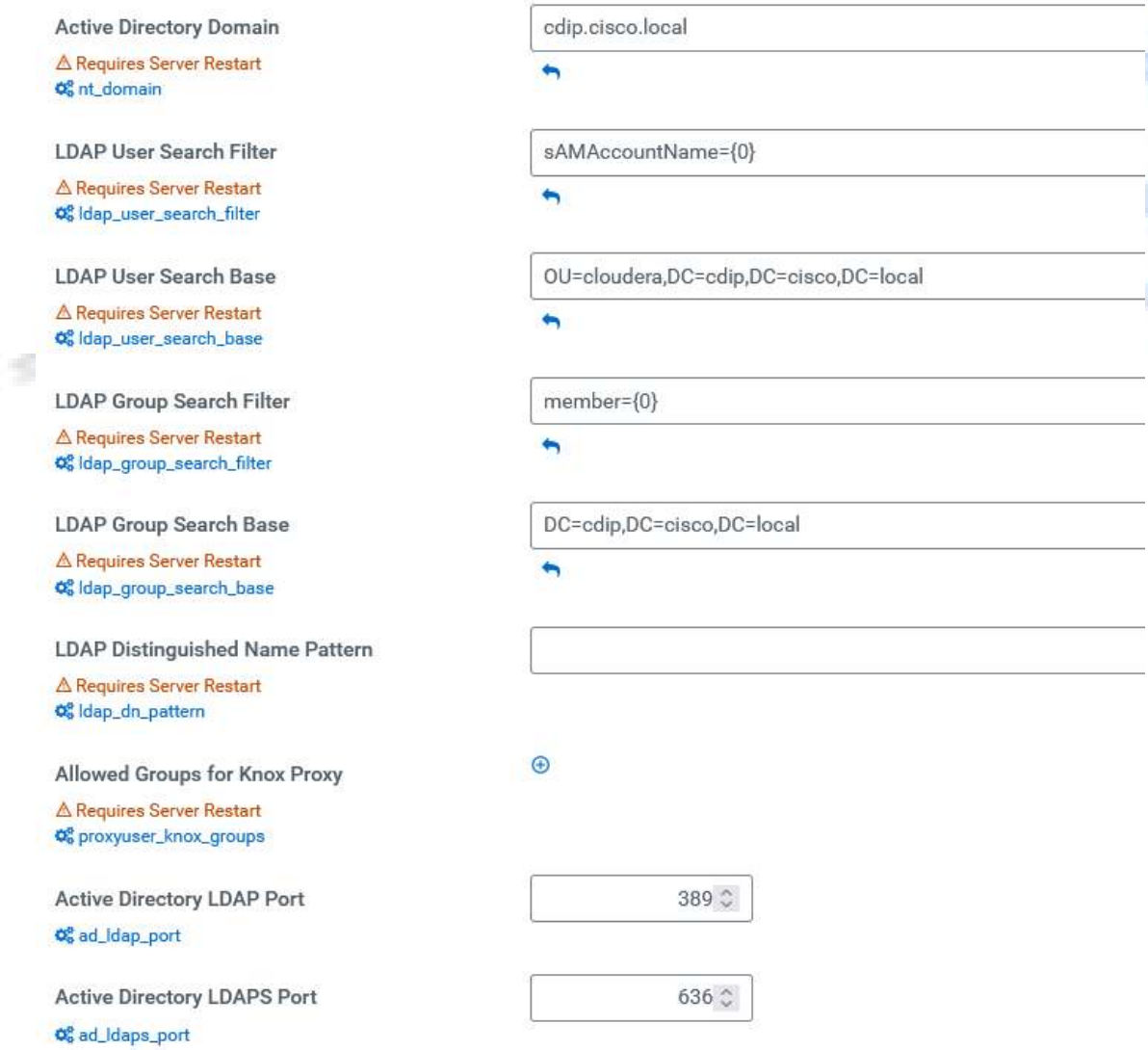
Component	Value
Authentication Backend Order:	Database then EXTERNAL

Component	Value
Authorization Backend Order:	Database and EXTERNAL
External Authentication Type:	LDAP
LDAP URL:	ldap://ipaserver.cdp.rdu2.scalelab.redhat.com:389/
LDAP Bind User Distinguished Name:	uid=admin,cn=users,cn=accounts,dc=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com
LDAP Bind Password:	*<redhat123>* (password for KDC admin, configured earlier)
Active Directory Domain: **(For AD Based LDAP)**	<AD DOMAIN> (e.g cdp.rdu2.scalelab.redhat.com)
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=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com
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=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com
LDAP DistName Pattern:	uid=admin,cn=users,cn=accounts,dc=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com

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



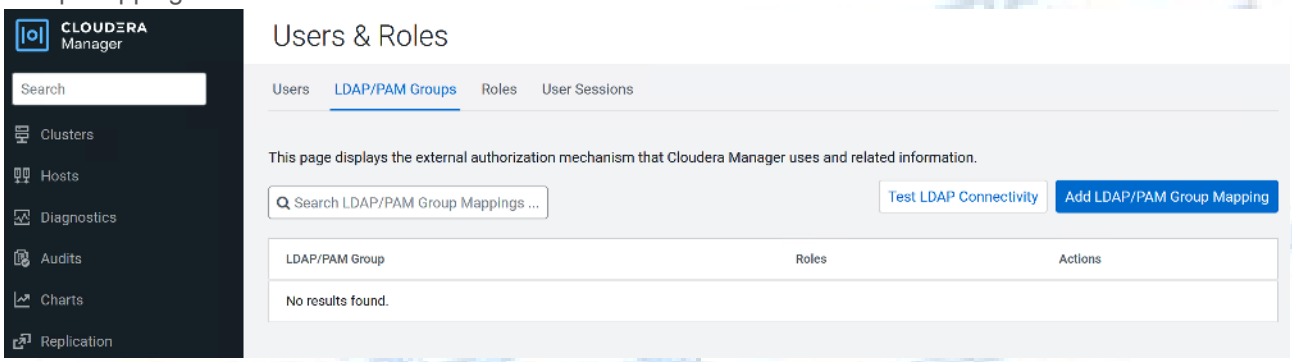
Additional Parameters for AD Based Setup:



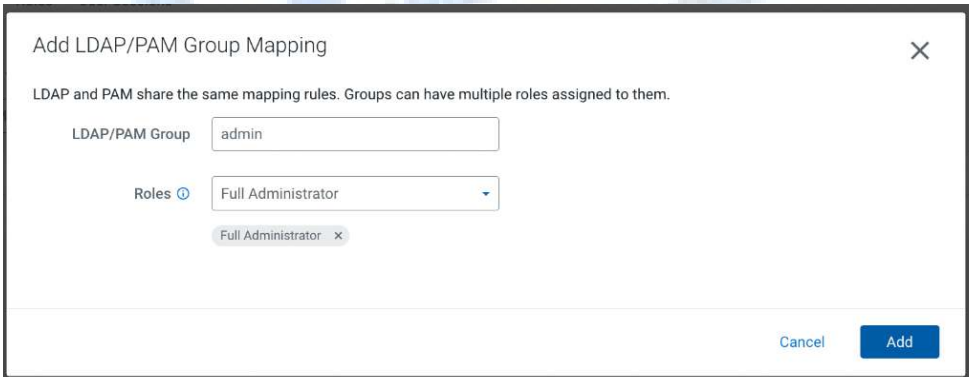
Step8. Click ***Save***. Once you click on the Save button, it will tell you to restart the CM Server, in order to bring the changes in effect. When you restart the CM Server from Backend. You will see the below entries in the logs.

```
[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
```

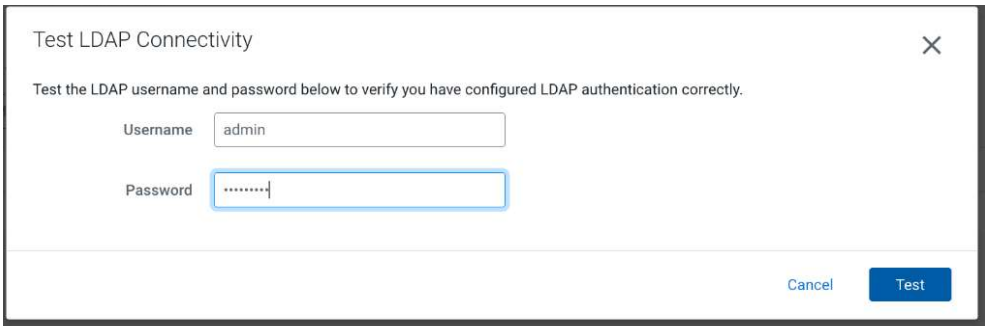
Step9. Login again to CM-UI and in ***Administration > Users & Roles > LDAP/PAM Groups***, add LDAP/PAM Group mapping.



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: <redhat123>)***

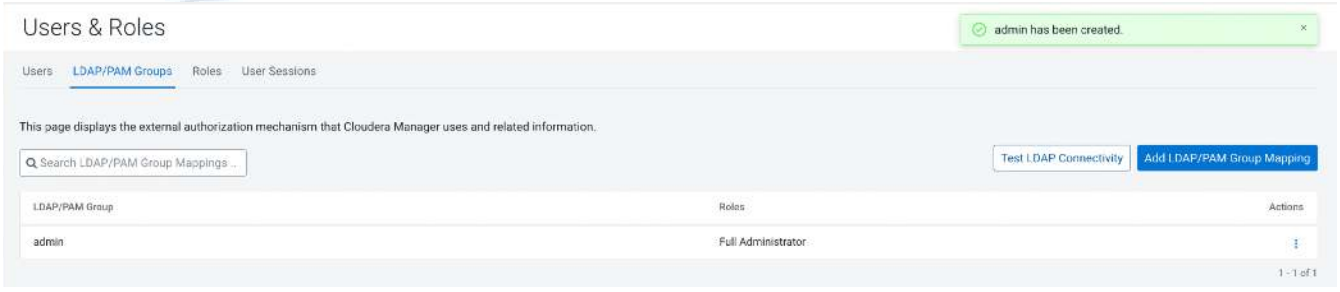


Step12. Click on ***Test*** to verify LDAP configuration is set up and working fine, as expected.

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.redhat.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)***



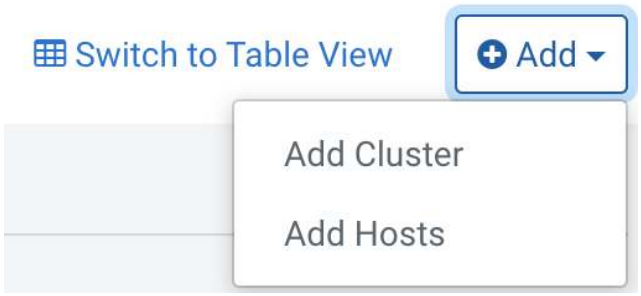
****

Setup Cloudera on premises (PvC) Base Cluster

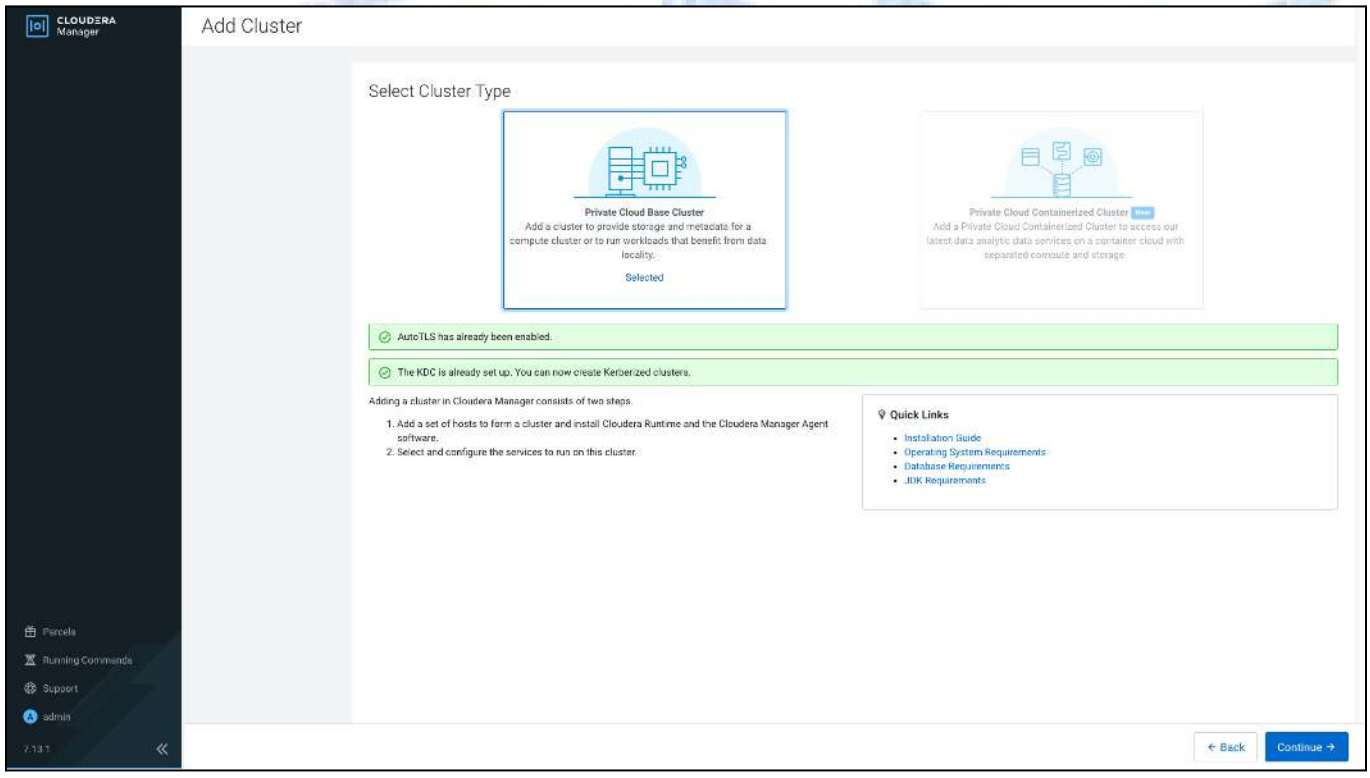
In this step, we will setup the Base cluster which will serve as the DataLake for the CDP Data Services that need the SDK capabilities for the cluster wide features like lineage, governance, security etc..,

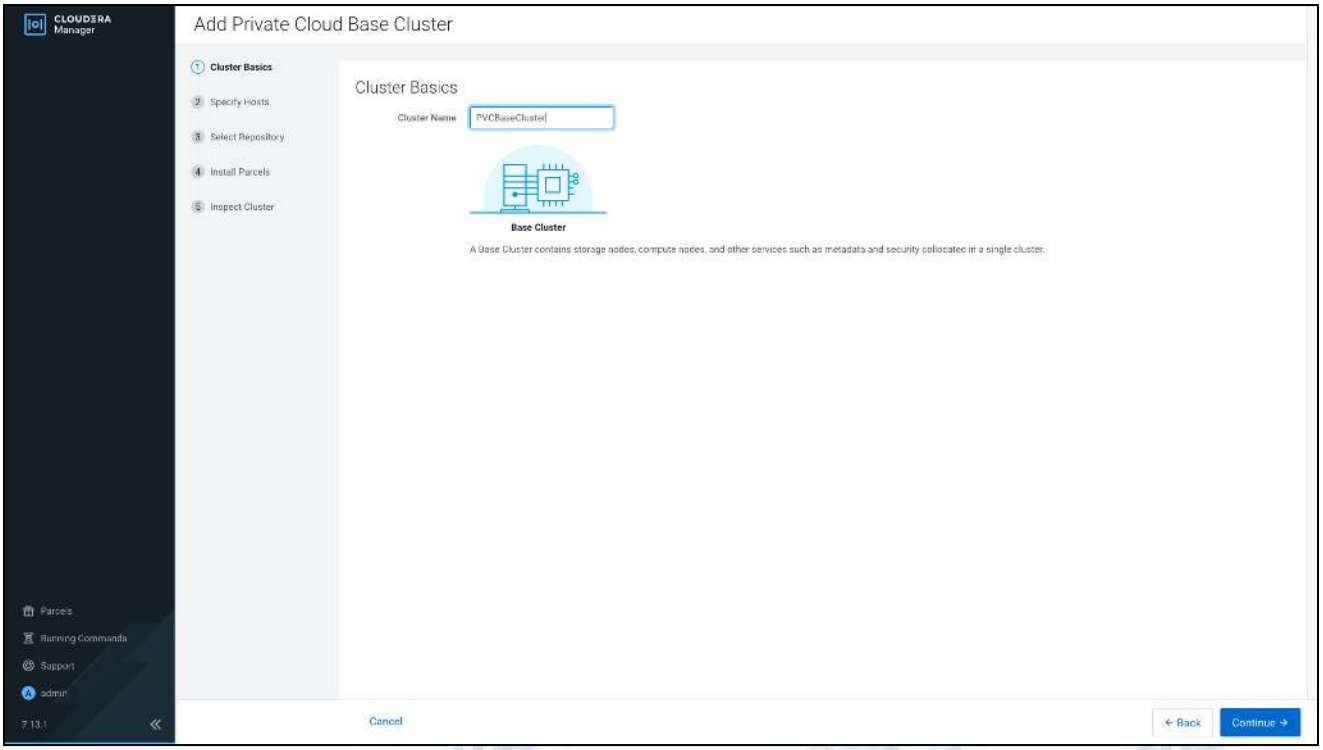
Procedure 8. Install Cloudera on premises Base using the Cloudera Manager WebUI

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***.



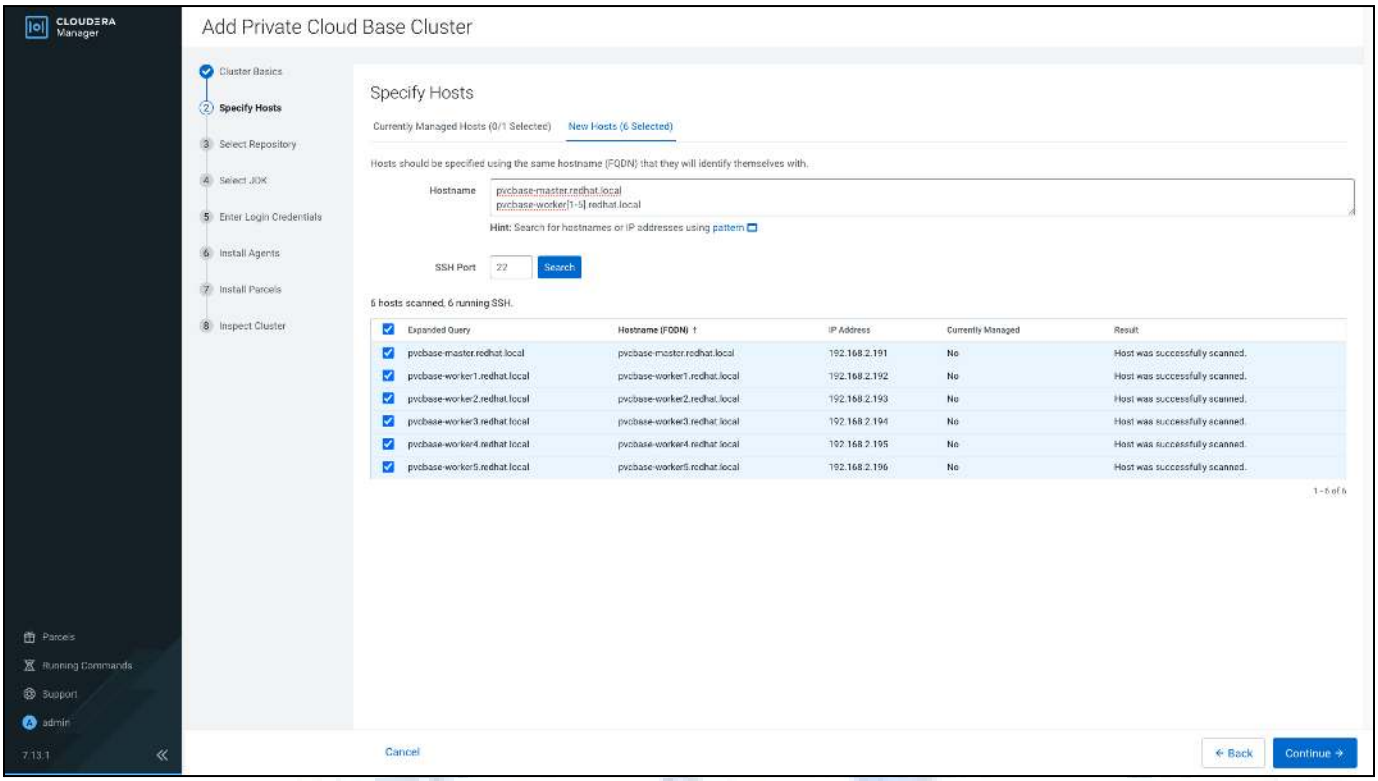


Step 3. Enter the cluster host names or IP addresses in the Hostnames field. You can Provide the host pattern pvcbase-master, pvcbase-worker[1-3] or pvcbase-worker[1-3].redhat.local etc separated with a new line and Click on ***Search***.

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.

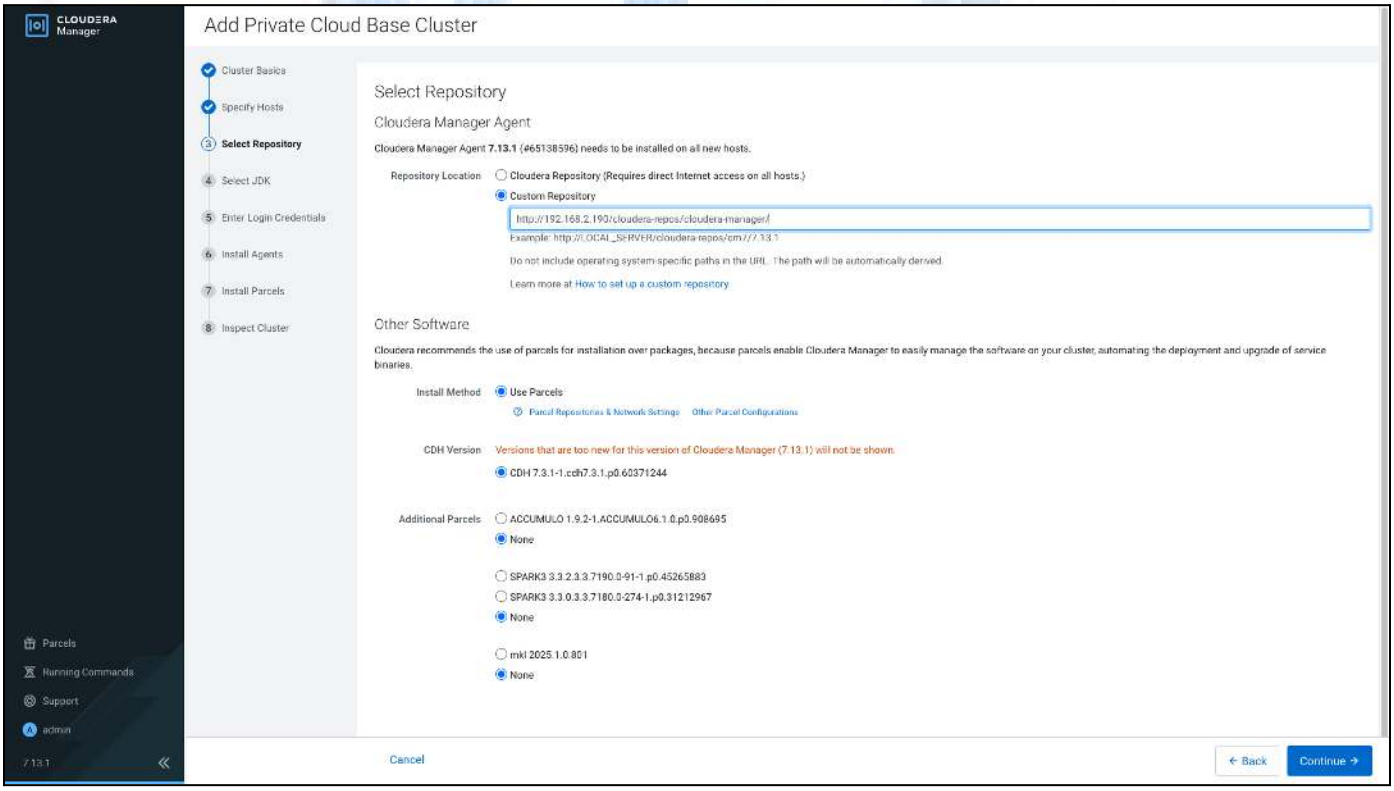
Step 4. Specify the hosts that are part of the cluster using their IP addresses or hostname. The figure below shows a pattern that specifies the IP addresses range. Cloudera Manager will "discover" the hosts based on matching the pattern provided by you to add in the cluster. Verify that all desired 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.redhat.local
pvcbase-worker[1-3].redhat.local
```

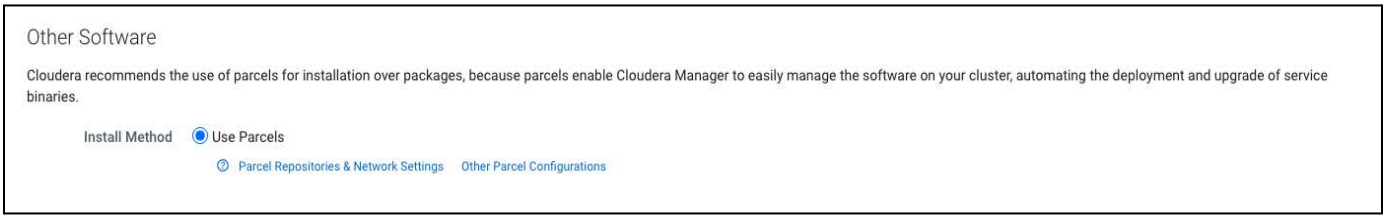


Step 5. The Select Repository section appears. Select Cloudera Repository option as mentioned. Enter ***Custom Repository*** or Cloudera Repository to install Cloudera Manager Agent on all nodes in the cluster. We have earlier configured the private yum repository on ***cldr-mngr*** node. Please provide the path here:

i.e. http://13.251.65.11/cloudera-repos/cloudera-manager/ and click on ***Continue***.

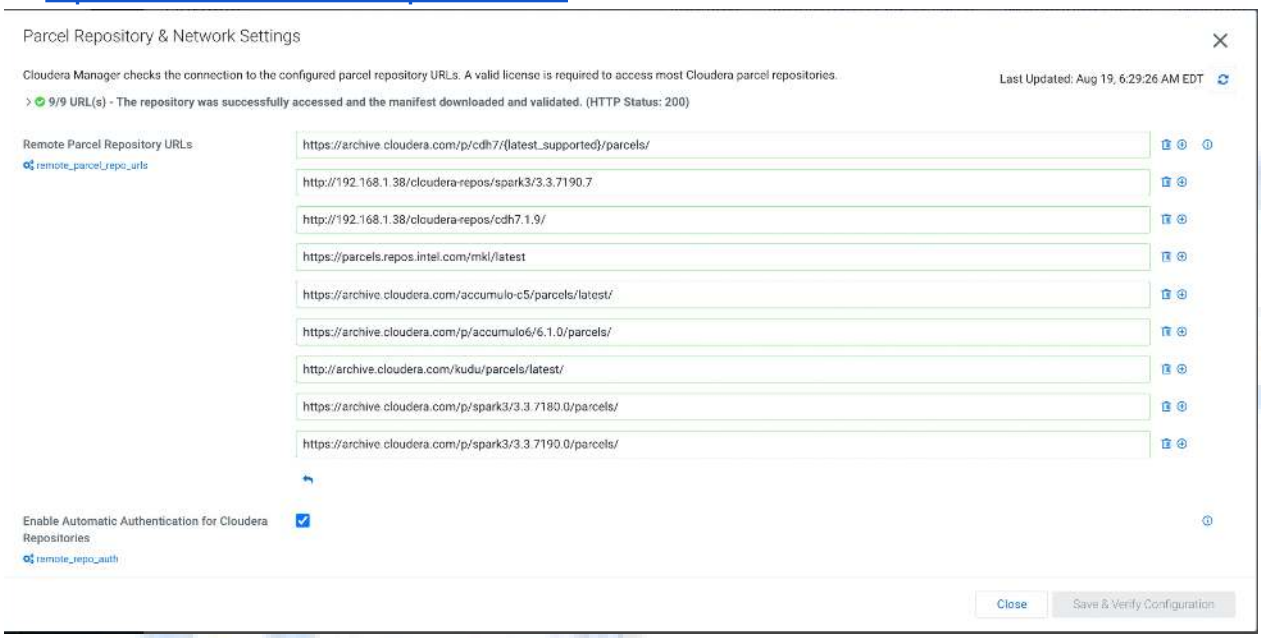


Step 6. In the other software section, select ***Use Parcels (Recommended)*** and click ***Parcel Repository & Network Settings*** to provide a custom Parcels location to be installed ***(in a new tab in the same browser window)***.

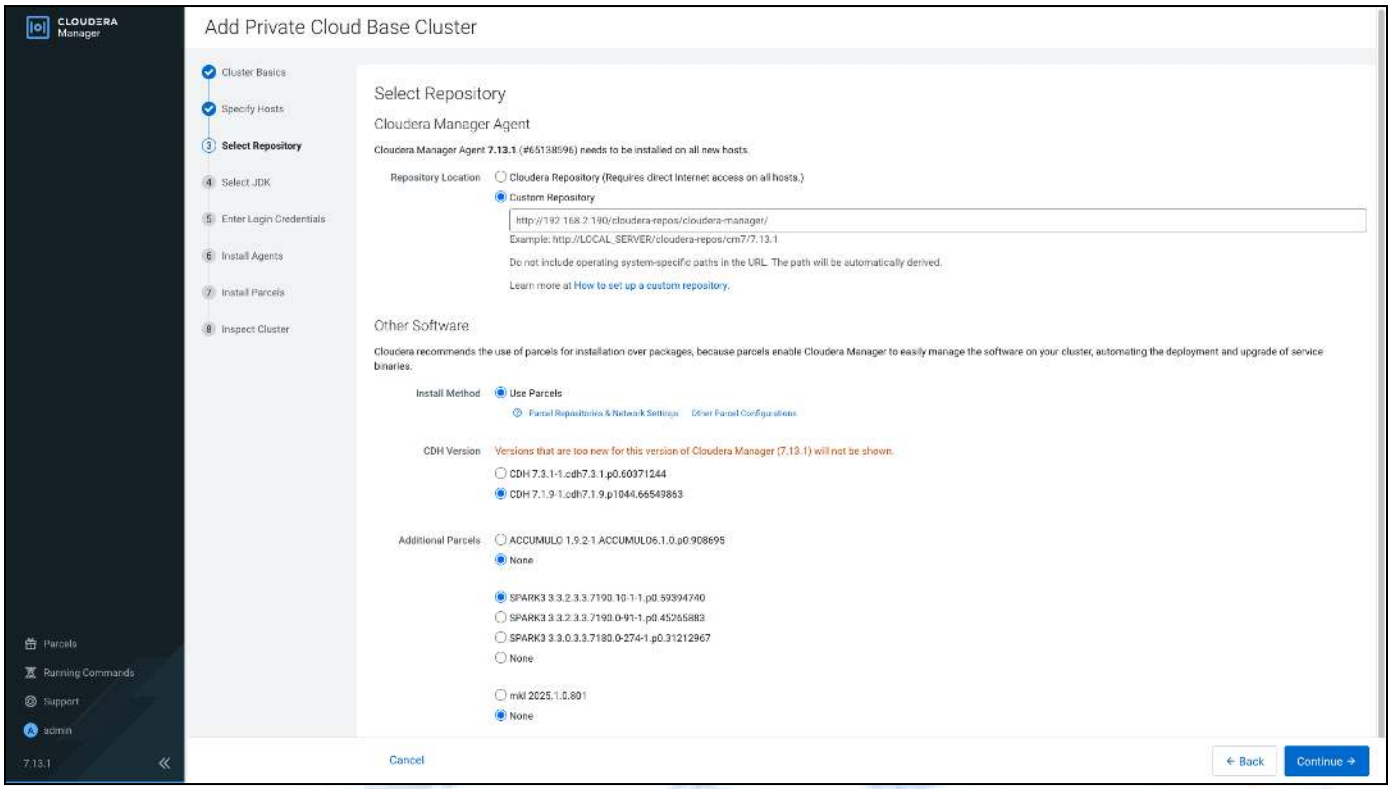


Step 7. Enter custom repository URL for CDH7 and CDS 3.3 parcels. Click on ***Save and Verify Configuration***. Close the Parcel Repository & Network Settings wizard.

i.e. http://13.251.65.11/cloudera-repos/spark3/3.3.7191000.4/
http://13.251.65.11/cloudera-repos/cdh7.3.1/

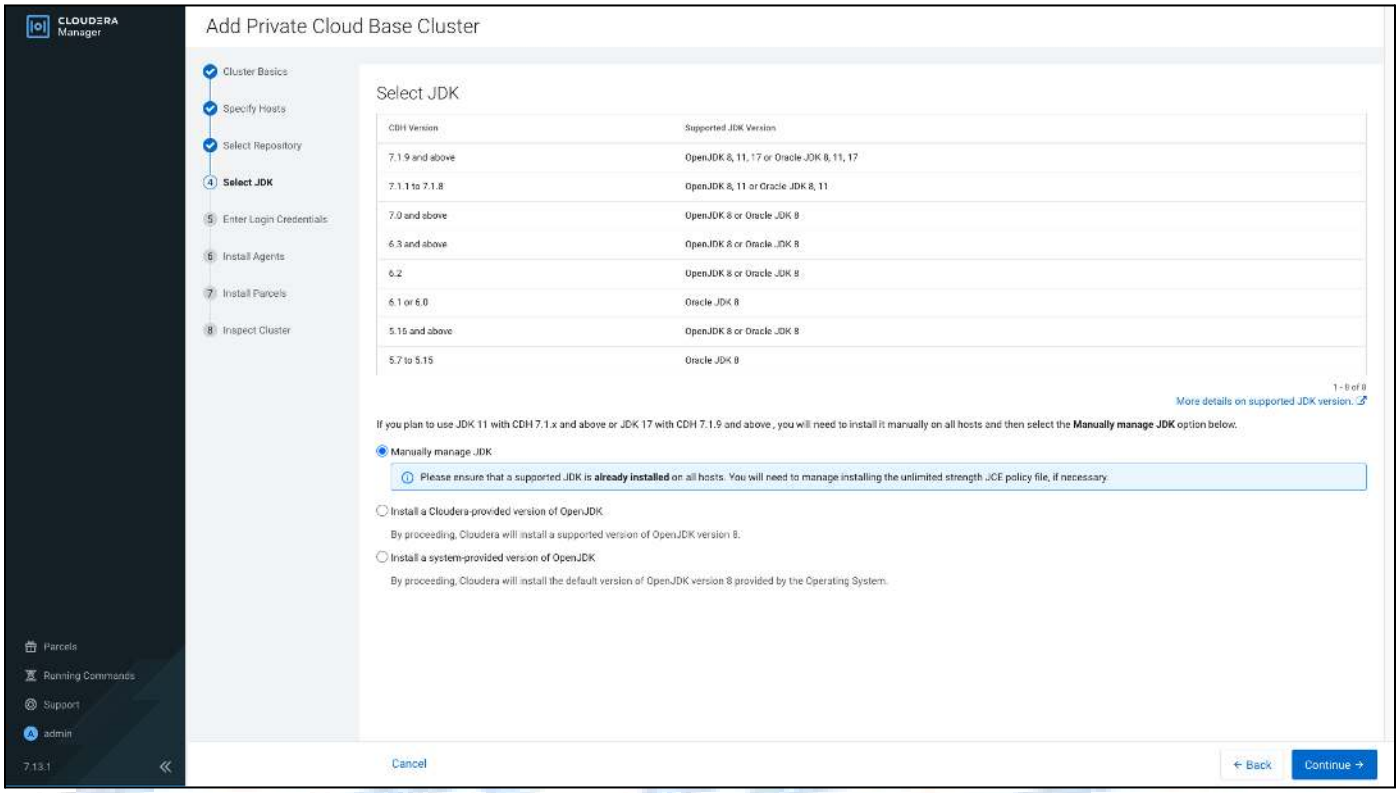


Step 8. Select the parcels for installation.



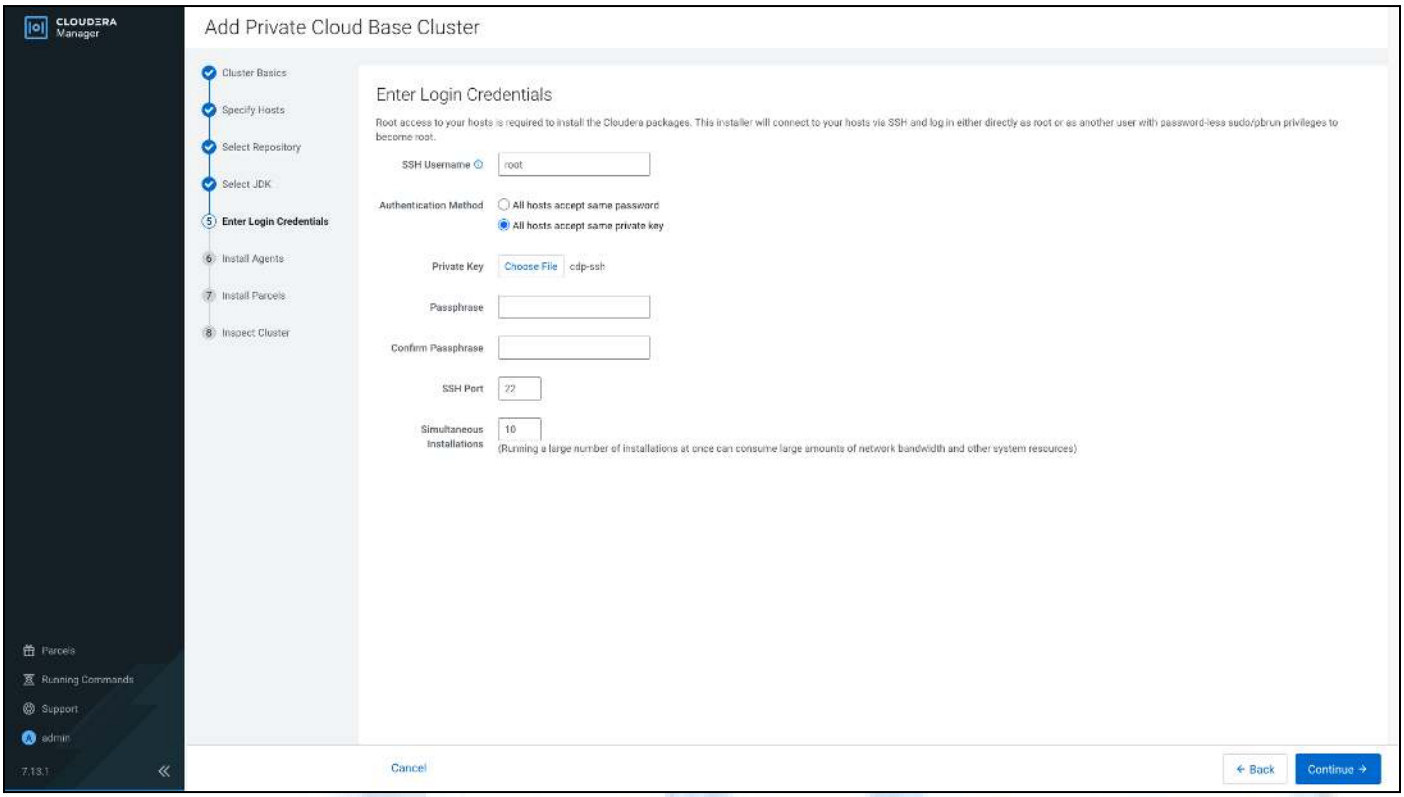
Step 9. Click *Continue*.

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***.

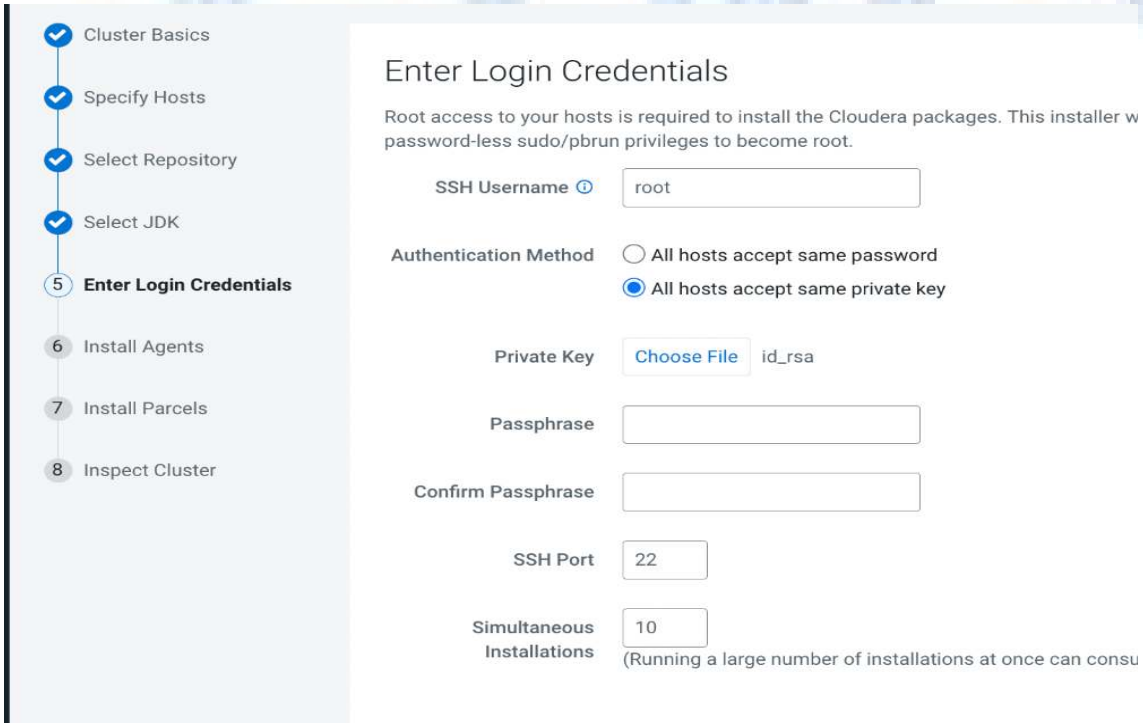


Step 11. The ***Enter Login Credentials section*** appears. On this page, provide the required ***details*** as mentioned ***in the below table***, for the hosts to install Cloudera packages. Click ***Continue***.

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. redhat123)***
Confirm Password (If selected password based auth)	<password_for_vm_root_user> ***(e.g. redhat123) (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

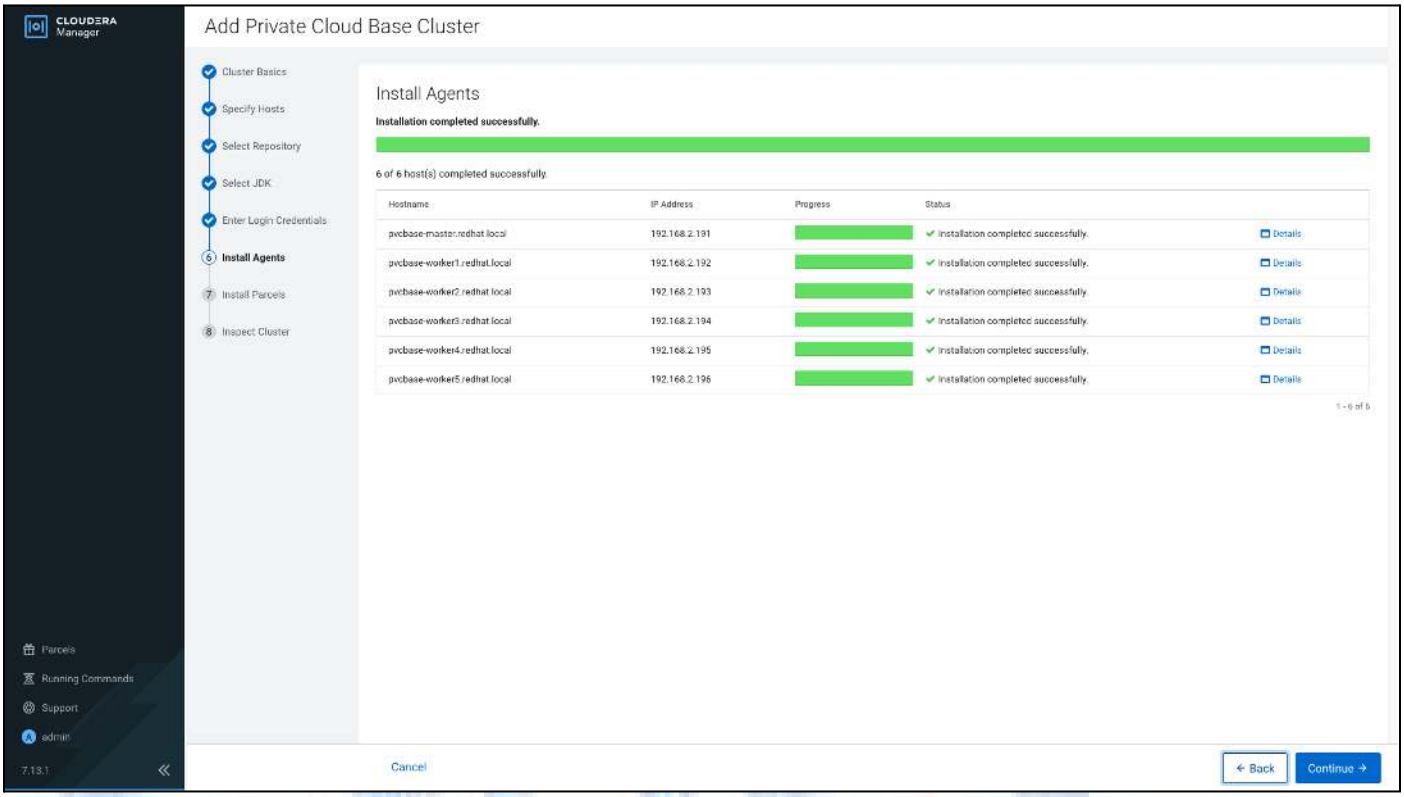


a. If Selected the ***Authentication method*** as ***All hosts accept same password***



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

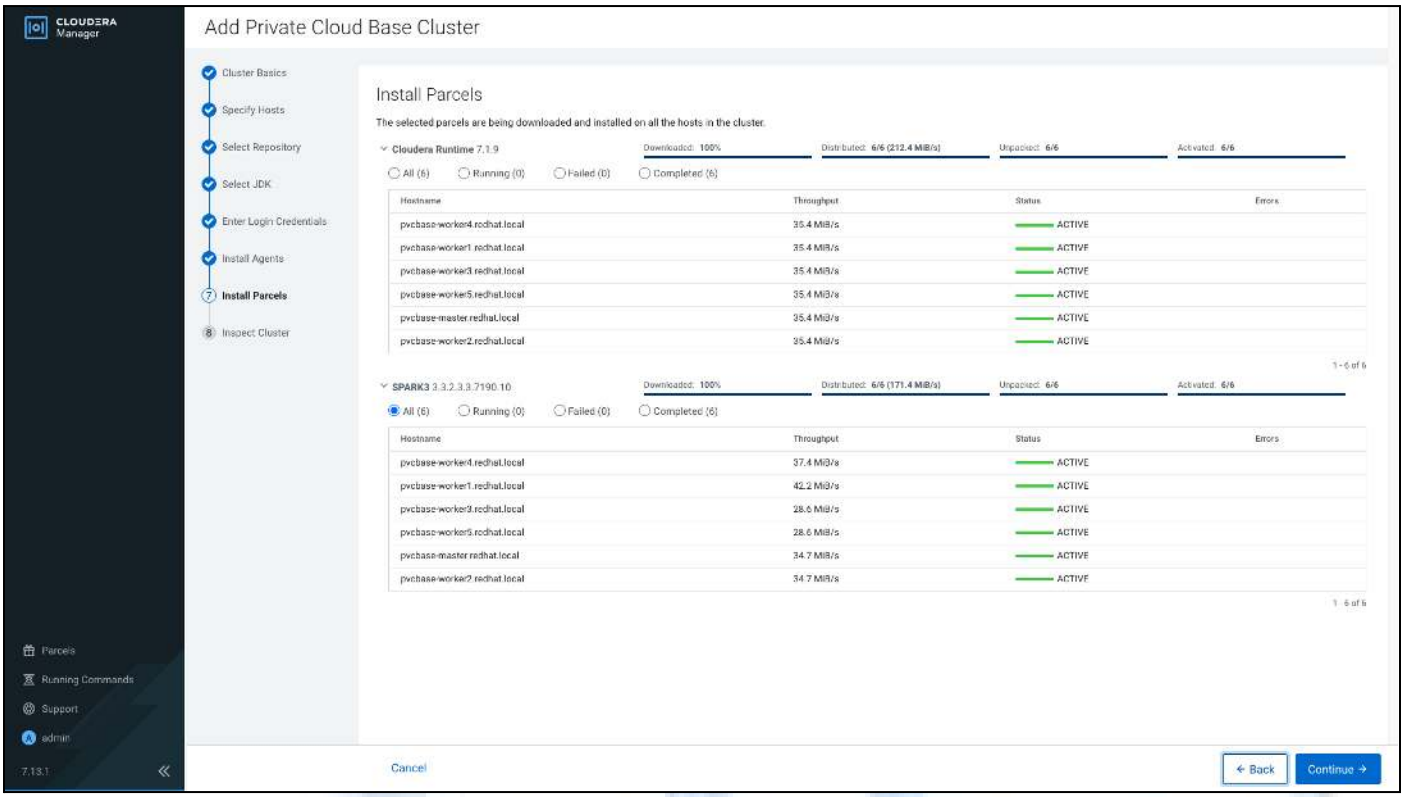
Step 12. The *Install Agents* section appears showing the progress of the installation. It will check for and install the *JDK* (If not already there) and *cloudera-scm-agent* software on all the Base Cluster nodes. Click *Continue* after the Cloudera Agent *Installation completed successfully* on all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

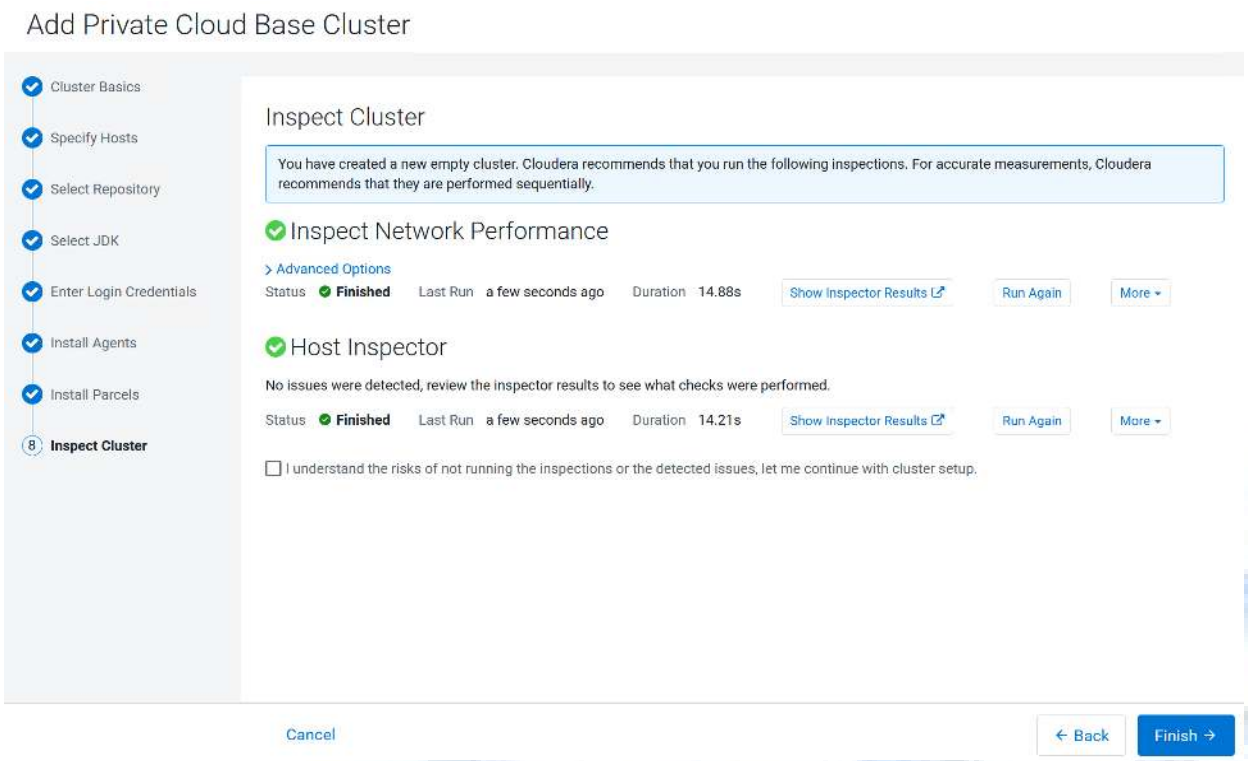
```
/var/lib/hadoop-hdfs
drwxr-xr-x [owner: hdfs, group: hadoop]

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***.



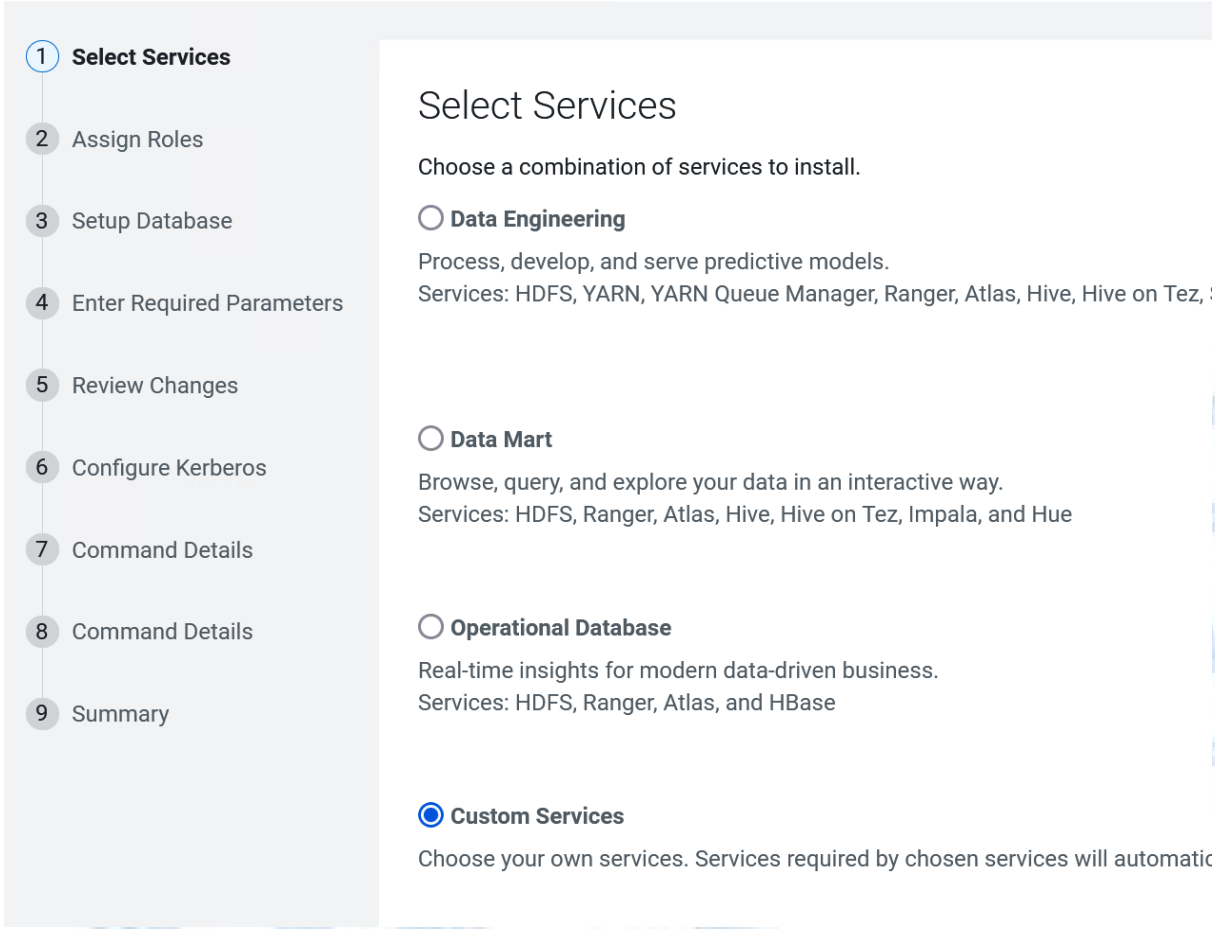

```
If java path mismatch error occurs in network/host inspection:
Take - Java Path from the host ( /usr/lib/jvm/java-17-openjdk-17.0.13.0.11-4.el9.x86_64/ ), goto CM,
open a new tab of CM-UI on browser and search for JAVA PATH in search bar present at left hand side:
# Override JAVA PATH in CM-UI> Search 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.



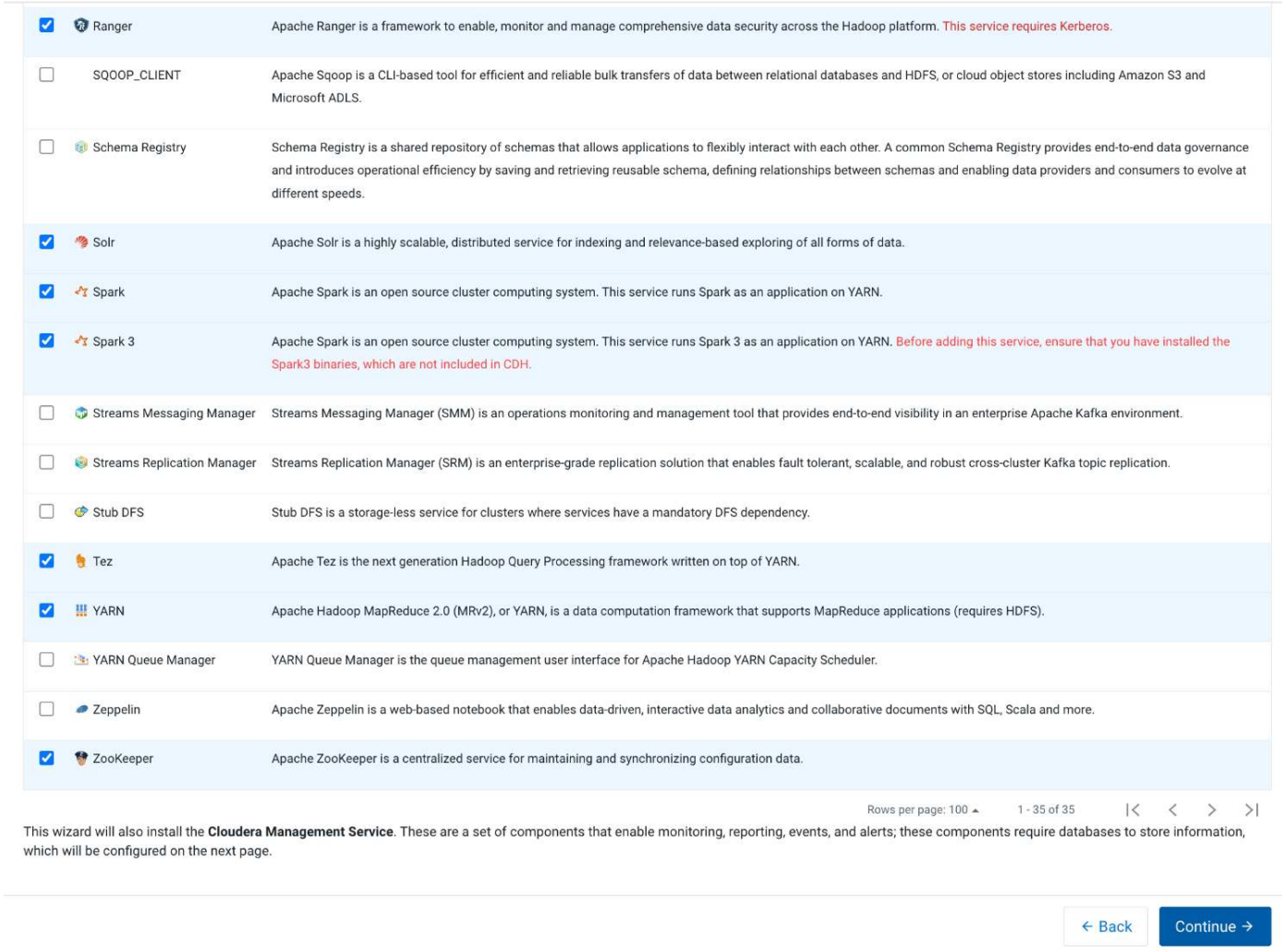
Note: It is important to select host(s) to deploy services based on the role intended for it. For detailed information, go to: [Runtime Cluster Hosts and Role Assignments]

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[1-3]
YARN	**ResourceManager** : pvcbase-master **NodeManager** : pvcbase-worker[1-3] **(Same as DataNode)** **JobHistoryServer** : pvcbase-master
Core Configuration	**Gateway** : pvcbase-master
Iceberg Replication	**Gateway** : pvcbase-master

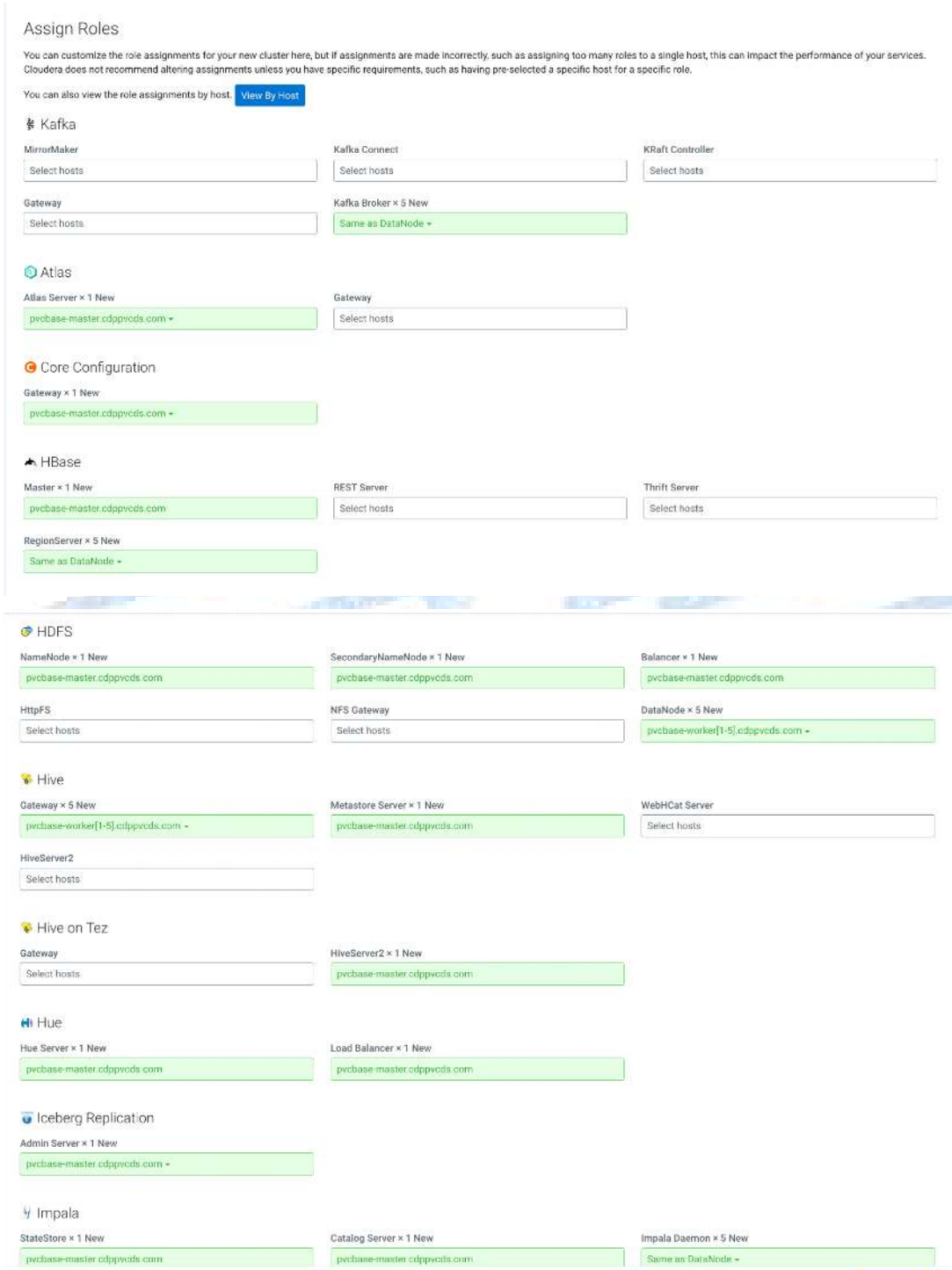
Service Name (Role Instance)	Host
Knox	**Gateway** : pvcbase-master

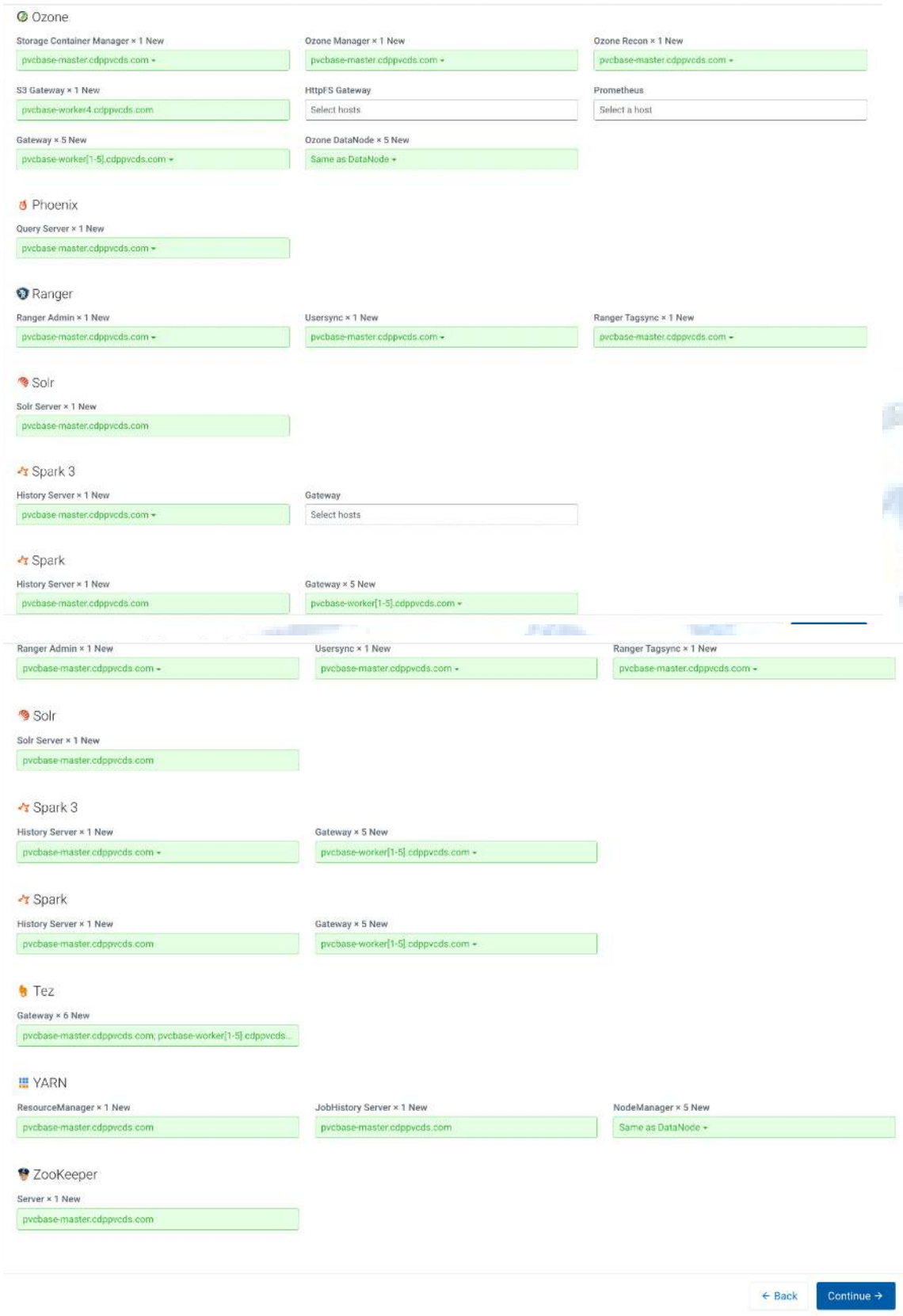
Cloudera Management Service
 	
This is already configured in previous steps
	`Service Monitor : cldr-utility`
	`Host Monitor : cldr-utility`
	`Reports Manager : cldr-utility`
	`Event Server : cldr-utility`
	`Alert Publisher : cldr-utility`

Spark2 (i.e. SparkOnYARN)/ Spark3	**Spark History Server** : pvcbase-master **Spark Gateway** : pvcbase-worker[1-3]
Hive	**Hive Metastore Server (HMS)** : pvcbase-master **Gateway** : pvcbase-worker[1-3]
Tez	**Gateway** : pvcbase-worker[1-3]
Hive on Tez	**HiveServer2** : pvcbase-master **Gateway** : pvcbase-worker[1-3]
Impala	**Impala Catalog Server** : pvcbase-master **Impala State Store** : pvcbase-master **Impala Daemon** : pvcbase-worker[1-3] **(Same as DataNode)**
HUE	**HUE Server** : pvcbase-master **LoadBalancer** : pvcbase-master
HBase	**HBase Master** : pvcbase-master **RegionServer** : pvcbase-worker[1-3] **(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[1-3] **OzoneDataNode** : pvcbase-worker[1-3] **(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** : pvcbase-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.

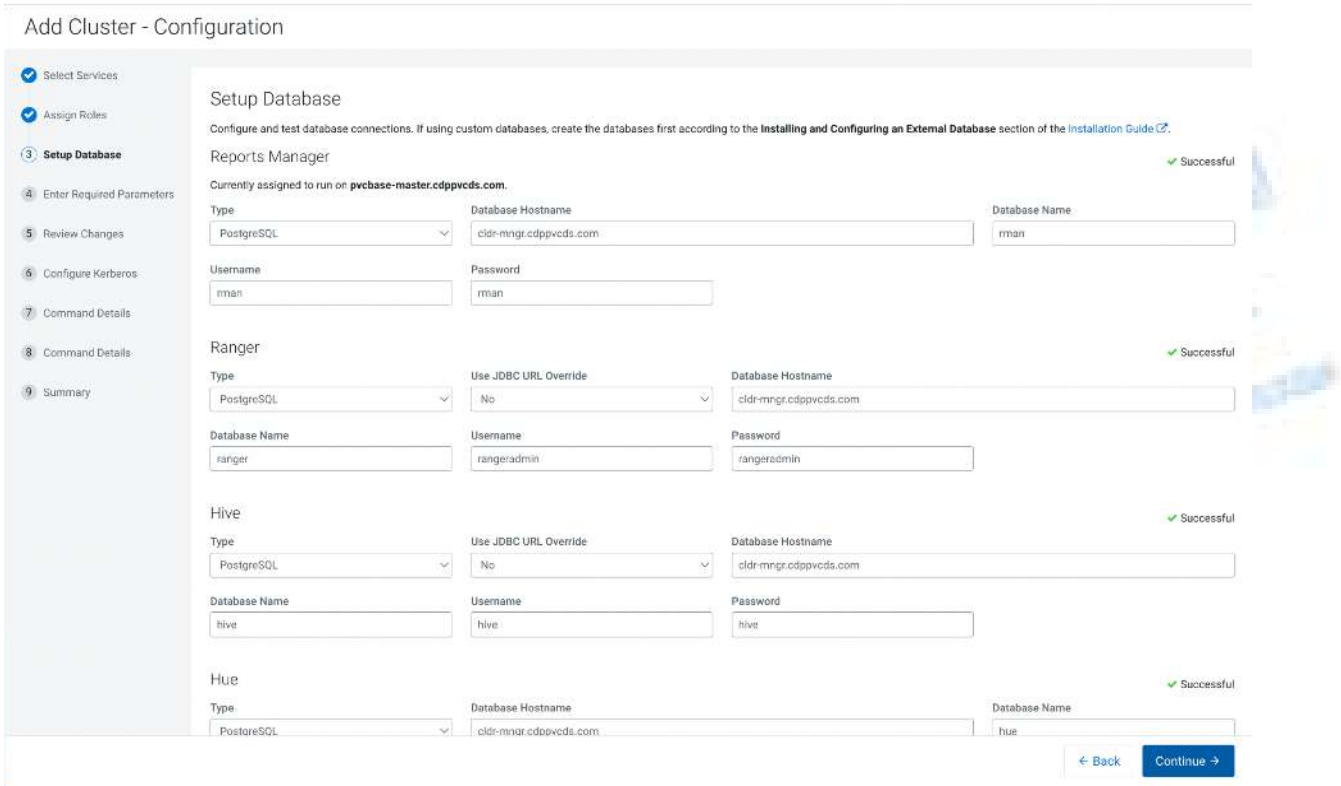




Step 23. Click ***Continue***. When you're doing the set-up for the first time on this CM Sever. ***Cloudera Management Services*** will be installed only once along with other control plane services.

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.

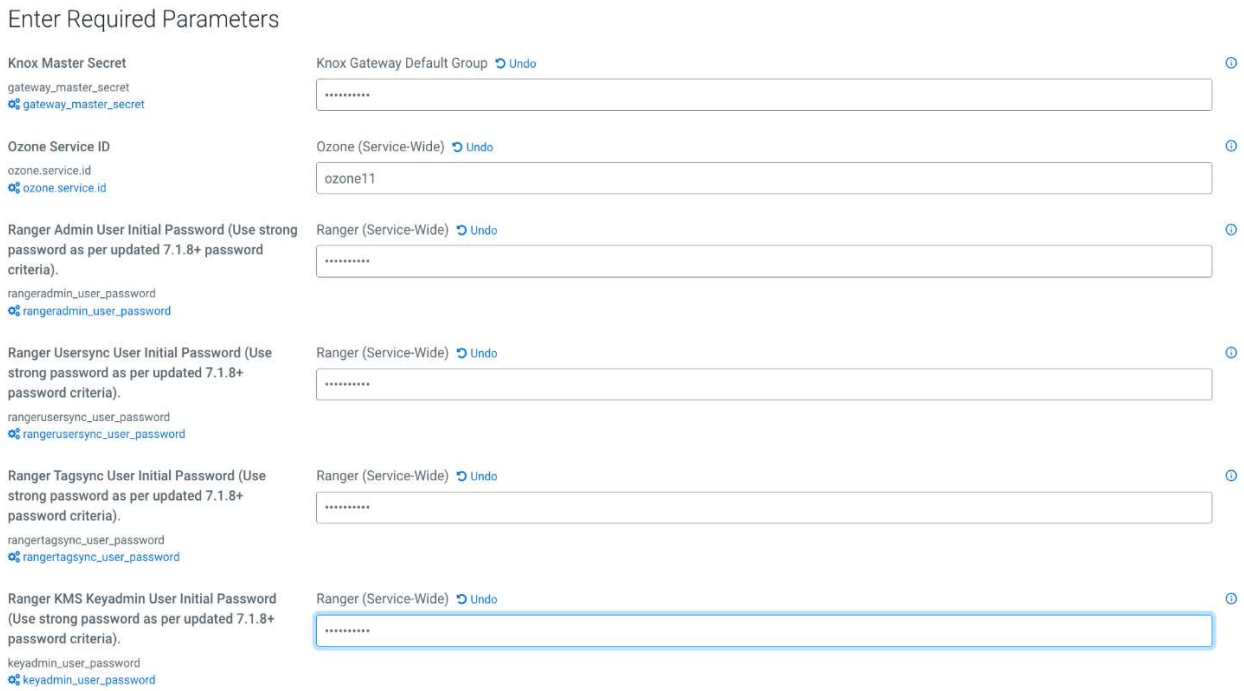
- Reports Manager (For Cloudera Management Service)
- Oozie Server (If selected to install, as optional)
- Ranger
- Hive
- YARN Queue Manager
- Hue



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:

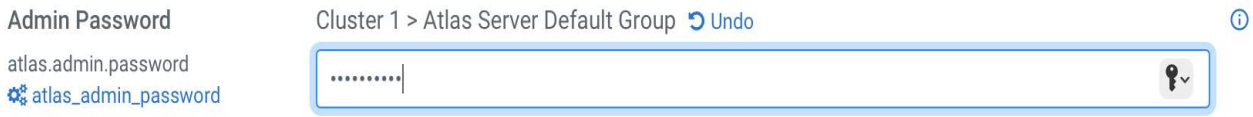
- *Knox Master Secret:* ***RedHat@123***
- *Knox IDBroker Master Secret:* ***RedHat@123***
- *Enter a suitable name for Ozone Service ID. i.e.:* ***ozone11***
- Ranger Admin: ***RedHat@123***
- Ranger Usersync: ***RedHat@123***
- Ranger Tagsync: ***RedHat@123***
- Ranger KMS Keyadmin: ***RedHat@123***



Step 27. Click ***Continue***.

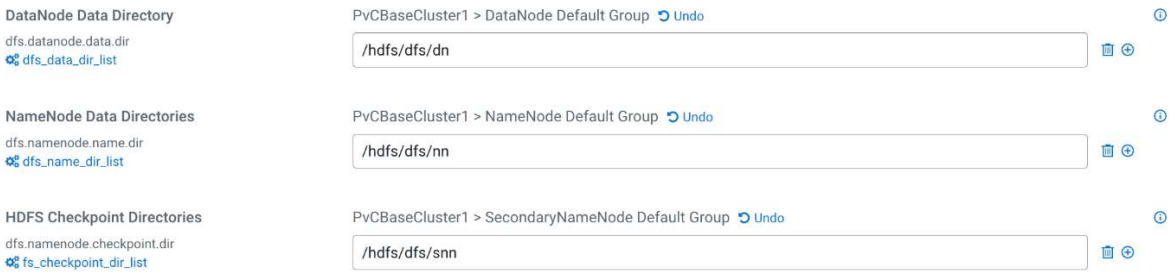
Step 28. The ***Review Changes*** page appears. Set a ***password*** for ***Atlas***. You can set it to the same value set for Ranger above.

- Atlas Admin password: ***RedHat@123***



Step 29. Scroll down and verify/update the HDFS disks configuration according to the below.

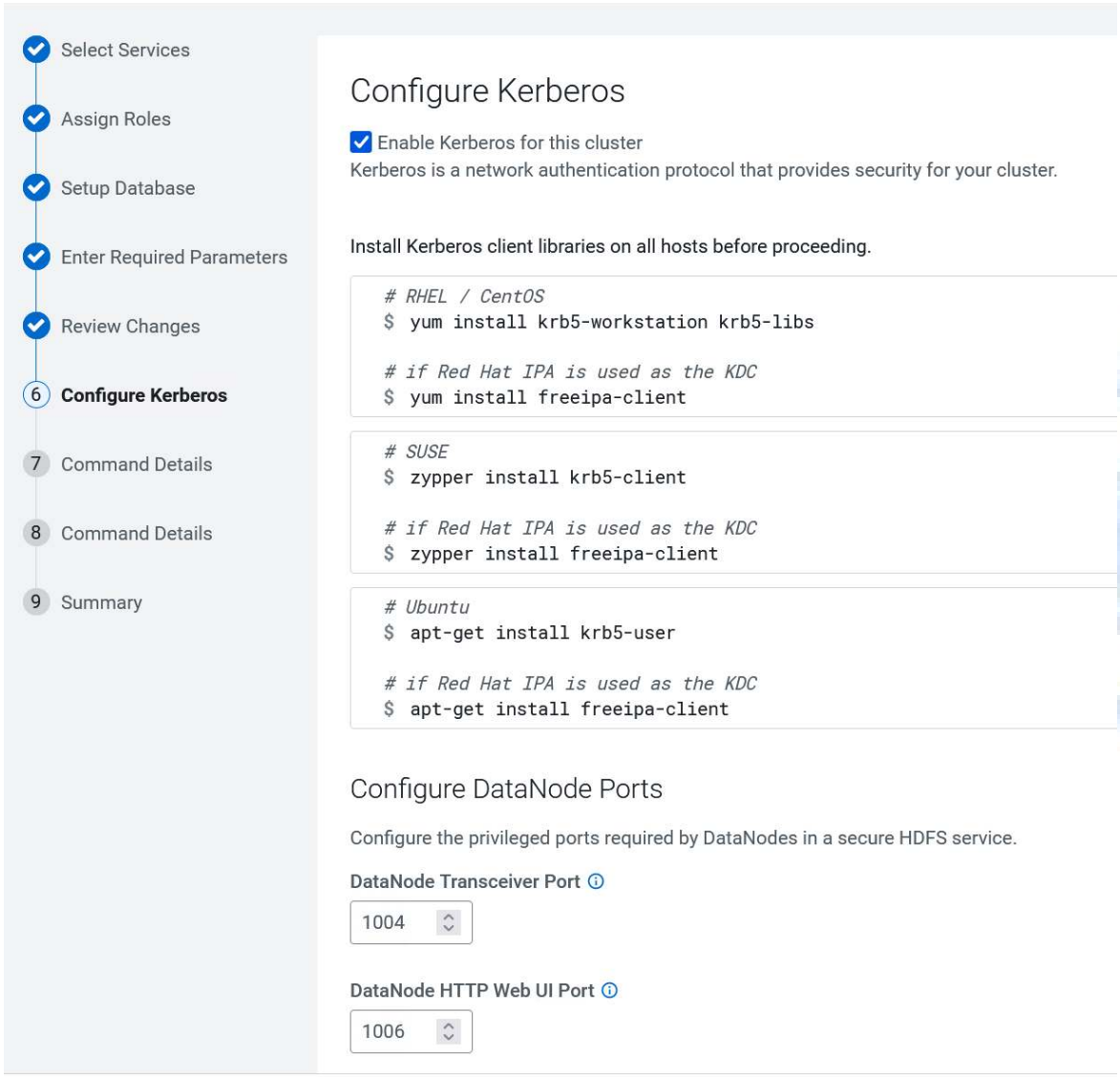
- DataNode Data Directory: ***/hdfs/dfs/dn***
- NameNode Data Directories: ***/hdfs/dfs/nn***
- HDFS Checkpoint Directories: ***/hdfs/dfs/snn***



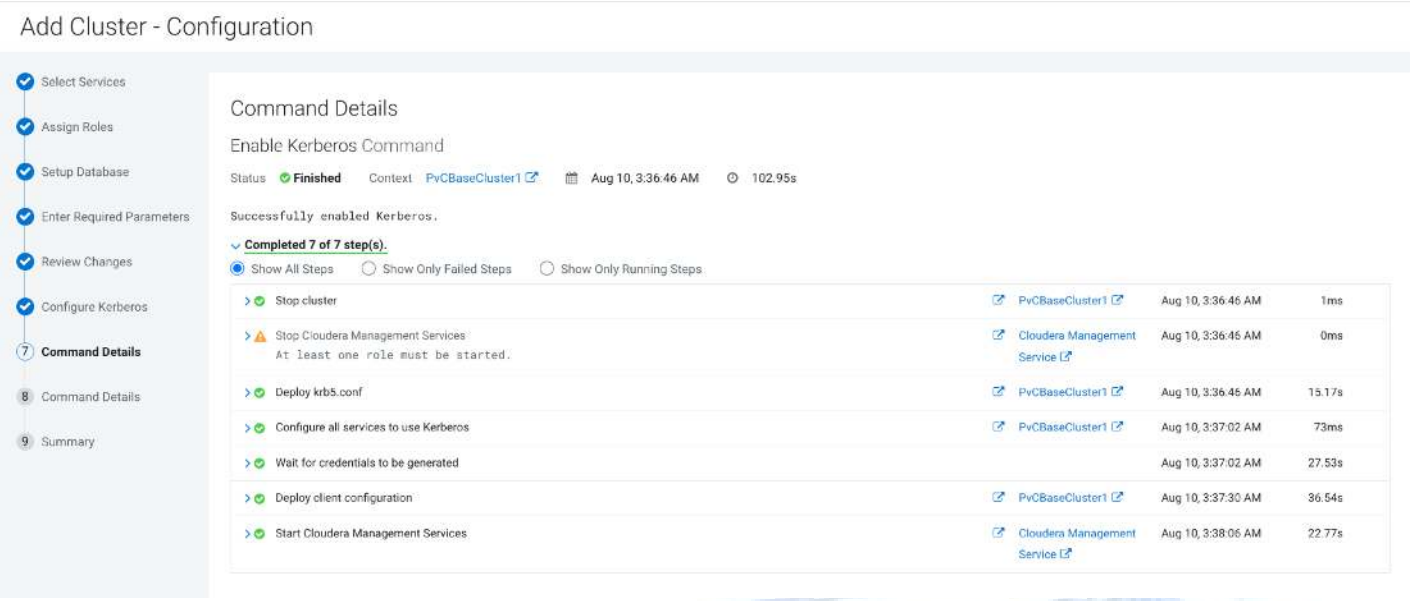
Step 30. Review the changes for all the services on the ***Review Changes*** page and verify/edit the configuration parameters as per your requirements. Click ***Continue***.

Step 31. A few sets of commands are running in the background. Wait for them to get executed successfully. Once done, Click ***Continue***.

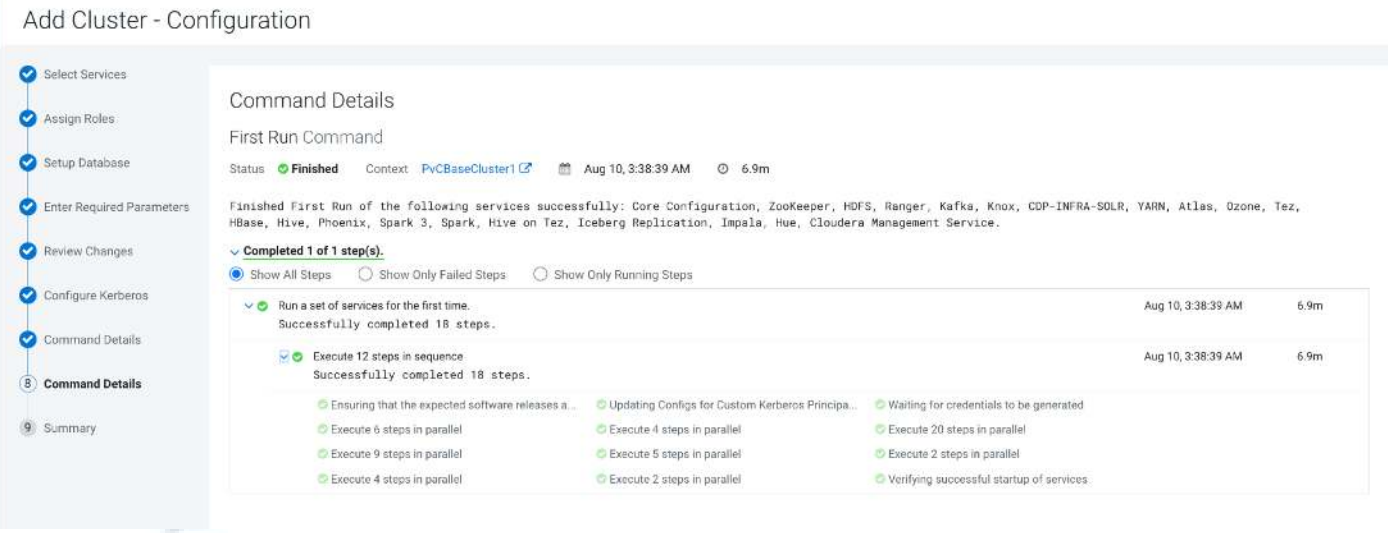
Step 32. Configure Kerberos and Keep Review and customize the configuration changes based on your requirements. Check the box for ***Enable Kerberos for this cluster***. **Required libraries i.e. krb5-workstation, krb5-libs and freeipa-client are already installed on all servers in prior steps.**



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.redhat.local@redhat.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

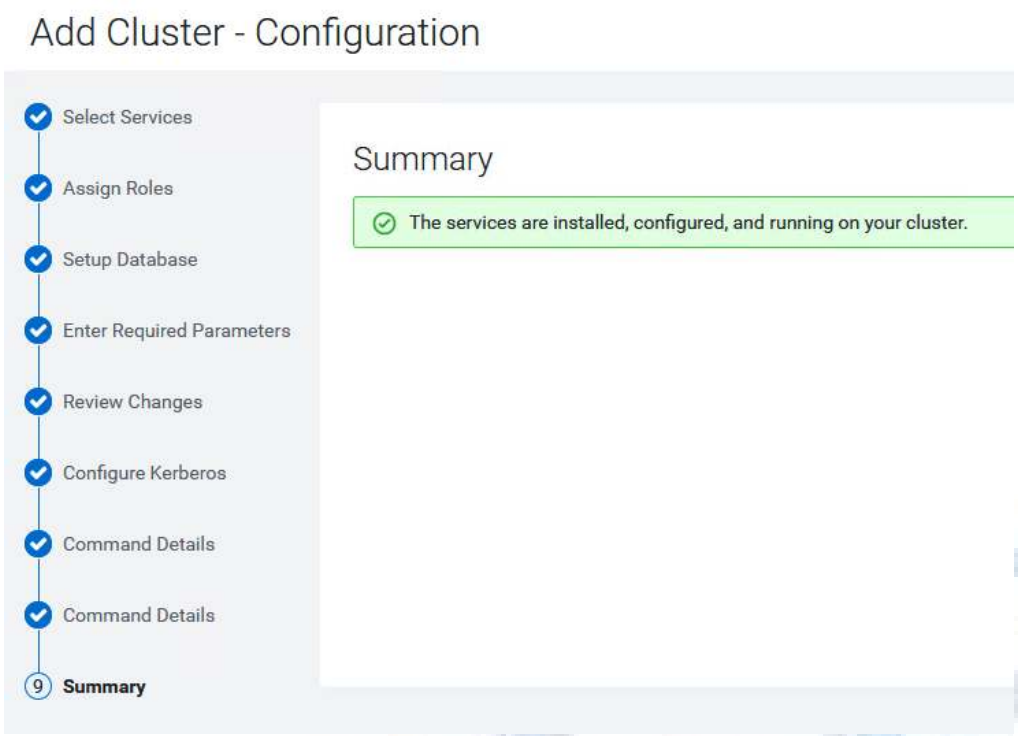
==========

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.redhat.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.
```

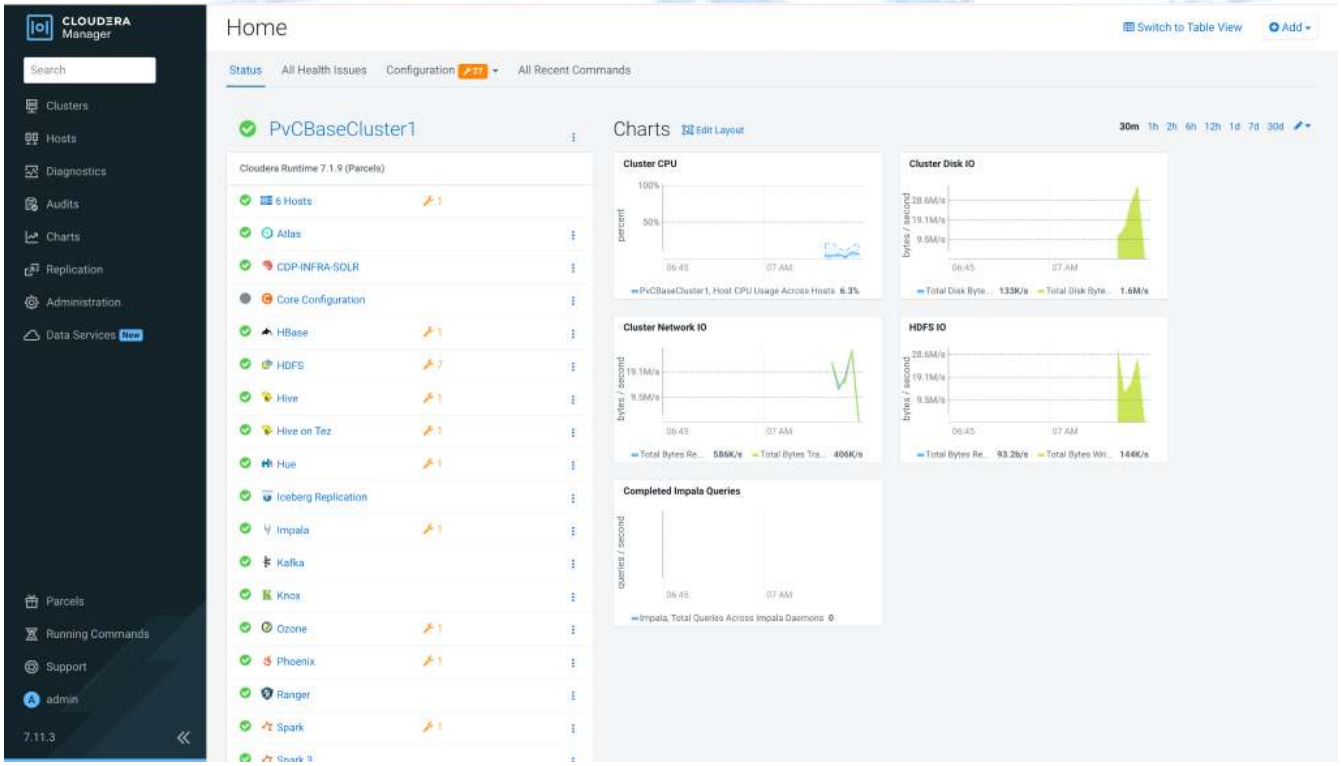
Step 36. Next, all the Cloudera services would get started and their prerequisite operations would also be run. These processes will run in a combination of serial and parallel processes. Wait for them to complete. Once completed, you will see a Green tick next to all the steps, as shown in the screenshot 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.redhat.local:7180/cmf/
HTTPS: https://cldr-mngr.redhat.local:7183/cmf/

HDFS:
HDFS-NAMENODE UI: https://pvcbase-master.redhat.local:9871/dfshealth.html

YARN:
HDFS-YARN JobHistory UI: https://pvcbase-master.redhat.local:19890/jobhistory
YARN RM UI: https://pvcbase-master.redhat.local:8090/ui2/#/cluster-overview

Ranger: https://pvcbase-master.redhat.local:6182/index.html#/policymanager/resource

Atlas: https://pvcbase-master.redhat.local:31443/login.jsp
(Login works with PAM-Linux Server Local User Credentials)

Knox:
https://pvcbase-master.redhat.local:8443/gateway/knoxsso/knoxauth/login.html?originalUrl=https://pvcbase-master.redhat.local:8443/gateway/homepage/home/?profile=token
(Login works with PAM-Linux Server Local User Credentials)

HiveServer2 UI: https://pvcbase-master.redhat.local:10002/

HUE: https://pvcbase-master.redhat.local:8889/hue/editor/?type=hive

HBASE: https://pvcbase-master.redhat.local:16010/master-status

Ozone:
Ozone Recon: https://pvcbase-master.redhat.local:9889/#/Overview
Ozone SCM: https://pvcbase-master.redhat.local:9877/#!/
Ozone Manager: https://pvcbase-master.redhat.local:
S3 Gateway: https://pvcbase-master.redhat.local:
Gateway: https://pvcbase-workter1.redhat.local:
OzoneDataNode: https://pvcbase-worker1.redhat.local:

Spark JobHistory Server:
Spark2: https://pvcbase-master.redhat.local:18488/
Spark3: https://pvcbase-master.redhat.local:18489/

Impala:
Impala Catalog: https://pvcbase-master.redhat.local:25020/
Impala Statestore: https://pvcbase-master.redhat.local:25010/

Job History Server: https://pvcbase-master.redhat.local:9991/

Step 1. Port requirements for different services on PvC Base Cluster/Data Services (OCP) 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)

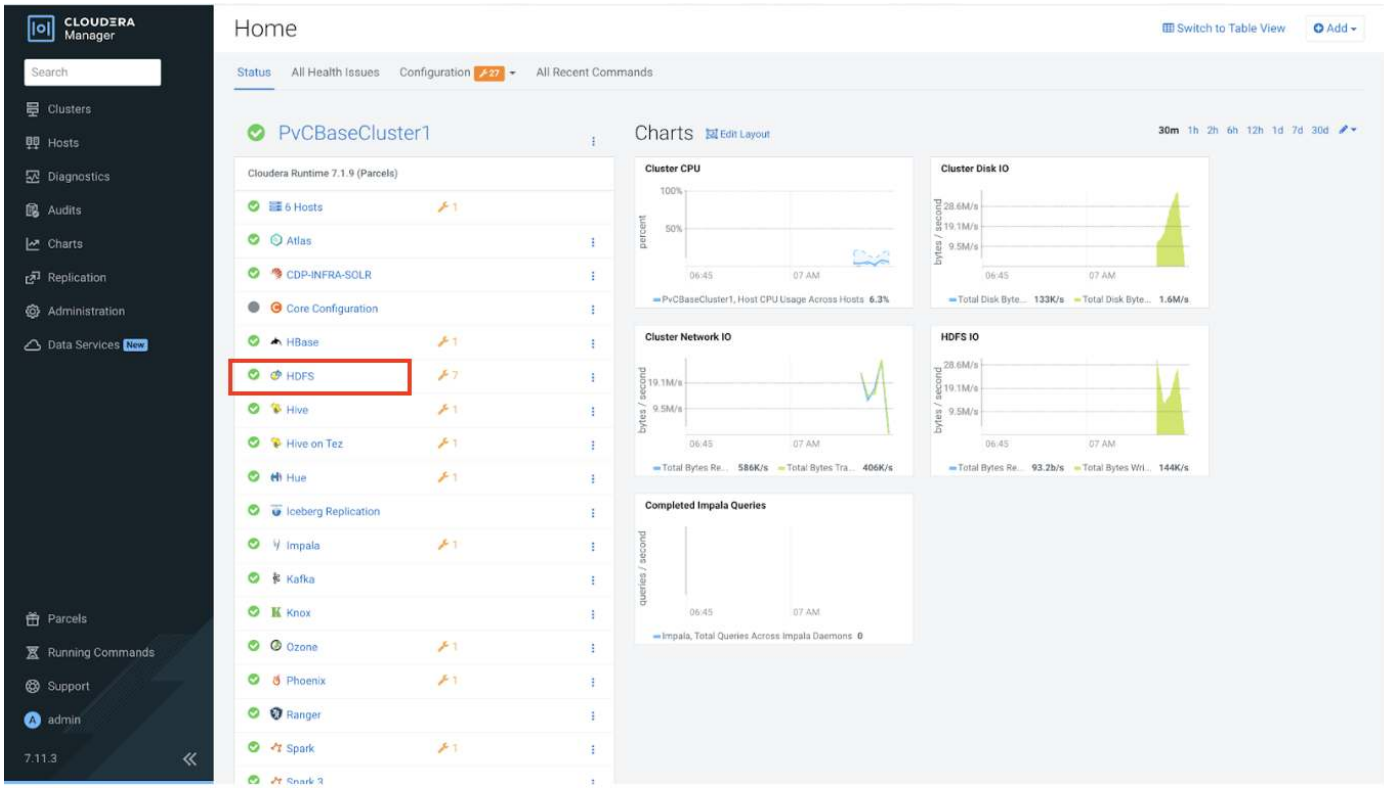
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-ports-used-by-runtime.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-ports-third-party-components.html
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/installation/topics/cdpdc-ports-used-by-cm.html
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/1.5.5/installation/topics/cdppvc-ocp-install.html

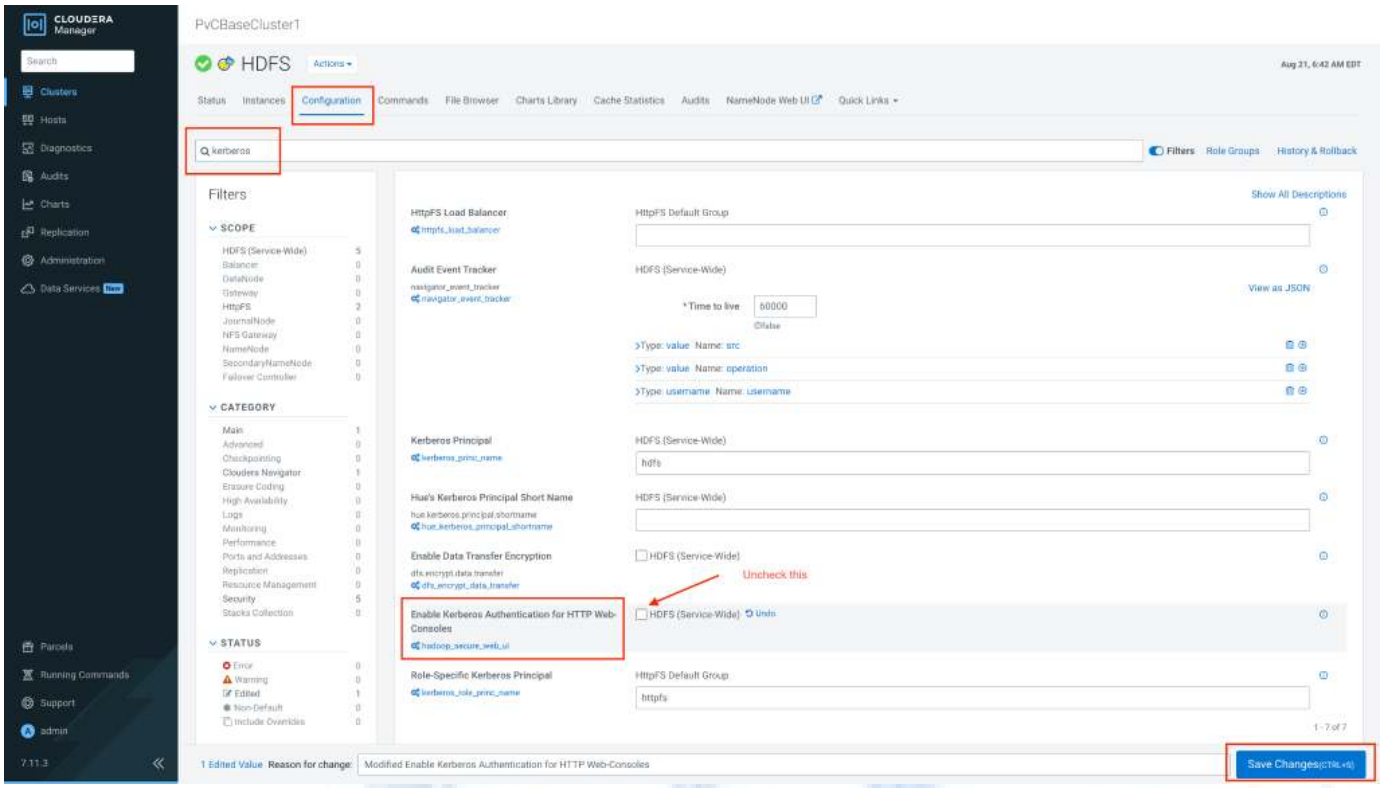
Step 2. Disable Kerberized Web-UI for YARN, Spark, HBase, HDFS, etc. from configuration:

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 pvcocp-master.redhat.local
52.221.202.246 hue-kd-hive-vw1.apps.redhat.local
52.221.202.246 cml-task-bo6klv.kuldeep-cml.apps.redhat.local
52.221.202.246 kuldeep-cml.apps.redhat.local
18.139.222.78 pvcbase-master.redhat.local
13.251.65.11 cldr-mngr.redhat.local

# OCP Links
52.221.202.246 console-cdp.apps.redhat.local prometheus-cp.apps.redhat.local
infra-prometheus.apps.redhat.local validation-cdp.apps.redhat.local kube-dashboard.apps.redhat.local
longhorn.apps.redhat.local fluent-console-cdp.apps.redhat.local vault.localhost.localdomain

# PvC Base Cluster Nodes
18.139.222.78 pvcbase-master.redhat.local pvcbase-master
13.215.202.164 pvcbase-worker1.redhat.local pvcbase-worker1
172.31.23.0   pvcbase-worker2.redhat.local pvcbase-worker2
18.141.13.157 pvcbase-worker3.redhat.local pvcbase-worker3

# PvC Data Services Cluster Nodes
172.31.30.239 pvcocp-master.redhat.local pvcocp-master
172.31.22.43  pvcocp-worker1.redhat.local pvcocp-worker1
172.31.30.249 pvcocp-worker2.redhat.local pvcocp-worker2
172.31.26.24  pvcocp-worker3.redhat.local pvcocp-worker3
172.31.24.198 pvcocp-worker4.redhat.local pvcocp-worker4
172.31.24.53  pvcocp-worker5.redhat.local pvcocp-worker5
```

Step 4. Optionally, Update the /etc/hosts file on your working machine/JumpHost (The /etc/hosts entries required for OCP Data Services)
https://docs.cloudera.com/management-console/1.5.5/private-cloud-security-overview/mc-private-cloud-security-overview.pdf

Embedded Container Service (OCP) :
- 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 OCP cluster. For example, if your console URL is "console-cdp.apps.redhat.local", then the APPDOMAIN is "redhat.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.

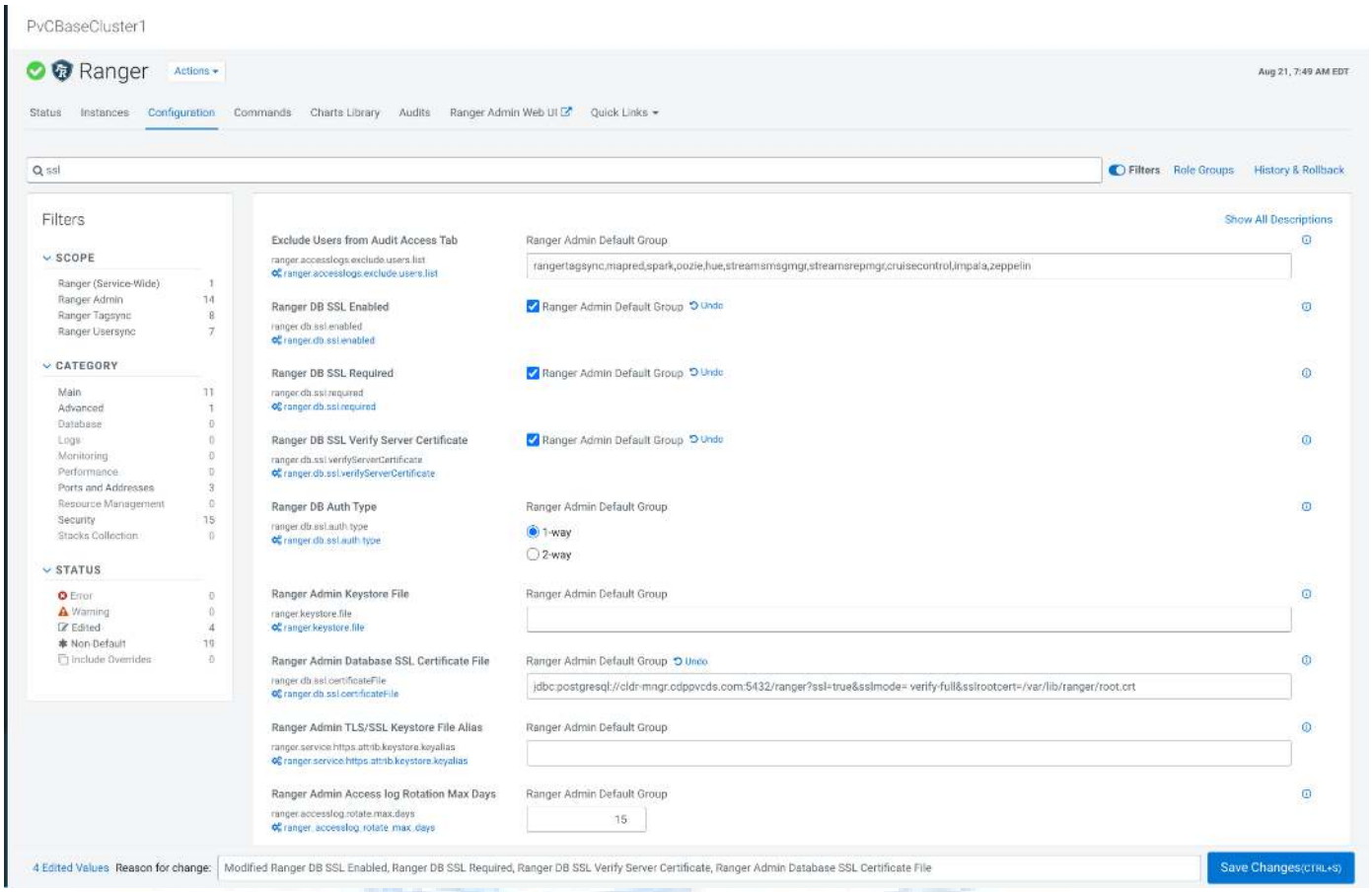
```
[root@pvcbase-master ~]# cp -rv /root/.postgresql/root.crt /var/lib/ranger/root.crt
```

- **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.redhat.local:5432/ranger?ssl=true&sslmode=verify-full&ss t=/var/lib/ranger/root.crt***
Set Load Balancer Address (Optional)	http://<ranger_host>:6080 http://pvcbase-master.redhat.local:6080 https://<ranger_host>:6182 https://pvcbase-master.redhat.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.redhat.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

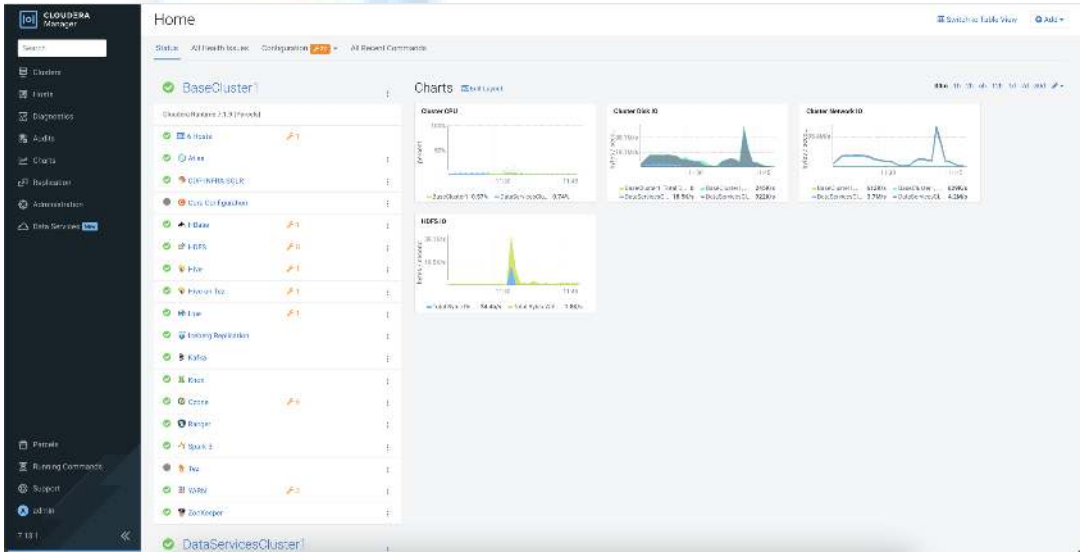
Configure Browsers for Kerberos Authentication

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



****

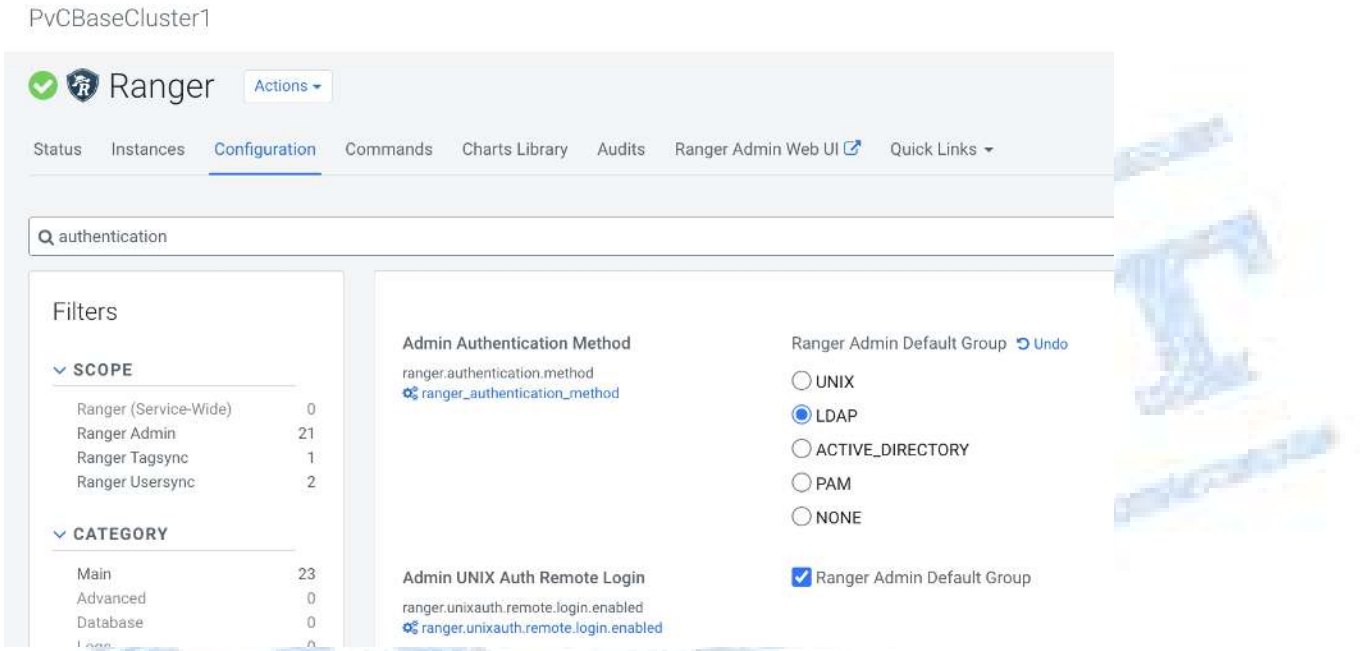
Configure Ranger authentication for LDAP (Optional– Skip this Step)

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".



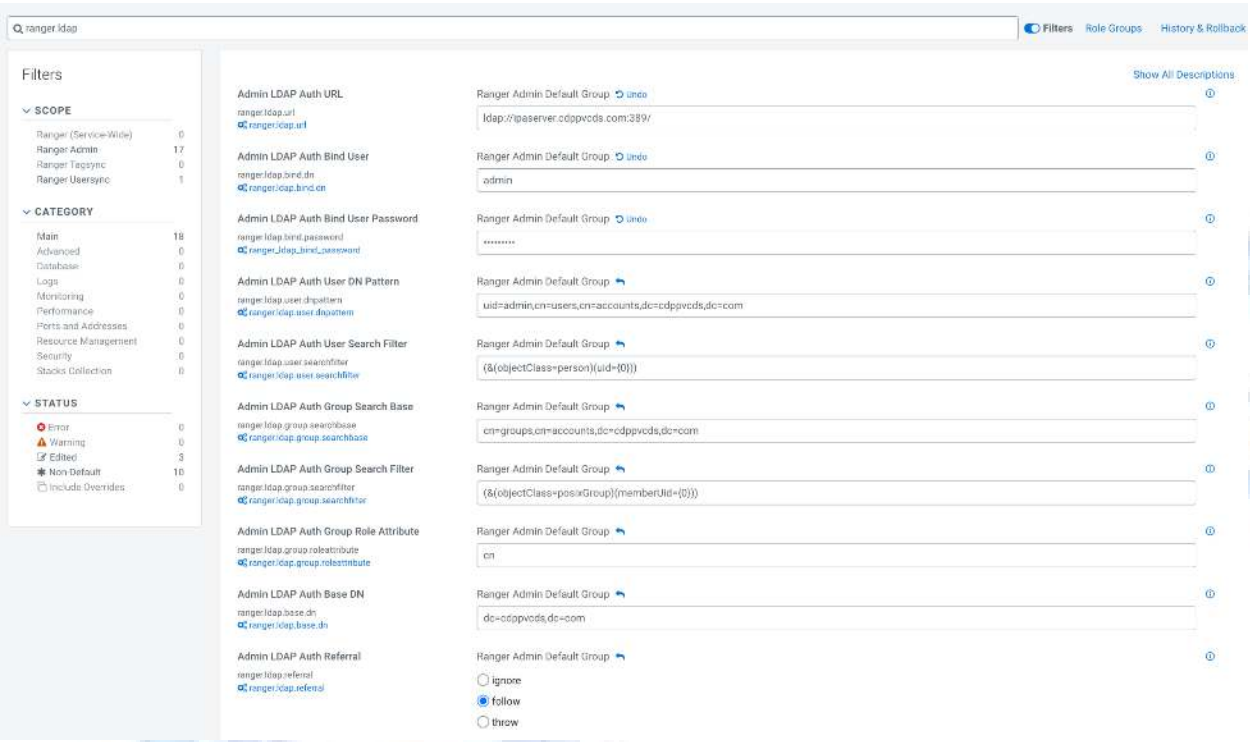
4. Configure the following settings for LDAP authentication as shown below, the details depends on your configuration, based on existing LDAP/AD setup:

Table 8. User LDAP Integration

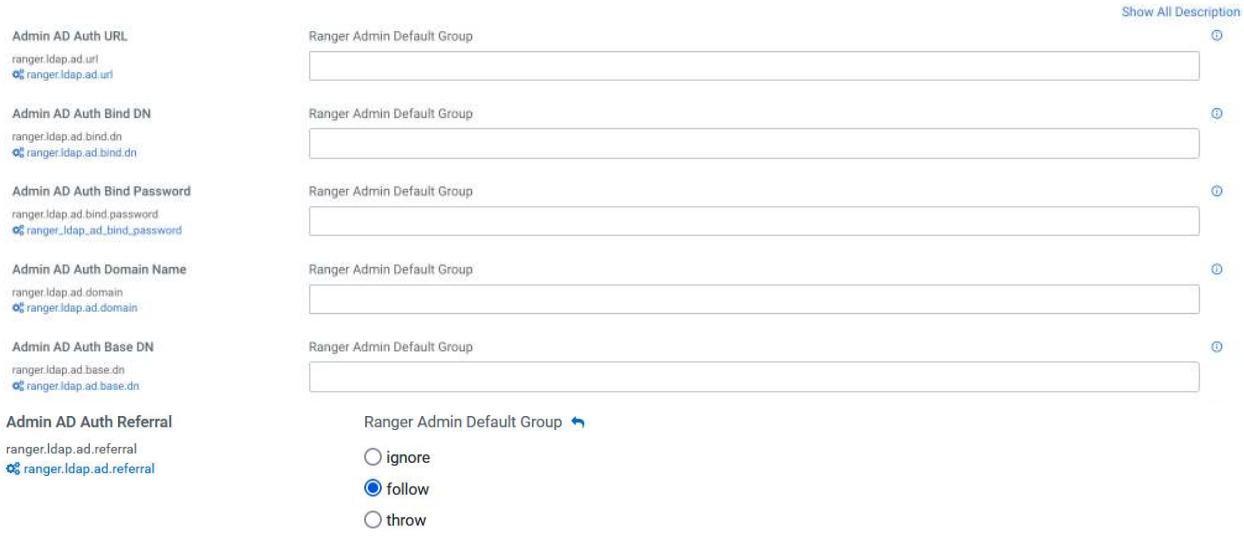
Component	Value
Admin **LDAP**/AD Auth URL:	ldap://ipaserver.redhat.local:389/ ***(Give LDAP or AD Server LDAP ADDR)***
Admin **LDAP**/AD Auth **Bind User**/ Bind DN:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
Admin **LDAP**/AD Auth Bind User Password:	***<redhat123>*** (password for KDC admin, 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)	redhat.local

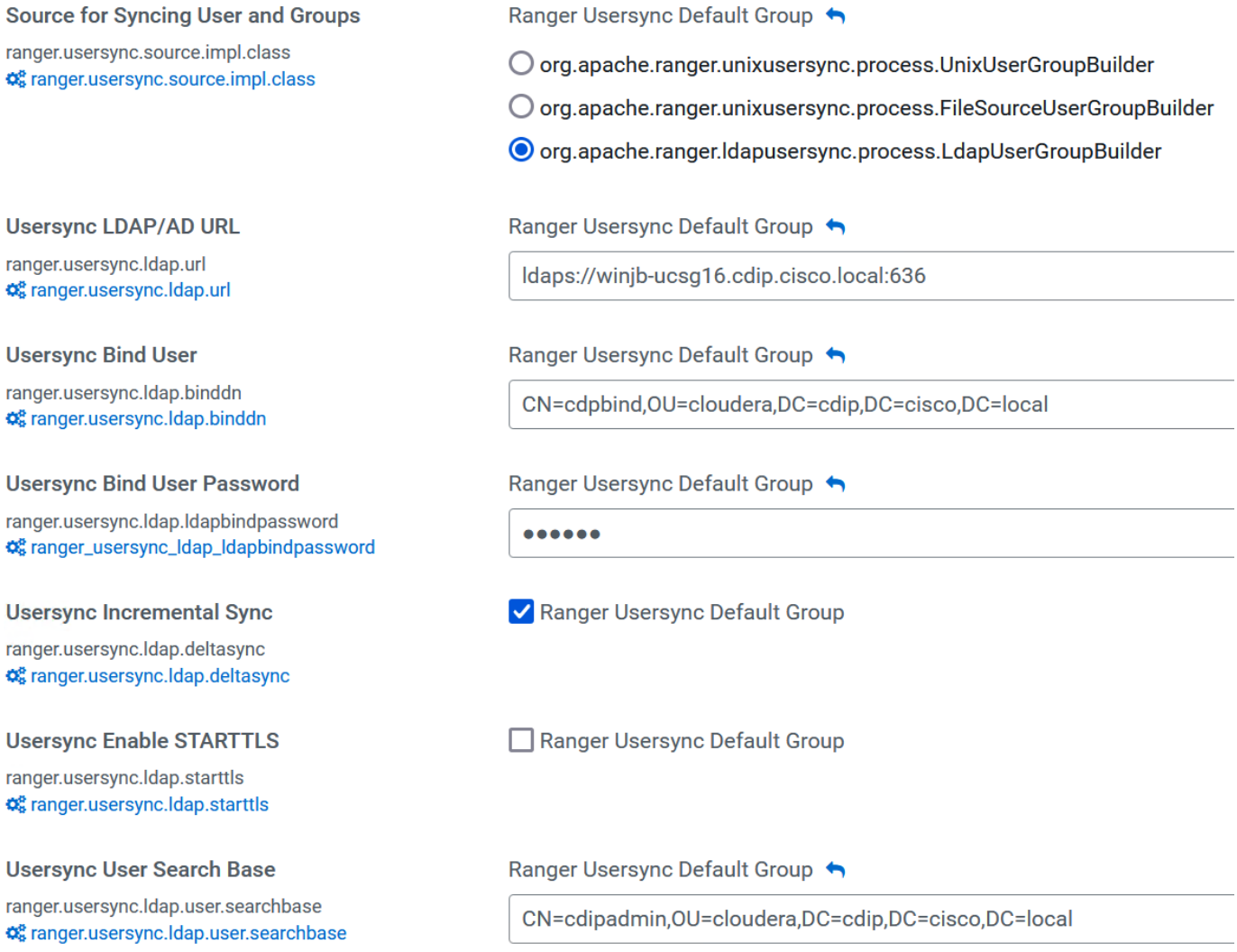
(Search under Configuration for- *ranger.ldap*)



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.redhat.local:389/
Usersync Bind User:	uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local
Usersync Bind User Password:	*<redhat123>* (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

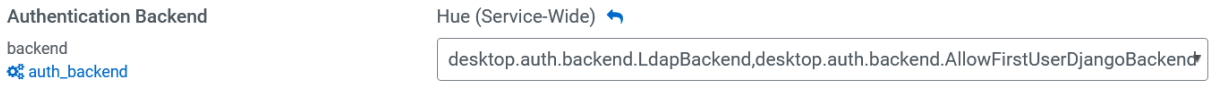
****

Configure Hue for LDAP Authentication (Optional– Skip this Step)

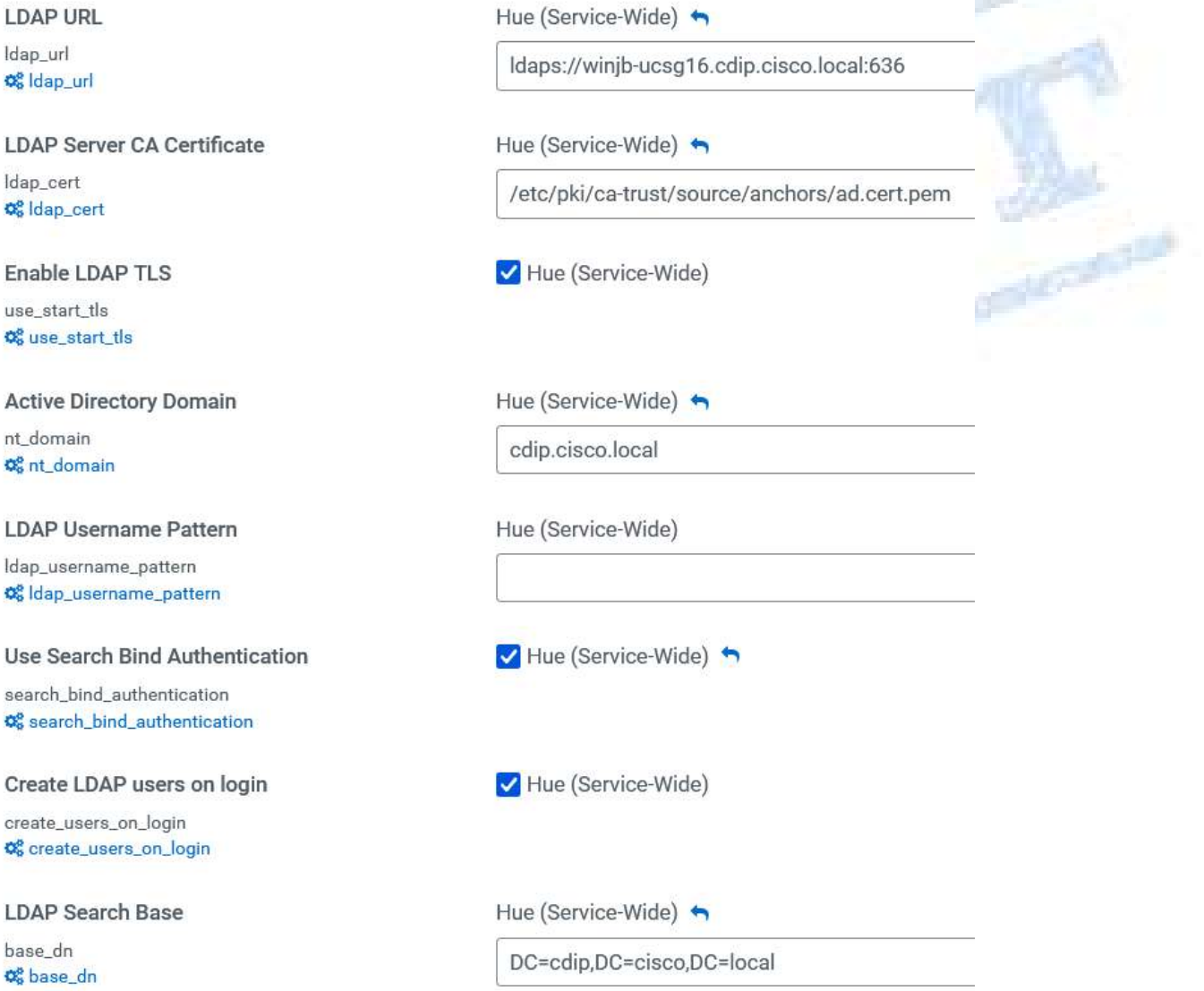
Configuring Hue for Lightweight Directory Access Protocol (LDAP) enables you to import users and groups from a directory service, synchronize group membership manually or automatically at login, and authenticate with an LDAP server. Hue supports Microsoft Active Directory (AD) and open standard LDAP 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:



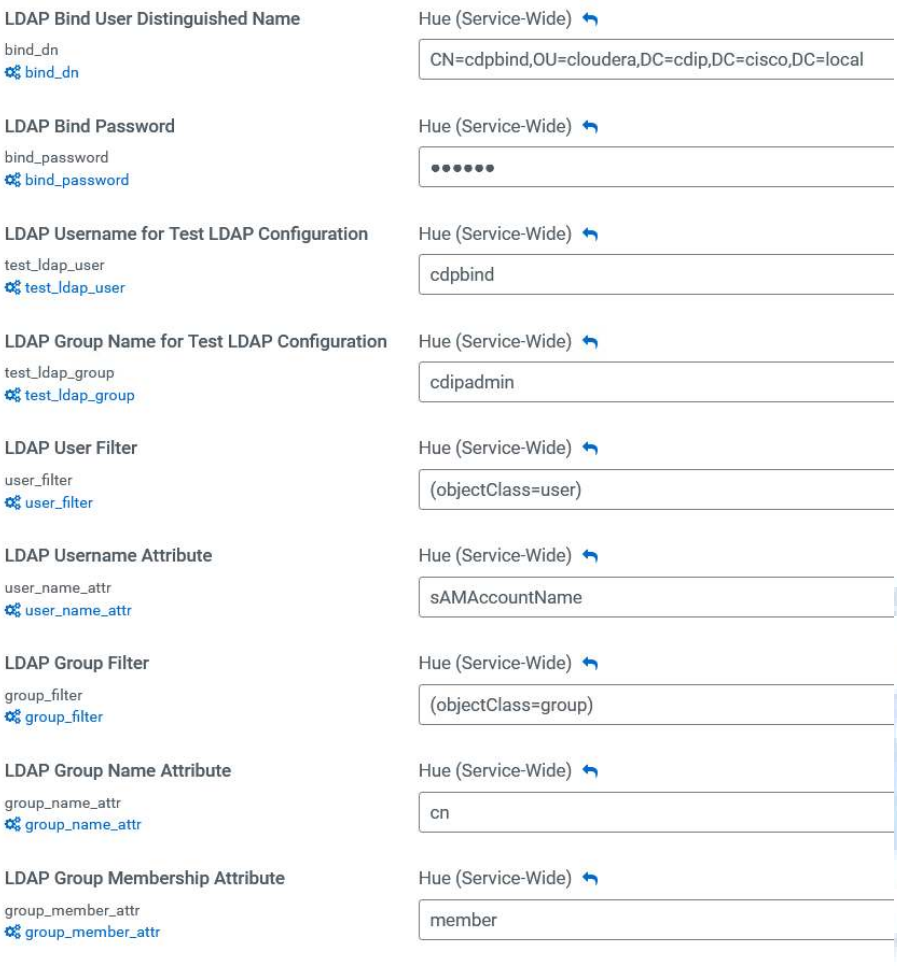


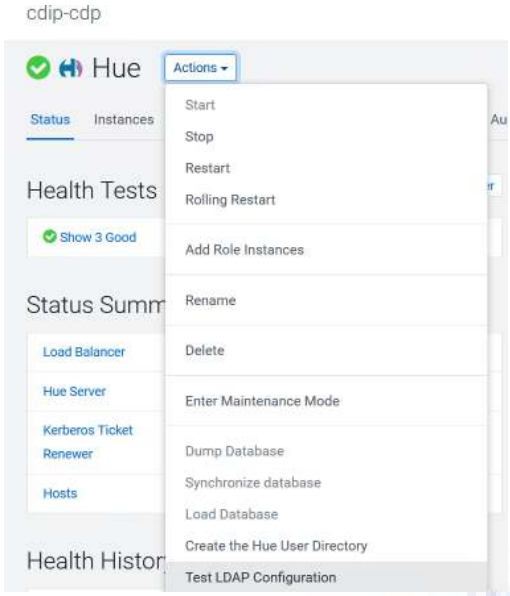
Table: **LDAP Integration**

Component	Value
LDAP URL:	ldap://ipaserver.redhat.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:	*<redhat123>* (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

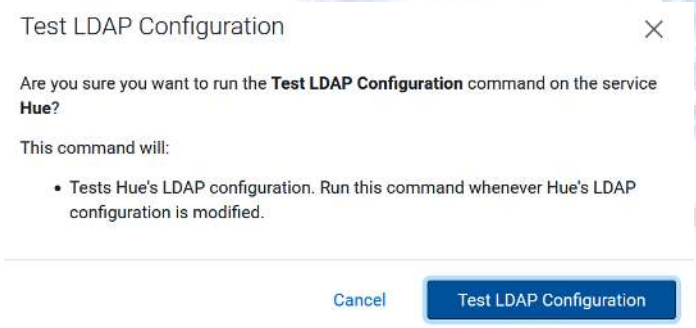
4. Click on save changes

5. Restart HUE service.

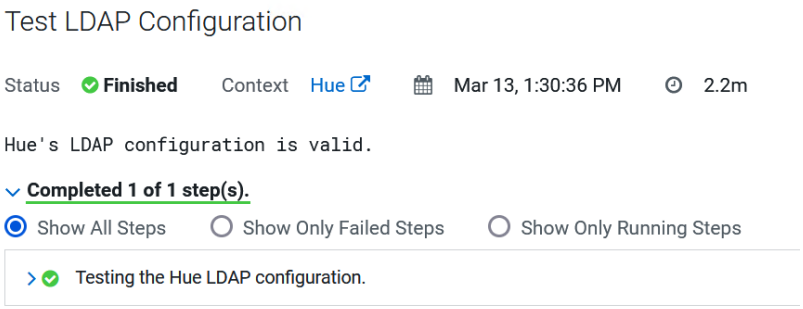
6. Click on Actions next to Hue. Click on Test LDAP Configuration.



7. Click on Test LDAP Configuration.



8. Click on Finish.



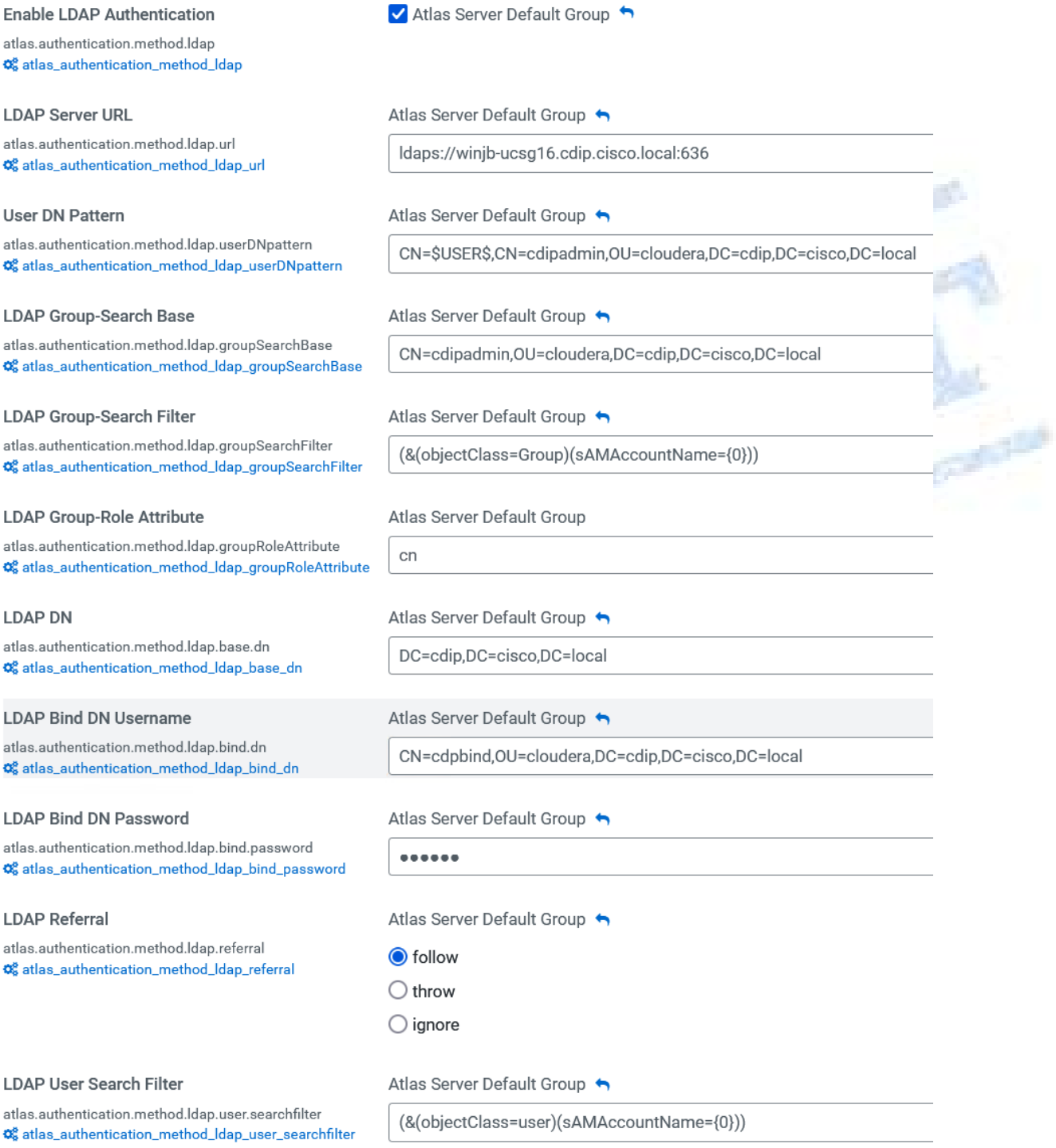
For more details:
https://docs.cloudera.com/cdp-private-cloud-base/7.3.1/securing-hue/topics/hue-authenticate-users-with-ldap.html

****

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:



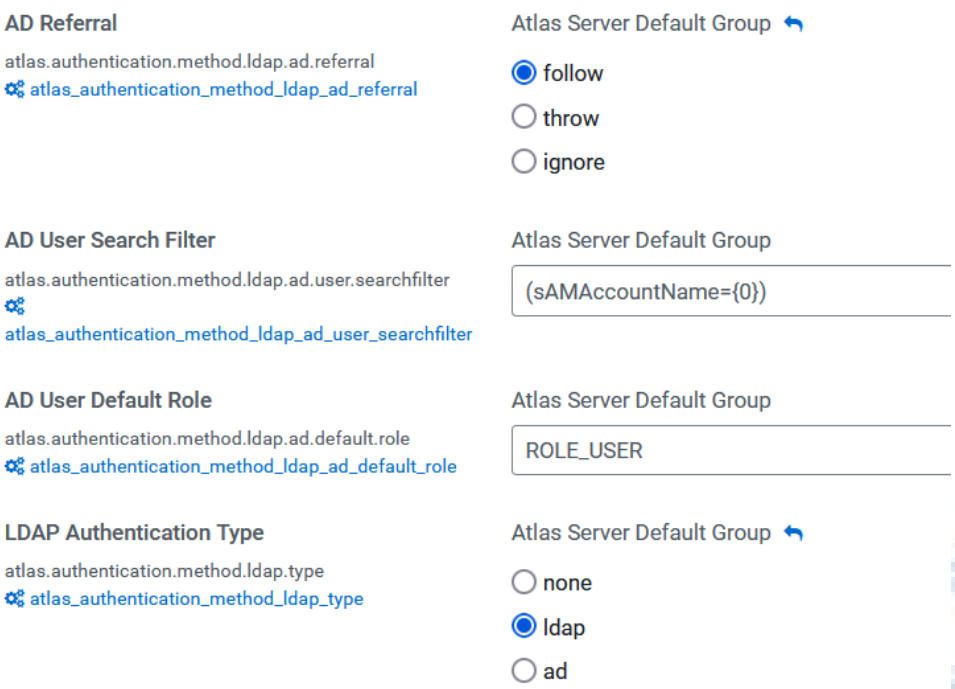


Table 10. Atlas LDAP Integration

Component	Value
Enable LDAP Authentication (Atlas):	True (Checked)
LDAP Server URL:	ldap://ipaserver.redhat.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:	*<redhat123>* (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)

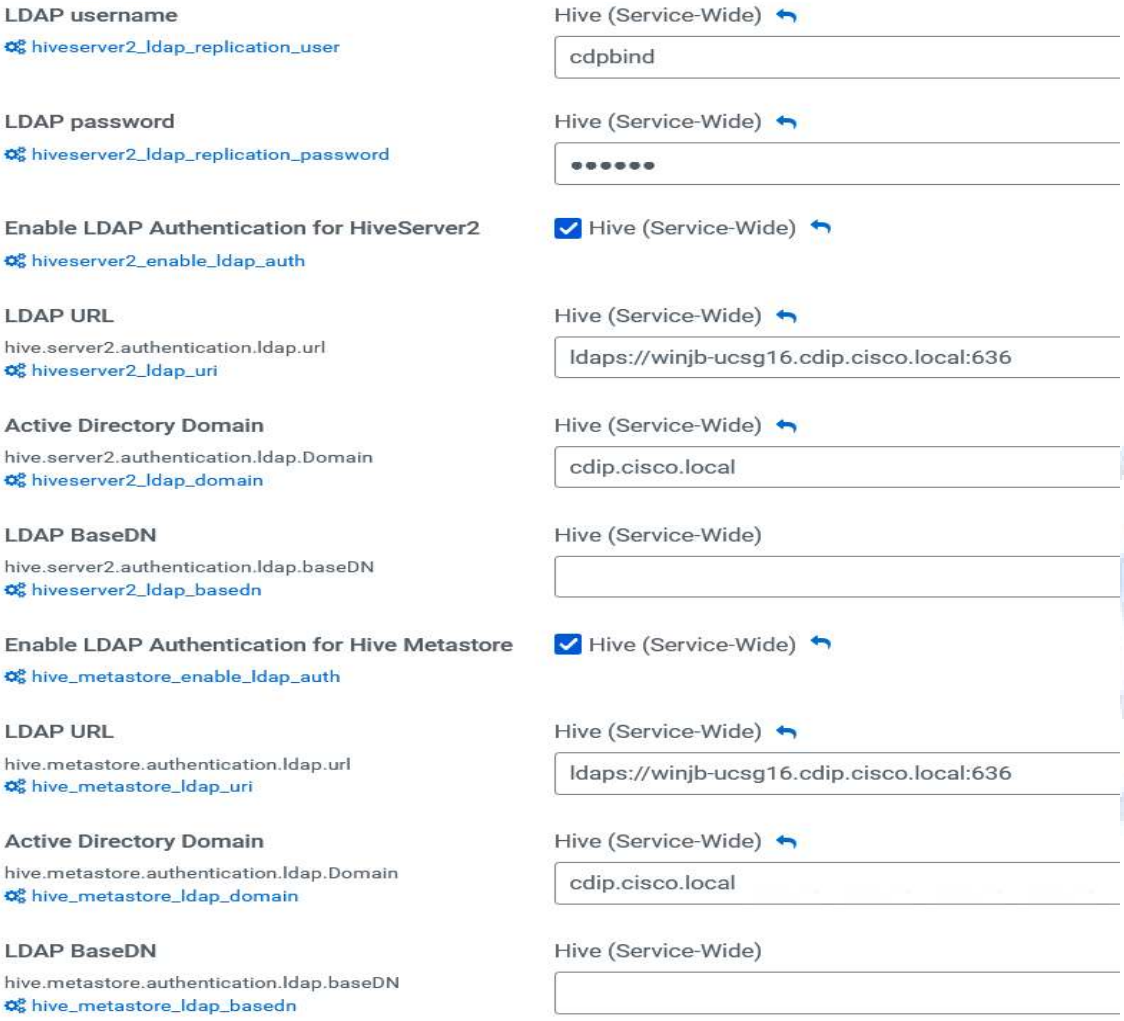


Table 11. LDAP Integration-Hive

Component	Value
LDAP Username:	admin
LDAP Password:	*<redhat123>* (password for KDC admin, configured earlier)
Enable LDAP Authentication for HiveS2:	True (Checked)
LDAP URL:	ldap://ipaserver.redhat.local:389/
LDAP Base DN:	dc=cldrsetup,dc=local
Enable LDAP Authentication for HMS:	True (Checked)
LDAP URL:	ldap://ipaserver.redhat.local:389/
LDAP Base DN:	dc=cldrsetup,dc=local

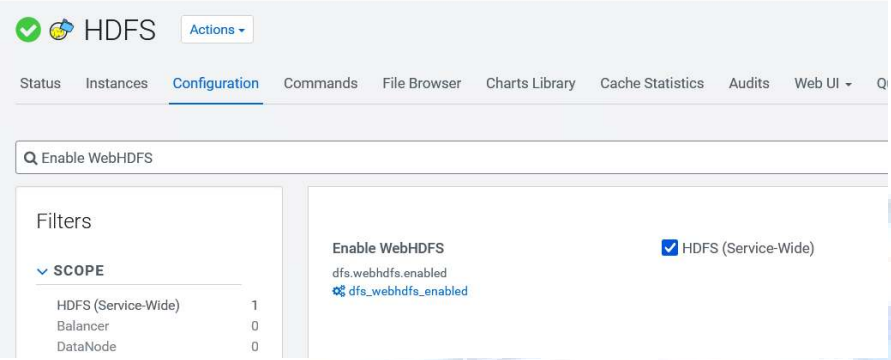
****

Configure HDFS properties to optimize log collection (Optional– Skip this Step)

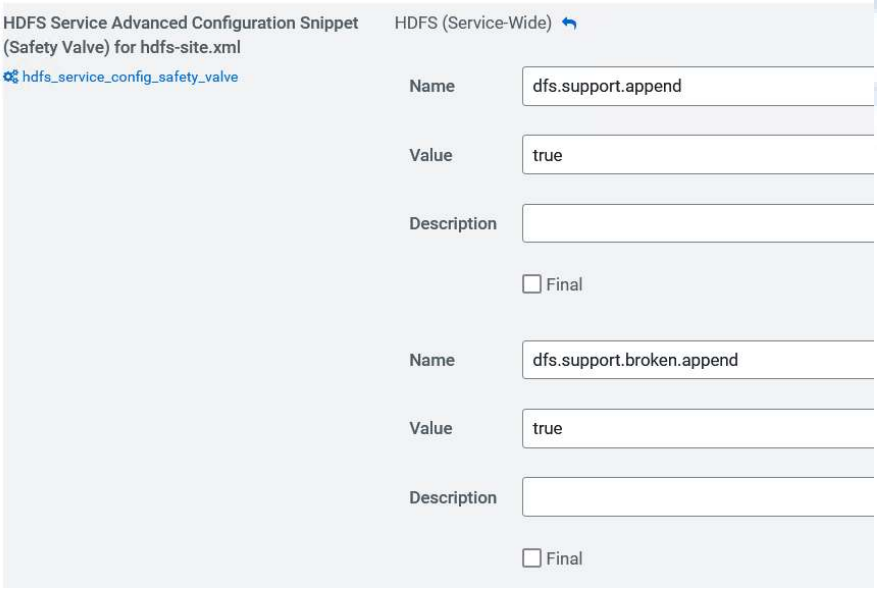
CDP uses “out_webhdfs” Fluentd output plugin to write records into HDFS, in the form of log files, which are then used by different Data Services to generate diagnostic bundles. Over time, these log files can grow in size. To optimize the size of logs that are captured and stored on HDFS, you must update 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.

****

CDP Private Cloud (PvC) Data Services (DS) Installation

CDP Private Cloud Data Services lets you deploy and use the Cloudera Data Warehouse (CDW), Cloudera Machine Learning (CML), and Cloudera Data Engineering (CDE) Data Services.

This section summarizes Cloudera Private Cloud Data Science v1.5.5 installation through Openshift Container Platform on Cloudera Private Cloud Base 7.1.9.

A CDP Private Cloud 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 CDP Private Cloud Data Services you need a CDP Private Cloud Base cluster, along with container-based clusters that run the Data Services. You can either use a dedicated ***Red Hat OpenShift Container Platform cluster (OCP)*** or deploy an ***Embedded Container Service (ECS)*** container cluster.

The Private Cloud 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.

CDP Private Cloud Base provides the following components and services that are used by CDP Private Cloud 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.

Openshift Container Platform (OCP) checklist

Use the checklist for OpenShift Container Platform (OCP) for CDP Private Cloud Data Services:
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-requirements-ocp.html

CDP Private Cloud Data Services software requirements:

1. You must have a minimum of one agent node for OCP.
2. Enable TLS on the Cloudera Manager cluster for communication with components and services.
3. Set up Kerberos on these clusters. (i.e. krb5-workstation, krb5-libs, NTP etc is set-up and configured)
4. Follow the steps in this topic to install CDP Private Cloud.
5. Ensure that your Kubernetes ***kubeconfig*** has permissions to create Kubernetes namespaces.
6. You must require persistent storage classes defined in your OpenShift cluster. Storage classes can be defined by OpenShift cluster administrators.
7. In case of a non-embedded Docker registry, only TLS-enabled custom Docker Registry is supported. Ensure that you use a TLS certificate to secure the custom Docker Registry. The TLS certificate can be self-signed, or signed by a private or public trusted Certificate Authority (CA).
8. Only TLS 1.2 is supported for authentication with Active Directory/LDAP. You require TLS 1.2 to authenticate the CDP control plane with your LDAP directory service like Active Directory.
9. OCP network configurations that restrict pod communication are not supported. For example, multi-tenancy isolation with network policy is not supported.

10. If this Cloudera Manager instance or your Kubernetes cluster does not have connectivity to https://archive.cloudera.com/p/cdp-pvc-ds/, you must mirror the Cloudera archive URL using a local HTTP server. You can launch the Private Cloud installation wizard from Cloudera Manager and follow the steps to install CDP Private Cloud Data Services in an air gapped environment when your Cloudera Manager instance or your Kubernetes cluster does not have access to the Internet.

11. 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.APP.DOMAIN***".

12. It is recommended that you leave IPv6 enabled at the OS level on all OCP nodes.

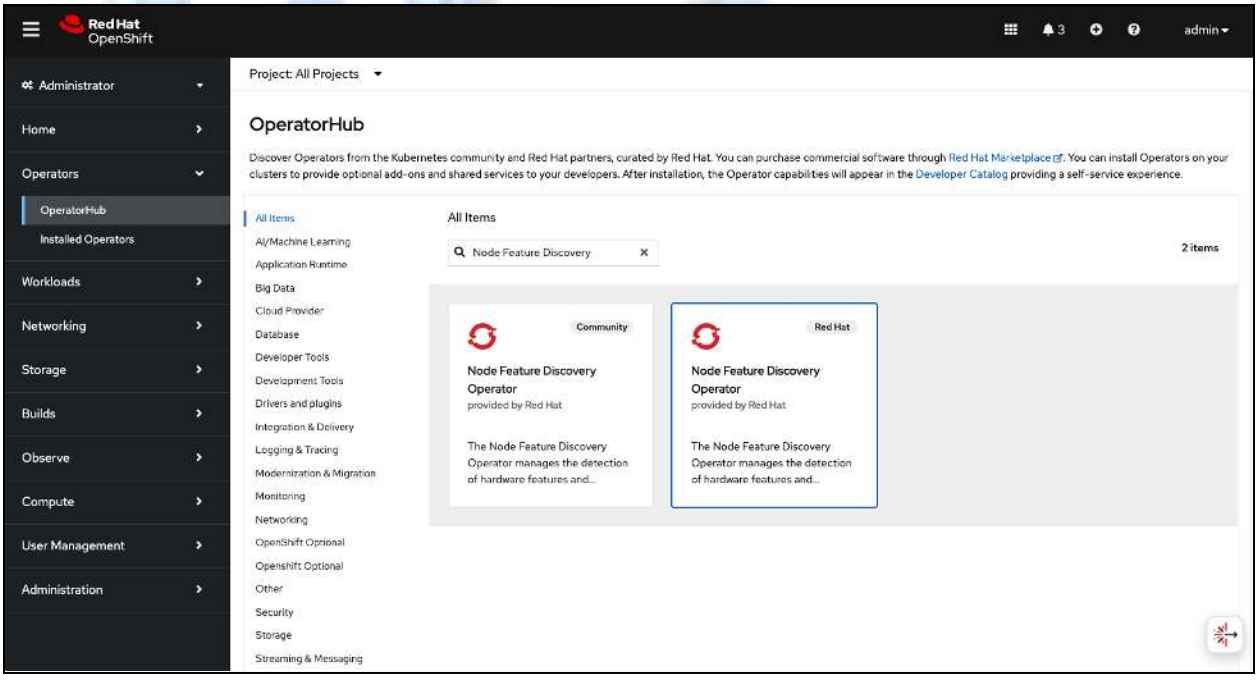
13. Take care that enough disk space is available on each host in the OCP cluster.

Installing NVIDIA OPERATOR ON OCP

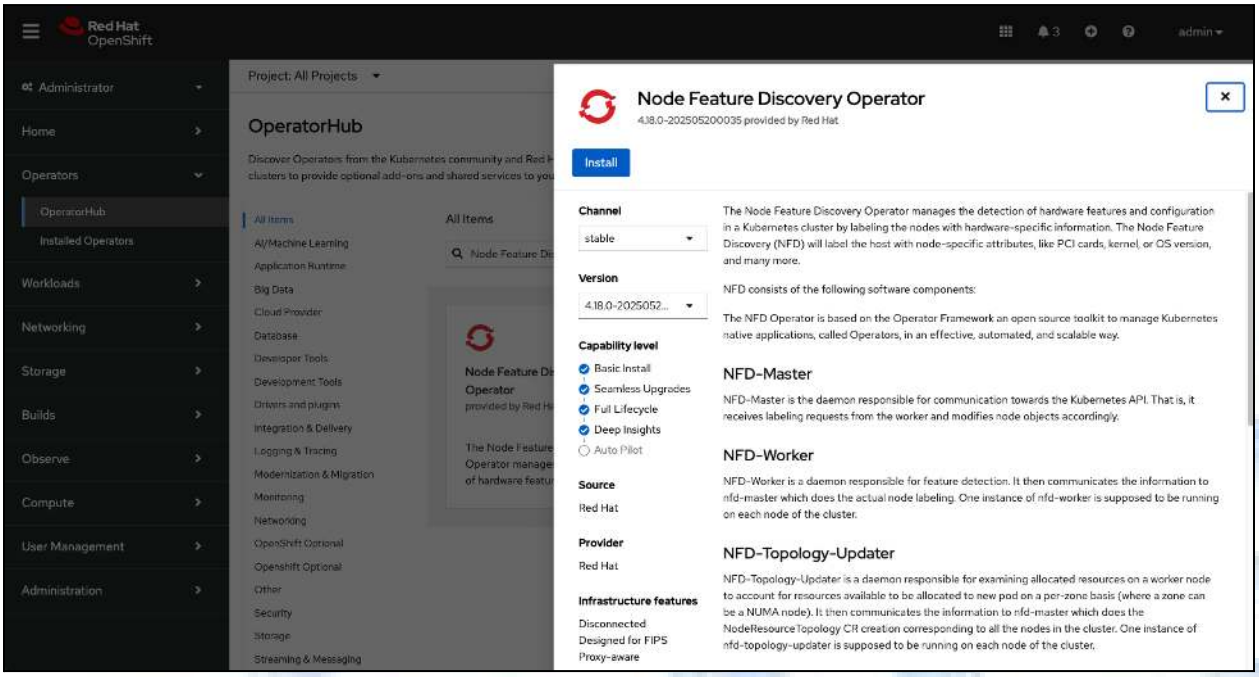
Referencing the following: NVIDIA GPU Operator on Red Hat OpenShift Container Platform AND Nvidia GPU in Openshift

Install NFD Operator

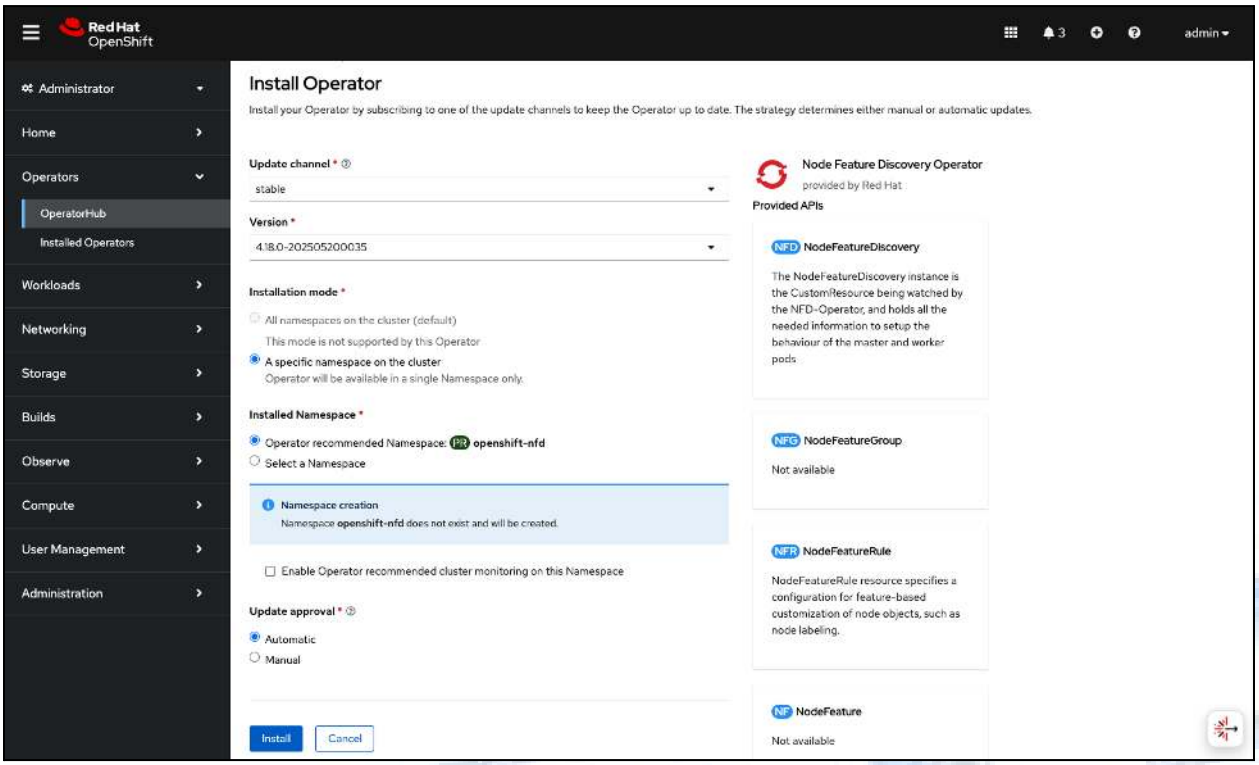
STEP 1. In the ***Openshift console***, navigate to ***Operators > OperatorHub***. Search for ***Node Feature Discovery***. Click ***Node Feature Discovery*** *that is* provided by Red Hat.



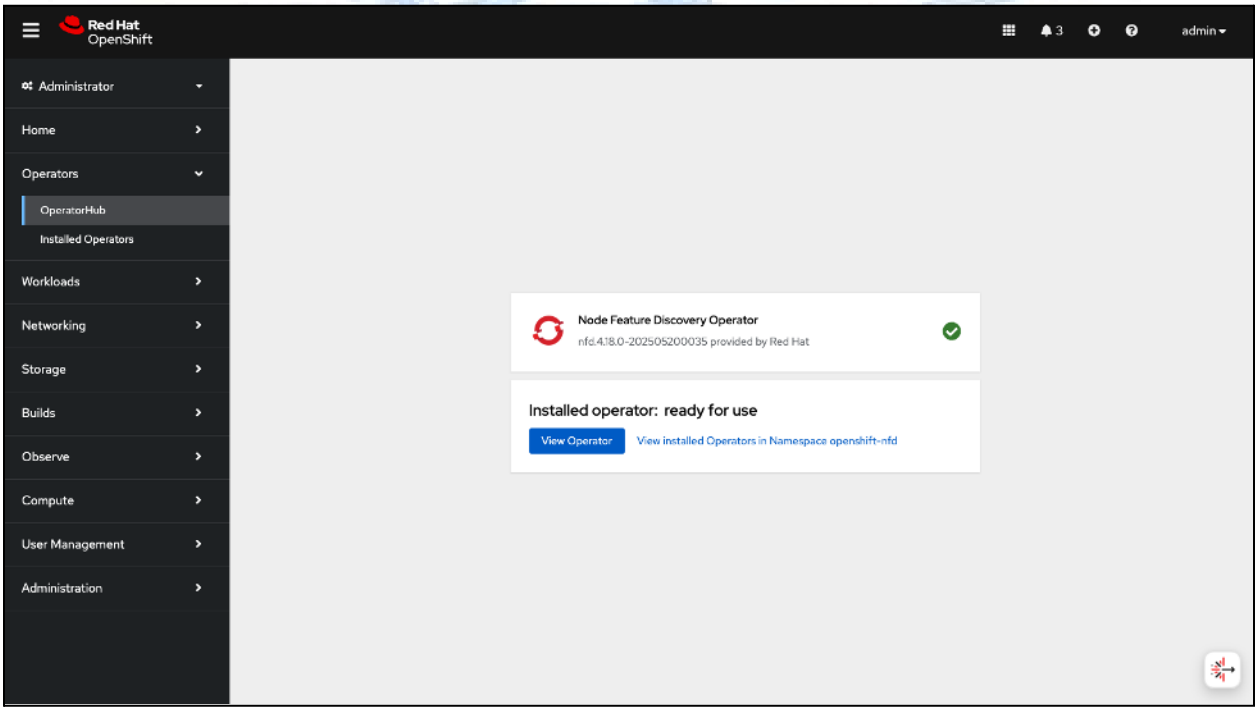
STEP 2. Click on ***Install***.



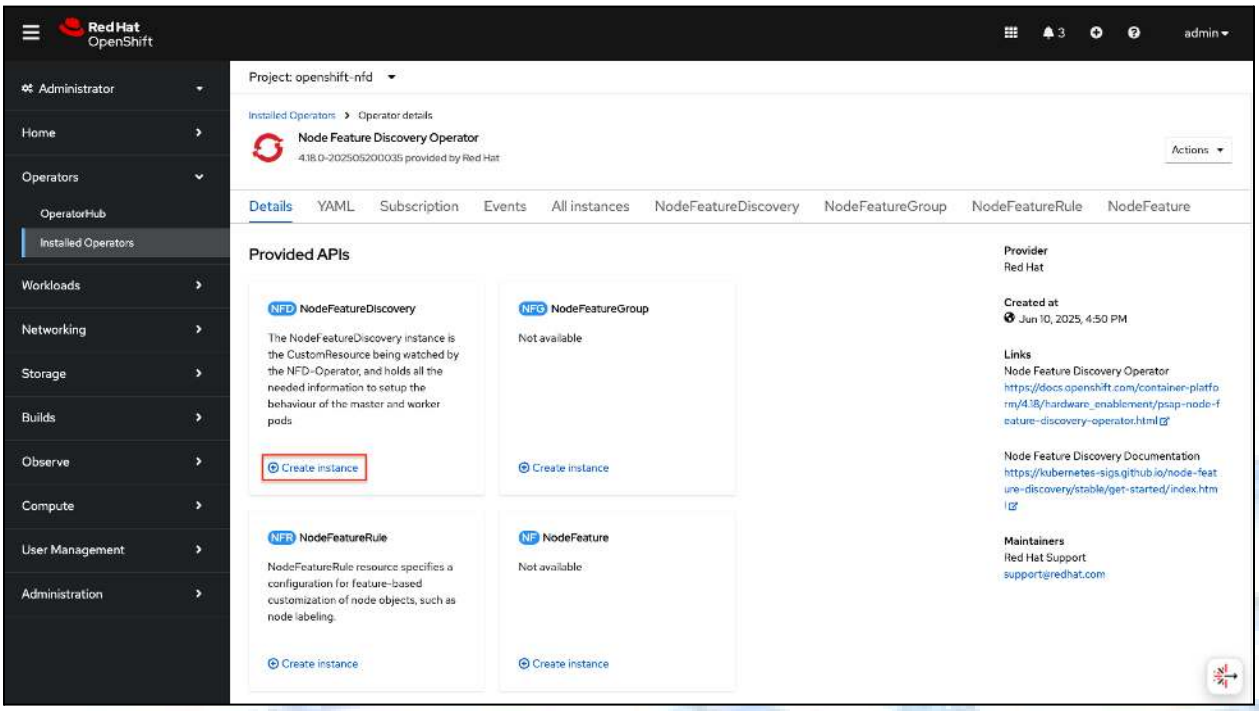
STEP 3. Click on Install.



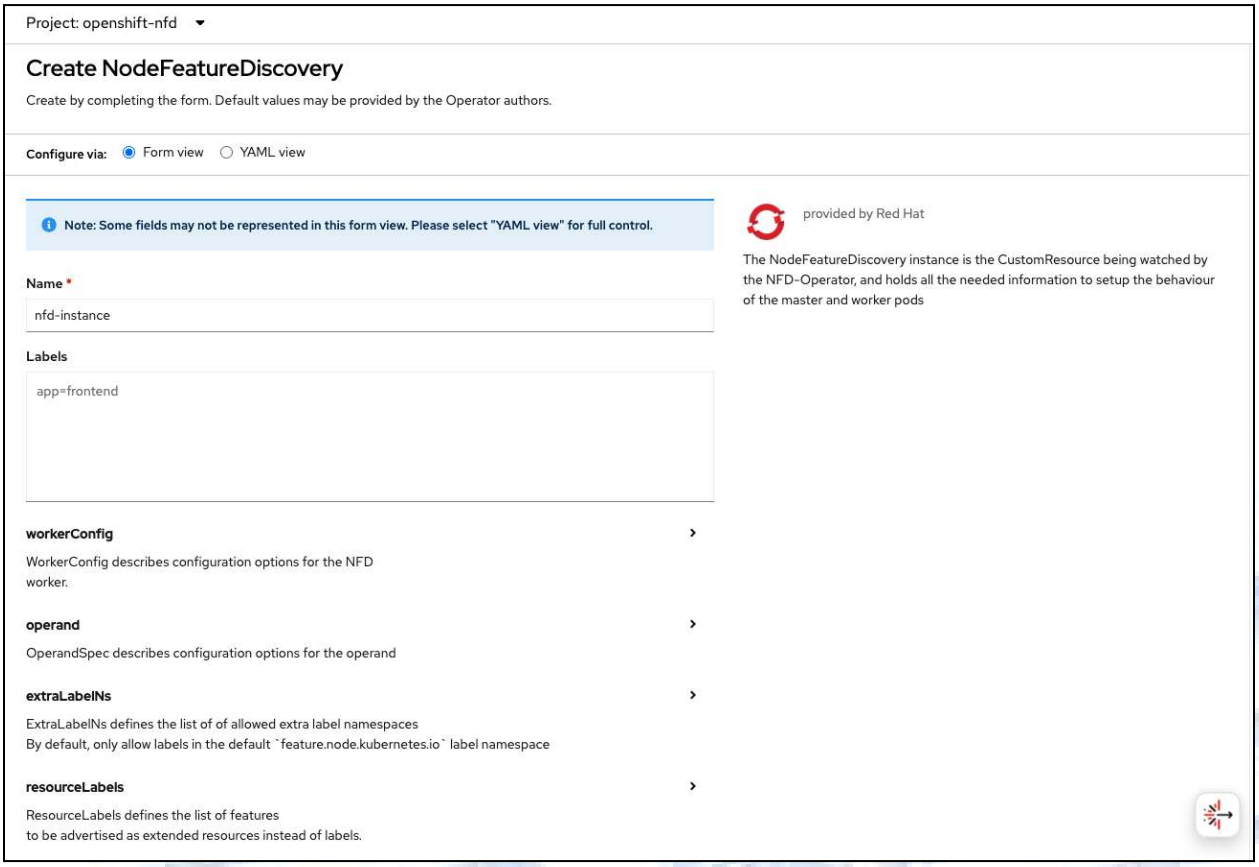
STEP 4. Once the Operator Installation is complete, click on ***View Operator***.

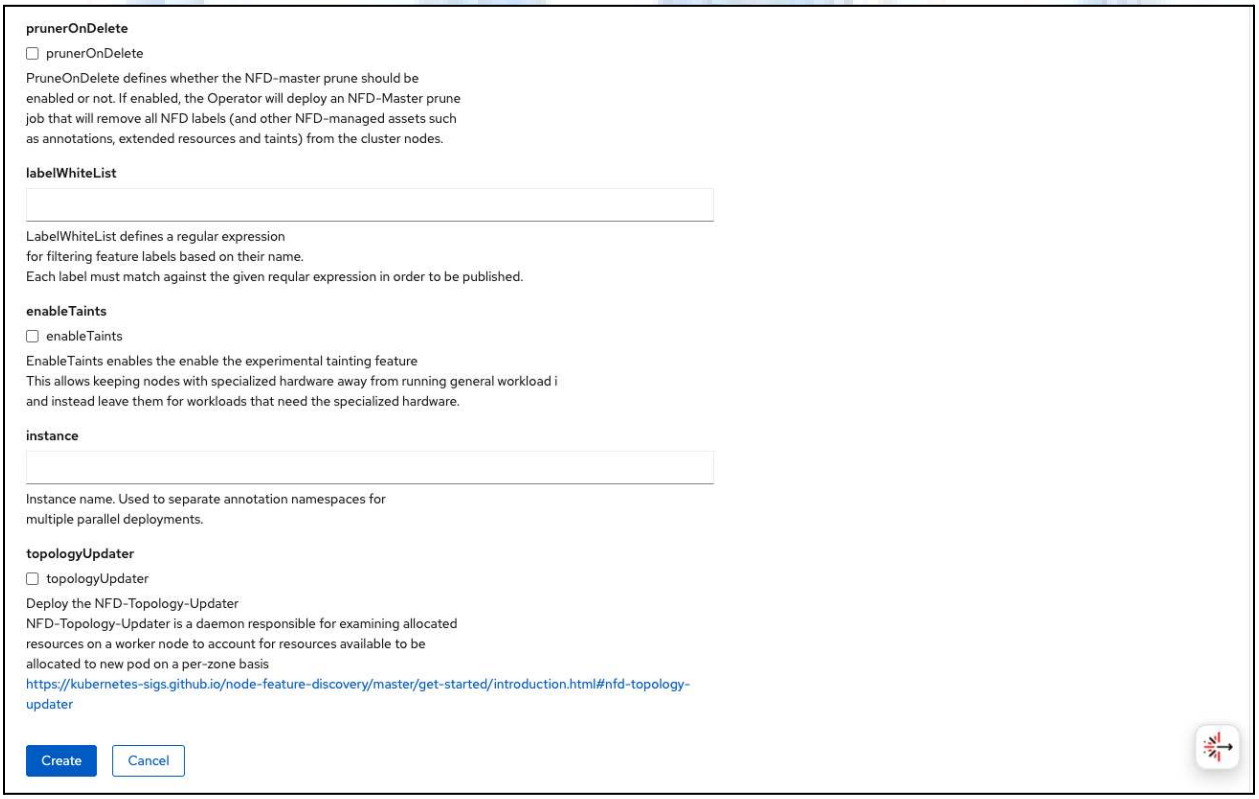


STEP 5. Click on ***Create instance*** under the ***NodeFeatureDiscovery***.

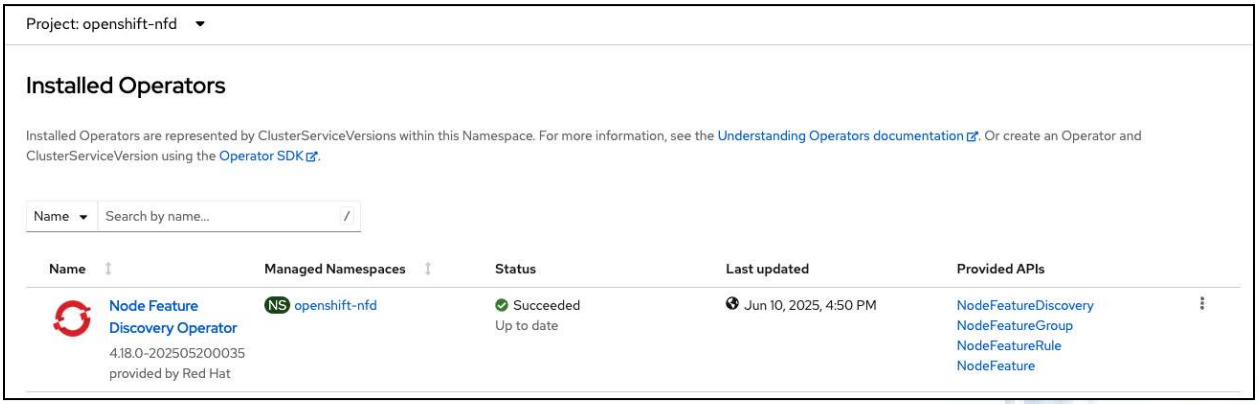


STEP 6. Scroll to the bottom of the screen and click on ***Create***.





STEP 7. When complete, navigate to ***Operators > Installed Operators***. The NFD operator should show a status of Succeeded.

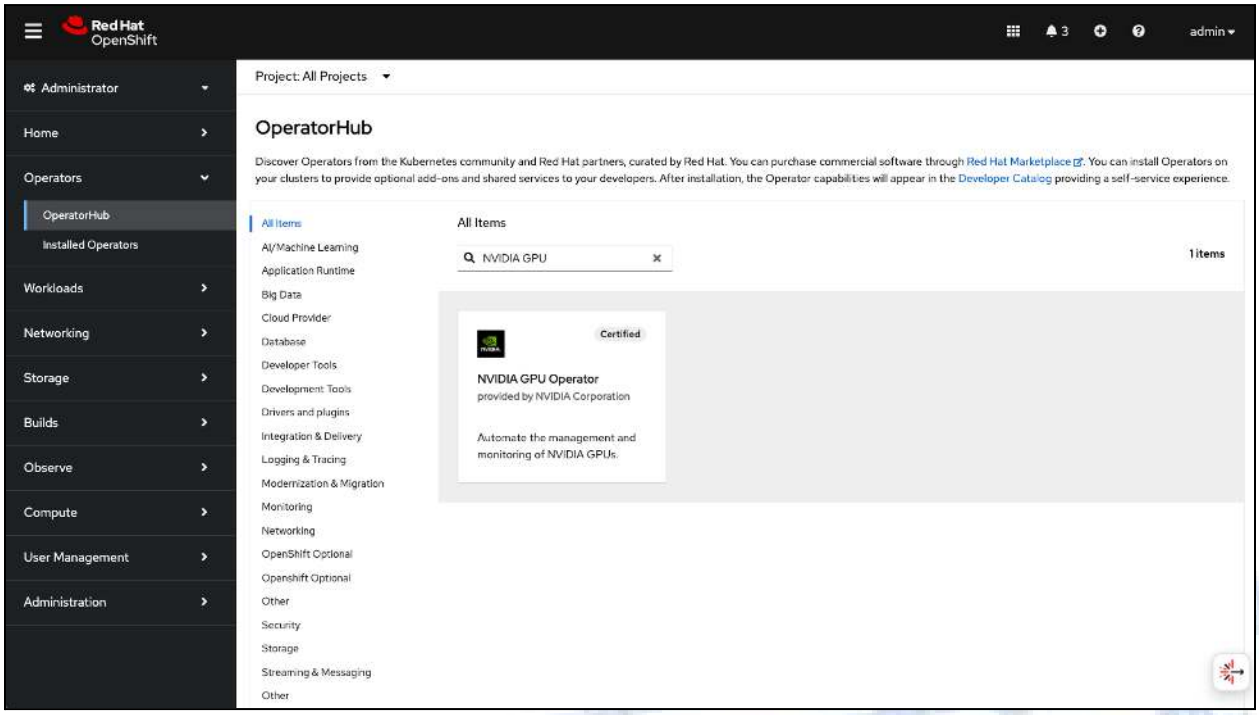


SSH into the Openshift bastion node and run the following command to ensure that ocpgpu.cdpkvm.cldr host (with GPU card installed) has pci-10de.present=true field in the node specification. This indicates the presence of an Nvidia GPU card in this particular worker node.

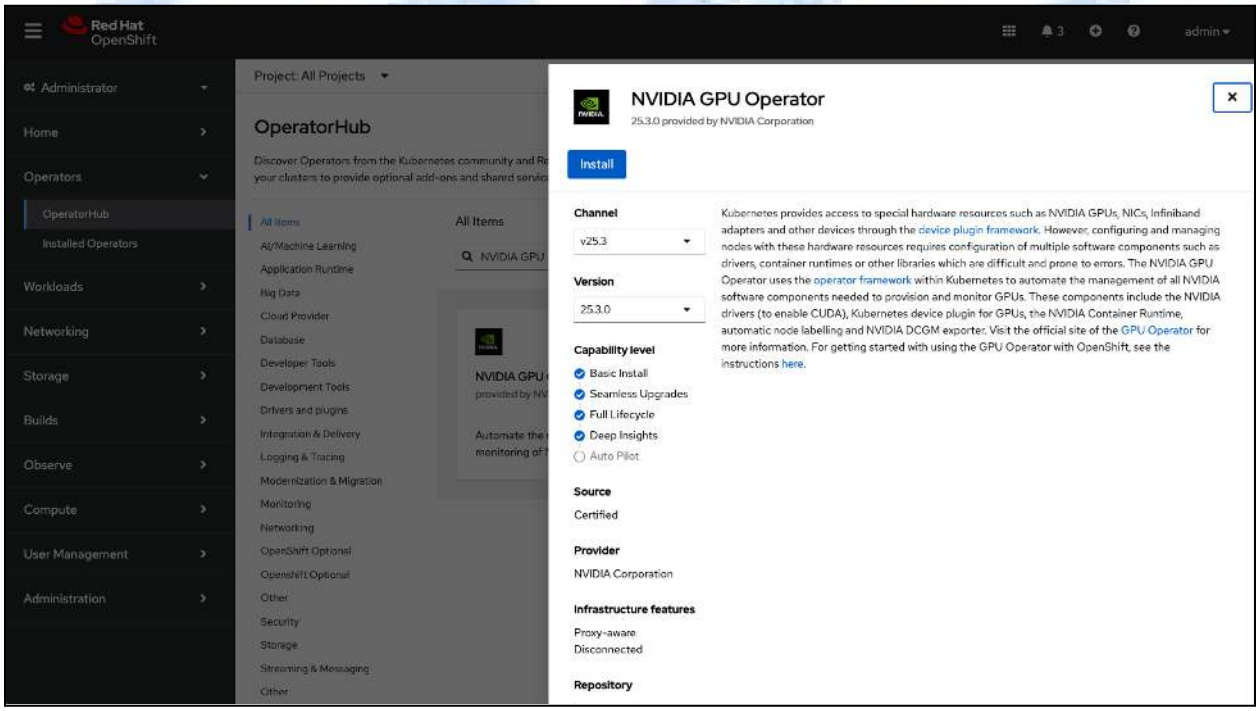
```
[root@bastion ~]# oc describe node c240m4-01.redhat.local | grep pci-10de.present
                    feature.node.kubernetes.io/pci-10de.present=true

[root@bastion ~]# oc describe node worker0.cdppvcds.redhat.local | grep
    pci-10de.present
                    feature.node.kubernetes.io/pci-10de.present=true
```

Install NVIDIA GPU Operator

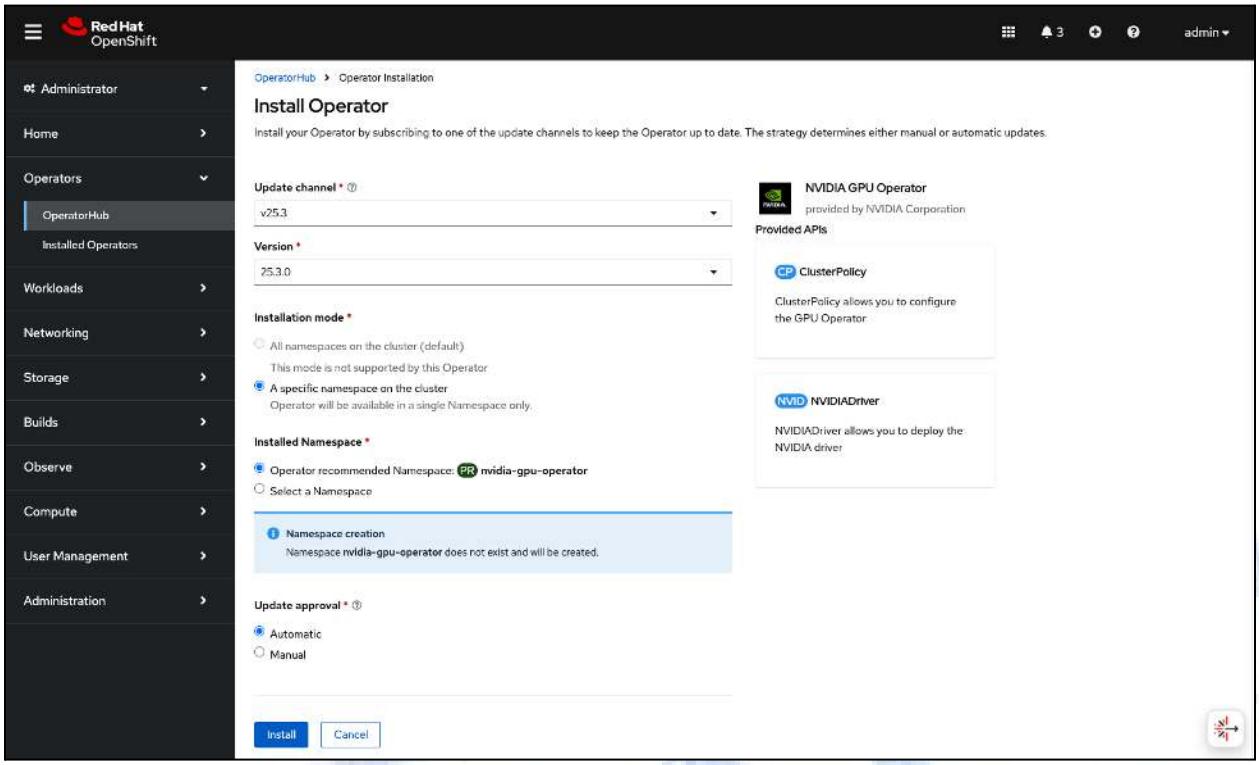
STEP 1. In the ***Openshift console***, navigate to ***Operators > OperatorHub***. Search for the ***NVIDIA GPU*** Operator. Click on ***NVIDIA GPU Operator***.



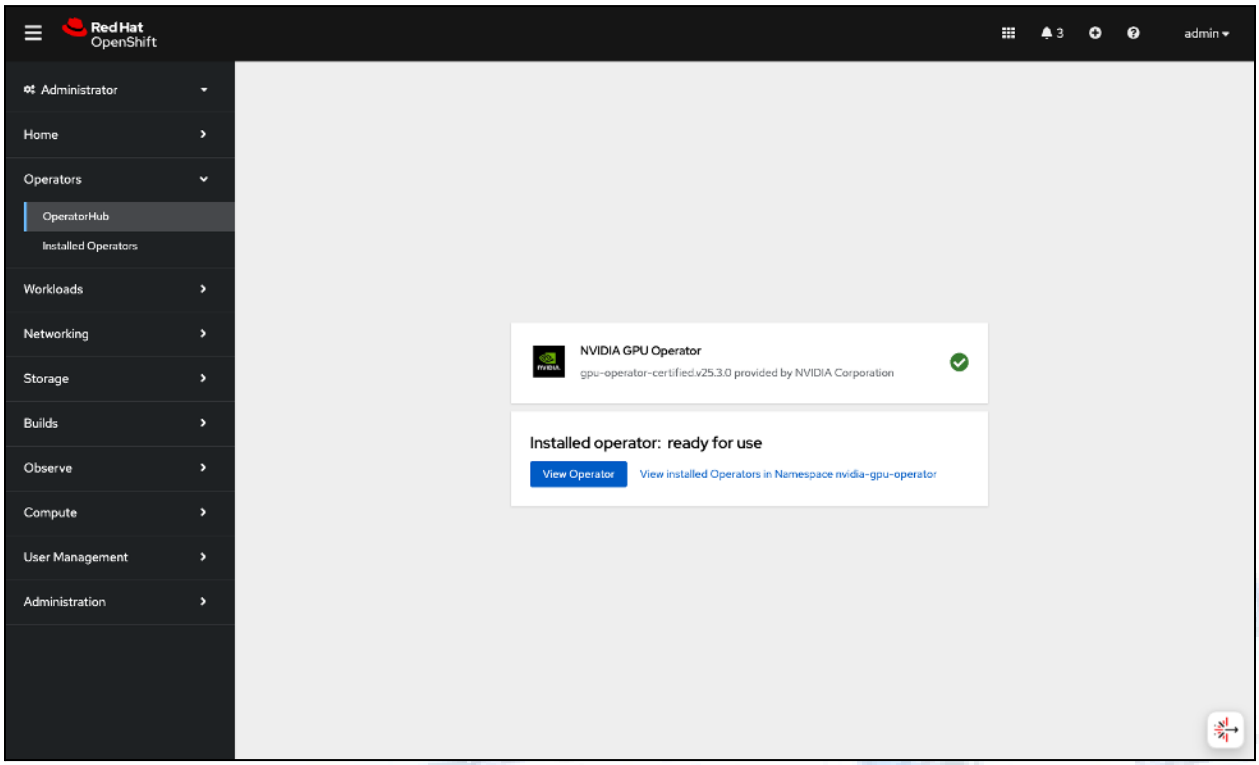
STEP 2. Click on ***Install***.



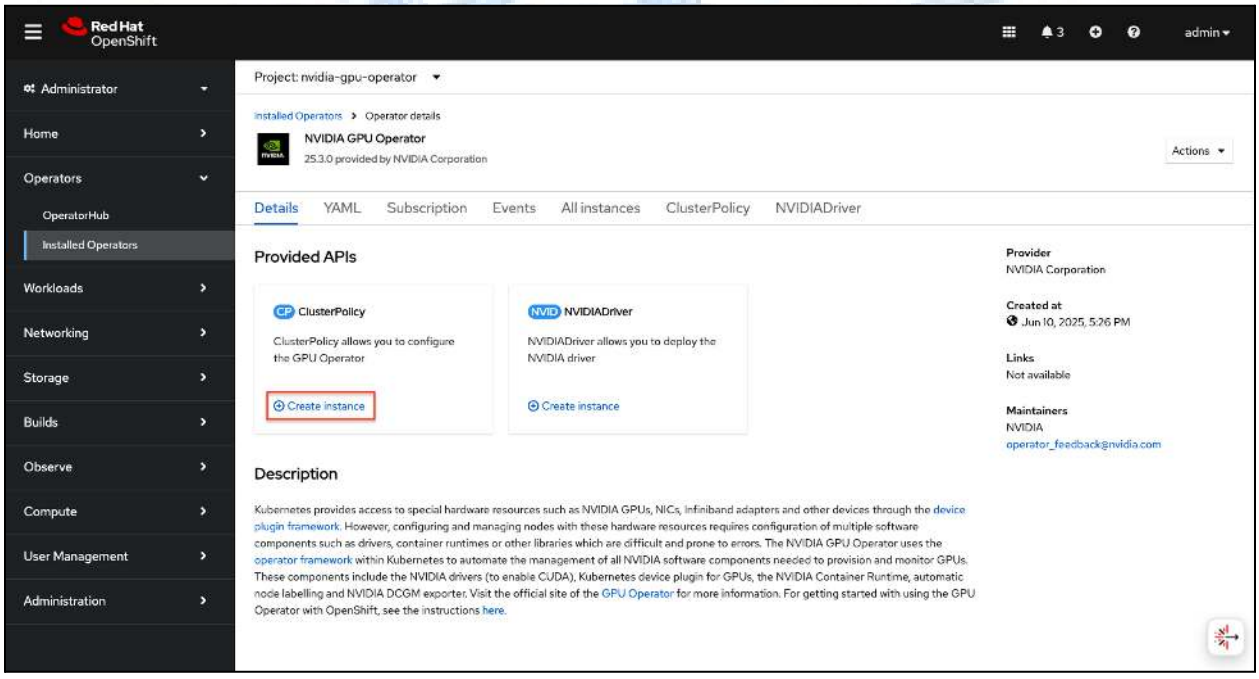
STEP 3. Click on ***Install***.



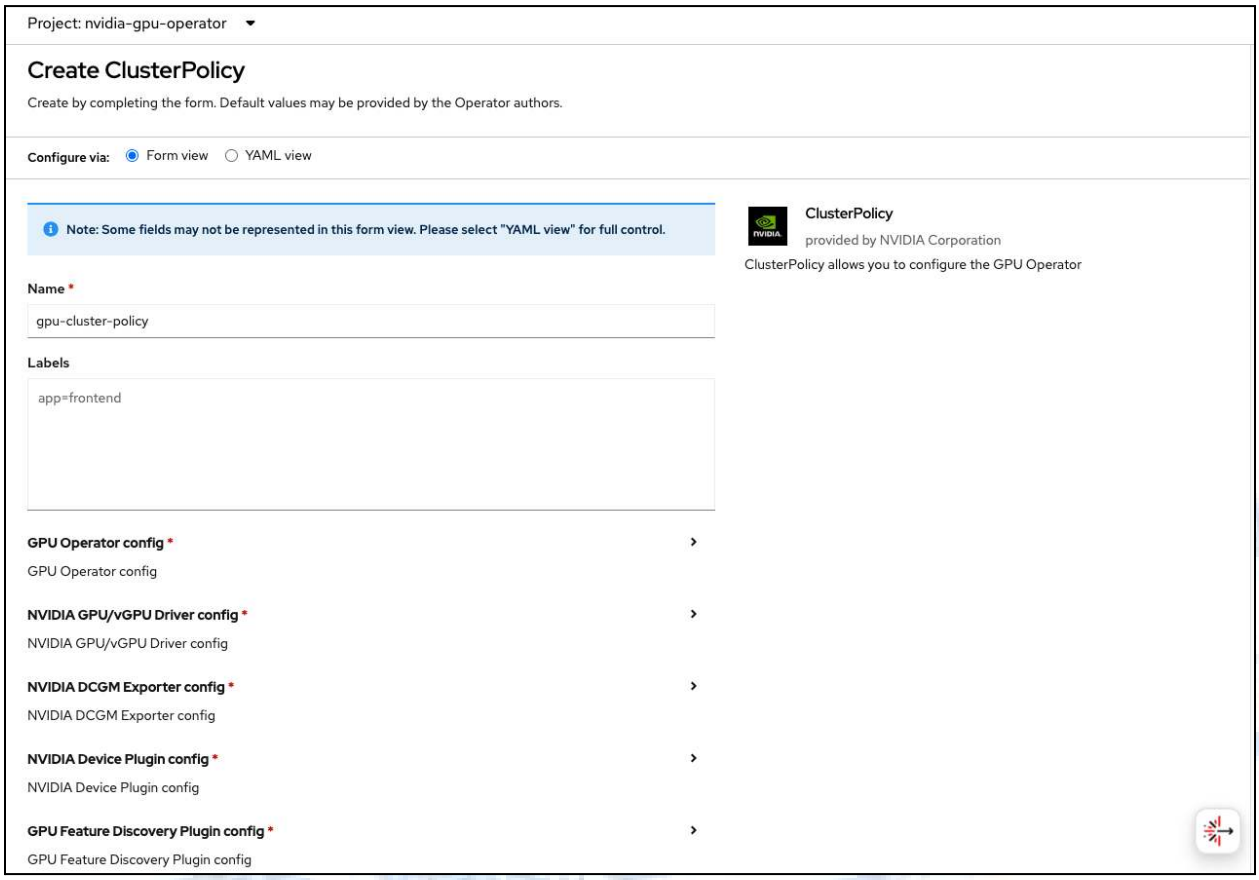
STEP 4. Once the operator completes the installation, click on ***View Operator***.

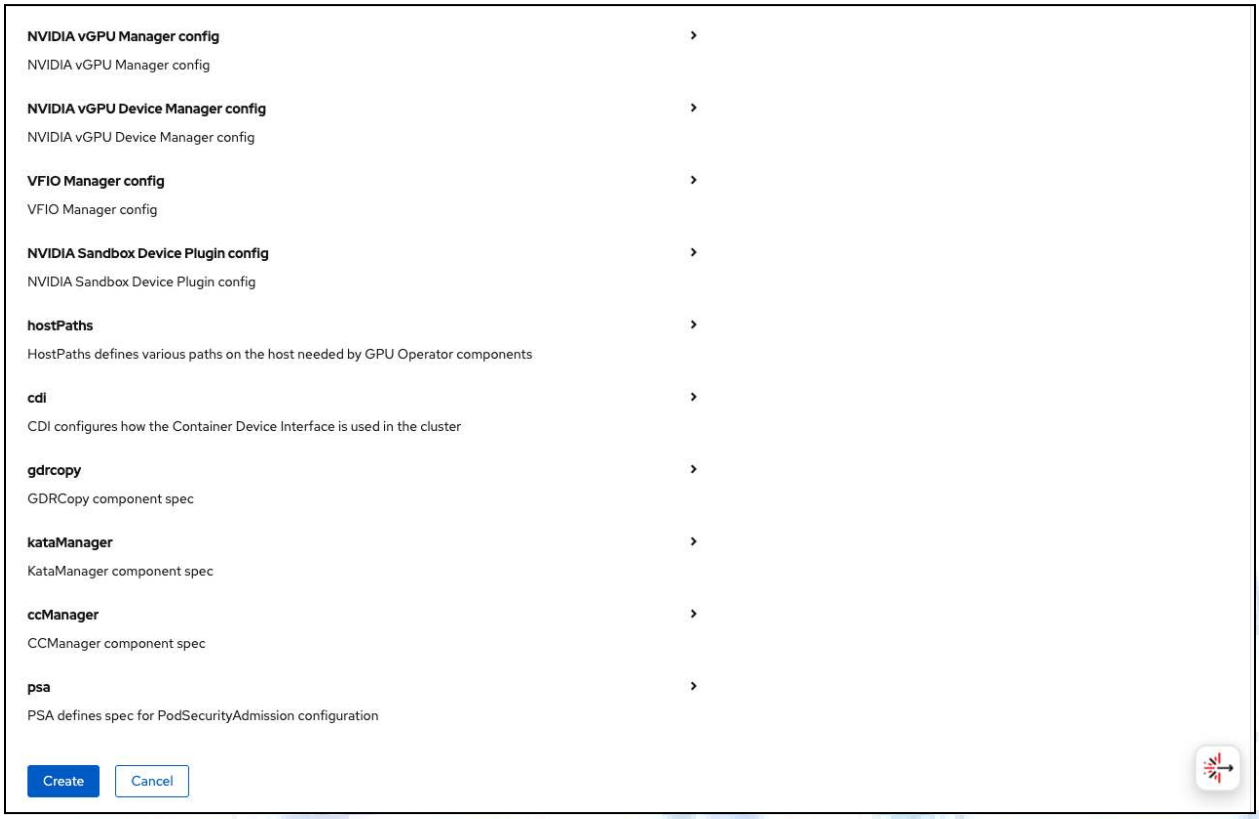


STEP 5. Click on ***Create instance*** under the ***ClusterPolicy***.

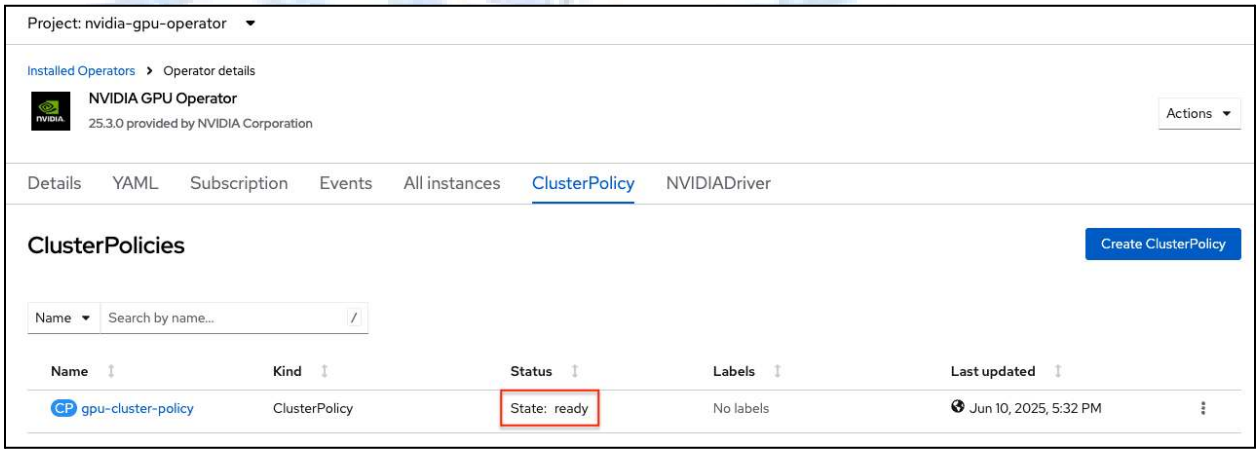


STEP 6. Scroll to the bottom of the screen and click on ***Create***.





STEP 7. The policy installation is complete when the status shows ***ready***.



STEP 8. SSH into the Openshift bastion node and run the following command to verify the successful installation of the operator and the clusterPolicy.

```
[root@ocpbastion ~]# oc get pods,daemonset -n nvidia-gpu-operator
NAME                                                  READY   STATUS      RESTARTS   AGE
```

```
pod/gpu-feature-discovery-l7pqc                          1/1     Running     0          6m46s
pod/gpu-operator-765ff6c665-mznvk                        1/1     Running     0          7m50s
pod/nvidia-container-toolkit-daemonset-6brmr             1/1     Running     0          6m47s
pod/nvidia-cuda-validator-8brpz                          0/1     Completed   0          2m50s
pod/nvidia-dcgm-5txs7                                    1/1     Running     0          6m47s
pod/nvidia-dcgm-exporter-wj4dg                           1/1     Running     0          6m46s
pod/nvidia-device-plugin-daemonset-b2k5x                 1/1     Running     0          6m47s
pod/nvidia-device-plugin-validator-29b9g                 0/1     Completed   0          2m32s
pod/nvidia-driver-daemonset-48.84.202208152344-0-cxsld   2/2     Running     0          6m47s
pod/nvidia-mig-manager-kqzk5                             1/1     Running     0          87s
pod/nvidia-node-status-exporter-2xsdc                    1/1     Running     0          6m47s
pod/nvidia-operator-validator-f2zdm                      1/1     Running     0          6m47s

NAME                                                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
SELECTOR                                                                                                         AGE
daemonset.apps/gpu-feature-discovery                      1         1         1       1            1
nvidia.com/gpu.deploy.gpu-feature-discovery=true                                                                 6m46s
daemonset.apps/nvidia-container-toolkit-daemonset         1         1         1       1            1
nvidia.com/gpu.deploy.container-toolkit=true                                                                     6m47s
daemonset.apps/nvidia-dcgm                                1         1         1       1            1
nvidia.com/gpu.deploy.dcgm=true                                                                                  6m47s
daemonset.apps/nvidia-dcgm-exporter                       1         1         1       1            1
nvidia.com/gpu.deploy.dcgm-exporter=true                                                                         6m46s
daemonset.apps/nvidia-device-plugin-daemonset             1         1         1       1            1
nvidia.com/gpu.deploy.device-plugin=true                                                                         6m47s
daemonset.apps/nvidia-driver-daemonset-48.84.202208152344-0   1         1         1       1            1
feature.node.kubernetes.io/system-os_release.OSTREE_VERSION=48.84.202208152344-0,nvidia.com/gpu.deploy.driver=true   6m47s
daemonset.apps/nvidia-mig-manager                         1         1         1       1            1
nvidia.com/gpu.deploy.mig-manager=true                                                                           6m46s
daemonset.apps/nvidia-node-status-exporter                1         1         1       1            1
nvidia.com/gpu.deploy.node-status-exporter=true                                                                  6m47s
daemonset.apps/nvidia-operator-validator                  1         1         1       1            1
nvidia.com/gpu.deploy.operator-validator=true                                                                    6m47s
```

STEP 9. Verify that the NVIDIA GPU card can be consumed by a newly provisioned CUDA pod.

```
[root@ocpbastion ~]# oc new-project nvidia-test

[root@ocpbastion ~]# cat << EOF | oc create -f -
>
> apiVersion: v1
> kind: Pod
> metadata:
>   name: cuda-vectoradd
> spec:
>  restartPolicy: OnFailure
>  containers:
>  - name: cuda-vectoradd
>    image: "nvidia/samples:vectoradd-cuda11.2.1"
>    resources:
>      limits:
>        nvidia.com/gpu: 1
> EOF
```

```
pod/cuda-vectoradd created

[root@ocpbastion ~]# oc get pods
NAME             READY   STATUS      RESTARTS   AGE
cuda-vectoradd   0/1     Completed   0          13s

[root@ocpbastion ~]# oc logs cuda-vectoradd
[Vector addition of 50000 elements]
Copy input data from the host memory to the CUDA device
CUDA kernel launch with 196 blocks of 256 threads
Copy output data from the CUDA device to the host memory
Test PASSED
Done

[root@ocpbastion ~]# oc describe pod cuda-vectoradd | grep -i Node:
Node:         ocpgpu.ocp4.cdpkvm.cldr/10.15.4.185

[root@ocpbastion ~]# oc exec -it
nvidia-driver-daemonset-48.84.202208152344-0-cxsld -- nvidia-smi
Defaulted container "nvidia-driver-ctr" out of: nvidia-driver-ctr,
openshift-driver-toolkit-ctr, k8s-driver-manager (init)
Fri Aug 26 06:07:36 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.82.01    Driver Version: 470.82.01    CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA A100-PCI...  On   | 00000000:07:00.0 Off |                    0 |
| N/A   28C    P0    33W / 250W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+

[root@ocpbastion ~]# oc describe pod
nvidia-driver-daemonset-48.84.202208152344-0-cxsld  | grep Node:
Node:                 ocpgpu.ocp4.cdpkvm.cldr/10.15.4.185
```

Note: Prepare CDP Private Cloud Base for the Private Cloud Data Services installation: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-pvcbase-checklist.html

Note: Use this checklist to ensure that your OpenShift Container Platform (OCP) is configured and ready for installing CDP Private Cloud Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-ds-checklist.html

Note: Use this checklist to ensure that you have all the requirements for Cloudera Data Warehouse in CDP Private Cloud Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-cdw-checklist.html

Note: Use this checklist to ensure that you have all the requirements for Cloudera Machine Learning in CDP Private Cloud Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-cml-checklist.html

Note: Use this checklist to ensure that you have all the requirements for Cloudera Data Engineering in CDP Private Cloud Data Services: https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-cml-checklist.html

Prerequisites Checklist (DataServices OCP 1.5.5 - Pre Install Checklist):

Section	Details
Checklist	Prior to attempting the install wizard, ensure the following tasks are completed:
ClouderaManager version 7.11.3 CHF6	- 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) - You have the account and password for LDAP Bind user account - Base Cluster & CM Agents should be TLS secured **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 - CM is setup with Kerberos admin account (cloudera-scm account & password)
Runtime 7.1.7 (SP3) or 7.1.8 (CHF22) or 7.1.9 (CHF6)	- Minimum Required: Zookeeper, HDFS, Ozone, HBase, Hive Metastore, Kafka, Solr-Infra, Ranger, Atlas, YARN (optional for CDW, but required for Spark pushdown in CDE, CML) Can Omit Ozone if purely a CML implementation Can Omit Atlas if purely CDE All Base components configured with TLS All Base components configured with Kerberos CDW - HMS db is configured to allow TLS connections (set to allow TLS, but not forced to require it) **New in 1.5.5**: CDW - Base HMS, mTLS can be used instead of user/password

DNS	- **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 OCP 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 "cdp-dev", then within that create another called "apps", then create an "A record" inside that folder, its name is "*", its IP will be the IP of the host you will install Ingress LB Please know that the concept of "folders" is an AD thing; other DNS providers may be more flexible in how you define the wildcard record. The end result must be verified by `dig` (or `nslookup`) Test this using the `dig` utility e.g., `$ dig foobar.apps.cdp-dev.company.com` Look for the "ANSWER" section in the result. This shows the host that you want your OCP server installed DNS needs to be the primary resolver, do not use entries in `/etc/hosts` (except for localhost)
OCP 4.14	- Requirements for CSI compliant Block Storage provider such as ODF (used to be branded as OCS) or Portworx - Ensure separate storage classes for the control plane and compute clusters. Both storage classes must be provisioned from Persistent Volumes, from your CSI Provider - `/etc/resolv.conf` must NOT have more than 3 "nameserver" entries - Ensure net interfaces are configured to a maximum transmission unit size (mtu) not smaller than 1450 (typical mtu is 1500), required by the CNI plugin - Ensure that the OpenShift cluster has access to a Container Image file registry from where it retrieves the container images for deployment - Ensure DNS and Reverse DNS are set up between OpenShift container hosts and CDP Private Cloud Base. This is required for obtaining Kerberos ticket-granting tickets - Ensure OpenShift application hostnames can be accessed from outside the cluster. Test this by creating an ingress point on the target cluster - Ensure access to the OpenShift Kubeconfig file, cluster administrator privileges, and sufficient expiry time for you to complete your installation **-- cluster-admin role is required**

	- When using a load balancer for your OpenShift Container Platform external API, allow WebSocket traffic in addition to HTTPS. The load balancer must allow WebSockets on port 80. Also, set the load balancer server timeout to 5 minutes - Ensure the NTP clock in CDP Private Cloud Base is in sync with the time configured in the OpenShift cluster. This is important if your setup does not have Internet access - Ensure OCP cluster is configured to run applications in multiple namespaces with the same domain name
CML	- If using CML, NFS Provisioner is required. NFS version 4.0 is required when in use.
Bits	- If a customer is air gapped, download all CDP-PvC bits and stage them behind an HTTP server. This is 100GB+ of content - Customer (or Cloudera employee) MUST have a valid license key that includes an entitlement for PrivateCloud - Ensure the OpenShift cluster has access to a Docker Container registry for retrieving container images for deployment - When setting up a local registry for the OpenShift cluster (recommended), follow instructions to copy Cloudera container images from the Cloudera hosted registry to this local registry. For more information, see Installing in air gap environment.
3rd Party Software	- VMware NSX-T can create blocking firewall rules. See: JIRA CDPVC-686
Ingress Certificates	- It's usually better to have a customer CA signed Ingress LB cert, but not required as OCP will generate self-signed certs (self-signed will need to be pre-trusted on a user's machine, or browser will show "insecure" TLS connection) - This cert must include 2 Subject Alternative Names (SubjAltNames). If the wildcard format is not allowed, let the installer generate its own self-signing **Example**: DNS.1 = `*.apps.cdp-dev.company.com` (may not be allowed at customer, but required if they need to have their own CA sign the cert; otherwise let RKE generate the certs) DNS.2 = `apps.cdp-dev.company.com`

CAI Workbench & CDE Virtual Cluster Certificates	- **CAI Workbench**: If installing in a network domain requiring strict host checking, HSTS, use TLS for CAI Workbenchs. This cert must include SubjAltNames **Example**: DNS.1 = `*.xxxx-xxxx.apps.cdp-dev.company.com` (where `xxxx-xxxx` is the CML workspace ID) - **CDE Virtual Cluster**: For HSTS, same applies for CDE, but CDE includes a utility to generate RKE signed certs for each virtual cluster. If making your own cert for a CDE virtual cluster, this cert must include SubjAltNames **Example**: DNS.1 = `*.xxxx-xxxx.apps.cdp-dev.company.com` (where `xxxx-xxxx` is the CDE virtual cluster ID)

Pre-Flight Checklist for OCP:

Task	Details
Validate OCP environment and prerequisites	
1. Validate OpenShift platform version	- **Objective**: Ensure that the OpenShift version matches the specified version during construction. - **Steps**: 1. Log in to the steppingstone server as a root user. 2. Run the command: `oc version`. - **Expected Outcome**: The version listed for "oc" should be **4.8**.
2. Validate DNS configuration	- **Objective**: Ensure OpenShift nodes are accessible from CDP nodes via the configured DNS. - **Steps**: 1. From each CDP node, run: `ssh <username>@<hostname of OCP node>`. 2. Enter the password for each node. - **Expected Outcome**: Login to the specified OCP node is successful.

3. Validate Storage classes configuration	- **Objective**: Verify that the OpenShift storage manifest file is using NFS. - **Steps**: 1. Log in as root on the steppingstone server. 2. Run: `oc get pvc`. - **Expected Outcome**: "nfs-pv" is listed in the "VOLUME".
4. Validate Kubeconfig and kubectl	- **Objective**: Ensure the CDP service account has the necessary permissions. - **Steps**: 1. Log in to the steppingstone server as root. 2. Run: `kubectl get sa`. - **Expected Outcome**: CDP service account name is displayed. 3. Run: `kubectl get clusterrolebinding`. - **Expected Outcome**: "ClusterRole/cluster-admin" is associated with the CDP service account. **Note**: Verify the service account name for CDP.
5. Validate Route admission policy	- **Objective**: Ensure root privileges are correctly set. - **Steps**: 1. On the steppingstone server, as root, run: `cat /etc/group`. 2. Check that only "root" and "admin" belong to the root group. 3. Run: `cat /etc/passwd`. - **Expected Outcome**: The OpenShift installation username should be listed. **Note**: Confirm the username for OpenShift installation. 4. On OCP nodes (Master/Worker/Bootstrap), run `cat /etc/group`. - **Expected Outcome**: Only root users should belong to the root group.
6. Validate clock synchronization with NTP	- **Objective**: Verify that the clock time on the CDP cluster is in sync with the NTP server. - **Steps**: 1. On each CDP node, execute the command to check NTP sync. - **Expected Outcome**: The line marked "*" should display the domain name or IP address of the TEPcube standard NTP server. **Reference**: Check the "RHEL" sheet in the infrastructure design document for NTP server details.

7. Validate NFS storage for CDE	- **Objective**: Ensure that the NFS storage class is properly configured to allow read and write access from the CDE side. - **Steps**: 1. Login as root on the steppingstone server. 2. Run: `oc get pvc`. - **Expected Outcome**: "RWX" should be specified under "ACCESS MODES".

****

Installing CDP Private Cloud Data Services using OCP

Follow the steps here to install CDP Private Cloud Data Services with the ***Openshift Container Platform (OCP)***.

Follow the steps outlined below to add hosts to be part of the Cloudera Private Cloud Data Services cluster and the install OCP (Openshift Container Platform) through either internet or air gapped method.

https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-steps.html
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-airgap.html
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/index.html

Note: We will be installing CDP Private Cloud Data Services via the internet method.

Note: For more details on dedicating OCP node for specific workload type please visit:
https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-ocp-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.

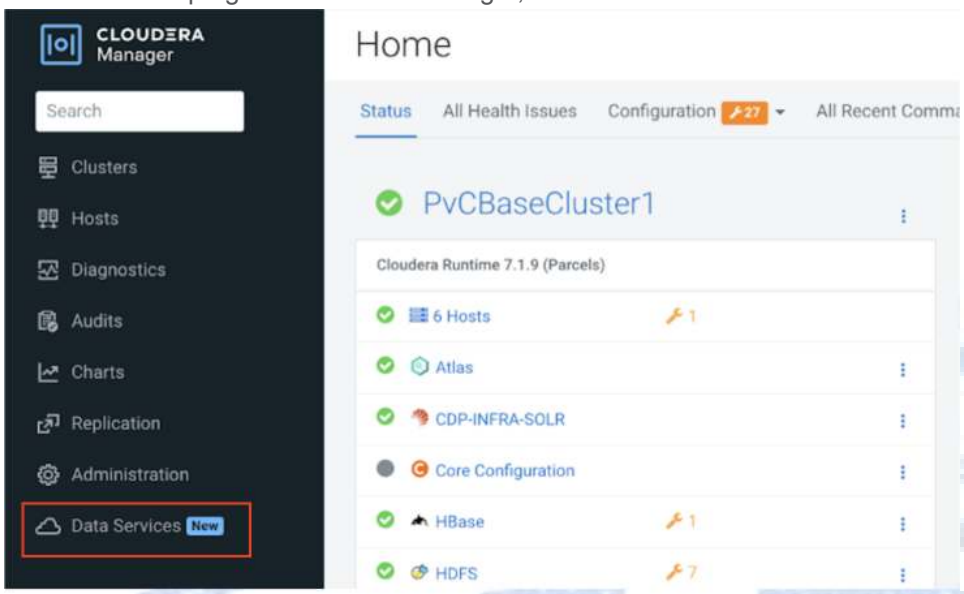
Latest OCP Supported Version Of Cloudera-Manager is *7.11.3 CHF9.1*

****

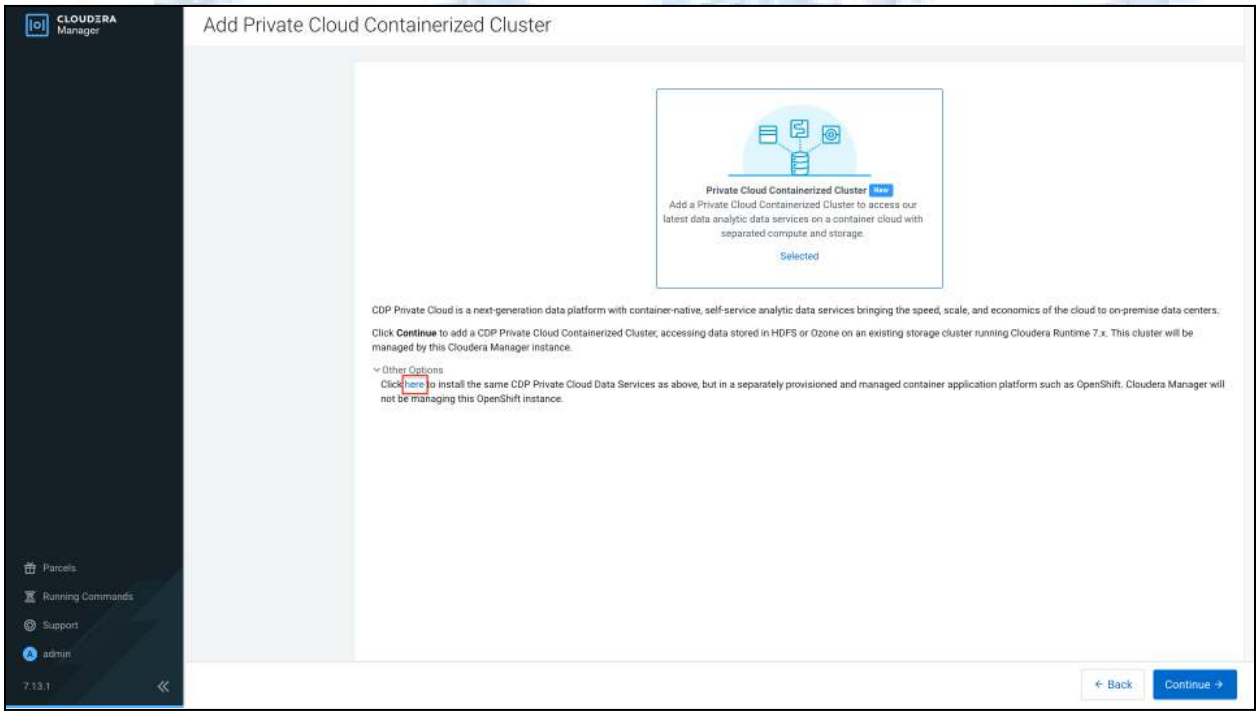
Installing OCP Cluster

Follow the steps in this topic to install CDP Private Cloud Data Services with the Openshift Container Platform (OCP).

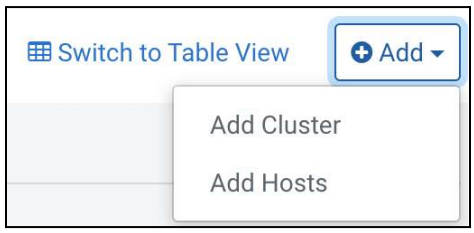
1) In the ***Cloudera Manager WebUI*** console, go to the ***Data Services*** page by clicking on the ***Data Services*** link on the Pane located at the Left Hand side of the browser window. Alternatively, you can also click ***(+) Add > Add Cluster*** at the top right in Cloudera Manager, then select ***Private Cloud Containerized Cluster*** as the cluster type.

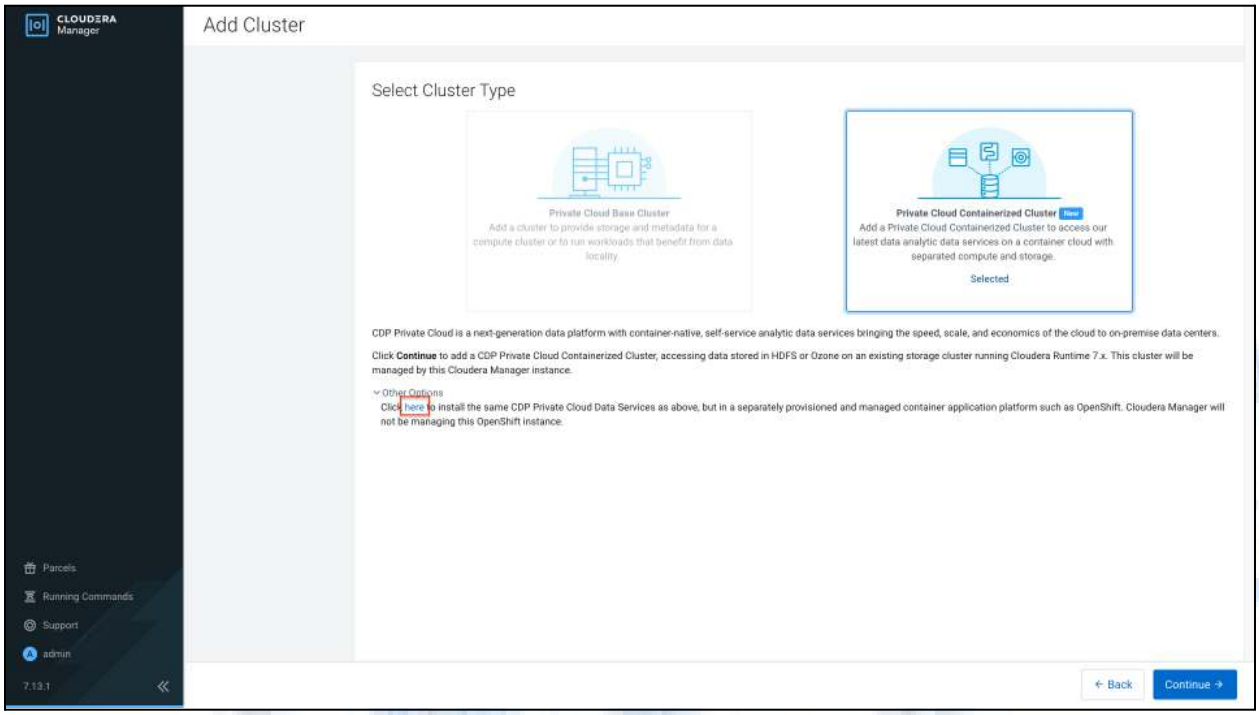


2) The ***Add Private Cloud Containerized Cluster*** page appears. Click ***here*** on the page.



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 ***here***.



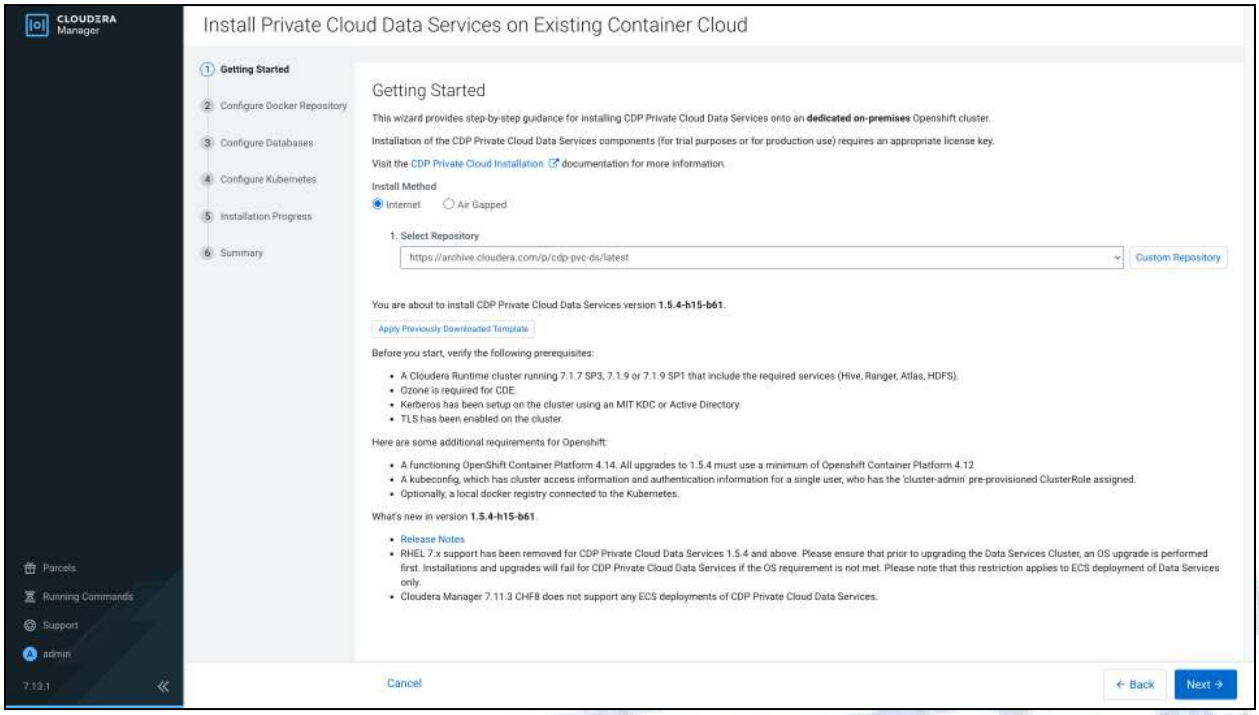


Step. Getting started

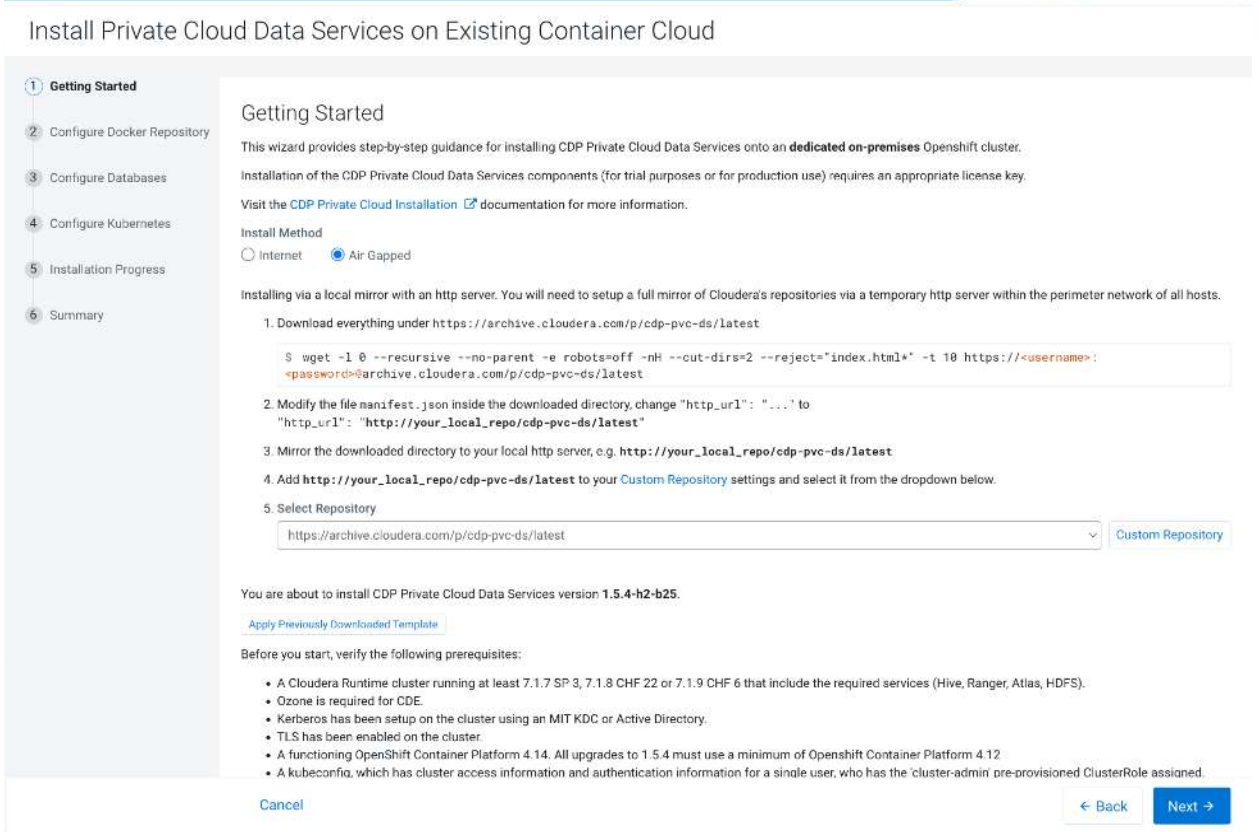
3) On the ***Getting Started*** page of the installation wizard, select either ***Internet*** or ***Air Gapped*** as the Install Method. ***(We are going with the Internet Installation method)***

Note: Verify the prerequisites for the version that you're installing and then click ***Next***.

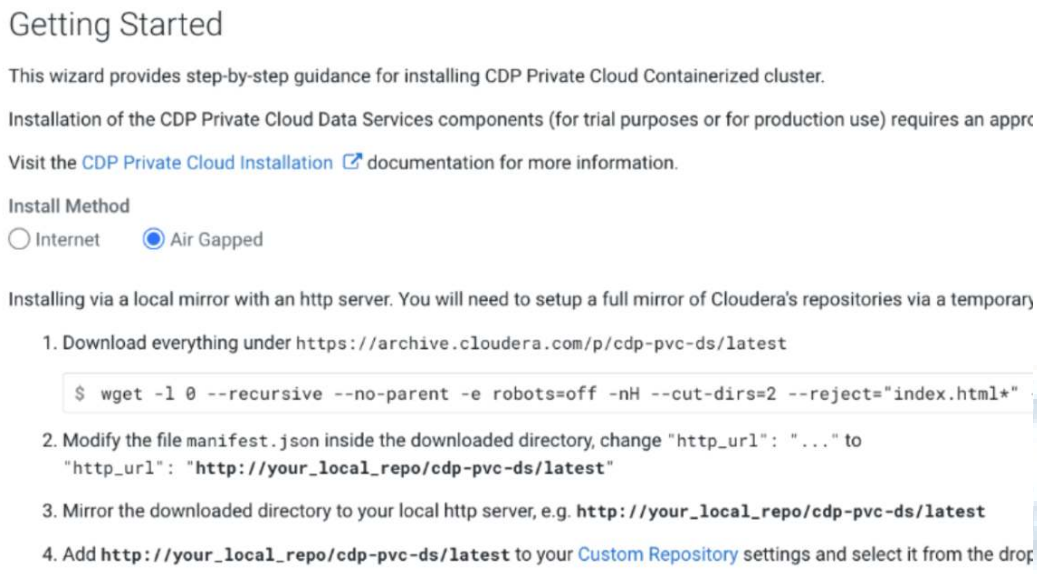
4) If you select the ***Internet Install*** Method option on the ***Getting Started*** page, images are copied over the internet from the ***Cloudera repository***. For this, select the ***Internet*** as the install method. Select ***Repository***. Click on ***Next***. (To use a ***custom repository*** link provided to you by Cloudera, click ***Custom Repository***): ***(We are using here Internet Installation Method only)***



5) If you select the ***Air Gapped*** option, images are copied from a local http mirror you have set up in your environment. Click on the ***Air Gapped*** radio Button. **(Skip this step, as we have chosen Internet installation method in previous steps to Install)**



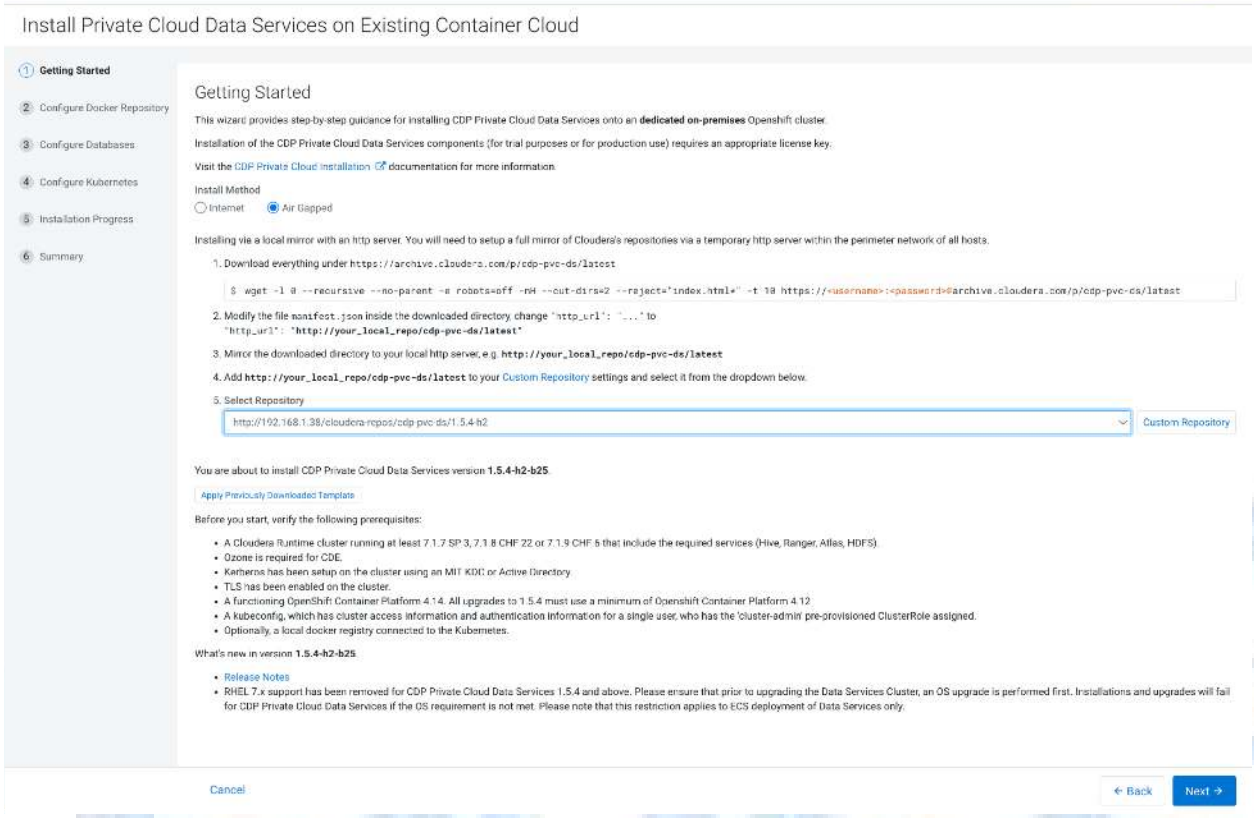
6) When you select the ***Air Gapped*** install option, extra steps are displayed. Follow these steps on the ***cldr-mngr*** node (our bits server), to download and mirror the Cloudera archive URL using a local HTTP server:
(For installing via a local mirror with an http server. You will need to set up a full mirror of Cloudera's repositories via a temporary HTTP server within the perimeter network of all hosts.) ***(We are using Internet Installation Method)***




```
[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*** on the CM-UI. Add ***http://your_local_repo/cloudera-repos/cdp-pvc-ds/1.5.5-h2/*** as a custom repository. Click on ***Save Changes***. ***(We are using here Air Gapped Installation Method only)***

8) Click the ***Select Repository*** drop-down and select ***http://your_local_repo/cloudera-repos/cdp-pvc-ds/1.5.5-h2/*** ***(We are using here Air Gapped Installation Method only)***



9) Click ***Next***.

Step. Configure Docker Repository

10) On the ***Configure Docker Repository*** page, you must select one of the ***Docker repository*** options. There are several options for configuring a Docker Repository. For more information about these options, see Docker repository access. ***(Select Cloudera's default Docker Repository option)***

11) If you select Use a ***custom Docker Repository*** option, enter your ***local Docker Repository*** in the ***Custom Docker Repository field*** in the following format: ***[*DOCKER REGISTRY*]/[*REPOSITORY NAME*]***. Alternatively, you can use ***Cloudera's default Docker Repository*** if you are setting up CDP Private Cloud in non-production environments. ***(We are using here Cloudera's default Docker Repository option)***

Note: Verify the prerequisites for the version that you're installing and then click ***Next***.

1. **Use a custom Docker Repository** - Copies all images (Internet or Air Gapped) to the embedded registry.
2. **Use Cloudera's default Docker Repository** - Copies images from Internet to the embedded registry. This uses the default repository that is in manifest.json. Cloudera's default Docker Repository option can be selected only if you have selected ***Internet*** as the ***install method***.

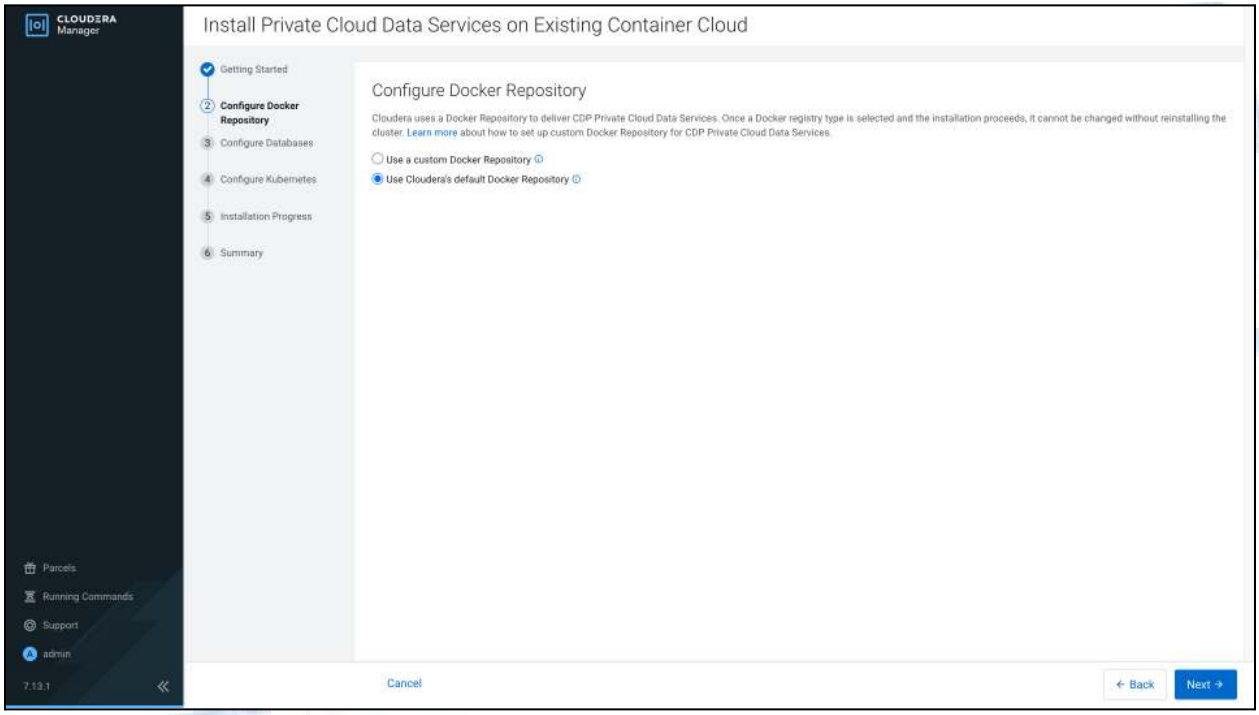
12) You can follow these steps to prepare your Docker Repository from a machine that is running Docker locally and has access to all the Docker images either directly from Cloudera or a local HTTP mirror in your network. ***(Skip this step, as we are using here Cloudera's default Docker Repository option)***

a) Click ***Generate the copy-docker script*** on the wizard or ***download*** the script file.

b) Log in to your custom Docker Registry and run the script using the following commands.
docker login <your_custom_registry> -u <user_with_write_access>
c) *bash copy-docker.txt*
Note: This command downloads 100+ Docker images and it will take some time to download.
d) Enter your Docker user name and password.
e) Click ***Choose File*** to upload your Docker certificate.
f) Click ***Next***.
g) Click ***Continue***.

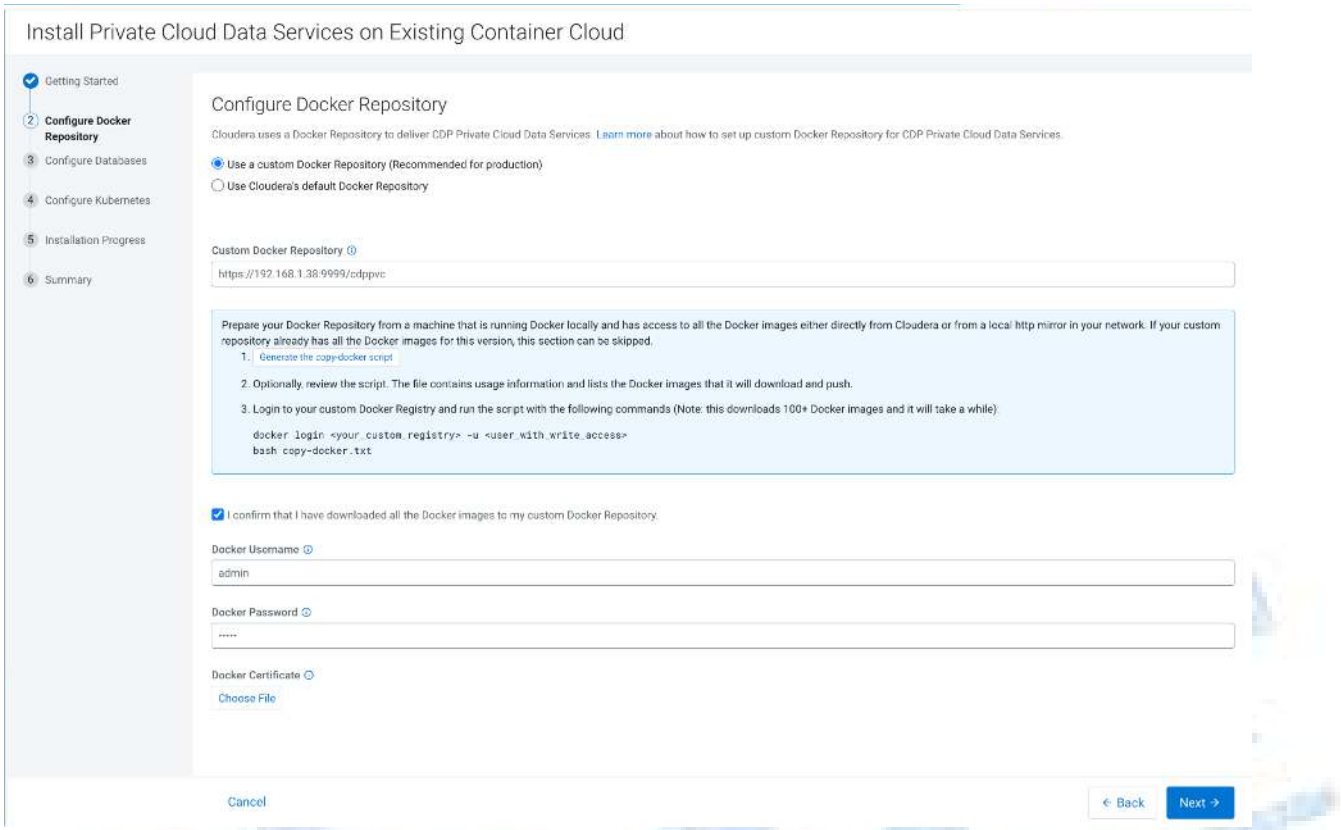
13) You can follow these steps to prepare your Docker Repository from a machine that is running Docker locally and has access to all the Docker images either directly from Cloudera or a local HTTP mirror in your network. ***(Skip this step, as we are using here Cloudera's default Docker Repository option)***

Cloudera default Docker Repository
This option requires that cluster hosts have access to the internet and you have selected Internet as the install method.



a. Screenshot for Use Cloudera's Docker Repository Option

Use a 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.



b. Screenshot for Use Custom Docker Repository Option

For using the Custom Docker Repository, you must enter the following options:

Option	Value
Custom Docker Repository:	Enter the URL for your Docker Repository **(Nexus Docker Repository)**
Docker Username:	Enter the username for the **Nexus** Docker Repository
Docker Password:	Enter the password for the **Nexus** Docker Repository
Docker Certificate:	Upload the ***nexus.crt*** file generated while configuring TLS for Nexus

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.

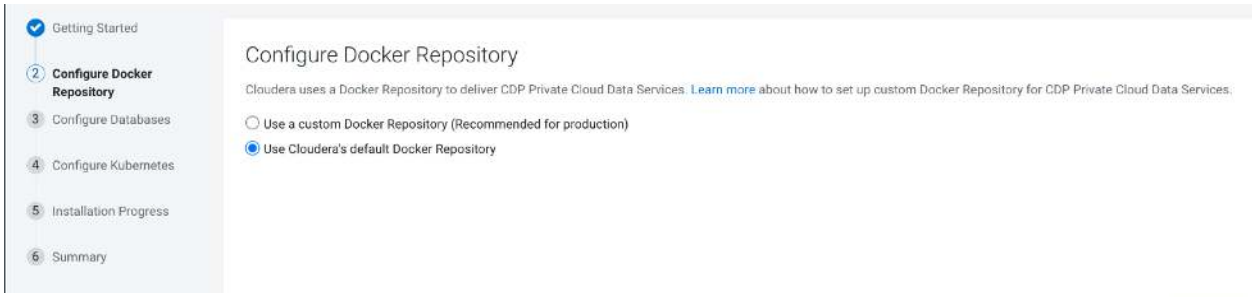
Click the ***Generate the copy-docker script*** button to generate and download a script that copies the Docker images from Cloudera, or (for air-gapped installation) from a local http mirror in your network.
Run the script from a machine that is running Docker locally and has access to the Docker images 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.

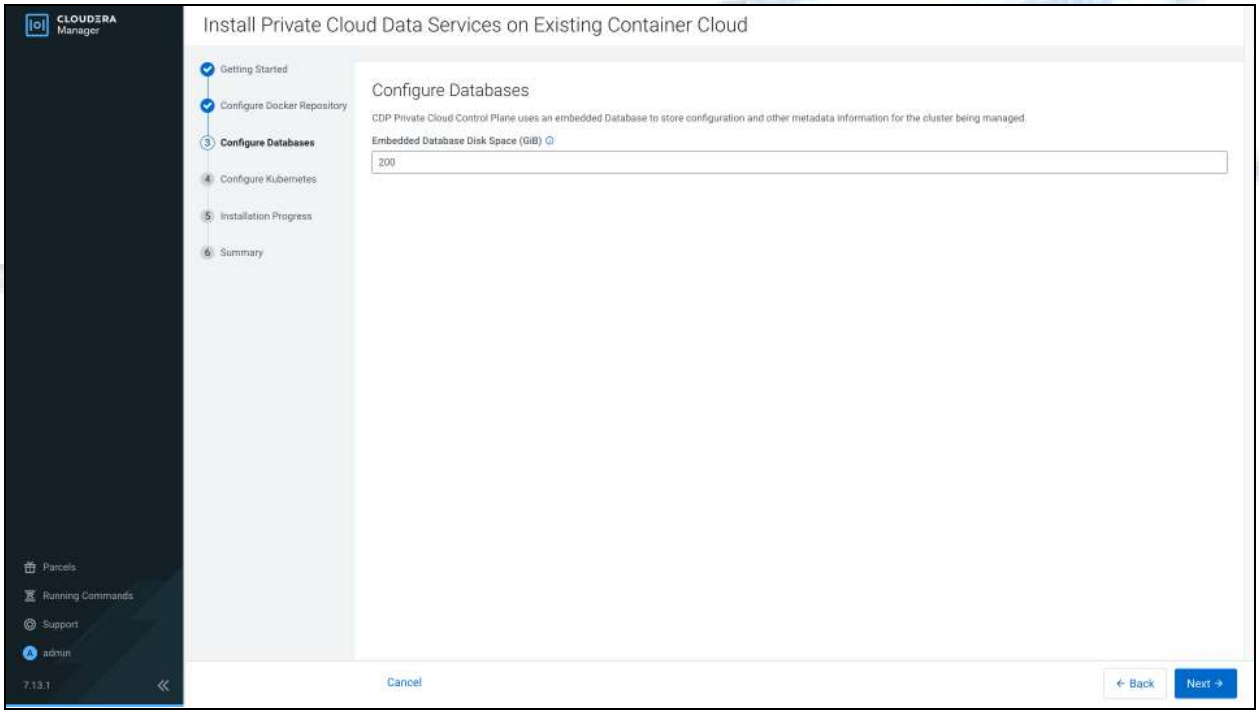
Note: ***Cloudera's Repository*** option is best suited for proof-of-concept, non-production deployments or deployments that do not have security requirements that disallow internet access. This option requires that cluster hosts have access to the internet, and an installation method selected as ***Internet***.

14) On the ***Configure Docker Repository*** page, select ***Use Cloudera's default Docker Repository*** option. Click ***Next***.



Step. Configure Databases

15) On the ***Configure Databases*** page, edit size for the ***Embedded Database Disk Space*** ***(we will keep the default value 200)***. Click ***Next***.



Step. Configure Kubernetes

16) On the ***Configure Kubernetes*** 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 LDAP Domain Services with "***REDHAT.LOCAL***" as domain name. Created a wildcard entry "*.apps.redhat.local". Click ***Continue***.

- On the ***Configure Kubernetes*** page, enter your Kubernetes, Docker, database, and vault information.
- Upload a ***Kubernetes configuration*** (***kubeconfig***) file from your existing environment. You can obtain this file from your OpenShift Container Platform administrator. Ensure that this kubeconfig has

permissions to create Kubernetes namespaces.
kubeconfig-pvccdpds

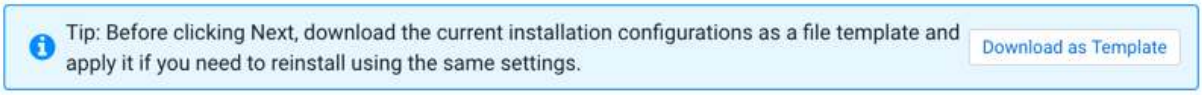
- In the Kubernetes Namespace field, enter the ***Kubernetes namespace*** that you want to use with this CDP Private Cloud deployment. Kubernetes virtual clusters are called namespaces. ***(we will keep the default value as 'cdp')*** For more information, see Kubernetes namespaces
- Enter your ***Vault*** information and upload a CA certificate. Cloudera recommends that you use an external Vault for production environments. Enter the Vault address and token, and upload a CA certificate. ***(we will use the embedded vault)***
- Enter a ***Storage Class*** to be configured on the Kubernetes cluster. CDP Private Cloud uses Persistent Volumes to provision storage. You can leave this field empty if you have a default storage class configured on your Openshift cluster. Click ***Continue***. ***(we will use the default storage class)***

ocs-storagecluster-ceph-rbd

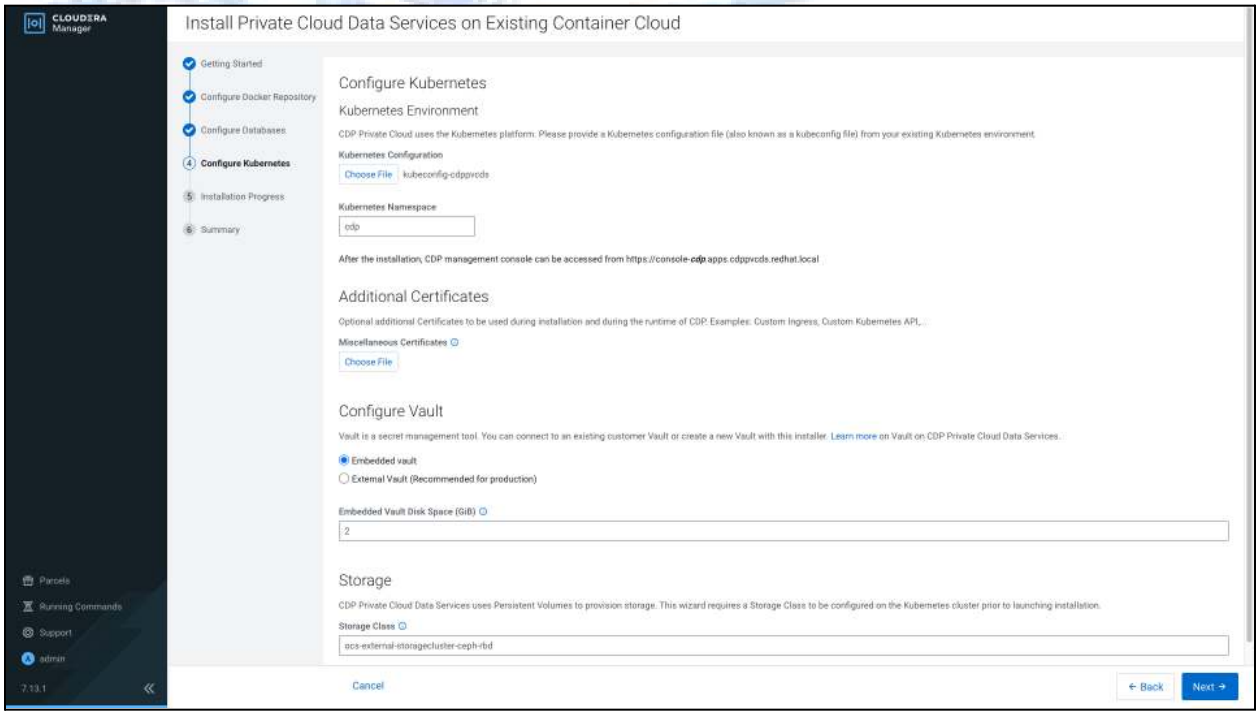
- Under the ***Additional Certificates*** section, click Choose File and add the SSL certificate for your HMS database (MariaDB, MySQL, ***PostgreSQL***, or Oracle). For Cloudera Data Warehouse, it is mandatory to secure the network connection between the default Database Catalog Hive MetaStore (HMS) in CDW and the relational database hosting the base cluster's HMS.

17) If you want to use this installation configuration again to install CDP Private Cloud, you have the option to ***download*** this information ***as a template***.

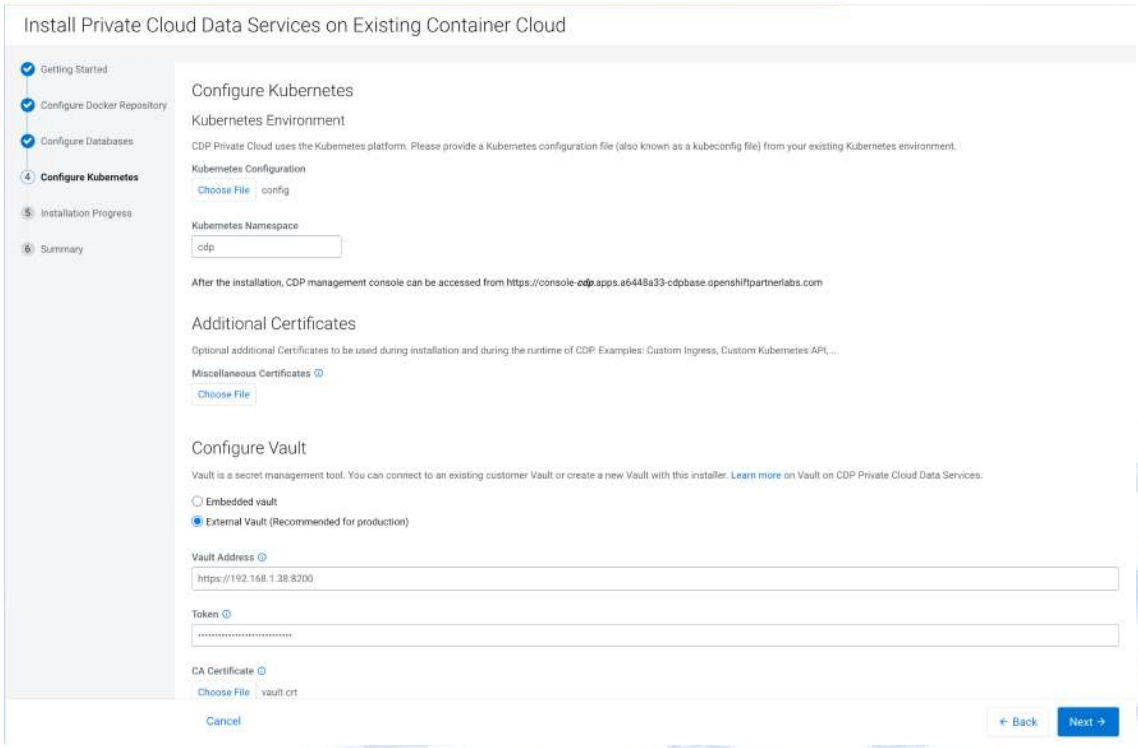
Note: You can also apply a template that you may have downloaded during a previous installation. The template contains all the installation configurations. Click ***Apply Previously Download Template*** to browse and upload a template stored on your machine.



The template file is a text file that contains the database and vault information that you entered for this installation. This template is useful if you will be installing Private Cloud again with the same databases, as the template will populate the fields here automatically. Note that the user password information is not saved in the template.



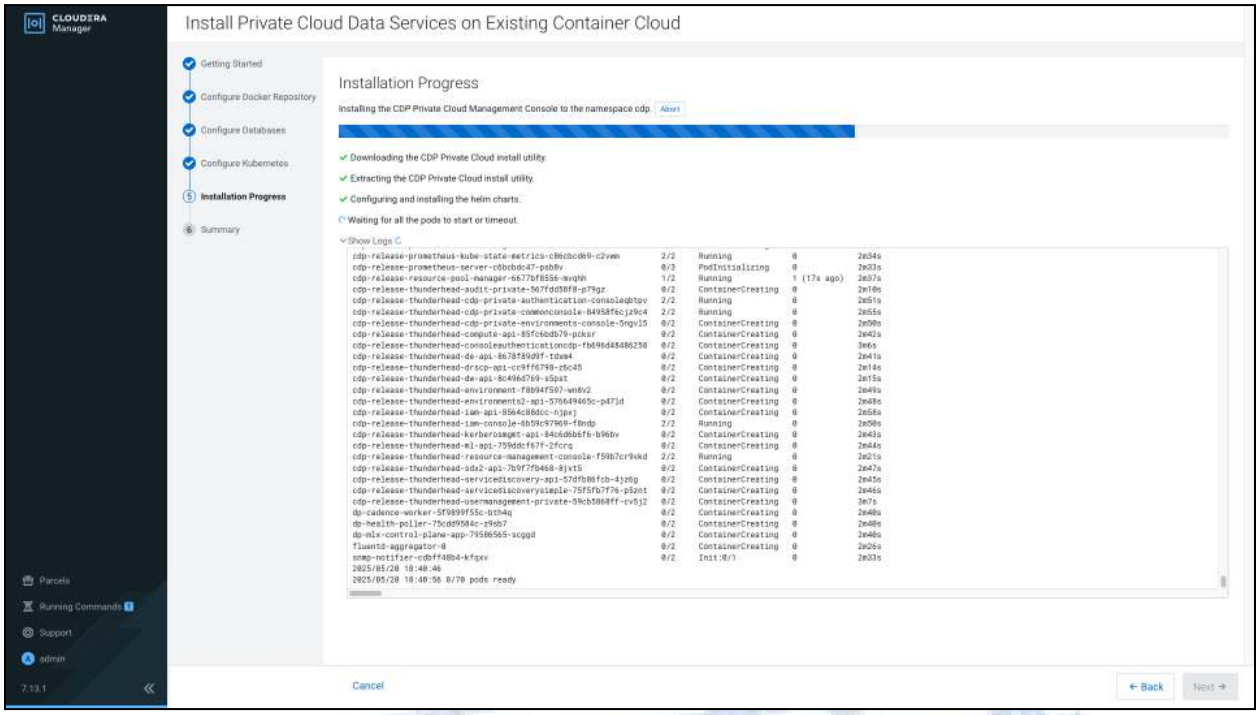
a. Screenshot for Using the Embedded Vault Option



b. Screenshot for Using the Embedded Vault Option

18) Click ***Next***.

19) The ***Installation Progress*** page appears. When the installation is complete, click ***Next***.



Install Private Cloud Data Services on Existing Container Cloud

Getting Started

Configure Docker Repository

Configure Databases

Configure Kubernetes

5 **Installation Progress**

6 Summary

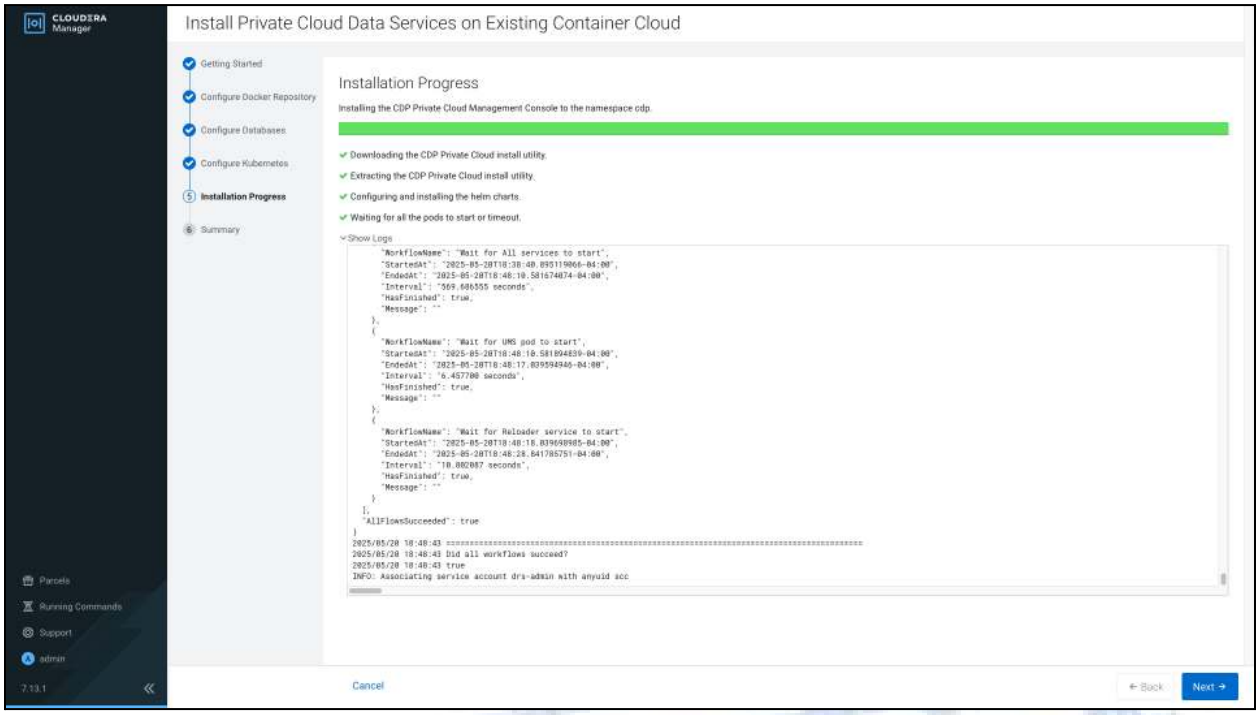
Installation Progress

Installing the CDP Private Cloud Management Console to the namespace cdp.

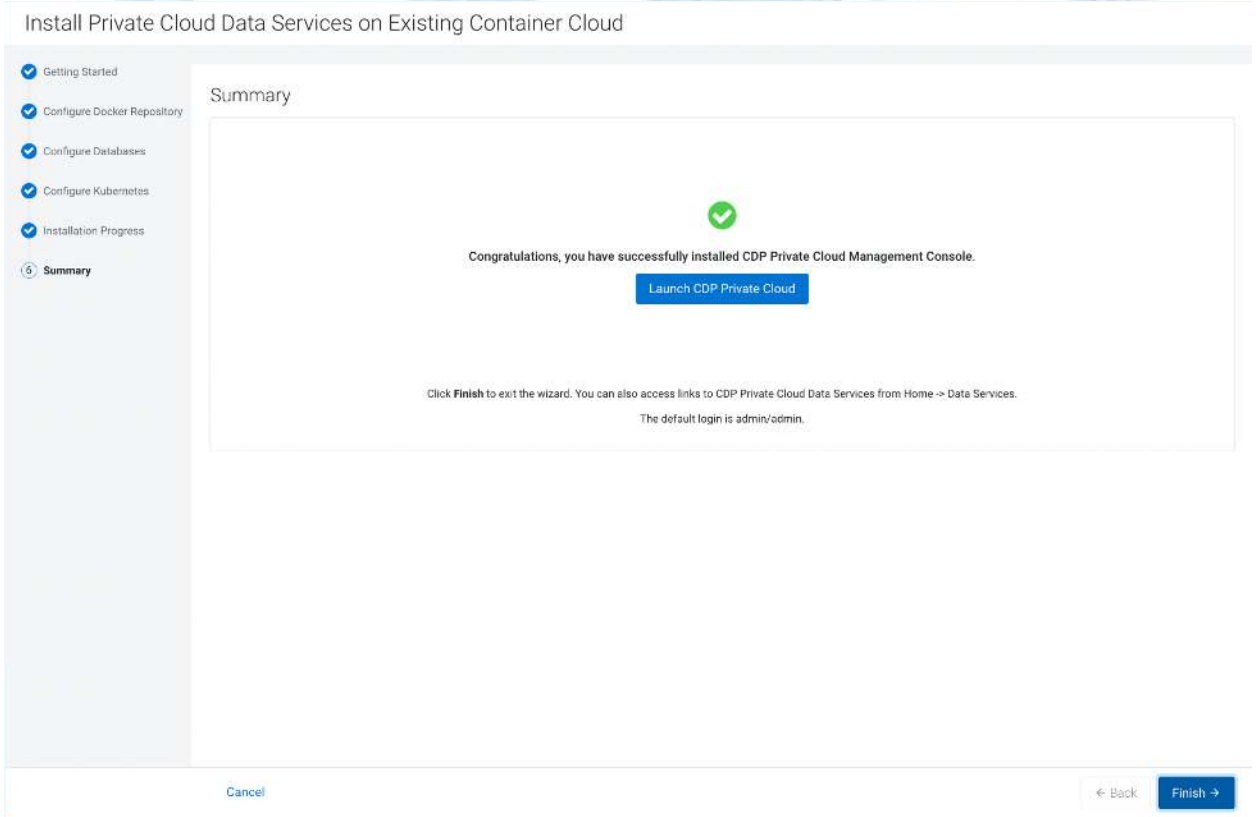
- ✓ Downloading the CDP Private Cloud install utility.
- ✓ Extracting the CDP Private Cloud install utility.
- ✓ Configuring and installing the helm charts.
- ✓ Waiting for all the pods to start or timeout.

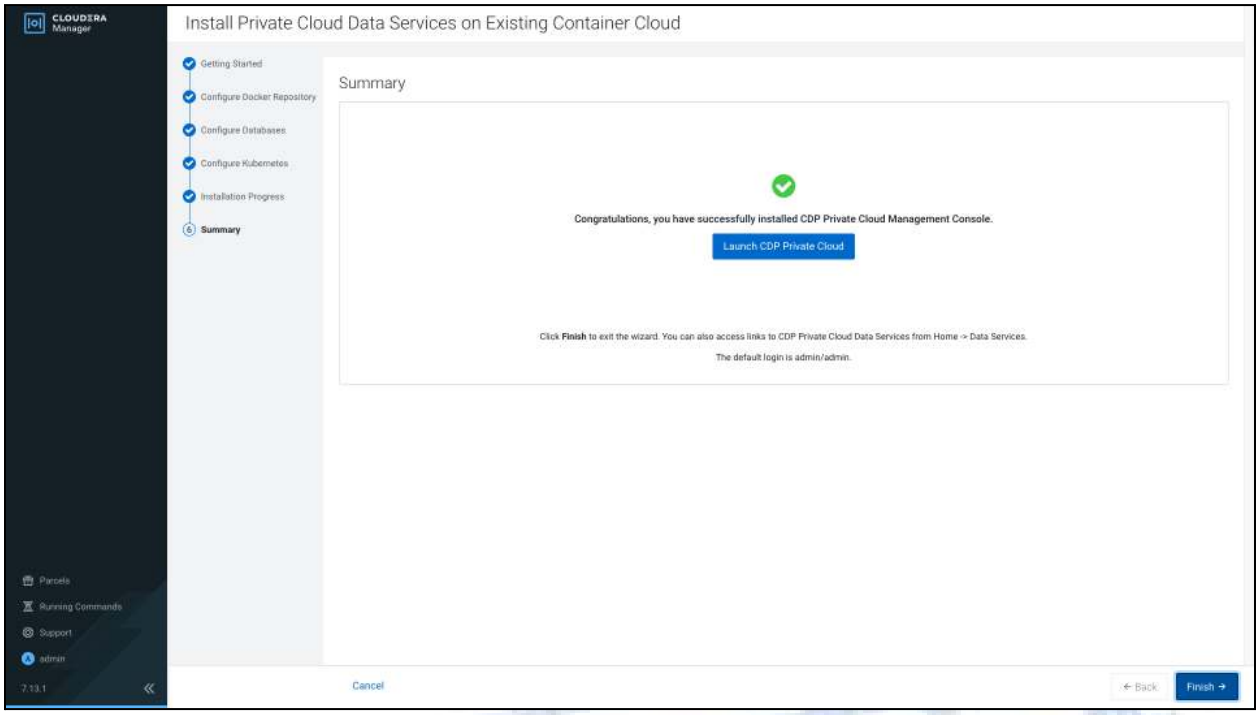
Show Logs

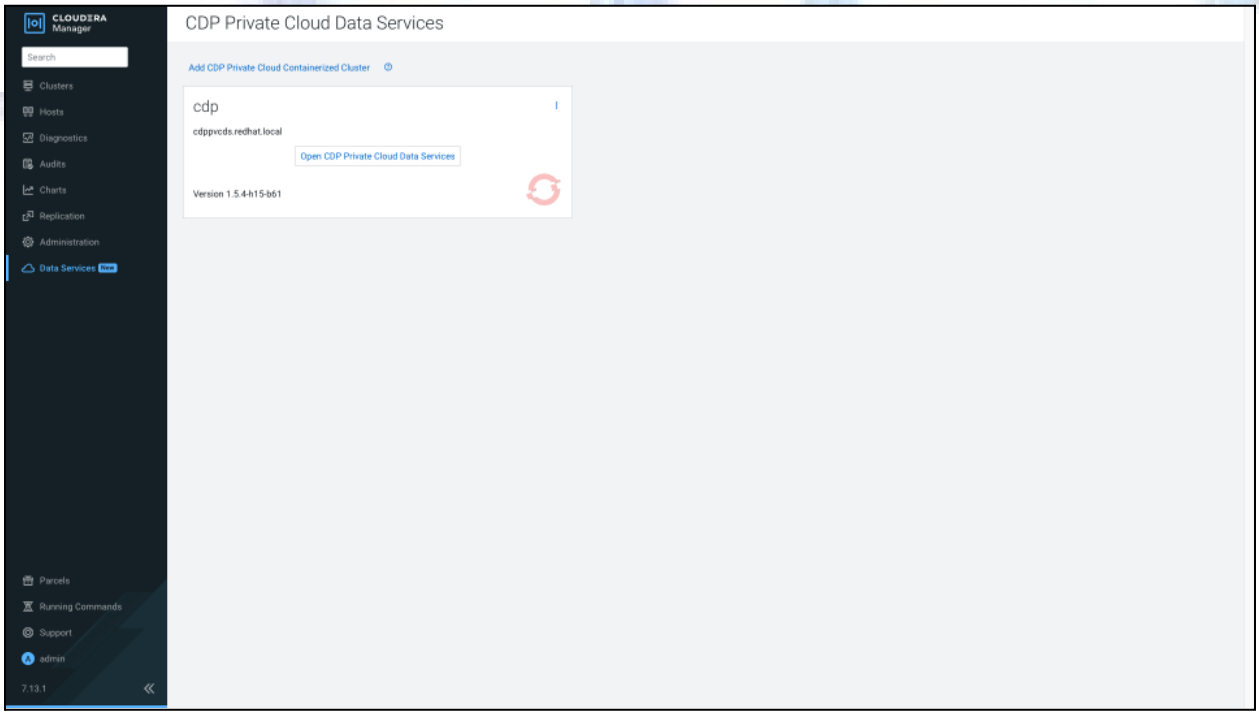
```
cdp-release-dps-gateway-1.0-cf7db56b-sm4s8                3/3     Running   0          4m31s
cdp-release-dwx-server-844cfb7899-g9jjn                   2/2     Running   0          3m58s
cdp-release-dwx-ui-698f4f85c6-4bpk5                       2/2     Running   0          3m58s
cdp-release-dwx-ui-698f4f85c6-bgrmf                       2/2     Running   0          3m56s
cdp-release-grafana-7c65c4566d-wx5tn                      3/3     Running   0          2m5s
cdp-release-logger-alert-receiver-86d67cdfb-4r2mh         2/2     Running   0          3m39s
cdp-release-metrics-server-exporter-6fb489845b-ch5cf      2/2     Running   0          3m38s
cdp-release-monitoring-app-67c7bf8fb4-cm82s               2/2     Running   0          3m26s
cdp-release-monitoring-metricproxy-7948d869df-c672g       2/2     Running   0          3m28s
cdp-release-monitoring-metricproxy-7948d869df-pjgsx       2/2     Running   0          3m27s
cdp-release-monitoring-pvcservice-75d986856d-skswq        2/2     Running   0          3m21s
cdp-release-prometheus-alertmanager-0                     3/3     Running   0          3m35s
cdp-release-prometheus-alertmanager-1                     3/3     Running   0          2m16s
cdp-release-prometheus-kube-state-metrics-658fbfc4f8-tb94h 2/2    Running   0          3m32s
cdp-release-prometheus-server-7dd745d8f7-zpxq6            3/3     Running   0          3m29s
cdp-release-resource-pool-manager-6967756fb4-kzcjs        2/2     Running   0          3m51s
cdp-release-thunderhead-cdp-private-authentication-consolewcm2w 2/2 Running 0          4m26s
cdp-release-thunderhead-cdp-private-commonconsole-65584957n8w9q 2/2 Running 0          4m29s
cdp-release-thunderhead-cdp-private-environments-console-6kfczm 2/2 Running 0          4m21s
cdp-release-thunderhead-compute-api-d5556b87d-82ql4       2/2     Running   0          4m6s
cdp-release-thunderhead-consoleauthenticationcdp-6d74fd8b4hhjfj 2/2 Running 0          4m40s
cdp-release-thunderhead-de-api-57d466787f-59tc7           2/2     Running   0          4m3s
cdp-release-thunderhead-environment-688965d7c8-gch8d      2/2     Running   1          4m20s
cdp-release-thunderhead-environments2-api-6b9fbc676-42jz7 2/2     Running   0          4m18s
cdp-release-thunderhead-iam-api-5475d7779c-qqgwr          2/2     Running   0          4m35s
cdp-release-thunderhead-iam-console-7b95d69df7-tpws4      2/2     Running   0          4m23s
cdp-release-thunderhead-kerberosmgmt-api-5544d69bbd-z7nnz 2/2     Running   0          4m8s
cdp-release-thunderhead-ml-api-8c684979f-bd8sc            2/2     Running   0          4m10s
cdp-release-thunderhead-resource-management-console-6f78c5z8brs 2/2 Running 0          3m13s
cdp-release-thunderhead-sdx2-api-54cdc9ccfb-qnldn         2/2     Running   0          4m17s
cdp-release-thunderhead-servicediscoverysimple-66d98ff555-khfkg 2/2 Running 0          4m14s
cdp-release-thunderhead-usermanagement-private-788d988b96-msh7f 2/2 Running 0          4m42s
dp-mlx-control-plane-app-7569dbd5bb-9z9vk                 2/2     Running   0          4m
dp-mlx-control-plane-app-health-poller-6c776fd948-rrn8w   2/2     Running   0          4m2s
fluentd-aggregator-0                                      2/2     Running   0          3m15s
snmp-notifier-855d984d7-k2kq6                             2/2     Running   0          65s
2022/04/28 16:15:51 To launch CDP Private Cloud, open https://console-cdp.apps.shared-os-qe-04.kcloud.cloudera.com/environments/welcome.html
2022/04/28 16:15:51 CDP Private Cloud Installation to cdp completed.
```



20) When the installation is completed, the ***Summary*** page with a link to ***Launch CDP Private Cloud*** appears. Click ***Launch CDP Private Cloud***. You can also click ***Finish*** and then access the ***Private Cloud Data Services*** instance from ***Cloudera Manager*** by clicking the ***Data Services in the*** left pane, then click ***Open Private Cloud Data Services*** for the applicable ***Data Services*** cluster.







****

NOTE: Run # kubectl get pods -A to review all pods and their status as either running or completed.

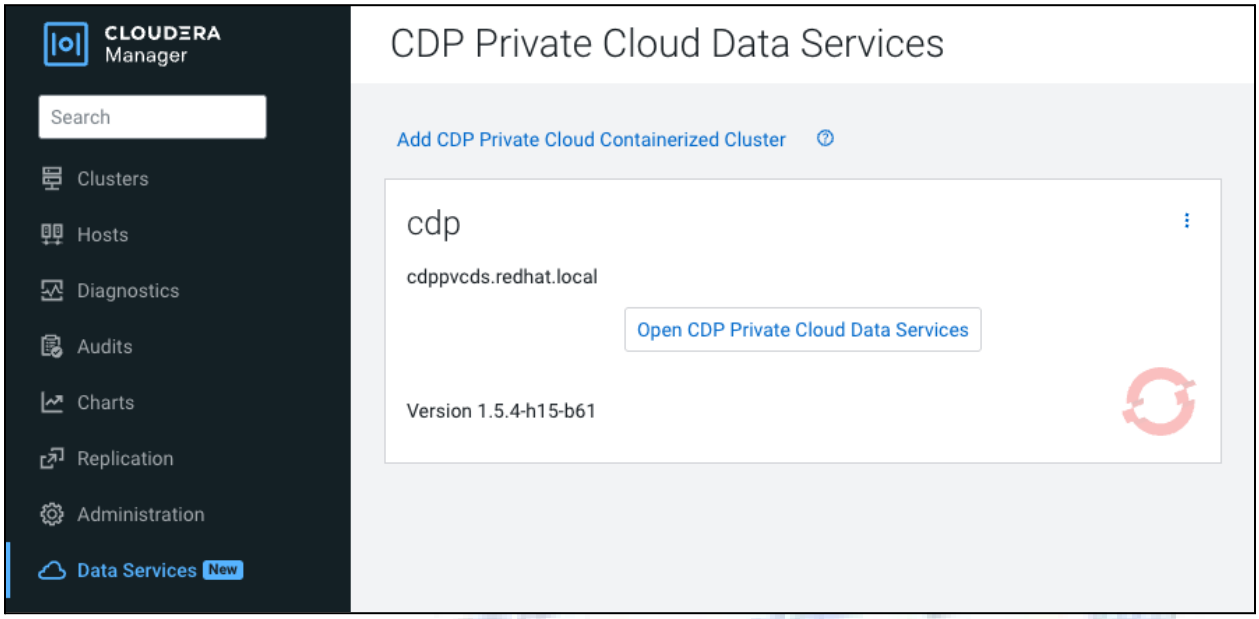
Installing Data Services can take several hours. The copying operation for Docker repository may take 4 - 5 hours

****
****

****

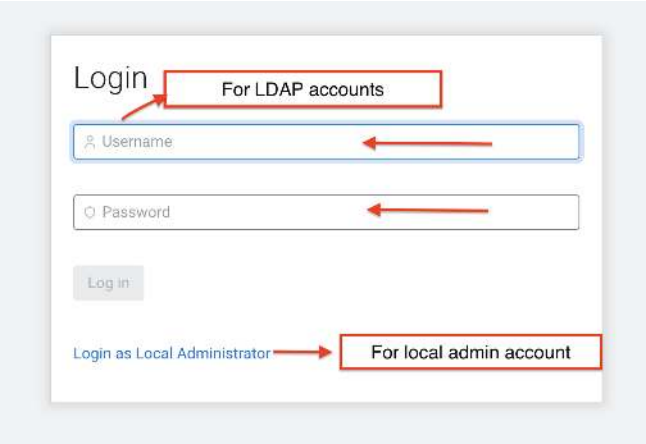
Accessing Cloudera on premises

Step 1. From the ***Cloudera Manager*** screen, click on ***Data Services(New)*** in the left pane.

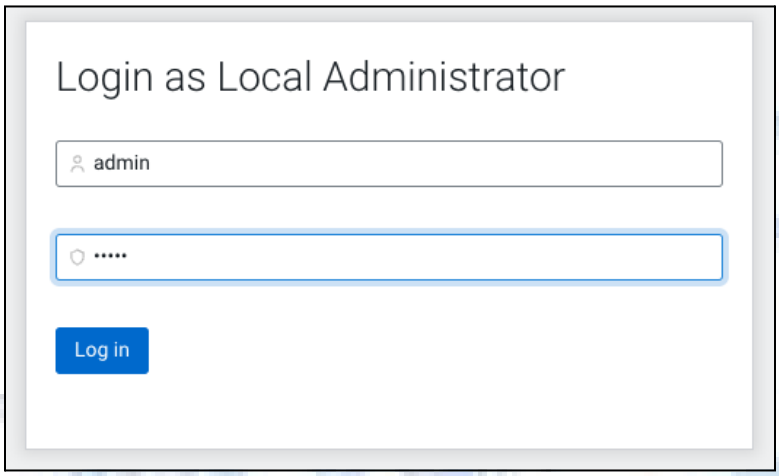


Step 2. On the ***CDP Private Cloud Containerized services*** page, click on the ***Open CDP Private Cloud Data Services*** button. This will open the ***Cloudera on premises authentication*** page.

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)***

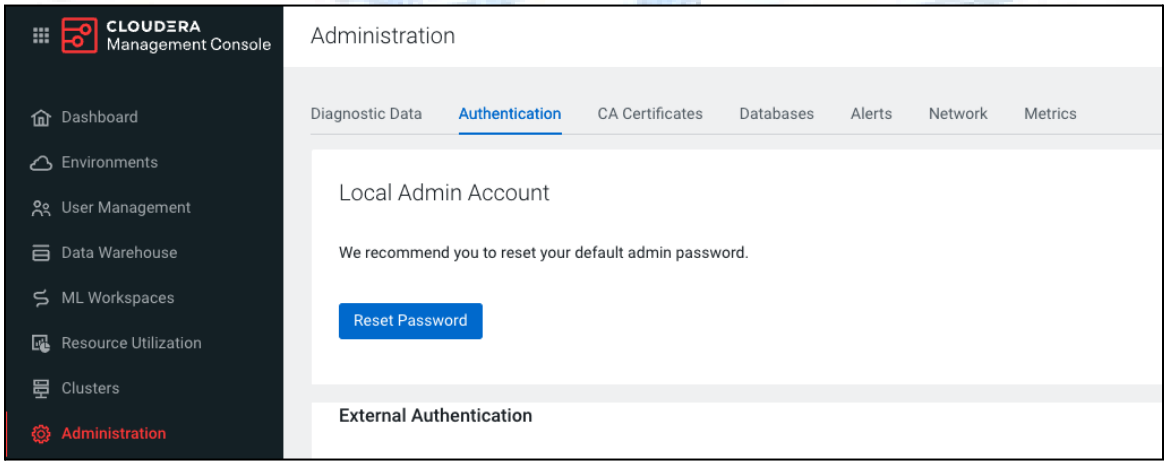


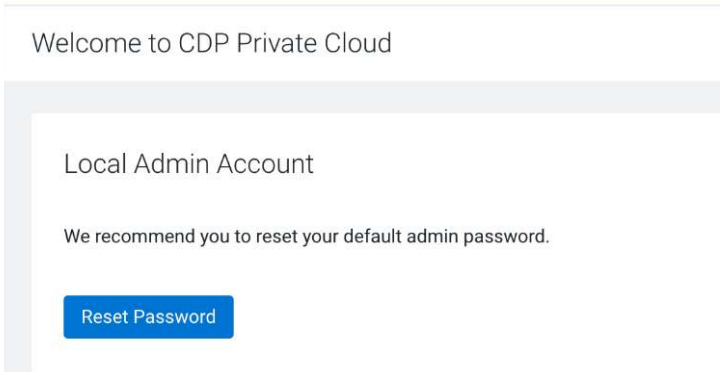
Step 4. *Login* to *Cloudera on premises Data Services* as *local administrator*. *(admin/admin)*



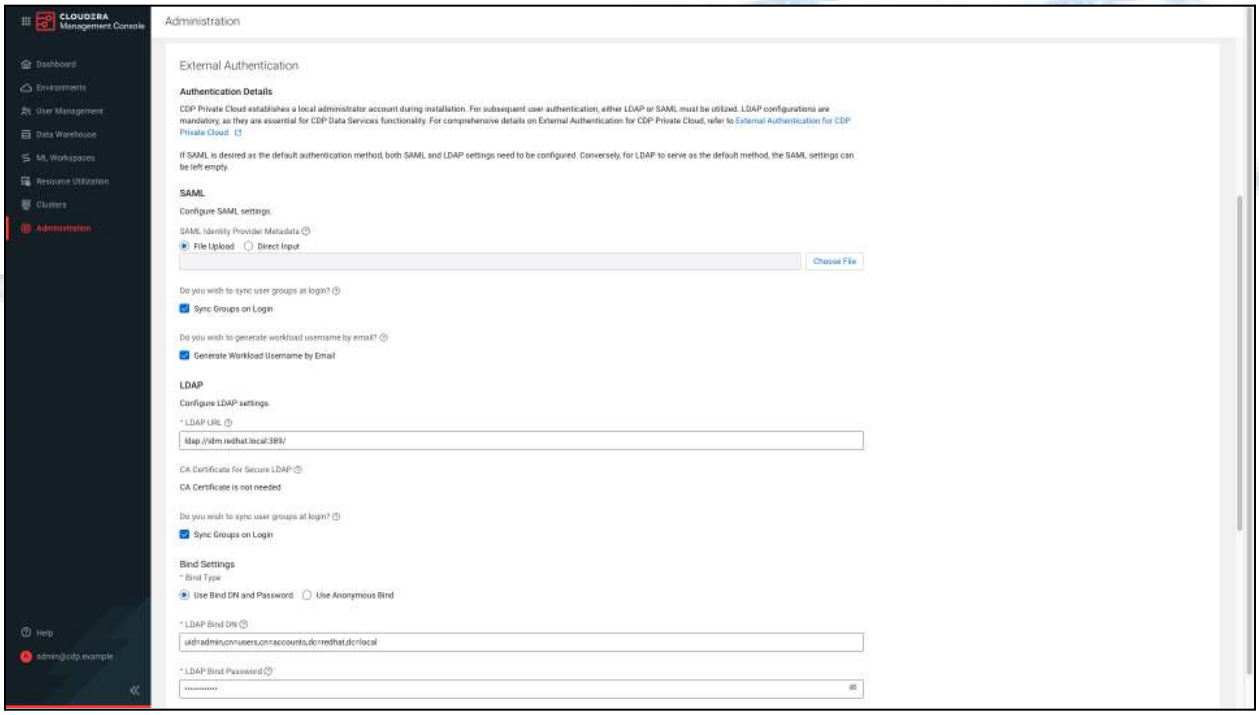
Step 5. After authenticating successfully, you will land at the *CDP console/Data Services* page. From this page, you can navigate to different data services and the management services. Click on the *Management Console*.

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.



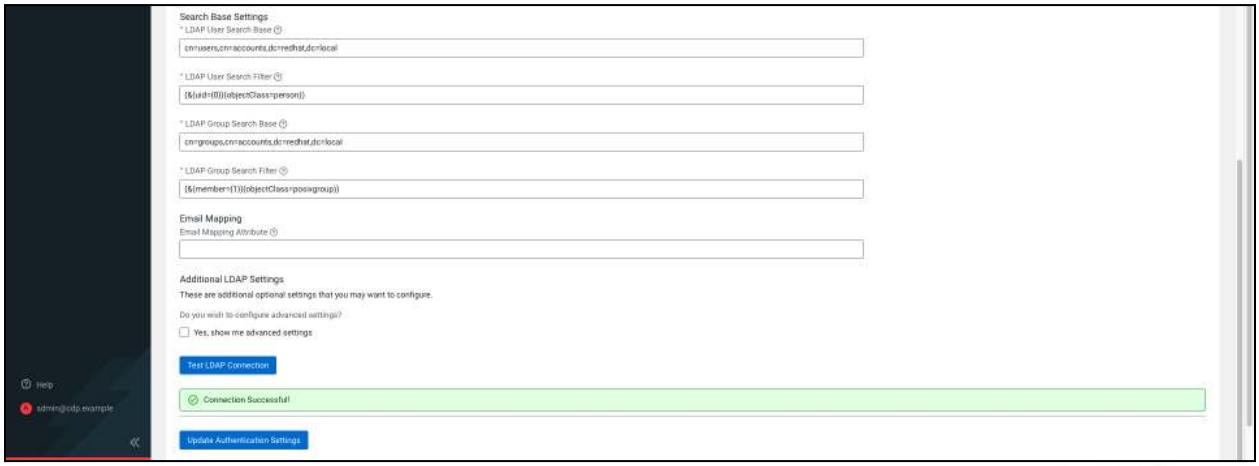
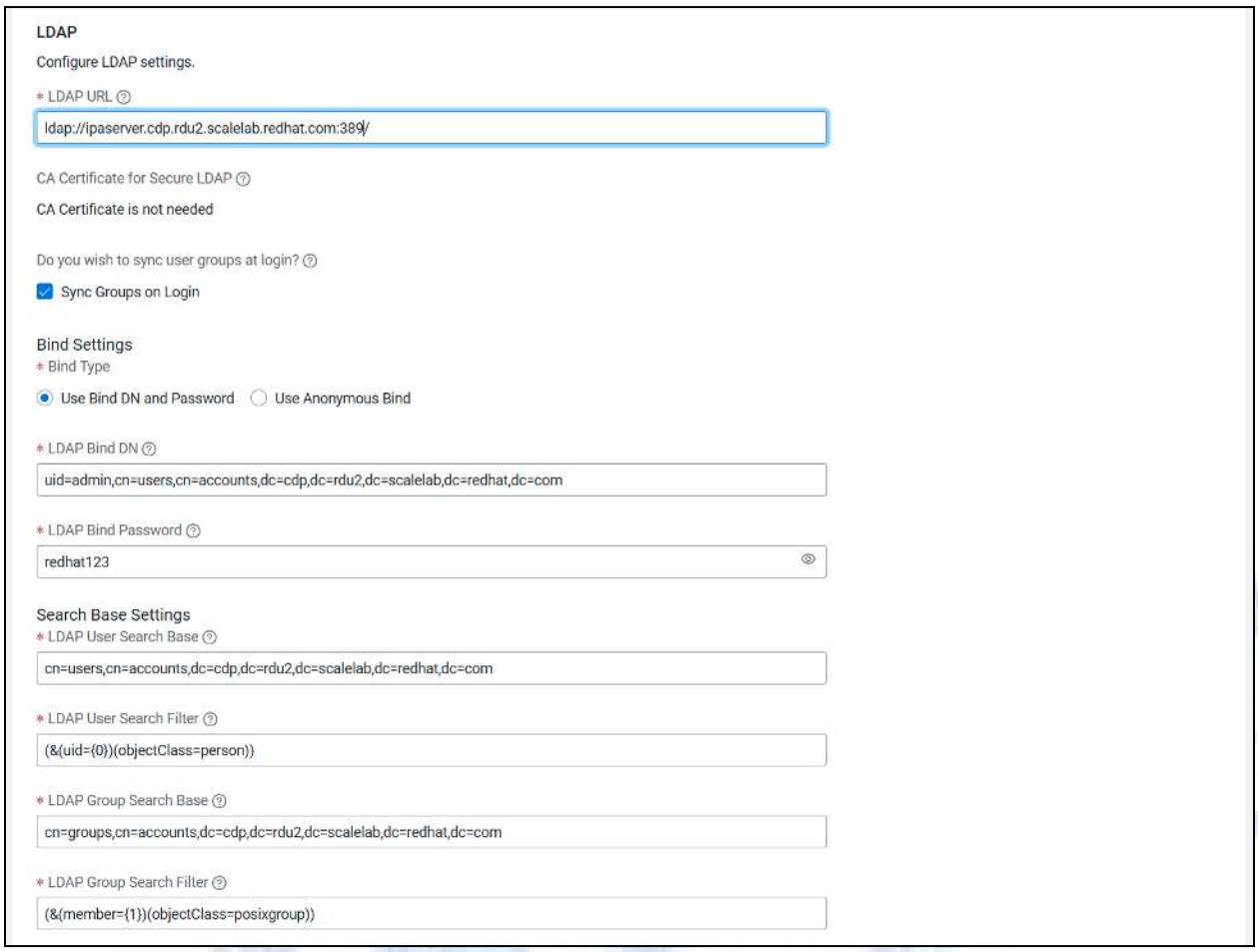


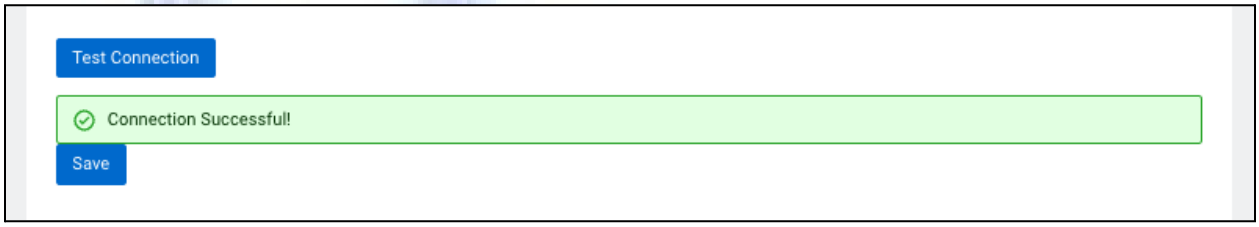
Table 7. LDAP Integration

Component	Value
LDAP URL:	`ldap://ipaserver.cdp.rdu2.scalelab.redhat.com:389/`
LDAP Bind User DN:	`uid=admin,cn=users,cn=accounts,dc=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com`
LDAP Bind Password:	***<redhat123>*** `(password for KDC admin, configured earlier)`
LDAP User Search Base:	`cn=users,cn=accounts,dc=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com`
LDAP User Search Filter:	`(&(uid={0})(objectClass=person))`
LDAP Group Search Base:	`cn=groups,cn=accounts,dc=cdp,dc=rdu2,dc=scalelab,dc=redhat,dc=com`
LDAP Group Search filter:	`(&(member={1})(objectClass=posixgroup))`

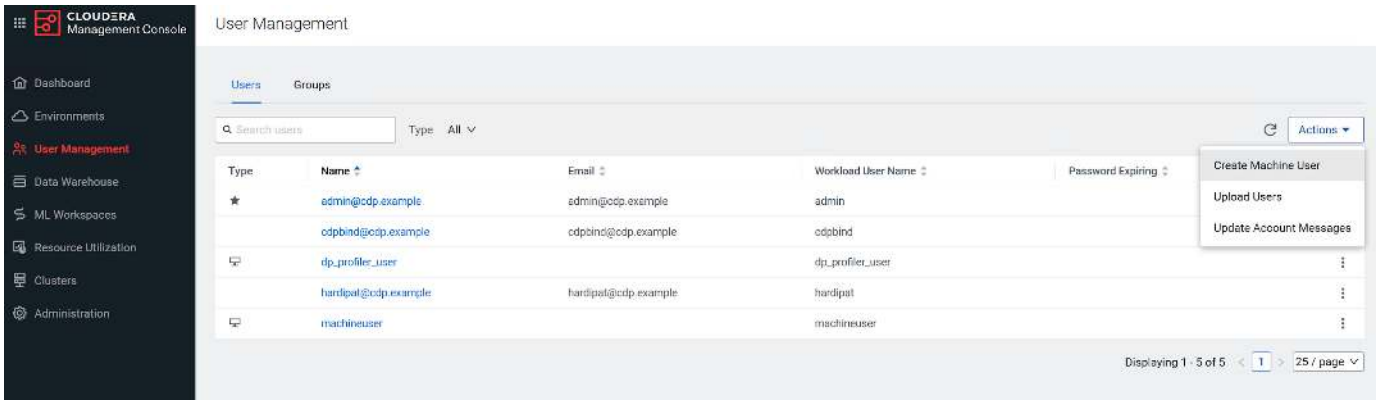


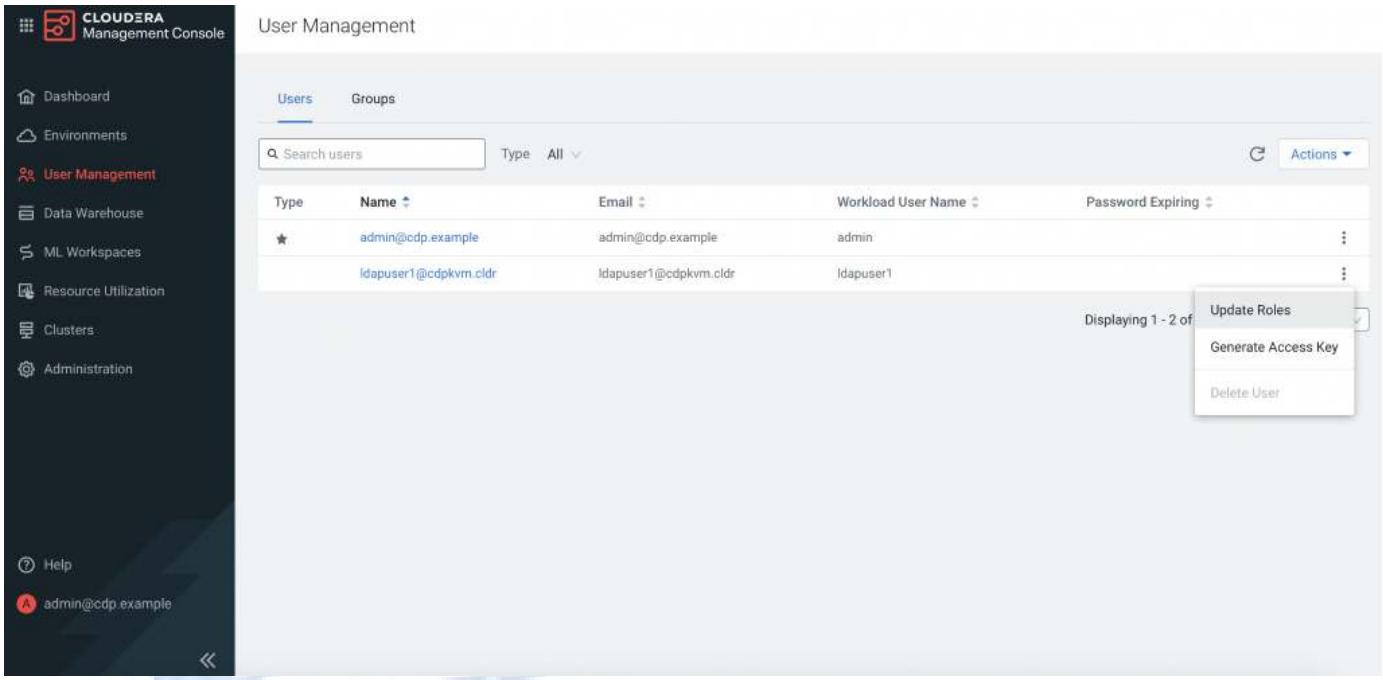
Step 8. Follow the instructions on the ***Welcome to CDP Private Cloud page*** to complete this step.

Step 9. Click ***Test Connection*** to ensure that you are able to connect to the ***configured LDAP server***.



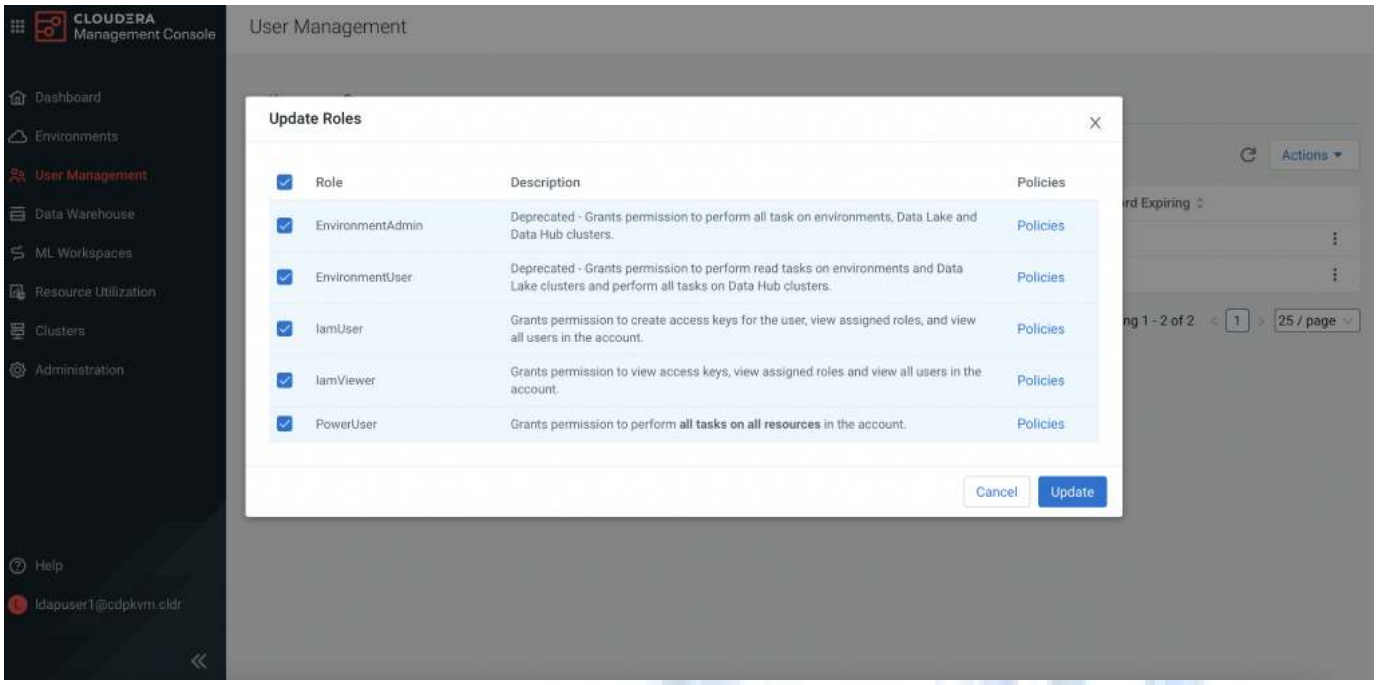
Step 10. The ***User Management*** tab allows users to add or update roles on existing users. ***Groups*** tab allows users to sync user groups from the active directory to access ***CDP Data Services***.





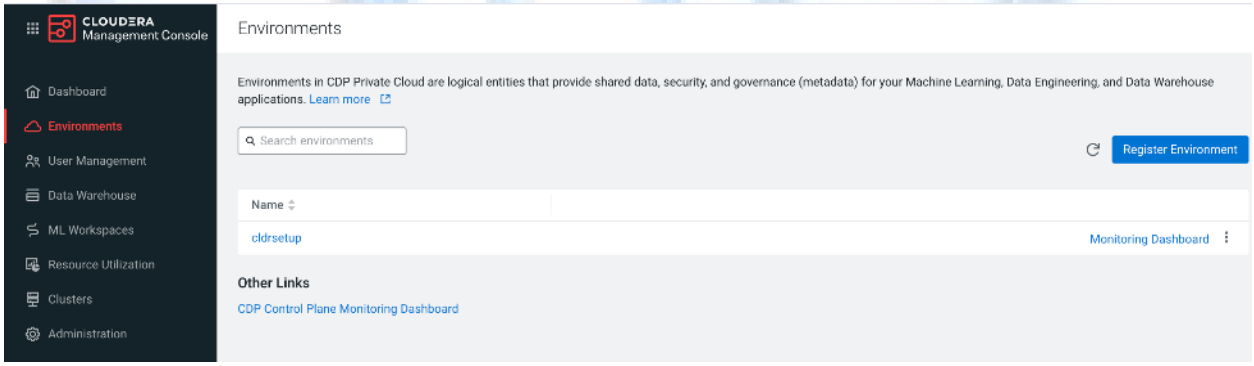
For more details on Cloudera on premises Management console please visit:

https://docs.cloudera.com/management-console/1.5.5/index.html

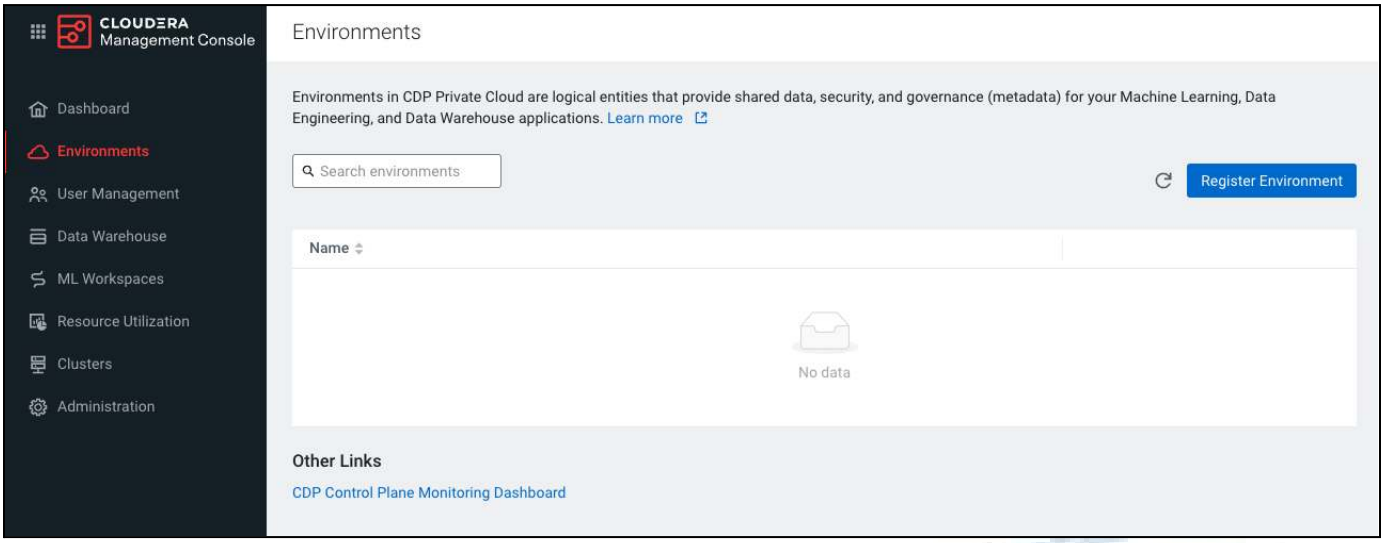


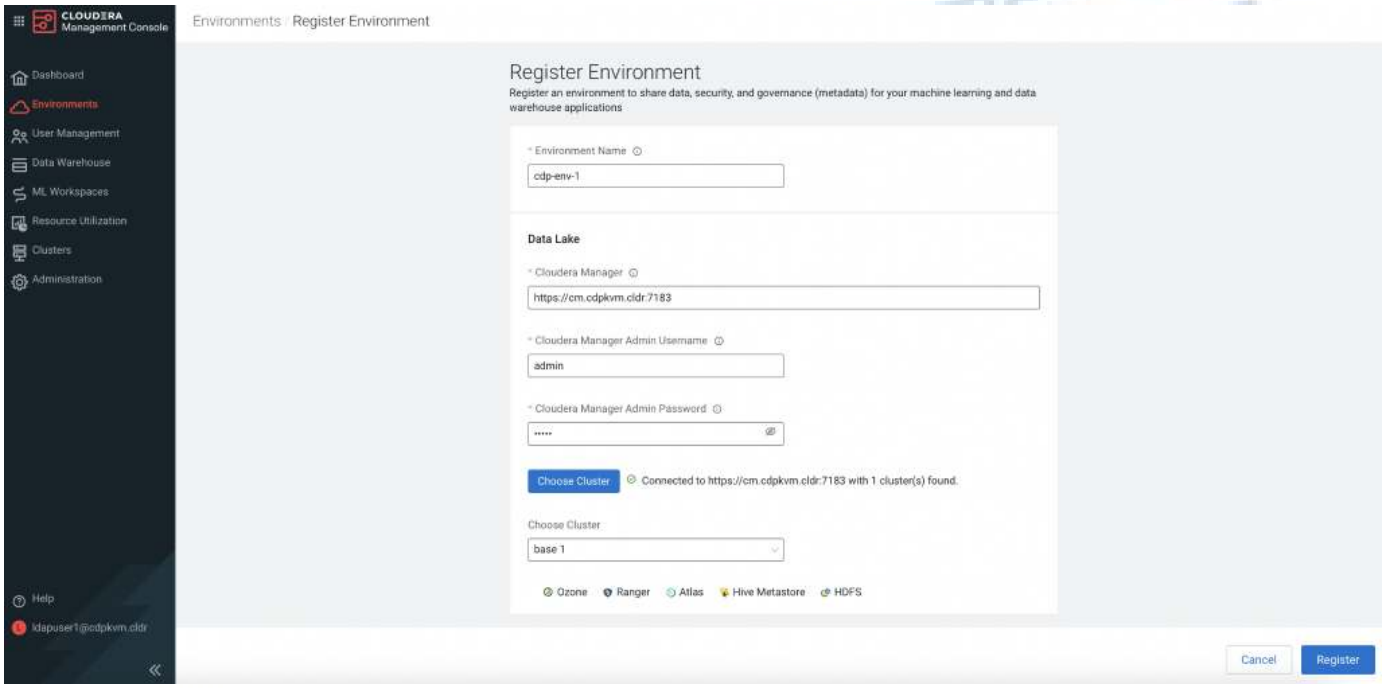
Step 11. After successfully configuring and testing the setup for ***LDAP integration***, the page will auto-redirect for the Environments Page, if not, navigate to the `Environments` page, by clicking into the menu in the left pane of your screen.

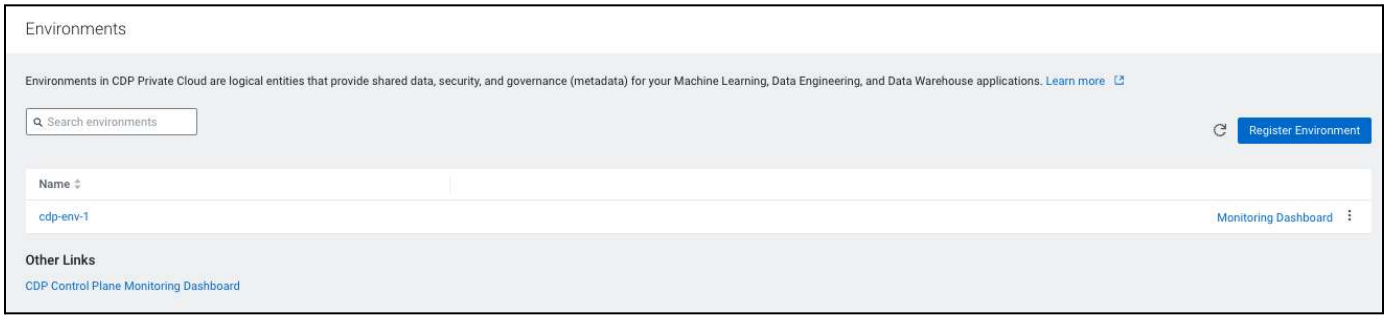
Step 12. If everything was correctly setup previously, then you will be able to see the environment registered by default for your cluster as shown below:

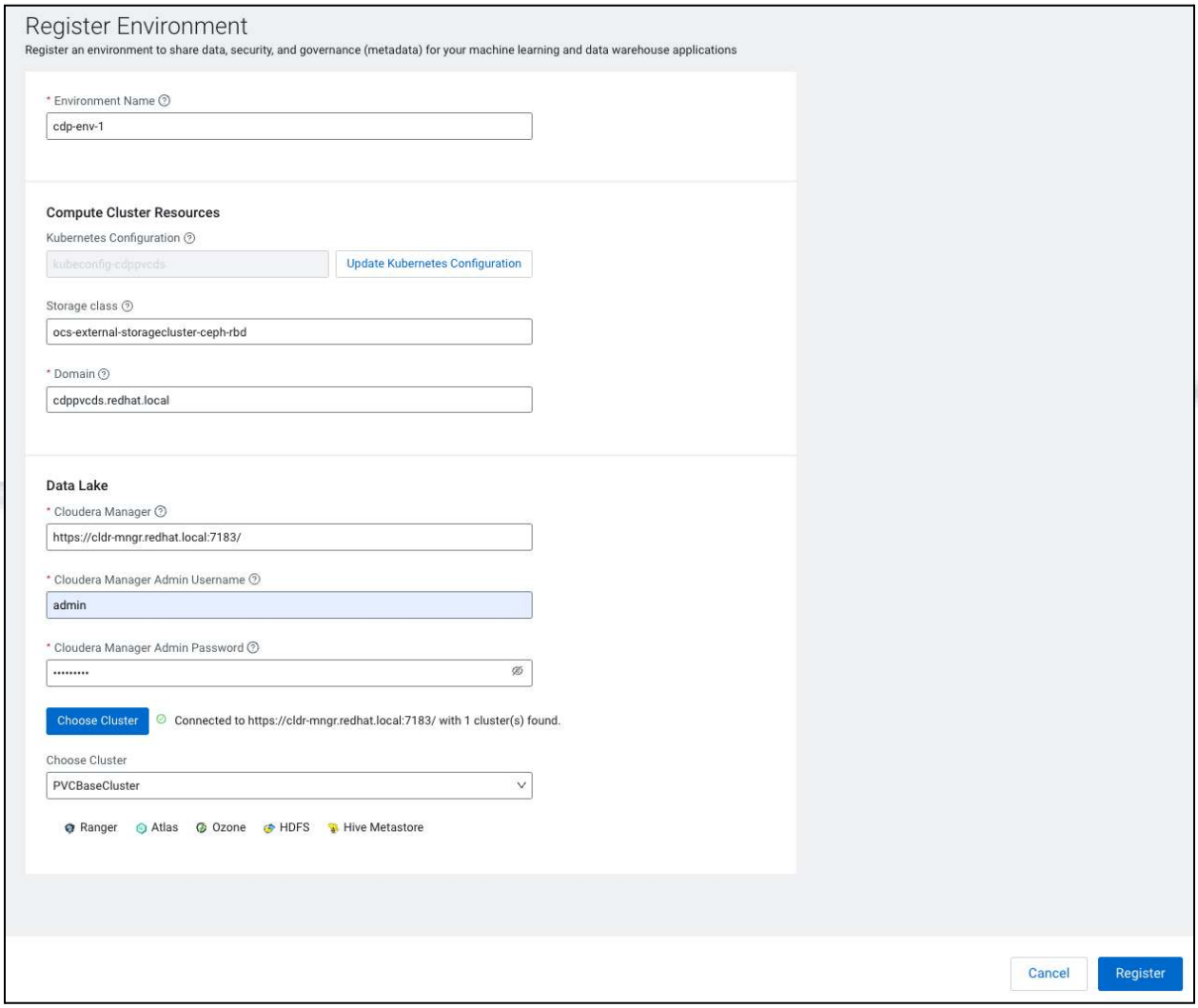


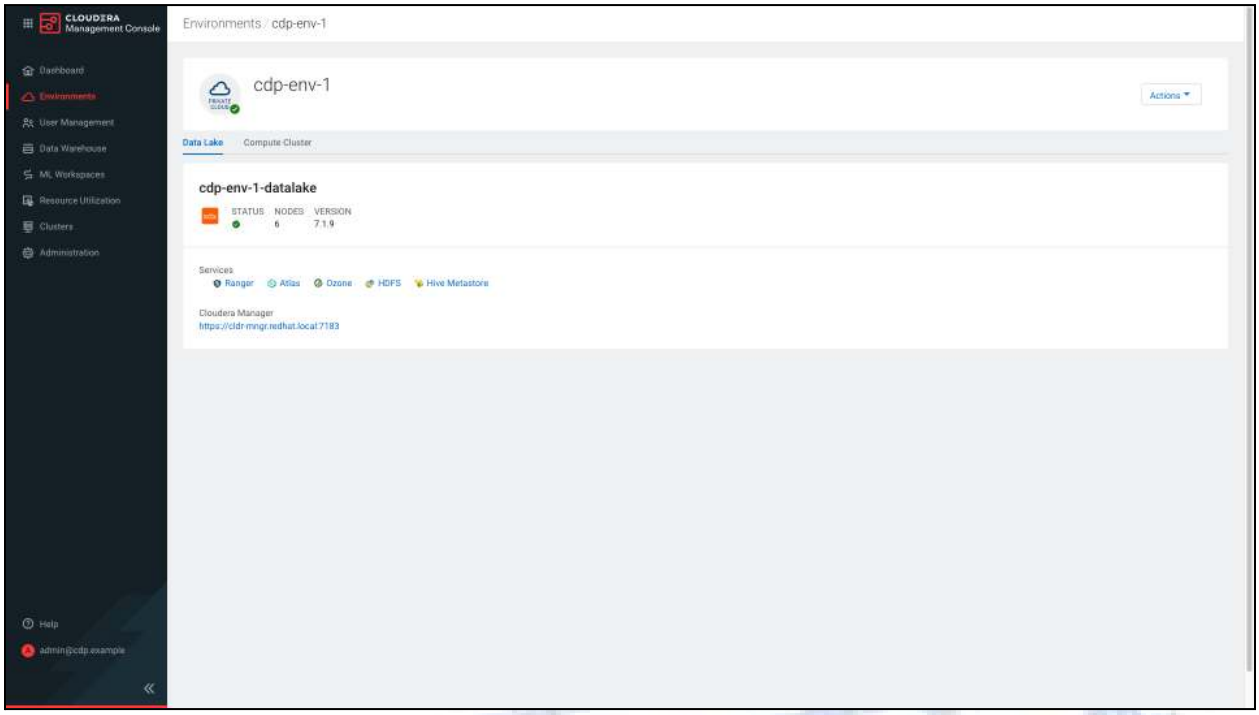
Step 13. If environment is not created somehow due to some issue during the installation, page will auto-redirect for ***Register Environment*** Page, if not, navigate to the `Environments` page, and select ***Register Environment*** where you will provide the Cloudera Manager details and credentials 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***.

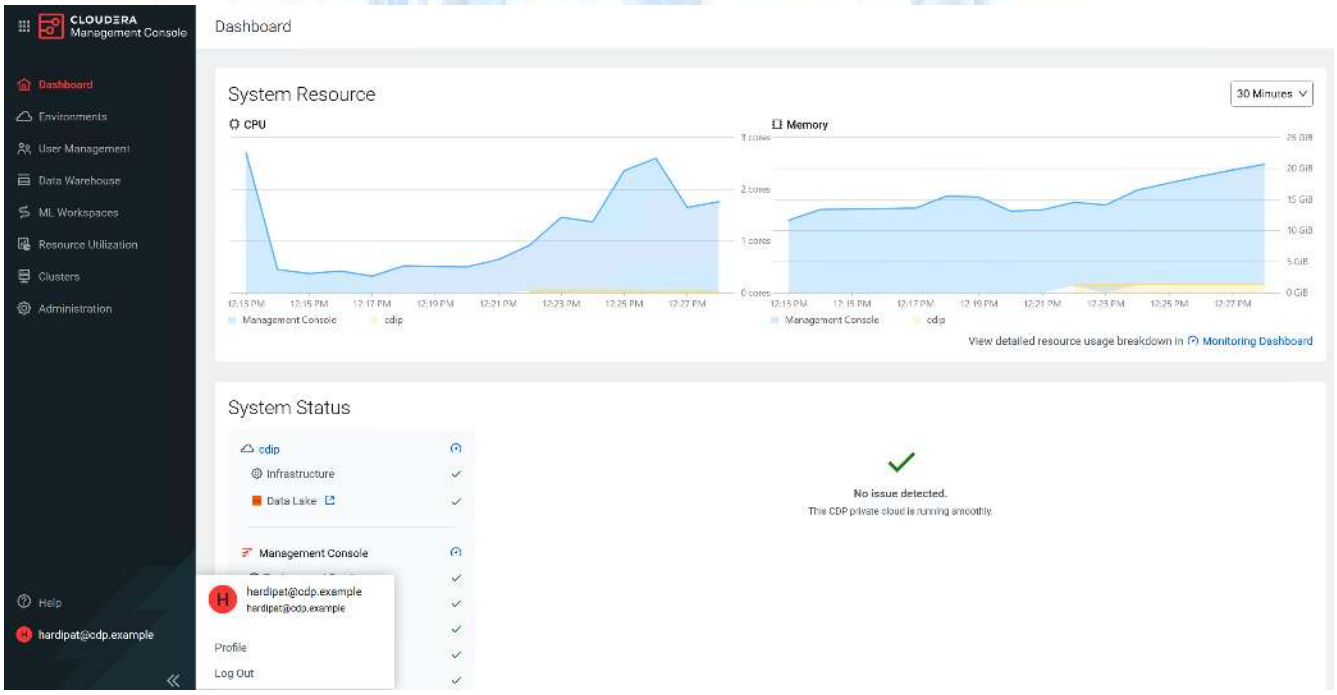












Step 14. To come to this section further, from the ***Cloudera Manager*** screen, click on ***Data Services(New)*** in the left pane and then click on ***Open Private Cloud Data Services*** to launch your ***CDP Private Cloud Data Services instance***. Log in using the default username and password ***admin***.

- Click ***Launch CDP*** to launch your CDP Private Cloud.
- 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 Private Cloud environment
- Create your first Virtual Warehouse in the CDW Data Services
- Provision an AI Workbench in the CML Data Services
- Add a CDE service in the CDE Data Service

Parent topic: Installation on the OpenShift Container Platform (OCP)

****

Configuring GPU Node Labeling Steps for OCP Cluster Setup:

https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-ocp-configuring-gpu-node-labeling-ocp.html

You can use ***NVIDIA Feature Discovery*** to generate labels for the set of ***GPUs*** available on ***OCP nodes***. You can use these ***node labels*** to assign workloads to specific ***GPU devices***. This feature is enabled by default on ***ECS***, but must be configured manually on ***OCP***.

Configuring GPU node labeling on OCP nodes

1. Review the prerequisites listed on the NVIDIA GPU feature discovery page.
2. Use the instructions under Deployment via helm to deploy the GPU node labeling feature.
3. Information about using GPU node labeling is also available on the NVIDIA GPU feature discovery page.

Note: ***GPU node labeling*** is only supported for ***GPU cards*** manufactured by ***NVIDIA***.

****

Dedicating OCP nodes for specific workloads

You use Cloudera Manager to dedicate Openshift Container Platform (OCP) cluster nodes for specific workloads. You can dedicate GPU nodes for CML workloads, and NVME nodes for CDW workloads.

https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-ocp-dedicating-nodes-for-workloads.html

You can use the kubectl taint command to dedicate OCP cluster nodes for specific workloads. You can dedicate GPU nodes for CML workloads, and NVME nodes for CDW workloads.

Run the following command to get a list of all of the cluster nodes:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl get nodes
```

Run the following command to list information about a specific cluster node:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl describe node <node_name>
```

In the returned output, look for the Taints field.

Dedicate a GPU node for CML workloads

1. Run the following command to dedicate a GPU node for CML workloads:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl taint nodes <node_name> nvidia.com/gpu=true:NoSchedule
```

2. No other workload pods will be allowed to run on the tainted node.

3. Run the following command to confirm that the taint has been successfully applied:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl describe node <node_name>
```

4. In the returned output, look for the Taints field.

5. To remove the taint, run the following command:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl taint nodes <node_name> nvidia.com/gpu=true:NoSchedule-
```

   This command returns:
   node/<node_name> untainted

Dedicate a SSD node for CDW workloads

1. Run the following command to dedicate a GPU node for CDW workloads:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl taint nodes <node_name> ssd/nvme=true:NoSchedule
```

2. No other workload pods will be allowed to run on the tainted node.

3. Run the following command to confirm that the taint has been successfully applied:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl describe node <node_name>
```

4. In the returned output, look for the Taints field.

5. To remove the taint, run the following command:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl taint nodes <node_name> nvidia.com/gpu=true:NoSchedule-
```

This command returns:
node/<node_name> untainted

Additional Notes

Note: To taint the node of an existing cluster which already has CML and CDW workspaces running, you must also run the following commands:

```
ksahu@Kuldeeps-MacBook-Air ~ % kubectl drain <node_name> --ignore-daemonsets --delete-emptydir-data
--timeout=600s
ksahu@Kuldeeps-MacBook-Air ~ % kubectl uncordon <node_name>
```

Note: You cannot apply a taint to a master node, or to a single-node cluster.

Related information

- Taints and Tolerations

****

Cloudera on premises Machine Learning (CAI)

Please review Cloudera on premises Machine Learning for more details.

Please review requirements page for OCP 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

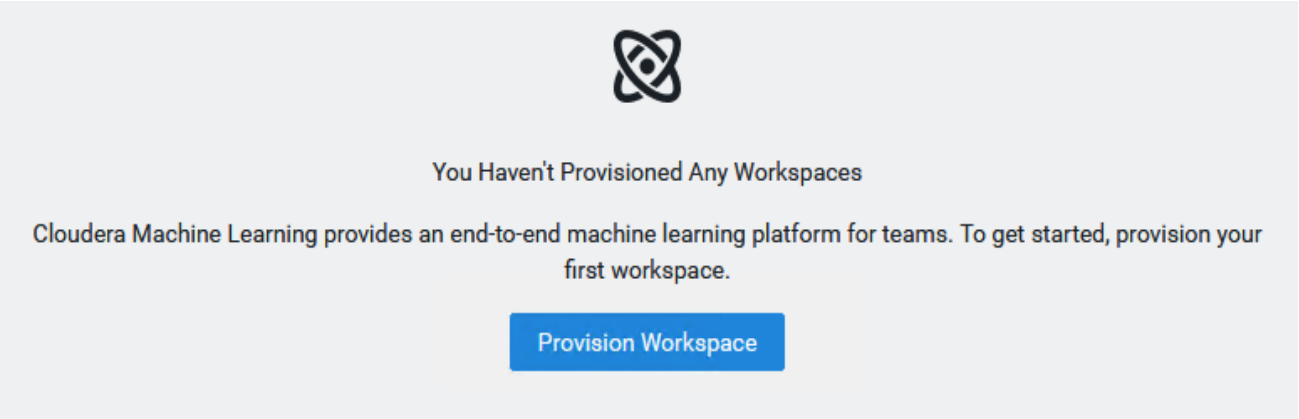
https://docs.cloudera.com/machine-learning/1.5.5/workspaces-privatecloud/topics/ml-pvc-provision-ml-workspace.html

AI Workbench 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.

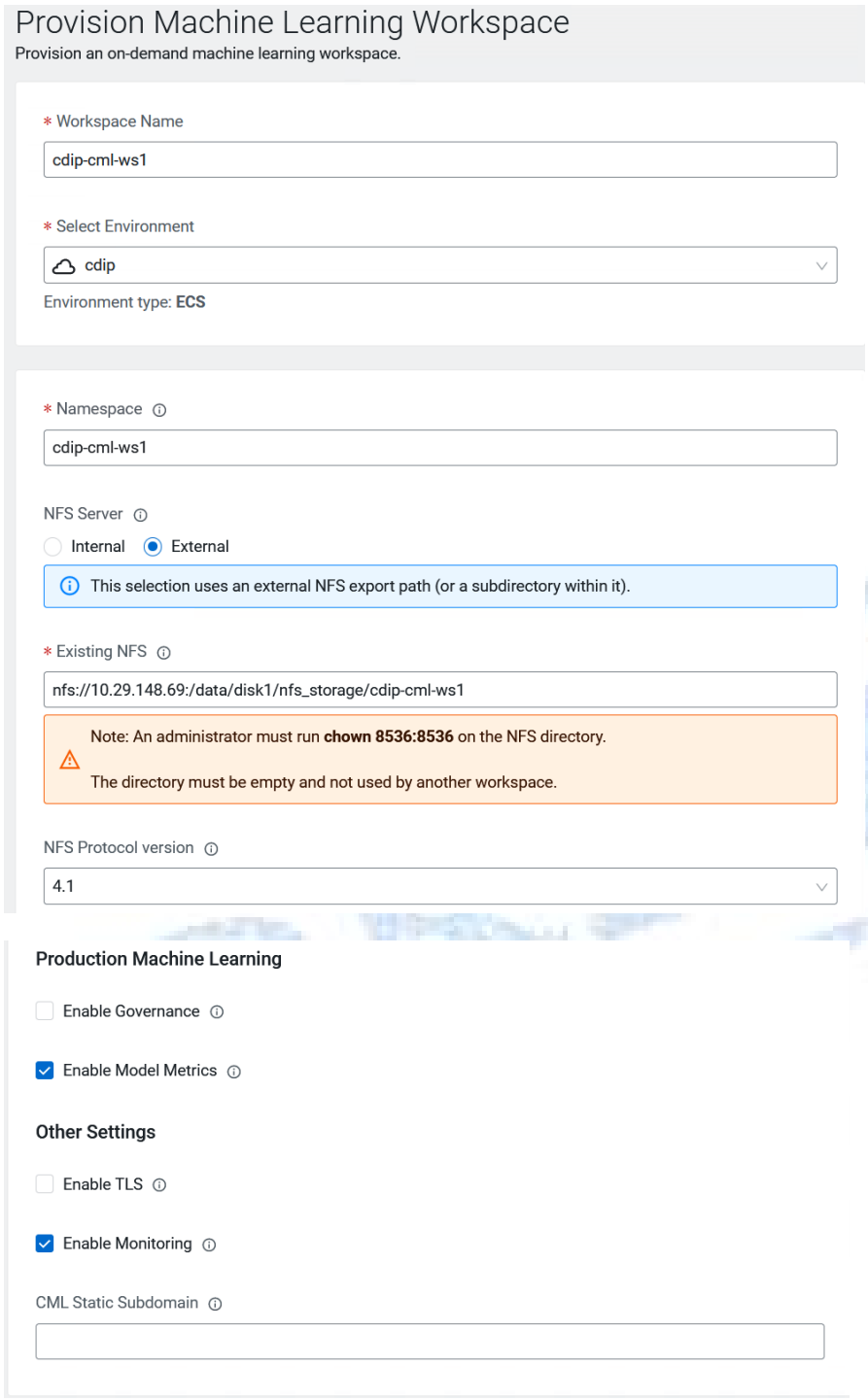


3. Provide input required to provision machine learning workspace. Click on provision workspace.

- 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 OCP 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@pvcocp-master ~]# mkdir -p /lhdata/nfs_storage/kuldeep-test-cml-w1
[root@pvcocp-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.

To enable TLS using the **default self-signed certificate used by OpenShift,** extract the certificates using the following command:

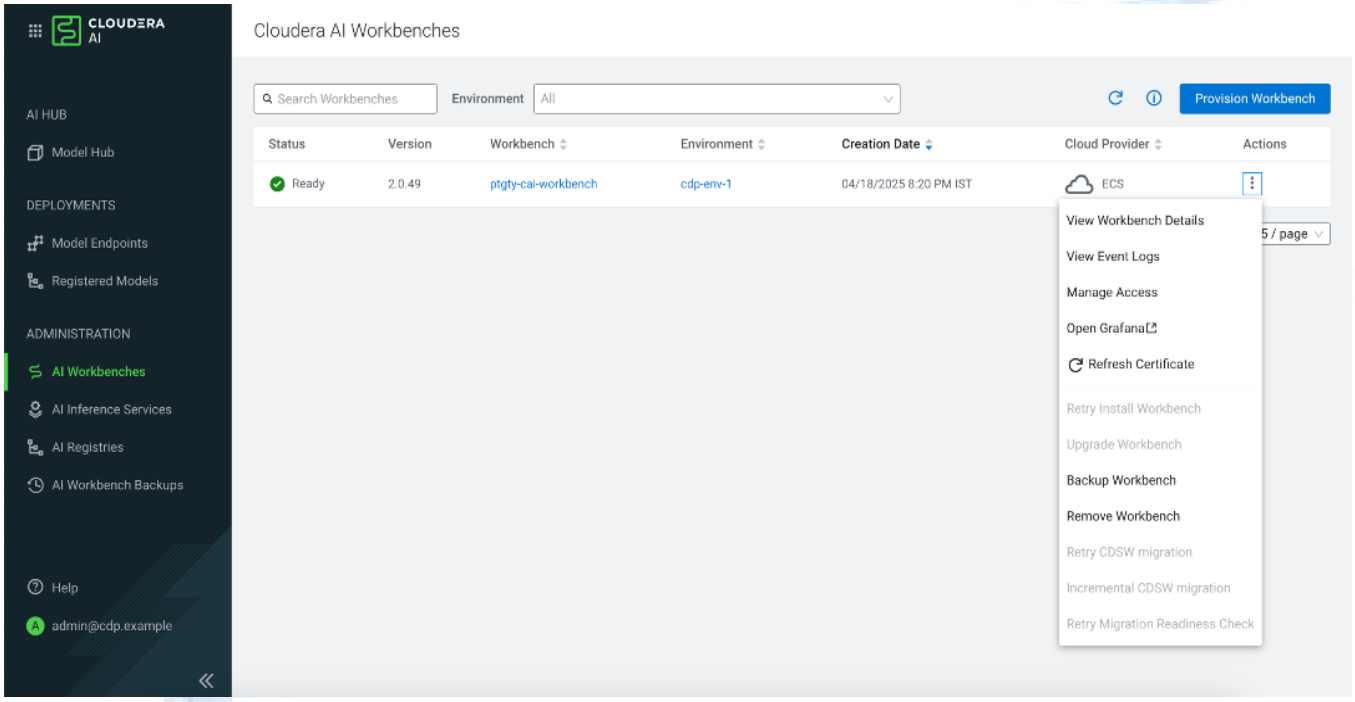
```
[root@ipasesrver ~]# oc extract secret/router-certs-default -n openshift-ingress --to=.

[root@ipasesrver ~]# ll tls.*
```

```
-rw-------. 1 root root 2388 Nov  6 21:26 tls.crt
-rw-------. 1 root root 1675 Nov  6 21:26 tls.key
```

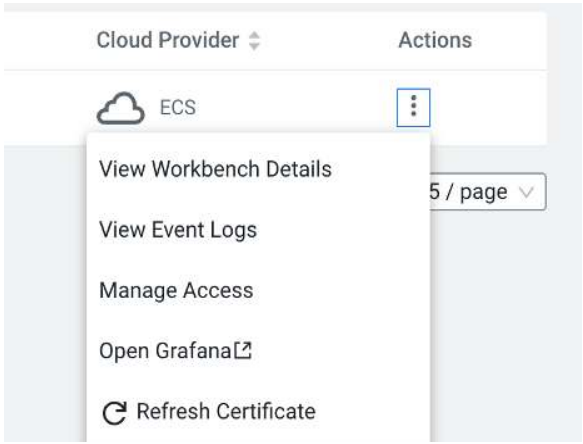
Then create a secret from the certificates named ***cml-tls-secret*** in the namespace of the workbench using the following command:

```
[root@ipasesrver ~]# oc create secret tls cml-tls-secret --cert=tls.crt --key=tls.key -o yaml --dry-run |
oc -n cdp-cml-ws1 create -f -
```

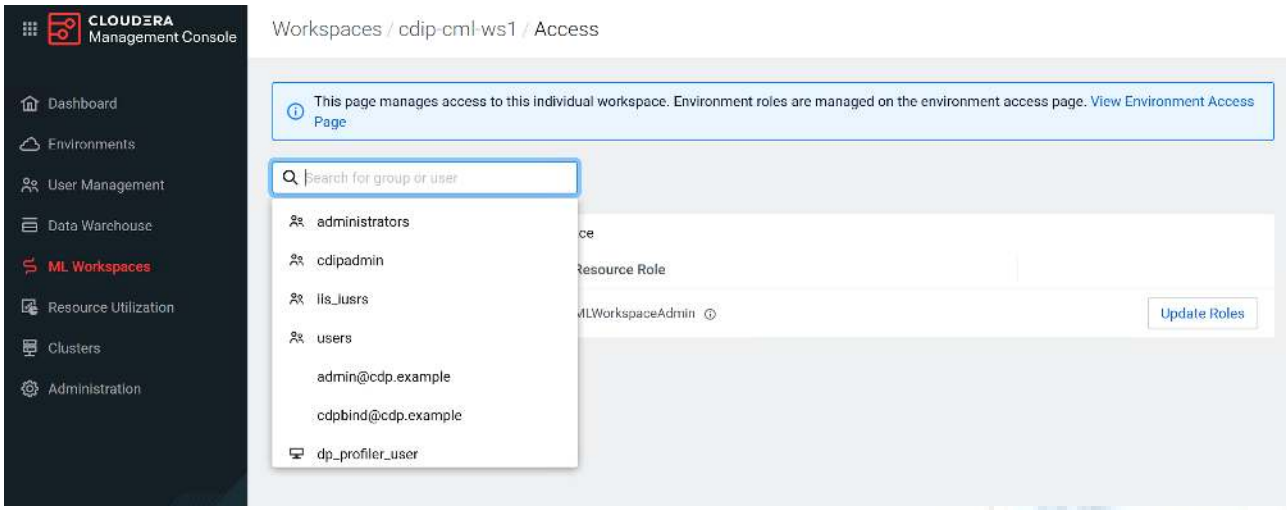
4. When provisioning of the workbench is completed the status reports as ***Ready***. Once it is created, it appears on the AI Workbenches page as shown below.



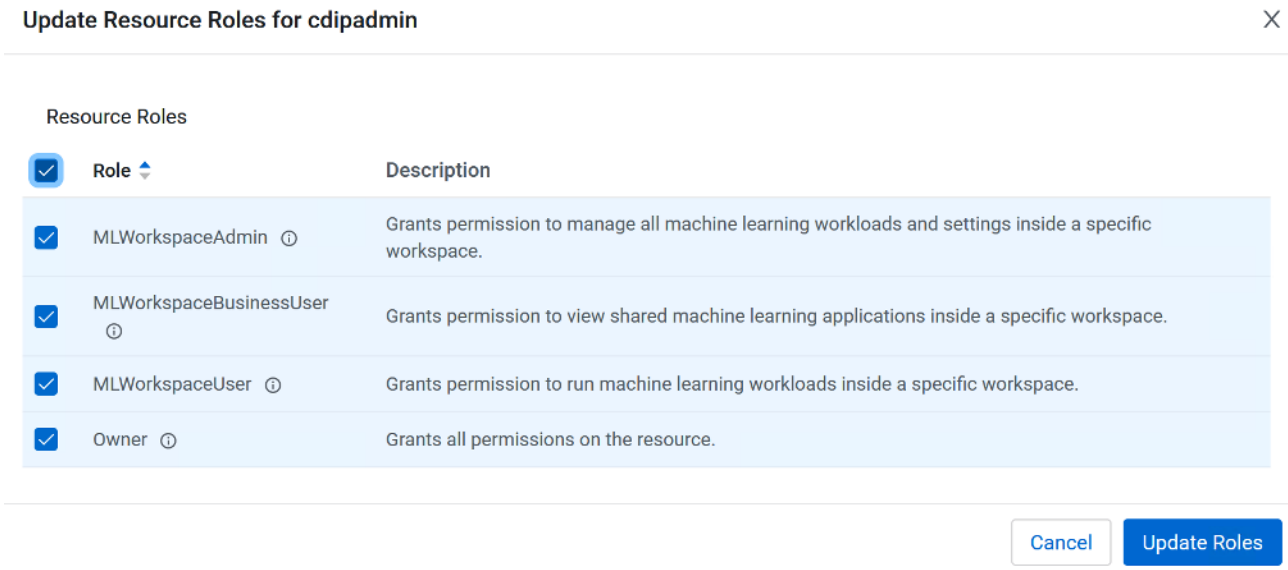
5. Click on ***Manage Access***.



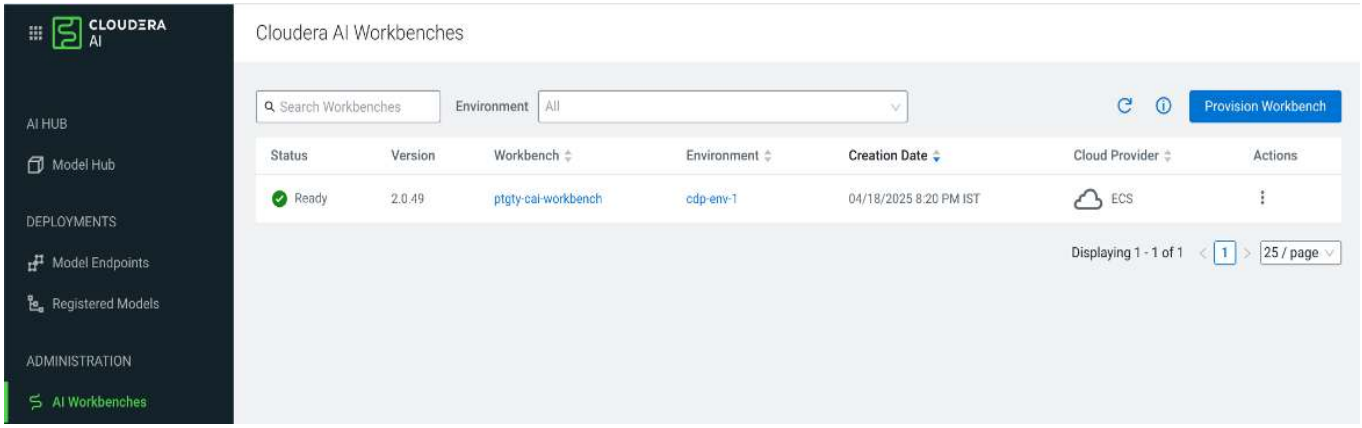
6. In the search field search for a user or group to be able to access AI Workbench.



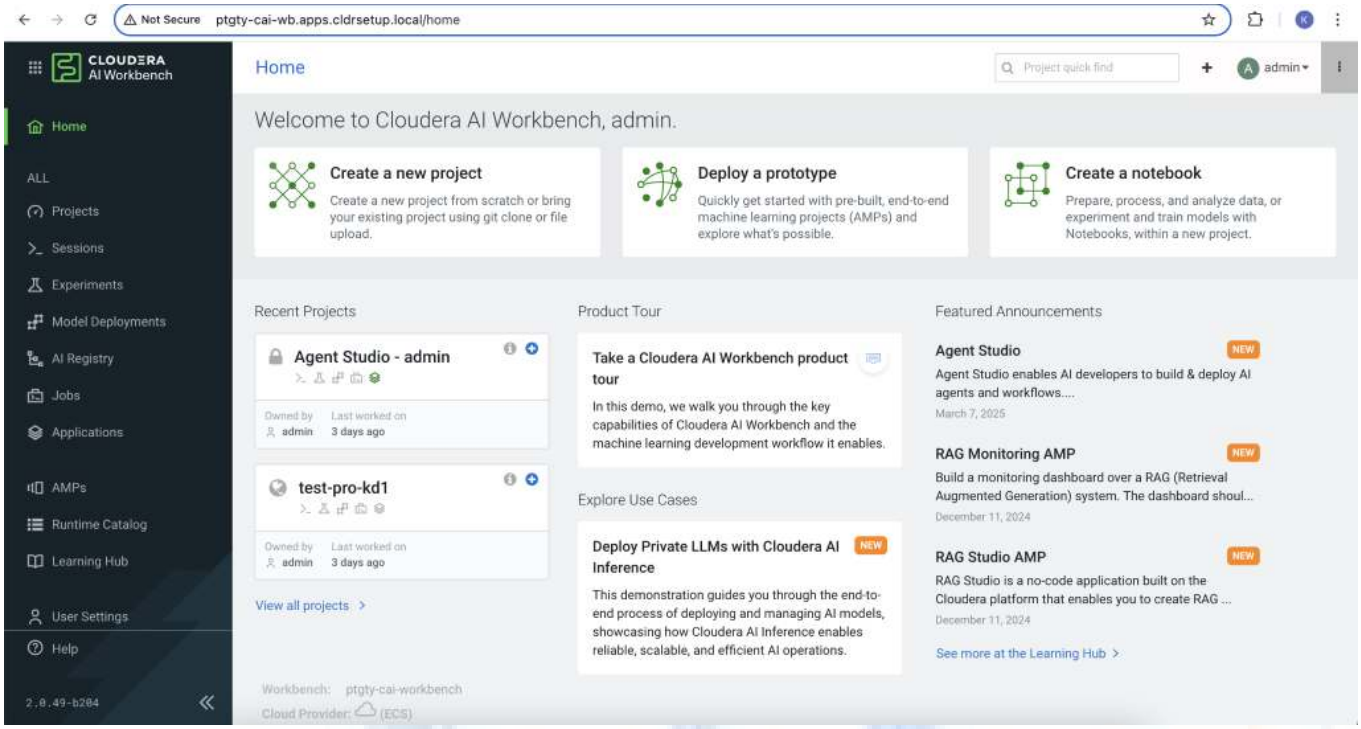
7. Update Resource role for user or group selected to manage access to workbench provisioned in Cloudera AI.



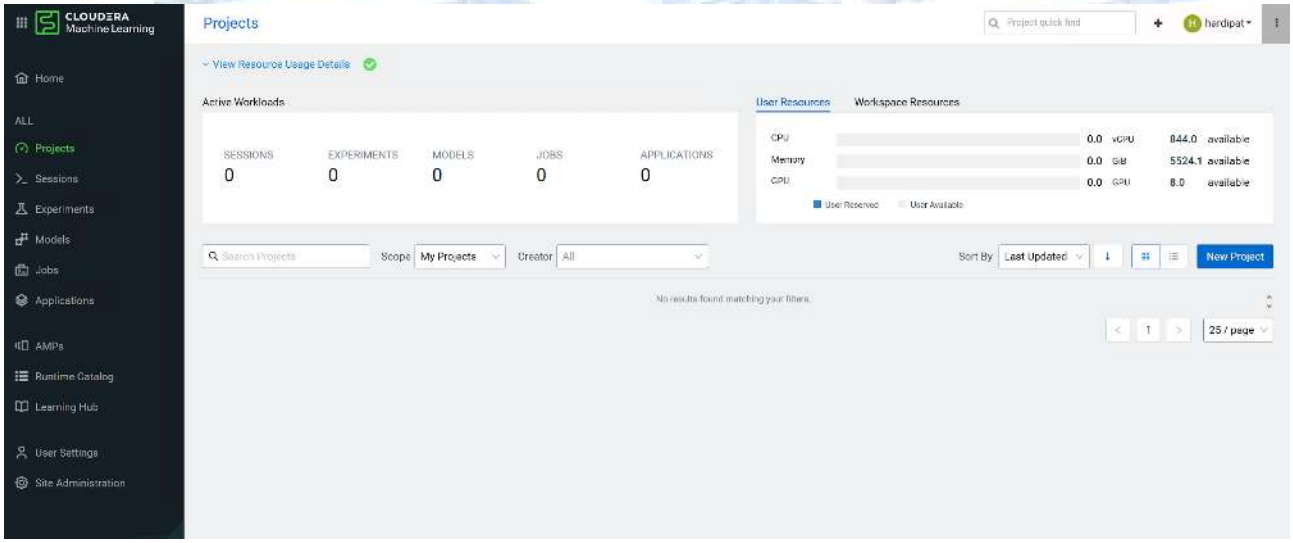
8. Click on the workbench name created.



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:

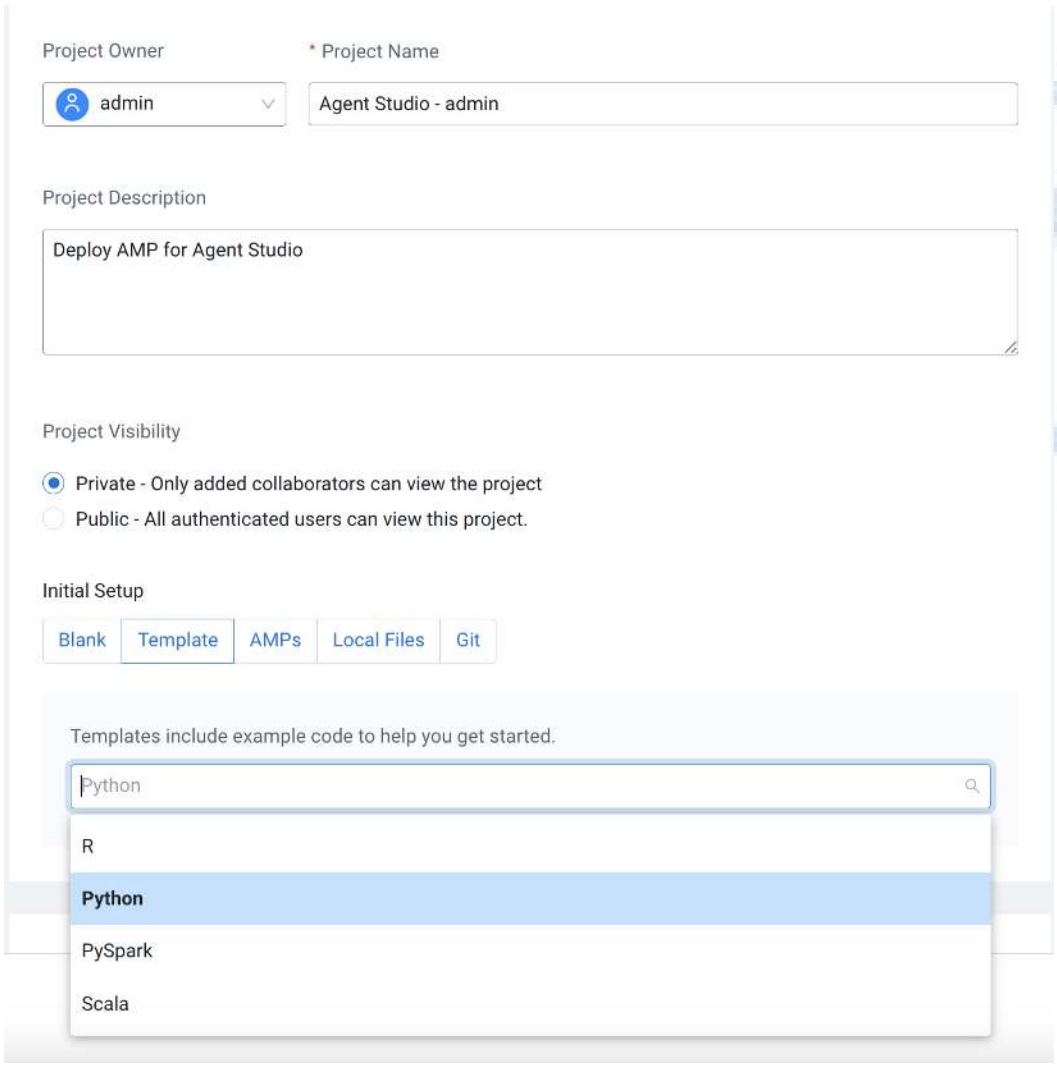
12. At the middle right, you will find the ***New Project*** button. Click on it. New Project page appears. Enter the details as described below. Enter project name and select type of initial setup.

- **Project Name**: Enter a suitable name for your project.

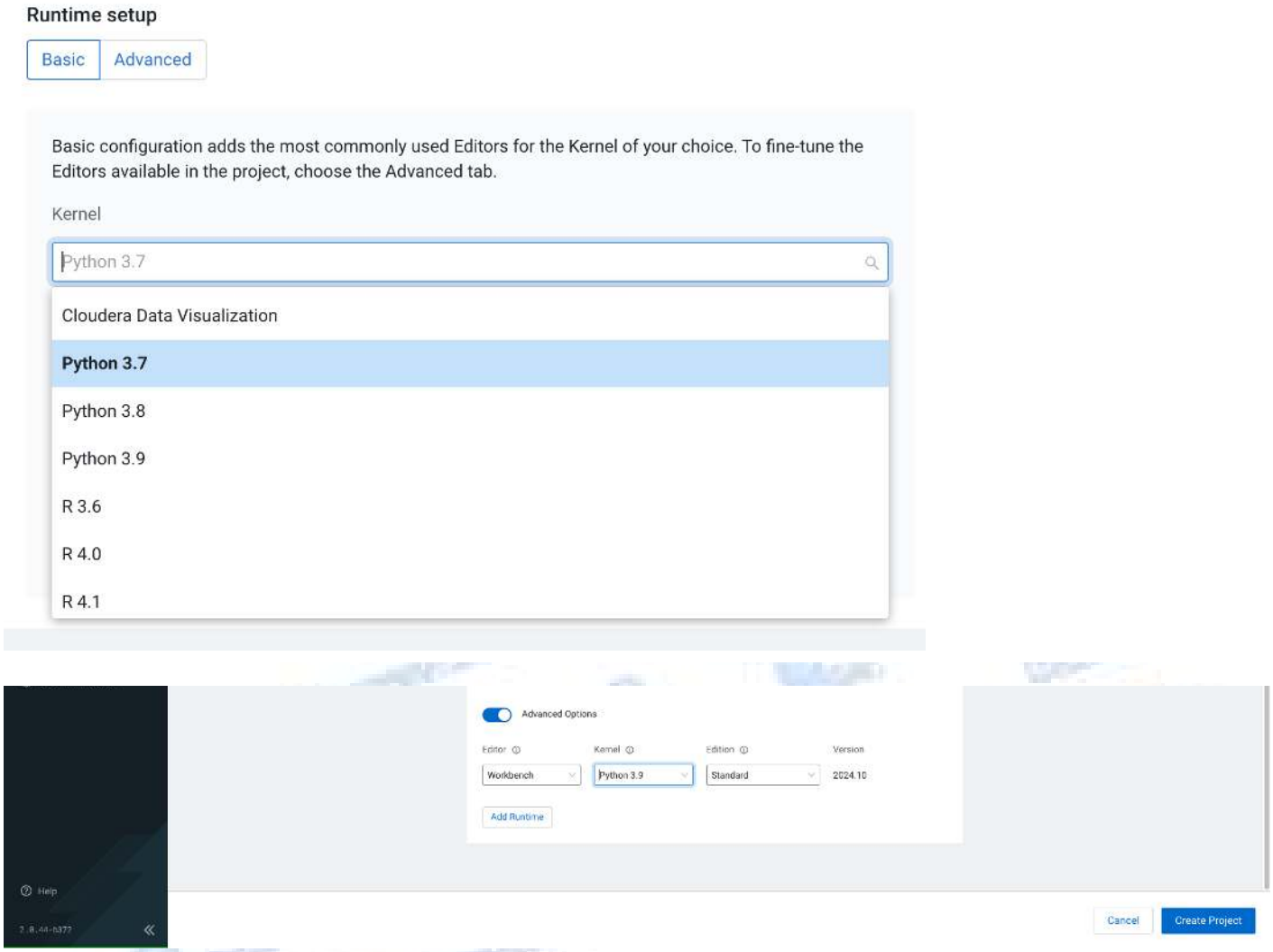
- **Project Description**: Enter a description for the project.
- **Project Visibility**: Keep it Public for any demos or PoC's. If you are creating this in a multi-tenant environment, choose Private.

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.

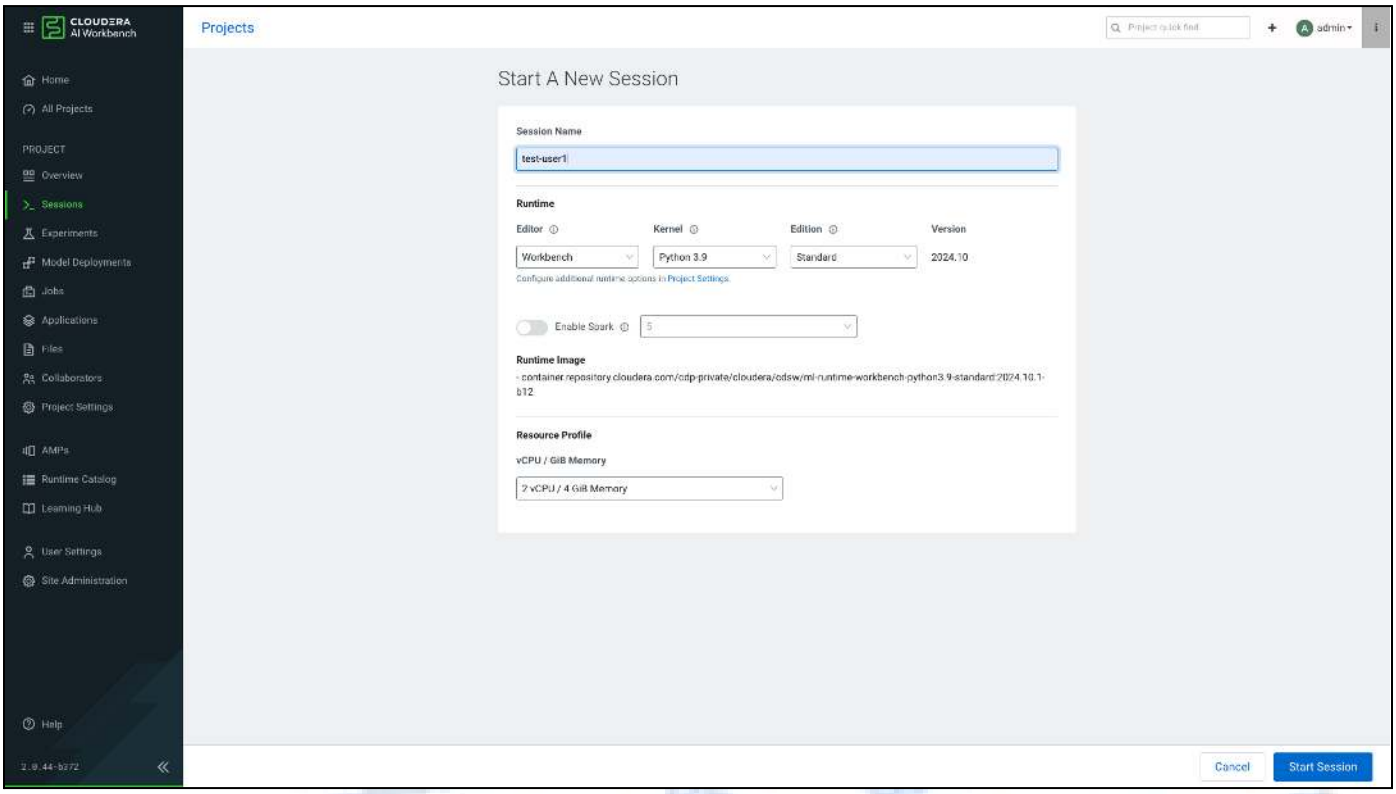


13. Select ***Runtime setup***. For initial exploration, select Basic and keep the kernel to ***Python3.9*** (enable checkbox to add GPU enabled Runtime variant, if applicable — **We are not using GPU in our current setup**).



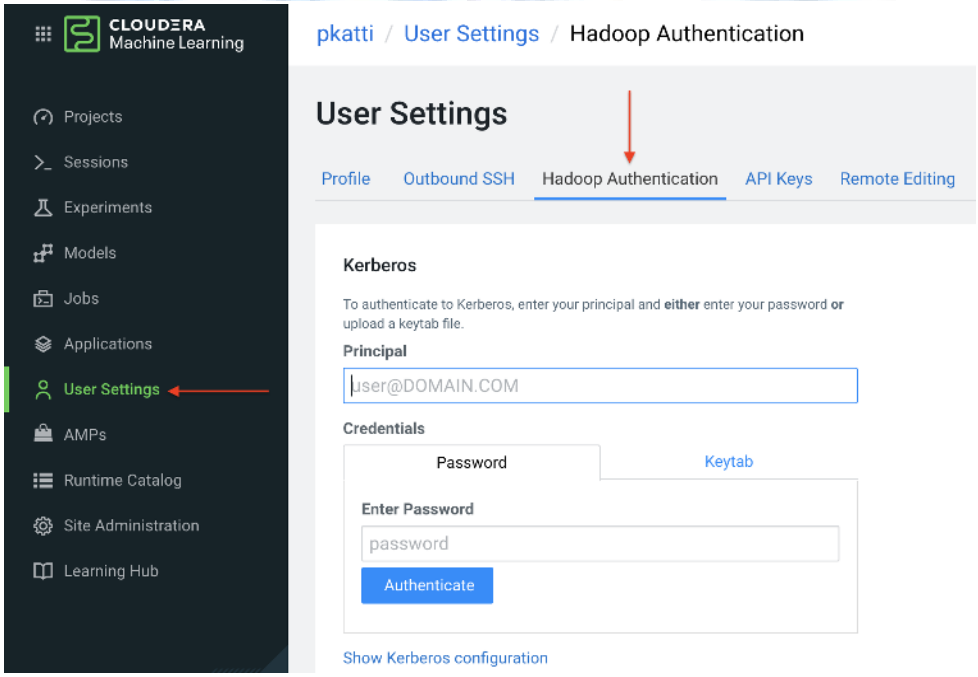
14. Click on *Create Project*. After some time, a new project will be created and will be available on the Projects page.

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.

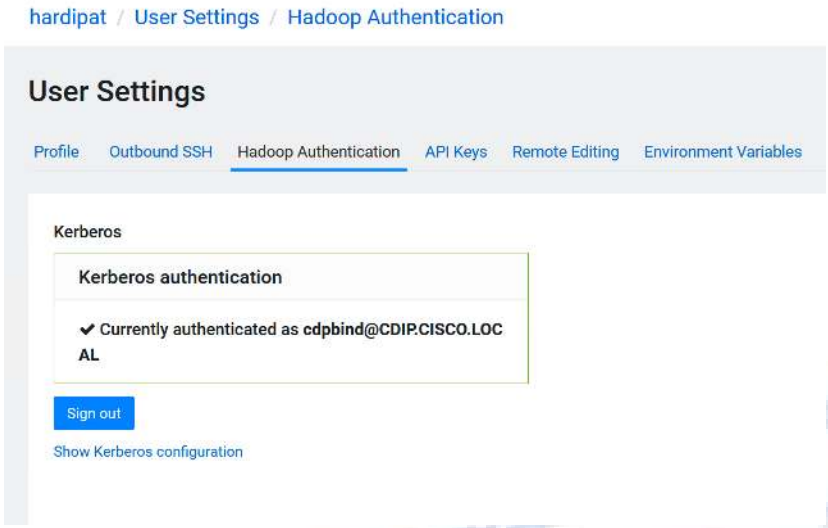
So, to be able to access data from Hadoop clusters ***go to user > user settings > Hadoop authentication.***
Open a ***new tab in the same browser*** window, by duplicating the existing tab. Go to the ***CAI home page*** and click on ***User Settings*** in the left pane. Click on ***Hadoop Authentication***.



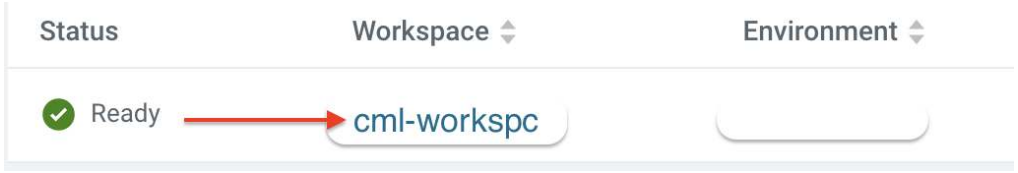
- Enter ***Principal*** e.g. username@DOMAIN.LOCAL

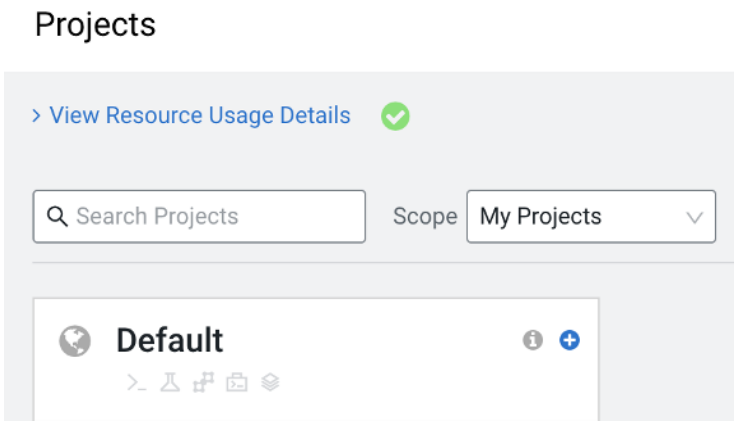
i.e. ***admin@REDHAT.LOCAL*** or ***admin@CDP.RDU2.SCALELAB.REDHAT.COM***

- Under the ***Credentials*** and password (i.e. ***redhat123***) 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.

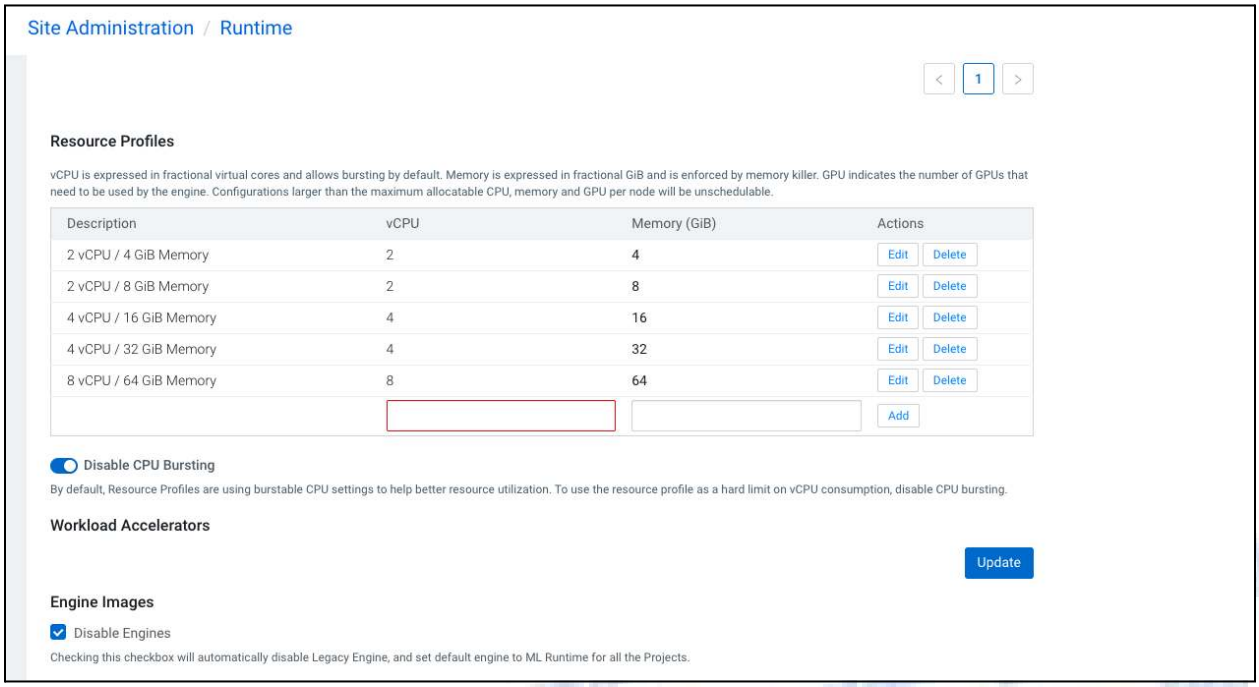


17. Now, to explore the CAI IDEs, click on the newly created workspace. It will open the Projects screen of CAI. Go to the ***Project page*** and click on the newly created project and then click on the ***New Session*** button on the top right. Explore CAI by running the jobs with different IDEs like Jupyterlab and Workbench.

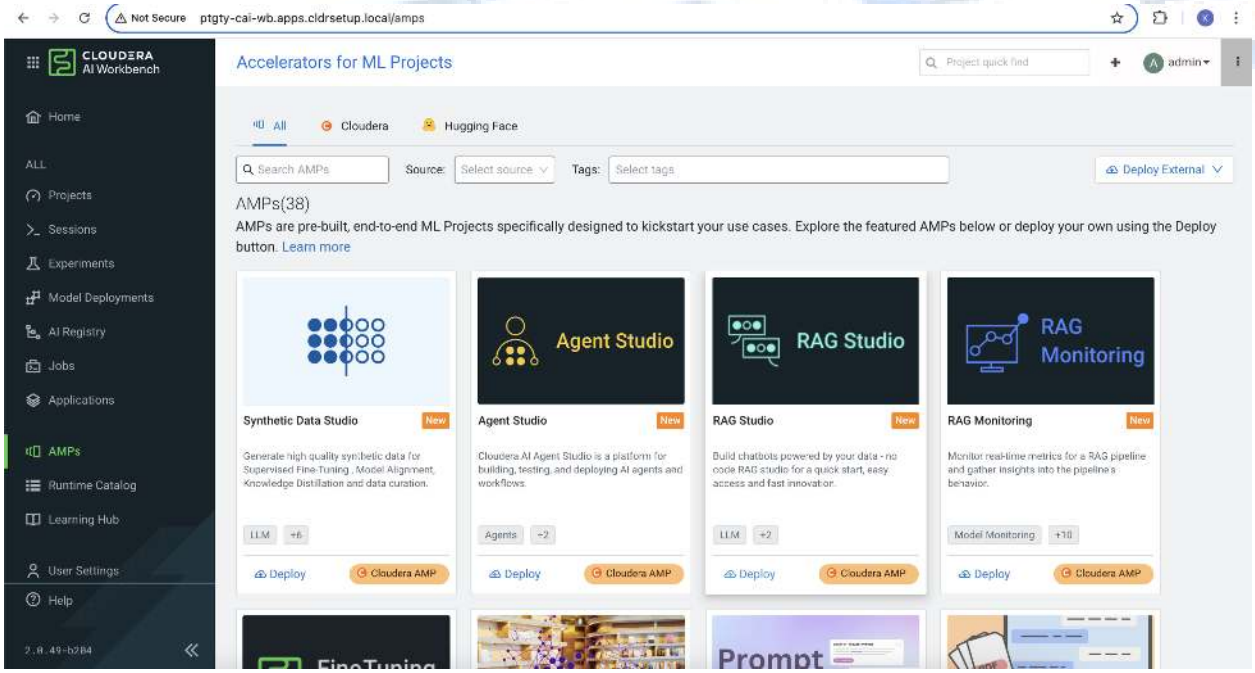




18. Go to ***Site Administration*** to edit ***Resource profile*** and ***GPU per session/ Job***.

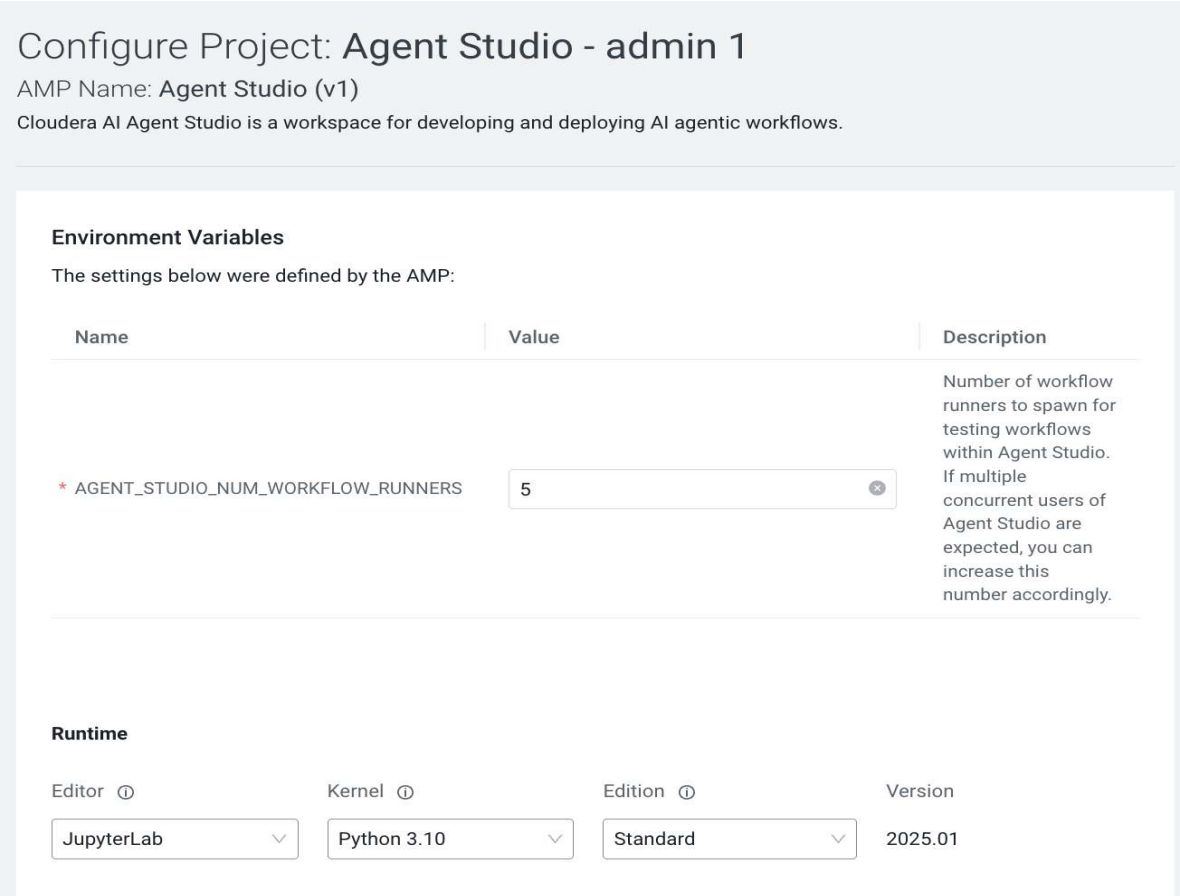


19. Go to the ***AMPs*** tab to get started with pre-built models.

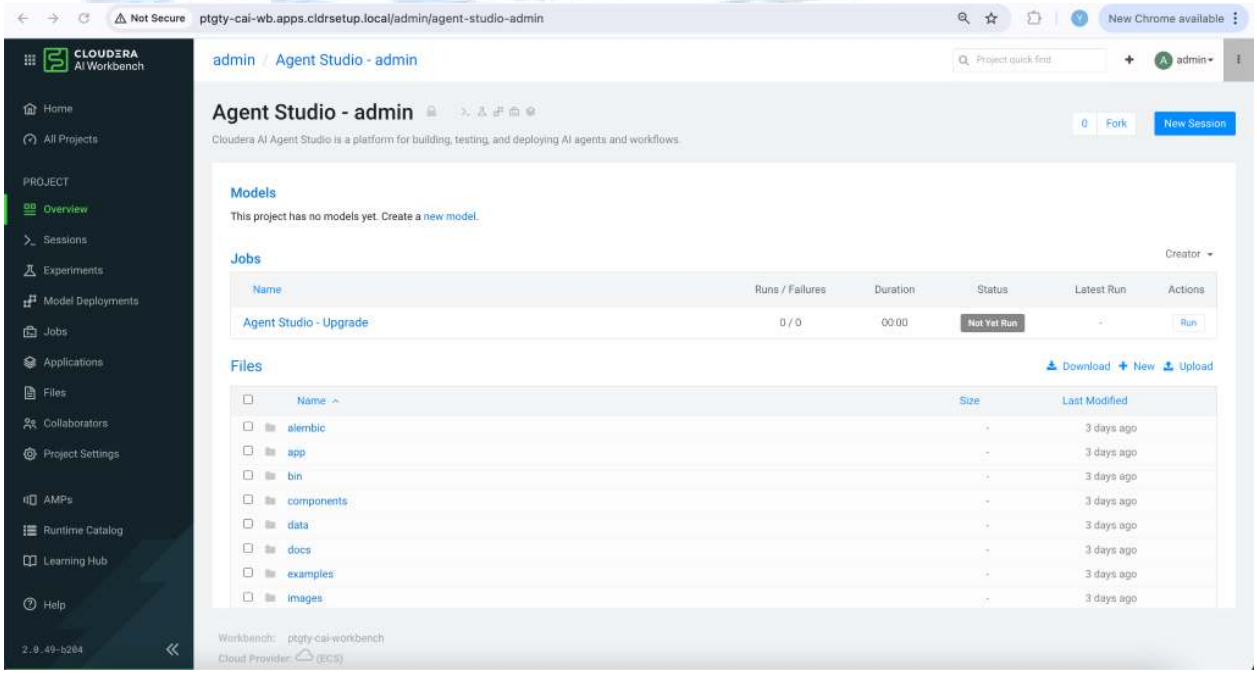


20. Select ***AMP*** and click on ***Configure Project***.

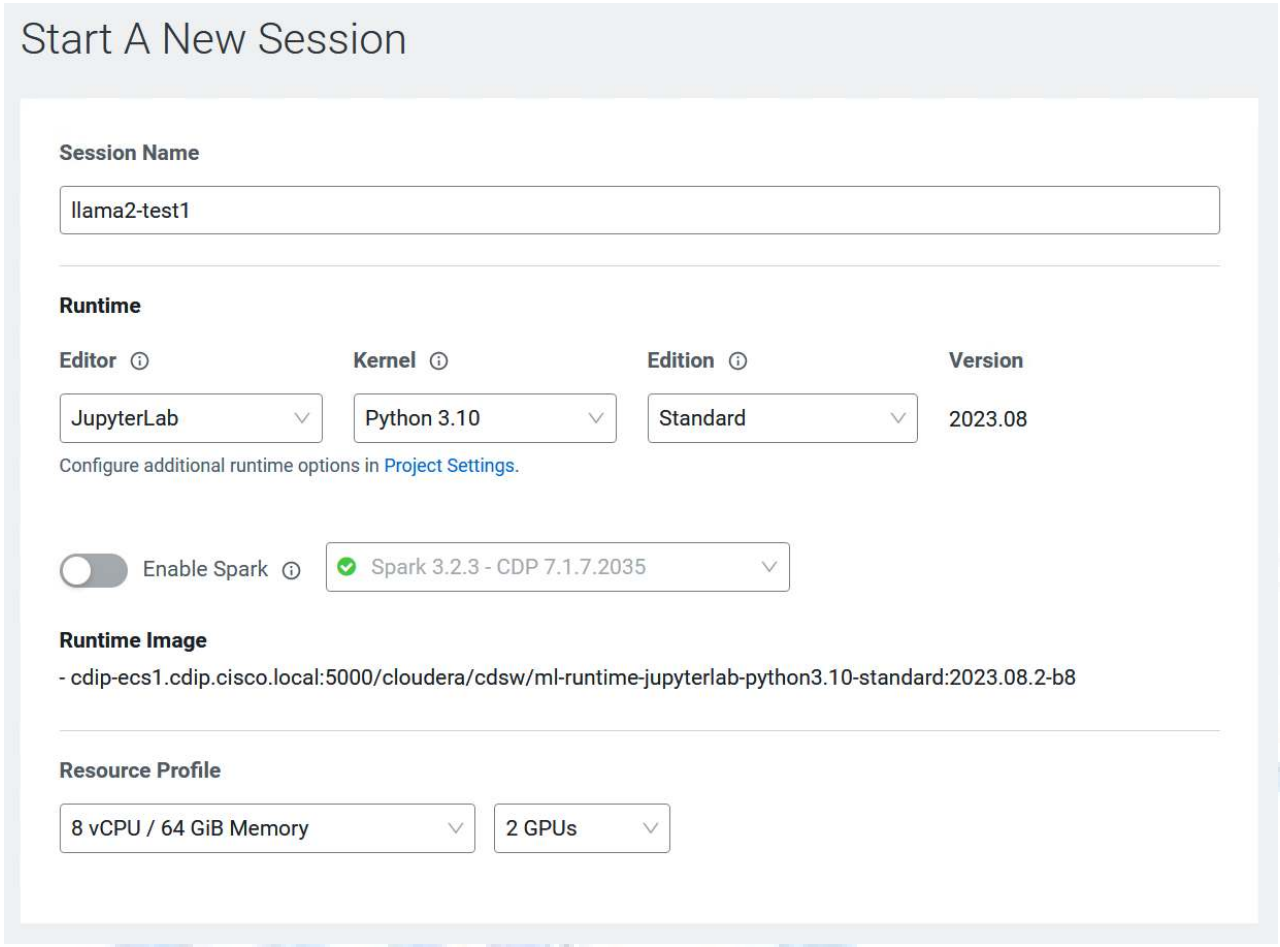
21. After editing the ***Runtime field*** for the new project, click on ***Launch Project.***



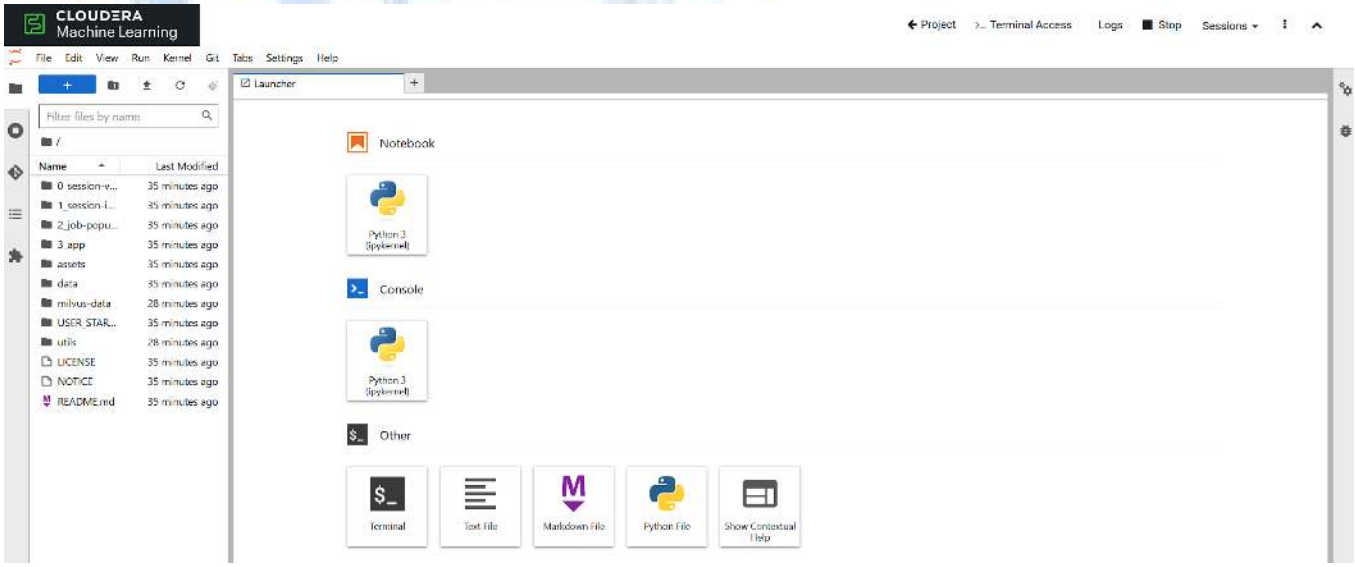
22. ***Agent Studio - admin – AMP*** project overview.



23. Create a new session by ***Start A New Session*** with desired resources, editor, kernel, and number of GPUs.

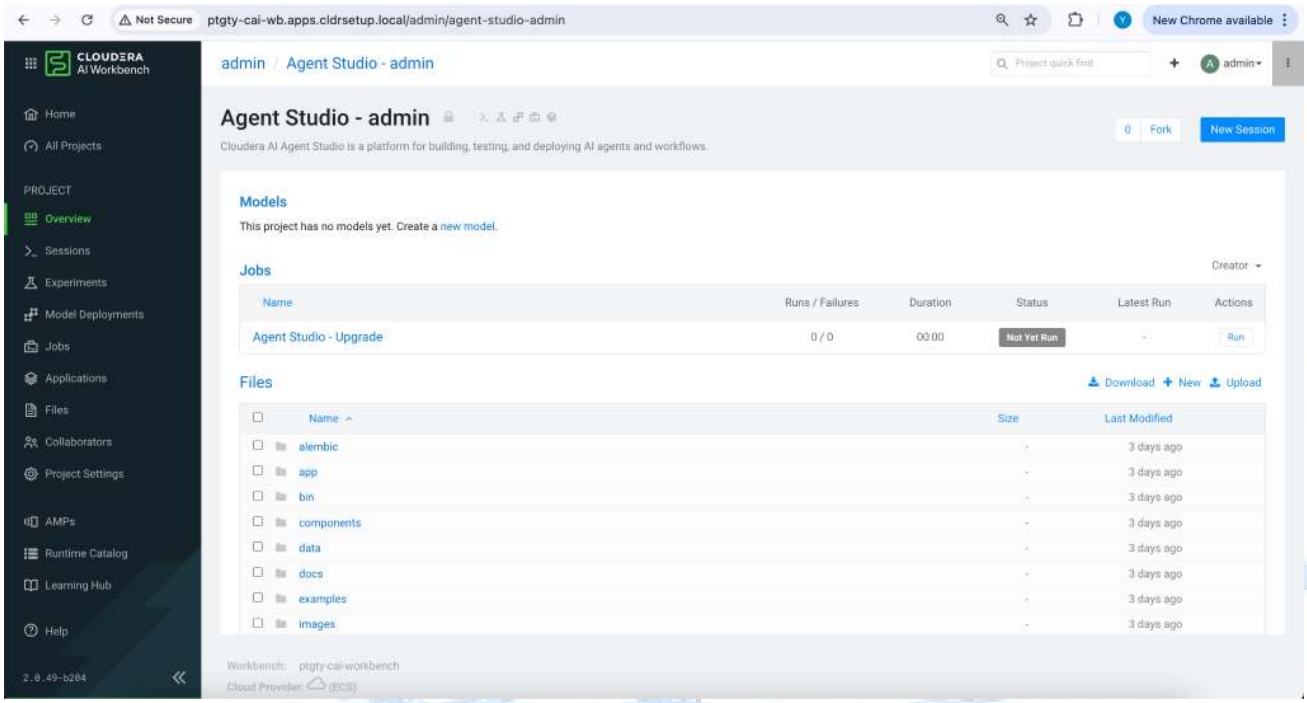


24. ***Jupyter notebook*** session in ***CAI***.

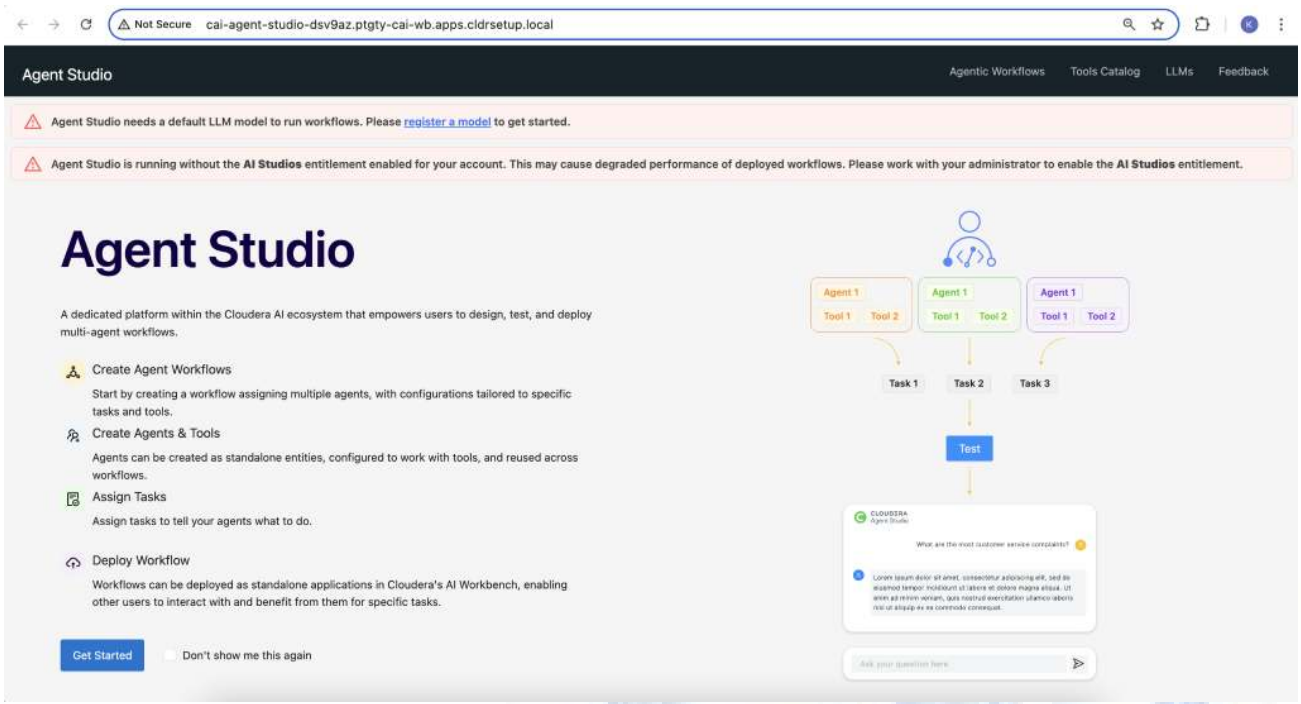


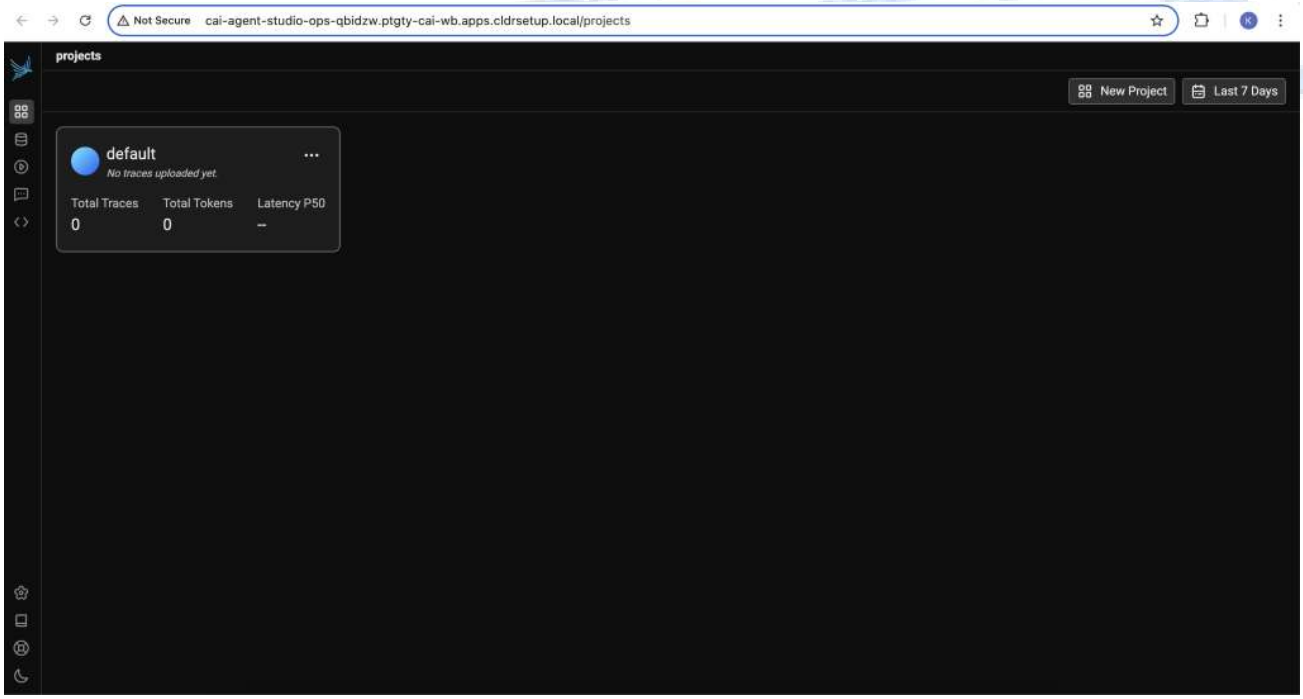
25. Open ***AMP*** created project. To access ***WebUI*** click on ***Open***.

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 OCP master's IP if you're mapping for Data Services. For example: **`ptgty-cai-wb.apps.redhat.local`** should map to the OCP master IP to be accessible via your browser.

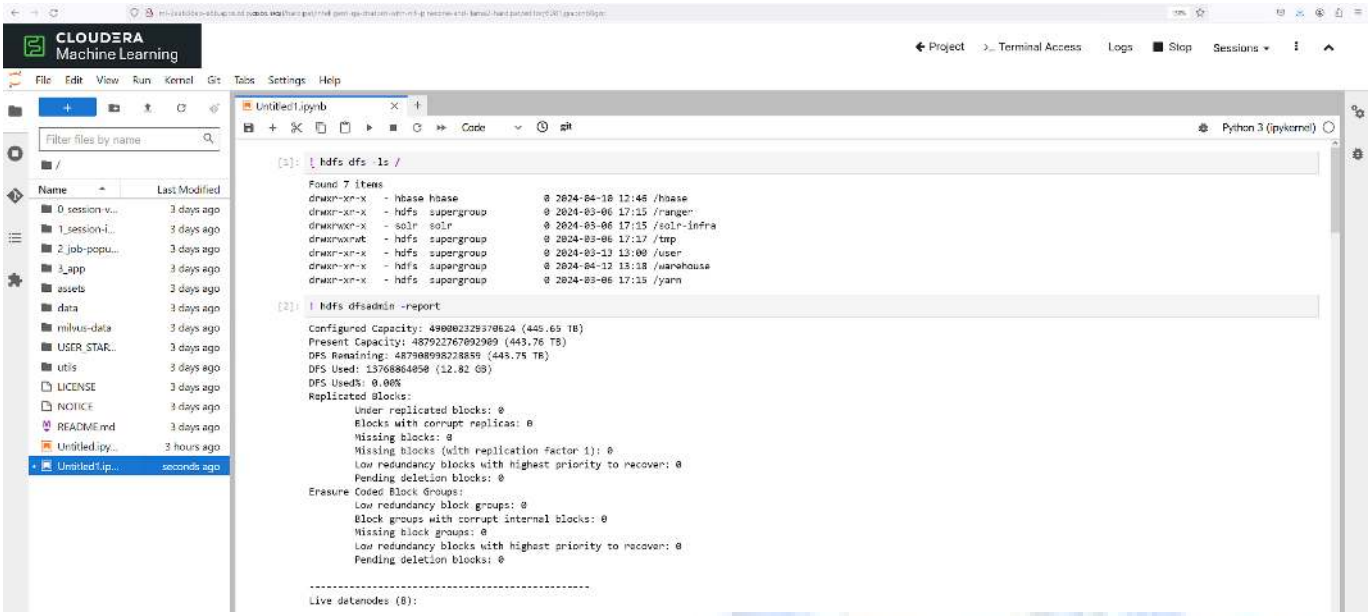


26. WebUI for ***Agent Studio - admin*** with pre-trained data is now available. ***Change settings*** on the WebUI or use them out of the box. For example, we questioned "***what is Cloudera Data Platform?***".





27. ***Access HDFS data*** from the ***jupyter notebook session*** in CAI.

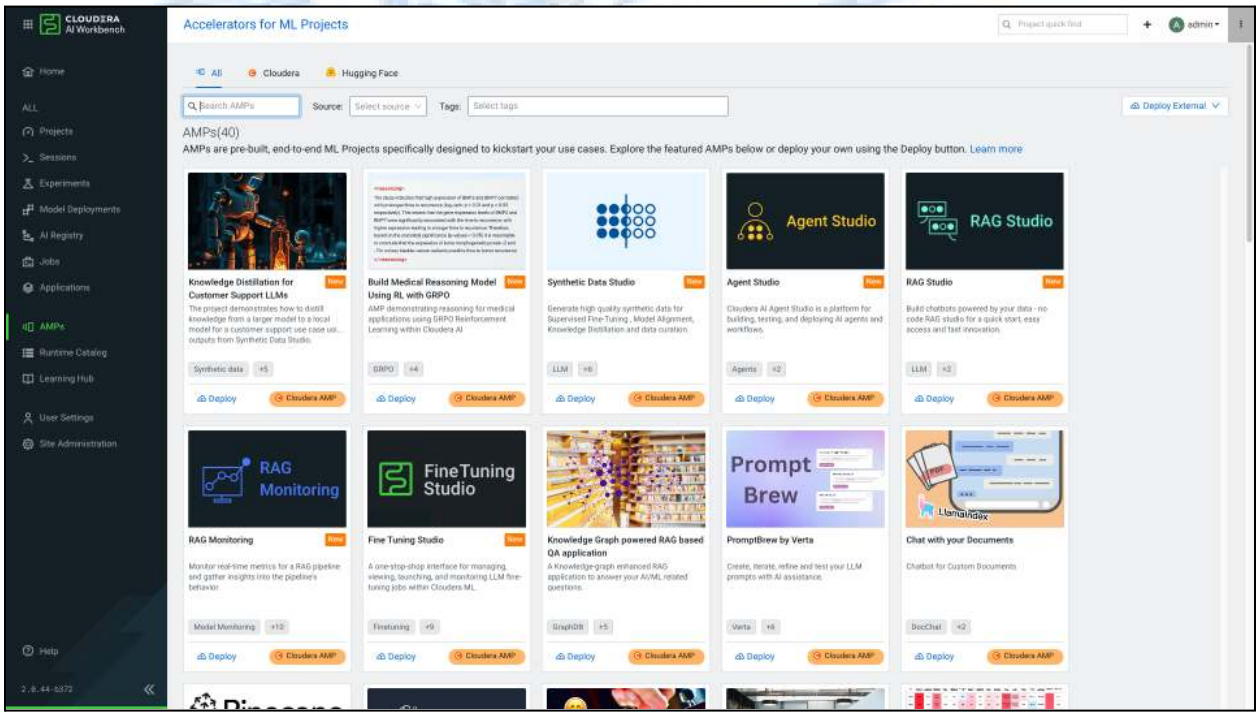


Note: Deploying and documentation of every aspect of ***AI Workbench, project, and user management*** is not covered here. Please refer to the related ***Cloudera documentation*** on ***Cloudera AI How to section*** for more details:

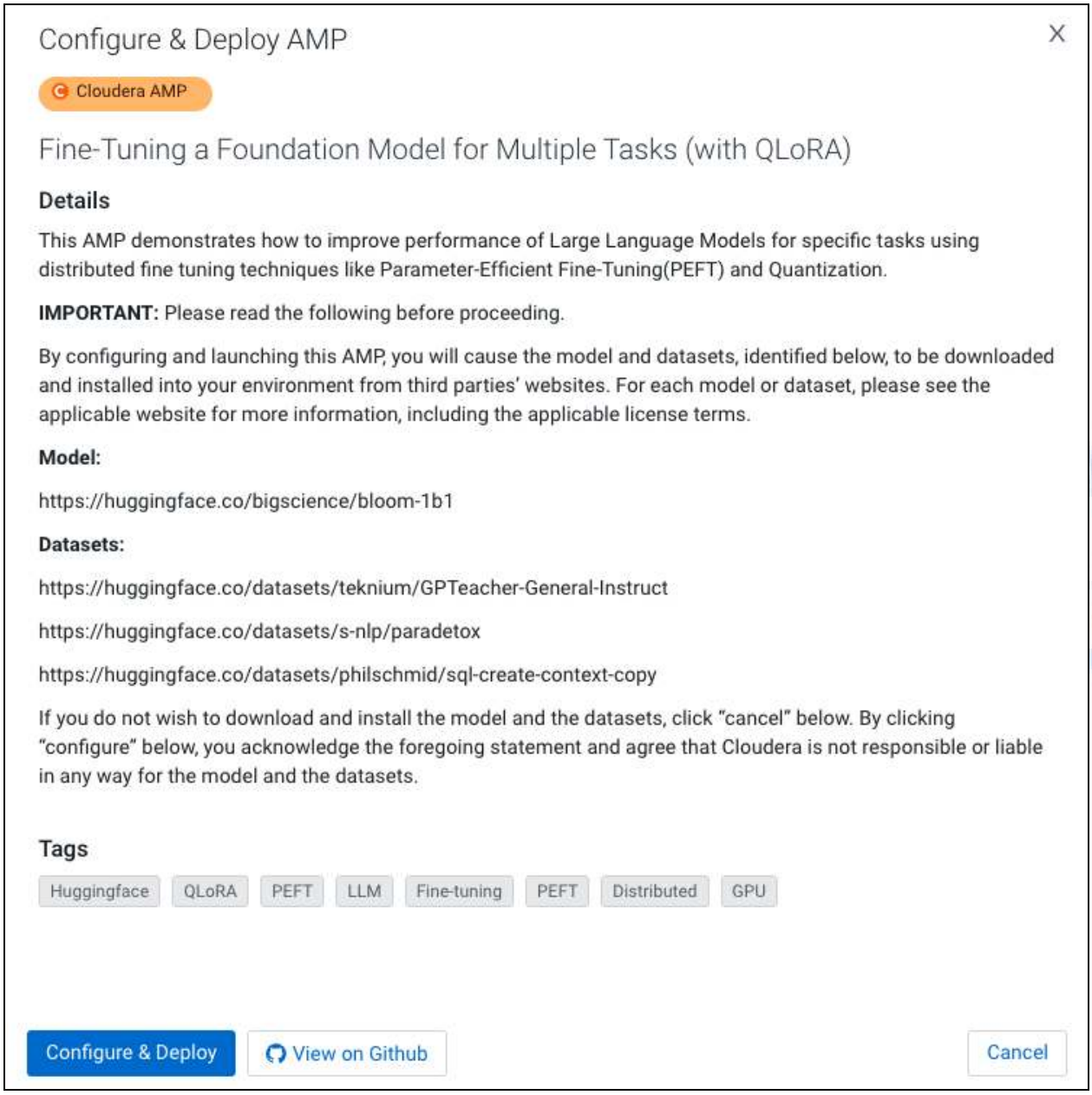
https://docs.cloudera.com/machine-learning/cloud/product/topics/ml-product-overview.html

Creation of another AMP - Fine-Tuning a Foundation Model for Multiple Tasks (with QLoRA)

28. Go to the AMPs tab to get started with pre-built models.

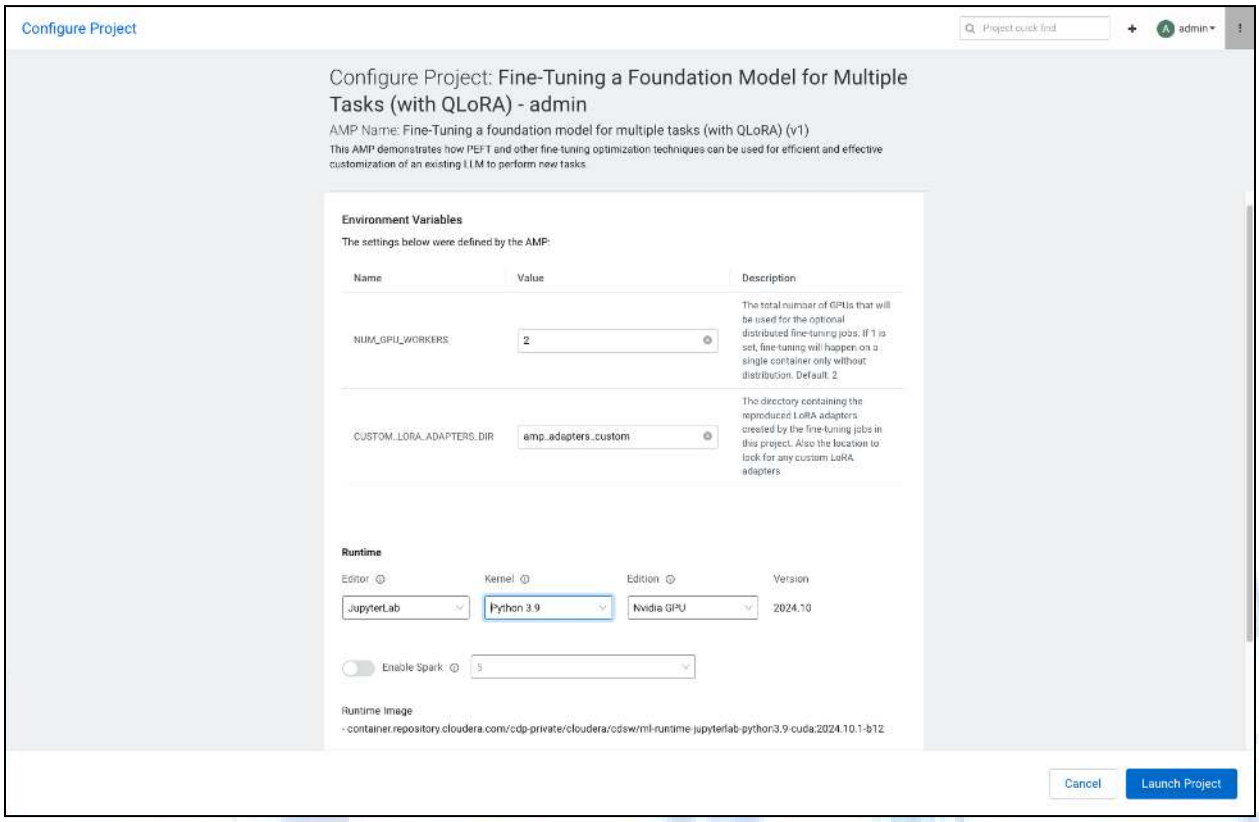


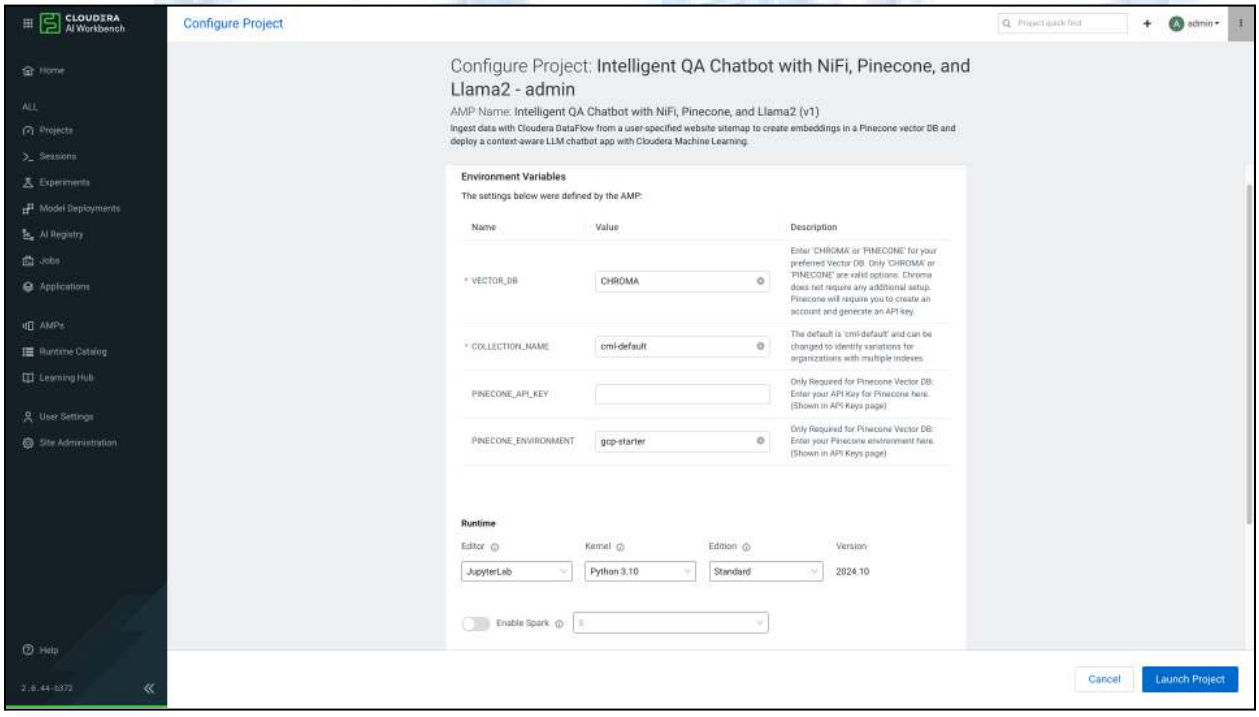
29. Select AMP and click on Configure & Deploy.



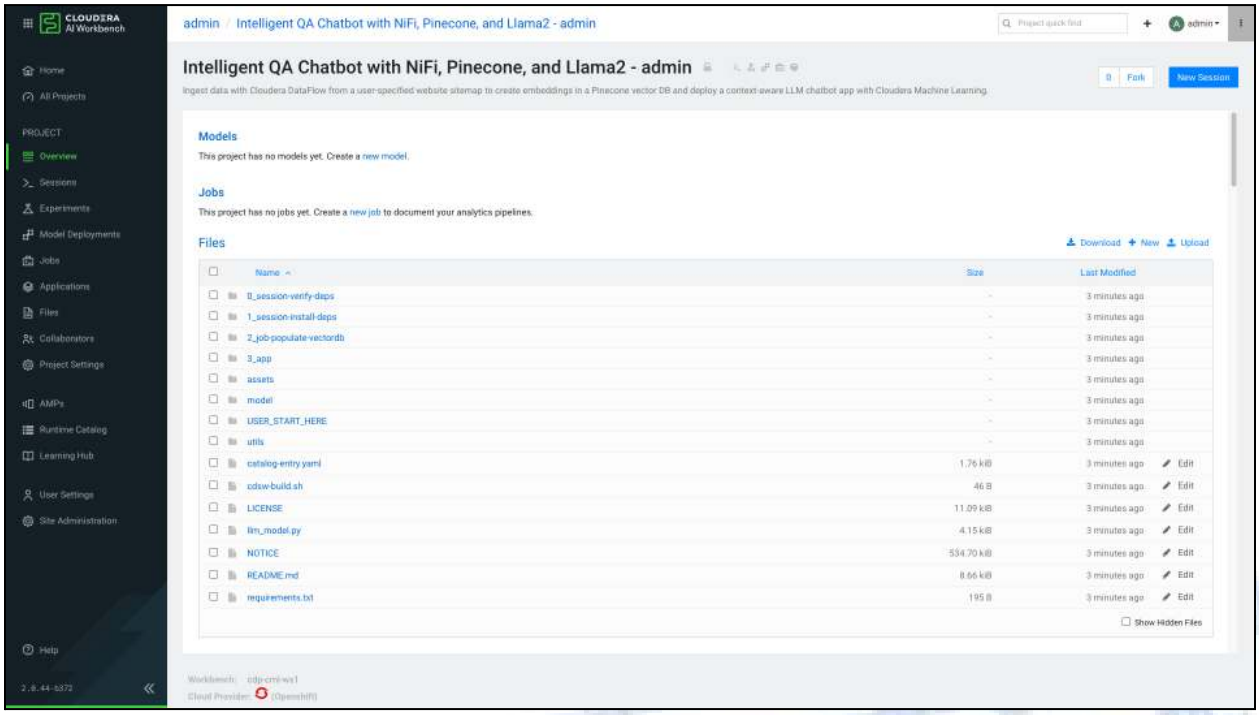
30. After editing the Runtime field for the new project, click on ***Launch Project***.

31. Intelligent QA Chatbot with NiFi, Pinecone, and Llama2 – AMP project overview.

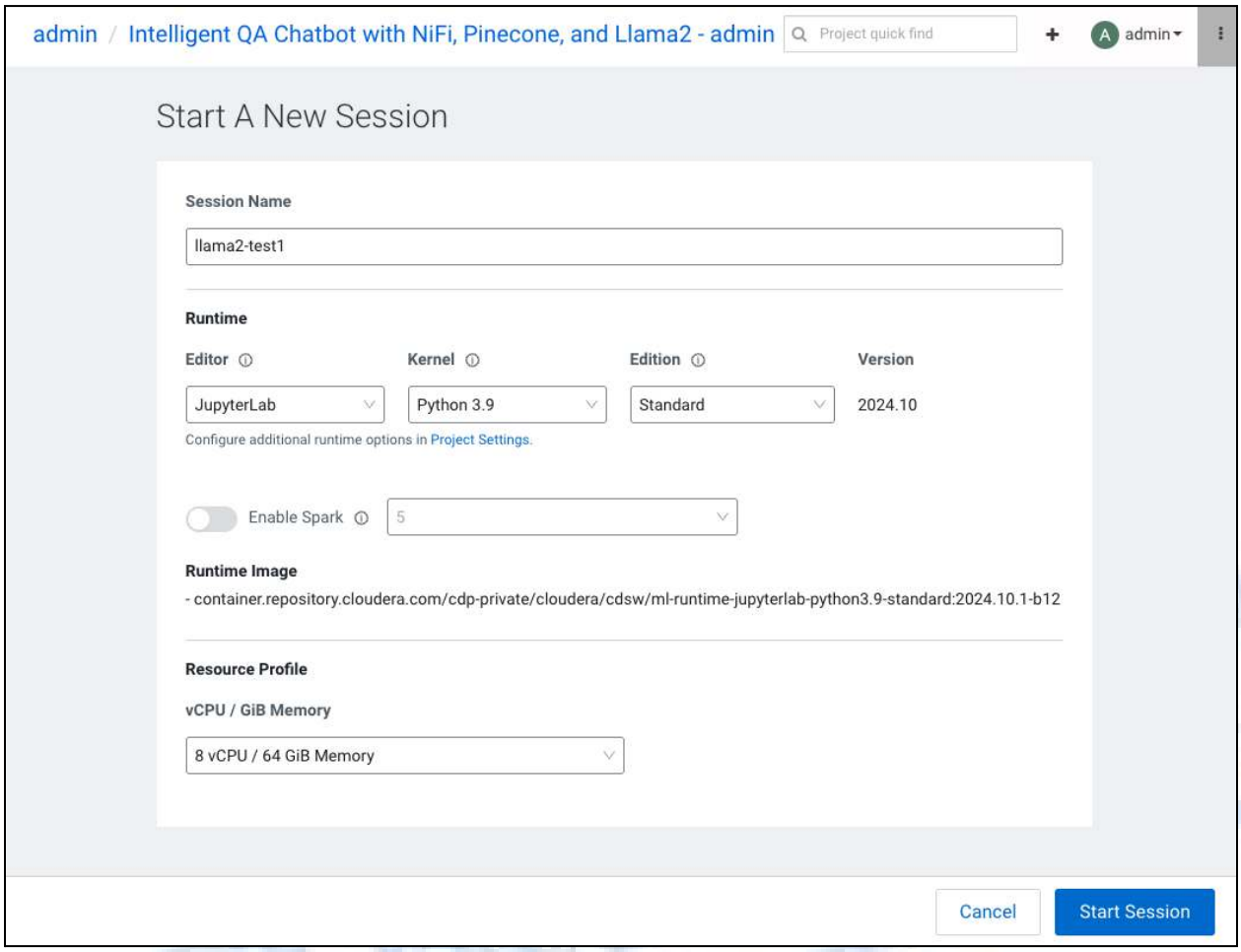




32. Create a new session; click on ***New Session***.



Then with desired resources, editor, kernel, and number of GPUs.



33. Jupyter notebook session in CML.

34. Open AMP created project. To access WebUI click on Open.

35. WebUI for Llama2 based chatbot 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 VMware Platform?”.

36. Access HDFS data from the jupyter notebook session in CML.

Note: Deploying and documentation of every aspect of CML workspace, project, and user management is not covered here. Please refer to the related Cloudera documentation on Cloudera Machine Learning How to section for more details:
https://docs.cloudera.com/machine-learning/cloud/product/topics/ml-product-overview.html

****
****

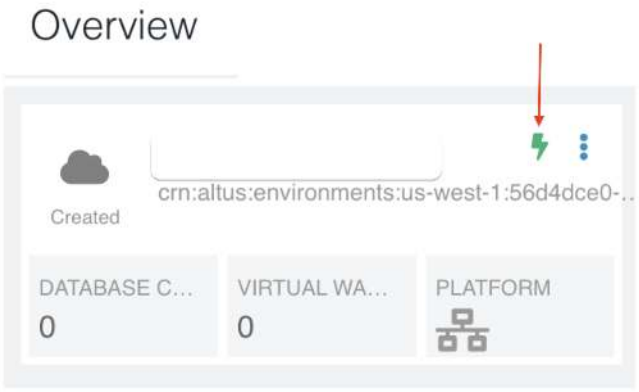
Cloudera on premises Data Warehouse (CDW)

https://docs.cloudera.com/data-warehouse/1.5.5/private-cloud-getting-started/topics/dw-private-cloud-create-virtual-warehouse-openshift-overview.html

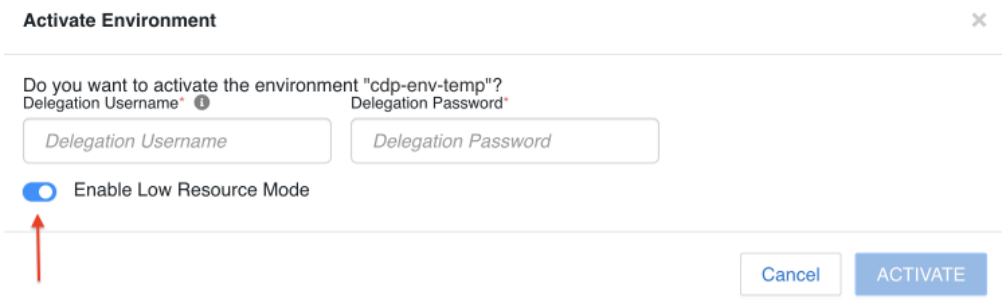
Enable CDW environment and creation of Database Catalog

- Open ***CDP Data Services*** page.
- Click on ***Data Warehouse***.

- On the ***Overview*** page, click on the ***Activate*** icon as shown below.

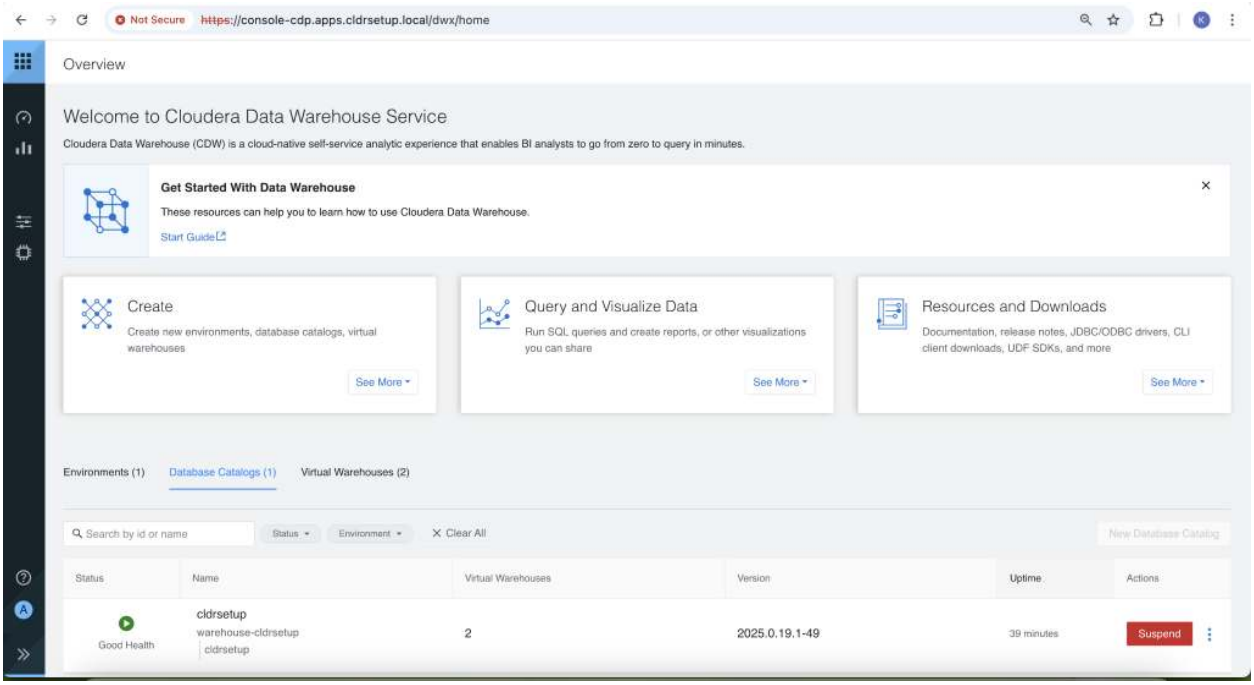


- On the ***Activate Environment*** page, enter the ***LDAP username and password***. Enable ***Low resource mode*** and click on ***Activate***.



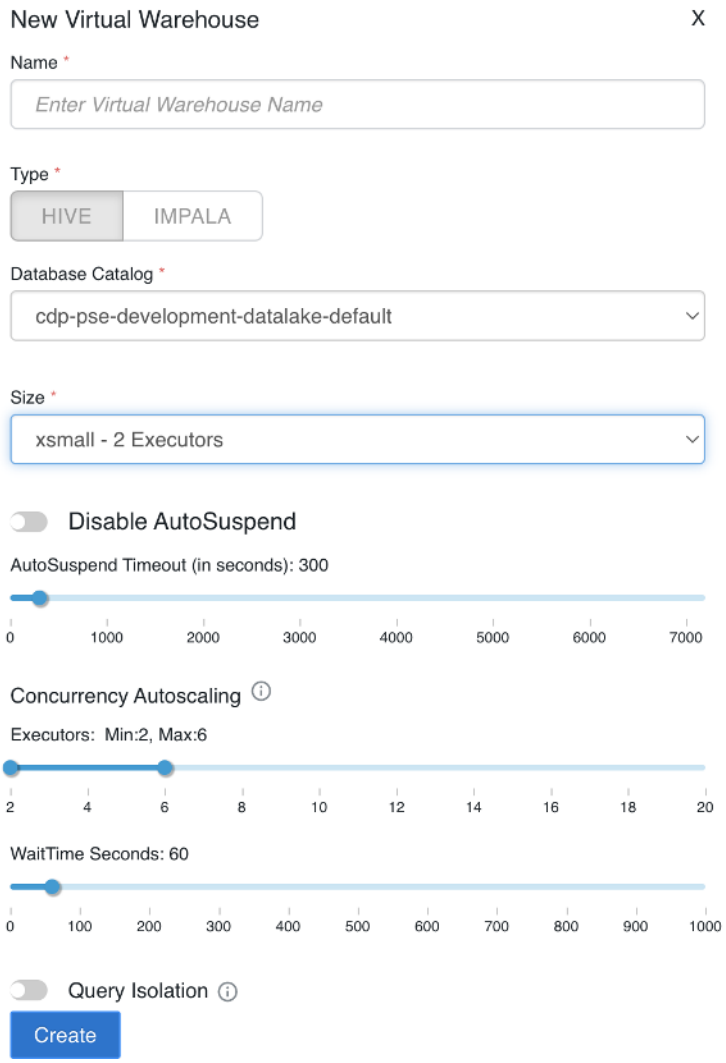
Create Virtual Warehouse

- Once the ***environment*** is ***activated***, a ***default Database Catalog*** gets created automatically.

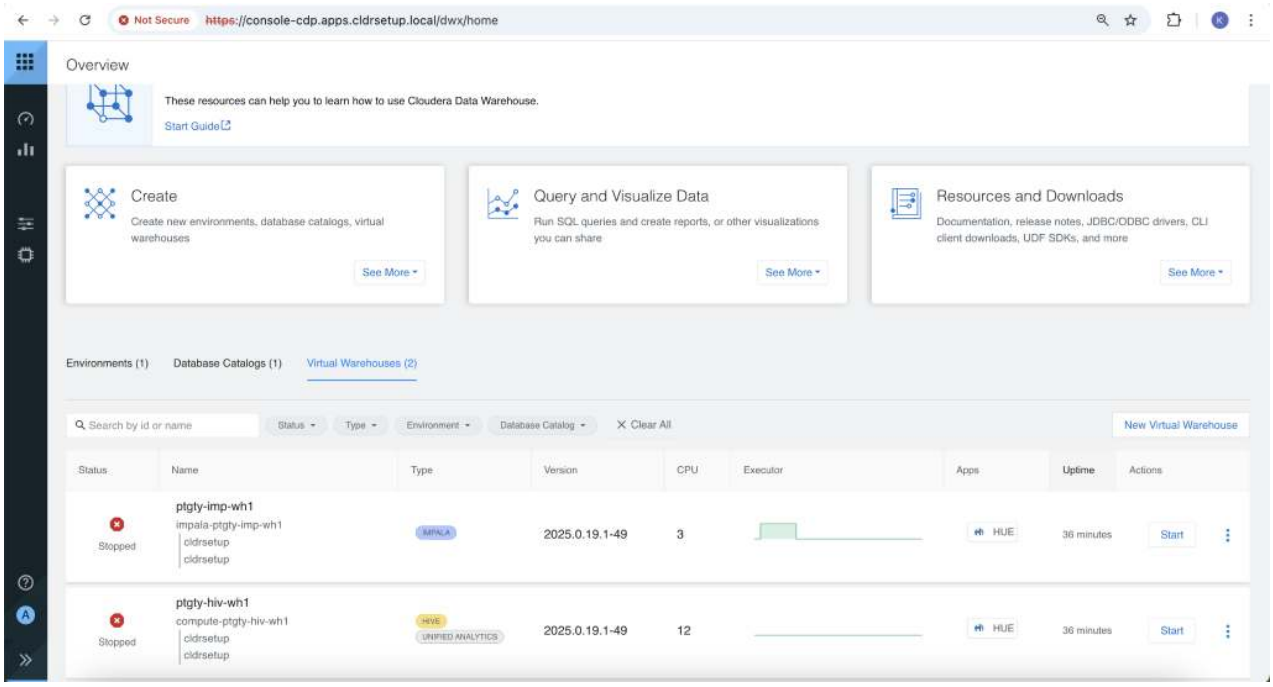


- Once the ***database catalog*** is created, click on **+** icon next to ***Virtual Warehouses***.

- 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)

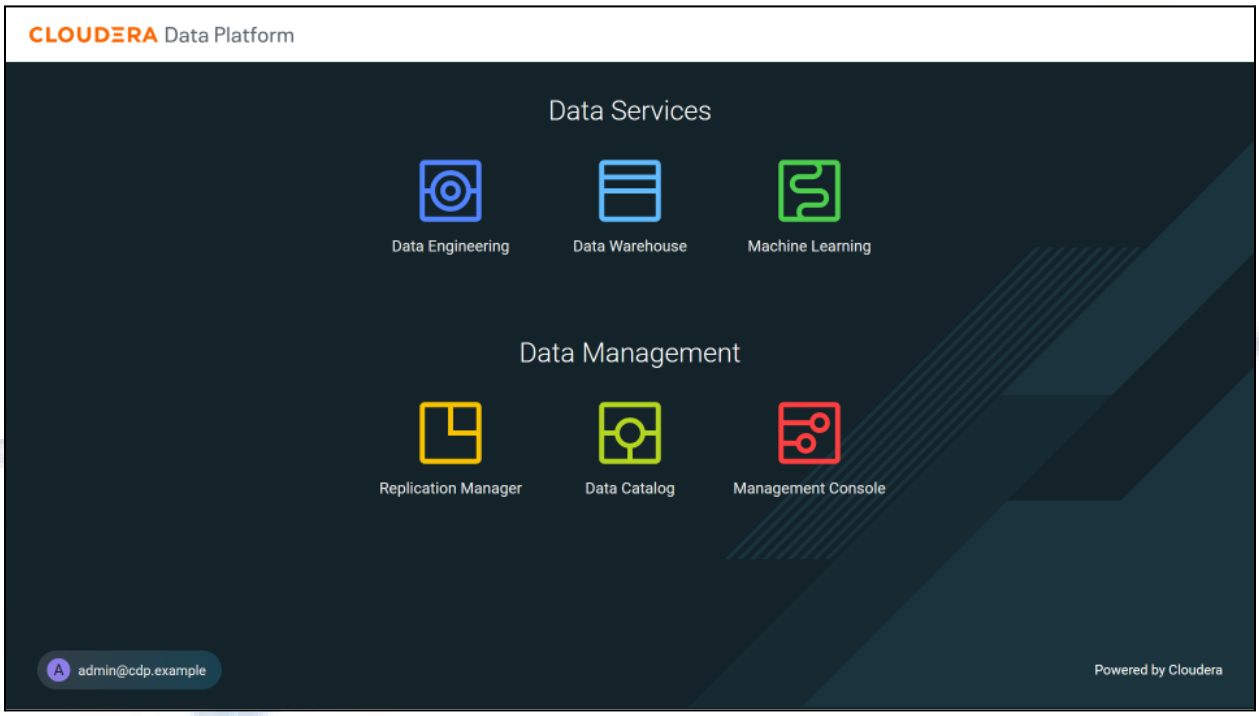
https://docs.cloudera.com/data-engineering/1.5.5/enable-data-engineering/topics/cde-private-cloud-add-cde-service.html

CDP Base cluster requirements:

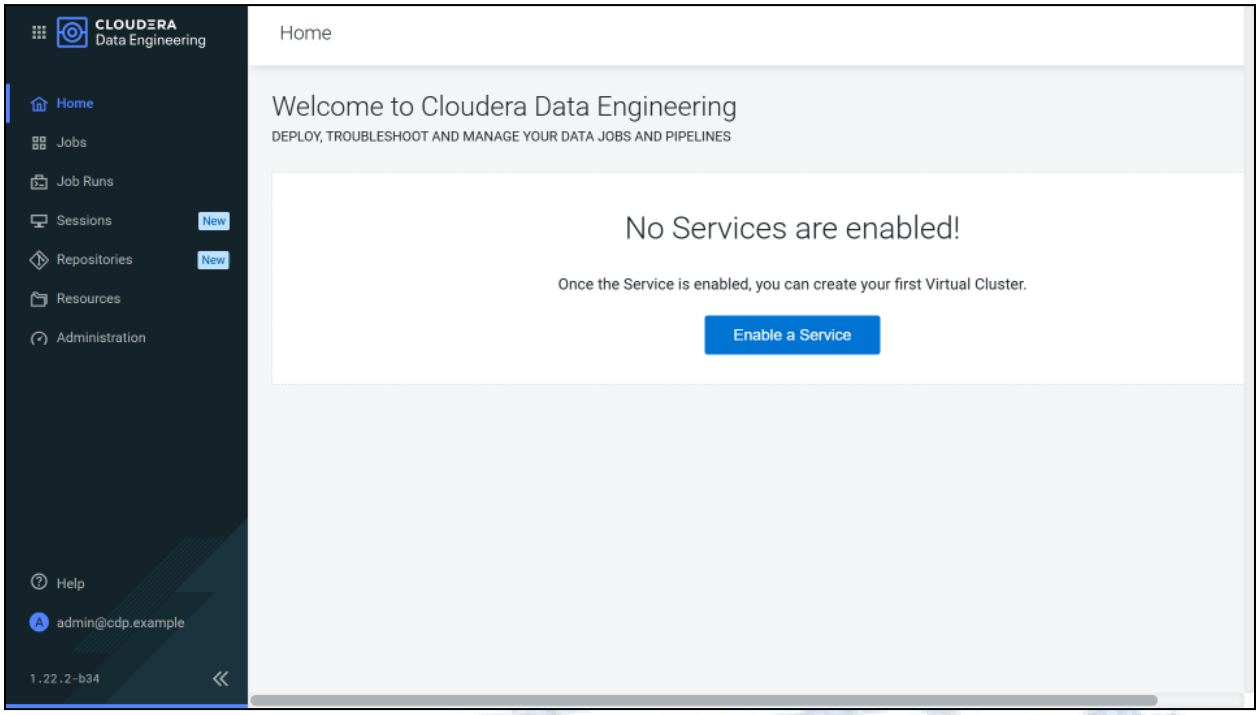
The ***Cloudera Data Engineering (CDE)*** service requires proper configuration of ***Ozone*** service in the Base cluster. Ensure that ***Ozone*** is running properly otherwise you will end up with issues while enabling CDE.

Enabling CDE Service:

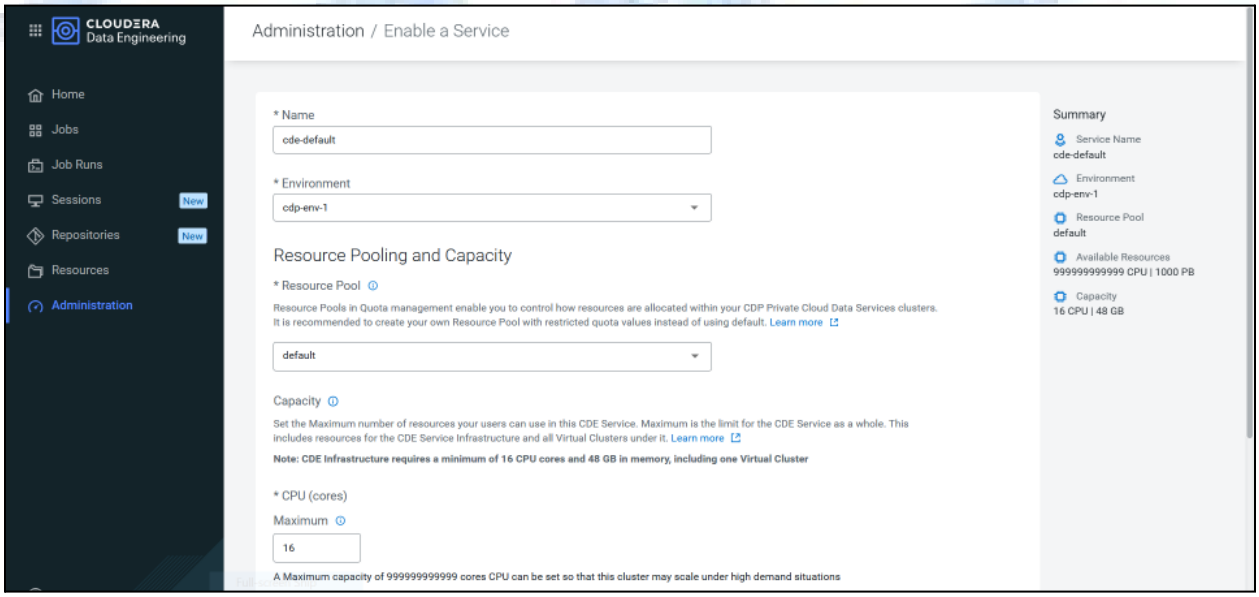
- From the ***CDP console page***, click on ***Data Engineering***.

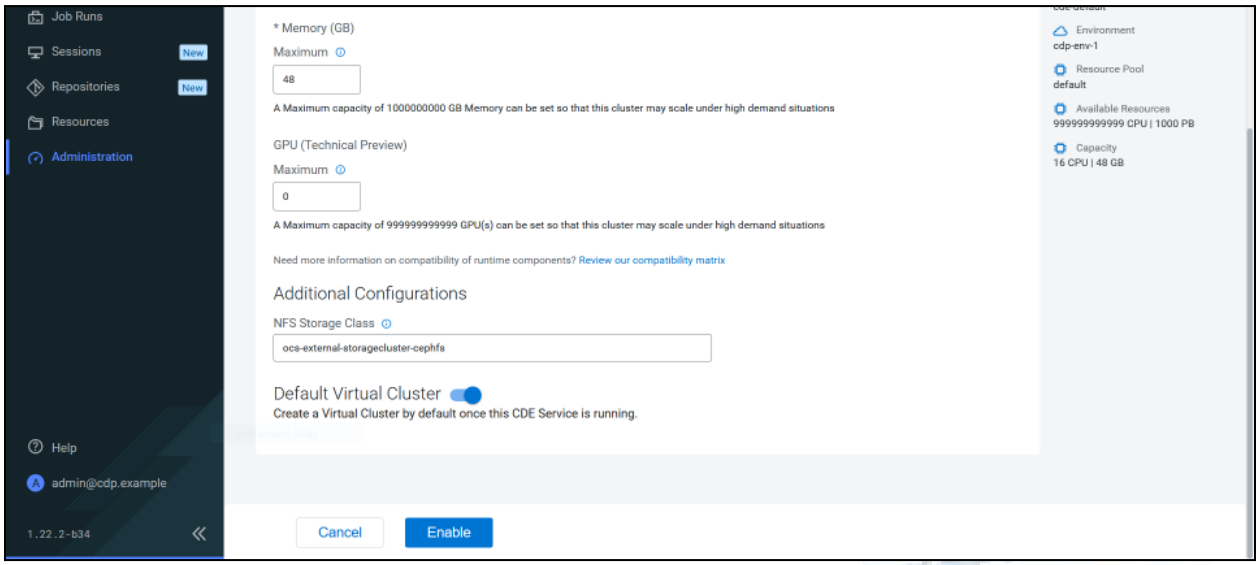


- This will open the ***CDE home page***. Since this will be the first time you will be opening CDE, you will not see any virtual clusters. Click on ***Enable a Service***.



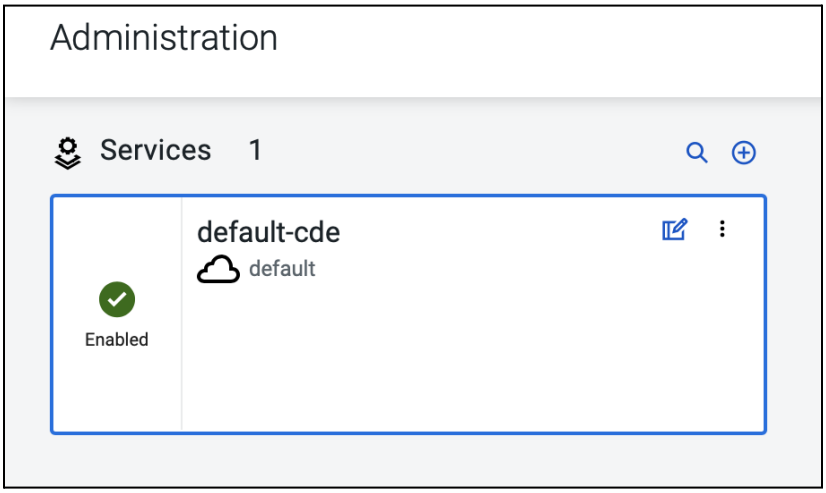
- On the ***Enable a Service*** page, enter the values as shown below and then click on ***Enable***.





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.

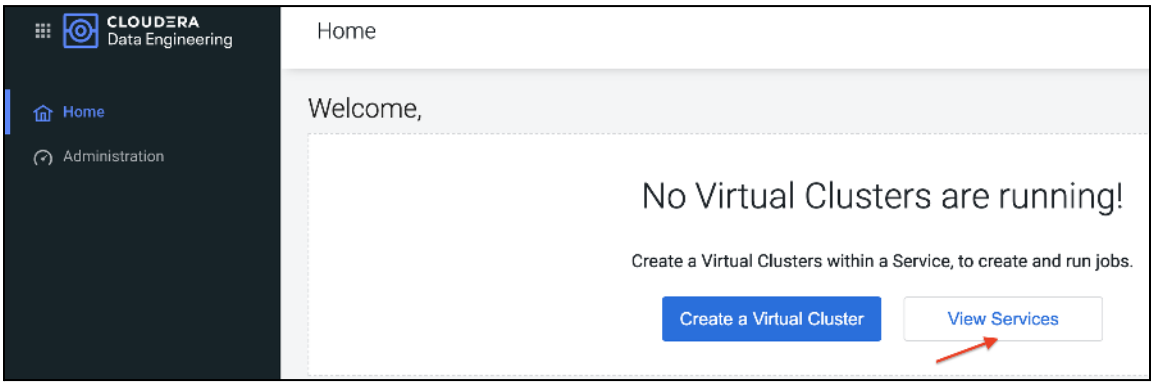


(default CDE is the name given as an example. You will see as per the value you entered in the previous step.)

- 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

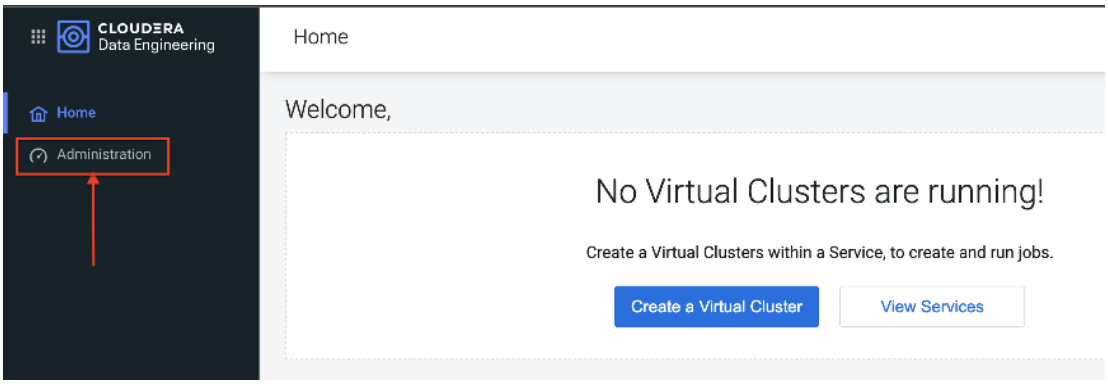
If you are unable to see the service, then the chances are that the default virtual cluster would not have been created properly. In this case, click on the ***View Services*** button and then you will be able to see the ***CDE*** service ***enabled***.



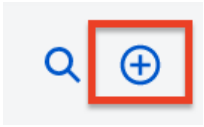
Enabling CDE Service:

- From the ***CDP console page***, click on ***Data Engineering***.

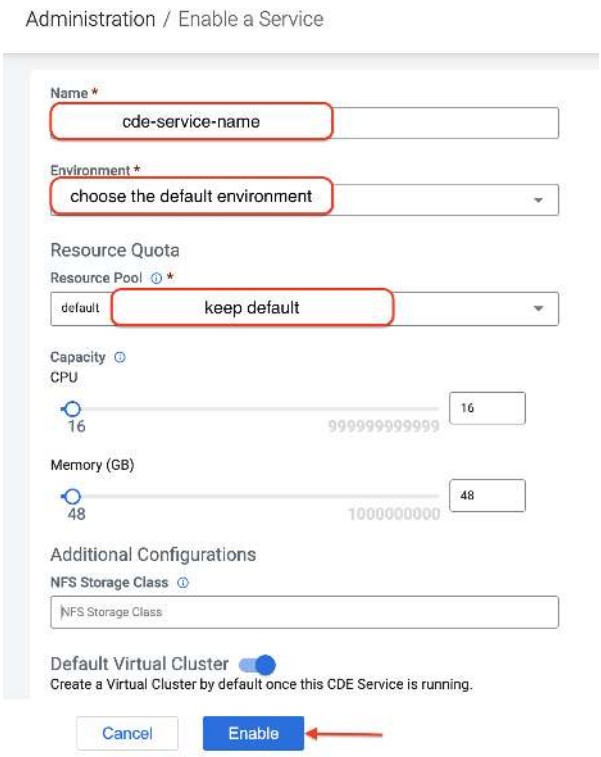
- This will open the ***CDE home page***. Since this will be the first time you will be opening CDE, you will not see any virtual clusters. Click on ***Administration*** in the left pane.



- Click on **+** icon as shown below which will allow you to ***enable CDE service*** post which you can ***create Virtual Clusters***.

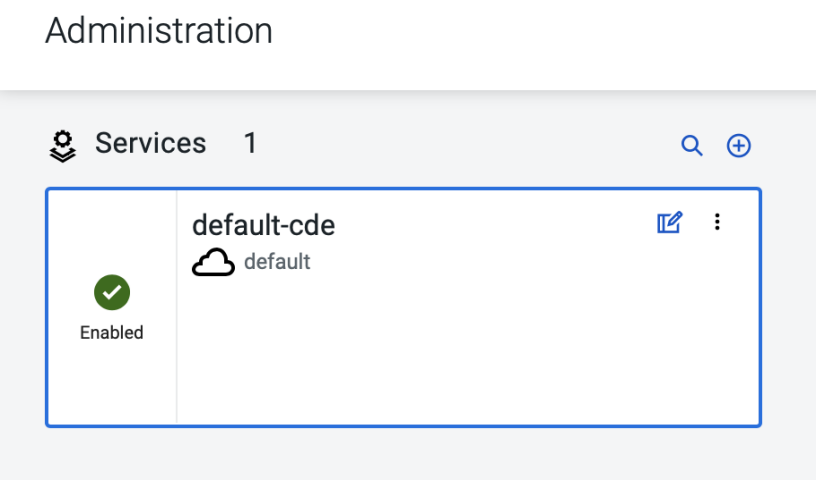


- On the ***Enable a Service*** page, enter the ***values*** as shown below and then click on ***Enable***.



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.

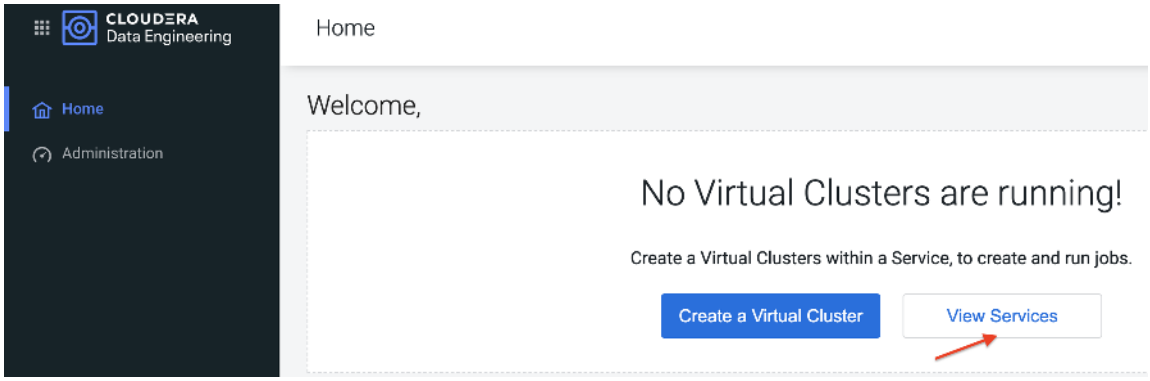


(*default **CDE** is the name given as an example. You will see as per the value you entered in the previous step*.)

- 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.

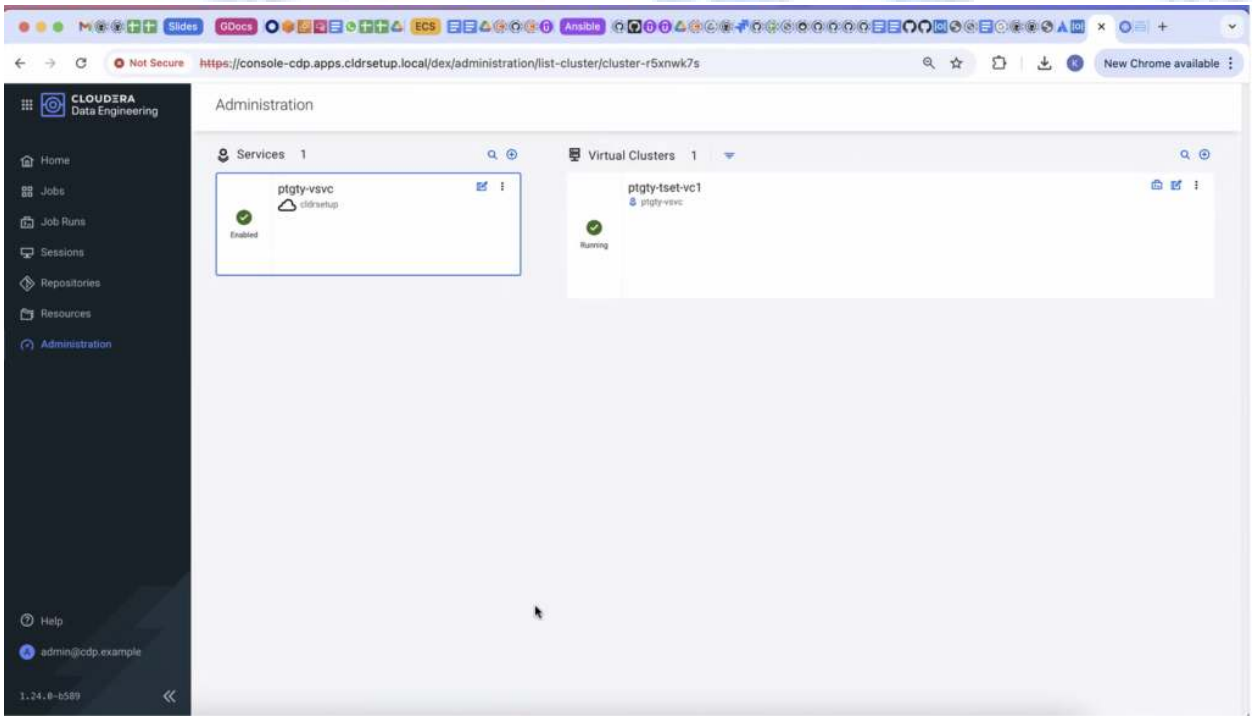
- On the ***CDE Home page***, click on the **+** icon next to ***Virtual clusters*** as shown below.

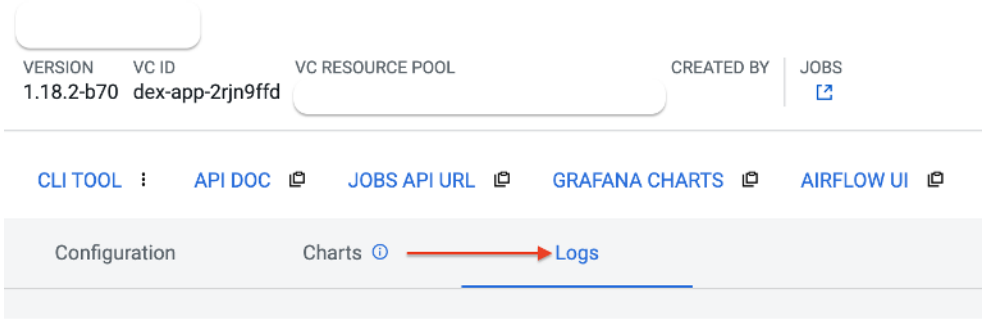
- On the ***Create a Virtual Cluster*** page, enter the below ***values*** and click on ***Create***.
  - ***Cluster Name:*** Cluster Name should adhere to the below conditions.

    - 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 ***OCP 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@pvcocp-master ~]# mkdir -p /tmp/cde-latest && cd /tmp/cde-latest
```

- Download the script cdp-cde-utils using wget.

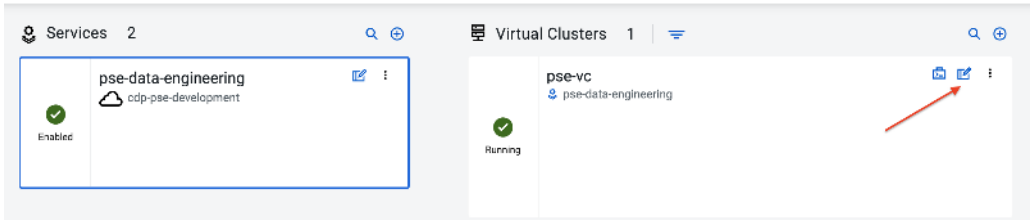
```
[root@pvcocp-master ~]# wget https://docs.cloudera.com/data-engineering/1.5.5/cdp-cde-utils.sh
```

- Add execute permission to this script.

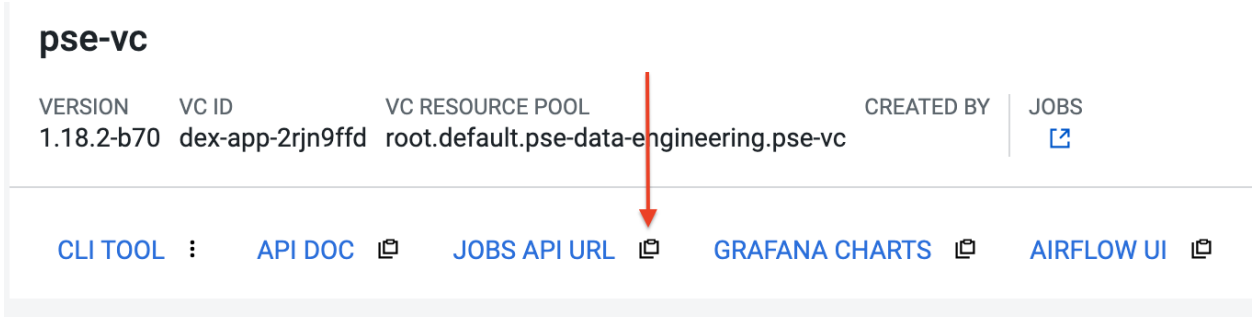
```
[root@pvcocp-master ~]# chmod +x /tmp/cde-latest/cdp-cde-utils.sh
```

- 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.ocp-demo.example.com/dex/api/v1
```

Then the endpoint will then be as shown below.

```
dfdj6kgx.cde-2cdxw5x5.ocp-demo.example.com
```

- Once you get the endpoint of the virtual cluster, ***login to the OCP master*** and navigate to ***/tmp/cde-latest*** directory where the ***cdp-cde-utils.sh*** script is present.

```
[root@pvcocp-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@pvcocp-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@pvcocp-master ~]# ./cdp-cde-utils.sh init-virtual-cluster -h
dfdj6kgx.cde-2cdxw5x5.ocp-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 OCP master host and navigate to the directory /tmp/cde-latest.

```
[root@pvcocp-master ~]# cd /tmp/cde-latest
```

- Install krb5-workstation package using dnf.

```
[root@pvcocp-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 ***redhat.local*** is the realm provided during IPA setup. You need to replace it with the realm you configured.

```
[root@pvcocp-master ~]# cat>> admin.principal
cdpuser@redhat.local
```

- Generate a keytab named ***<username>.keytab*** for the user using ***ktutil***:

```
[root@pvcocp-master ~]# cat>> admin.keytab
[root@pvcocp-master ~]# sudo ktutil
ktutil:  addent -password -p admin@redhat.local -k 1 -e aes256-cts
Password for admin@redhat.local:
ktutil:  addent -password -p admin@redhat.local -k 2 -e aes128-cts
Password for admin@redhat.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@pvcocp-master ~]# klist -ekt admin.keytab
```

- Validate the ***keytab*** using ***kinit***. This command should get executed successfully.

```
[root@pvcocp-master ~]# kinit -kt admin.keytab admin@redhat.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@pvcocp-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@pvcocp-master ~]# ./cdp-cde-utils.sh init-user-in-virtual-cluster -h
dfdj6kgx.cde-2cdxw5x5.ocp-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@REDHAT.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@redhat.local
  Principal alias: kdsahu@redhat.local
  User password expiration: 20240321113054Z
  Email address: kdsahu@redhat.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@redhat.local
ipa-getkeytab \
  --principal=cdptest@redhat.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.redhat.local

# Pull its keytab
ipa-getkeytab \
  --principal=cdptest/your-host.redhat.local@redhat.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@redhat.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@redhat.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.pvcocp-master.redhat.local.
ipa dnsrecord-del 16.172.in-addr.arpa. 231.31 --ptr-rec console-cdp.apps.pvcocp-master.redhat.local
ipa dnsrecord-find 16.172.in-addr.arpa.
ipa dnsrecord-add redhat.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@redhat.local:
Password expired.  You must change it now.
Enter new password:
```

```
Enter it again:
[kuldeep@pvcbasemaster ~]$
```

```
[root@ipaserver ~]# ldapsearch  -H ldap://ipaserver.redhat.local:389 -D
"uid=admin,cn=users,cn=accounts,dc=cldrsetup,dc=local" -w 'redhat123' -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@redhat.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@pvcocp-master ~]# ipa dnsrecord-find 16.172.in-addr.arpa.
  Record name: @
  NS record: ipaserver.redhat.local.
  Record name: 226.31
  PTR record: ipaserver.redhat.local.
  Record name: 227.31
  PTR record: pvcbase-master.redhat.local.
  Record name: 228.31
  PTR record: pvcbase-worker1.redhat.local.
  Record name: 231.31
  PTR record: pvcocp-master.redhat.local.
  Record name: 232.31
  PTR record: pvcocp-worker1.redhat.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.redhat.local:8020 , LocalHost:localPort
pvcbasemaster.redhat.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@redhat.local

Valid starting       Expires              Service principal
05/15/2024 21:24:12  05/16/2024 20:35:44  krbtgt/redhat.local@redhat.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.redhat.local@redhat.local
   2 05/19/2024 22:11:49 s3g/pvcbase-master.redhat.local@redhat.local

[root@pvcbase-master ~]# kinit -kt
/run/cloudera-scm-agent/process/1546347521-ozone-S3_GATEWAY/ozone.keytab
s3g/pvcbase-master.redhat.local@redhat.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.redhat.local@redhat.local
   2 05/19/2024 22:11:49 s3g/pvcbase-master.redhat.local@redhat.local

[root@pvcbase-master ~]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: s3g/pvcbase-master.redhat.local@redhat.local

Valid starting       Expires              Service principal
05/26/2024 07:36:13  05/27/2024 07:11:26  krbtgt/redhat.local@redhat.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/16/data/
mv -v /var/lib/pgsql/16/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
```

****

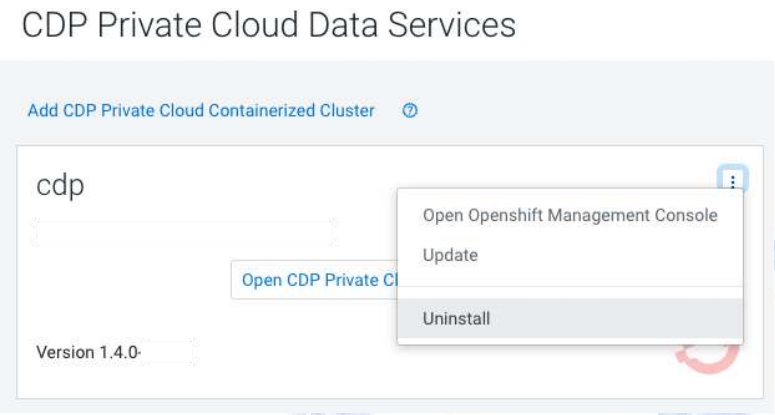
Cleanup CDP PvC Data Services-OCP Cluster:

https://docs.cloudera.com/cdp-private-cloud-data-services/1.5.5/installation/topics/cdppvc-installation-uninstall-pvc.html

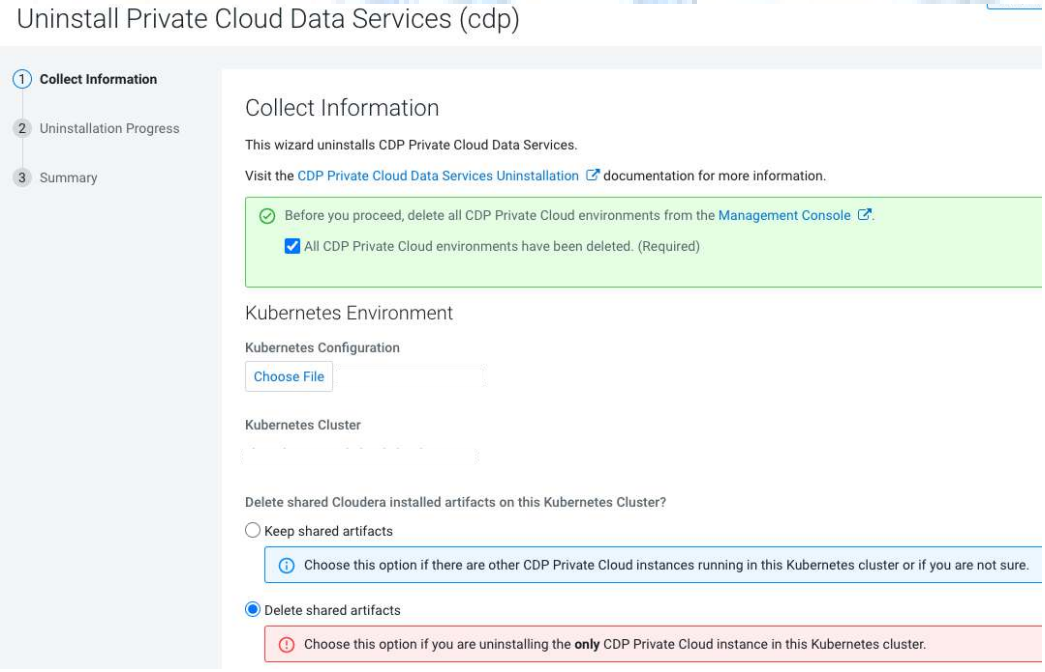
CLEANUP OCP

You can uninstall ***CDP Private Cloud Data Services*** from your ***CDP Private Cloud Base- Cloudera Manager***. Before you uninstall ***CDP Private Cloud Data Services***, ensure that you have deleted all the ***CDP Private Cloud environments*** registered in your ***CDP Private Cloud Data Services***. You can delete your registered environments using ***Management Console***.

1. In ***Cloudera Manager***, navigate to ***CDP Private Cloud Data Services*** and click ⋮ . Click ***Uninstall***.

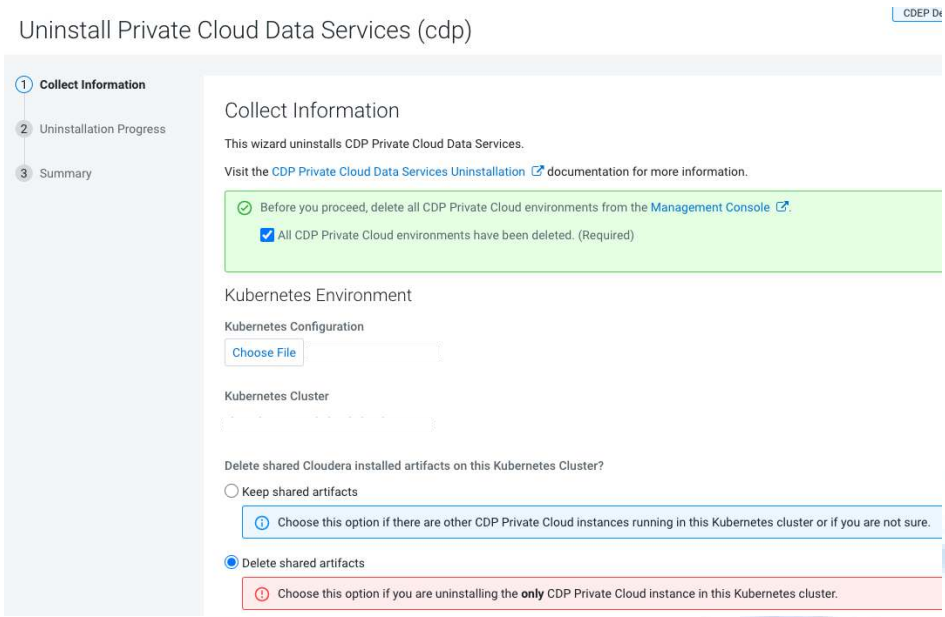


2. The ***Collect Information*** page appears. You must select the ***checkbox*** associated with your CDP Private Cloud Environments. Click ***Choose File*** to upload your ***kubeconfig*** file associated with your Kubernetes cluster.

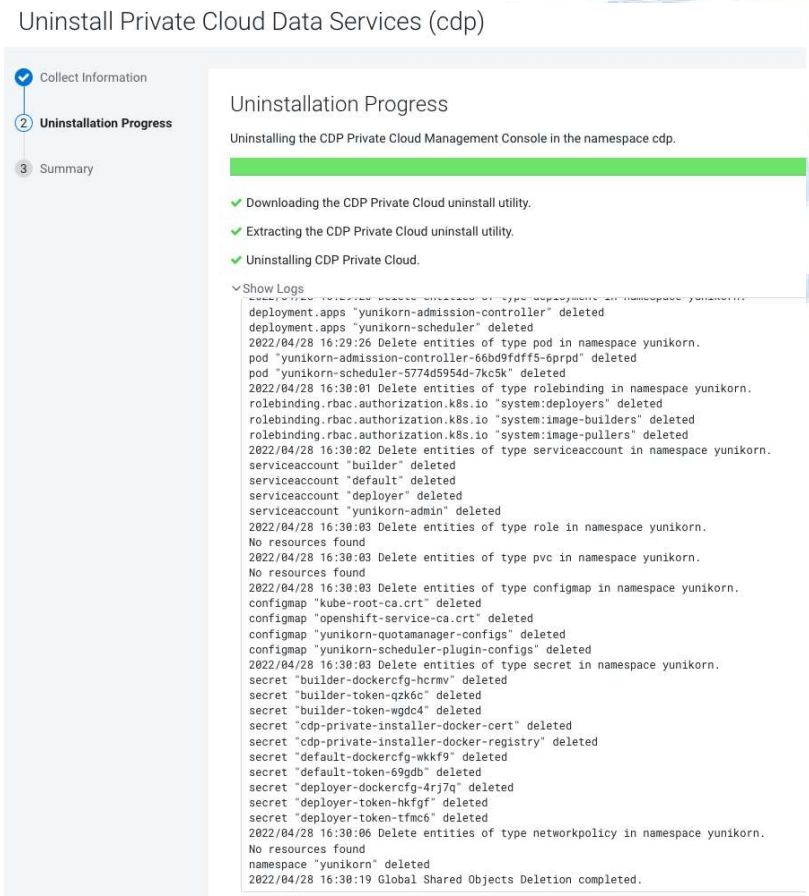


3. Select ***Keep shared artifacts*** if you have other ***CDP Private Cloud Data Services instances*** running in your Kubernetes cluster, or select ***Delete shared artifacts*** to remove any cluster global security policies or objects

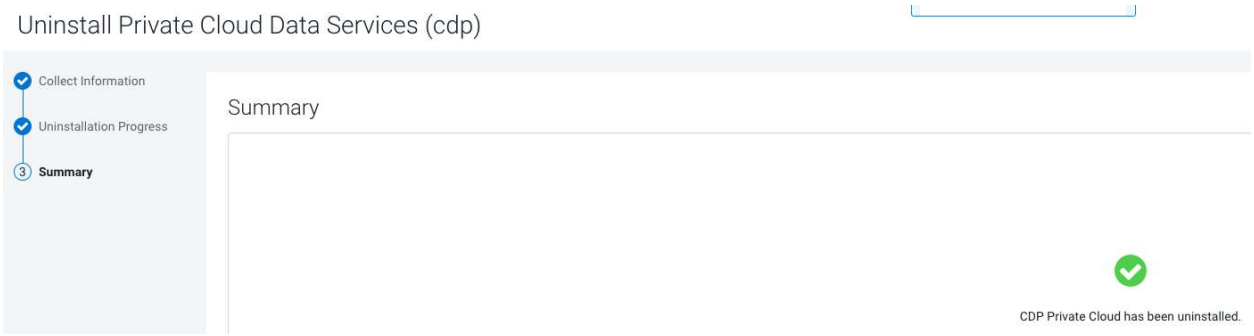
associated with this Kubernetes namespace.



4. Click ***Continue*** to complete the process.



5. You will now see that CDP Private Cloud has been uninstalled.



****

```
alternatives --set python /usr/bin/python2

openssl s_client -connect cldr-mngr.redhat.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.redhat.local
```

```
pvcocp[1-5].redhat.local; pvcocp-master.redhat.local: IOException thrown while collecting data from
host: Received fatal alert: internal_error
```

Solution:

```
openssl s_client -connect pvcbasemaster.redhat.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.redhat.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.redhat.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.redhat.local@redhat.local
[root@pvcbase-master ~]# /opt/cloudera/parcels/CDH/bin/zookeeper-client
```

Chart not showing--> install mgmt service.

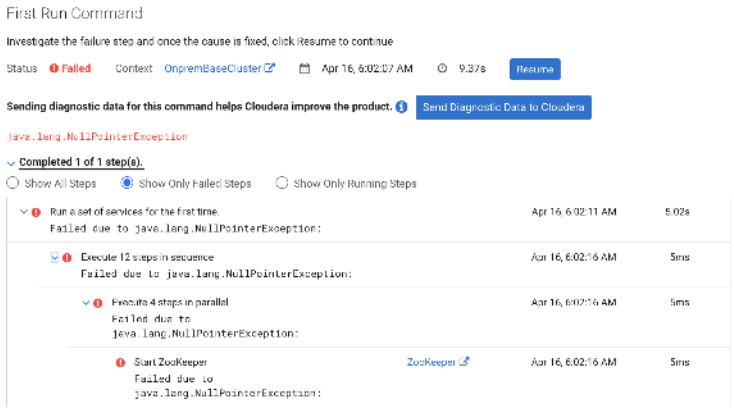
```
logfile=/var/log/cloudera-scm-agent/supervisord.log
```

****

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

****

Kubernetes Command Reference:

```
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/OCP-1.5.5-b10-ocp-1.5.5-b10.p0.50802651/kubernetes-dashboard/kubernetes-dashb
oard-5.10.0.tgz -n kubernetes-dashboard  -f
/opt/cloudera/parcels/OCP-1.5.5-b10-ocp-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.redhat.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.

****