Introduction

This document contains instructions on onboarding a CSS(Cloudera Semantic Search) cluster on the Public Cloud V2 Environment (K8s On EKS).

Prerequisites

Helm (v 3.14.x)	Installing Helm
Kubectl	Install Tools | Kubernetes (Refer to this if you are setting it up the first time)
curl	Linux curl commands
jq	You need to install jq in your command line. brew install jq For more info https://jqlang.github.io/jq/
Python3	Preferred Python 3.11.4
Virtualenv	https://virtualenv.pypa.io/en/latest/ Sample Command Unset `pip install virtualenv`
Access to CDP AWS environment	Should be able to log into the CDP console and get Environment CRN and actor CRN information. (Refer to this for more details)
AWS access	Sufficient privileges to create an EKS cluster (Refer to this for more details)
LIFTIE entitlements	LIFTIE_AWS_AZ_REBALANCE_DISABLED COMPUTE_API_LIFTIE_DEPLOYMENT LIFTIE_USE_ARM64_INSTANCES COMPUTE_API_LIFTIE COMPUTE_API_LIFTIE_INTERNAL COMPUTE_API_LIFTIE_BETA

Assumptions

1) You have access to the Docker private registry https://docker-private.infra.cloudera.com/
2) You already have a V2 environment in the respective mow environment. You can create the V2 environment both from UI or CDPCLI. For more information, please see How to Create Compute Enabled Environment. *Note: we should use --enable-compute-cluster aka V2 environment*
3) You need to have a working cdp on your computer. (Refer to this if you are setting it up the first time)

4) You have ~/.ssh and ~/.kube directory in your computer.

IMPORTANT: It's a requirement from the Security team that you must create a separate cluster with your own AWS account and not use the default cluster.

Deployment Steps

Refer to the following steps to deploy Cloudera Semantic Search in a Kubernetes Public Cloud environment. Here we will use scripts to create a liftie cluster and deploy the CSS nodes. With these steps you will get 3 master nodes, 1 data node, 1 ml node, 1 ingest node, 1 dashboard node. And the coordinator node will get deployed along with the data node
Note: you need to have 'internal cdp' and 'jq' installed in your command prompt

Step 1: Pull and unpack the charts

A. Pull the charts

You need to use 0.1.0-b28 version or more while pulling the charts, here I am using the charts from *docker-private.infra.cloudera.com* repo, this may change in prod. Note: Always advice to create a directory and pull the charts in it.

Unset
```
mkdir css-helm-charts
cd css-helm-charts
```

Example command to pull a chart, you need to pull 2 charts here `solr/opensearch, solr/opensearch-dashboards`

Unset
```
# Pull the charts
helm pull oci://docker-private.infra.cloudera.com/cloudera-helm/solr/opensearch --version
0.1.0-b28
helm pull
oci://docker-private.infra.cloudera.com/cloudera-helm/solr/opensearch-dashboards
--version 0.1.0-b28
```

B. Unpack helm charts bundle

Below is the sample command. You need to unpack all the charts, example as shown below

Unset
```
for file in *.tgz; do tar -vxf "$file"; done
```

After this, your css-helm-charts will have 2 more directories

Unset
```
opensearch                    opensearch-dashboards
```

Note: You may need to execute these steps in prod based on the registry you are using

Step 2: Install CDPCLI-INTERNAL

Run below commands to install cdpcli-internal commands, please run this in the same directory css-helm-charts. For more information you can follow this WIKI

```
Unset
virtualenv -p python3 cdpcli
. cdpcli/bin/activate
pip3 install --extra-index-url
https://pypi.infra.cloudera.com/api/pypi/pypi-cloudera/simple cdpcli-internal
```

Step 3: Export the environment variables to create a liftie cluster.

```
Unset
export ENVIRONMENT_CRN="<Environment_CRN>"
export CLUSTER_NAME="<cluster_name>"
export AUTHORIZED_IP_RANGES="Optional param, in prod you may need this"
export CDP_PROFILE="Set it if you are using different profile for local cdp to run"
```

Note:
1) you need to replace the `<Environment_CRN>` with the actual value of the environment crn
2) you also need to replace `<cluster_name>` with the name you want to create your liftie cluster.
3) AUTHORIZED_IP_RANGES value should be the CIDR that your workstation/laptop is using, e.g. a.b.c.d/24, add it to part of the `Kubernetes Server Authorized IP Addresses

Step 4: Run the scripts

A. Create liftie cluster

To run the scripts you first need to create a folder like “css-deployment” and copy the necessary scripts [*createLiftieCluster.sh, deployCSSHelmCharts.sh, deleteInstallation.sh*] inside the folder. The scripts are available at this location. Next you need to go inside the folder and run the script. You also need to give execute privileges to these scripts, here I gave 777 privilege by running *chmod 777 *.sh*

```
Unset
mkdir css-deployment
cd css-deployment
./createLiftieCluster.sh
```

B. Deploy CSS cluster

To deploy the css cluster we need to set the below environment variables followed by running the *deployCSSHelmCharts.sh* script. Export the necessary environment variables.

```
Unset
export CLUSTER_CRN="<cluster_crn>"
export HELM_CHARTS_DIRECTORY="<helm_chart_directory"
```

Note:

1. *CLUSTER_CRN value should be the value displayed in the output of the createLiftieCluster script*
2. *HELM_CHARTS_DIRECTORY value should be the complete path of the helm charts pull directory. As per the steps followed it should be the full path till this directory css-helm-charts*
3. *You may need to set CDP_PROFILE if your cdp command relies on that.*

Now you can run the deploy script

```
Unset
./deployCSSHelmCharts.sh
```

After this you will be having the kubeconfig file in the following path shown in the command output. Example ~/.kube/script-config. You need to export this file to kubeconfig. *export KUBECONFIG=<kube_config_path>*
Note: After this deployment you may need wait for few minutes to all the pods to be in ready state

C. Access Opensearch dashboard

To be able to connect to the dashboards service hosted on the pod, you need to access this url in your browser. http://<DASHBOARD_EXTERNAL_IP>:5601
Note: Follow this section on how to find DASHBOARD_EXTERNAL_IP

Validation Steps For CSS

Please check this section on how to validate the CSS cluster deployments.

Validation Steps For Dashboard

Please check this section on how to validate the CSS Dashboard.

Uninstall steps

Refer to the following steps to delete a Cloudera Semantic Search cluster from your public cloud environment.

Step 1: Verify the Environment variables

Here we need to verify whether we have seeing correct value to the CLUSTER_CRN variable, if not then set it to the actual value of the cluster crn which needs to be deleted, you need to do the same for CDP_PROFILE, KUBECONFIG as well.

```
Unset
echo $CLUSTER_CRN
echo $KUBECONFIG
echo $CDP_PROFILE
```

Step 2: Run the delete script

```
Unset
./deleteInstallation.sh
```

Additional Steps to use prod registry

A. Modify the helm charts to use Cloudera's public repository.

Change the values.yaml for opensearch:

- For image field, Replace docker-private.infra.cloudera.com with container.repository.cloudera.com
- For imagePullSecrets field, Replace [] with [{"name": "jfrog-dev"}]

```
Unset
vi opensearch/values.yaml

# replace docker-private.infra.cloudera.com with container.repository.cloudera.com
image:
  repository: "container.repository.cloudera.com/cloudera/opensearch"
  tag: "" ## Use Release version from Chart
  pullPolicy: "IfNotPresent"

# also replace [] with [{"name": "jfrog-dev"}]
imagePullSecrets: [{"name": "jfrog-dev"}]
```

Change the values.yaml for opensearch-dashboards:

- For image field, Replace docker-private.infra.cloudera.com with container.repository.cloudera.com
- For imagePullSecrets field, Replace [] with [{"name": "jfrog-dev"}]

```
Unset
vi opensearch-dashboards/values.yaml

# replace docker-private.infra.cloudera.com with container.repository.cloudera.com
image:
  repository: "container.repository.cloudera.com/cloudera/opensearch-dashboards"
  tag: "" ## Use Release version from Chart
  pullPolicy: "IfNotPresent"

# also replace [] with [{"name": "jfrog-dev"}]
imagePullSecrets: [{"name": "jfrog-dev"}]
```

Additional Links

1. Runbook to create ingest pipeline for neural search
2. For RAG Demo script you can follow this link RAG_NS_Demo