



Deploying Cloudera Semantic Search (CSS) in K8s (EKS) in V2 Environment

Version 0.1

Document Prepared by: Abhradeep Kundu

Revision History			
Version	Author(s)	Description	Date
0.1	Abhradeep Kundu	Initial version	2024/11/13

Important Notice

© 2010-2024 Cloudera, Inc. All rights reserved.

Cloudera, the Cloudera logo, and any other product or service names or slogans contained in this document, except as otherwise disclaimed, are trademarks of Cloudera and its suppliers or licensors, and may not be copied, imitated or used, in whole or in part, without the prior written permission of Cloudera or the applicable trademark holder.

Hadoop and the Hadoop elephant logo are trademarks of the Apache Software Foundation. All other trademarks, registered trademarks, product names and company names or logos mentioned in this document are the property of their respective owners to any products, services, processes or other information, by trade name, trademark, manufacturer, supplier or otherwise does not constitute or imply endorsement, sponsorship or recommendation thereof by us.

Complying with all applicable copyright Azure is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Cloudera.

Cloudera may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Cloudera, the furnishing of this document does not give you any license to these patents, trademarks copyrights, or other intellectual property.

The information in this document is subject to change without notice. Cloudera shall not be liable for any damages resulting from technical errors or omissions which may be present in this document, or from use of this document.

Cloudera, Inc.

Santa Clara, California
5470 Great America Pkwy
Santa Clara, CA 95054

US: +1 888 789 1488

Outside the US: +1 650 362 0488

<http://www.cloudera.com>

Disclaimer

This document is only for internal use purposes and should not be provided to customers.

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
CDP CLI (Internal)	<p>You need to use the internal CDP script in this doc</p> <div><p>Unset</p><pre>pip install cdpcli-internal --pre --index-url https://pypi.infra.cloudera.com/api/pypi/cloudera/simple</pre></div> <p>(Refer to this if you are setting it up the first time) You can also follow this WIKI</p>
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*

NOTE: 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.

Step 1: Create Cluster and instance group using Liftie API

A. Create Cluster

Use the following template and replace the appropriate values to create a cluster

Unset

```
cdp compute create-cluster --environment <ENV_NAME> --name <CLUSTER_NAME>
```

B. Prepare the JSON for the instance group

Use the following template and replace the appropriate values.

Unset

```
cdp compute create-instance-groups --cli-input-json '{
  "clusterCrn": "<cluster-crn>",
  "instanceGroups": [
    {
      "name": "css-master",
      "instanceCount": 3,
      "autoscaling": {
        "maxInstance": 3,
        "minInstance": 3,
        "enabled": true
      }
    }
  ]
}
```

```

    },
    "instanceTier": "ON_DEMAND",
    "labels": {
        "project": "css",
        "css-node-group": "master"
    },
    "rootVolume": {
        "size": 50
    },
    "instanceTypes": [
        "m5.xlarge"
    ]
},
{
    "name": "css-data",
    "instanceCount": 1,
    "autoscaling": {
        "maxInstance": 1,
        "minInstance": 1,
        "enabled": true
    },
    "instanceTier": "ON_DEMAND",
    "labels": {
        "project": "css",
        "css-node-group": "data"
    },
    "rootVolume": {
        "size": 50
    },
    "instanceTypes": [
        "m5.xlarge"
    ]
},
{
    "name": "css-ml",
    "instanceCount": 1,
    "autoscaling": {
        "maxInstance": 1,
        "minInstance": 1,
        "enabled": true
    },
    "instanceTier": "ON_DEMAND",
    "labels": {
        "project": "css",

```

```

        "css-node-group": "m1"
      },
      "rootVolume": {
        "size": 50
      },
      "instanceTypes": [
        "m5.xlarge"
      ]
    },
    {
      "name": "css-ingest",
      "instanceCount": 1,
      "autoscaling": {
        "maxInstance": 1,
        "minInstance": 1,
        "enabled": true
      },
      "instanceTier": "ON_DEMAND",
      "labels": {
        "project": "css",
        "css-node-group": "ingest"
      },
      "rootVolume": {
        "size": 50
      },
      "instanceTypes": [
        "m5.xlarge"
      ]
    }
  ],
  "clusterStateVersion": 0,
  "skipValidation": false
}'

```

Here replace the `<cluster-crn>` value with your cluster
 CLUSTER_CRN can be found by invoking the following command:

```

Unset
cdp compute list-clusters --env-name-or-crn <ENV_NAME>

```

C. Note down the cluster CRN

The create-instance-groups command should return a response as shown below

```
Unset
{
  "clusterCrn":
  "crn:cdp:compute:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:cluster:liftie-7j
rxbm0r",
  "clusterStatus": {
    "message": "Creating instance groups",
    "status": "UPDATING"
  },
  "uri": "/liftie/api/v1/cluster/liftie-7jrxbm0r"
}
```

Note down the cluster crn from the above payload. In subsequent sections, we will use it in place of <CLUSTER_CRN>.

Check the cluster creation status

```
Unset
cdp compute describe-cluster --cluster-crn <CLUSTER_CRN> | grep "status"

e.g. cdp compute describe-cluster --cluster-crn
"crn:cdp:compute:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:cluster:liftie-5k
w5qbb5" | grep "status"
```

Check the “instanceGroupStatus” for the instance group you just created. It should be “CREATE_COMPLETE” when everything is done. It might take around 2-3 minutes.

D. Add your CIDR to the cluster for accessing the k8s cluster

Find your own 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’

```
Unset
cdp compute update-cluster --cluster-state-version <clusterStateVersion found via
cdp compute describe-cluster> --cli-input-json '{
  "clusterCrn": "CLUSTER_CRN",
  "spec": {
    "security": {
      "apiServer": {
        "authorizedIpRanges": [
          "a.b.c.d/24"
        ]
      }
    }
  }
}
```

```

        ],
        "enabled": true
      },
      "secretEncryption": {
        "customerKmsKeyArn": "customerKmsKeyArn found via cdp
compute describe-cluster",
        "enabled": true
      },
      "volumeEncryption": {},
      "private": false
    }
  }
}'

```

E. Generate Kubeconfig

Dump the kubeconfig in ~/.kube/ops-cluster-configs file.

```

Unset
cdp compute get-kube-config --cluster-crn <CLUSTER_CRN> --service computex
--namespace computex | jq -r '.content' | sed 's/\\n/\\n/g' >
~/.kube/ops-cluster-configs

```

Here replace the <CLUSTER_CRN> value with your cluster.

F. Export the Kubeconfig to set the context

```

Unset
export KUBECONFIG=~/.kube/ops-cluster-configs

```

G. Test using Kubernetes command

```

Unset
kubectl get pods -A

```

It should show all the infra pods created by the liftie cluster.

Step 2: Deploy Helm charts

A. Pull the charts

We need to get this 0.1.0-b28 version while pulling the charts

```
Unset
# pull the helm charts
mkdir css-helm-charts
cd css-helm-charts

# 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
```

Unpack helm charts bundle

```
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

B. Execute Helm commands

Install Master

```
Unset
helm install opensearch-master opensearch -f opensearch/values.yaml -f
opensearch/master.yaml --set adminPassword=Cloudera@Test4321 --namespace css
--create-namespace
```

For the TINY T-shirt size, we do not need a separate coordinator role. The data node acts as data+coordinator. So we have set "coordinatorService=data". If you want a dedicated coordinator node (and do not want to use a data node as a coordinator), remove this flag while installing data nodes.

Install Data Node with dual roles data + coordinator:

Unset

```
helm install opensearch-data opensearch -f opensearch/values.yaml -f  
opensearch/data.yaml --set adminPassword=Cloudera@Test4321 --set  
coordinatorService=data --namespace css --create-namespace
```

Optional: Install the Data Node separately. If you want a dedicated data role

Unset

```
helm install opensearch-data opensearch -f opensearch/values.yaml -f  
opensearch/data.yaml --set adminPassword=Cloudera@Test4321 --namespace css  
--create-namespace
```

Optional: Install Coordinator Nodes. If you want a dedicated coordinator role

Note: This is not needed if coordinatorService=data is set while deploying opensearch-data

Unset

```
helm install opensearch-coordinator opensearch -f opensearch/values.yaml -f  
opensearch/coordinator.yaml --set adminPassword=Cloudera@Test4321 --namespace css  
--create-namespace
```

Install ML Node

Unset

```
helm install opensearch-ml opensearch -f opensearch/values.yaml -f  
opensearch/ml.yaml --set adminPassword=Cloudera@Test4321 --namespace css  
--create-namespace
```

Install Ingest Node

Unset

```
helm install opensearch-ingest opensearch -f opensearch/values.yaml -f  
opensearch/ingest.yaml --set adminPassword=Cloudera@Test4321 --namespace css  
--create-namespace
```

Install Dashboards

Follow these steps in the doc [Dashboards demo with helm charts](#) to install Dashboards

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.

Validation Steps for ML Node and Ingest Node

Please follow [this section](#) for ML Node and Ingest Node

Uninstall steps

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

Step 1: Uninstall the helm charts

```
Unset
helm uninstall opensearch-master -n css
helm uninstall opensearch-data -n css
```

Step 3: Delete the Instance Groups.

```
Unset

# Master instance group
# Getting the cluster_version
cdp compute describe-cluster --cluster-crn $CLUSTER_CRN | jq -r
'.clusterStateVersion'
# Delete the instance group
cdp compute delete-instance-group --cluster-crn <cluster-crn>
--instance-group-name css-master --cluster-state-version <CLUSTER_VERSION>

# Data instance group
# Getting the cluster_version
cdp compute describe-cluster --cluster-crn $CLUSTER_CRN | jq -r
'.clusterStateVersion'
# Delete the instance group
cdp compute delete-instance-group --cluster-crn <cluster-crn>
--instance-group-name css-data --cluster-state-version <CLUSTER_VERSION>

# ML instance group
# Getting the cluster_version
cdp compute describe-cluster --cluster-crn $CLUSTER_CRN | jq -r
'.clusterStateVersion'
```

```
# Delete the instance group
cdp compute delete-instance-group --cluster-crn <cluster-crn>
--instance-group-name css-m1 --cluster-state-version <CLUSTER_VERSION>

# Ingest instance group
# Getting the cluster_version
cdp compute describe-cluster --cluster-crn $CLUSTER_CRN | jq -r
'.clusterStateVersion'
# Delete the instance group
cdp compute delete-instance-group --cluster-crn <cluster-crn>
--instance-group-name css-ingest --cluster-state-version <CLUSTER_VERSION>
```

Step 2: Delete the cluster.

```
Unset
cdp compute delete-cluster --cluster-crn <cluster-crn>
```

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"}]
```

Appendix

1. You can get the environment CRN by logging into one of your environments. For example,

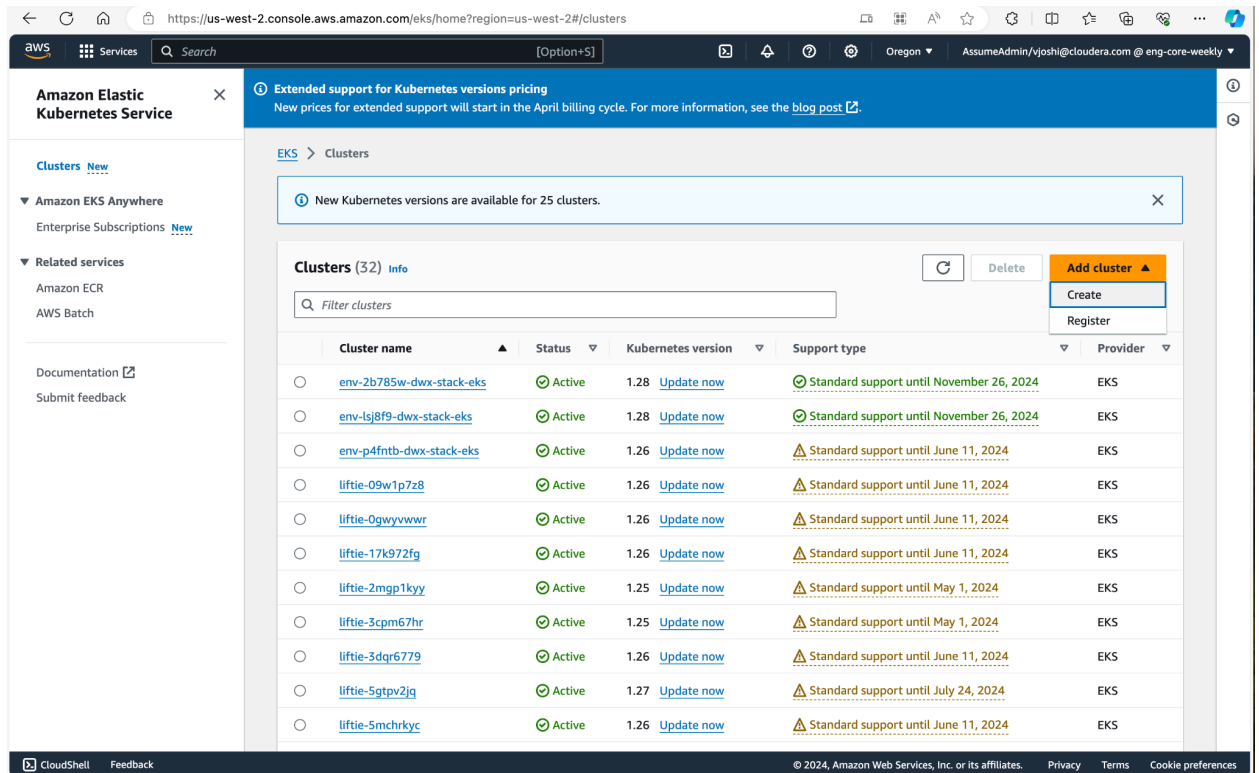
<https://console.dps.mow-dev.cloudera.com/cloud/environments/details/css-7218-micro-v2-2>

2. You can get the actor CRN and the tenant ID by logging into the CDP Management Console and going into your user's profile. For example,

<https://cloudera.dps.mow-dev.cloudera.com/iam/index.html#/my-account>

3. How can I check if I have sufficient privileges to create an EKS cluster?

Ans: Login to AWS console -> Search EKS in the search bar->Click on EKS link ->Click on Add Cluster. You should see the option "Create" as shown in the screenshot.





4. How to configure CDP client environment for the first time


Normally, when a `cdpcli` is used to talk to a CDP instance (e.g. `manowar-dev`), the REST gateway will add the appropriate HTTP headers that contain your authenticated identity. Specifically, this is the `x-cdp-actor-crn` and `x-cdp-request-id` headers. When we run our services locally, we bypass the REST gateway, which means these headers are not automatically set. To work around this, we must configure the `cdpcli` to automatically set this header talking to your local services.

- Get access rights to the CDP instance you are trying to connect to. For example to get access right to `manowar-dev` (note: SSO sign-in might fail, try in incognito mode!). Then access your profile, and generate an access key. You can generate an access key and private key by visiting the [CDP's mow-dev console](#).

Users / Josh Elser

Name	Josh Elser
Email	jelser@cloudera.com
Workload User Name ⓘ	csso_jelser
CRN ⓘ	crn:altus:iam:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:user:c00c2cae-ac1a-4717-b104-30294ea2b27c 
Identity Provider ⓘ	cloudera-sso
Last Interactive Login	08/27/2020 3:11 PM EDT
Profile Management	View profile 
Workload Password ⓘ	Set Workload Password

[Access Keys](#) Roles Resources Groups SSH Keys

[Generate Access Key](#) 

This will generate a new access key and private key. Ensure to copy the private key as CDP will only show you this key once. You can either download the credentials file or run the `cdp configure` command and enter the access key ID and private key manually:

Unset
`cdp configure`

b. Validate that your `~/.cdp/credentials` file looks like the following:

Unset
[default]
`cdp_access_key_id = 01234567-0123-0123-012345678901`
`cdp_private_key = TheGeneratedKeyFromCDP`

c. Ensure to configure(`~/.cdp/config`) the CDP instance to connect to, in manowar case:

```
Unset
[profile default]
endpoint_url = https://%sapi.thunderhead-dev.cloudera.com/
cdp_endpoint_url = https://api.dps.mow-dev.cloudera.com/
```

Aside: the above uses the profile name `default`. This name is the only “special” name. You can use any profile name as long as you are consistent between the `credentials` file and the `config` file. `default` is special only in that it is what is used when no `--profile <name>` is provided to `cdp`.

- d. Validate that you can communicate with CDP in manowar-dev:

```
Unset
cdp iam get-user
```

If this command returns a JSON message with your name, congratulations, you have a functioning `cdpcli` against `manowar-dev`.

- e. To configure your `cdpcli` to talk to your pseudo-local COD deployment, rerun the previous command can extract the `crn` field (you can use `jq` to help if you want).

```
Unset
cdp iam get-user | jq -r '.user.crn'
```

- f. Add a new profile to `~/.cdp/config` called “local”, using the `crn` you extracted in the previous step.

```
Unset
[profile local]
form_factor = public
request_headers = x-cdp-actor-crn=<YOUR-CRN-GOES-HERE>,x-cdp-request-id=generate::uuid
endpoint_url = https://%sapi.thunderhead-dev.cloudera.com/
cdp_endpoint_url = https://api.dps.mow-dev.cloudera.com/
```

This creates a `cdpcli` profile called “local” which injects the necessary headers and configures endpoints to still talk to `manowar-dev`. Finally, add the same local profile to your `.cdp/credentials` file. These credentials are the same as those you placed for the default profile.


```
Unset
[local]
cdp_access_key_id = 01234567-0123-0123-012345678901
cdp_private_key = TheGeneratedKeyFromCDP
```

Final cdpcli configuration files

Your ~/.cdp/config will look similar to this:

```
Unset
[profile default]
  endpoint_url = https://%sapi.thunderhead-dev.cloudera.com/
  cdp_endpoint_url = https://api.dps.mow-dev.cloudera.com/
[profile local]
  form_factor = public
  request_headers =
x-cdp-actor-crn=<YOUR-CRN-GOES-HERE>, x-cdp-request-id=generate::uuid
  endpoint_url = https://%sapi.thunderhead-dev.cloudera.com/
  cdp_endpoint_url = https://api.dps.mow-dev.cloudera.com/
```

And your ~/.cdp/credentials file will look similar to this:

```
Unset
[default]
cdp_access_key_id = 01234567-0123-0123-012345678901
cdp_private_key = TheGeneratedKeyFromCDP
[local]
cdp_access_key_id = 01234567-0123-0123-012345678901
cdp_private_key = TheGeneratedKeyFromCDP
```

5. How should I set up kubectl

kubectl is required to port forward other services running in manowar-dev to your local machine.

In case you have not already set “kube config”, you must follow the instructions listed in the [Cloud Services Infra setup](#). When you enter kubectl config get-contexts and do not see a similar result

(not an empty result) as below, please read this page [Troubleshooting Guide | Kubectl-Cluster-Access](#) and configure with update-kubeconfig.

```
Unset
% kubectl config get-contexts

CURRENT  NAME          CLUSTER                                     AUTHINFO
NAMESPACE
*         mow-dev        arn:aws:eks:us-west-2:392479084068:cluster/mow
arn:aws:eks:us-west-2:392479084068:cluster/mow

         mow-int     arn:aws:eks:us-west-2:706388717391:cluster/mow-int
arn:aws:eks:us-west-2:706388717391:cluster/mow-int

         mow-priv   arn:aws:eks:us-west-2:392479084068:cluster/mow-priv
arn:aws:eks:us-west-2:392479084068:cluster/mow-priv
```

6. Running the cdp cli command to create a liftie cluster gives the following error:

Unable to locate CDP credentials: No credentials found anywhere in chain. The shared credentials file should be stored at /Users/anand.srinivasan/.cdp/credentials

How do we create this credentials file?

Answer:

Generate access keys first from the user profile page.

<https://cloudera.atlassian.net/wiki/spaces/ENG/pages/160465619/Getting+Started+-+CDP+CLI>

7. Running cdp --profile gives the following error:

```
cdp --profile dev environments list-environments
/Users/anand.srinivasan/e/lib/python3.9/site-packages/urllib3/__init__.py:35:
NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module
is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
warnings.warn(
2024-03-11 17:36:03,421 - MainThread - cdpcli.clidriver - WARNING - You are running an
INTERNAL release of the CDP CLI, which has different capabilities from the standard
public release. Find the public release at: https://pypi.org/project/cdpcli/
An error occurred: 5 NOT_FOUND: Access key
'e42e0ebb-59f8-4aad-bd20-cf73a4499908' not found (Status Code: 401; Error Code: ;
Service: environments; Operation: listEnvironments; Request ID:
756293d1-f18d-429a-b6b2-c75a80861faa;)
```

Answer:

After editing ~/.cdp/config with the following contents, I'm able to run cdp --profile dev environments list-environments command successfully.

[profile dev]

cdp_endpoint_url = https://cloudera.dps.mow-dev.cloudera.com

endpoint_url = <https://%sapi.thunderhead-dev.cloudera.com>

8. Running cdp --profile gives the following validation error:

```
{
  "name": "AWS Subnet Tagging",
  "description": "In order for load balancers to choose subnets correctly a subnet
needs to have either the public or private ELB tags defined.",
  "category": "NETWORK",
  "status": "FAILED",
  "message": "The following subnets are missing tags to denote whether they
should be assigned a public or private ELB: subnet-0c65769c38fc79822.",
  "detailedMessage": "Each subnet in the VPC must be tagged with either
kubernetes.io/role/elb or kubernetes.io/role/internal-elb.",
  "duration": "172ms"
},
```

9. Running cdp create-cluster with --profile dev

```
cdp compute create-cluster --profile dev --cli-input-json '{"metadata": {"environmentCrn":
"crn:cdp:environments:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:environment:1
9ee19b2-85e2-4fcc-b9bc-c5f5bbbec71f", "name": "as-css-cluster", "clusterType":
"Dedicated", "description": "CSS cluster", "labels": {"owner": "anand.srinivasan",
"lifter-user-email": "anand.srinivasan@cloudera.com"}}, "clusterOwner": {"email":
"anand.srinivasan@cloudera.com", "firstName": "Anand", "lastName": "s", "userId":
"Srinivasan", "accountId": "9d74eee4-1cad-45d7-b645-7ccf9edbb73d", "crn":
"crn:altus:iam:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:user:65c40c34-2d5c-4
311-8ecb-13131ea258f5"}}, "spec": {"deployments": {"logging": {"enabled": false},
"monitoring": {"enabled": false}}, "instanceGroups": [{"name":
"css-cluster", "instanceCount": 5, "autoscaling": {"maxInstance": 5, "minInstance": 2,
"enabled": true}, "instanceTier": "SPOT", "labels": {"owner": "anand.srinivasan", "project":
"open-search"}, "rootVolume": {"size": 50}, "instanceTypes": ["m5.xlarge"]}],
"kubernetes": {"version": "1.26"}}, "skipValidation": false}'
/Users/anand.srinivasan/e/lib/python3.9/site-packages/urllib3/__init__.py:35:
NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module
is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
warnings.warn(
```

2024-03-11 17:45:41,084 - MainThread - cdpcli.clidriver - WARNING - You are running an INTERNAL release of the CDP CLI, which has different capabilities from the standard public release. Find the public release at: <https://pypi.org/project/cdpcli/>

```
{
  "clusterId": "liftie-hgy6h4jb",
  "clusterStatus": {
    "status": "CREATING"
  },
  "uri": "/liftie/api/v1/cluster/liftie-hgy6h4jb",
  "clusterCrn":
"crn:cdp:compute:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:cluster:liftie-hgy6h4jb"
}
```

Please note: If you have created your profile (user_defined), any cdp cli command should include `--profile <your_user_defined_profile_name>` otherwise the command will fail with an unrelated error.

10. Pods go into a pending state very soon.

```
kubectl describe pods opensearch-cluster-coordinator
```

Events:

Type	Reason	Age	From	Message
Warning	FailedScheduling	3m50s	default-scheduler	0/3 nodes are available: 1 node(s) were unschedulable, 2 node(s) had intolerated taint {role.node.kubernetes.io/liftie-infra: true}. preemption: 0/3 nodes are available: 3 Preemption is not helpful for scheduling..
Normal	TriggeredScaleUp	3m43s	cluster-autoscaler	pod triggered scale-up: [{liftie-hgy6h4jb-css-cluster-NodeGroup 2->3 (max: 5)}]
Warning	FailedScheduling	29s (x3 over 36s)	default-scheduler	0/4 nodes are available: 1 node(s) had volume node affinity conflict, 1 node(s) were unschedulable, 2 node(s) had intolerated taint {role.node.kubernetes.io/liftie-infra: true}. preemption: 0/4 nodes are available: 4 Preemption is not helpful for scheduling..
Warning	FailedScheduling	10s	default-scheduler	0/3 nodes are available: 1 node(s) had volume node affinity conflict, 2 node(s) had intolerated taint {role.node.kubernetes.io/liftie-infra: true}. preemption: 0/3 nodes are available: 3 Preemption is not helpful for scheduling..

Answer:

This could be because we are using SPOT instances and hence the nodes may be shutdown/terminated without notice. Use ON_DEMAND if we need dedicated instances.