**Lab work:**

**Prerequisites**

To gain access to your cluster, download and install a few CLI tools and the IBM Cloud Container Service plug-in.

1. Download the [**IBM Cloud CLI**](https://console-tok02.bluemix.net/docs/cli/reference/bluemix_cli/get_started.html#getting-started).

2. Download the [**Kubernetes CLI**](https://kubernetes.io/docs/user-guide/prereqs/).

3. Install the container service plugin.

bx plugin install container-service -r Bluemix

4. Log in to your IBM Cloud account

bx login

5. Set the context for the cluster in in your CLI

Get the command to set the environment variable and download the Kubernetes configuration files

bx cs cluster-config <<cluste-name>>

6. Set the KUBECONFIG environment variable. Copy the output from the previous command and paste it in your terminal. The command output should look similar to the following.

export KUBECONFIG=/Users/ibm/.bluemix/plugins/container-service/clusters/mycluster/kube-config-hou02-mycluster.yml

7. Verify that you can connect to your cluster by listing your worker nodes.

kubectl get nodes

8. Verify whether the client and server is properly set by running the following command:

kubectl version

You should see the following output:

Client Version: version.Info{Major:"1", Minor:"9", GitVersion:"v1.9.3", GitCommit:"d2835416544f298c919e2ead3be3d0864b52323b", GitTreeState:"clean", BuildDate:"2018-02-07T12:22:21Z", GoVersion:"go1.9.2", Compiler:"gc", Platform:"darwin/amd64"}

Server Version: version.Info{Major:"1", Minor:"8+", GitVersion:"v1.8.8-2+9d6e0610086578", GitCommit:"9d6e06100865789613cbac936edce948f0710a2f", GitTreeState:"clean", BuildDate:"2018-02-23T08:20:09Z", GoVersion:"go1.8.3", Compiler:"gc", Platform:"linux/amd64"}

**Creating a Pod using YAML**

1. Use the nginx\_pod.yaml file for creation of POD. To create the POD, run the following command.

kubectl create -f nginx\_pod.yaml

You should view the following output:

kubectl create -f nginx\_pod.yaml

pod "nginx" created

1. To list the number of PODS, run the following command.

kubectl get pods

You should view the following output:

kubectl get pods

NAME                        READY     STATUS    RESTARTS   AGE

nginx                       1/1       Running   0          1m

1. To get more details on the POD, then run the following command.

kubectl describe pods nginx

You should view the following output:

kubectl describe pods nginx

Name:         nginx

Namespace:    default

Node:         10.47.122.78/10.47.122.78

Start Time:   Mon, 23 Apr 2018 10:17:40 +0530

Labels:       app=nginx

Annotations:  <none>

Status:       Running

IP:           172.30.179.218

Containers:

  nginx:

    Container ID:   docker://8890d80740913807183b7c13d658c50f163d51e06296021221ef83b495e2125d

    Image:          nginx

    Image ID:       docker-pullable://nginx@sha256:18156dcd747677b03968621b2729d46021ce83a5bc15118e5bcced925fb4ebb9

    Port:           80/TCP

    State:          Running

      Started:      Mon, 23 Apr 2018 10:17:42 +0530

    Ready:          True

    Restart Count:  0

    Environment:    <none>

    Mounts:

      /var/run/secrets/kubernetes.io/serviceaccount from default-token-bj984 (ro)

Conditions:

  Type           Status

  Initialized    True

  Ready          True

  PodScheduled   True

Volumes:

  default-token-bj984:

    Type:        Secret (a volume populated by a Secret)

    SecretName:  default-token-bj984

    Optional:    false

QoS Class:       BestEffort

Node-Selectors:  <none>

Tolerations:     node.alpha.kubernetes.io/notReady:NoExecute for 300s

                 node.alpha.kubernetes.io/unreachable:NoExecute for 300s

Events:

  Type    Reason                 Age   From                   Message

  ----    ------                 ----  ----                   -------

  Normal  Scheduled              4m    default-scheduler      Successfully assigned nginx to 10.47.122.78

  Normal  SuccessfulMountVolume  4m    kubelet, 10.47.122.78  MountVolume.SetUp succeeded for volume "default-token-bj984"

  Normal  Pulling                4m    kubelet, 10.47.122.78  pulling image "nginx"

  Normal  Pulled                 4m    kubelet, 10.47.122.78  Successfully pulled image "nginx"

  Normal  Created                4m    kubelet, 10.47.122.78  Created container

  Normal  Started                4m    kubelet, 10.47.122.78  Started container

1. Anything that the application would normally send to STDOUT becomes logs for the container within the Pod.

We can retrieve these logs using the command:

kubectl logs $POD\_NAME

1. Start a bash session in the Pod’s container, run the following command:

kubectl exec -ti $POD\_NAME bash

You should be able to view the following output:

kubectl exec -ti nginx bash

root@nginx:/#

**Creating a Service using YAML**

1. Use the nginx-service.yaml file to create the Service. To create the service run the following command:

kubectl create –f nginx-service.yaml

You should view the following output:

kubectl create -f nginx-service.yaml

service "nginxservice-4" created

1. To list the number of services, run the following command:

kubectl get services

You should view the following output:

kubectl get services

NAME             TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE

kubernetes       ClusterIP   172.21.0.1       <none>        443/TCP          44d

nginxservice-4   NodePort    172.21.201.231   <none>        80:30091/TCP     2m

1. To view more details about the service, run the following command:

kubectl describe service nginxservice-4

You should view the following output:

kubectl describe service nginxservice-4

Name:                     nginxservice-4

Namespace:                default

Labels:                   app=nginx

Annotations:              <none>

Selector:                 app=nginx,tier=display

Type:                     NodePort

IP:                       172.21.201.231

Port:                     <unset>  80/TCP

TargetPort:               80/TCP

NodePort:                 <unset>  30091/TCP

Endpoints:                <none>

Session Affinity:         None

External Traffic Policy:  Cluster

Events:                   <none>

**Creating a Deployment using YAML**

1. To create a deployment use the nginx-deployment.yaml from the github. To create a deployment run the following command:

kubectl create –f nginx-deployment.yaml

You should view the following output:

kubectl create -f nginx-deployment.yaml

deployment "nginxdeploy-4" created

1. To view more details on the deployment, run the following command:

kubectl describe deployment nginxdeploy-4

You should view the following output:

kubectl describe deployment nginxdeploy-4

Name:               nginxdeploy-4

Namespace:          default

CreationTimestamp:  Mon, 23 Apr 2018 10:42:29 +0530

Labels:             app=nginx

                    tier=display

Annotations:        deployment.kubernetes.io/revision=1

Selector:           app=nginx,tier=display

Replicas:           1 desired | 1 updated | 1 total | 1 available | 0 unavailable

StrategyType:       Recreate

MinReadySeconds:    0

Pod Template:

  Labels:  app=nginx

           tier=display

  Containers:

   front-end:

    Image:        nginx

    Port:         <none>

    Environment:  <none>

    Mounts:       <none>

  Volumes:        <none>

Conditions:

  Type           Status  Reason

  ----           ------  ------

  Available      True    MinimumReplicasAvailable

OldReplicaSets:  <none>

NewReplicaSet:   nginxdeploy-4-7db997cfc5 (1/1 replicas created)

Events:

  Type    Reason             Age   From                   Message

  ----    ------             ----  ----                   -------

  Normal  ScalingReplicaSet  55s   deployment-controller  Scaled up replica set nginxdeploy-4-7db997cfc5 to 1

1. To get the list of deployments, run the following command:

kubectl get deployments

You should view the following output:

kubectl get deployments

NAME            DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE

nginxdeploy-4   1         1         1            1           3m

**Search the PODS based on Labels**

To retrieve the list of PODS using the labels attached to the PODS, run the following command:

kubectl get pods -l <<label name>>=<<label value>>

You should view the following command:

kubectl get pods -l app=nginx

NAME                             READY     STATUS    RESTARTS   AGE

nginx                            1/1       Running   0          30m

nginxdeploy-4-7db997cfc5-sg8qk   1/1       Running   0          5m

**Search the Services based on Labels**

To retrieve the list of Services using the labels attached to the services, run the following command:

kubectl get service -l <<label name>>=<<label value>>

You should view the following command:

kubectl get service -l app=nginx

NAME             TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE

nginxservice-4   NodePort   172.21.201.231   <none>        80:30091/TCP   22m

**Some of the kubectl commands for exploration:**

**Basic Commands (Beginner):**

create Create a resource from a file or from stdin.

expose Take a replication controller, service, deployment or pod and expose it as a new Kubernetes Service

run Run a particular image on the cluster

set Set specific features on objects

**Basic Commands (Intermediate):**

get Display one or many resources

explain Documentation of resources

edit Edit a resource on the server

delete Delete resources by filenames, stdin, resources and names, or by resources and label selector

**Deploy Commands:**

rollout Manage the rollout of a resource

rolling-update Perform a rolling update of the given ReplicationController

scale Set a new size for a Deployment, ReplicaSet, Replication Controller, or Job

autoscale Auto-scale a Deployment, ReplicaSet, or ReplicationController

**Cluster Management Commands:**

cluster-info Display cluster info

top Display Resource (CPU/Memory/Storage) usage.

drain Drain node in preparation for maintenance

taint Update the taints on one or more nodes

**Troubleshooting and Debugging Commands:**

describe Show details of a specific resource or group of resources

logs Print the logs for a container in a pod

attach Attach to a running container

exec Execute a command in a container

port-forward Forward one or more local ports to a pod

proxy Run a proxy to the Kubernetes API server

cp Copy files and directories to and from containers.

auth Inspect authorization

**Advanced Commands:**

apply Apply a configuration to a resource by filename or stdin

patch Update field(s) of a resource using strategic merge patch

replace Replace a resource by filename or stdin

convert Convert config files between different API versions

**Settings Commands:**

label Update the labels on a resource

annotate Update the annotations on a resource

completion Output shell completion code for the specified shell (bash or zsh)

**Other Commands:**

api-versions Print the supported API versions on the server, in the form of "group/version"

config Modify kubeconfig files

help Help about any command

plugin Runs a command-line plugin

version Print the client and server version information