

Container Workshop

Lab: ACS-Kubernetes

Table of Contents

[Exercise 1: Create Service Principle 3](#_Toc473487689)

[Task 1: Get Azure CLI 2.0 & login to Azure 3](#_Toc473487690)

[Task 2: Create a service principle 4](#_Toc473487691)

[Exercise 2: Working with ACS-Kubernetes 5](#_Toc473487692)

[Task 1: Create Kubernetes Cluster 5](#_Toc473487693)

[Task 2: Install kubectl 6](#_Toc473487694)

[Task 3: Wait 6](#_Toc473487695)

[Task 4: Configure kubectl to connect to cluster 6](#_Toc473487696)

[Task 5: Deploy the application to kubernetes 7](#_Toc473487697)

[Exercise 3: Setup Kubernetes dashboard 9](#_Toc473487698)

[Task 1: Start the dashboard proxy 9](#_Toc473487699)

[Exercise 4: Cleanup 10](#_Toc473487700)

[Task 1: Cleanup 10](#_Toc473487701)

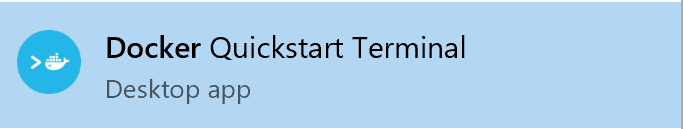
# Exercise 1: Create Service Principle

In this lab, you will create a service principle. In Azure Container Service, Kubernetes requires an Azure Active Directory service principal as a service account to interact with Azure APIs

Estimated time to complete this lab: **20 minutes**

### Task 1: Get Azure CLI 2.0 & login to Azure

1. Open Docker Quick start Terminal. If you have Docker Quick start Terminal open from previous lab, please close and reopen.



1. Execute the following command to download and run a Docker images that comes with Azure CLI 2.0

docker run -it -p 8001:8001 azuresdk/azure-cli-python:latest bash

1. Execute the following command at bash command prompt

az login -u <username> -p <password>

You will see JSON output like the following:

[

{

"cloudName": "AzureCloud",

"id": "6e7ce629-5859-4837-bce5-571fe7b268c5",

"isDefault": false,

"name": "MTC Houston Labs",

"state": "Enabled",

"tenantId": "a8e59e50-6360-4372-a629-9a9bf465158e",

"user": {

"name": "ratella@mtchouston.net",

"type": "user"

}

}

]

### Task 2: Create a service principle

1. Execute the following commands at bash command prompt

az account set --subscription="mySubscriptionID"

az ad sp create-for-rbac --role="Contributor" --scopes="/subscriptions/mySubscriptionID"

mySubscriptionID=id from the output of the previous command

1. Output would look like the following:

"appId": "186c2dbc-43bb-4b8c-9f56-1de777e58519",

"displayName": "azure-cli-2017-01-16-04-53-47",

"name": "http://azure-cli-2017-01-16-04-53-47",

"password": "c42e3268-988f-43bf-bfab-38e0ed8bd2b9",

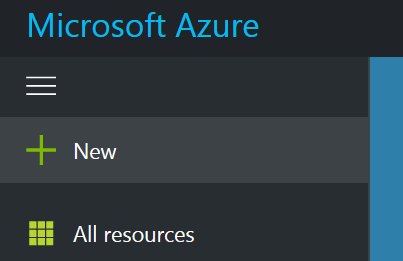
"tenant": "a8e59e50-6360-4372-a629-9a9bf465158e"

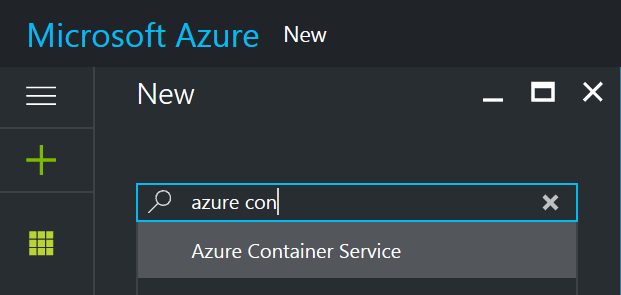
1. Copy “appid” and “password” into text document. You will need them for the next step

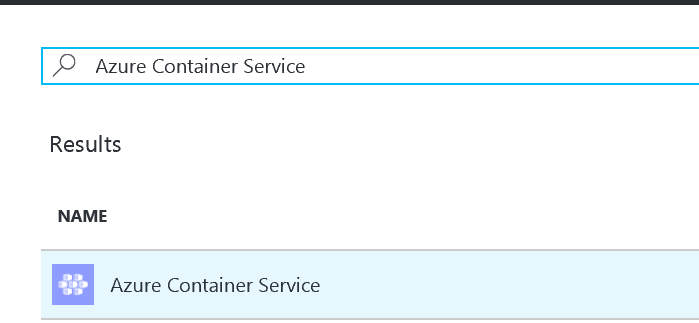
# Exercise 2: Working with ACS-Kubernetes

### Task 1: Create Kubernetes Cluster

1. Create kubernetes cluster via Azure portal







*Username:* acsadmin *SSH public Key:* content from acs\_rsa.pub (Content will start with ssh-rsa) *Resource group:* workshop-acskubernetes\_rg *Location:* South Central US *Orchestrator:* kubernetes *Agent count:* 3 *DNS prefix:* <first2lettersfirstname><first5letterslastname>kubernetes (has to be unique)

### Task 2: Install kubectl

1. Execute the following command at the bash command prompt

az acs kubernetes install-cli

1. Copy private key into the Docker container running the bash command prompt and save it.

mkdir -p /root/.ssh && vi /root/.ssh/id\_rsa

copy content from your local acs\_rsa (C:\Users\<username>\.ssh) and paste it in the above file (content will being with -----BEGIN RSA)

### Task 3: Wait

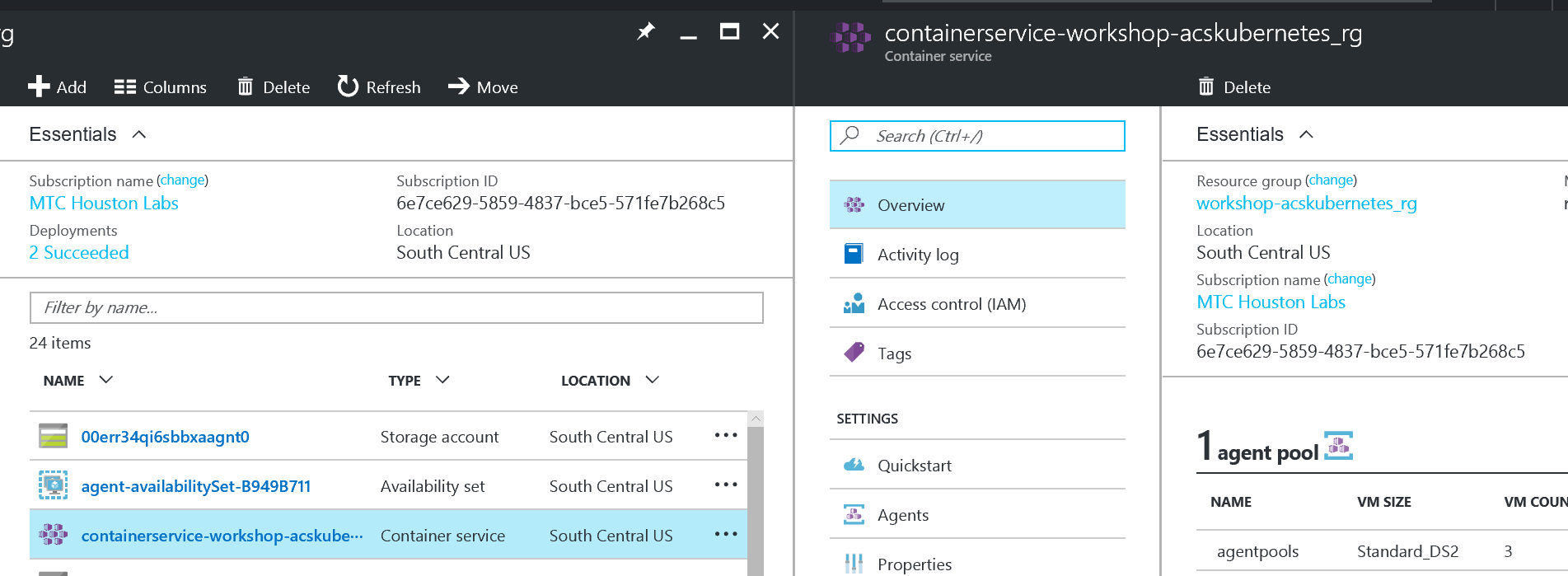
1. Wait till the cluster creation completes before proceeding to the next task.

### Task 4: Configure kubectl to connect to cluster

1. Execute the following command in the bash command prompt

az acs kubernetes get-credentials --resource-group=<RESOURCE\_GROUP> --name=<SERVICE\_NAME>

RESOURCE\_GROUP would be the name used in the cluster creation and SERVICE\_NAME can be found in azure portal

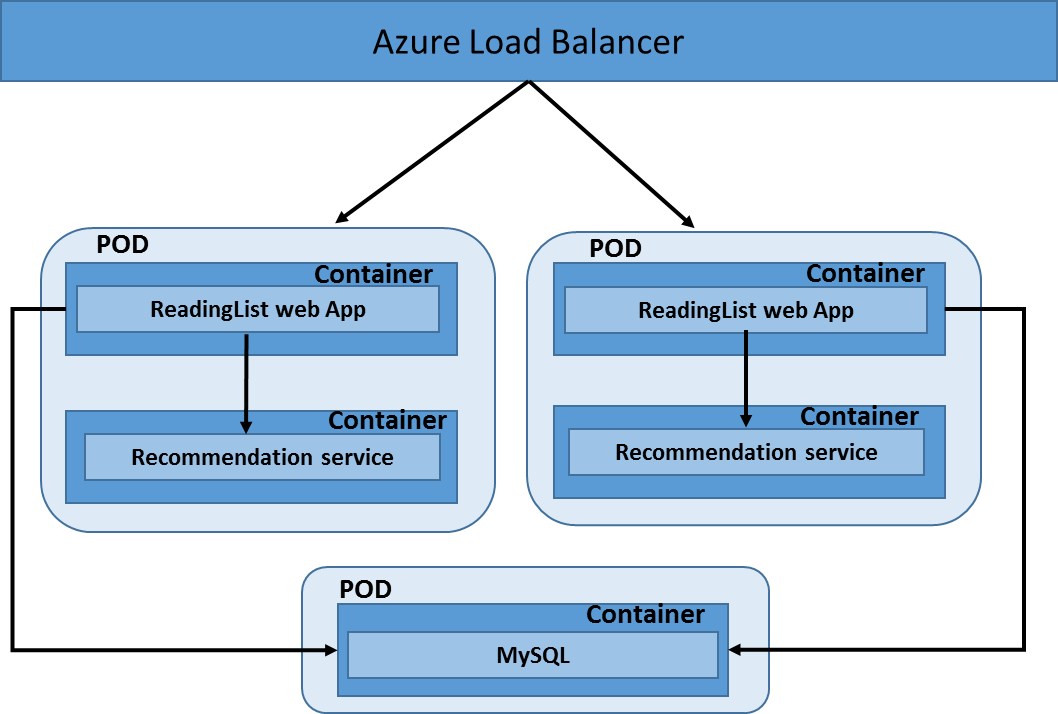


**Example:** az acs kubernetes get-credentials --resource-group="workshop-acskubernetes\_rg" --name="containerservice-workshop-acskubernetes\_rg"

### Task 5: Deploy the application to kubernetes

In this task, you will deploy the readinglist application stack to Kubernetes cluster. In kubernetes a group of one or more containers run as a pod. Pods can also have shared storage for the containers running in the pod.

At the end of this task you will have a total of 3 pods. Two for the app tier and one for MySQL.. The app tier pods will have both “ReadingList web app” and “Recommendation service”. There will be a total of 5 containers across 3 pods.



1. Execute the following command to install Git

apk add git

1. Execute the following command to download the deployment files

git clone <https://github.com/RaviTella/PCFAzureReadingListApplication>

1. Make the Kubernetes directory your current directory

cd PCFAzureReadingListApplication/src/main/kubernetes/

1. Execute the following command to deploy the mysql pod

kubectl create -f ./mysql-deployment.yaml

1. Execute the following command to create a service for the mysql pod

kubectl create -f mysql-service.yaml

1. Execute the following command to deploy the app tier pod containing a web application and the web service

kubectl create -f ./readinglist-deployment.yaml

1. Execute the following command to create a service for the app tier pod

kubectl create -f ./readinglist-service.yaml

1. Access the application from a browser

http://<IP>/readinglist

Use EXTERNAL-IP from the output of the following command as the IP

kubectl get svc web

1. Check the number of pods

kubectl get pods

1. Scale the web tier pod

kubectl scale deployment/web --replicas=2

1. Check the number of pods. You will now see 2 app tier pods

kubectl get pods

1. Execute the following command to display the web tier pod endpoints being load balanced by the app tier service

kubectl get ep web

# Exercise 3: Setup Kubernetes dashboard

Kubernetes dashboard is web interface that provides general-purpose monitoring and operations for Kubernetes clusters

### Task 1: Start the dashboard proxy

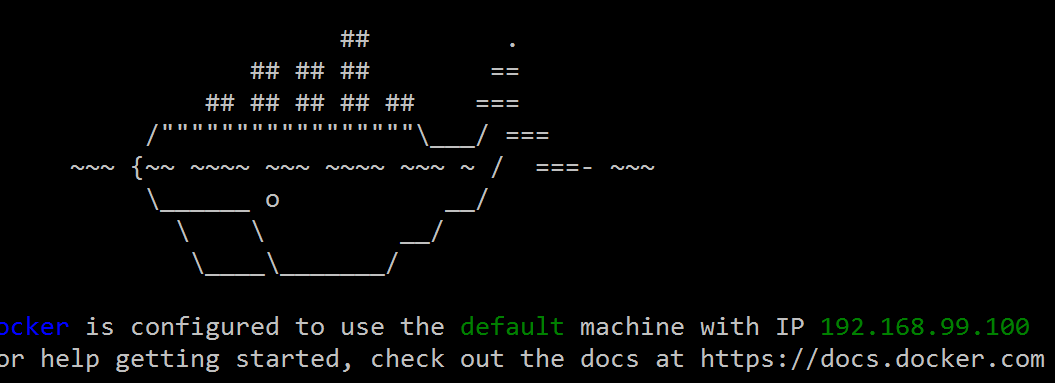
1. Start the dashboard

kubectl proxy --accept-hosts "" --address='0.0.0.0'

1. Access the dashboard

http://<IP>:8001/ui

IP address can be found at start of the docker quickstart terminal



# Exercise 4: Cleanup

Optionally. cleanup the resources you have created in this lab

### Task 1: Cleanup

1. Delete the resource group for the cluster. This will delete all the resources you have created in this lab.

