# Azure Container Services – AKS – Kubernetes

**Introduction:**

This document will help you to deploy kubernetes on azure.

**Requirements:**

* Microsoft Azure Subscription

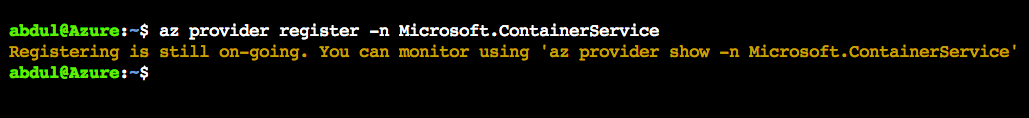
**Step – 1:**

Login to your azure account and open cloud shell environment from azure.

****

Before you deploy any resource towards azure you need to make sure that you have registered the resource provider on your subscription, you can register the resource provider on your subscription using the below command:

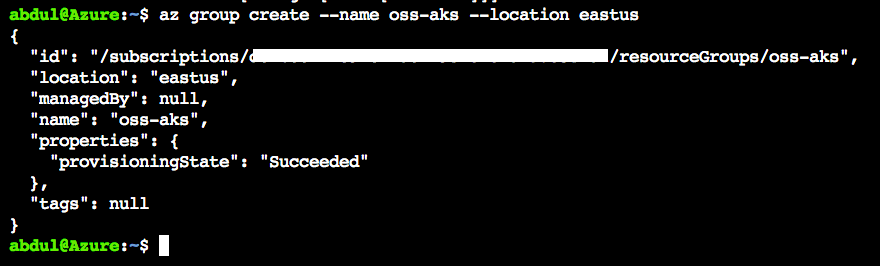
**az provider register –n Microsoft.ContainerService**



**Step – 2:**

Once when the resource provider is registered, now try creating a resource group on which the container service will be located

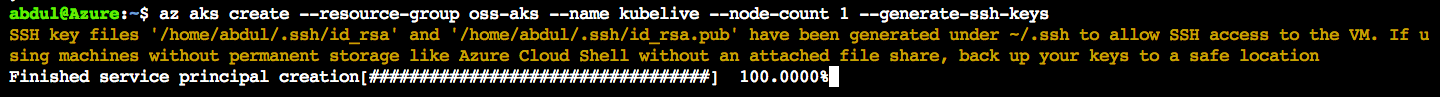
**az group create --name oss-aks --location eastus**

****

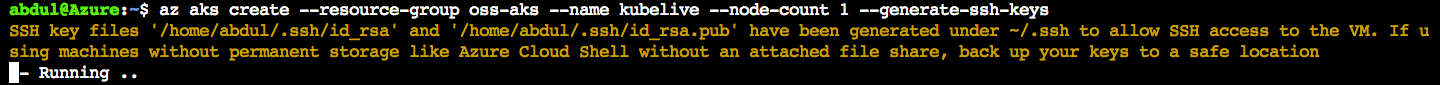
**Step – 3:**

Lets now try creating nodes and generate ssh keys using the below command by considering the resource group and location which has been created previously.

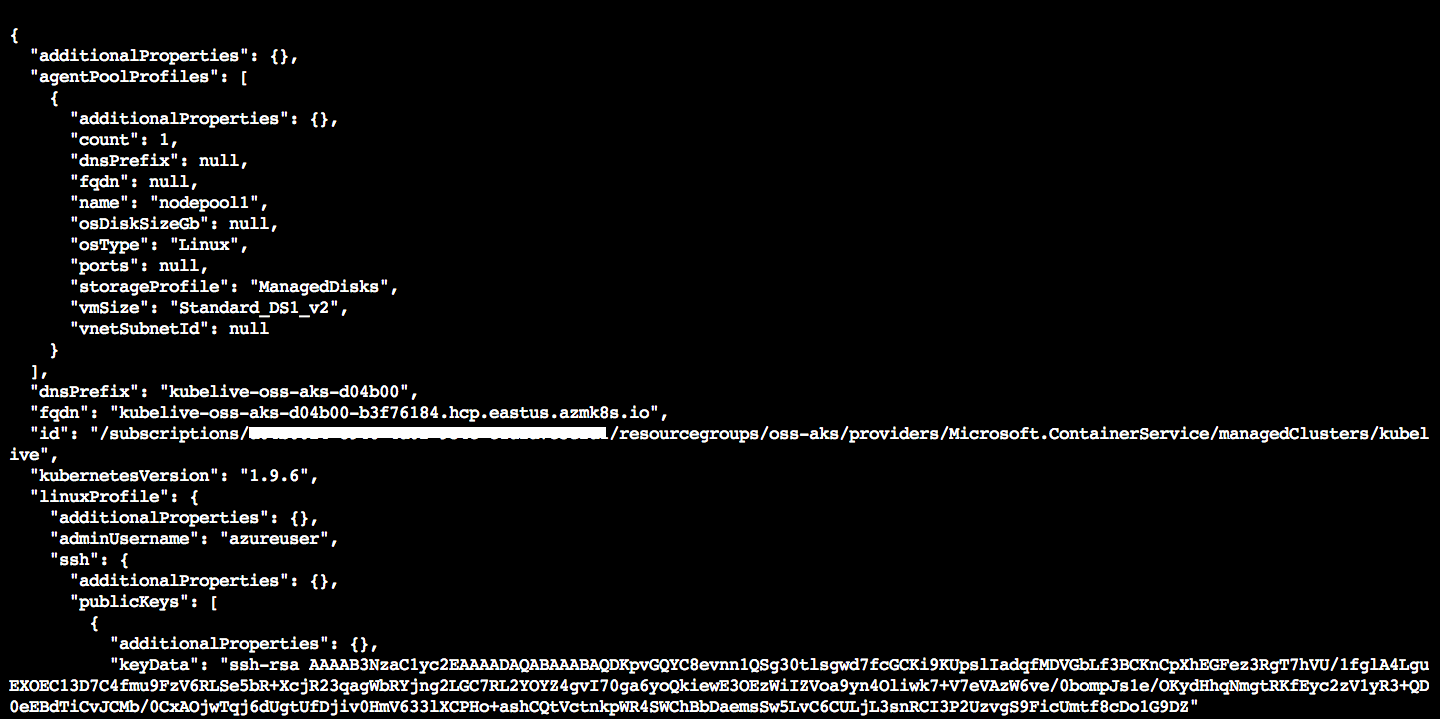
**az aks create --resource-group oss-aks --name kubelive --node-count 1 --generate-ssh-keys**

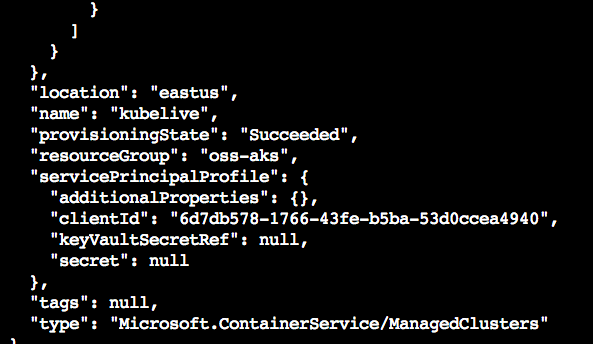


Note – the below status as running means that the node is getting created.



Once when the nodes are deployed you will get the following output as shown:





**Step – 4: (optional)**

**Note –** you no need to do this if you are working from cloud shell as kubectl is installed by default on cloud shell, you can perform the below steps of kubectl installation if you are working from powershell.

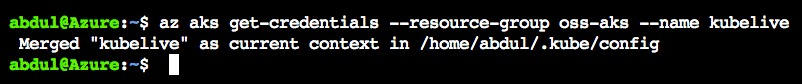
Lets now grab the latest kubectl using the below command:

**az aks install-cli**

**Step – 5:** You can configure the kubectl using the below command which will get the certificates, credentials and other details to communicate with the node:

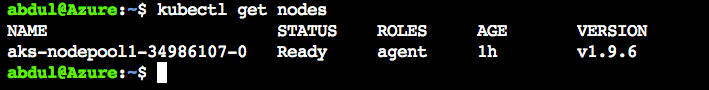
**az aks get-credentials --resource-group oss-aks --name kubelive**

Once when you run the above command you will get the output as shown below

****

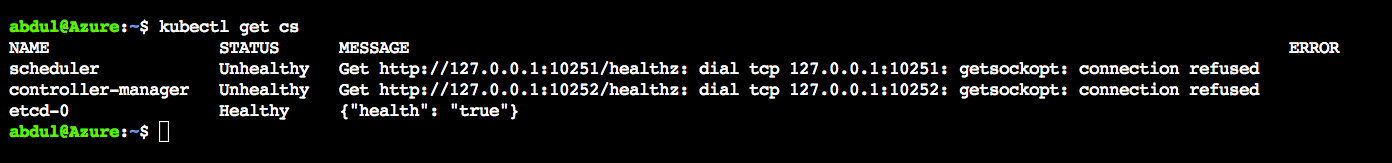
you can get the nodes using the below command for which the kubectl has been configured:

**kubectl get nodes**

****

you can also get the health status of the nodes by using the below command:

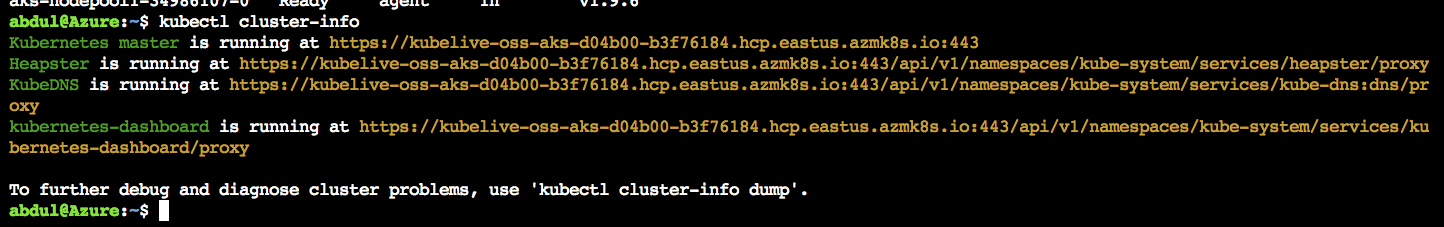
**kubectl get cs**

****

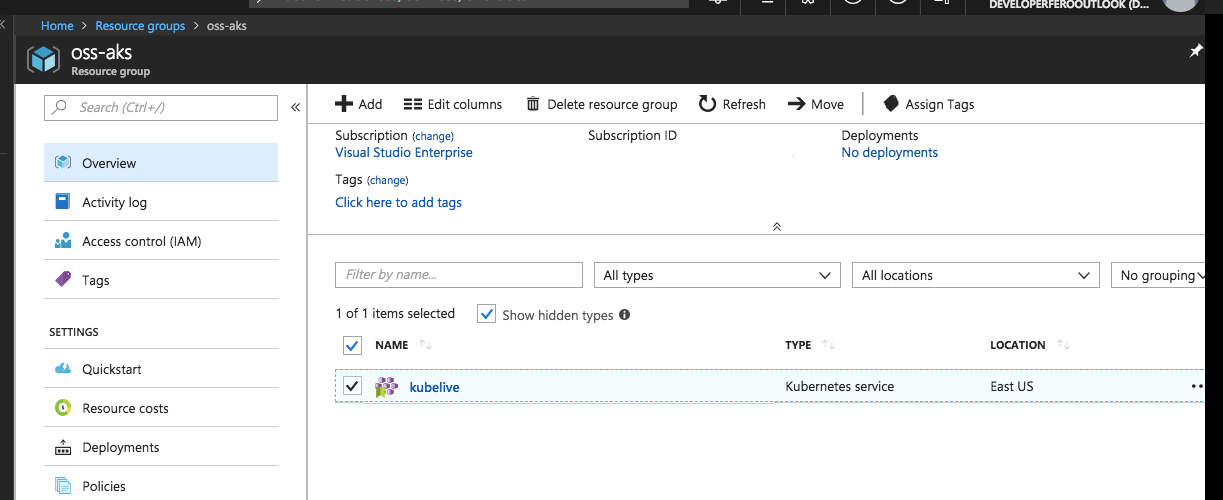
You can now try getting the cluster information using the below command:

**Kubectl cluster-info**

This will list out the endpoint information for the entire cluster as shown below.

****

Once when the cluster is deployed you can find the resource named “kubelive” which is a pointer that will not have access but it’s a single point of entry which maps to multi tenants of nodes. This will be there available on the resource group which was created by us via cloud shell.



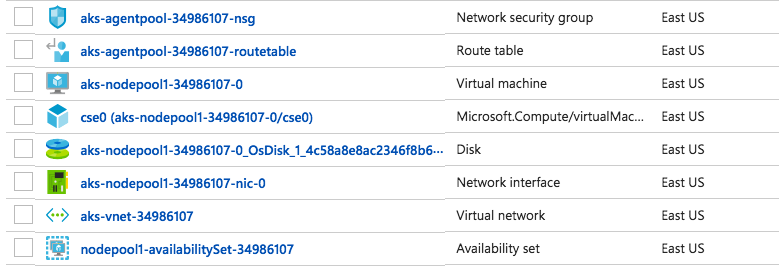
**Other components of Kubernetes:**

You should also find another resource group which has been created by azure automatically as shown below:

../../../../Desktop/Screen%20Shot%202018-05-03%20at%205.45.52%20PM.png

This resource group holds the following components of the kubernetes cluster which was deployed:

* Network security group
* Route table
* Virtual machine
* Disk
* Network interface
* Virtual network
* Availability set



Further you can also try deploying a nginx image on the k’8 cluster by running the below command:

**kubectl run –image=nginx myweb**

You can get the deployment status of the same by using the below command:

**Kubectl get deploy**

You can get the pod status by running the below command:

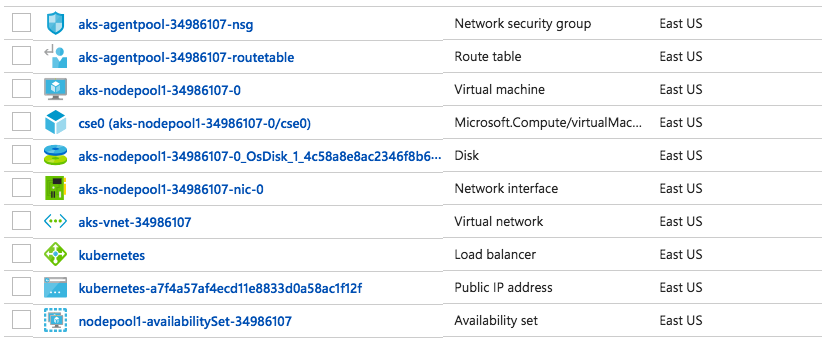
**Kubectl get pods**

Further you can expose the web app which has been deployed with a load balancer by opening the port 80 using the below command:

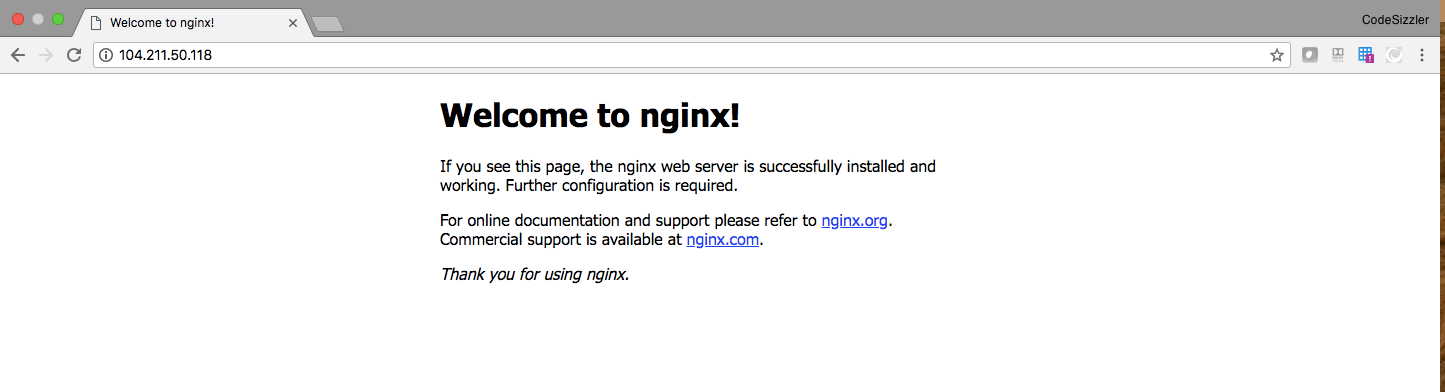
**Kubectl expose pod myweb-999d5fcbc-vhp95 --port=80 --target-port=80 --type=LoadBalancer**

****

**So now here goes a load balancer which has been created:**

****

**You can check the application which has been deployed on the cluster using the public ip address as shown below:**

****