**Managed K8s**

* Managed K8s cluster will manage master nodes i.e., we don’t have explicit access to master nodes
* so, cluster administrational activities such as
  + backing up k8s cluster
  + upgrading k8s cluster
  + cloud controller manager configurations to
    - have access to native cloud networks
    - have csi implementations specific to cloud provider
    - have cni implementations specific to cloud provider
* Cloud providers charge hourly for cluster and they give sla’s
* If you want you can manage nodes or even that you can have with cloud provider.

**AKS cluster creation and features**

* AKS: <https://azure.microsoft.com/en-in/products/kubernetes-service>
* Addons <https://learn.microsoft.com/en-us/azure/aks/integrations#available-add-ons>
* Extensions: <https://learn.microsoft.com/en-us/azure/aks/cluster-extensions?tabs=azure-cli#currently-available-extensions>
* To create aks cluster <https://learn.microsoft.com/en-us/azure/aks/learn/quick-kubernetes-deploy-cli>

**EKS Cluster**

* Elastic Kubernetes Services is a managed k8s from aws
* EKS cluster can be created in many ways
  + aws console
  + aws cli
  + terraform
  + eksctl this will be used
* Features <https://aws.amazon.com/eks/features/>
* Create a Linux instance, install aws cli, create iam credentials
* install kubectl <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/#install-using-other-package-management>
* we had followed direct installation <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/#install-kubectl-binary-with-curl-on-linux>
* Install eksctl <https://eksctl.io/introduction/#for-unix>

# for ARM systems, set ARCH to: `arm64`, `armv6` or `armv7`

ARCH=amd64

PLATFORM=$(uname -s)\_$ARCH

curl -sLO "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl\_$PLATFORM.tar.gz"

# (Optional) Verify checksum

curl -sL "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl\_checksums.txt" | grep $PLATFORM | sha256sum --check

tar -xzf eksctl\_$PLATFORM.tar.gz -C /tmp && rm eksctl\_$PLATFORM.tar.gz

sudo mv /tmp/eksctl /usr/local/bin

* Create a file called as cluster.yaml with the following content

apiVersion: eksctl.io/v1alpha5

kind: ClusterConfig

metadata:

name: basic-cluster

region: us-west-2

nodeGroups:

- name: basic

instanceType: t2.large

desiredCapacity: 2

volumeSize: 20

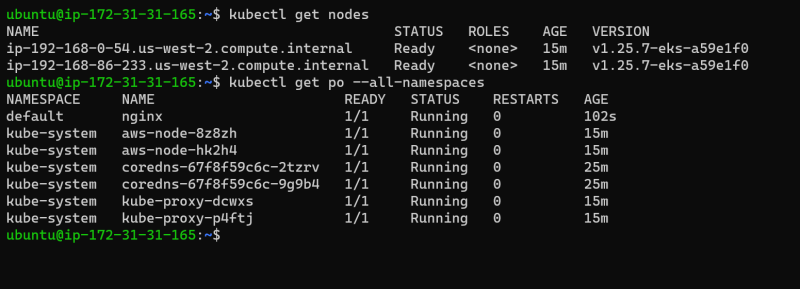
ssh:

allow: true # will use ~/.ssh/id\_rsa.pub as the default ssh key

* Execute ssh-keygen ssh-keygen
* Now execute the command eksctl create cluster -f cluster.yaml
* After creation execute

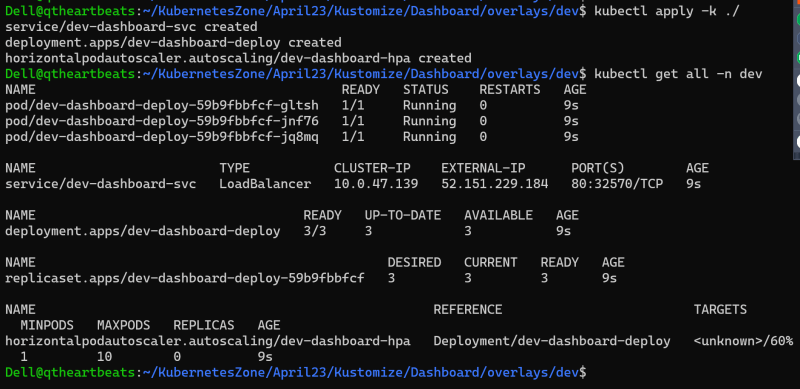
kubectl get nodes

kubectl get pods --all-namespaces



# **AWS Fargate**: Serverless compute for containers. In AWS Fargate billing would be as per pod.

**Kustomize [Base and overlay configuration]**

* Kustomize is a tool where we can natively manage configurations
* <https://kustomize.io/> for kustomize
* Natively manage dynamic configurations to k8s manifests
* Let’s write a k8s manifest
  + to deploy shaikkhajaibrahim/dashboardservice:1.0.0 which runs on 80 port
  + create a service file exposed as LoadBalancer
* <https://github.com/asquarezone/KubernetesZone/commit/866afac25f2867fc53669794d4a76b4a8546f2e0> for manifests
* <https://www.vultr.com/docs/how-to-configure-kubernetes-resources-with-kustomize/> for tutorial from vultr to use customize
* <https://github.com/asquarezone/KubernetesZone/commit/e731bf4f070e9675dfe26aa2fe4ee87df22f33ea> for the manifest folder structure
* Now lets add name prefix per environment
* <https://kubectl.docs.kubernetes.io/references/kustomize/kustomization/nameprefix/> for nameprefix docs.
* <https://github.com/asquarezone/KubernetesZone/commit/45fe363ca28c39fd738c9a15f27d1282fbe544c4> for the changes done  
  
* <https://github.com/asquarezone/KubernetesZone/commit/734b945733ea08d652012bc599acd4736de35720> for labels per env
* <https://github.com/kubernetes-sigs/kustomize/tree/master/examples> for kustomize examples