Virtual machine vs containers :

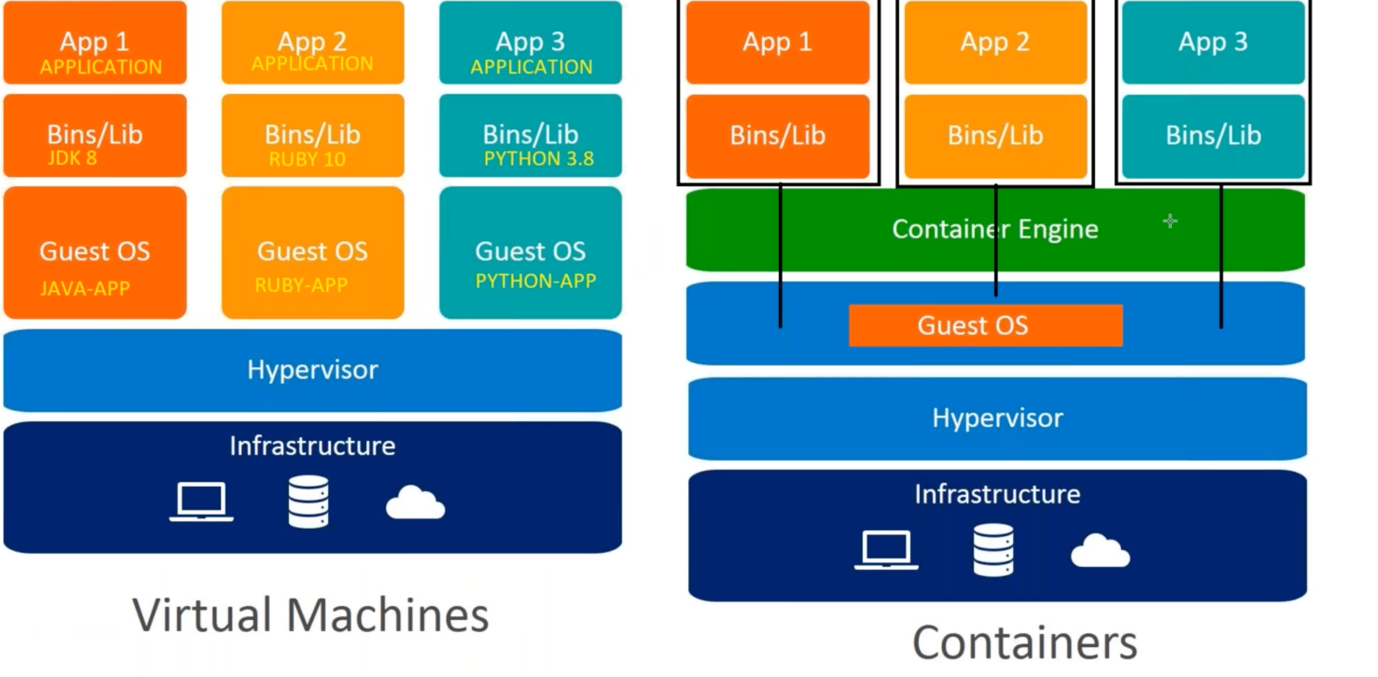
Monolitics vs microservices :

Why do we need to go with containers?

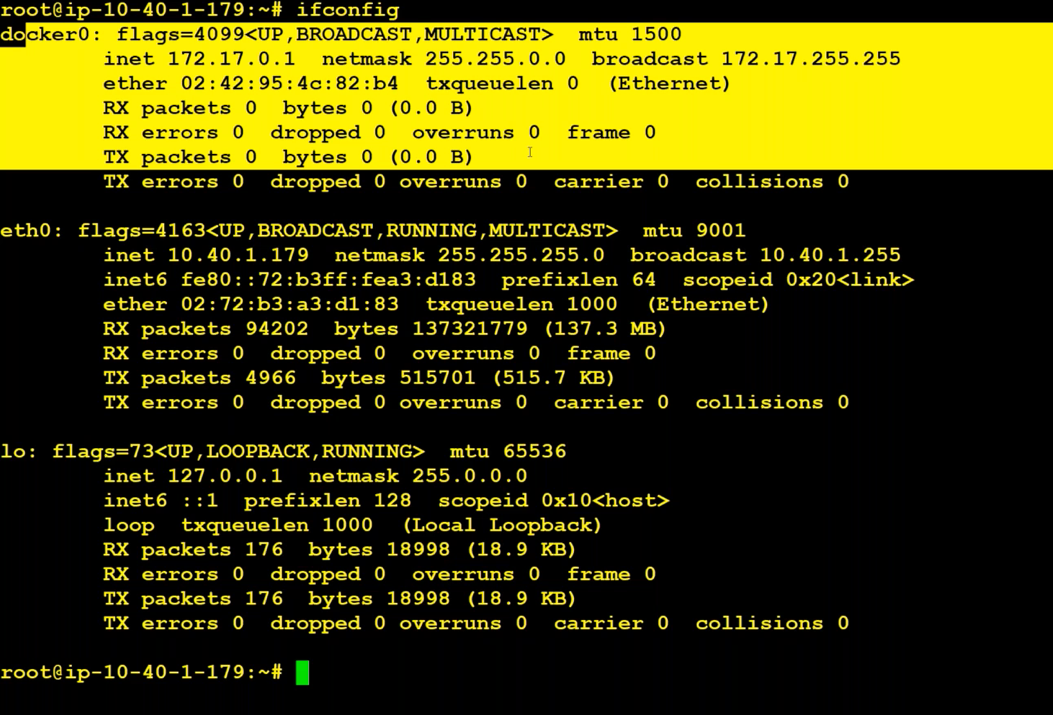
* To save money.
* Deploy app to production asap. The image we used in dev can be used in production also, If we go with containers.

What is a container ?

It is a process. It does not have os. It uses underlying host kernel.



Once we install docker(curl <https://get.docker.com/>), then a virtual network gets created, all containers uses this network.



**Kubernetes architecture:**

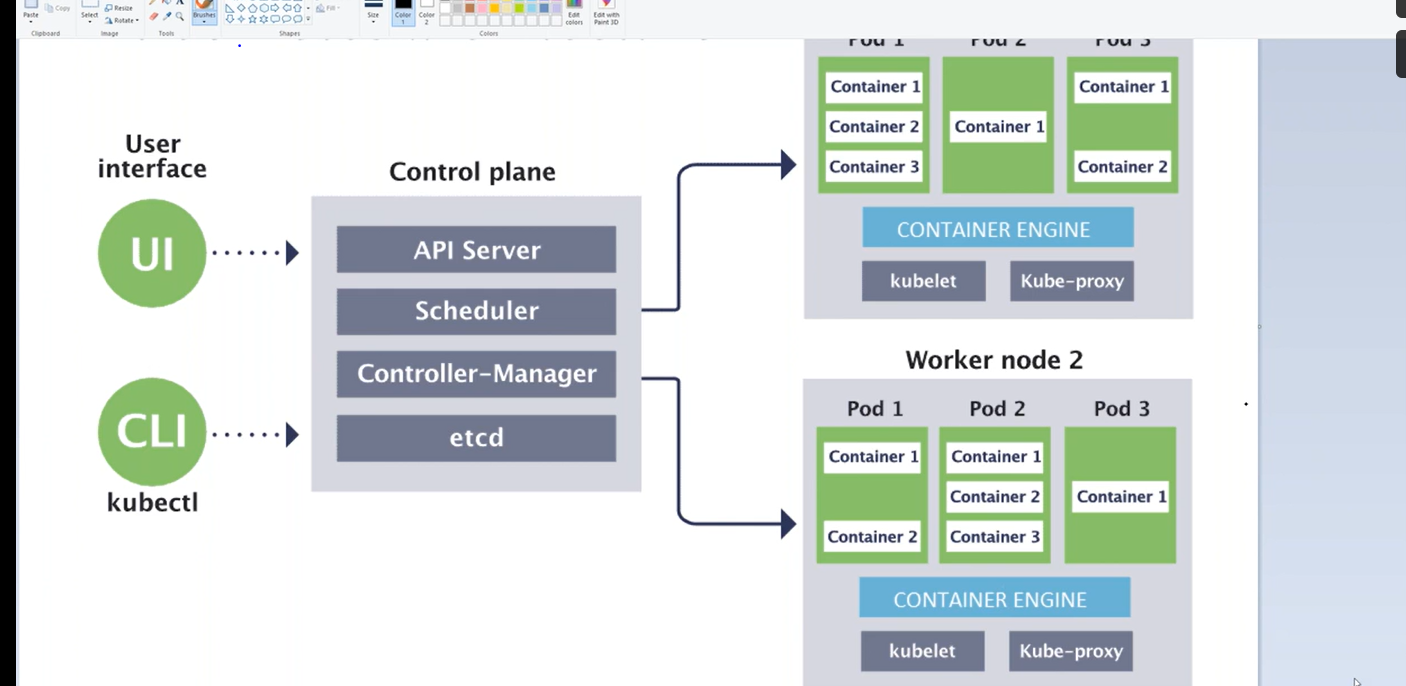
Kubernetes architecture is divided into two parts : 1. Control plane and Worker Plane.

Control plane contains API Server, Scheduler, Control manager, etcd. To engage with Kubernetes cluster, we need to touch API Server. Etcd is a encrypted database that contains everything happens in the cluster. API Servers instructs Scheduler to schedule the replicas we asked, etc.,

Worker Node contains pods, container engine, kubectl and kube-proxy. Kubelet is a demon/continuous service, that communicates with API Server for updates. Kube-proxy is used to proxy the incoming traffic or getting the request and sending appropriate response. Kube-proxy maintains network rules.

Applications will be deployed only in Worker nodes. Control plane is just for management purpose.

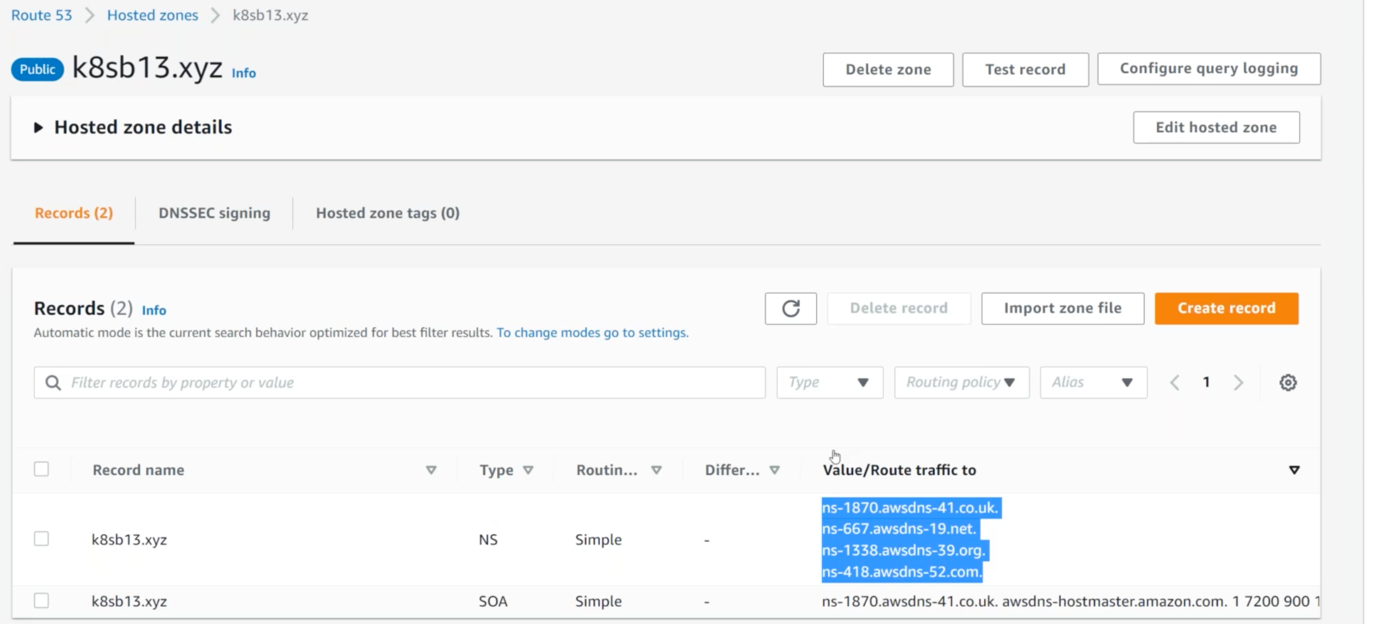
**Previous written lines :** We should communicate with API Server to do anything in the cluster. API server writes everything into the etcd. This is a encrypted database. Scheduler tells the kubelet on how many containers to be deployed. Kubectl keeps track of api server on the new updates



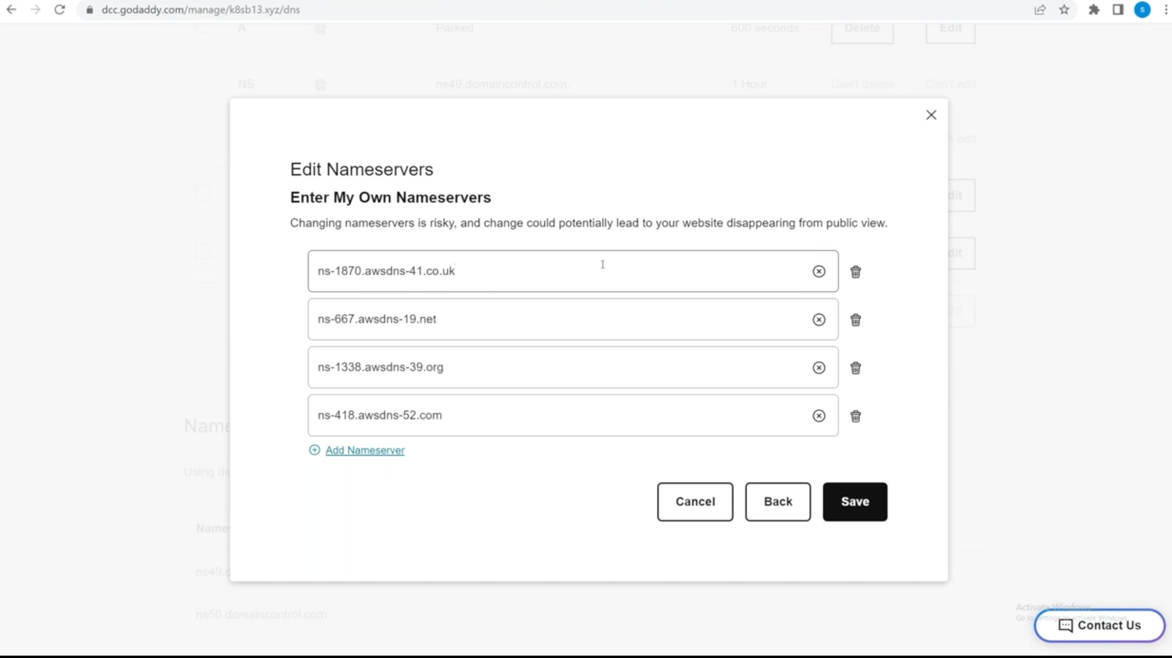
Dependencies for creating K8S Cluster :

1. DNS name – we can buy that in GoDaddy.
   1. Buy a domain from godaddy.com
2. Create zone in Route53 using the domain name you have purchased.

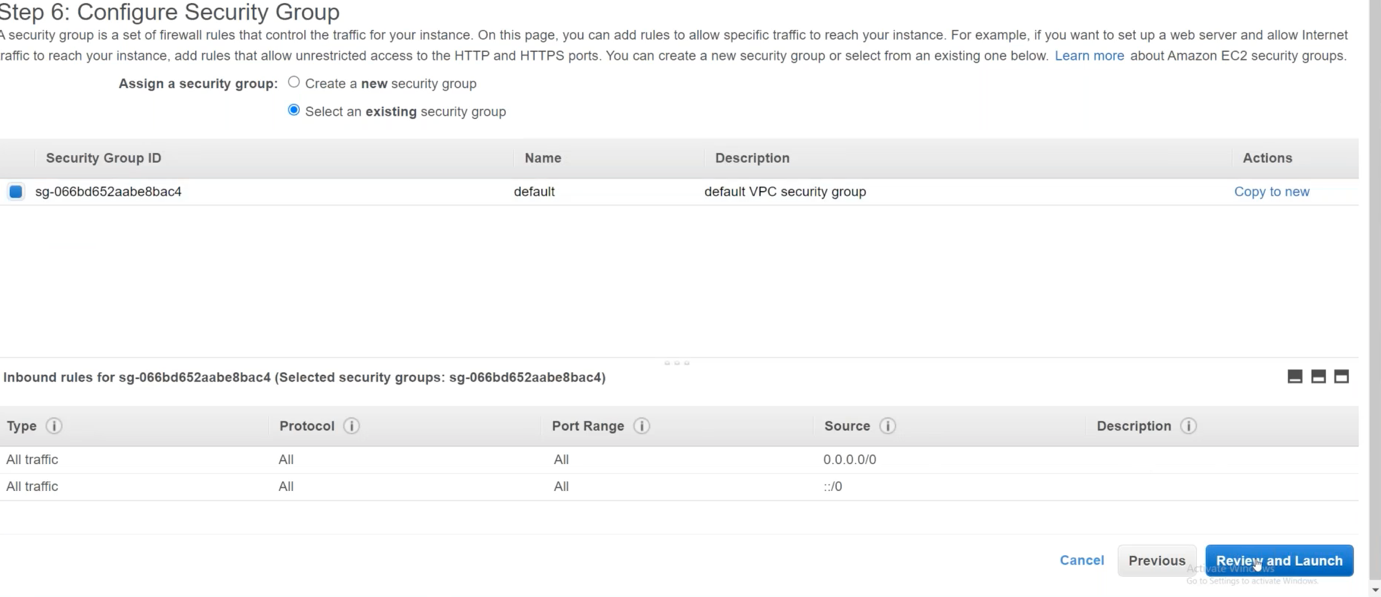
Once we create zone, we get nameserver ips. We have to give those ips in the godaddy website.



1. Create S3 Bucket for saving the config.
2. Create EC2 Server and apply a role with admin access.
   1. Create a Role
      1. Select EC2 and select AdministratorAccess polic, give name to role and click on enter
      2. Then apply role to instance.
3. Download kops(to create a cluster) and kubectl(to manage the cluster).
   1. Open ‘kops github’
      1. <https://github.com/kubernetes/kops>
      2. Go to previous releases(<https://github.com/kubernetes/kops>).
      3. Copy link address of kops-linux-amd64(https://github.com/kubernetes/kops/releases/download/v1.24.2/kops-linux-amd64).
      4. Check 1.21.2 version available or not by pasting the link in url and check it is downloading or not.( <https://github.com/kubernetes/kops/releases/download/v1.21.2/kops-linux-amd64>)
      5. Login to this server.
      6. Run commands
         1. sudo su –
         2. apt update && apt install -y unzip jq net-tools
         3. cd /usr/local/bin
         4. wget <kops 1.21.2 version link> 🡪 to download kops
         5. wget <https://github.com/kubernetes/kops/releases/download/v1.21.2/kops-linux-amd64>
      7. Rename and change the permission using below commands.
         1. mv kops-linux-amd64 kops
         2. chmod 777 kops.
      8. Install kubectl -- <https://kubernetes.io/docs/tasks/tools/> -- https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/
         1. Move to /use/local/bin and Download using curl -LO <https://dl.k8s.io/release/v1.25.0/bin/linux/amd64/kubectl>
         2. Change the permission using chmod 777 kubectl
4. Generate ssh-keys. (If we download these keys, automatically these keys will be used to create cluster).
   1. Change directory by giving command 🡪 cd 🡪 it comes to root.
   2. Use command ssh-keygen to create a public and private key automatically.
5. Copy the nameserver-ips from Route53 and paste it in under godaddy websites DNS server.



1. Export environment variables in ec2-instance.
   * 1. export NAME=k8sb14.xyz
     2. export KOPS\_STATE\_STORE=s3://k8sb14.xyz
     3. export AWS\_REGION=us-east-1
     4. export CLUSTER\_NAME=k8sb14.xyz
     5. export EDITOR=’usr/bin/nano’
     6. or in root directory give command nano .bashrc, we will get a file add these exports there save and give source .bashrc.
     7. Check these are added properly or not by giving command ‘env’.
2. Use K8S command to create a K8S cluster.



export NAME=k8sb14.xyz

export KOPS\_STATE\_STORE=s3://k8sb14.xyz

export AWS\_REGION=us-east-1

export CLUSTER\_NAME=k8sb14.xyz

export EDITOR=’usr/bin/nano’

check kops and kubectl version using below commands :

kops version

kubectl version

sudo su –

apt update && apt install -y unzip jq net-tools

cd /usr/local/bin

wget <https://github.com/kubernetes/kops/releases/download/v1.21.2/kops-linux-amd64>

mv kops-linux-amd64 kops

chmod 777 kops.

curl -LO <https://dl.k8s.io/release/v1.25.0/bin/linux/amd64/kubectl>

chmod 777 kubectl

cd

ssh-keygen