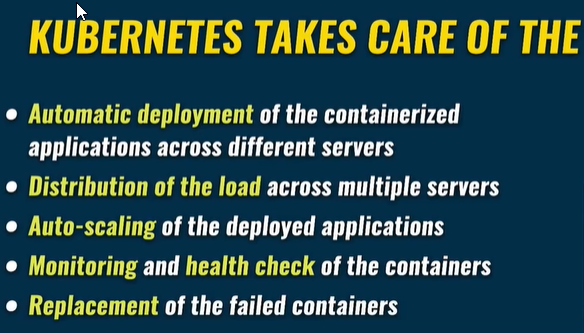


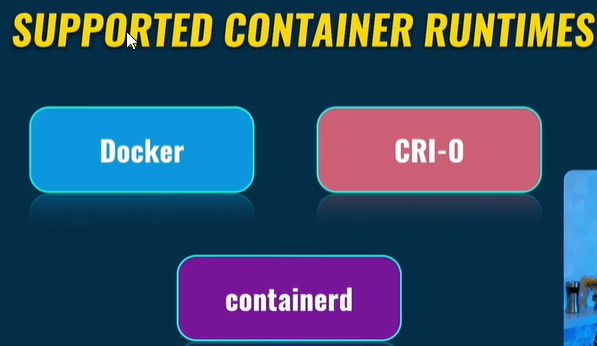
Using docker you could create a container on any computer. For creating multiple containers on different servers/computers you could get into trouble. Kubernetes allows that, in physical or virtual servers.

Kubernetes takes care of the automatic deployment of the containerized services across different servers.

Distribution of the load across different servers.

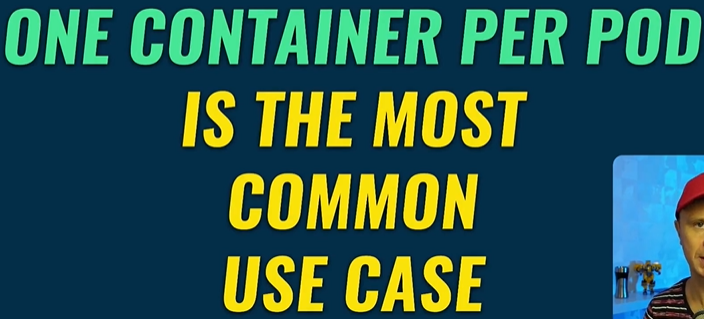
Autoscaling of the deployed applications.

Monitoring and health check of the containers.

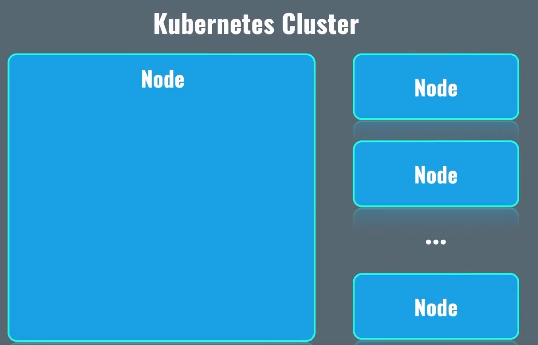
Replacement of the failed containers.

A screenshot of a computer

Description automatically generatedThus Kubernetes can run without Docker at all.



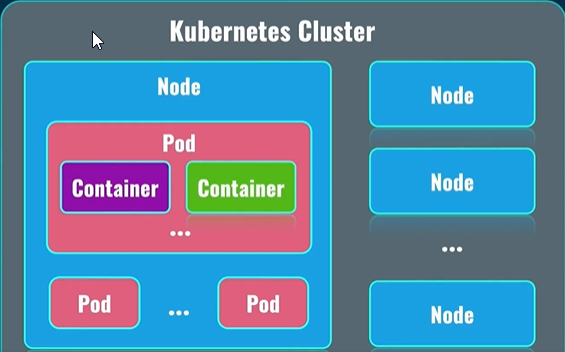
Containers are inside the pod. All containers inside the pod share the same volumes and the same IP Address.

One pod: One server, more servers not possible.

Kubernetes Cluster consist of nodes, nodes are nothing but server either bare-metal or virtual server.

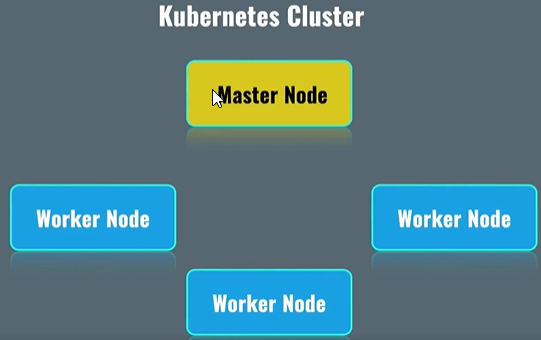
A screenshot of a computer

Description automatically generatedUsually nodes which are in the same Kubernetes cluster, are located close to each other.

Inside of the nodes, there are pods. Pods are the again, smallest possible unit in Kubernetes.

Inside the pods, there are containers, usually single containers per pod. Such pods are created on different nodes. All of this is done automatically for you. Your job is to create such nodes, and form clusters, based on those nodes. Nodes will not automatically form cluster without intervention.

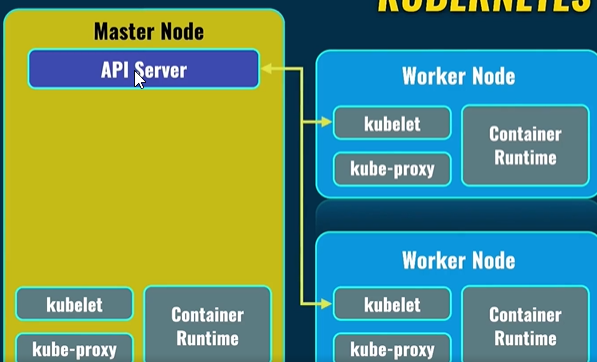
But after such intervention, everything will be automated. Kubernetes will automatically form pods on different nodes.

How the nodes communicate with each other and how are they managed:

Master Node’s job to manage worker nodes. Distribute load across worker nodes. All pods related to your applications are deployed on worker nodes. Master Nodes work on system pods, which are responsible for actual work on system pods in general.

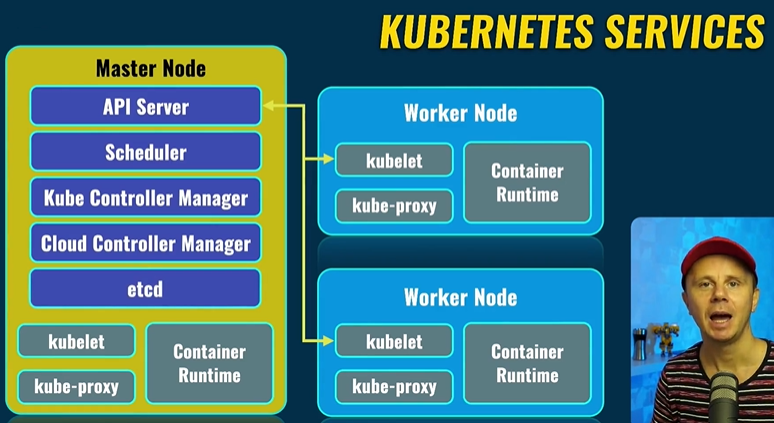
A screenshot of a computer

Description automatically generatedKubernetes Services:

Container Runtime: It runs actual containers inside of each node. Some container runtimes are docker, crio and container-d.

Kubelet: Communicates with API Server service in the master node. API Server service is the main point of communication between different nodes and inside of each node in the Kuberenetes world.

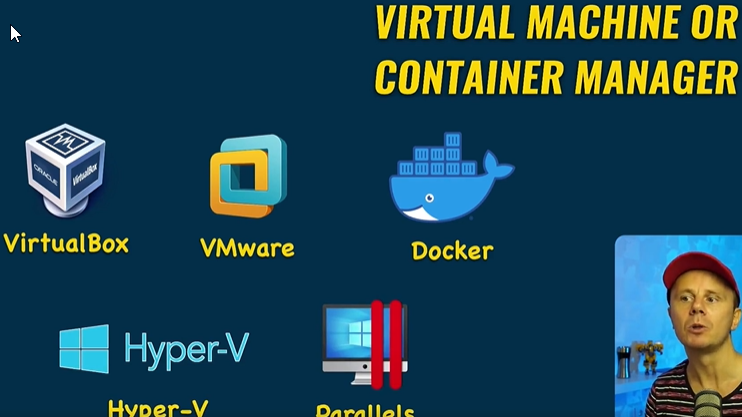
Scheduler: Load distribution.

Cloud Controller Manager: Interact with the cloud provider.

A diagram of a cluster

Description automatically generated

A blue background with yellow text

Description automatically generatedKubectl: Kube Control manages cluster.

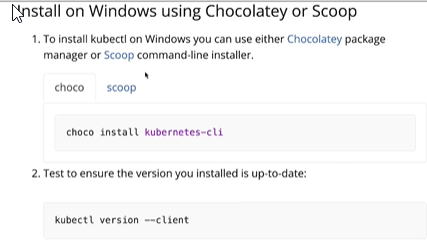
Create Cluster for free, with one node, which will be both worker node and master node.

To run minikube successfully virtual machine or container manager is required.

Don’t use Docker since crio or containerd options not available.

A screenshot of a computer

Description automatically generated



That’s all that is required, minikube, cubectl and VS Code.



A screenshot of a software

Description automatically generated

A screenshot of a computer

Description automatically generatedA white background with black text

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

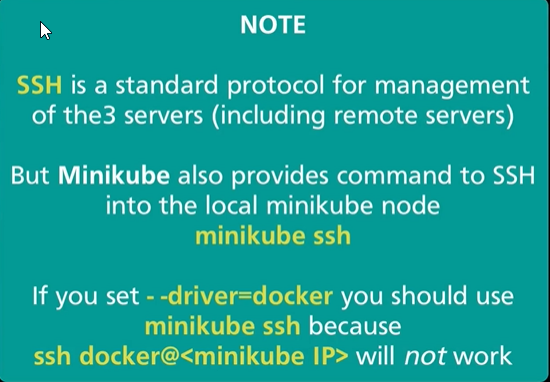
Description automatically generated

A computer screen shot of a program

Description automatically generated



IP assigned to the virtual machine which is running our Kubernetes node created by minikube.



A screen shot of a computer

Description automatically generated



A screenshot of a computer screen

Description automatically generatedOnce inside the node, list the docker containers

A black background with white text

Description automatically generated

A number on a black background

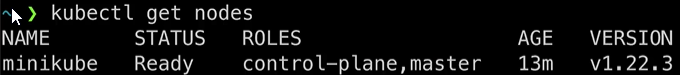
Description automatically generatedKubectl is not available inside the node.

A blue sign with white text

Description automatically generatedThat’s why exit the ssh connection.

A black background with yellow text

Description automatically generated



A black background with white letters

Description automatically generated

A computer screen shot of a black background

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generatedA yellow sign with blue letters

Description automatically generatedSystem pods running on this master node.

A black background with white text

Description automatically generated

A black background with white text

Description automatically generatedNginx is the name of the docker image which will be pulled automatically, and the new container will be created based on the image. This container will run inside of the Kubernetes pod.

A black background with white text

Description automatically generated

A screenshot of a computer

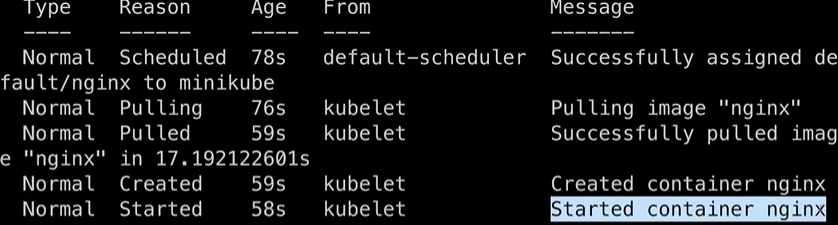
Description automatically generatedDocker inside the Kubernetes node, fetches the nginx image from DockerHub, and creates corresponding container based on the image.

Node: Means where the specific pod was created.

Recall: Kubernetes distributes load across a cluster and selects specific node for a specific pod.

A computer screen with text and numbers

Description automatically generatedBut in order to access the pod, we will not be able to use the internal IP. We have to use services for that.

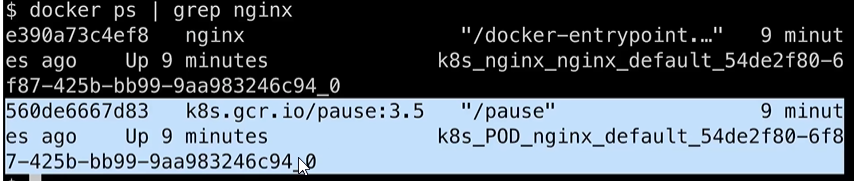


A black background with white text

Description automatically generated

A yellow sign with blue letters

Description automatically generated



Two docker containers related to nginx.

A black background with white text

Description automatically generatedThe pause container is required to keep namespace of the pod.

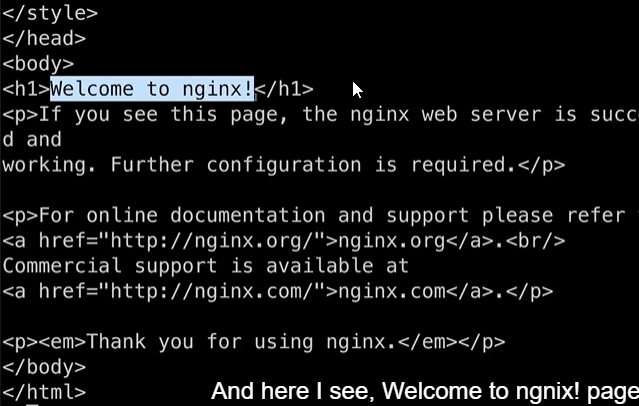
A black screen with white text

Description automatically generatedGo inside of the container using: docker exec -it id\_of\_the\_container

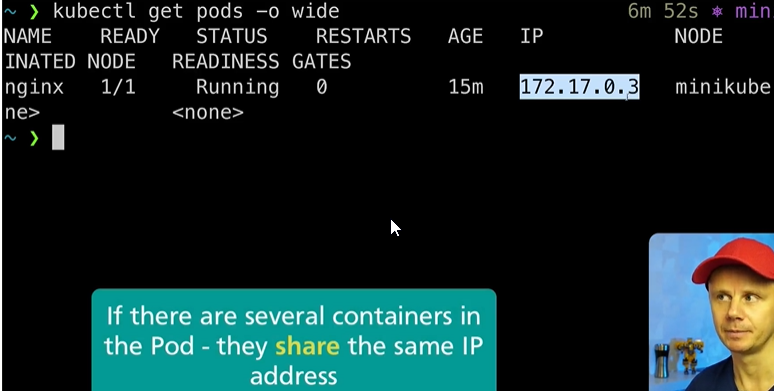
Check hostname and IP Address.

Now, to connect to the web server running inside of the container, using the above IP Address, we use the curl command.







Now we go out of this container, using the exit command, but we are still inside the Kubernetes node.

The IP Address is the internal IP address, and you will not be able to connect to the pod, from outside the cluster.

A black background with white text

Description automatically generatedWe created just a single pod using the kubectl run command. It is not scalable.

Creating alias for kubectl command.



For Windows user git is required.





Most efficient way to create pods, when you are able to increase quantity of the pods, decrease quantity, modify configuration etc. is by using deployments.

Deployments will be responsible for creation of the actual pods. Note: All pods inside the deployment will be the same, exactly.

But you could create multiple copies of the same pod, and distribute the load across different nodes in the Kubernetes cluster.

A black screen with white text

Description automatically generated

The given pod, is managed by the given deployment.

A screenshot of a computer program

Description automatically generated

Selectors are way to connect individual pods to deployments.

Replica sets are a set of replicas for your application. Because there could be many pods inside the deployment and all of them are included in the replica set.

