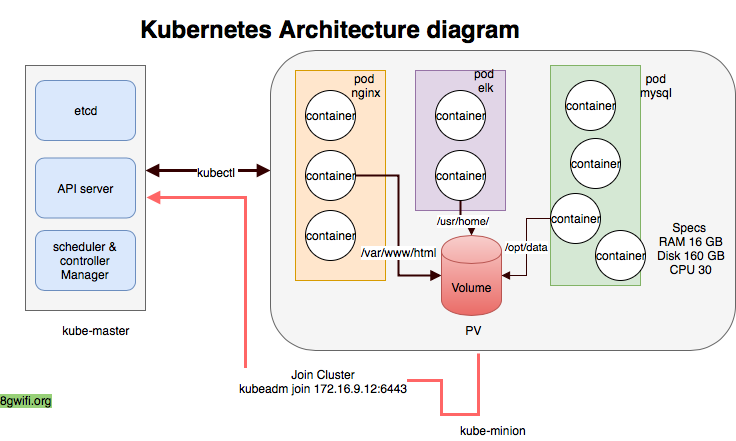
Wednesday, November 28, 2018

11:32 AM



From <<https://8gwifi.org/docs/kube-pods.jsp>>

Introduction

In this section we will learn the core concept of **kubernetes** like **Pod,cluster,Deployment,Replica Set**. The idea is to keep it simple and making more intuitive learning

At the beginning we have setup one master node and minion node

1. kube-master
2. kube-minion

**kubectl** - Main CLI tool for running commands and managing Kubernetes clusters. **so what is the cluster**

root@kube-master:$ kubectl cluster-info  
Kubernetes master is running at <https://172.16.9.12:6443>  
KubeDNS is running at <https://172.16.9.12:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy>

**cluster** consists of at least one **cluster** master and multiple worker machines called **nodes**

Verify the **nodes** which has join the cluster

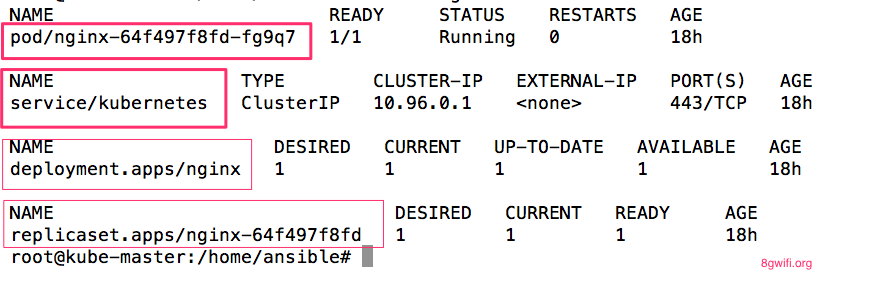
root@kube-master:$ kubectl get nodes   
NAME STATUS ROLES AGE VERSION  
kube-master Ready master 18h v1.11.0  
kube-minion Ready <**none**> 18h v1.11.0

Kubernetes API Resources

1. **po**: Pod : **Kubernetes pod** is a group of containers that are deployed together on the same host
2. **svc**: Service : A **Service** in **Kubernetes** is a REST object, similar to a Pod
3. **deploy**: Deployment : A ***Deployment*** controller provides declarative updates for Pods and ReplicaSets
4. **rs**: Replica Set : A ***ReplicaSet*** ensures that a specified number of pod replicas are running at any given time

Example : List all resources in the name space

kubectl get all



To List all supported resource types along with their **shortnames**, API group, whether they are namespaced, and Kind: run the kubectl api-resources

root@kube-master:$ kubectl api-resources  
NAME SHORTNAMES APIGROUP NAMESPACED KIND  
bindings true Binding  
pods po true Pod  
services svc true Service  
deployments deploy apps true Deployment  
replicasets rs extensions true ReplicaSet  
clusterroles rbac.authorization.k8s.io false ClusterRole  
rolebindings rbac.authorization.k8s.io true RoleBinding  
.......  
.......

Dig lighter to kubernetes flow

Start a single instance of **nginx** with one **replicaset**

root@kube-master:$ kubectl run nginx --image=nginx --replicas=1

The output will be deployment.apps/nginx created

Ok, so lets see if we actually have a ***Kubernetes pods up and running:***

root@kube-master:$ kubectl get pods  
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-sqmjk 0/1 ContainerCreating 0 12s

The **container is creating on nginx pod** , again hitting the same command, this time shows pods are up and running

root@kube-master:$ kubectl get pods   
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-sqmjk 1/1 Running 0 46s

We can retrieve a lot more information about each of these pods using kubectl describe pod. For example:

root@kube-master:$ kubectl describe pod nginx-64f497f8fd-sqmjk

There will be big YAML output

......  
 ......  
 Containers:  
 nginx:  
 Container ID: docker://a7bc2921ca62187778c5f65da4e139516f2701caf32e325cbeef2a1ee082da0b  
 Image: nginx  
 Image ID: docker-pullable://nginx@sha256:a65beb8c90a08b22a9ff6a219c2f363e16c477b6d610da28fe9cba37c2c3a2ac  
 Port: <**none**>  
 Host Port: <**none**>  
 State: Running  
 Started: Mon, 16 Jul 2018 18:48:53 +0530  
 Ready: True  
 Restart Count: 0  
 Environment: <**none**>  
 Mounts:  
 /var/run/secrets/kubernetes.io/serviceaccount from default-token-8wxrj (ro)

Here you can see configuration information about the **container(s) and Pod (labels, resource requirements, etc.),** as well as **status** information about the container(s) and **Pod (state, readiness, restart count, events, etc.).**

The container state is one of **Waiting, Running, or Terminated**. *here you can see that for a container in Running state*, the system tells you when the container started.

**Ready** tells you whether the container passed its last readiness probe.

**Look for Events** Look for the event generated to perform this action kubectl run nginx --image=nginx --replicas=1 in the kubernetes master node

Events:  
 Type Reason Age From Message  
 ---- ------ ---- ---- -------  
 Normal Scheduled 11m default-scheduler Successfully assigned default/nginx-64f497f8fd-7w5mn to kube-minion  
 Normal Pulling 10m kubelet, kube-minion pulling image "nginx"  
 Normal Pulled 10m kubelet, kube-minion Successfully pulled image "nginx"  
 Normal Created 10m kubelet, kube-minion Created container  
 Normal Started 10m kubelet, kube-minion Started container

In the **Minion** node docker images are pulled and container **created and started** , for docker background user can issue docker related command to see how **docker** & **kubernetes** are orchestrating the deployment

root@kube-minion:$ docker images  
REPOSITORY TAG IMAGE ID CREATED SIZE  
nginx latest 8b89e48b5f15 2 hours ago 109 MB  
nginx <**none**> 3c5a05123222 10 days ago 109 MB

To list all events you can use kubectl get events

Get the **deployment configuration** of nginx by the command

root@kube-master:$ kubectl get deployment nginx  
NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE  
nginx 1 1 1 1 6m

* **DESIRED** =1
* **CURRENT** =1

During the initializing state we have told the kubernetes to **maintain one replica only** by setting the flag --replicas=1

**Delete the nginx pod** to delete the pod **specify the pod name**

root@kube-master:$ kubectl delete pod nginx-64f497f8fd-sqmjk

Query on pod still s**hows one pod is running with diffrent id**, this happen because ***initially we have tell the kuberntes to keep one running replica always*** *Great Kubernetes !!! you have save my production server from accidently destroyed*

root@kube-master:$ kubectl get pods   
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-fg9q7 1/1 Running 0 1m

Scale Up the Pods

To scale up the pods tell to kubectl how many current replica is there (**current-replicas**) and how many needs to be scaled (**replicas**)

* **DESIRED** =3
* **CURRENT** =1

root@kube-master:$ kubectl scale --current-replicas=1 --replicas=3 deployment/nginx

In the background t**wo new container will get created** and will get deployed

root@kube-master:$ kubectl get pods  
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-brn22 0/1 ContainerCreating 0 17s  
nginx-64f497f8fd-fg9q7 1/1 Running 0 2h  
nginx-64f497f8fd-z2vbb 0/1 ContainerCreating 0 17s

After creating the containers check the status of these pods

root@kube-master:$ kubectl get pods  
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-brn22 1/1 Running 0 30s  
nginx-64f497f8fd-fg9q7 1/1 Running 0 2h  
nginx-64f497f8fd-z2vbb 1/1 Running 0 30s

**Look at the events**

kubectl get events



Scale Down the Pods

* **DESIRED** =1
* **CURRENT** =3

root@kube-master:$ kubectl scale --current-replicas=3 --replicas=1 deployment/nginx   
deployment.extensions/nginx scaled

Pods are **terminating**

root@kube-master:$ kubectl get pods   
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-fg9q7 1/1 Running 0 2h  
nginx-64f497f8fd-fpjk9 0/1 Terminating 0 35s  
nginx-64f497f8fd-lk2pw 0/1 Terminating 0 35s

After scaling down **only one nginx pod is running**

root@kube-master:$ kubectl get pods  
NAME READY STATUS RESTARTS AGE  
nginx-64f497f8fd-fg9q7 1/1 Running 0 2h

**Look at the events**



Adding New Node to Cluster

To add new node to the **kubernetes** cluster requires **token** and **discovery-token-ca-cert-hash**

Forget your token :), first create a token using **kubeadm** command in the kube-master setup,

root@kube-master:$ kubeadm token create   
I0717 10:32:47.753179 22047 feature\_gate.go:230] feature gates: &{map[]}  
yy8zho.n3w5inti3twy7v0y

Forget your **discovery-token-ca-cert-hash**,

Get rooCA cert fingerprint

root@kube-master:$ openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.\* //'

The outpted discovery-token-ca-cert-hash value

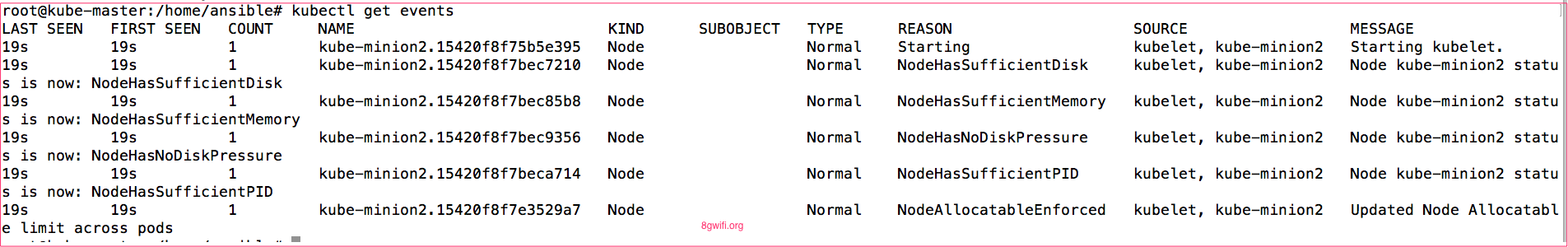
30e3baf5cb4474b23d5d2500836f6b4da19fa629b64339b1301d3e04892e08aa

Once the Token is created join the new node name **kube-minion2** using the **token** and **tokencacert**value

root@kube-minion2:$ kubeadm join 172.16.9.12:6443 --token yy8zho.n3w5inti3twy7v0y --discovery-token-ca-cert-hash sha256:30e3baf5cb4474b23d5d2500836f6b4da19fa629b64339b1301d3e04892e08aa

**Checkout for the events** kubectl get events

A set of event shows when adding a new node to the cluster



Now in this cluster **we have two minion nodes** and one master node

root@kube-master:$ kubectl get nodes  
NAME STATUS ROLES AGE VERSION  
kube-master Ready master 18h v1.11.0  
kube-minion Ready <**none**> 18h v1.11.0  
kube-minion2 Ready <**none**> 2m v1.11.0

**Again Now scale up the nginx**

kubectl scale --current-replicas=1 --replicas=5 deployment/nginx

Look for the Events to verify on the new node pods are created and nginx are deployed

kubectl describe pods | grep kube-minion



From <<https://8gwifi.org/docs/kube-pods.jsp>>