Pod LAB

03 January 2021 16:03

PODS

Check Pod existence on system

$kubectl get pods

```
controlplane $ kubectl get pods
No resources found in default namespace.
```

Pod Creation

$kubectl run nginx –image nginx

```
controlplane $ kubectl run nginx --image nginx
pod/nginx created
```

```
controlplane $ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          9s
```

Detail status of Pod:

$kubectl get pods -o wide

```
controlplane $ kubectl get pods -o wide
NAME    READY   STATUS    RESTARTS   AGE    IP           NODE     NOMINATED NODE   READINESS GATES
nginx   1/1     Running   0          2m7s   10.244.1.2   node01   <none>           <none>
```

Detailed logs of Pod:

$kubectl describe pod nginx

```
controlplane $ kubectl describe pod nginx
Name:         nginx
Namespace:    default
Priority:     0
Node:         node01/172.17.0.19
Start Time:   Sat, 02 Jan 2021 17:35:47 +0000
Labels:       run=nginx
Annotations:  <none>
```

```
Annotations:  <none>
Status:       Running
IP:           10.244.1.2
IPs:
  IP:  10.244.1.2
Containers:
  nginx:
    Container ID:   docker://fe533503e209d6ac103e4173bb7006f1134e4ab350b0d6a2db81a78c056e95e7
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:4cf620a5c81390ee209398ecc18e5fb9dd0f5155cd82adcbae532fec94006fb9
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Sat, 02 Jan 2021 17:35:54 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-9h4tm (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
```

```
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-9h4tm (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-9h4tm:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-9h4tm
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                 node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age    From               Message
  ----    ------     ----   ----               -------
  Normal  Scheduled  6m24s  default-scheduler  Successfully assigned default/nginx to node01
  Normal  Pulling    6m23s  kubelet, node01    Pulling image "nginx"
  Normal  Pulled     6m18s  kubelet, node01    Successfully pulled image "nginx" in 5.195897611s
  Normal  Created    6m17s  kubelet, node01    Created container nginx
  Normal  Started    6m17s  kubelet, node01    Started container nginx
controlplane $
```

Command auto completion

1. # yum install bash-completion
2. add the following to your ~/.bashrc file:

   source /usr/share/bash-completion/bash_completion
3. # type _init_completion

4. Source the completion script in your ~/.bashrc file:
   echo 'source <(kubectl completion bash)' >>~/.bashrc

   From <https://kubernetes.io/docs/tasks/tools/install-kubectl/#enabling-shell-autocompletion>

```
[root@k8s-master ~]# cat .bashrc
# .bashrc

# User specific aliases and functions

alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
source /usr/share/bash-completion/bash_completion
source <(kubectl completion bash)
```

Pod Deletion

$kubeclt delete pod nginx

```
controlplane $ kubectl delete pod nginx
pod "nginx" deleted
controlplane $
controlplane $
controlplane $ kubectl get pods
No resources found in default namespace.
controlplane $
```

 # kubectl delete pod mypod --grace-period=0 --force

YAML

https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.20/#pod-v1-core

```
[root@k8s-master ~]# cat yaml.yaml
# Mandatory fields are - apiVersioin, kind, metadta and spec
apiVersion: v1
kind: Pod
metadata:
  name: mypod
  labels:
    env: prod
    type: vm
spec:
  containers:
    - name: nginx
      image: nginx:1.14.2
      ports:
      - containerPort: 80
```

Another example:

```
[root@k8s-master ~]# kubectl run my-cool-app --image me/my-cool-app:v1 -o yaml --dry-run=client -o yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: my-cool-app
  name: my-cool-app
spec:
  containers:
  - image: me/my-cool-app:v1
    name: my-cool-app
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
```

Pod creation through yaml file

```
controlplane $ cat pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - image: nginx
    name: nginx
controlplane $
```

$kubectl apply -f pod.yaml

```
controlplane $ kubectl apply -f pod.yaml
pod/nginx created
controlplane $ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          7s
controlplane $
```

Generate POD Manifest YAML file (-o yaml). Don't create it(--dry-run)

$kubectl run nginx –image nginx –dry-run=client -o yaml

```
controlplane $ kubectl run nginx --image=nginx  --dry-run=client -o yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx
spec:
  containers:
  - image: nginx
    name: nginx
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
controlplane $
```

Replicaset

apiVersion = apps/v1
Kind:
Metadata:
Spec:
    Replicas:
    Selector:
Template:

https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/

Creation of replicaset through yaml file

```yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  labels:
    run: nginx
  name: nginx-replicaset
spec:
  replicas: 2
  selector:
    matchLabels:
      run: nginx
  template:
    metadata:
      labels:
        run: nginx
    spec:
      containers:
      - image: nginx
        name: nginx
```

$ kubectl apply -f replicaset.yaml

```
controlplane $ kubectl apply -f replicaset.yaml
replicaset.apps/nginx-replicaset created
controlplane $
```

Status of replicaset

$ kubectl get replicasets

```
controlplane $ kubectl get replicasets
NAME              DESIRED   CURRENT   READY   AGE
nginx-replicaset   2         2         2       12s
controlplane $
controlplane $
controlplane $ kubectl get rs
NAME              DESIRED   CURRENT   READY   AGE
nginx-replicaset   2         2         2       19s
controlplane $
```

Deletion of replicaset

$ kubectl delete replicaset nginx-replicaset

```
controlplane $ kubectl delete replicaset nginx-replicaset
replicaset.apps "nginx-replicaset" deleted
controlplane $
```

DEPLOYMENT

Create a deployment

$kubectl create deployment --image=nginx nginx

```
controlplane $ kubectl create deployment --image=nginx nginx
deployment.apps/nginx created
```

$kubectl get deployments

```
controlplane $ kubectl get deployments
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nginx   1/1     1            1           16s
```

```
controlplane $ kubectl get deployments -o wide
NAME    READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES   SELECTOR
nginx   1/1     1            1           65s   nginx        nginx    app=nginx
```

Generate Deployment YAML file (-o yaml). Don't create it(--dry-run)

$kubectl create deployment --image=nginx nginx --dry-run -o yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: nginx
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx
        name: nginx
        resources: {}
status: {}
controlplane $
```

Generate Deployment with 4 Replicas

$ kubectl create deployment nginx –image=nginx –replicas=4

```
controlplane $ kubectl create deployment nginx --image=nginx --replicas=4
deployment.apps/nginx created
controlplane $
controlplane $ kubectl get deployments
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nginx   3/4     4            3           5s
controlplane $
controlplane $ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6799fc88d8-65jkc   1/1     Running   0          16s
nginx-6799fc88d8-pwl5p   1/1     Running   0          16s
nginx-6799fc88d8-s5k88   1/1     Running   0          16s
nginx-6799fc88d8-zxclh   1/1     Running   0          16s
controlplane $
```

Deployment Deletion

$ kubectl delete deployments nginx

```
controlplane $ kubectl delete deployments nginx
deployment.apps "nginx" deleted
controlplane $
controlplane $ kubectl get deployments
No resources found in default namespace.
controlplane $
```

Deployment Creation through yaml file

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    run: nginx
  name: nginx-deploy
spec:
  replicas: 2
  selector:
    matchLabels:
      run: nginx
  template:
    metadata:
      labels:
        run: nginx
    spec:
      containers:
      - image: nginx
        name: nginx
```

$kubectl apply -f deployments.yaml

```
controlplane $ kubectl apply -f deployments.yaml
deployment.apps/nginx-deploy created
controlplane $
controlplane $
controlplane $ kubectl get deployments
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deploy   0/2     2            0           11s
controlplane $
```

```
controlplane $ kubectl get pods
NAME                            READY   STATUS    RESTARTS   AGE
nginx-deploy-598b589c46-mj57b   1/1     Running   0          61s
nginx-deploy-598b589c46-qkqsr   1/1     Running   0          61s
controlplane $
```

Deletion of Deployments

$ kubectl delete deployments nginx-deploy

```
controlplane $ kubectl delete deployments nginx-deploy
deployment.apps "nginx-deploy" deleted
controlplane $
controlplane $ kubectl get deployments
No resources found in default namespace.
controlplane $
```