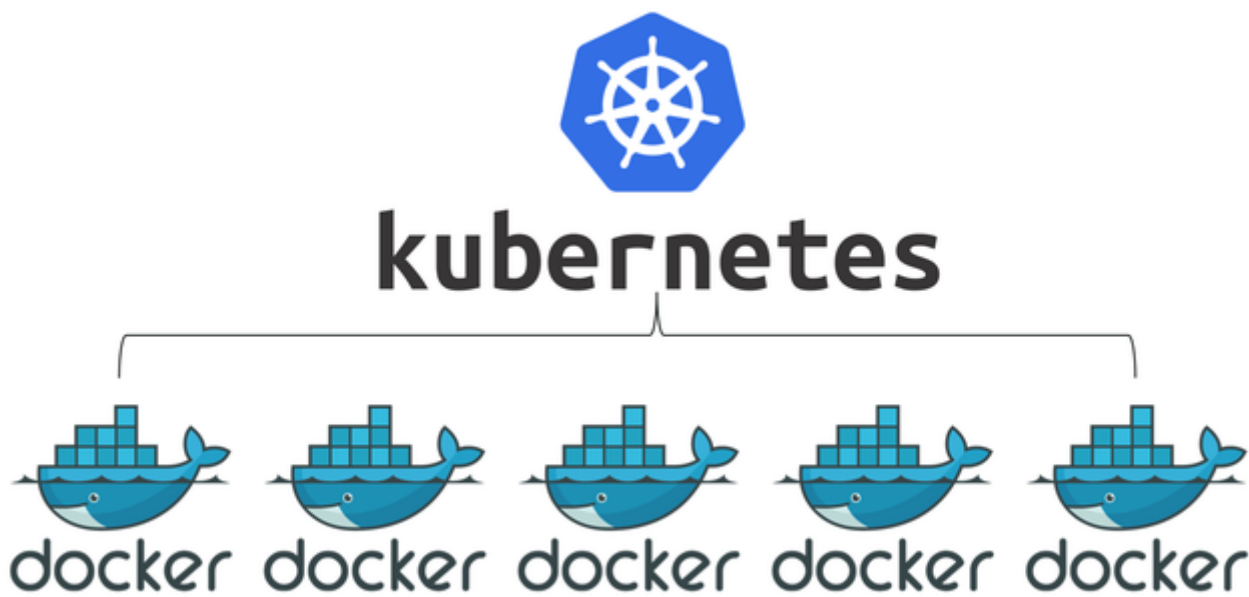


Lệnh cơ bản trong Kubernetes

Kubernetes là một Platform được áp dụng để tự động hoá việc quản lý, scaling(mở rộng) và triển khai ứng dụng dưới dạng container hoá cụ thể là các docker.



Dưới đây là các lệnh cơ bản trong Kubernetes.

1. Cấu hình kubectl

```
kubectl config view # Hiển thị các thiết lập kubeconfig đã được merged

# sử dụng nhiều tệp kubeconfig cùng một lúc và xem cấu hình hợp nhất
KUBECONFIG=~/.kube/config: ~/.kube/kubconfig2

kubectl config view

# lấy mật khẩu cho người dùng e2e
kubectl config view -o jsonpath='{.users[?(@.name == "e2e")].user.password}'

kubectl config view -o jsonpath='{.users[0].name}' # hiển thị người dùng đầu tiên
kubectl config view -o jsonpath='{.users[*].name}' # lấy danh sách người dùng
```

```

kubectl config get-contexts                # hiển thị danh sách các ngữ cảnh
kubectl config current-context             # hiển thị ngữ cảnh hiện tại
kubectl config use-context my-cluster-name # thiết lập ngữ cảnh mặc định cho my-cluster-name

# thêm một cụm mới vào kubeconf hỗ trợ xác thực cơ bản
kubectl config set-credentials kubeuser/foo.kubernetes.com --username=kubeuser --password=kubepassword

# lưu vĩnh viễn namespace cho tất cả các lệnh kubectl tiếp theo trong ngữ cảnh đó
kubectl config set-context --current --namespace=ggckad-s2

# thiết lập ngữ cảnh sử dụng tên người dùng và namespace cụ thể
kubectl config set-context gce --user=cluster-admin --namespace=foo \
    && kubectl config use-context gce

kubectl config unset users.foo              # xóa người dùng foo

```

2. Kubectl apply

`apply` quản lý các ứng dụng thông qua các tệp định nghĩa tài nguyên Kubernetes. Nó tạo và cập nhật các tài nguyên trong một cụm thông qua việc chạy `kubectl apply`. Đây là cách được đề xuất để quản lý các ứng dụng Kubernetes trong thực tế.

```

kubectl apply -f ./my-manifest.yaml        # tạo tài nguyên
kubectl apply -f ./my1.yaml -f ./my2.yaml # tạo từ nhiều tệp
kubectl apply -f ./dir                     # tạo tài nguyên từ tất cả các tệp manifest trong thư mục dir
kubectl apply -f https://git.io/vPieo      # tạo tài nguyên từ url
kubectl create deployment nginx --image=nginx # tạo một deployment nginx
kubectl explain pods,svc                   # lấy thông tin pod và service manifest

# Tạo nhiều đối tượng YAML từ stdin
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: busybox-sleep
spec:
  containers:
    - name: busybox

```

```

    image: busybox
    args:
    - sleep
    - "1000000"
---
apiVersion: v1
kind: Pod
metadata:
  name: busybox-sleep-less
spec:
  containers:
  - name: busybox
    image: busybox
    args:
    - sleep
    - "1000"
EOF

```

```

# Tạo một secret với một số keys
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  password: $(echo -n "s33msi4" | base64 -w0)
  username: $(echo -n "jane" | base64 -w0)
EOF

```

3. Xem, tìm các tài nguyên Pod, Node

# Lệnh get với một số đầu ra cơ bản	
kubectl get services	# Liệt kê tất cả các services trong namespace
kubectl get pods --all-namespaces	# Liệt kê tất cả các pods trong tất cả các namespaces
kubectl get pods -o wide	# Liệt kê tất cả các pods namespace, với nhiều thông tin hơn
kubectl get deployment my-dep	# Liệt kê một deployment cụ thể
kubectl get pods	# Liệt kê tất cả các pods trong namespace

```

kubectrl get pod my-pod -o yaml          # Lấy thông tin của một pod ở dạng YAML
kubectrl get pod my-pod -o yaml --export  # Lấy thông tin của một pod ở dạng YAML mà không có
thông tin cụ thể về cụm

# Lệnh describe
kubectrl describe nodes my-node
kubectrl describe pods my-pod

# Liệt kê các services được sắp xếp theo tên
kubectrl get services --sort-by=.metadata.name

# Liệt kê các pods được sắp xếp theo số lần khởi động lại
kubectrl get pods --sort-by='.status.containerStatuses[0].restartCount'

# Liệt kê các pods được sắp xếp theo dung lượng trong namespace có tên là test

kubectrl get pods -n test --sort-by='.spec.capacity.storage'

# Lấy thông tin phiên bản của tất cả các pods có nhãn app=cassandra
kubectrl get pods --selector=app=cassandra -o \
  jsonpath='{.items[*].metadata.labels.version}'

# Liệt kê tất cả các worker nodes (sử dụng một selector để loại trừ kết quả có một nhãn
# có tên 'node-role.kubernetes.io/master'
kubectrl get node --selector='!node-role.kubernetes.io/master'

# Liệt kê tất cả các pods đang chạy trong namespace
kubectrl get pods --field-selector=status.phase=Running

# Liệt kê tất cả các ExternalIPs của tất cả các nodes
kubectrl get nodes -o jsonpath='{.items[*].status.addresses[?(@.type=="ExternalIP")].address}'

# Liệt kê tên của các pods thuộc về một RC nhất định
# Lệnh "jq" hữu ích cho các chuyển đổi quá mức phức tạp cho jsonpath, xem thêm tại
https://stedolan.github.io/jq/
sel=${$(kubectrl get rc my-rc --output=json | jq -j '.spec.selector | to_entries | .[] |
"\(.key)=\(.value),"' )%?}
echo $(kubectrl get pods --selector=$sel --output=jsonpath='{.items..metadata.name}')

# Hiển thị nhãn của tất cả các pods (hoặc các đối tượng Kubernetes khác hỗ trợ gán nhãn)

```

```
kubectl get pods --show-labels
```

```
# Kiểm tra xem nodes nào đã sẵn sàng
```

```
JSONPATH='{range .items[*]}{@.metadata.name}:{range  
@.status.conditions[*]}{@.type}={@.status}};{end}{end}\' \\\n    && kubectl get nodes -o jsonpath="$JSONPATH" | grep "Ready=True"
```

```
# Liệt kê tất cả Secrets hiện đang sử dụng bởi một pod
```

```
kubectl get pods -o json | jq '.items[].spec.containers[].env[]?.valueFrom.secretKeyRef.name'  
| grep -v null | sort | uniq
```

```
# Liệt kê tất cả các sự kiện được sắp xếp theo thời gian
```

```
kubectl get events --sort-by=.metadata.creationTimestamp
```

4. Cập nhật các tài nguyên

Từ phiên bản 1.11, rolling-update đã không còn được dùng nữa, sử dụng rollout thay thế.

```
kubectl set image deployment/frontend www=image:v2          # Cập nhật container "www" của  
deployment "frontend", cập nhật image  
kubectl rollout history deployment/frontend                  # Kiểm tra lịch sử deployment  
bao gồm các sửa đổi  
kubectl rollout undo deployment/frontend                      # Quay trở lại deployment  
trước đó  
kubectl rollout undo deployment/frontend --to-revision=2     # Quay trở lại một phiên bản cụ  
thể  
kubectl rollout status -w deployment/frontend                # Xem trạng thái cập nhật của  
deployment "frontend" cho đến khi hoàn thành  
  
# không còn được sử dụng kể từ phiên bản 1.11  
kubectl rolling-update frontend-v1 -f frontend-v2.json      # (không dùng nữa) Cập nhật  
pods của frontend-v1  
kubectl rolling-update frontend-v1 frontend-v2 --image=image:v2 # (không dùng nữa) Đổi tên tài  
nguyên và cập nhật image  
kubectl rolling-update frontend --image=image:v2            # (không dùng nữa) Cập nhật  
image của pod của frontend  
kubectl rolling-update frontend-v1 frontend-v2 --rollback    # (không dùng nữa) Hồi bộ  
trình cập nhật hiện tại
```

```

cat pod.json | kubectl replace -f - # Thay thế một pod dựa trên
JSON được truy cập vào std

# Buộc thay thế, xóa và sau đó tạo lại tài nguyên, sẽ gây ra sự cố ngưng services
kubectl replace --force -f ./pod.json

# Tạo một services cho nginx, phục vụ trên cổng 80 và kết nối đến các container trên cổng 8000
kubectl expose rc nginx --port=80 --target-port=8000

# Cập nhật phiên bản image của một container đơn lẻ lên v4
kubectl get pod mypod -o yaml | sed 's/\(image: myimage\):.*$/\1: v4/' | kubectl replace -f -

kubectl label pods my-pod new-label=awesome # Thêm một nhãn
kubectl annotate pods my-pod icon-url=http://goo.gl/XXBTWq # Thêm một chú thích
kubectl autoscale deployment foo --min=2 --max=10 # Tự động scale deployment
"foo"

```

5. Cập nhật các tài nguyên

```

# Cập nhật một pod một node
kubectl patch node k8s-node-1 -p '{"spec":{"unschedulable":true}}'

# Cập nhật image của container; spec.containers[*].name là bắt buộc vì đó là khóa hợp lệ
kubectl patch pod valid-pod -p '{"spec":{"containers":[{"name":"kubernetes-serve-
hostname","image":"new image"}]}}'

# Cập nhật image của container sử dụng một bản vá json với các mảng vị trí
kubectl patch pod valid-pod --type='json' -p='[{"op": "replace", "path":
"/spec/containers/0/image", "value":"new image"}]'

# Vô hiệu hóa một deployment livenessProbe sử dụng một bản vá json với các mảng vị trí
kubectl patch deployment valid-deployment --type json -p='[{"op": "remove", "path":
"/spec/template/spec/containers/0/livenessProbe"}]'

# Thêm một pod mới vào một mảng vị trí
kubectl patch sa default --type='json' -p='[{"op": "add", "path": "/secrets/1", "value":
{"name": "whatever" } }]'

```

6. Chỉnh sửa các tài nguyên

Chỉnh sửa bất kỳ API tài nguyên nào trong trình soạn thảo ưa thích của bạn.

```
kubectl edit svc/docker-registry # Chỉnh sửa services có tên docker-registry
KUBE_EDITOR="nano" kubectl edit svc/docker-registry # Sử dụng một trình soạn thảo thay thế
```

7. Scaling tài nguyên

```
kubectl scale --replicas=3 rs/foo # Scale một replicaset có tên 'foo' thành 3
kubectl scale --replicas=3 -f foo.yaml # Scale một tài nguyên được xác định trong "foo.yaml" thành 3
kubectl scale --current-replicas=2 --replicas=3 deployment/mysql # Nếu kích thước hiện tại của deployment mysql là 2, scale mysql thành 3
kubectl scale --replicas=5 rc/foo rc/bar rc/baz # Scale nhiều replication controllers
```

8. Xóa tài nguyên

```
kubectl delete -f ./pod.json # Xóa một pod sử dụng loại và tên được xác định trong pod.json
kubectl delete pod,service baz foo # Xóa pods và services có tên "baz" và "foo"
kubectl delete pods,services -l name=myLabel # Xóa pods và services có nhãn name=myLabel
kubectl -n my-ns delete pod,svc --all # Xóa tất cả pods và services trong namespace my-ns,
# Xóa tất cả pods matching với pattern1 hoặc pattern2
kubectl get pods -n mynamespace --no-headers=true | awk '/pattern1|pattern2/{print $1}' |
xargs kubectl delete -n mynamespace pod
```

9. Xem log với các pods đang chạy

```
kubectl logs my-pod # Kiểm xuất logs của pod (stdout)
kubectl logs -l name=myLabel # Kiểm xuất logs của pod có nhãn name=myLabel (stdout)
kubectl logs my-pod --previous # Kiểm xuất logs của pod (stdout) cho khi tạo trước của một container
kubectl logs my-pod -c my-container # Kiểm xuất logs của container của pod (stdout, trường hợp có nhiều container)
kubectl logs -l name=myLabel -c my-container # Kiểm xuất logs của container có tên my-container và có nhãn name=myLabel (stdout)
kubectl logs my-pod -c my-container --previous # Kiểm xuất logs của container my-container của pod my-pod (stdout, trường hợp có nhiều container) cho khi tạo trước của một container
kubectl logs -f my-pod # Lắng logs của pod my-pod (stdout)
kubectl logs -f my-pod -c my-container # Lắng logs của container my-container trong pod my-pod (stdout, trường hợp nhiều container)
kubectl logs -f -l name=myLabel --all-containers # Lắng logs của tất cả các container của pod có nhãn name=myLabel (stdout)
```

10. Tương tác với các Pod đang chạy

```
kubectl run -i --tty busybox --image=busybox -- sh # Chạy pod trong một shell tương tác
kubectl run nginx --image=nginx --restart=Never -n
mynamespace # Chạy pod nginx trong một namespace cụ thể
kubectl run nginx --image=nginx --restart=Never # Chạy pod nginx và ghi spec của nó vào file có
tên pod.yaml
--dry-run -o yaml > pod.yaml

kubectl attach my-pod -i # Đính kèm với container đang chạy
kubectl port-forward my-pod 5000:6000 # Lắng nghe trên cổng 5000 của máy local và chuyển tiếp sang
cổng 6000 trên pod my-pod
kubectl exec my-pod -- ls / # Chạy lệnh trong một pod (trường hợp 1 container)
kubectl exec my-pod -c my-container -- ls / # Chạy lệnh trong pod (trường hợp nhiều container)
kubectl top pod POD_NAME --containers # Hiển thị số liệu của pod và container chạy trong nó
```

11. Tương tác với các node và cụm cluster

```
kubectl cordon my-node # Đánh dấu my-node là không thể lập lịch
kubectl drain my-node # Gỡ my-node ra khỏi cụm để chuẩn bị cho việc bảo trì
kubectl uncordon my-node # Đánh dấu my-node có thể lập lịch trở lại
kubectl top node my-node # Hiển thị số liệu của node
kubectl cluster-info # Hiển thị địa chỉ master và các services
kubectl cluster-info dump # Kịch xuất trạng thái hiện tại của cụm ra ngoài stdout
kubectl cluster-info dump --output-directory=/path/to/cluster-state # Kịch xuất trạng thái hiện
tại của cụm vào /path/to/cluster-state

kubectl taint nodes foo dedicated=special-user:NoSchedule
```

12. Các loại tài nguyên

Liệt kê tất cả các loại tài nguyên được hỗ trợ cùng với tên viết tắt của chúng, API group, cho dù chúng là namespaced, và Kind:

```
kubectl api-resources
```

Các hoạt động khác để khám phá các tài nguyên API:

```
kubectl api-resources --namespaced=true # Liệt kê các tài nguyên được đặt tên
kubectl api-resources --namespaced=false # Liệt kê các tài nguyên không được đặt tên
kubectl api-resources -o name # Liệt kê các tài nguyên với đầu ra đơn giản (chỉ gồm tên tài nguyên)
kubectl api-resources -o wide # Liệt kê các tài nguyên với đầu ra mở rộng
```

```
kubectl api-resources --verbs=list,get # Tắt c̣ các tài nguyên ḥ trợ yêu c̣ "list" và "get"  
kubectl api-resources --api-group=extensions # Tắt c̣ tài nguyên trong nhóm API "tiện ích ṃ  
rộng"
```

Revision #2

Created 20 February 2022 15:42:13 by Laptrinh.vn

Updated 21 February 2022 14:25:00 by Laptrinh.vn