



kubernetes

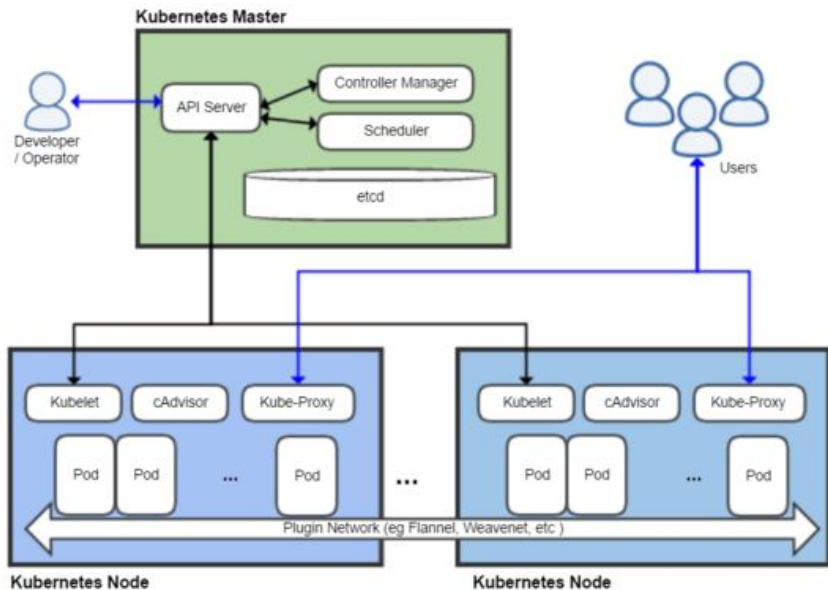


Amazon EKS

Kubernetes (K8s) 簡介

Kubernetes由三大元件組成Cluster:

- Pod
- Worker Node
- Master Node



參考: [維基百科](#)

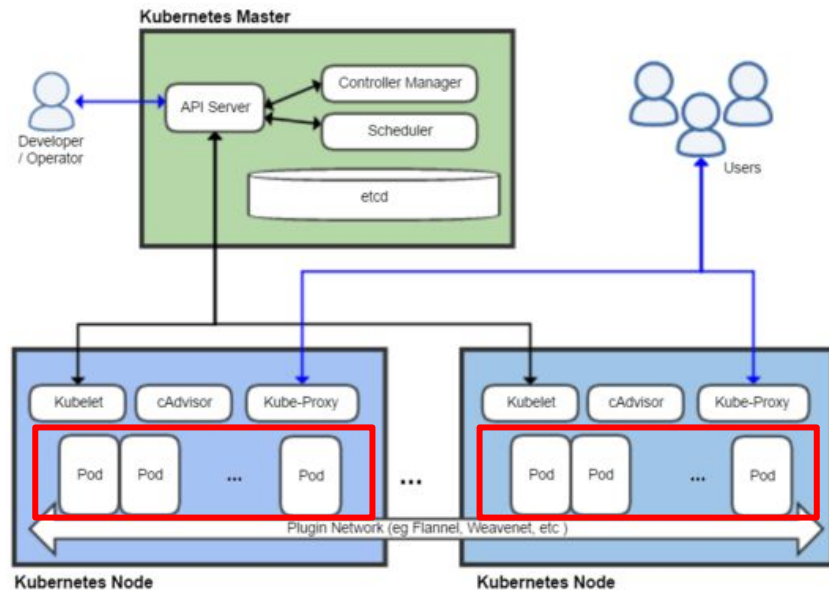
Kubernetes (K8s) 簡介

Pod

Pod是Kubernetes中最小的運作單位，每個Pod都會對應到一個應用程式。

一個Pod通常存在一個或以上的容器

同一個Pod中的容器網路互通，透過內部Port進行溝通



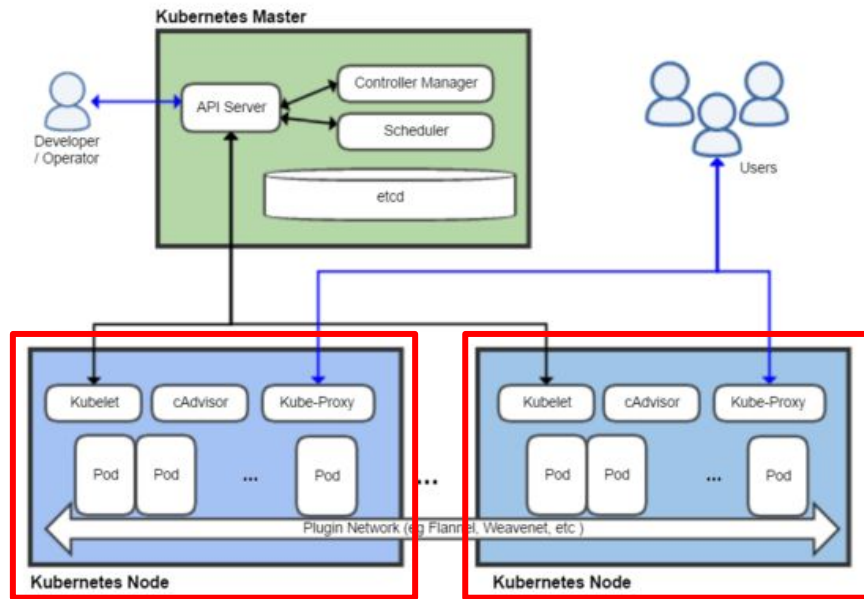
參考：[維基百科](#)

Kubernetes (K8s) 簡介

Worker Node

一個Node對應到的是一台實體或是虛擬機器，每個Node中都還有三個元件。

- kubelet：負責管理所有Node中的Pod狀態及Master Node溝通
- kube-proxy：執行簡單的網路轉發 (TCP、UDP、SCTP)
- cAdvisor：監視容器的CPU、記憶體、檔案、網路等的資源使用情況



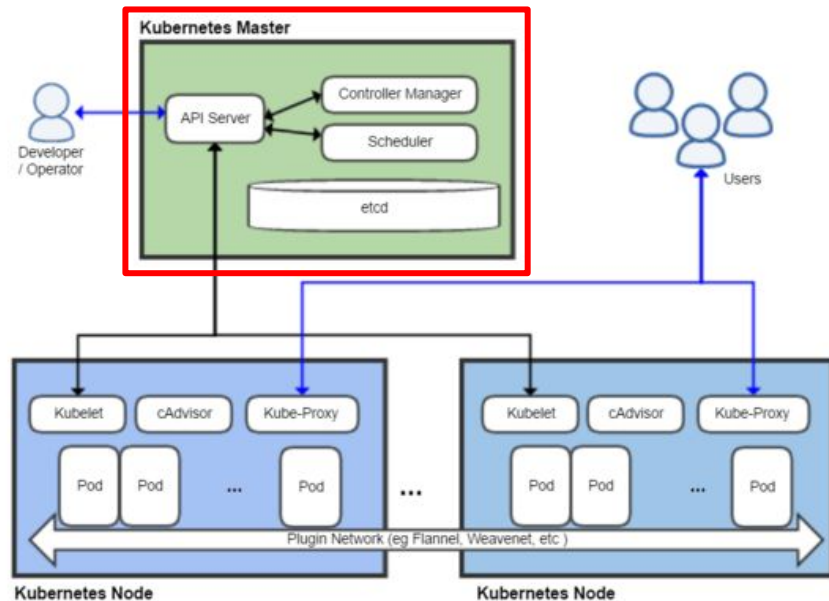
參考：[維基百科](#)

Kubernetes (K8s) 簡介

Master Node

Master Node 是Kubernetes的指揮中心，負責管理所有Node，其中有四個元件：

- kube-apiserver：作為Node之間溝通的橋樑，提供請求的身分認證與授權
- etcd：整個Kubernetes備份中心，在Master發生異常時可還原至正常狀態

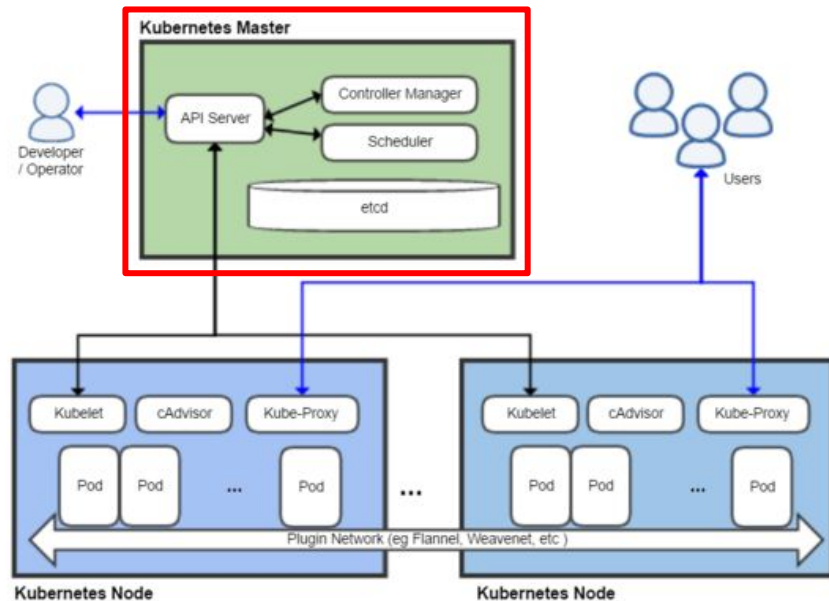


參考：[維基百科](#)

Kubernetes (K8s) 簡介

Master Node

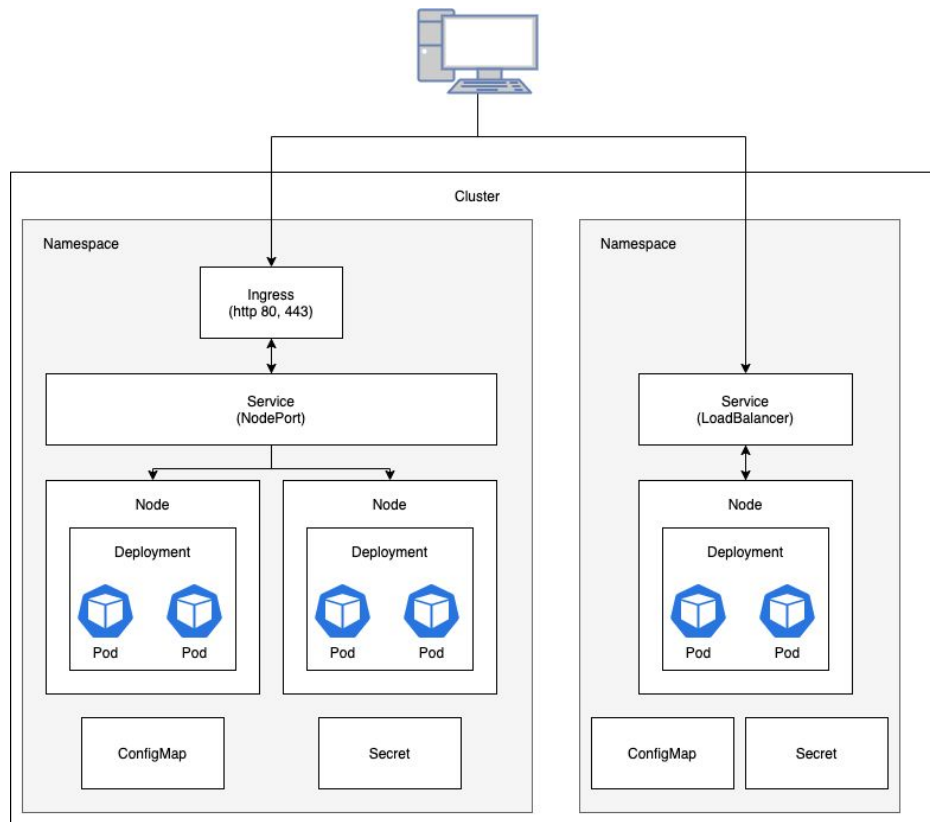
- kube-scheduler：負責將新建立的Pod分派到適合的Node中，其根據Node的規則與硬體限制條件等資訊來進行
- kube-controller-manager：管理所有處理程序、監控資源使用，包含Node數量增減、處理程序指派等



參考：[維基百科](#)

Kubernetes 常見 kind

1. Namespace
2. ConfigMap
3. Secret
4. Deployment
5. Service
6. Ingress
7. HorizontalPodAutoscaler



環境安裝

- Kubernetes
 - minikube [[官方文件](#)]
 - kubectl [[官方文件](#)] [[AWS安裝文件](#)]
 - helm [[官方文件](#)]
- Visual Studio Code
 - Kubernetes [[marketplace](#)]
 - Kubernetes Support [[marketplace](#)]

環境安裝

- AWS
 - AWS CLI [[官方文件](#)]
 - eksctl [[官方文件](#)]

DEMO



<https://github.com/xtony77/k8s-demo>

minikube 常用指令

```
minikube start --vm=true
```

```
minikube dashboard
```

```
minikube stop
```

```
minikube delete
```

```
minikube addons list
```

```
minikube addons enable ingress
```

```
minikube addons enable metrics-server
```

```
minikube tunnel
```

kubectl 常用指令

```
kubectl get all -n <namespace-name>
```

```
kubectl apply -f <yaml-file-path>
```

```
kubectl delete <pod,svc,ingress,node> <kind-name>
```

```
kubectl get pod <pod-name> -n <namespace-name>
```

```
kubectl get pod <pod-name> -o wide
```

```
kubectl get deploy <deploy-name> -o yaml
```

```
kubectl exec -ti <pod-name> -- bash
```

```
kubectl logs <pod-name> -f --tail 10 -n <namespace-name>
```

kubectl 常用指令

```
kubectl get deploy
```

```
kubectl get svc
```

```
kubectl get configmap
```

```
kubectl get secret
```

```
kubectl get hpa
```

```
kubectl get ns
```

```
kubectl describe <kind> <kind-name>
```

kubectl 常用指令

```
kubectl config get-contexts
```

```
kubectl config get-contexts <context-name>
```

```
kubectl config use-context <context-name>
```

```
kubectl config set-context <context-name> --namespace=<namespace-name>
```

```
kubectl rollout history <kind> <kind-name>
```

```
kubectl rollout undo <kind> <kind-name> --to-revision=<REVISION>
```

helm 常用指令

```
helm create <chart-name>
```

```
helm install <chart-name> .
```

```
helm list -A
```

```
helm upgrade <chart-name> . --set-string releaseVersion=20201114
```

```
helm delete <chart-name>
```

```
helm history <release-name>
```

```
helm rollback <release-name> <REVISION>
```

AWS EKS 環境建立

- IAM 建立 user
 - 建立存取金鑰
 - Permissions policies 許可新增 AmazonEKSClusterPolicy

```
aws configure
```

```
~/ aws configure
AWS Access Key ID [None]: AKIA3F7CMMI0VVYMZ4SK
AWS Secret Access Key [None]: 
Default region name [None]: ap-northeast-1
Default output format [None]: json
```

```
aws ec2 create-volume --availability-zone=ap-northeast-1a --size=10 --volume-type=gp2
```


AWS EKS 環境建立

```
eksctl create cluster \  
--name <my-cluster> \  
--version <1.18> \  
--region <ap-northeast-1> \  
--fargate
```

AWS EKS 環境建立

```
eksctl create cluster \  
--name <my-cluster> \  
--version <1.18> \  
--region <ap-northeast-1> \  
--node-type <m5.large,t2.micro> \  
--nodegroup-name <linux-nodes> \  
--nodes <2> \  
--nodes-min <2> \  
--nodes-max <3> \  
--asg-access \  
--managed
```

AWS EKS 環境建立

```
eksctl get cluster
```

```
eksctl delete cluster
```

AWS EKS 環境建立

- IAM
- Security Groups
- EC2 auto scaling 策略
- EC2 Volume

Q&A