Kubernetes Install Manual

202131333 컴퓨터공학과 정원준

TABLE OF CONTENTS

01

What is Kubernetes?

컨테이너는 뭐지??

쿠버네티스

Kubernetes, also known as K8s, is an open source system for managing containerized applications across multiple hosts. It provides basic mechanisms for deployment, maintenance, and scaling of applications.

Made by Google Engieneers

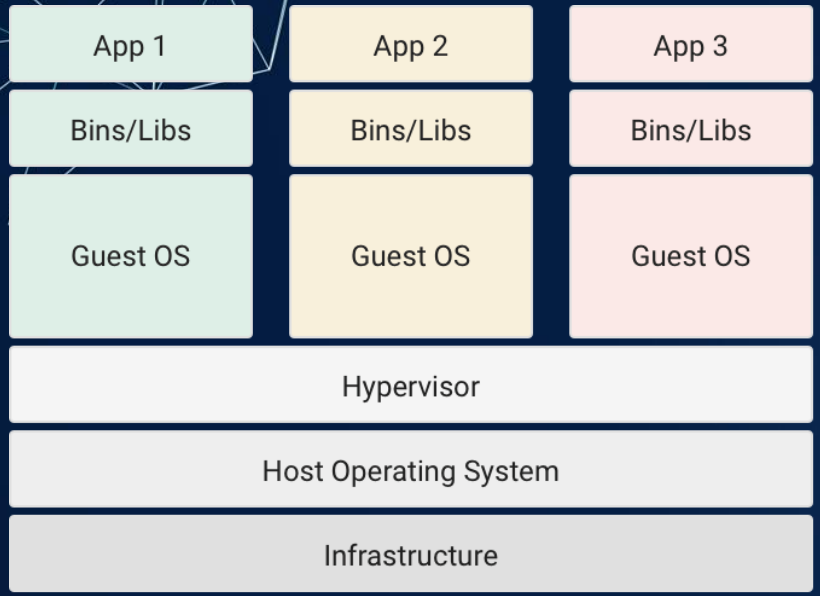
Difference between VM and Container

VM은 자체 CPU, 메모리, 네트워크 인터페이스, 스토리지가 있는 가상 컴퓨터 시스템처럼 기능하는 가상 환경으로, 물리적 하드웨어 시스템에 생성된다.

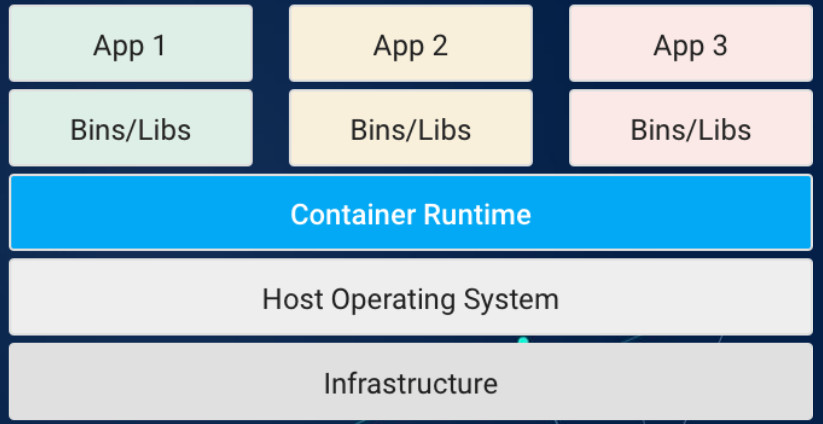
Host OS위에 Guest OS를 가상화하는 방식

VM같은 방식은 무겁고 성능 문제가 발생하기에 프로세스를 격리하는 방안

하드웨어 자원들을 cgroups을 통해 할당하고 namespace를 활용하여 프로세스나 IPC 등을 격리

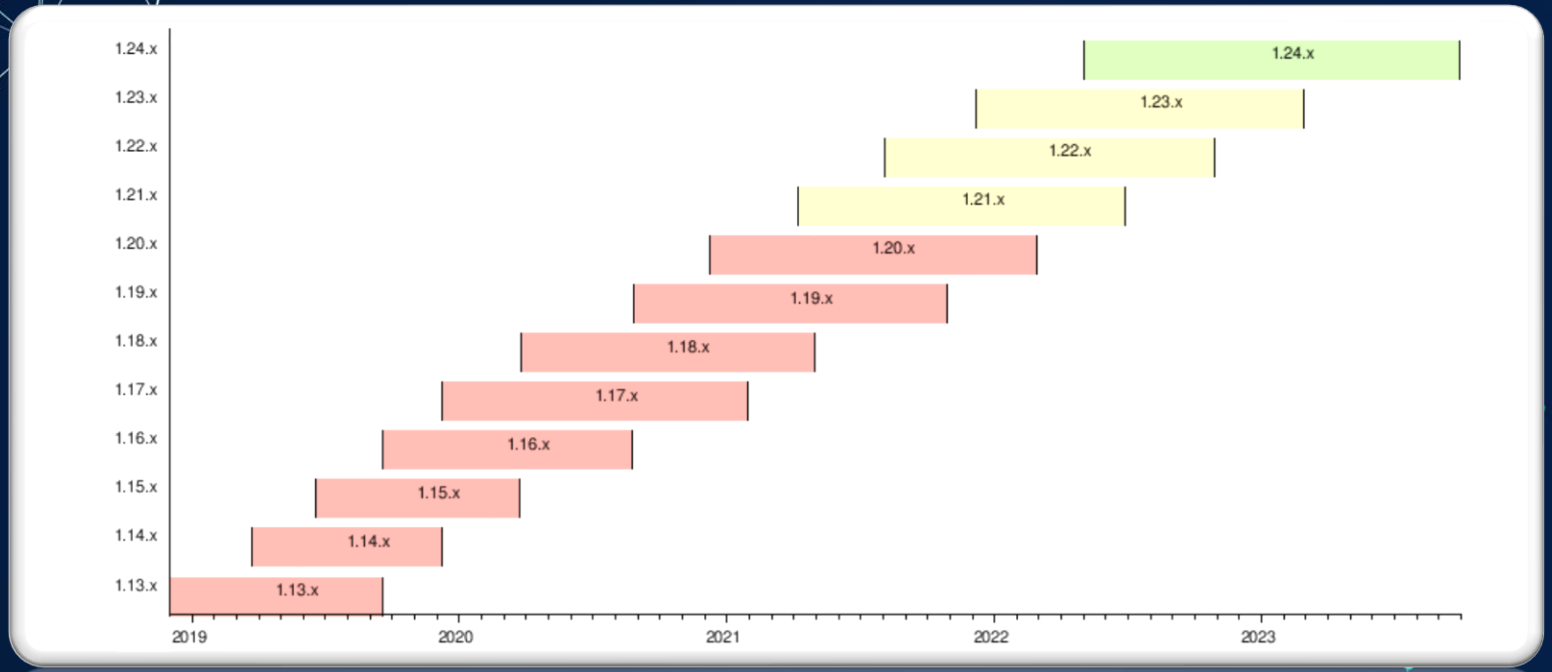


VM



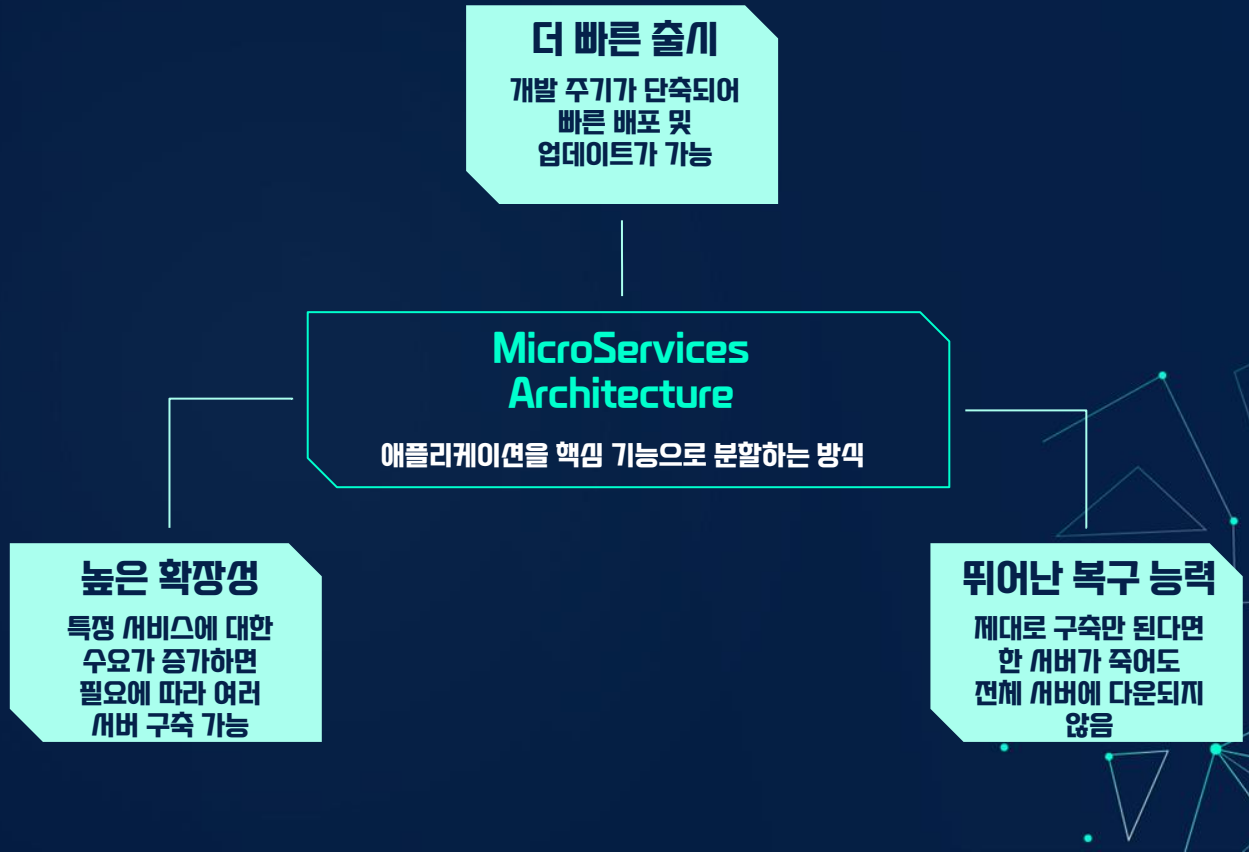
Container

Support Version




02
Why use Kubernetes?
Microservice Architecture를 위해선 꼭 필요하겠는걸??

Microservices Architecture



Why use Kubernetes?

서비스 디스커버리와 로드 밸런싱

DNS 이름을 사용하거나 자체 IP 주소를 사용하여 컨테이너를 노출할 수 있음.
또한, 트래픽이 많으면 로드밸런싱을 해준다.

쿠버네티스를 자동화하여 배포용 새 컨테이너를 만들고, 기존 컨테이너를 제거하고, 모든 리소스를 새 컨테이너에 적용

자동화된 롤아웃과 롤백

자동화된 복구

실패한 컨테이너를 다시 시작하고, 컨테이너를 교체하며, 서비스 준비가 끝날 때까지 그러한 과정을 클라이언트에게 보여주지 않음


03
Install
Docker and Golang
쿠버네티스는 Docker와 Golang이 필요구나!

Environment Information

- HostOS : Windows 11
- VM : Vmware Workstation 16 Player (Non-commercial)
- GuestOS : Ubuntu 20.04.4-desktop-amd64

Repository Setting
before install docker

```
$) sudo apt-get update

$) sudo apt-get install -y \
        apt-transport-https \
        ca-certificates \
        curl \
        gnupg \
        lsb-release
```

→ install의 y 옵션 : 설치시 Y/N 물어보는 것을 자동으로 Y를 눌러줌

다운로드 받은 gpg 파일을 압축해제 ↓

```
 $) curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
```

↓ x86_64 또는 amd64면 그대로, armhf계열일 경우 armhf를 적어줘야함

```
$)  echo \
  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Docker Engine 설치하기

$⟩ sudo apt-get update → 아까 해줬다고 안하면 안되고 다시 쳐줘야함.

$⟩ sudo apt-get install docker-ce docker-ce-cli containerd.io

$⟩ sudo service docker start

설치 확인

```
pongchi@ubuntu:~$ sudo docker --version
Docker version 20.10.16, build aa7e414
```

Golang 설치하기

```
$) sudo wget  https://go.dev/dl/go1.18.3.linux-amd64.tar.gz

$) sudo rm -rf /usr/local/go && sudo tar -C /usr/local -xzf go1.18.3.linux-amd64.tar.gz

$) echo 'export PATH=${PATH}:/usr/local/go/bin'  >> ~/.bashrc
$) echo 'export GOPATH_K8S=${HOME}/go/src/k8s.io/kubernetes' >> ~/.bashrc
$) echo 'export PATH=${GOPATH_K8S}/third_party/etcd:${PATH}' >> ~/.bashrc
$) source ~/.bashrc
```

설치 확인

```
pongchi@ubuntu:~$ go version
go version go1.18.3 linux/amd64
```

04

Install Kubernetes

날 따라해봐요 요로케~

설치하는 2가지 방법

1. https://github.com/Kubernetes/kubernetes 에서 파일을 받아와 make 하기.

2. apt-get 또는 curl로 설치하기

Make로 Kubernetes 설치하기

```
$) systemctl enable docker
$) systemctl start docker

$) git clone https://github.com/kubernetes/kubernetes ${GOPATH_K8S}

$) cd ${GOPATH_K8S}

$) git checkout v1.23.7

$) hack/install-etcd.sh

$) sudo apt-get install make

$) sudo time make quick-release
```

→ time은 명령어 시간을 재는 용으로, 안해도 상관없음

Curl로 Kubernete 설치하기

```
$) curl -LO https://dl.k8s.io/release/v1.23.7/bin/linux/amd64/kebectl
$) curl -LO https://dl.k8s.io/release/v1.23.7/bin/linux/amd64/kebeadm
$) curl -LO https://dl.k8s.io/release/v1.23.7/bin/linux/amd64/kubelet

$) sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
$) sudo install -o root -g root -m 0755 kubeadm /usr/local/bin/kubeadm
$) sudo install -o root -g root -m 0755 kubelet /usr/local/bin/kubelet
```

Kubernetes 설치 확인

```
pongchi@ubuntu:~$ kubectl version --client
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.7", GitCo
mmit:"42c05a547468804b2053ecf60a3bd15560362fc2", GitTreeState:"clean", BuildDat
e:"2022-05-24T12:30:55Z", GoVersion:"go1.17.10", Compiler:"gc", Platform:"linux
/amd64"}
pongchi@ubuntu:~/Downloads$ kubelet --version
Kubernetes v1.23.7
pongchi@ubuntu:~$ kubeadm version
kubeadm version: &version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.7", Git
Commit:"42c05a547468804b2053ecf60a3bd15560362fc2", GitTreeState:"clean", BuildD
ate:"2022-05-24T12:29:44Z", GoVersion:"go1.17.10", Compiler:"gc", Platform:"lin
ux/amd64"}
```

05

How to use?

쓰는 방법이 생당히 어려운걸..?

Kubernetes의 기본 명령어

1. kubeadm : 클러스터를 부트스트랩하는 명령어

2. kublet : 클러스터의 모든 머신에서 실행되는 파드와
컨테이너 시작과 같은 작업을 수행하는 컴포넌트

3. kubectl : 클러스터와 통신하기 위한 커맨드 라인 유틸리티

사용법

1. 클러스터를 구축하기 위해 두 개의 노드를 준비해 마스터와 워커 노드로 나눠야 하지만 복잡하기에 Minikube라는 것을 이용해보자.

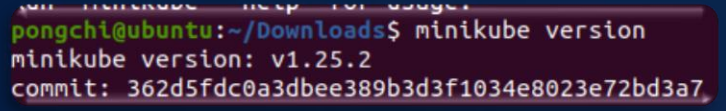
2. Minikube란 쿠버네티스의 클러스터 구축 과정을 대폭 줄이고, 가능한 하나의 단일 로컬 환경에서 쉽게 쿠버네티스를 체험해보기 위해 개발된 것.

Minikube 설치하기

$) curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

$) sudo install minikube-linux-amd64 /usr/local/bin/minikube

Minikube 설치 확인



Minikube 사용하기

$⟩ minikube start
$⟩ minikube addons enable dashboard → CLI 가 불편한 사람들을 위한 GUI. (필수X)

```
ongchi@ubuntu:~/Downloads$ minikube start
😄  minikube v1.25.2 on Ubuntu 20.04
✨  Automatically selected the docker driver. Other choices: none, ssh
👍  Starting control plane node minikube in cluster minikube
🚜  Pulling base image ...
💾  Downloading Kubernetes v1.23.3 preload ...
    > gcr.io/k8s-minikube/kicbase: 379.06 MiB / 379.06 MiB  100.00% 3.94 MiB p/
    > preloaded-images-k8s-v17-v1...: 505.68 MiB / 505.68 MiB  100.00% 5.19 MiB
🔥  Creating docker container (CPUs=2, Memory=2200MB) ...
🐳  Preparing Kubernetes v1.23.3 on Docker 20.10.12 ...
    ▪ kubelet.housekeeping-interval=5m
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎  Verifying Kubernetes components...
    Executing "docker container inspect minikube --format={{.State.Status}}" took an unusu
ally long time: 6.747228196s
💡  Restarting the docker service may improve performance.
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by d
efault
```

Minikube 사용하기

↓ 튜토리얼을 위한 쿠버네티스의 공식 이미지

$> kubectl create deployment hello-node2 --image=k8s.gcr.io/echoserver:1.4

$> kubectl expose deployment hello-node2 --type=LoadBalancer --port=8080

```
pongchi@ubuntu:~$ kubectl get services
NAME          TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hello-node2   LoadBalancer   10.99.220.231   <pending>     8080:32152/TCP   3m39
s
kubernetes    ClusterIP      10.96.0.1       <none>        443/TCP          58m
```

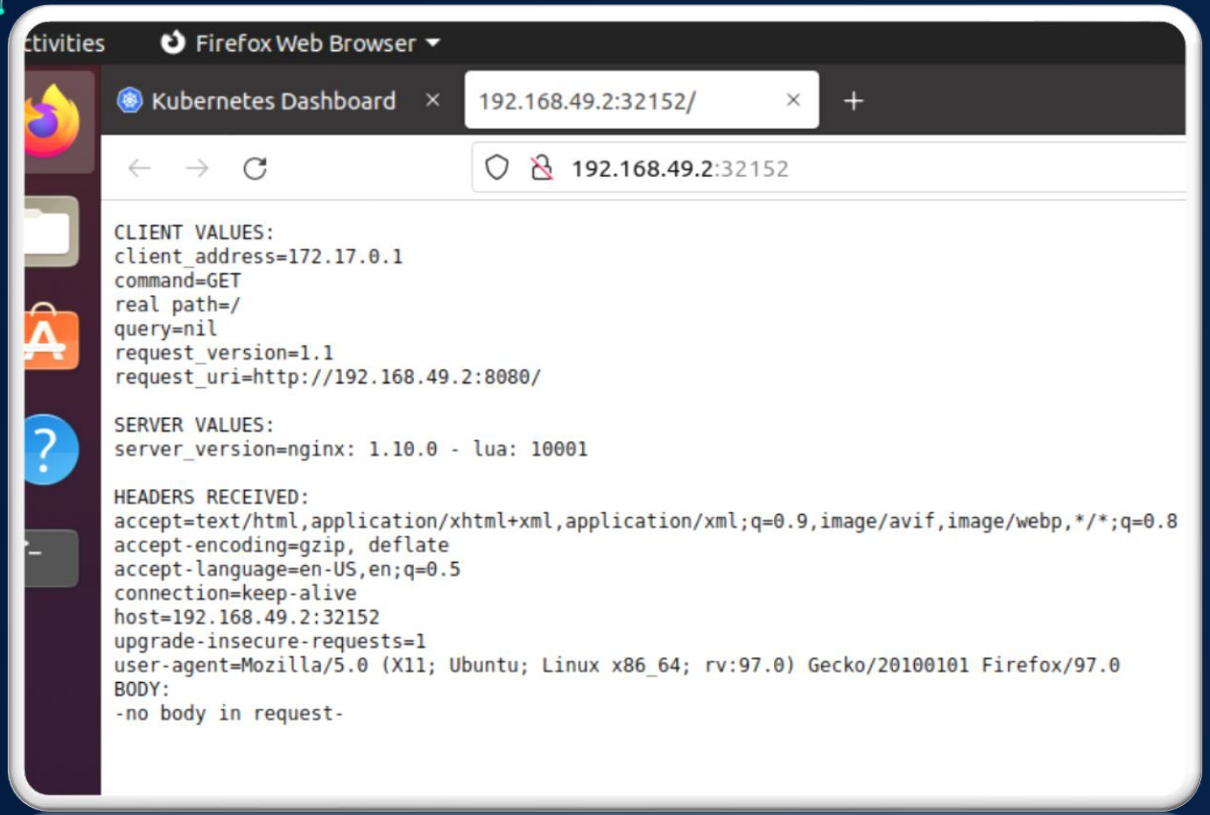
(CLUSTER-IP에 hello-node2가 IP 주소를 할당되어 있는 것을 볼 수 있음. EXTERNAL-IP 항목에는 외부 IP가 할당되는데, pending이라고 뜨는 이유는 현재 외부 IP를 할당 받지 못했다는 뜻인데, 실제 K8s 환경에서는 되지만 minikube 환경에서는 추가적인 명령을 통해 외부 IP를 할당할 수 있음.)

$> minikube service hello-node2 -url

```
pongchi@ubuntu:~$ minikube service hello-node2 --url
http://192.168.49.2:32152
```

(접속)

Minikube 사용하기



06

Q & A

질문 받습니다!

THANKS

Does anyone have any questions?

[Resource]

- https://github.com/Kubernetes/kubernetes
- https://www.redhat.com/ko/topics/containers/what-is-kubernetes
- https://dev.to/rahulku48837211/how-to-build-and-run-k8s-locally-5e3m