

**2<sup>nd</sup>**  
**Week**

# 두번째 뵙겠습니다 ?!

▷ 잠시만 기다렸다가 30분 되면 시작하겠습니다~^^

▷ 출석 체크도 한 번 해보시면 어떠세요?!

- <https://modulabs.co.kr/> : 마이페이지 → 참여한 랩·풀잎 → 자세히 보기 → 내 풀잎스쿨 출석 확인하기

▷ Camera는 가급적 켜 주시면 대단히 감사하겠습니다 !!!

- 너무 부끄러우면 Snap Camera를 사용하시는 것까지는~ ^^

▷ 오늘 수업 자료는 아래 링크에서 다운로드 받으실 수 있어요.

- <https://github.com/whatwant-school/kubernetes>



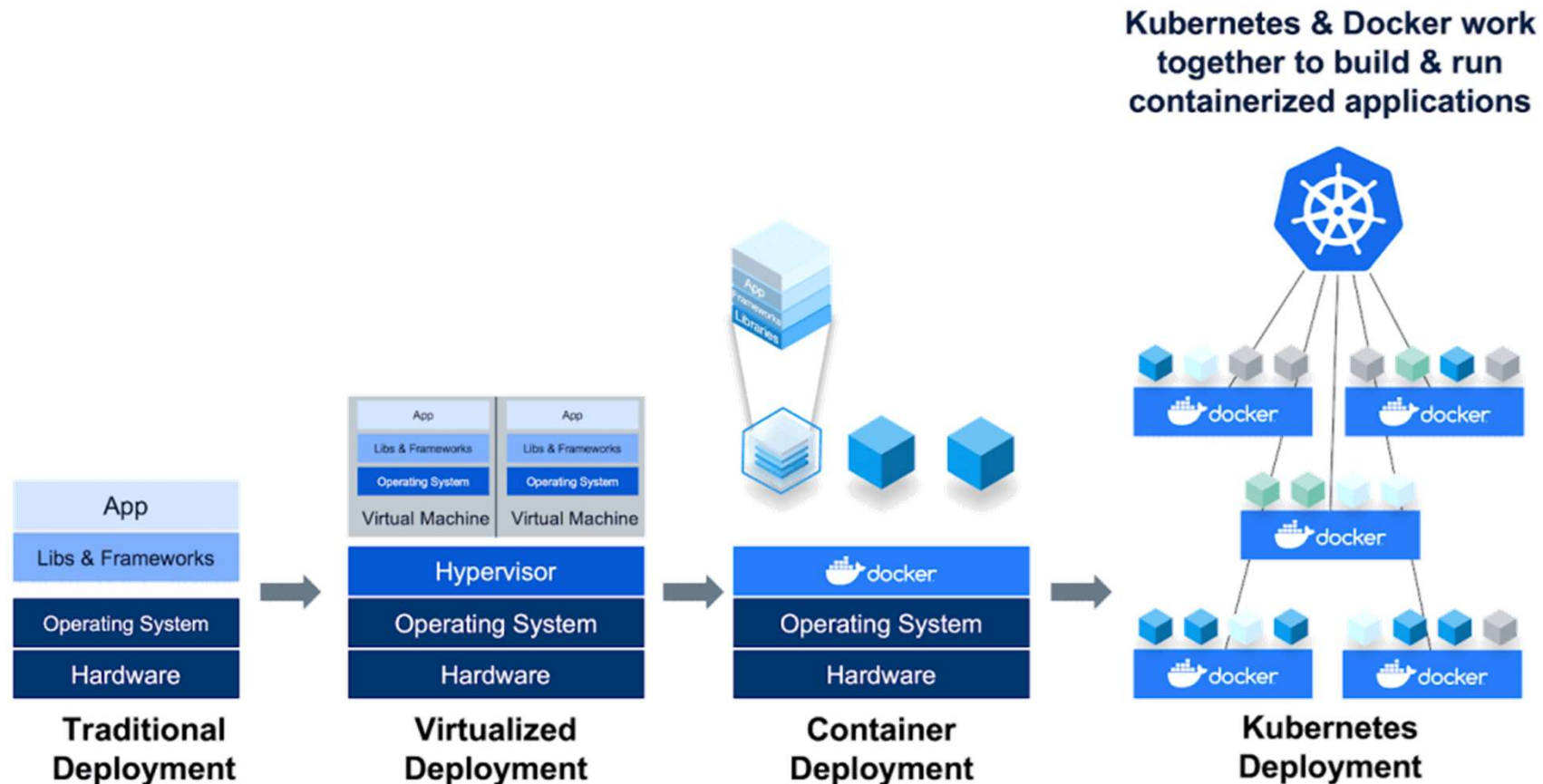
# Kubernetes Overview

# Kubernetes is ...

**Kubernetes**, also known as **K8s**,  
is an open-source system for  
automating deployment, scaling, and management  
of containerized applications.

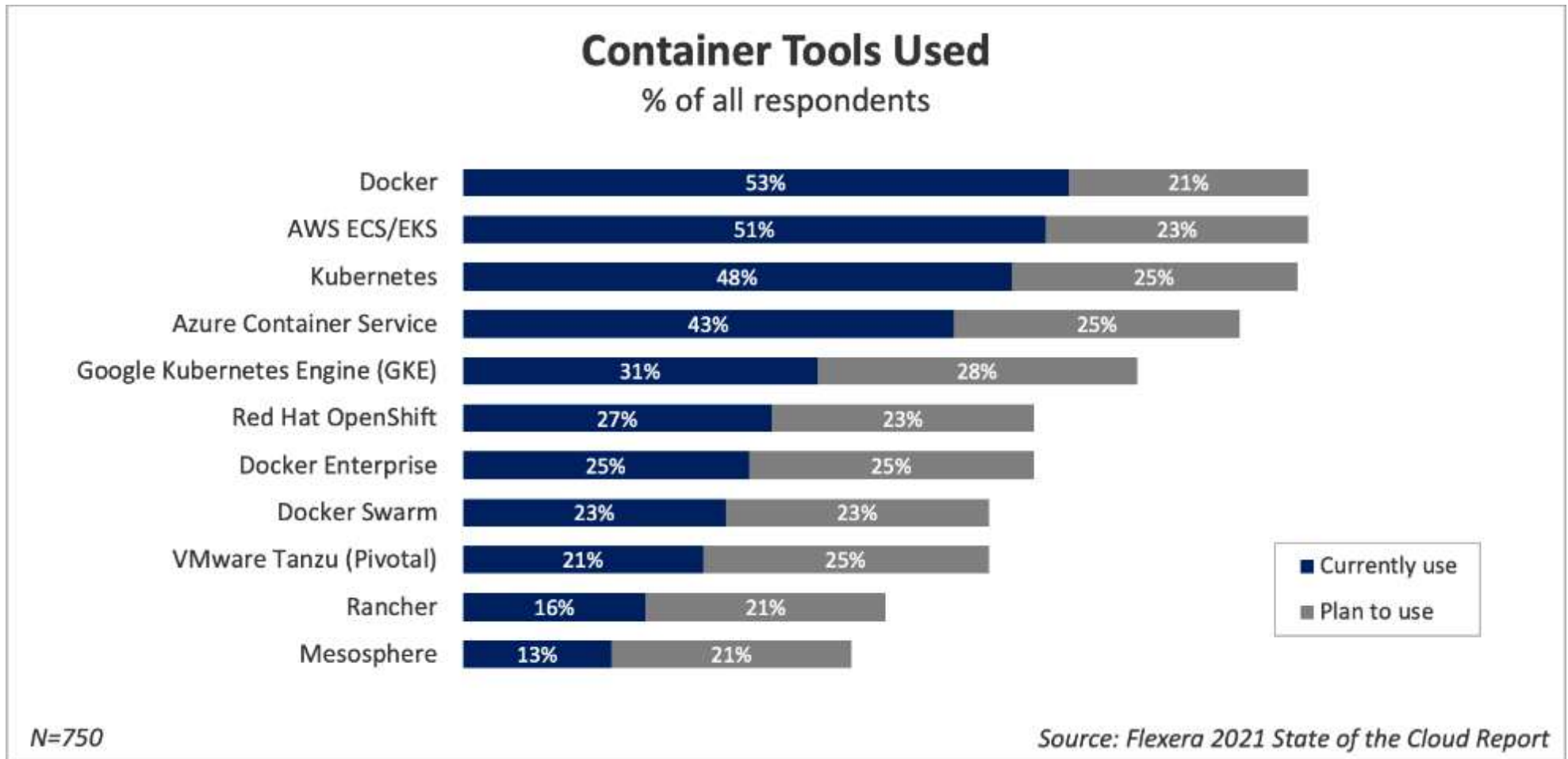
※ 참고 : <https://kubernetes.io/>

# Why Kubernetes ... ?



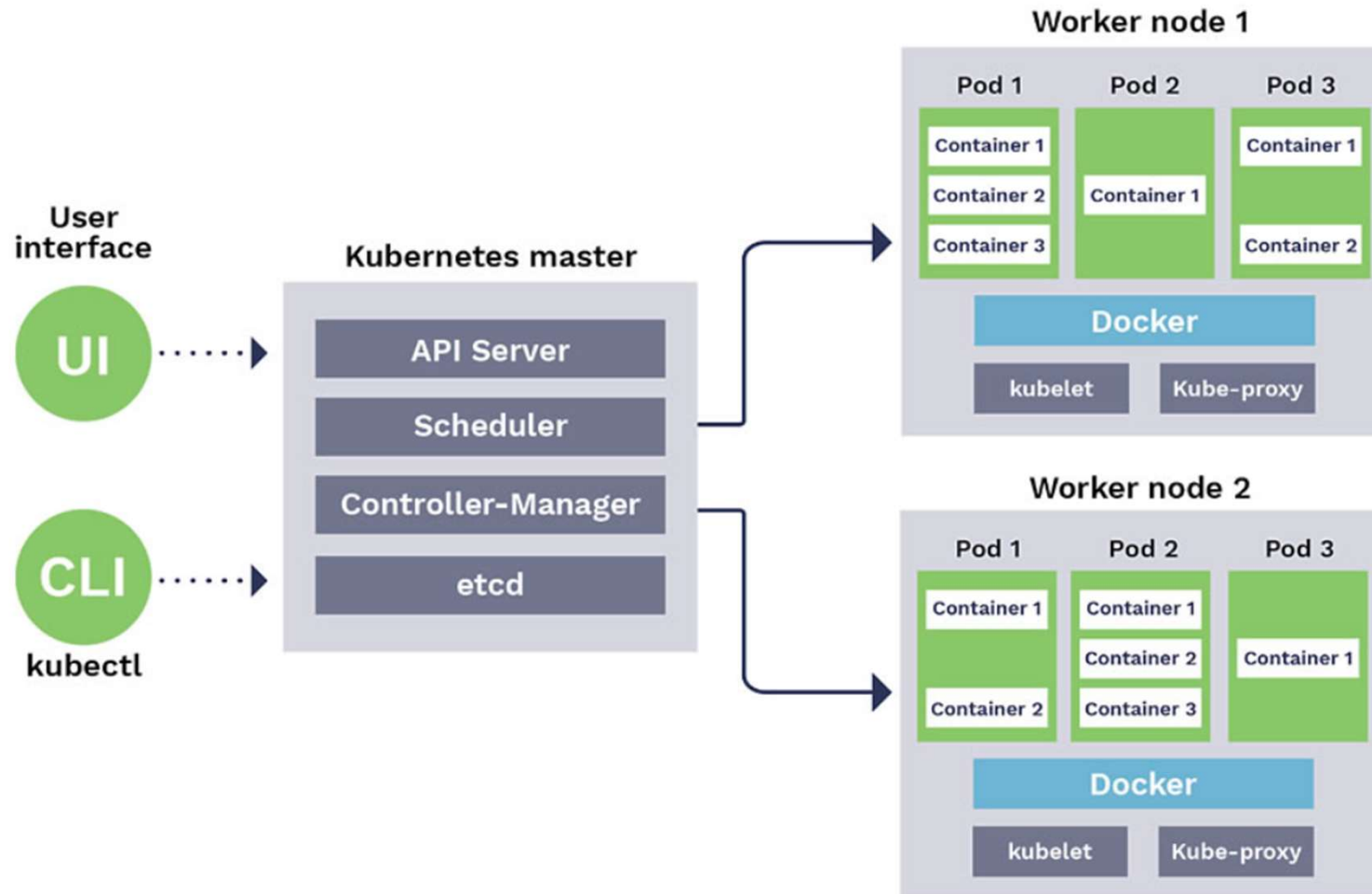
※ 참고 : <https://www.docker.com/blog/top-questions-docker-kubernetes-competitors-or-together/>

# Market Share



※ 참고 : <https://blog.flant.com/kubernetes-and-containers-market-trends-2021/>

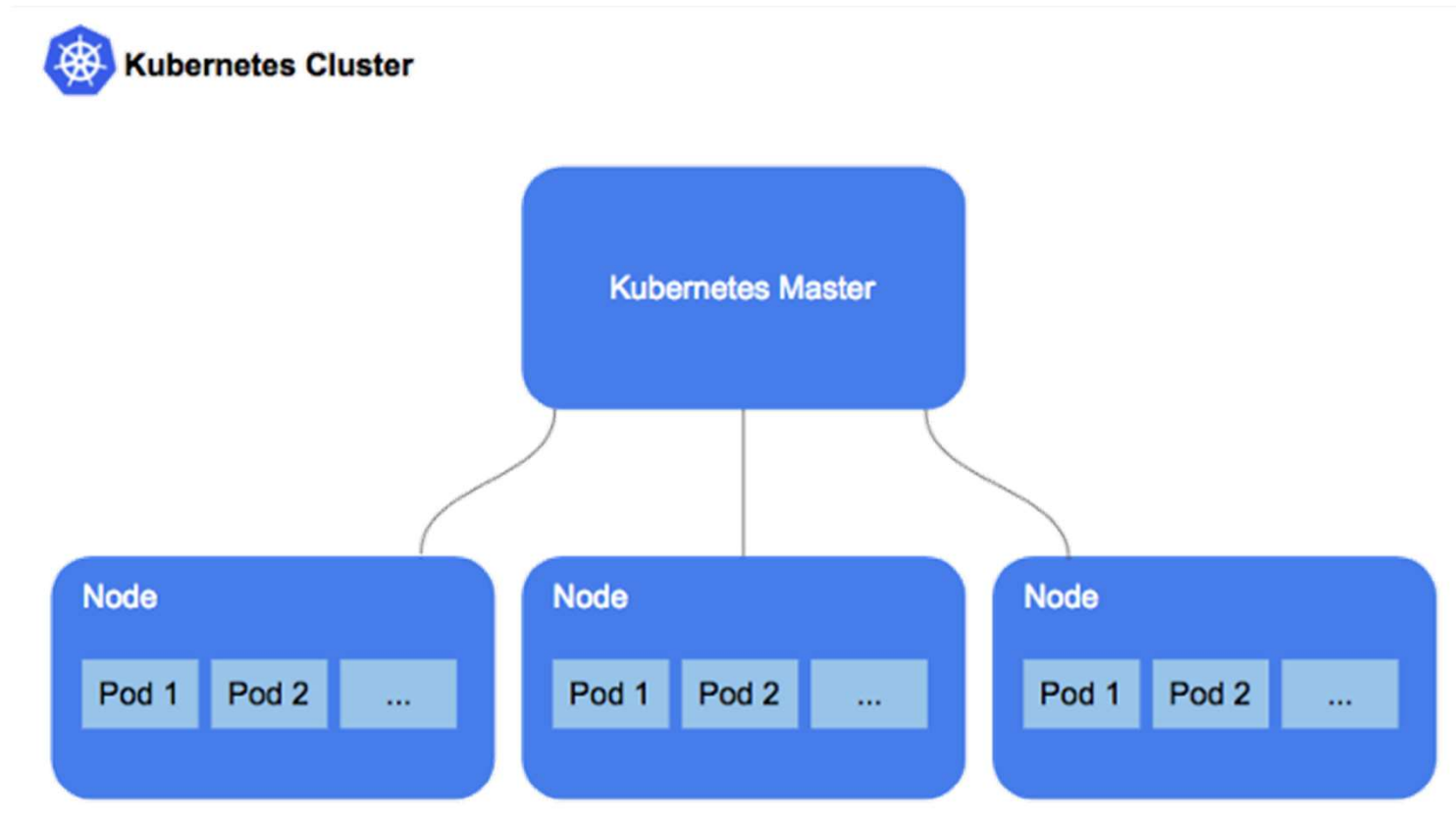
# Kubernetes Architecture



※ 참고 : <https://keetmalin.wixsite.com/keetmalin/post/understanding-container-orchestration-with-kubernetes>

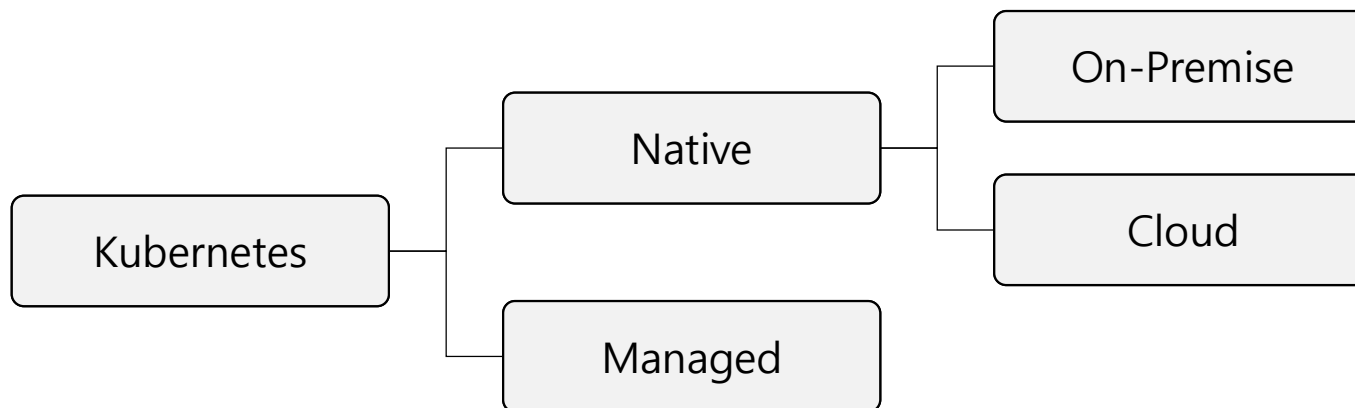


# Cluster



※ 참고 : <https://medium.com/@tomerf/so-you-want-to-configure-the-perfect-db-cluster-inside-a-kubernetes-cluster-a4d2c26aca7a>

# How ...



항목	자체 서버환경	Cloud 환경	Managed K8s
인프라 관리	사용자	벤더사	벤더사
K8s 설치, 관리	사용자	사용자	벤더사
K8s(클러스터) 백업	사용자	사용자	벤더사
K8s(클러스터) 스케일링	사용자	사용자	벤더사
워크로드 프로비저닝	사용자	사용자	벤더사
어플리케이션 스케일링	사용자	사용자	사용자
어플리케이션 배포	사용자	사용자	사용자

AWS	EKS (Elastic Kubernetes Service)
Azure	AKS (Azure Kubernetes Service)
GCP	GKE (Google Kubernetes Service)
NCP	NKS (Naver Kubernetes Service)

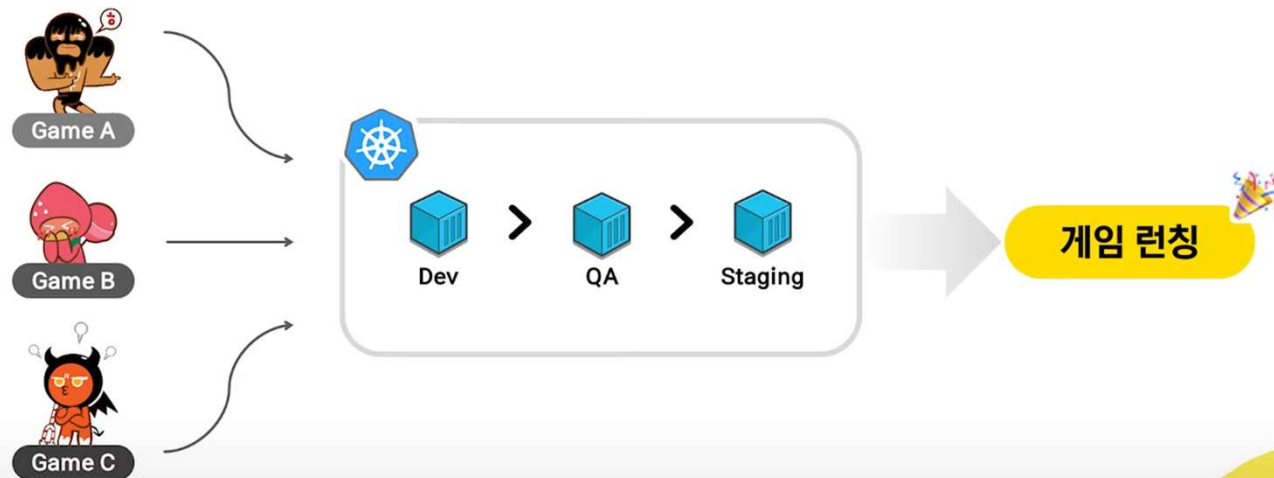
※ 참고 : [https://www.cloocus.com/insight-kubernetes\\_2/](https://www.cloocus.com/insight-kubernetes_2/)

# [자습 #1] 게임 서버를 품은 쿠버네티스

[NDC21-프로그래밍] 게임 서버를 품은 쿠버네티스

01-1 효율적으로 게임을 퍼블리싱할 수 있는 인프라

DEVSISTERS



퍼블리싱 스펙을 만족하는 게임이라면

**동일한 인프라에서 동일한 구조로** 게임을 런칭할 수 있어야 함

쿠버네티스를 활용하면 이를 매우 쉽게 달성할 수 있습니다

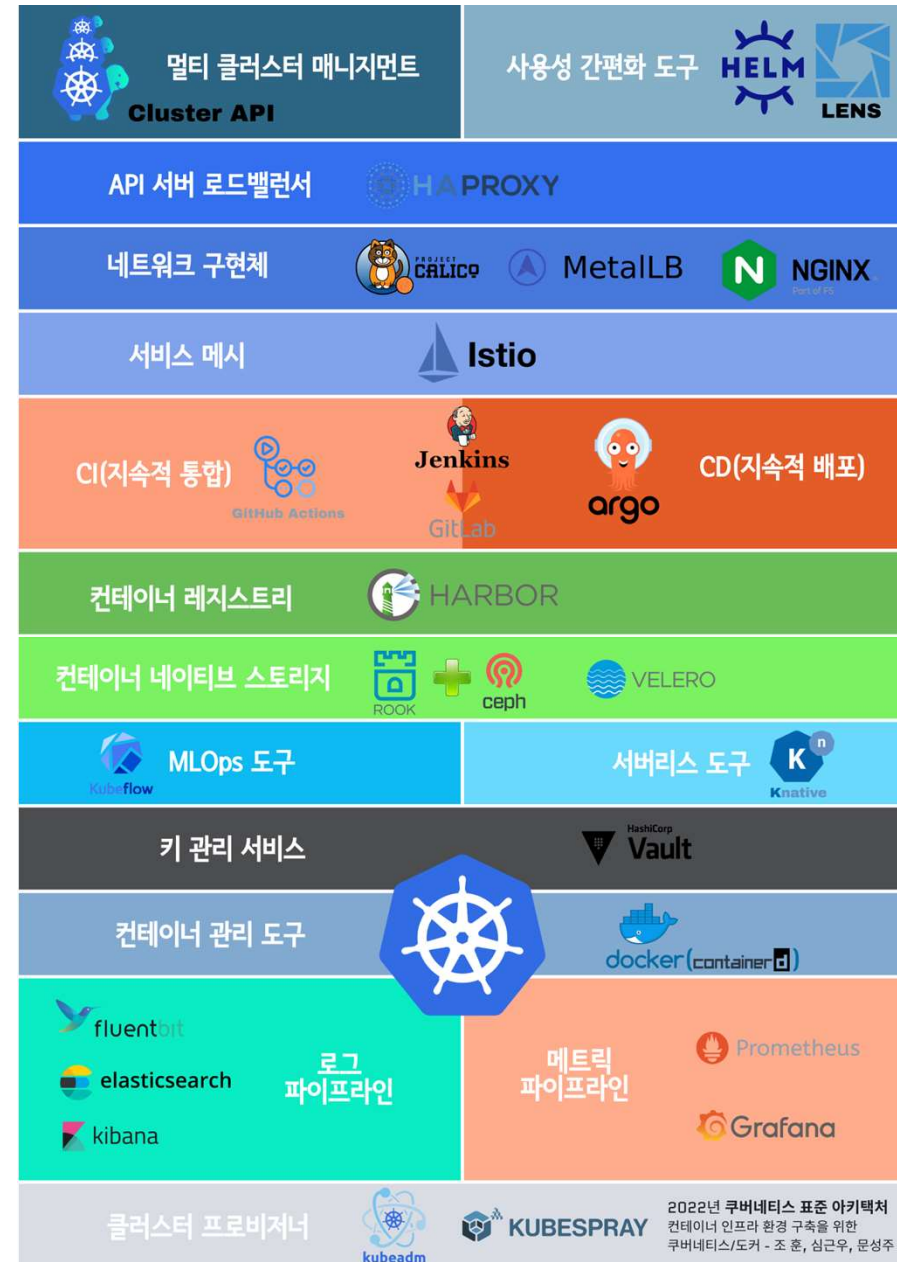


※ 참고 : <https://www.youtube.com/watch?v=8R4DDEqjc0I>

# [자습 #2] Kubernetes 표준 아키텍처

- K8s를 안정적으로 구축하기 위해 (공부가) 필요한 도구들

※ 참고 : [https://github.com/sysnet4admin/Book\\_k8sInfra/tree/main/docs/k8s-stnd-arch/2022](https://github.com/sysnet4admin/Book_k8sInfra/tree/main/docs/k8s-stnd-arch/2022)





# Kubernetes Install

# Windows 10

## - 최소 사양

- . Memory: 8GB
- . Storage: 50GB

## - 권장 사양

- . Memory: 16GB
- . Storage: 100GB

설정

시스템

- 디스플레이
- 소리
- 알림 및 작업
- 집중 지원
- 전원 및 절전
- 저장소
- 태블릿
- 멀티태스킹
- 이 PC에 화면 표시
- 공유 환경
- 클립보드

정보

PC가 모니터링되고 보호됩니다.

자세한 내용은 [Windows 보안을 참조하세요.](#)

장치 사양

장치 이름	SUPER-PC
프로세서	AMD Ryzen 5 3600X 6-Core Processor 3.79 GHz
설치된 RAM	31.9GB
장치 ID	F271527C-5D70-42AD-A5D6-0A2F00F1D001
제품 ID	9C3-1-00000-0000-000000000000
시스템 종류	64비트 운영 체제, x64 기반 프로세서
펜 및 터치	펜 지원

복사

이 PC의 이름 바꾸기

Windows 사양

에디션	Windows 10 Pro
버전	21H1
설치 날짜	2020-08-30
OS 빌드	19043.1826
경험	Windows Feature Experience Pack 120.2212.4180.0

관련 설정

- [BitLocker 설정](#)
- [장치 관리자](#)
- [원격 데스크톱](#)
- [시스템 보호](#)
- [고급 시스템 설정](#)
- [이 PC의 이름 바꾸기\(고급\)](#)
- [도움말 보기](#)
- [피드백 보내기](#)

# VirtualBox



The screenshot shows the Oracle VM VirtualBox website. The browser window has a single tab titled 'Oracle VM VirtualBox' and the address bar shows 'virtualbox.org'. The website features a blue header with the VirtualBox logo on the left and a search bar on the right. The main content area has a large 'VirtualBox' title and a 'Welcome to VirtualBox.org!' message. Below this, there is a paragraph describing VirtualBox as a powerful x86 and AMD64/Intel64 virtualization product. A large blue button with white text says 'Download VirtualBox 6.1'. To the left of the main content, there is a sidebar with links: 'About', 'Screenshots', 'Downloads', 'Documentation' (with sub-links for 'End-user docs' and 'Technical docs'), 'Contribute', and 'Community'. To the right, there is a 'News Flash' section with a list of recent releases and updates, including dates like 'May 17th, 2021' and 'November 22nd, 2021'.

Oracle VM VirtualBox

virtualbox.org

search...  
Login Preferences

## VirtualBox

Welcome to VirtualBox.org!

VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers, it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2. See "About VirtualBox" for an introduction.

Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and OpenSolaris, OS/2, and OpenBSD.

VirtualBox is being actively developed with frequent releases and has an ever growing list of features, supported guest operating systems and platforms it runs on. VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while Oracle ensures the product always meets professional quality criteria.

### Download VirtualBox 6.1

Hot picks:

- Pre-built virtual machines for developers at [Oracle Tech Network](#)
- **Hyperbox** Open-source Virtual Infrastructure Manager [project site](#)
- **phpVirtualBox** AJAX web interface [project site](#)

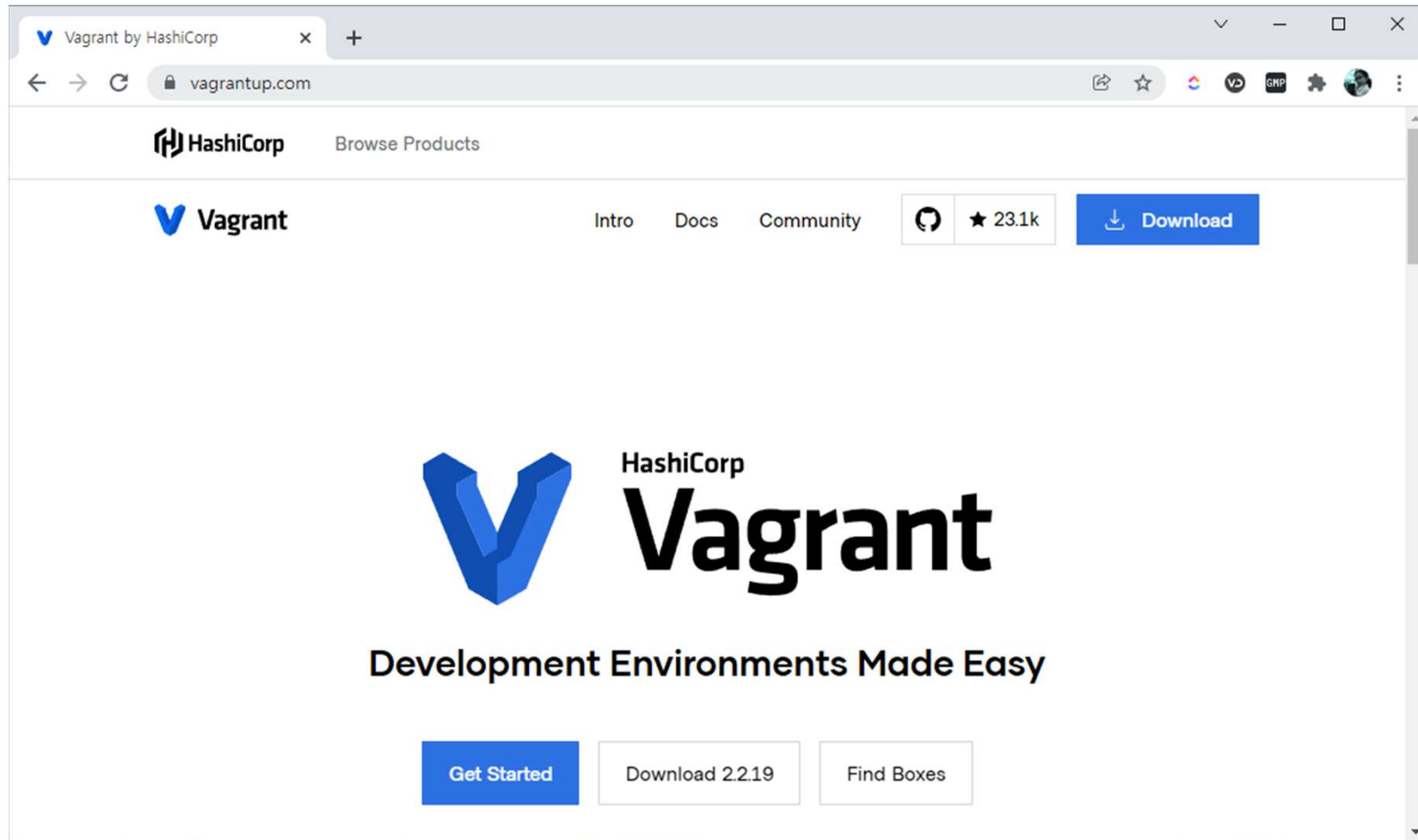
#### News Flash

- **Important** May 17th, 2021 **We're hiring!**  
Looking for a new challenge? We're hiring a VirtualBox senior developer in 3D area (Europe/Russia/India).
- **New** November 22nd, 2021 **VirtualBox 6.1.30 released!**  
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- **New** October 19th, 2021 **VirtualBox 6.1.28 released!**  
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- **New** July 28th, 2021 **VirtualBox 6.1.26 released!**  
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- **New** July 20th, 2021 **VirtualBox 6.1.24 released!**  
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- **New** April 29th, 2021 **VirtualBox 6.1.22 released!**  
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.

※ 참고 : <https://www.virtualbox.org/>

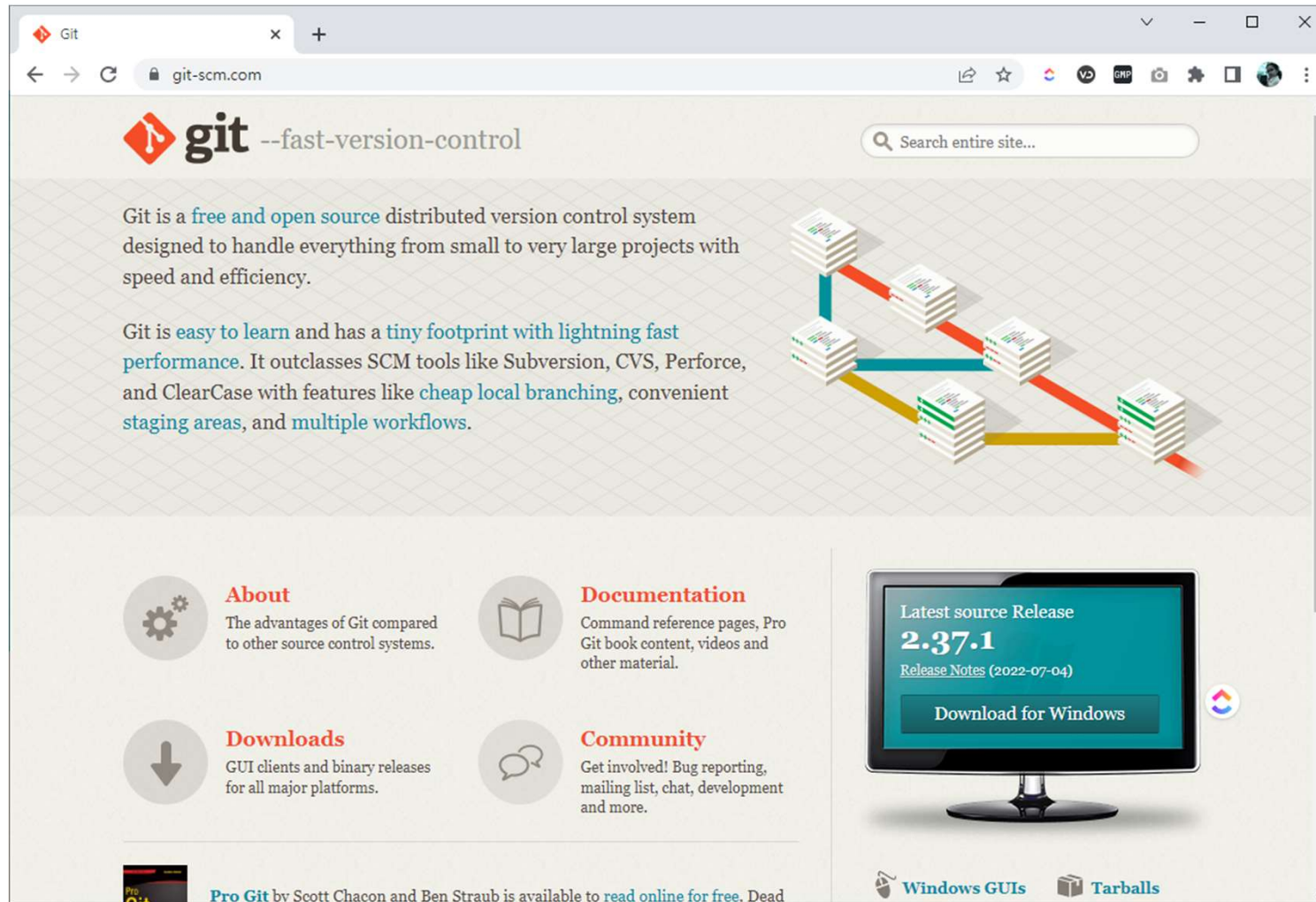


# Vagrant



※ 참고 : <https://www.vagrantup.com/>

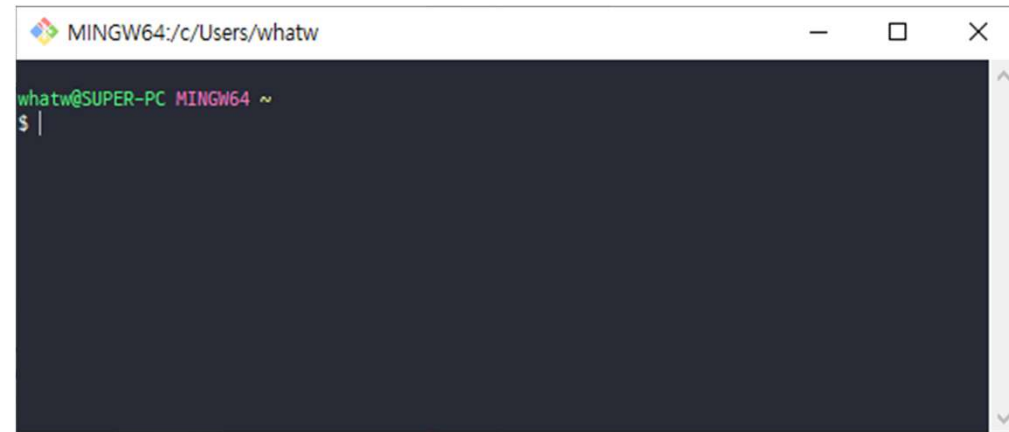
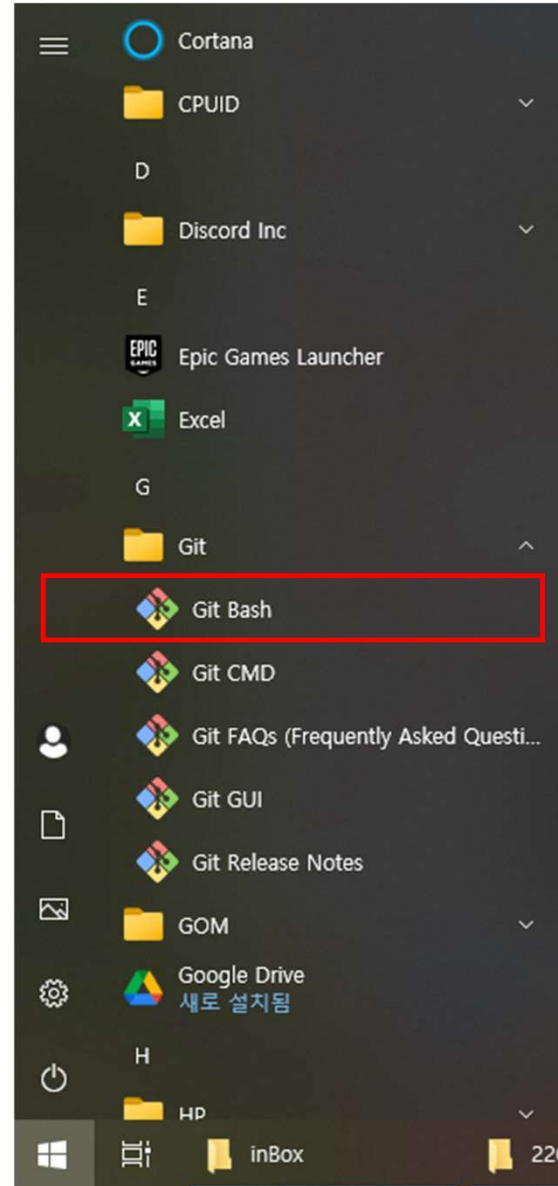
# Git



※ 참고 : <https://git-scm.com/>

# Git Bash

- Git 설치가 잘되었다면,  
"Git Bash"를 사용할 수 있다.



※ 참고 : <https://git-scm.com/>

# Network (공유기)

- 사용할 수 있는 IP 확인
- . 가급적 연속적으로 가능한 범위 확보

DHCP 주소 예약 목록

사용함	호스트명	IP 주소	MAC 주소		
<input checked="" type="checkbox"/>	HP-PC	192.168.100.100	28:00:00:00:00:00		
<input checked="" type="checkbox"/>	master	192.168.100.117	b4:74:00:00:00:00		
<input checked="" type="checkbox"/>	master-stg	192.168.100.111	08:00:27:00:00:00		
<input checked="" type="checkbox"/>	worker1	192.168.100.112	00:00:00:00:00:00		
<input checked="" type="checkbox"/>	worker2	192.168.100.113	08:00:00:00:00:00		

DHCP 클라이언트 목록

호스트명	IP 주소	MAC 주소	만료시간
LAPTOP-DI9USFQ8	192.168.100.103	78:00:00:00:00:00	6 일 22 시 32 분
jk7744-park03	192.168.100.104	60:00:00:00:00:00	6 일 21 시 31 분
jaegyui-Galaxy-Note10-5G	192.168.100.105	6e:53:00:00:00:00	6 일 22 시 39 분
SUPER-PC	192.168.100.101	00:00:00:00:00:00	6 일 17 시 58 분
Galaxy-Note8	192.168.100.106	50:00:00:00:00:00	20 시 34 분
cgllc-airmonitor-b1	192.168.100.110	90:97:00:00:00:00	6 일 3 시 49 분
jaegyui-Galaxy-Tab-S5e	192.168.100.107	00:00:00:00:00:00	4 일 6 시 19 분
Parkui-iPhone	192.168.100.108	8c:00:00:00:00:00	6 일 13 시 56 분

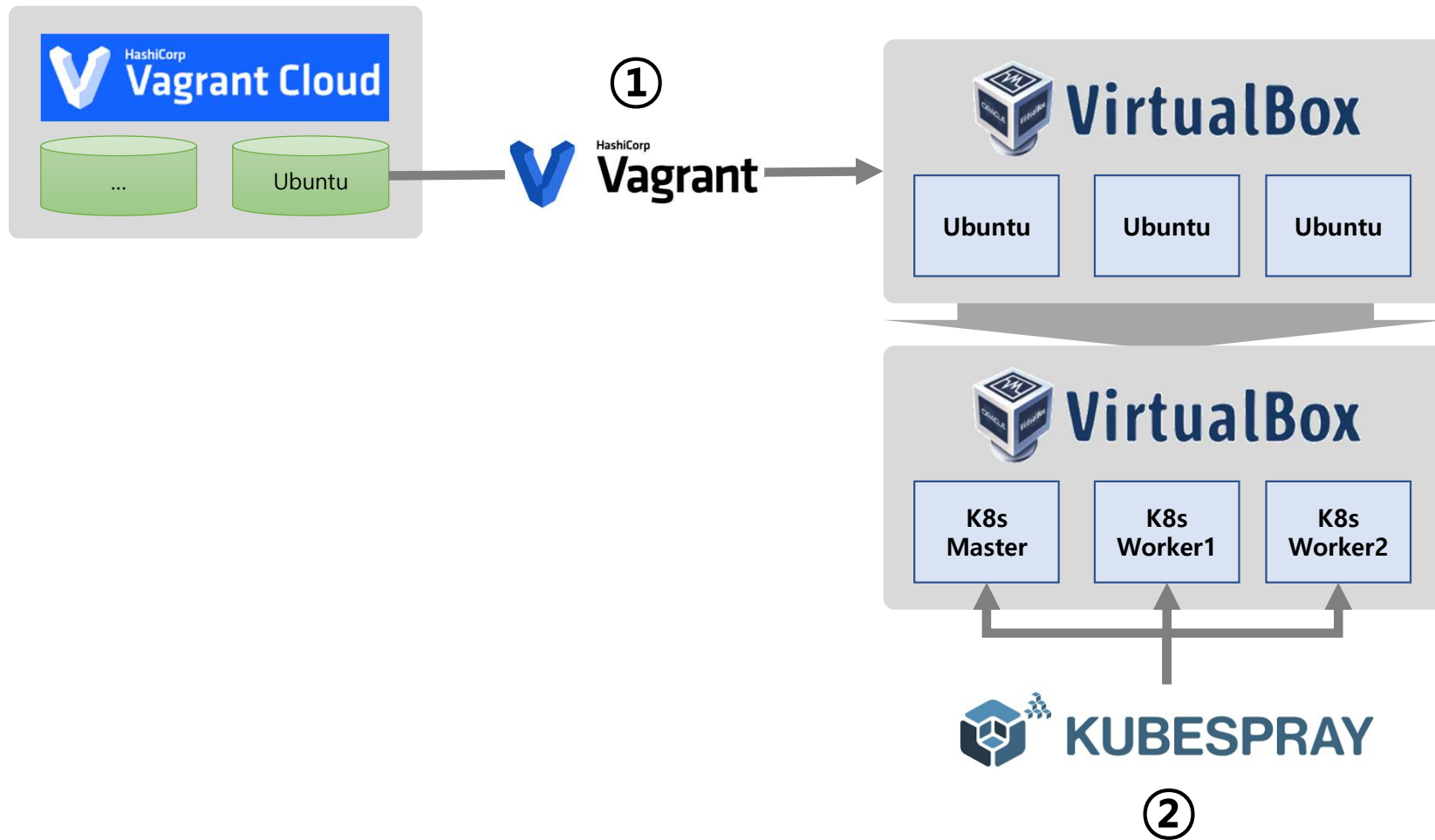
설정저장

설정취소

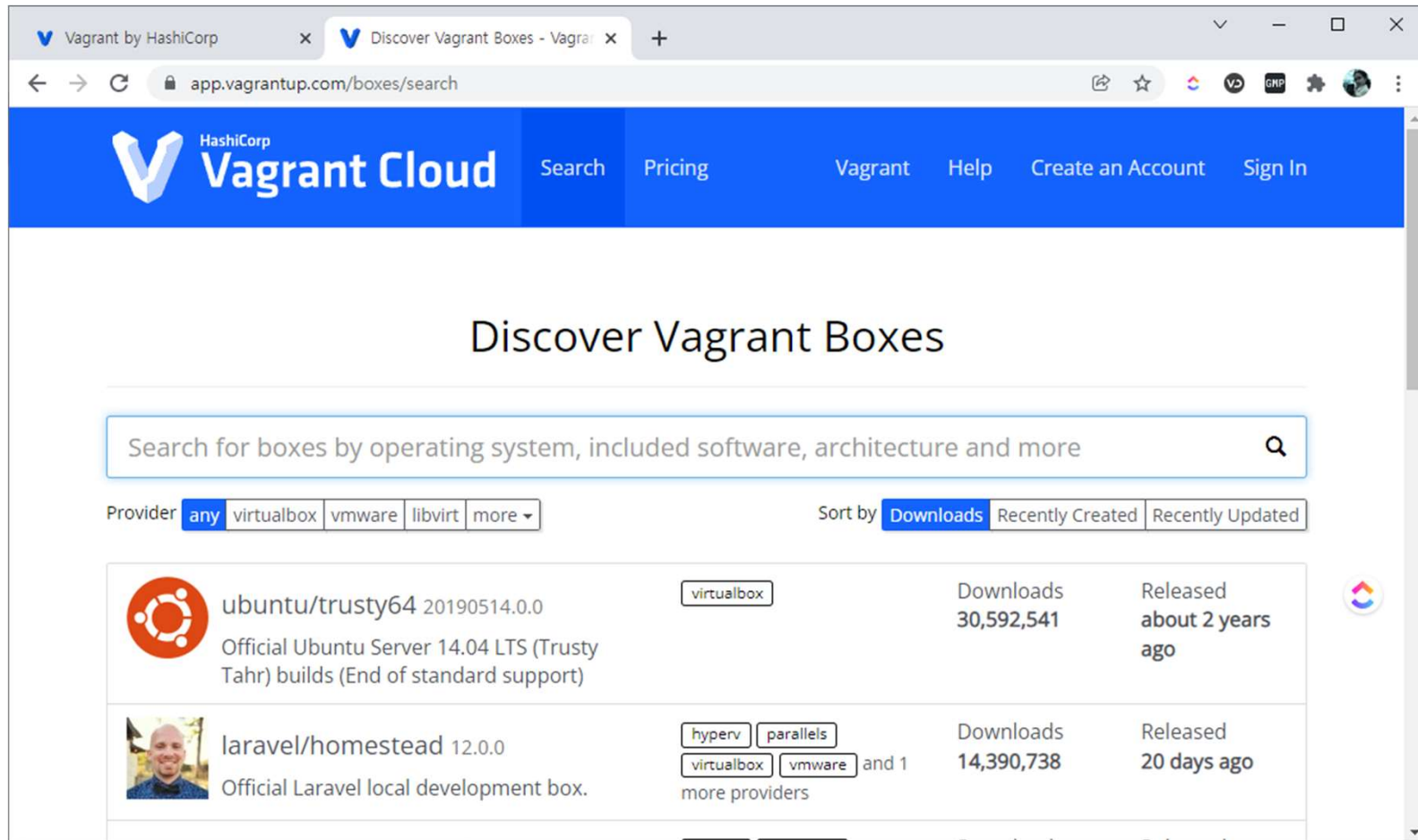
WIRELESS

Copyright © 2013 D-Link Corporation. All rights reserved.

# Action-Item



# Vagrant Cloud (Hub)



The screenshot shows the Vagrant Cloud website interface. The header is blue with the HashiCorp Vagrant Cloud logo and navigation links: Search, Pricing, Vagrant, Help, Create an Account, and Sign In. The main heading is "Discover Vagrant Boxes". Below this is a search bar with the placeholder text "Search for boxes by operating system, included software, architecture and more". Under the search bar, there are filters for "Provider" (any, virtualbox, vmware, libvirt, more) and "Sort by" (Downloads, Recently Created, Recently Updated). The results list shows two boxes:

Box Name	Version	Provider	Downloads	Released
ubuntu/trusty64	20190514.0.0	virtualbox	30,592,541	about 2 years ago
laravel/homestead	12.0.0	hyperv, parallels, virtualbox, vmware and 1 more providers	14,390,738	Released 20 days ago

※ 참고 : <https://app.vagrantup.com/boxes/search>

# Vagrantfile

- Kubernetes 환경을 구축할 Ubuntu Server 3대를 VirtualBox에 설치하기 위해 Vagrant를 활용해보자.

```
# -*- mode: ruby -*-

N = 2

Vagrant.configure("2") do |config|

  config.ssh.username = "vagrant"
  config.ssh.password = "vagrant"

  config.vm.define "k8s-master" do |cfg|

    cfg.vm.box = "whatwant/Ubuntu-20.04-Server"
    cfg.vm.box_version = "0.5.0"

    cfg.vm.hostname = "master"
    cfg.vm.network "public_network", ip: "192.168.100.200"

    cfg.vm.provider "virtualbox" do |vb|
      vb.gui = false
      vb.cpus = "2"
      vb.memory = "2048"
    end

    cfg.vm.provision "shell", inline: <<-SHELL
      apt-get update
      apt-get upgrade -y
    SHELL
  end
end
```

```
(1..N).each do |i|

  config.vm.define "k8s-worker#{i}" do |cfg|

    cfg.vm.box = "whatwant/Ubuntu-20.04-Server"
    cfg.vm.box_version = "0.5.0"

    cfg.vm.hostname = "worker#{i}"
    cfg.vm.network "public_network", ip: "192.168.100.20#{i}"

    cfg.vm.provider "virtualbox" do |vb|
      vb.gui = false
      vb.cpus = "1"
      vb.memory = "1280"
    end

    cfg.vm.provision "shell", inline: <<-SHELL
      apt-get update
      apt-get upgrade -y
    SHELL
  end
end
end
```

# Create VMs

- 실습용 파일을 다운로드 받아서 활용해보자.

```
host > git clone https://github.com/whatwant-school/kubernetes.git
host > cd kubernetes/02-Pod-Namespaces/hands-on

host > ls -al

total 3
drwxr-xr-x 1 whatw 197609    0  7월 21 00:05 ./
drwxr-xr-x 1 whatw 197609    0  7월 21 00:03 ../
-rw-r--r-- 1 whatw 197609 1409  7월 21 00:20 Vagrantfile

host > vagrant up

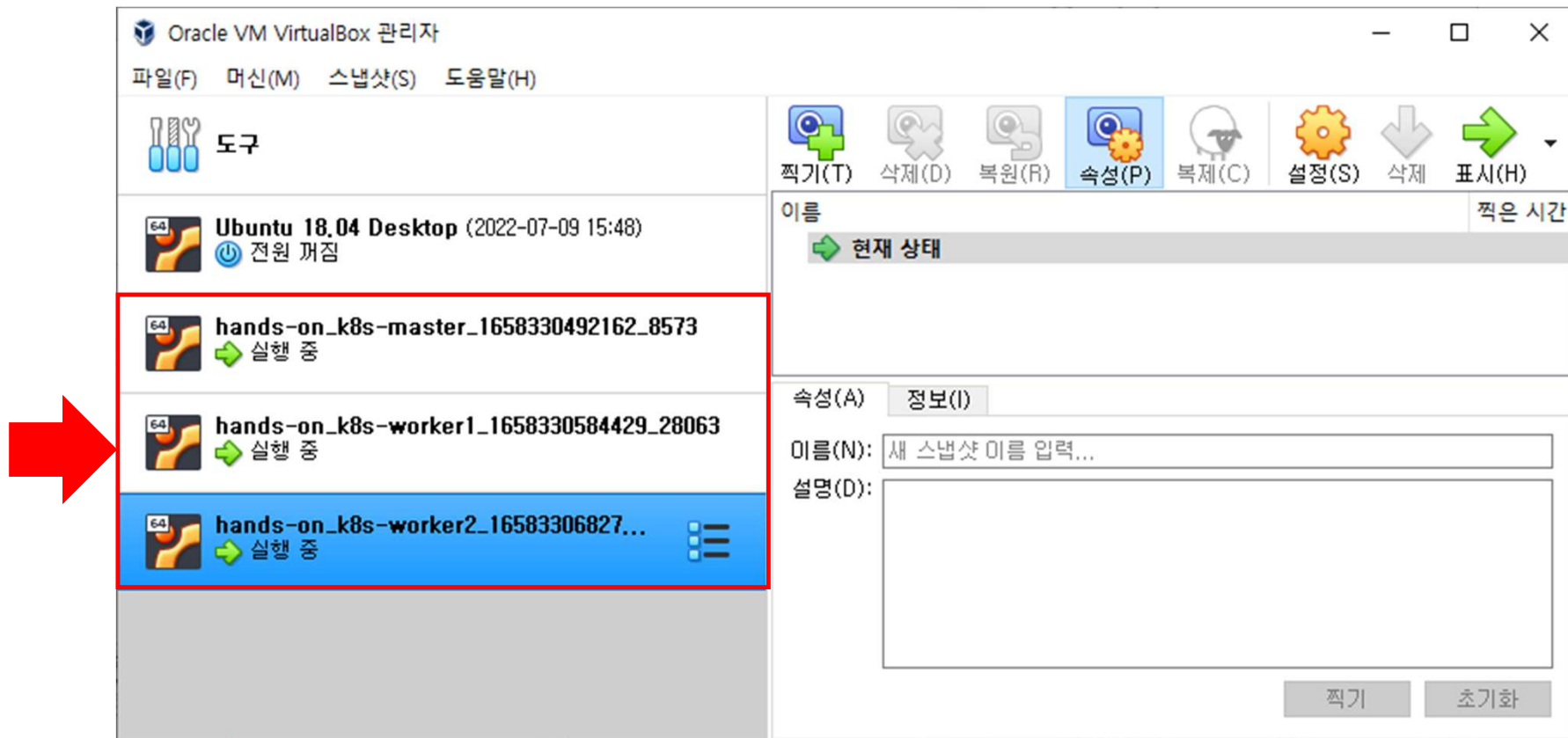
Bringing machine 'k8s-master' up with 'virtualbox' provider...
Bringing machine 'k8s-worker1' up with 'virtualbox' provider...
Bringing machine 'k8s-worker2' up with 'virtualbox' provider...
==> k8s-master: Importing base box 'whatwant/Ubuntu-20.04-Server'...
...
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>



# Create VMs

- VirtualBox에 잘 만들어져서 실행 중인 것을 볼 수 있다.



※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# SSH-Key 등록

- Kubernetes 설치에 master node로 사용될 server에서 실행을 한다.
- "Kubespray"라는 도구를 이용하기 위해서는 master node에서 worker nodes에 SSH 접근이 가능해야 한다.

① master node 접속 (id/password = vagrant/vagrant)

```
host > ssh vagrant@192.168.100.200
```

② SSH-Key 생성 (default로 계속 enter 진행해도 무방)

```
master > ssh-keygen
```

③ worker nodes에 SSH-Key 등록 (master 자신에게도 등록)

```
master > ssh-copy-id 192.168.100.200
```

```
master > ssh-copy-id 192.168.100.201
```

```
master > ssh-copy-id 192.168.100.202
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# Server configuration

- master/worker node로 사용될 모든 server는 Kubernetes를 위한 설정이 필요하다.

## ① swap off

```
master/worker1/worker2 > sudo swapoff -a  
master/worker1/worker2 > sudo cat /etc/fstab
```

```
...  
# /swap.img    none    swap    sw    0    0  
...
```

## ② ip\_forward

```
master/worker1/worker2 > sudo sh -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'
```

## ③ hostname

```
master/worker1/worker2 > sudo nano /etc/hosts
```

```
192.168.100.200 master  
192.168.100.201 worker1  
192.168.100.202 worker2
```

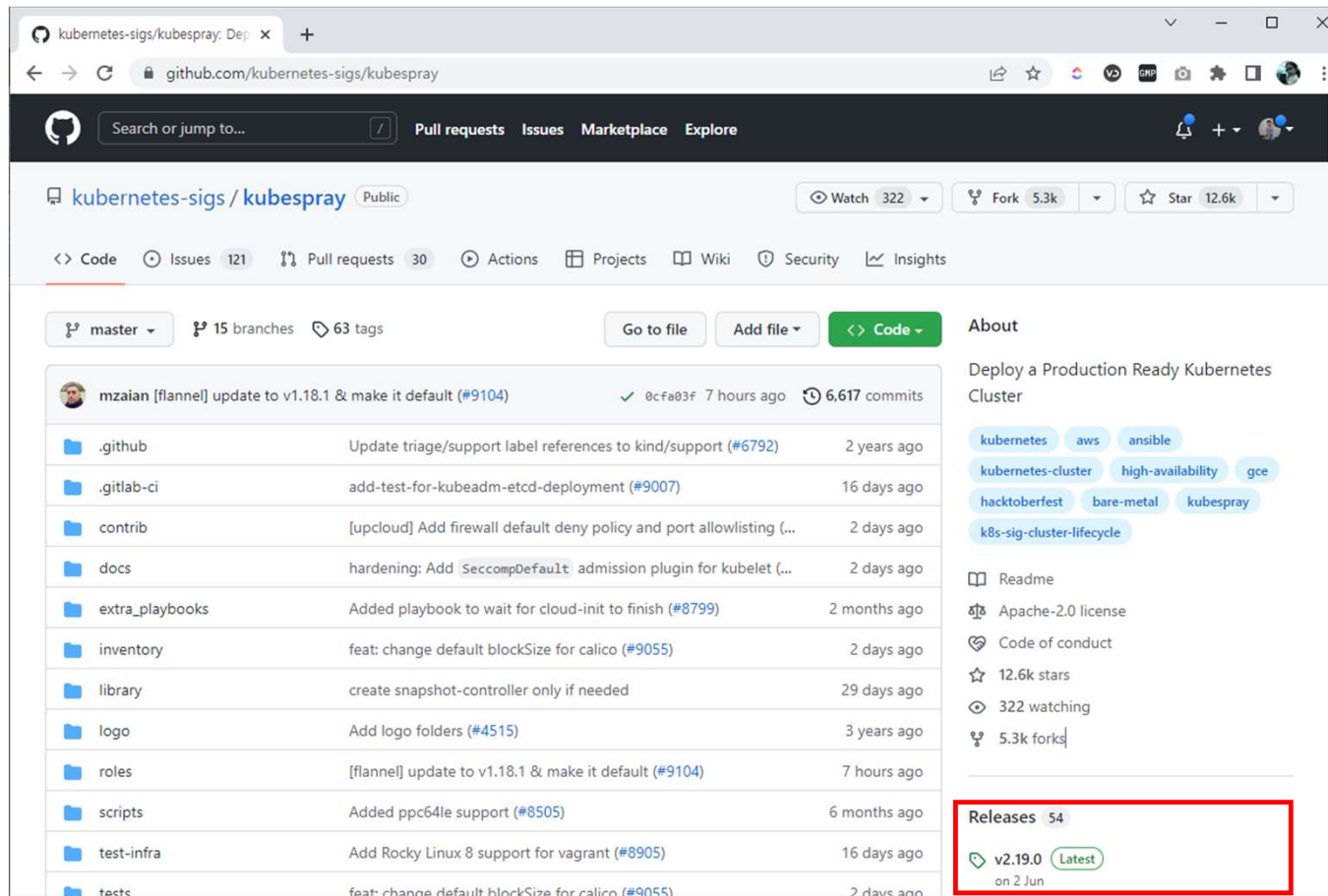
## ④ reboot

```
master/worker1/worker2 > sudo reboot
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# Kubespray

- Latest Release를 확인해보자.



github.com/kubernetes-sigs/kubespray

Search or jump to... Pull requests Issues Marketplace Explore

kubernetes-sigs / kubespray Public

Watch 322 Fork 5.3k Star 12.6k

<> Code Issues 121 Pull requests 30 Actions Projects Wiki Security Insights

master 15 branches 63 tags

Go to file Add file <> Code

mzaian [flannel] update to v1.18.1 & make it default (#9104) 7 hours ago 6,617 commits

.github	Update triage/support label references to kind/support (#6792)	2 years ago
.gitlab-ci	add-test-for-kubeadm-etcd-deployment (#9007)	16 days ago
contrib	[upcloud] Add firewall default deny policy and port allowlisting (...)	2 days ago
docs	hardening: Add SeccompDefault admission plugin for kubelet (...)	2 days ago
extra_playbooks	Added playbook to wait for cloud-init to finish (#8799)	2 months ago
inventory	feat: change default blockSize for calico (#9055)	2 days ago
library	create snapshot-controller only if needed	29 days ago
logo	Add logo folders (#4515)	3 years ago
roles	[flannel] update to v1.18.1 & make it default (#9104)	7 hours ago
scripts	Added ppc64le support (#8505)	6 months ago
test-infra	Add Rocky Linux 8 support for vagrant (#8905)	16 days ago
tests	feat: change default blockSize for calico (#9055)	2 days ago

About

Deploy a Production Ready Kubernetes Cluster

kubernetes aws ansible  
kubernetes-cluster high-availability gce  
hacktoberfest bare-metal kubespray  
k8s-sig-cluster-lifecycle

Readme  
Apache-2.0 license  
Code of conduct  
12.6k stars  
322 watching  
5.3k forks

Releases 54

v2.19.0 Latest  
on 2 Jun

※ 참고 : <https://github.com/kubernetes-sigs/kubespray>

# Kubespray - download

## ① clone

```
master > cd /srv/install  
master > git clone https://github.com/kubernetes-sigs/kubespray.git  
master > cd kubespray  
  
master > git switch -c v2.19.0 v2.19.0
```

## ② uninstall ansible

```
master > sudo apt purge ansible  
  
master > sudo pip uninstall ansible
```

## ③ install package

```
master > sudo pip install -r requirements.txt
```

## ④ generate configuration

```
master > cp -rfp inventory/sample inventory/mycluster  
  
master > declare -a IPS=(192.168.100.200 192.168.100.201 192.168.100.202)  
  
master > CONFIG_FILE=inventory/mycluster/hosts.yaml python contrib/inventory_builder/inventory.py ${IPS[@]}
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# Kubespray - hosts.yaml / kube\_proxy\_mode

- 전체적인 구성은 교육을 위한 일반적인 형태로 진행하겠다.

```
master > nano ./inventory/mycluster/hosts.yaml
```

```
all:
  hosts:
    master:
      ansible_host: 192.168.100.200
      ip: 192.168.100.200
      access_ip: 192.168.100.200
    worker1:
      ansible_host: 192.168.100.201
      ip: 192.168.100.201
      access_ip: 192.168.100.201
    worker2:
      ansible_host: 192.168.100.202
      ip: 192.168.100.202
      access_ip: 192.168.100.202
```

```
children:
  kube_control_plane:
    hosts:
      master:
  kube_node:
    hosts:
      worker1:
      worker2:
  etcd:
    hosts:
      master:
  k8s_cluster:
    children:
      kube_control_plane:
      kube_node:
  calico_rr:
    hosts: {}
```

- 교육을 위해 일단 iptables로 설정하겠다.

```
master > nano ./inventory/mycluster/group_vars/k8s_cluster/k8s-cluster.yml
```

```
# Kube-proxy proxyMode configuration.
# Can be ipvs, iptables
kube_proxy_mode: iptables
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# Kubespray install

- 준비는 끝났다. 이제 설치 !!!

```
master > ansible-playbook -i inventory/mycluster/hosts.yaml --become --become-user=root cluster.yml
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# kubectl setup in master

- master node에서 일반 사용자 계정에서 'kubectl'을 사용하기 위한 설정을 진행하자.

```
master > mkdir -p $HOME/.kube  
  
master > sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  
  
master > sudo chown $(id -u):$(id -g) $HOME/.kube/config  
  
master > echo "source <(kubectl completion zsh)" >> ~/.zshrc  
  
master > source ~/.zshrc
```

- node 정보를 확인해보자 !

```
master > kubectl get nodes -o wide
```

NAME	STATUS	ROLES	AGE	VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE	KERNEL-VERSION	CONTAINER-RUNTIME
master	Ready	control-plane,master	15m	v1.23.7	192.168.100.200	<none>	Ubuntu 20.04.4 LTS	5.4.0-122-generic	containerd://1.6.4
worker1	Ready	<none>	14m	v1.23.7	192.168.100.201	<none>	Ubuntu 20.04.4 LTS	5.4.0-122-generic	containerd://1.6.4
worker2	Ready	<none>	14m	v1.23.7	192.168.100.202	<none>	Ubuntu 20.04.4 LTS	5.4.0-122-generic	containerd://1.6.4

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>



# kubectl setup in workspace - 1/2

- 내가 작업할 PC에서 Kubernetes 사용을 하기 위해서는 우선 "kubectl" 설치가 필요하다.
- "kubectl"을 설치할 때에는 사용할 Kubernetes version과 같은 version으로 설치해야 한다.

```
remote > cd /srv/install/kubectl  
  
remote > curl -LO "https://dl.k8s.io/release/v1.23.7/bin/linux/amd64/kubectl"  
  
remote > sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
```

- 그런데, 잘 설치되었는지 확인해보면 문제가 보인다.

```
remote > kubectl version  
  
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.7", GitCommit:"42c05a547468804b2053ecf60a3bd15560362fc2", GitTreeState:"clean",  
BuildDate:"2022-05-24T12:30:55Z", GoVersion:"go1.17.10", Compiler:"gc", Platform:"linux/amd64"}  
The connection to the server localhost:8080 was refused - did you specify the right host or port?
```

- API Server (master node) 정보를 제대로 찾지 못해서 발생하는 현상이다.

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>

# kubectl setup in workspace - 2/2

- master node로부터 config 정보를 얻어보도록 하겠다.

```
remote > mkdir ~/.kube
```

```
remote > scp vagrant@192.168.100.200:/home/vagrant/.kube/config ~/.kube/
```

```
The authenticity of host '192.168.100.200 (192.168.100.200)' can't be established.  
ECDSA key fingerprint is SHA256:RX1VW+w652onI+Tz8gPnJRm7SM1urzscf8iHQeJFF0o.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '192.168.100.200' (ECDSA) to the list of known hosts.  
config
```

- API Server (master node) 정보를 업데이트 하자.

```
remote > nano ~/.kube/config
```

```
...  
- cluster:  
  certificate-authority-data:...  
  server: https://192.168.100.200:6443  
  name: cluster.local  
...
```

- 이제 잘 나온다 !!!

```
remote > kubectl version
```

```
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.7", GitCommit:"42c05a547468804b2053ecf60a3bd15560362fc2", GitTreeState:"clean",  
BuildDate:"2022-05-24T12:30:55Z", GoVersion:"go1.17.10", Compiler:"gc", Platform:"linux/amd64"}  
Server Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.7", GitCommit:"42c05a547468804b2053ecf60a3bd15560362fc2", GitTreeState:"clean",  
BuildDate:"2022-05-24T12:24:41Z", GoVersion:"go1.17.10", Compiler:"gc", Platform:"linux/amd64"}
```

※ 참고 : <https://www.whatwant.com/entry/Kubernetes-Vagrant-VirtualBox-Kubespray>



**Break**



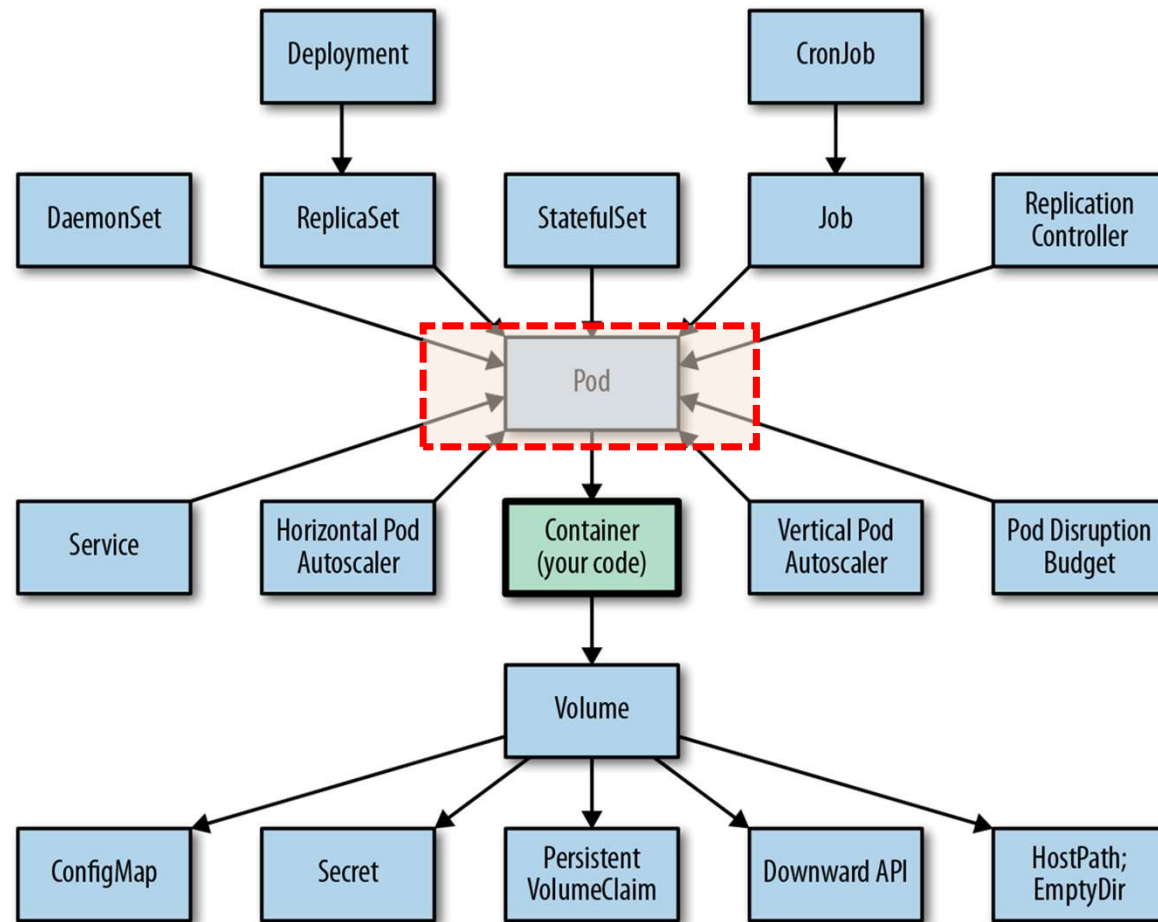
# **Flip Learning**

## **(Pods & Namespaces)**

**염혜원 님**



## Kubernetes concepts for developers



※ 참고 : <https://www.oreilly.com/library/view/kubernetes-patterns/9781492050278/ch01.html>



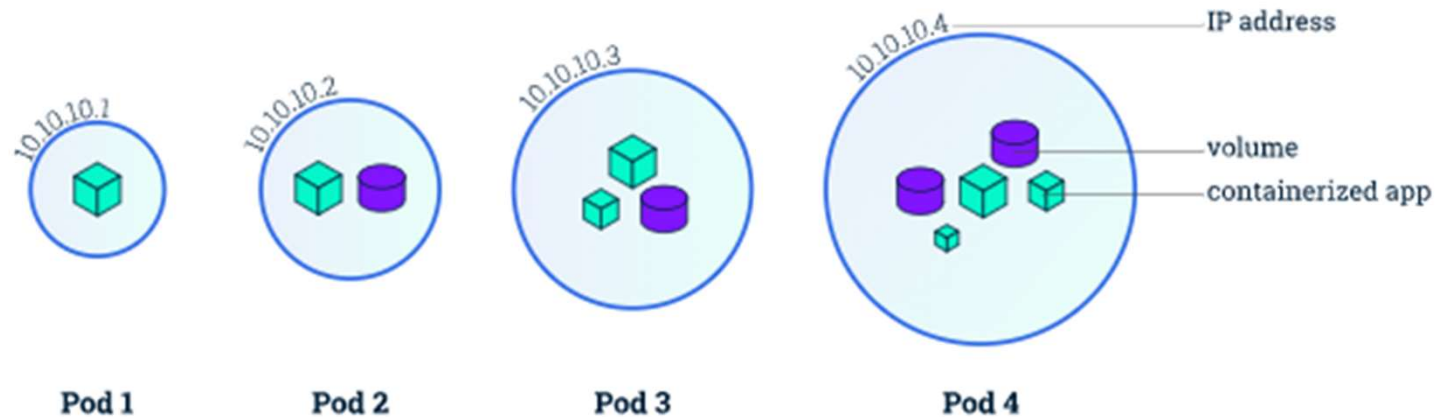
# Kubernetes

## Pod

# Pod is ... #1

**Pod**는 Kubernetes에서 생성하고 관리할 수 있는 배포 가능한 가장 작은 컴퓨팅 단위이다.

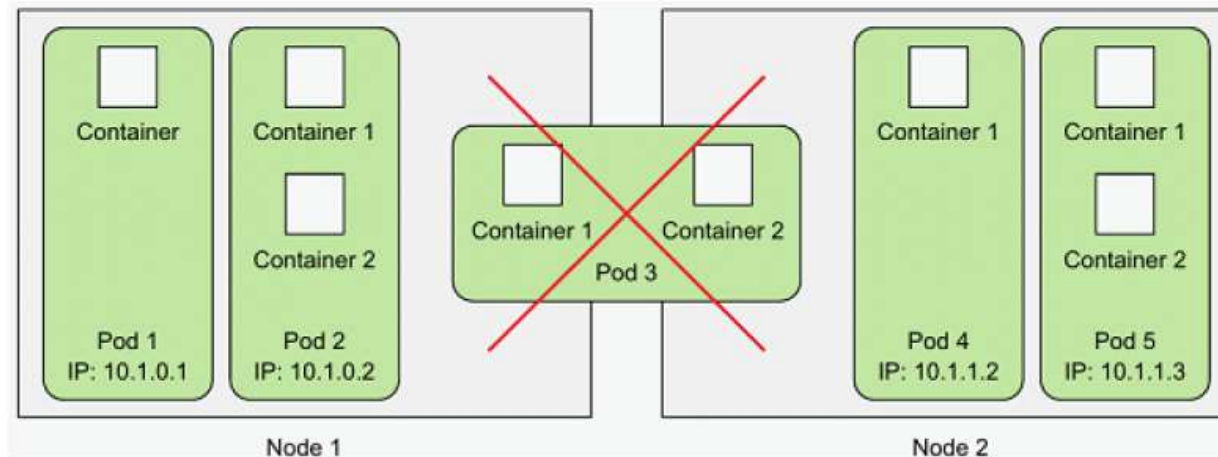
**Pod**는 하나 이상의 컨테이너 그룹이다.



※ 참고 : <https://kubernetes.io/ko/docs/tutorials/kubernetes-basics/explore/explore-intro/>

## Pod is ... #2

- Pod는 함께 배치된 Container 그룹을 의미
- Container는 단일 프로세스를 실행하는 것을 목적으로 설계
- 따라서, 여러 Container를 묶고 하나의 단위로 관리할 수 있는 상위 구조가 필요 → Pod
- Kubernetes는 Pod 단위로 배포하고 운영



▲ 그림 3.1 파드 안에 있는 모든 컨테이너는 같은 노드에서 실행된다. 절대로 두 노드에 걸쳐 배포되지 않는다.

※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-3/10>

# Pod YAML

## - 기본 요소들

- . **apiVersion**: 리소스에 따라 알맞은 API Version 명시
- . **kind**: 리소스 종류를 정의
- . **metadata**: 리소스의 메타데이터(이름/라벨) 작성
- . **spec**: 생성할 리소스 상세 스펙 설정

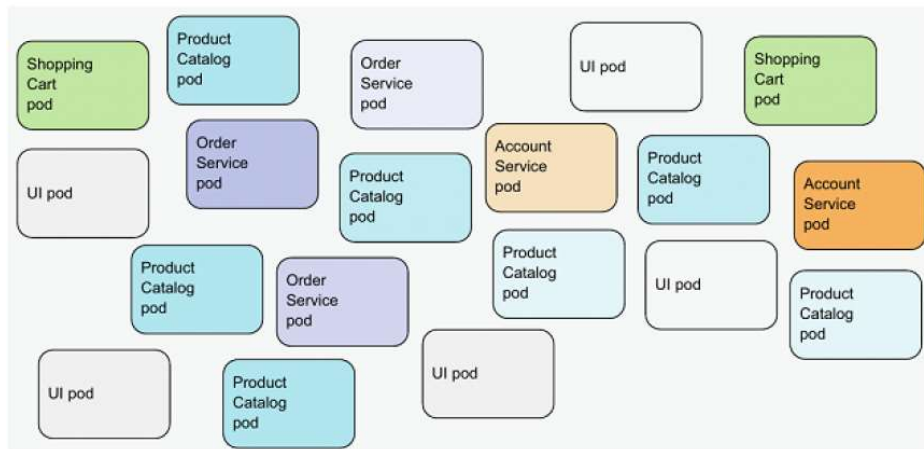
```
apiVersion: v1
kind: Pod

metadata:
  name: Test-Pod
  labels:
    app: web

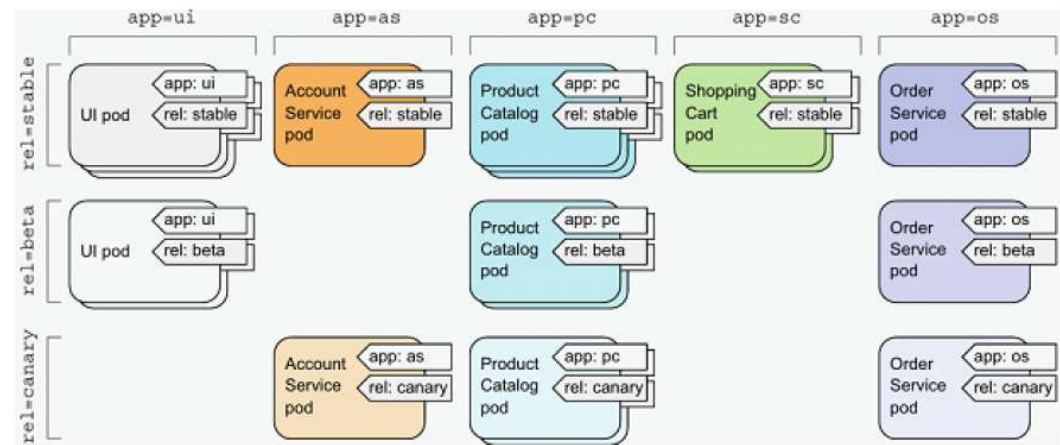
spec:
  containers:
    - name: nginx
      image: nginx:latest
      ports:
        - containerPort: 8080
```

# labels

- Label은 Kubernetes 리소스를 분류할 수 있는 기능
- 각 오브젝트는 하나 이상의 레이블을 가질 수 있으며 label은 Key-Value Pair로 이루어짐
- Kubernetes 명령어에서 동일한 label을 가진 오브젝트를 선택할 수 있음



▲ 그림 3.6 마이크로서비스 아키텍처 안에 있는 분류되지 않는 파드



▲ 그림 3.7. 파드 레이블을 이용해 마이크로서비스 아키텍처 안에 파드를 조직화했다.



# **K8s : Pod Hands-On**

# Scenario

- 다음과 같이 실습을 해보겠다.

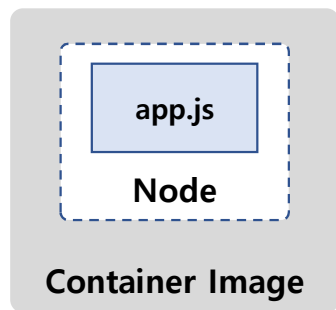
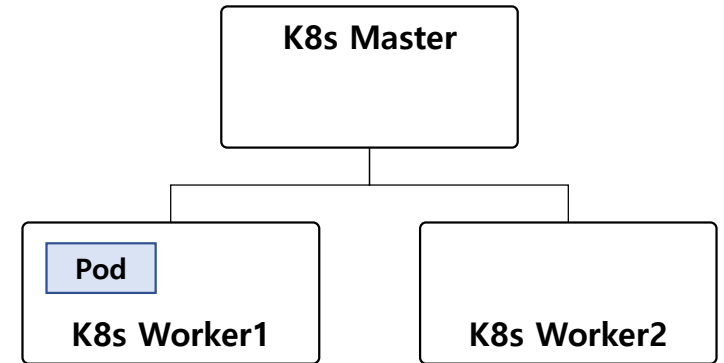


Image Build



Image Push



Pod Create

※ 참고 : <https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia>



# Make image

- node web image를 하나 만들어 보자.

```
const http = require('http');
const os = require('os');

console.log("node-web server starting...");

var handler = function(request, response) {
  console.log("Received request from " +
    request.connection.remoteAddress);
  response.writeHead(200);
  response.end("You've hit " + os.hostname() + "\n");
};

var www = http.createServer(handler);
www.listen(8080);
```

**app.js**

```
FROM node:latest
ADD app.js /app.js
ENTRYPOINT ["node", "app.js"]
```

**Dockerfile**

```
remote > git clone https://github.com/whatwant-school/kubernetes.git
remote > cd kubernetes/02-Pod-Namespaces/hands-on/

remote > docker build -t node-web:1.0 .

remote > docker tag node-web:1.0 <user-id>/node-web:1.0

remote > docker push <user-id>/node-web:1.0
```

※ 참고 : <https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia>

# Create pod with YAML

- 앞에서 만든 container를 실행하기 위한 Pod를 생성해보자.

```
apiVersion: v1                                pod-node-web.yaml
kind: Pod

metadata:
  name: node-web
  labels:
    creation_method: manual
    env: stage

spec:
  containers:
    - image: whatwant/node-web:1.0
      name: node-web
      ports:
        - containerPort: 8080
          protocol: TCP
```

- 웹 연결은 당연히 안된다.

```
remote > git clone https://github.com/whatwant-school/kubernetes.git
remote > cd advanced-kubernetes/02-week/Pod
```

```
remote > kubectl get pods
```

No resources found in default namespace.

```
remote > kubectl create -f pod-node-web.yaml
```

pod/node-web created

```
remote > kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
node-web	1/1	Running	0	71s

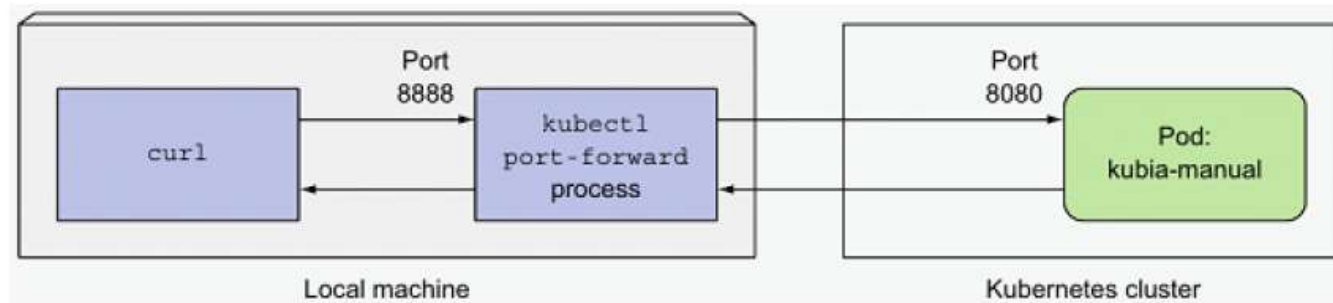
```
remote > curl http://localhost:8080
```

curl: (7) Failed to connect to localhost port 8080: 연결이 거부됨

※ 참고 : <https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia>

# Port forward

- Container에서 실행되고 있는 Web을 연결하기 위해 port-forwarding을 해보자.



▲ 그림 3.5 curl을 kubectl port-forward와 함께 사용할 때 일어나는 일을 간략하게 설명한다.

```
remote > kubectl port-forward node-web 8080:8080 &
```

```
[1] 8466
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
```

```
remote > curl http://localhost:8080
```

```
Handling connection for 8080
You've hit node-web
```

※ 참고 : <https://github.com/luksa/kubernetes-in-action/tree/master/Chapter02/kubia>

# Delete pod

- port-forward 삭제

```
remote > ps -ef | grep kubectl
```

```
chani      8466    3572    0 15:35 pts/1    00:00:00 kubectl port-forward node-web 8080:8080
```

```
remote > kill -9 8466
```

```
[1]  + 8466 killed      kubectl port-forward node-web 8080:8080
```

- pod 삭제

```
remote > kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web	1/1	Running	0	28m	10.233.103.67	worker2	<none>	<none>

```
remote > kubectl delete pods node-web
```

```
pod "node-web" deleted
```

```
remote > kubectl get pods -o wide
```

```
No resources found in default namespace.
```

# Create pod with CLI

- YAML 파일 없이도 그냥 Pod를 생성할 수 있다.

```
remote > kubectl run node-web-command --image whatwant/node-web:1.0 --port=8080
```

```
pod/node-web-command created
```

```
remote > kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-command	1/1	Running	0	4s	10.233.103.68	worker2	<none>	<none>

- 웹 연결 및 삭제는 앞에서 진행한 내용과 동일하다.

```
remote > kubectl port-forward node-web-command 8080:8080 &
```

```
remote > curl http://localhost:8080
```

```
remote > ps -ef | grep kubectl
```

```
remote > kill -9 11766
```

```
remote > kubectl delete pods node-web-command
```

```
pod "node-web-command" deleted
```

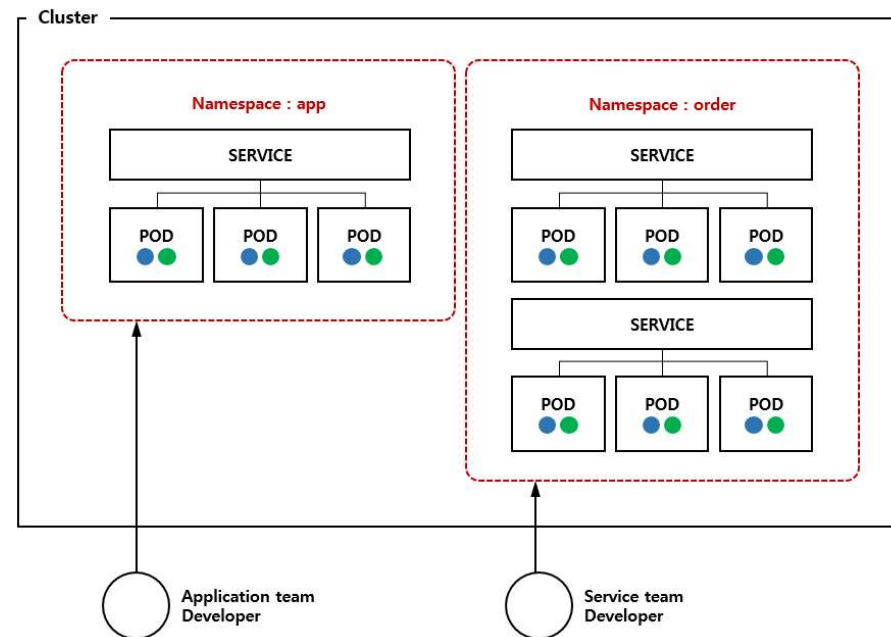


**Kubernetes**

**Namespace**

# Namespace is ...

- 네임스페이스는 클러스터 자원을 (리소스 쿼터를 통해) 여러 사용자 사이에서 나누는 방법
  - . 여러 개의 팀이나, 프로젝트에 걸쳐서 많은 사용자가 있는 환경에서 사용
  - . 네임스페이스는 이름의 범위를 제공
  - . 네임스페이스는 서로 중첩될 수 없으며, 각 쿠버네티스 리소스는 하나의 네임스페이스에만 존재
  - . Pod, ReplicaSet, Deployment, Service 등을 묶어 놓을 수 있는 하나의 가상 공간 또는 그룹



※ 참고 : <https://wiki.webnori.com/display/kubernetes/Namespaces>



# Namespace YAML

- No Comment

```
apiVersion: v1
kind: Namespace

metadata:
  name: whatwant
```



# **K8s : Namespace Hands-On**

# Namespace

- 이미 많은 namespace가 있다. 확인해보자.

```
remote > kubectl get namespaces
```

NAME	STATUS	AGE
default	Active	24h
kube-node-lease	Active	24h
kube-public	Active	24h
kube-system	Active	24h

- 명시적으로 지정하지 않으면 "default"

```
remote > kubectl get pods --namespace default
```

```
No resources found in default namespace.
```

```
remote > kubectl get pods --namespace kube-system
```

NAME	READY	STATUS	RESTARTS	AGE
calico-kube-controllers-6dd874f784-rjxtz	1/1	Running	2 (88m ago)	24h
calico-node-599ck	1/1	Running	1 (89m ago)	24h
calico-node-qlhvf	1/1	Running	1 (88m ago)	24h
calico-node-tpwvg	1/1	Running	1 (89m ago)	24h
coredns-76b4fb4578-kc7vm	1/1	Running	1 (89m ago)	24h
coredns-76b4fb4578-zbtvb	1/1	Running	1 (89m ago)	24h
dns-autoscaler-7979fb6659-p5fpm	1/1	Running	1 (89m ago)	24h
kube-apiserver-master	1/1	Running	2 (89m ago)	24h
kube-controller-manager-master	1/1	Running	2 (89m ago)	24h
kube-proxy-67mzz	1/1	Running	1 (88m ago)	24h
kube-proxy-hznhg	1/1	Running	1 (89m ago)	24h
kube-proxy-jhptq	1/1	Running	1 (89m ago)	24h
kube-scheduler-master	1/1	Running	2 (89m ago)	24h
nginx-proxy-worker1	1/1	Running	1 (89m ago)	24h
nginx-proxy-worker2	1/1	Running	1 (88m ago)	24h
nodelocaldns-7d25c	1/1	Running	1 (88m ago)	24h
nodelocaldns-ct8zg	1/1	Running	1 (89m ago)	24h
nodelocaldns-lb5pl	1/1	Running	3 (89m ago)	24h

# Create namespace with YAML

- YAML 파일을 이용해서 namespace를 생성해보자

```
apiVersion: v1          namespace-whatwant.yaml
kind: Namespace
metadata:
  name: whatwant
```

```
remote > git clone https://github.com/whatwant-school/kubernetes.git
remote > cd kubernetes/02-Pod-Namespaces/hands-on
```

```
remote > kubectl create -f namespace-whatwant.yaml
```

```
namespace/whatwant created
```

```
remote > kubectl get namespaces
```

NAME	STATUS	AGE
default	Active	24h
kube-node-lease	Active	24h
kube-public	Active	24h
kube-system	Active	24h
whatwant	Active	16s

# Create pod in the namespace

- namespace를 지정해서 pod를 생성할 수 있다.

```
remote > git clone https://github.com/whatwant-school/kubernetes.git
remote > cd kubernetes/02-Pod-Namespace/hands-on
```

```
remote > kubectl get pods
```

No resources found in default namespace.

```
remote > kubectl create -f pod-node-web.yaml --namespace whatwant
```

pod/node-web created

```
remote > kubectl get pods
```

No resources found in default namespace.

```
remote > kubectl get pods --namespace whatwant
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web	1/1	Running	0	25s	10.233.103.3	worker2	<none>	<none>

# Delete namespace

- pod를 갖고 있는 namespace를 삭제하면 어떻게 될까?

```
remote > kubectl get pods --namespace whatwant -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web	1/1	Running	0	58s	10.233.103.4	worker2	<none>	<none>

```
remote > kubectl delete namespace whatwant
```

```
namespace "whatwant" deleted
```

```
remote > kubectl get namespaces
```

NAME	STATUS	AGE
default	Active	24h
kube-node-lease	Active	24h
kube-public	Active	24h
kube-system	Active	24h

- 된다.

# Create namespace with CLI

- resource 생성할 때 YAML 이용하는 것이 좋지만, namespace는 CLI로 생성해도 괜찮을 정도로 simple 하다.

```
remote > kubectl create namespace whatwant
```

```
namespace/whatwant created
```

```
remote > kubectl get namespaces
```

NAME	STATUS	AGE
NAME	STATUS	AGE
default	Active	24h
kube-node-lease	Active	24h
kube-public	Active	24h
kube-system	Active	24h

```
remote > kubectl delete namespace whatwant
```

```
namespace "whatwant" deleted
```





**Tip #1**

**K8s Cheat Sheet**

# Kubernetes Cheat Sheet

```
> kubectl cluster-info
```

```
> kubectl get nodes
```

```
> kubectl get namespaces
```

```
> kubectl create -f <yaml file>
```

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

```
> kubectl run <name> --image <image>
```

```
> kubectl get pods
```

```
> kubectl logs <pod name>
```

```
> kubectl describe pod <pod name>
```

```
> kubectl delete pod <pod name>
```

```
> kubectl get pods -w
```

```
> kubectl get pods -o wide
```

```
> kubectl get pods -n <namespace>
```

```
> kubectl get pods -l <label>
```



**Error**

# Container Image missing - 1/2

Image를 찾지 못하는 경우 어떻게 해야 할까?

pod-error.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-error
  labels:
    env: trouble
spec:
  containers:
  - image: whatwant/err-pod:1.0
    name: pod-error
```

Q1. 'kubectl create' vs. 'kubectl apply' ?

Q2. 에러에 대한 상세 정보를 확인하려면 ?

Q3. Pod 삭제 직접 해보기 !

```
remote > cd advanced-kubernetes/02-week/Pod
```

```
remote > kubectl create -f ./pod-error.yaml
```

```
pod/pod-error created
```

```
remote > kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
pod-error	0/1	ImagePullBackOff	0	58s	10.233.103.75	worker2	<none>	<none>

# Container Image missing - 2/2

```
remote > kubectl describe pod pod-error
```

```
Name:          pod-error
Namespace:     default
Priority:       0
Node:          worker2/192.168.100.202
Start Time:    Fri, 31 Dec 2021 20:59:11 +0900
Labels:        env=trouble
Annotations:    cni.projectcalico.org/podIP: 10.233.103.75/32
                cni.projectcalico.org/podIPs: 10.233.103.75/32

Status:        Pending
IP:            10.233.103.75
IPs:
  IP: 10.233.103.75
Containers:
  pod-error:
    Container ID:
    Image:         whatwant/err-pod:1.0
    Image ID:
    Port:          <none>
    Host Port:     <none>
    State:         Waiting
      Reason:      ImagePullBackOff
    Ready:         False
    Restart Count: 0
    Environment:   <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from
      default-token-mdnvq (ro)
Conditions:
  Type          Status
  Initialized    True
  Ready          False
  ContainersReady False
  PodScheduled   True
```

## Volumes:

default-token-mdnvq:

Type: Secret (a volume populated by a Secret)

SecretName: default-token-mdnvq

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	7m32s	default-scheduler	Successfully assigned default/pod-error to worker2
Normal	Pulling	5m54s (x4 over 7m31s)	kubelet	Pulling image "whatwant/err-pod:1.0"
Warning	Failed	5m51s (x4 over 7m29s)	kubelet	Failed to pull image "whatwant/err-pod:1.0": rpc error: code = Unknown desc = Error response from daemon: pull access denied for whatwant/err-pod, repository does not exist or may require 'docker login': denied: requested access to the resource is denied
Warning	Failed	5m51s (x4 over 7m29s)	kubelet	Error: ErrImagePull
Warning	Failed	5m29s (x7 over 7m28s)	kubelet	Error: ImagePullBackOff
Normal	BackOff	2m29s (x20 over 7m28s)	kubelet	Back-off pulling image "whatwant/err-pod:1.0"





# 보충 수업

데이터센터 모놀리스에서  
클라우드 쿠버네티스로

# KCD Korea 2021

## Kubernetes Community Days Korea 2021

- <https://community.cncf.io/events/details/cncf-kcd-korea-presents-kubernetes-community-days-korea/>
- [https://www.youtube.com/playlist?list=PLj6h78yzYM2OO9\\_EWXS13LxAe-Bkn0xXt](https://www.youtube.com/playlist?list=PLj6h78yzYM2OO9_EWXS13LxAe-Bkn0xXt)



성민 이

Netflix

02:00 오후 — 02:40 오후

### Hermes | 데이터센터 모놀리스에서 클라우드 쿠버네티스로: 클라우드 네이티브로의 성공적인 이전을 위한 전략 | Netflix

잘 짜여진 모놀리스는 효율적이고 직관적이며 운용하기 편리한 합리적인 아키텍처입니다. 대부분의 작은 규모의 회사는 모놀리스로 서비스를 시작하지만 이 중 대부분은 서비스의 규모와 인력이 점차 늘어남에 따라 효율적인 아키텍처를 장기적으로 유지하는데 실패합니다. 오히려 모놀리스라는 이름의 수년간 쌓아온 프랑켄슈타인에 세번째 팔과 다섯번째 다리를 용접하고 있는 자신을 문득 깨닫게 되는 일도 드물지 않습니다. 뒤늦게 개발도 운용도 정상적으로 진행 할 수 없는 하향 나선의 상황에 직면에 있음을 깨닫게 되어도, 이미 하루에도 수만명이 이용하는 서비스를 근본적으로 손 대는 것 또한 좀처럼 엄두가 나지 않습니다. 우리는 수년간 쌓아 온 모놀리스 서비스를 해체하고 클라우드의 쿠버네티스로 옮길 수 있을까요? 어떻게하면 서비스를 사용하는 수 많은 유저들이 알아차리지 못하는 사이에 우리의 서비스를 안전하게 이전 할 수 있을까요? 클라우드 네이티브로의 이전은 과연 우리들의 Velocity에 큰 영향을 줄까요? 이 세션을 통해 데이터센터의 모놀리스를 클라우드의 쿠버네티스로 성공적으로 이전하기 위한 전략과 클라우드 네이티브 서비스의 장점, 그리고 주의점들을 짚어보려 합니다.

[발표자료] : [https://drive.google.com/file/d/1phjKwPQp7fSqyDeHRQFAbGn5D0\\_H\\_U9G/view](https://drive.google.com/file/d/1phjKwPQp7fSqyDeHRQFAbGn5D0_H_U9G/view)

[강의영상] : [https://www.youtube.com/watch?v=lyrg4x-A\\_Jk&list=PLj6h78yzYM2OO9\\_EWXS13LxAe-Bkn0xXt&index=5](https://www.youtube.com/watch?v=lyrg4x-A_Jk&list=PLj6h78yzYM2OO9_EWXS13LxAe-Bkn0xXt&index=5)



**<https://kahoot.it/>**



# 자습 (복습)

▷ Lab Setting

▷ K8s Pod

- Pod 심화: <https://speakerdeck.com/devinjeon/containerbuteo-dasi-salpyeoboneun-kubernetes-pod-dongjag-weonri>

▷ K8s Namespace