System Programming & OS 실습

Kubenetes

정지헌, 안석현, 김선재

Dankook University

{wlgjsjames7224, seokhyun, rlatjswo0824}@dankook.ac.kr

Index

- Kubernetes
- Kubernetes basic command
- Kubernetes components and resources

Kubernetes

❖ Kubernetes

- 클라우드 환경에서 쉽고 간편하게 개발할 수 있게 해주는 플랫폼

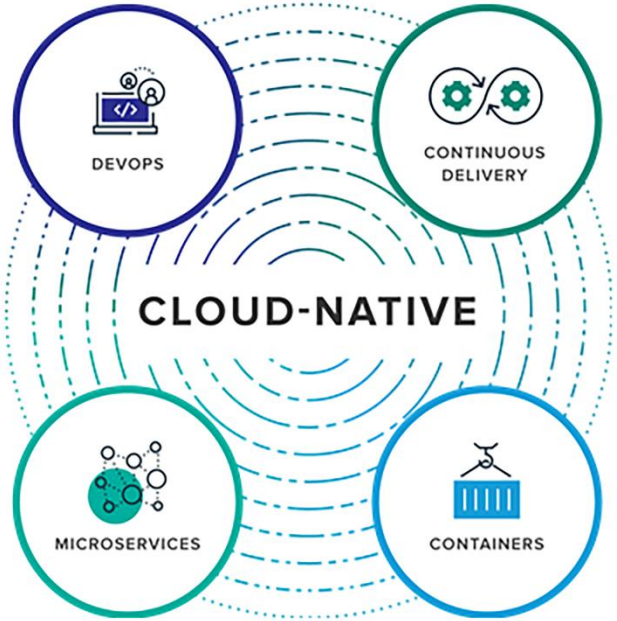
❖ Cloud Native

- Cloud의 장점을 최대한 활용할 수 있도록 애플리케이션을 개발하고 구축, 실행하는 방식
- 애플리케이션을 클라우드 환경에서 최적화된 방식으로 설계/개발/운영하는 방식

Kubernetes

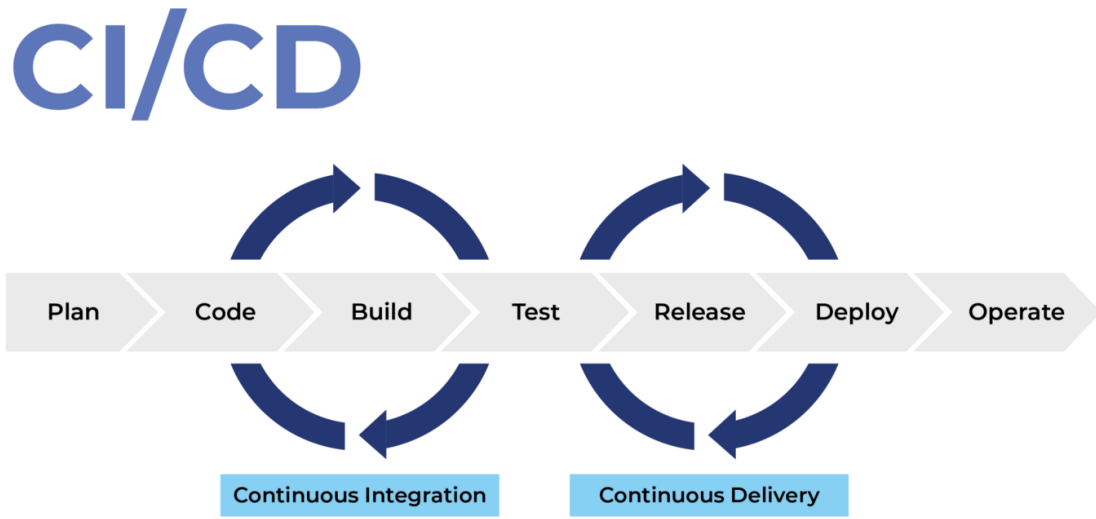
❖ Cloud Native

- 4가지 핵심 구성요소 CI/CD, DevOps, MicroServices, Container



❖ CI/CD – Continuous Integration/Continuous Deployment

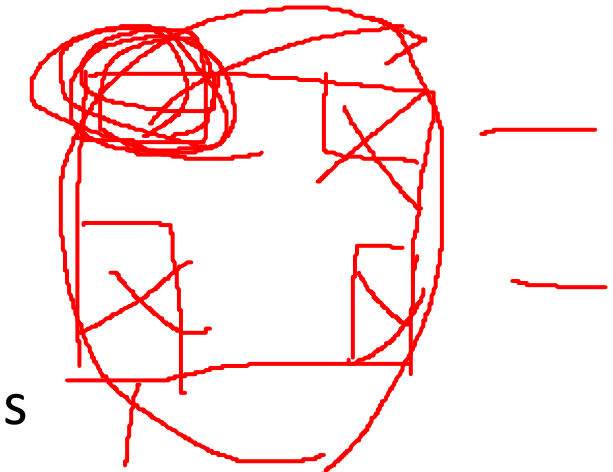
- 지속적인 통합과 배포를 통해 코드를 변경할 때마다 자동으로 적용
- 자동화된 테스트를 통해 신속하게 배포가 가능

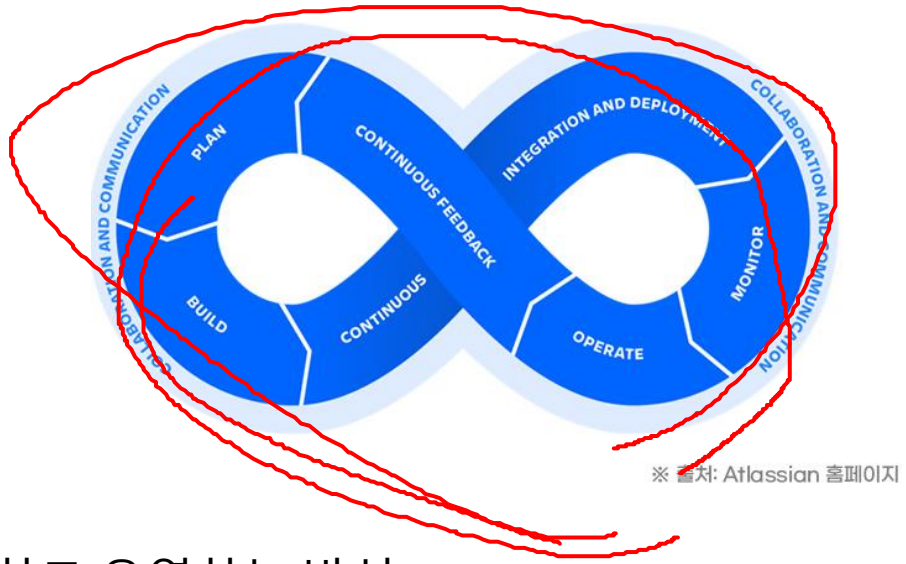


Kubernetes

❖ DevOps – Development + Operation

- 개발과 운영을 결합한 개념
- 개발조직과 운영조직의 긴밀한 협업/통합을 강조하는 개발 문화/프로세스/업무 프레임워크
- 소프트웨어의 품질과 출시 속도를 높이기 위한 방식





※ 출처: Atlassian 홈페이지

❖ Microservices

- 애플리케이션을 여러 개의 독립적인 서비스로 분리하여 개발하고 운영하는 방식
- 기존의 방식과 달리 각 서비스가 독립적으로 배포, 관리되기 때문에 유연성과 확장성이 높음

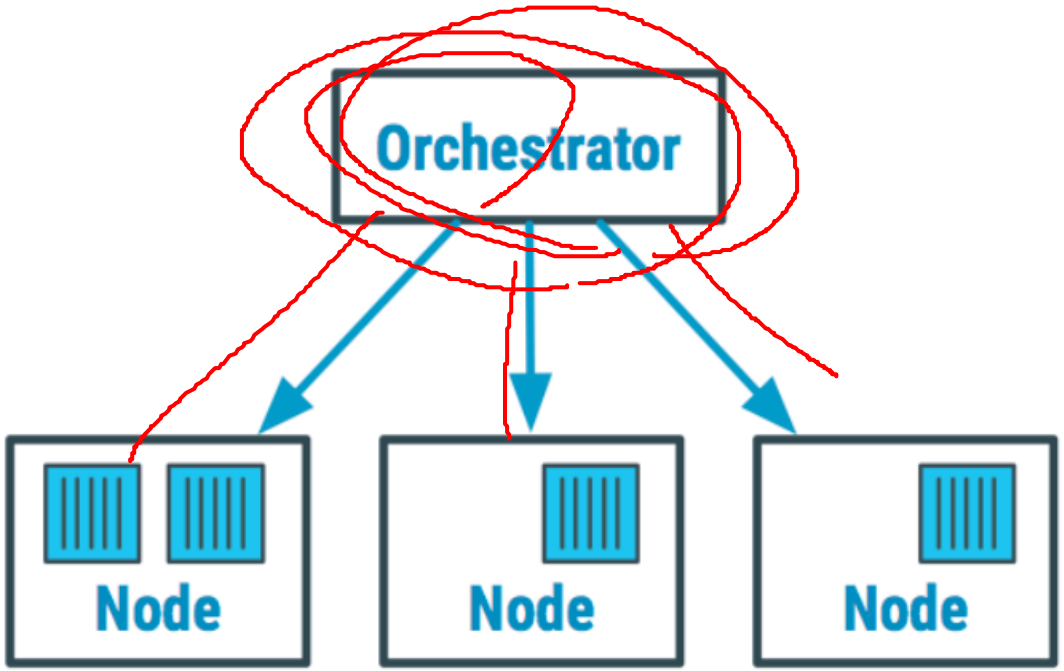
Kubernetes

❖ Contanier

- 일관된 실행 환경을 제공하며 애플리케이션이 어디서나 동일하게 실행될 수 있게 해줌
- 컨테이너마다 분리된 공간에 애플리케이션과 운영환경이 포함
- 서버보다 훨씬 가벼운 프로세스 수준의 가상서버로 마이크로서비스의 배포 및 실행환경으로 사용이 가능
- 어떠한 클라우드 환경에서도 즉각적으로 배포가 가능

Kubernetes

- **컨테이너 오케스트레이션이란?**
  - 컨테이너화 된 워크로드 및 서비스의 실행, 네트워킹, 운영을 자동화하는 기술
  - 컨테이너 오케스트레이션을 통해 서비스 구축 / 확장 / 예약 / 모니터링을 쉽게 수행 가능



출처 : https://devopedia.org/container-orchestration

Kubernetes

❖ 오케스트레이션의 필요성

- 확장성
- 보안강화
- 고가용성
- 생산성 향상

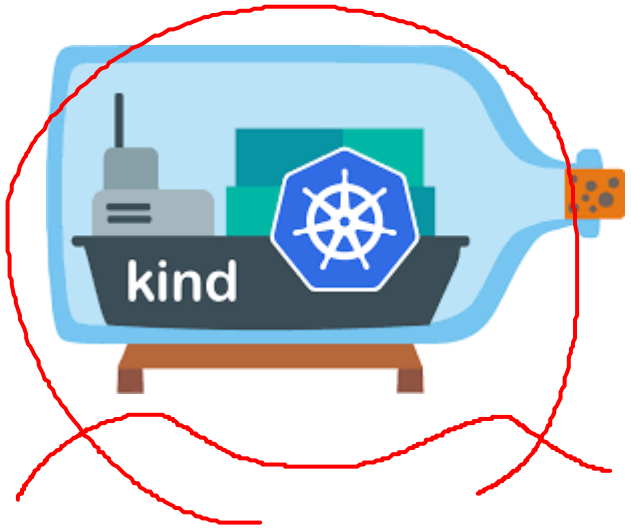
Kubernetes

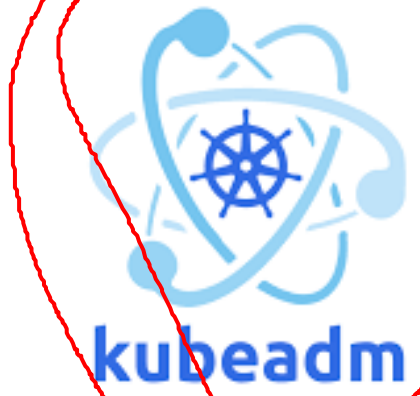
❖ Kubenetes의 특징

- 선언적 구성
  - 명시적으로 상태를 선언
- 확장성
  - 사용자가 원하는 리소스를 작성하여 클러스터 기능 확장에 용이
- 유연성
  - 클라우드 네이티브 플랫폼
- 마스터-워커 아키텍처

Kubernetes

- Kubernetes
  - Kind / Minikube / Kubeadm 등 다양하게 있음





Kubernetes

❖ Kubernetes 설치

- brew 명령어 설치 (https://brew.sh/)

```
==> Homebrew has enabled anonymous aggregate formulae and cask analytics.
Read the analytics documentation (and how to opt-out) here:
  https://docs.brew.sh/Analytics
No analytics data has been sent yet (nor will any be during this install run).

==> Homebrew is run entirely by unpaid volunteers. Please consider donating:
  https://github.com/Homebrew/brew#donations

==> Next steps:
- Run these two commands in your terminal to add Homebrew to your PATH:
    (echo; echo 'eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"') >> /home/seokhyun/.bashrc
    eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"
- Install Homebrew's dependencies if you have sudo access:
    sudo yum groupinstall 'Development Tools'
  For more information, see:
    https://docs.brew.sh/Homebrew-on-Linux
- We recommend that you install GCC:
    brew install gcc
- Run brew help to get started
- Further documentation:
    https://docs.brew.sh
```

Kubernetes

❖Kubernets 설치

```
[seokhyun@localhost home]$ (echo; echo 'eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"') >> /home/s
eokhyun/.bashrc
[seokhyun@localhost home]$ ^C
[seokhyun@localhost home]$ eval "$(/home/linuxbrew/.linuxbrew/bin/brew shellenv)"
[seokhyun@localhost home]$ sudo yum groupinstall 'Development Tools'
[sudo] password for seokhyun:
Last metadata expiration check: 2:48:22 ago on Sat 07 Sep 2024 07:07:01 PM KST.
Dependencies resolved.
================================================================================
```

```
[seokhyun@localhost home]$ brew install hello
==> Downloading https://ghcr.io/v2/homebrew/core/hello/manifests/2.12.1
######################################################################## 100.0%
==> Fetching dependencies for hello: linux-headers@5.15, glibc, gmp, isl, mpfr, libmpc, lz4, xz, zlib, zstd
, binutils and gcc
==> Downloading https://ghcr.io/v2/homebrew/core/linux-headers/5.15/manifests/5.15.166
######################################################################## 100.0%
==> Fetching linux-headers@5.15
==> Downloading https://ghcr.io/v2/homebrew/core/linux-headers/5.15/blobs/sha256:68cf92d4ae632182e97b759e2d
######################################################################## 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/glibc/manifests/2.35_1-1
######################################################################## 100.0%
==> Fetching glibc
```

Kubernetes

❖Kubernets 설치

```
[seokhyun@localhost home]$ brew install minikube
==> Downloading https://formulae.brew.sh/api/formula.jws.json
######################################################################################################## 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/minikube/manifests/1.33.1-1
######################################################################################################## 100.0%
==> Fetching dependencies for minikube: kubernetes-cli
==> Downloading https://ghcr.io/v2/homebrew/core/kubernetes-cli/manifests/1.31.0
######################################################################################################## 100.0%
==> Fetching kubernetes-cli
==> Downloading https://ghcr.io/v2/homebrew/core/kubernetes-cli/blobs/sha256:c125853b863e89cfd9777a5613dfc4
######################################################################################################## 100.0%
==> Fetching minikube
==> Downloading https://ghcr.io/v2/homebrew/core/minikube/blobs/sha256:5598f962e772b41a19c95d533708b09e872d
######################################################################################################## 100.0%
==> Installing dependencies for minikube: kubernetes-cli
==> Installing minikube dependency: kubernetes-cli
==> Downloading https://ghcr.io/v2/homebrew/core/kubernetes-cli/manifests/1.31.0
Already downloaded: /home/seokhyun/.cache/Homebrew/downloads/5b8afb8ec25c952fd913e4cd975ee0e484b26daa1fb9d8
b4edccd74ea67a829c--kubernetes-cli-1.31.0.bottle_manifest.json
==> Pouring kubernetes-cli--1.31.0.x86_64_linux.bottle.tar.gz
==> Downloading https://formulae.brew.sh/api/cask.jws.json
######################################################################################################## 100.0%
🍺  /home/linuxbrew/.linuxbrew/Cellar/kubernetes-cli/1.31.0: 237 files, 54.7MB
==> Installing minikube
```

Kubernetes

❖Kubernets 설치

```
[seokhyun@localhost home]$ sudo yum install -y yum-utils
sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
Last metadata expiration check: 3:01:46 ago on Sat 07 Sep 2024 07:07:01 PM KST.
Dependencies resolved.
================================================================================
 Package                 Architecture          Version                    Repository              Size
================================================================================
Installing:
 yum-utils               noarch                4.3.0-16.el9               baseos                  40 k

Transaction Summary
================================================================================
Install  1 Package
```

-> https://docs.docker.com/engine/install/centos/

Kubernetes

❖Kubernets 설치

```
[seokhyun@localhost home]$ sudo yum install docker-ce docker-ce-cli containerd.io docker-buildx-plugin dock
er-compose-plugin
Docker CE Stable - x86_64                                  106 kB/s |  55 kB     00:00
Dependencies resolved.
================================================================================
 Package                    Architecture   Version                 Repository                Size
================================================================================
Installing:
 containerd.io              x86_64         1.7.21-3.1.el9          docker-ce-stable          43 M
 docker-buildx-plugin       x86_64         0.16.2-1.el9            docker-ce-stable          14 M
 docker-ce                  x86_64         3:27.2.0-1.el9          docker-ce-stable          27 M
 docker-ce-cli              x86_64         1:27.2.0-1.el9          docker-ce-stable         7.9 M
 docker-compose-plugin      x86_64         2.29.2-1.el9            docker-ce-stable          13 M
Installing weak dependencies:
 docker-ce-rootless-extras  x86_64         27.2.0-1.el9            docker-ce-stable         4.0 M

Transaction Summary
================================================================================
Install  6 Packages
```

Kubernetes

❖Kubernets 설치

```
[seokhyun@localhost home]$ sudo systemctl start docker
[seokhyun@localhost home]$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:53cc4d415d839c98be39331c948609b659ed725170ad2ca8eb36951288f81b75
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
```

Kubernetes

❖Kubernets 설치

```
[seokhyun@localhost home]$ sudo chmod 666 /var/run/docker.sock
[seokhyun@localhost home]$ minikube start --driver=docker --container-runtime=containerd
* minikube v1.33.1 on Centos 9
* Using the docker driver based on user configuration
* Using Docker driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.44 ...
* Downloading Kubernetes v1.30.0 preload ...
    > gcr.io/k8s-minikube/kicbase...:  481.58 MiB / 481.58 MiB  100.00% 21.82 M
    > preloaded-images-k8s-v18-v1...:  375.69 MiB / 375.69 MiB  100.00% 13.35 M
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Preparing Kubernetes v1.30.0 on containerd 1.6.31 ...
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - 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 default
```

Kubernetes

❖Kubernets 설치 확인

- kubectl get pod –A
  - Kubenetes 기본 명령어로 pod의 주요 정보를 출력하는 명령어

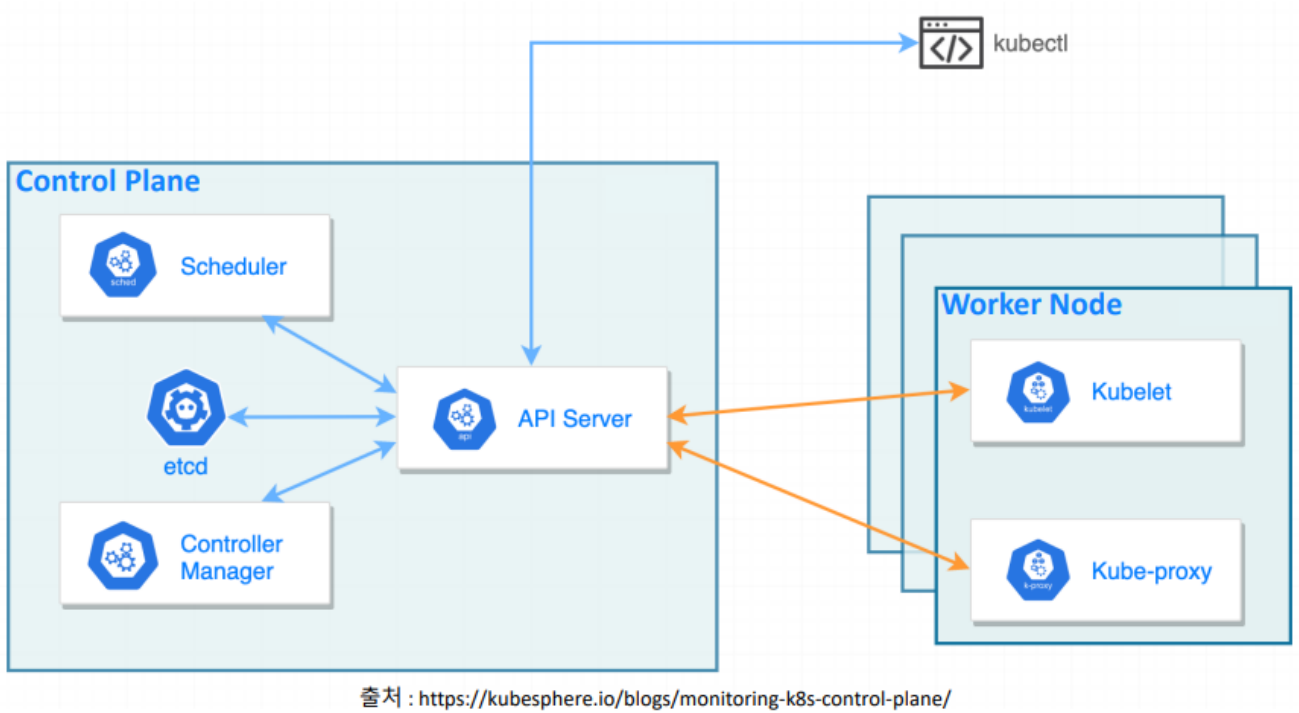
```
[seokhyun@localhost home]$ kubectl get pod -A
NAMESPACE     NAME                               READY   STATUS    RESTARTS      AGE
kube-system   coredns-7db6d8ff4d-9hrw5           1/1     Running   0             47s
kube-system   etcd-minikube                      1/1     Running   0             60s
kube-system   kindnet-vclqt                      1/1     Running   0             47s
kube-system   kube-apiserver-minikube            1/1     Running   0             60s
kube-system   kube-controller-manager-minikube   1/1     Running   0             60s
kube-system   kube-proxy-df7bh                   1/1     Running   0             47s
kube-system   kube-scheduler-minikube            1/1     Running   0             60s
kube-system   storage-provisioner                1/1     Running   1 (16s ago)   58s
```

Kubernetes 구성요소

❖Kubernets 클러스터

- **Control Plane**
  - 스케줄링 수행
  - 클러스터 이벤트 감지 및 처리
  - Worker node & pod 관리

- **Worker node**
  - 동작 중인 pod 유지
  - 런타임 환경 제공

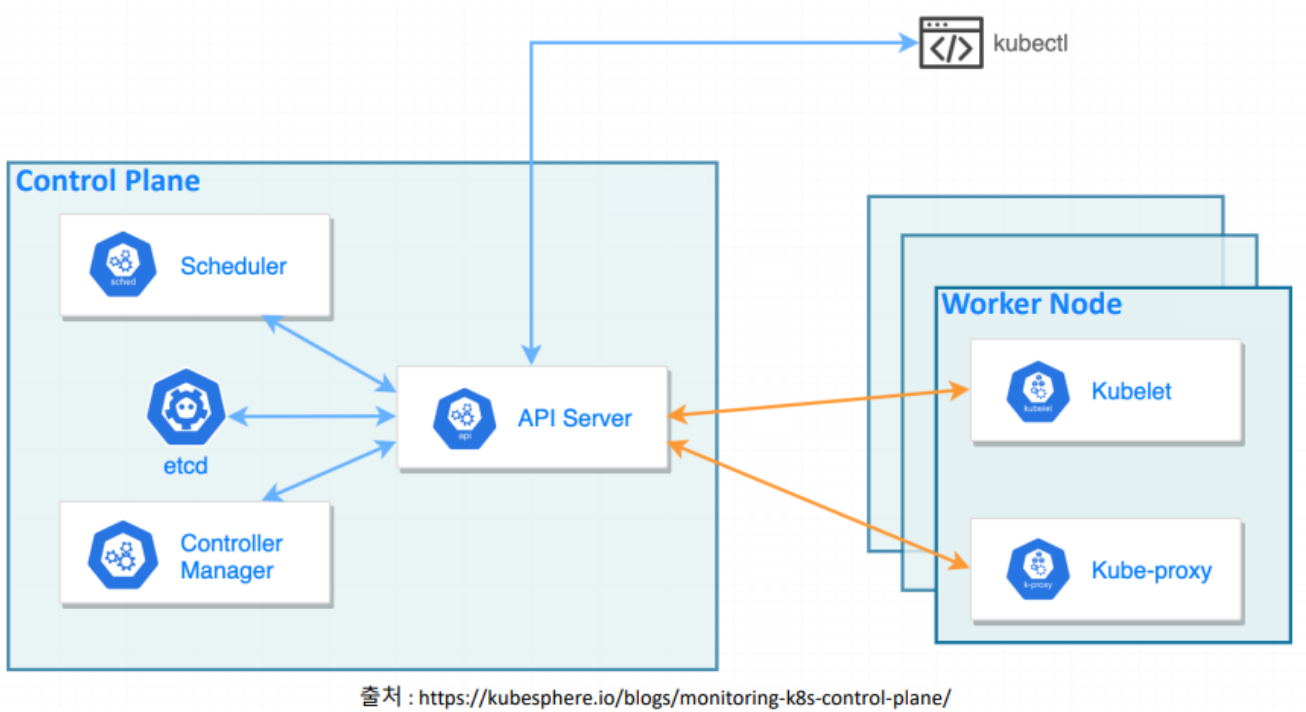


출처 : https://kubesphere.io/blogs/monitoring-k8s-control-plane/

Kubernetes 구성요소

❖ Control Plane 구성요소

- **Kube-apiserver**
  - 사용자가 클러스터와 상호작용하는 API
- **Etcd**
  - 클러스터의 정보를 보관하는 키-값 저장소
- **Kube-scheduler**
  - 노드가 배정되지 않은 Pod 감지
  - 노드에 pod 할당
- **Kube-controller-manager**

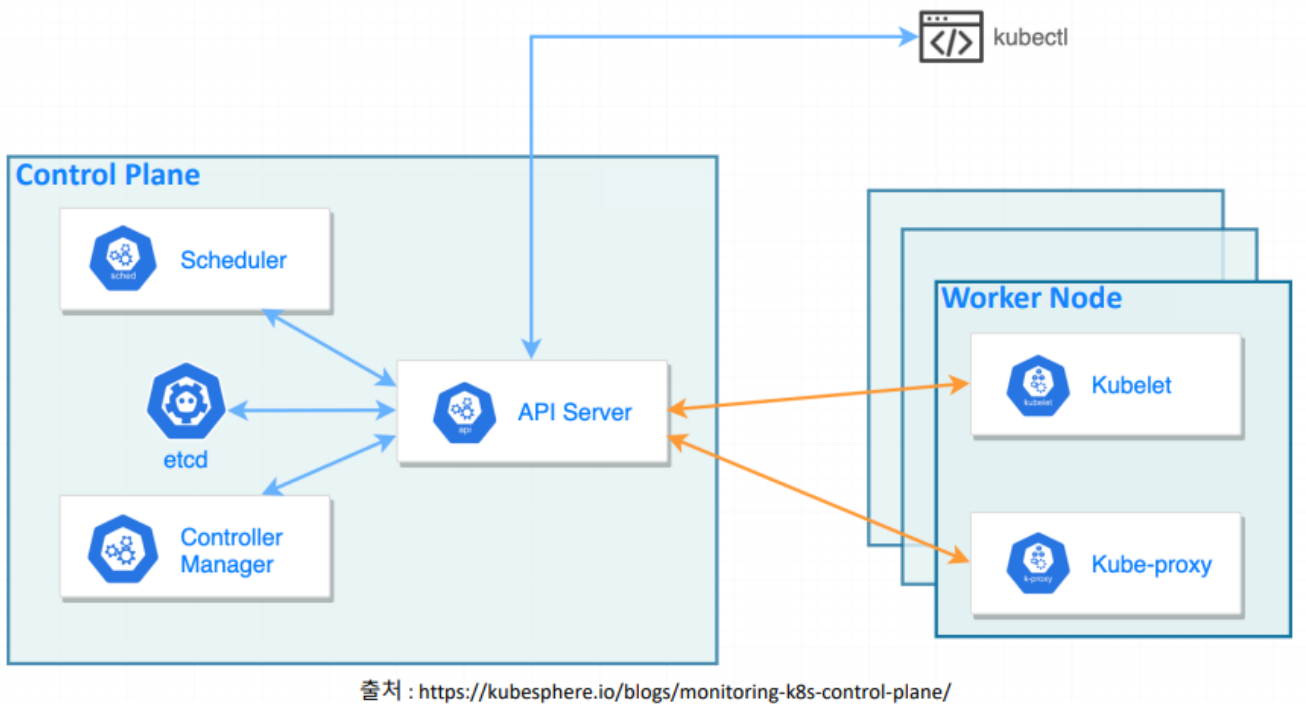


출처 : https://kubesphere.io/blogs/monitoring-k8s-control-plane/

Kubernetes 구성요소

❖ Worker Node 구성요소

- **kubelet**
  - **Pod에서 컨테이너가 동작하도록 관리**
  - **Pod 스펙에 따라 동작하도록 관리**
- **Kube-proxy**
  - 네트워크 규칙을 유지 및 관리
    - 해당 네트워크 규칙이 내부 네트워크 세션이니 클러스터 외부에서 pod로 접근 가능하게 함
- **Container Runtime**
  - 컨테이너 실행 담당
  - Containerd, CRI-O 등 존재



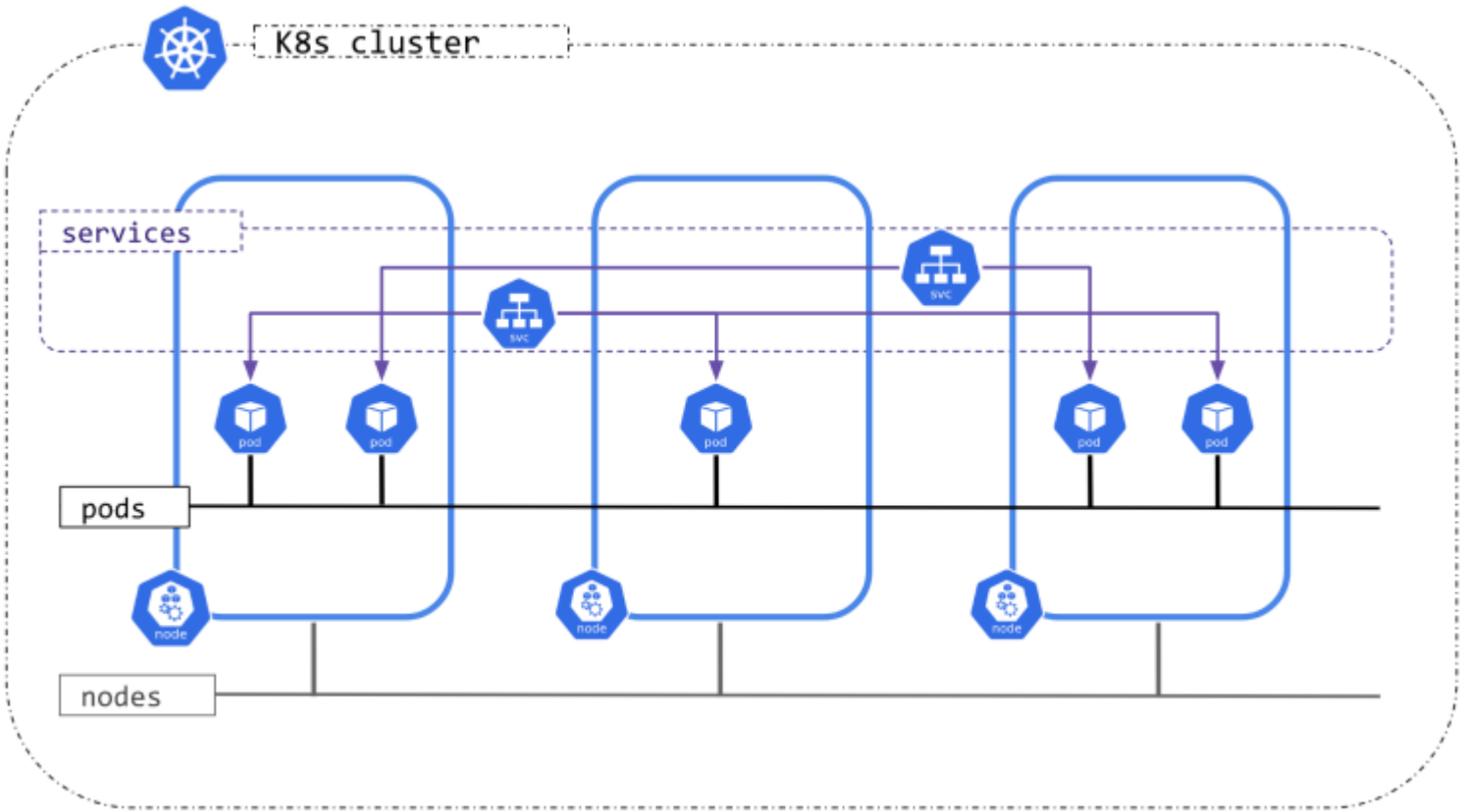
출처 : https://kubesphere.io/blogs/monitoring-k8s-control-plane/

Kubernetes

❖ Kubernetes의 Networking

- 결합된 컨테이너간 Networking
  - 동일한 가상 네트워크 인터페이스 공유
- Pod와 Pod 간의 Networking
  - CNI(Container Networking Interface)

  를 통해 이루어짐
- Pod와 외부 간의 Networking
  - Service를 통해 이루어짐



Kubernetes 실습

❖ Kubectl

- Kubernetes API를 사용하여 Control Plane과 통신하는 도구

```
kubectl [command] [TYPE] [NAME] [flags]
```

- 주요 명령어
  - get, create, apply, run, describe, edit, logs, exec, delete

Kubernetes 실습

❖ 자주 쓰이는 옵션

- -n (namespace)
  - 특정 명령어를 실행할 때 namespace 지정
- -A (all-namespace)
  - 모든 namespace에서 명령어를 실행
- -f (filename)
  - 특정 파일을 지정하여 명령어 실행
- -o (output)
  - 출력 형식 변경할 때 사용

Kubernetes 실습

❖ get

- 한 개 이상의 리소스를 출력하는 명령
- Pod, service, NameSpace(NS) 등 리소스 출력
- Kubectl get pod
  - 현재의 default namespace에서 pod의 목록 출력

```
[seokhyun@localhost ~]$ kubectl get pod -A
NAMESPACE     NAME                                   READY   STATUS    RESTARTS        AGE
kube-system   coredns-7db6d8ff4d-9hrw5               1/1     Running   0               2d19h
kube-system   etcd-minikube                          1/1     Running   0               2d19h
kube-system   kindnet-vclqt                          1/1     Running   0               2d19h
kube-system   kube-apiserver-minikube                1/1     Running   0               2d19h
kube-system   kube-controller-manager-minikube       1/1     Running   0               2d19h
kube-system   kube-proxy-df7bh                       1/1     Running   0               2d19h
kube-system   kube-scheduler-minikube                1/1     Running   0               2d19h
kube-system   storage-provisioner                    1/1     Running   1 (2d19h ago)   2d19h
[seokhyun@localhost ~]$ kubectl get pod -o wide -A
NAMESPACE     NAME                                   READY   STATUS    RESTARTS        AGE     IP              NODE
kube-system   coredns-7db6d8ff4d-9hrw5               1/1     Running   0               2d19h   10.244.0.2      minikube
kube-system   etcd-minikube                          1/1     Running   0               2d19h   192.168.49.2    minikube
kube-system   kindnet-vclqt                          1/1     Running   0               2d19h   192.168.49.2    minikube
kube-system   kube-apiserver-minikube                1/1     Running   0               2d19h   192.168.49.2    minikube
kube-system   kube-controller-manager-minikube       1/1     Running   0               2d19h   192.168.49.2    minikube
kube-system   kube-proxy-df7bh                       1/1     Running   0               2d19h   192.168.49.2    minikube
kube-system   kube-scheduler-minikube                1/1     Running   0               2d19h   192.168.49.2    minikube
kube-system   storage-provisioner                    1/1     Running   1 (2d19h ago)   2d19h   192.168.49.2    minikube
[seokhyun@localhost ~]$ kubectl get pod -n kube-system
NAME                                   READY   STATUS    RESTARTS        AGE
coredns-7db6d8ff4d-9hrw5               1/1     Running   0               2d19h
etcd-minikube                          1/1     Running   0               2d19h
kindnet-vclqt                          1/1     Running   0               2d19h
kube-apiserver-minikube                1/1     Running   0               2d19h
kube-controller-manager-minikube       1/1     Running   0               2d19h
kube-proxy-df7bh                       1/1     Running   0               2d19h
kube-scheduler-minikube                1/1     Running   0               2d19h
storage-provisioner                    1/1     Running   1 (2d19h ago)   2d19h
```

- 주요 옵션
  - -A : 모든 NS의 pod 목록 출력
  - -o : 출력 형식을 지정
  - -n : kube-system namespace 내의 pod 목록 출력

Kubernetes 실습

❖ create

- 입력이나 파일을 통해 리소스를 생성하는 명령
- kubectl create –f Filename [options]
- kubectl create ns <Namespace>

```
[seokhyun@localhost ~]$ kubectl create ns test
namespace/test created
[seokhyun@localhost ~]$ kubectl get ns
NAME              STATUS   AGE
default           Active   2d19h
kube-node-lease   Active   2d19h
kube-public       Active   2d19h
kube-system       Active   2d19h
test              Active   6s
```

- 주요 옵션
  - -f (filename)
    - Kubectl create –f <filename>
    - 파일, 디렉토리, URL을 지정하여 리소스 생성

```
[seokhyun@localhost ~]$ kubectl create -f nginx-pod.yaml
pod/my-nginx-pod created
[seokhyun@localhost ~]$ kubectl get pod -A
NAMESPACE     NAME                                READY   STATUS    RESTARTS      AGE
default       my-nginx-pod                        1/1     Running   0             18s
kube-system   coredns-7db6d8ff4d-9hrw5            1/1     Running   0             3d2h
kube-system   etcd-minikube                       1/1     Running   0             3d2h
kube-system   kindnet-vclqt                       1/1     Running   0             3d2h
kube-system   kube-apiserver-minikube             1/1     Running   0             3d2h
kube-system   kube-controller-manager-minikube    1/1     Running   0             3d2h
kube-system   kube-proxy-df7bh                    1/1     Running   0             3d2h
```

Kubernetes 실습

❖ apply

- 파일이나 입력을 통해 리소스 구성을 추가하는 명령
- Kubectl apply (-f Filename | -k Directory) [option]
  - File 또는 directory의 내용을 토대로 리소스 구성
    - File 단위, directory 단위로 추가 가능

```
[seokhyun@localhost ~]$ vim nginx-pod.yaml
[seokhyun@localhost ~]$ cat nginx-pod.yaml
apiVersion : v1
kind : Pod
metadata :
  name : my-nginx-pod
spec :
  container :
    - name : my-nginx-container
      image : nginx:latest
  ports :
    - containerPort : 80
      protocol : TCP
```

```
[seokhyun@localhost ~]$ cat nginx.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginxdemos/hello:plain-text
    ports:
    - name: http
      containerPort: 80
      protocol: TCP
```

```
[seokhyun@localhost ~]$ kubectl apply -f nginx-pod.yaml
pod/my-nginx-pod unchanged
```

- 주요 옵션
  - -f (filename)
    - Kubectl apply –f <filename>
    - 파일, 디렉토리, url 지정하여 리소스 생성/변경

Kubernetes 실습

- **describe**
  - 특정 리소스나 그룹의 세부정보를 출력

$ *kubectl describe (-f FILENAME | TYPE [NAME_PREFIX | -l label] | TYPE/NAME) [options]*

$ *kubectl describe pod nginx*
  - nginx pod의 상세 정보 출력

- 주요 옵션
  - -A

    $ *kubectl describe pod nginx -A*
    - 모든 namespace의 nginx pod 세부 정보를 출력

```
[seokhyun@localhost ~]$ kubectl get nodes
NAME       STATUS   ROLES           AGE    VERSION
minikube   Ready    control-plane   3d2h   v1.30.0
[seokhyun@localhost ~]$ kubectl describe pods my-nginx-pod
Name:             my-nginx-pod
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Wed, 11 Sep 2024 00:52:22 +0900
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.244.0.3
IPs:
  IP:  10.244.0.3
Containers:
  my-nginx-container:
    Container ID:   containerd://a923db02492b36190fdd2da626cda7b55663a916dd5b2e4f8e1f75501cc6bf74
    Image:          nginx:latest
    Image ID:       docker.io/library/nginx@sha256:04ba374043ccd2fc5c593885c0eacddebabd5ca375f9323666f28dfd5a9
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Wed, 11 Sep 2024 00:52:30 +0900
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-dh4zz (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
```