(1) 쿠버네티스 클러스터 기초

▼ 목차

❤️클러스터 아키텍처❤️

Master node

: 중앙에서 manage, plan, schedule, monitor

- **kube-aipserver**
  - 클러스터 내에서 모든 작업을 오케스트레이션함
- **ETCD cluster**
  - 관련 정보들이 저장됨
  - 키-값 형식으로 값들을 저장하는 데이터베이스
- **kube scheduler**
  - 노드의 응용프로그램/컨테이너의 스케줄을 짬
  - 컨테이너를 설치하기 위한 올바른 노드를 식별함
- **Kube Controller-Manager**

  : 다양한 controller 존재

  예시)
  - **Node-controller**
    - 새 노드를 클러스터에 온보딩, 노드가 사용 불가능/파괴되는 상황 처리

Worker nodes

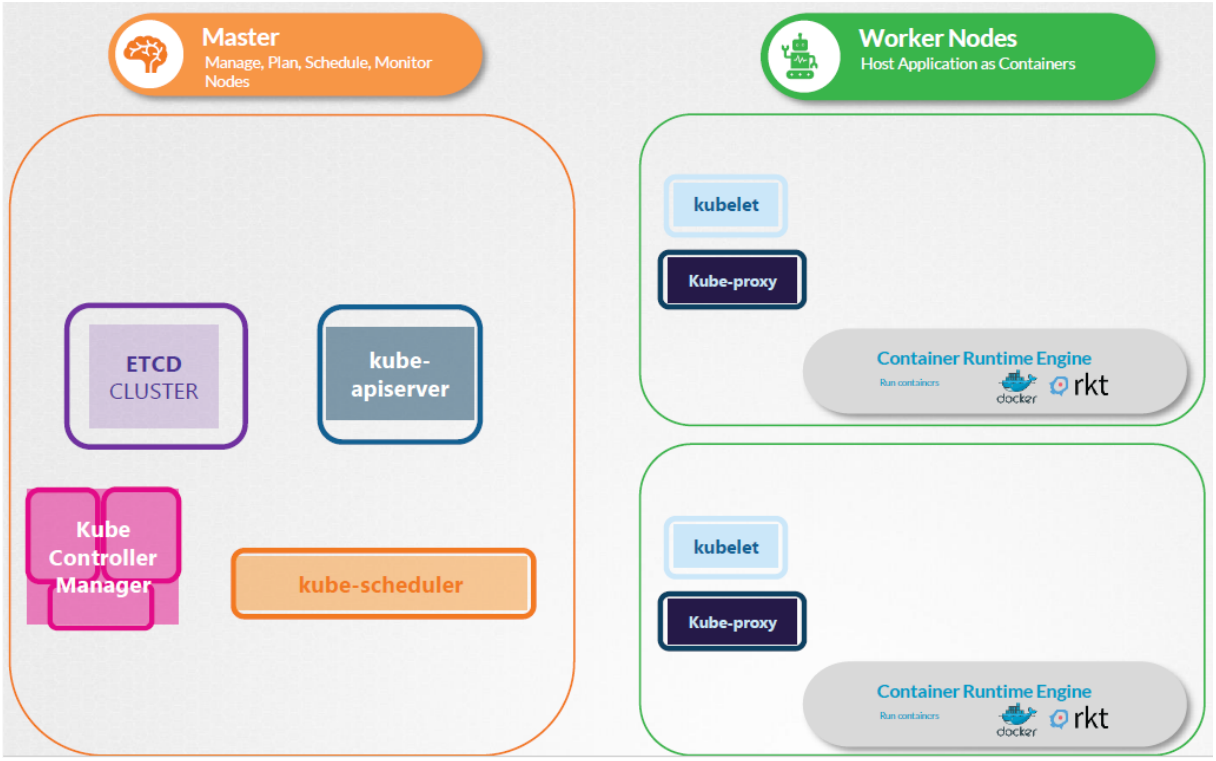
: Host application as containers

- **Container runtime engine**

  ex) 도커, rkt 등
- **kubelet**
  - kube-apiserver의 지시를 듣고 컨테이너, kube-proxy를 관리, 필요한대로 노드에서 컨테이너를 삭제하거나 배포함

    → 클러스터 내부의 서비스 간 통신을 가능하게 함
  - kube-apiserver에서는 노드와 컨테이너의 상태를 모니터하기 위해, 주기적으로 kubelet에서 상태 보고서를 가져옴
- **kube-proxy**
  - worker node 간의 통신을 가능하게 함
  - work node에 필요한 규칙이 실행되도록 함 (노드 위에서 실행되고 있는 컨테이너 간의 연결이 가능하게 함)

- **Replication-controller**
  - 원하는 컨테이너 수만큼 복제 및 실행될 수 있도록 관리함



🧡Docker vs Containerd🧡

💡 한동안 쿠버네티스는 도커만 지원했고, 이후 도커가 아닌 다른 rkt 같은 서비스들도 지원하기 위해 생겨난 것이 CRI 임

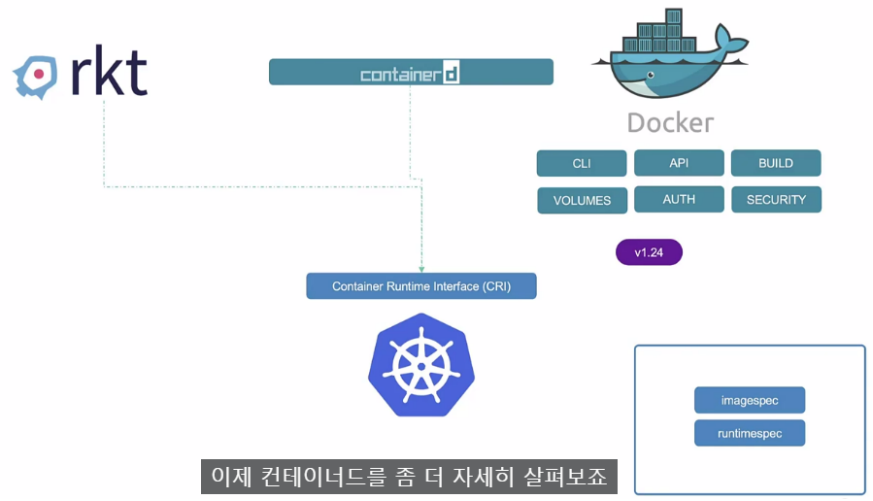
Container Runtime Interface (CRI)

- 도커가 아니더라도 OCI 표준을 따른다면, 쿠버네티스의 컨테이너 런타임으로 작업할 수 있도록 해줌
  - **OCI (Open Container Initiative)**

    : 컨테이너 기술에 대한 표준화
    - **imagespec**

      : image를 어떻게 만드는지 정의함 (image 빌드 방식에 대한 기준 정의)
    - **runtimespec**

      : 초기 컨테이너 runtime이 어떻게 개발되어야 하는지 정의함



dockershim

- CRI 밖에서 도커만을 계속 지원하기 위한 방법

> 부담이 커져 이후(ver1.24)에는 지원 종료함

Docker의 containerd

- CRI 호환이 가능하고, 다른 런타임처럼 쿠버네티스와 직접적으로 작업할 수 있음
- 도커와 별도의 런타임으로도 사용 가능함

> 기존에는 도커의 일부였지만, 지금은 분리됨

Containerd의 CLI

Purpose	Debugging	General Purpose	Debugging
Community	ContainerD	ContainerD	Kubernetes
Works With	ContainerD	ContainerD	All CRI Compatible Runtimes

1. ctr

- **containerd의 가장 기본적인 cli** (containerd 설치하면 기본적으로 깔림)
- 디버깅 목적으로만 사용, 한정된 기능만 지원함
- 사용자 친화적이지 X

  → 제품 환경에서 컨테이너를 실행/관리할 때 사용되지 X
- 사용 예시

```
# redis 이미지 불러오기
$ ctr images pull [이미지 주소]
```

2. nerdtctl

- ctr보다 향상됨
- **containerd에서 도커와 비슷한 CLI 환경 사용할 수 있도록 지원**함

  (도커가 지원하는 대부분의 옵션 지원)
- docker compose 지원
- containerd의 최신 기능 지원

  예시)
  - 암호화된 container images
  - Lazy Pulling
  - P2P image distribution

```
# 컨테이너 실행하기
$ ctr run [이미지 주소]
```

3. crictl

- CRI 호환 가능한 컨테이너 런타임과 상호작용하는 데 사용되는 cli
- **쿠버네티스 관점에서 다른 다양한 컨테이너 런타임에 걸쳐 작동**함

  → 쿠버네티스에서 개발하고 관리

  → 별도 설치 필요
- inspect/debug containger 런타임으로 주로 사용됨

  (컨테이너 생성이 용이하지는 않기 때문)
- 사용 예시

```
$ crictl
$ crictl pull busybox
$ crictl images
$ crictl ps -a
$ crictl exec -i it [컨테이너 IT] ls
$ crictl logs [컨테이너 IT] ls
# docker와 달리 pod를 인식하므로
# 명령을 실행하면 pod를 리스팅할 수 있음
$ crictl pods
```

  - Image signing and verifying
  - Namespace in Kubernetes (도커에서는 사용 불가)
- 사용 예시

```
# docker의 명령어들을 다음과 같이 유사하게 사용할 수 있음

# docker
$ nerdctl

# docker run --name redis:alpine
$ nerdctl run --name redis redis:alpine

# docker run --name webserver -p 80:80 -d nginx
$ nerdctl run --name webserver -p 80:80 -d nginx
```