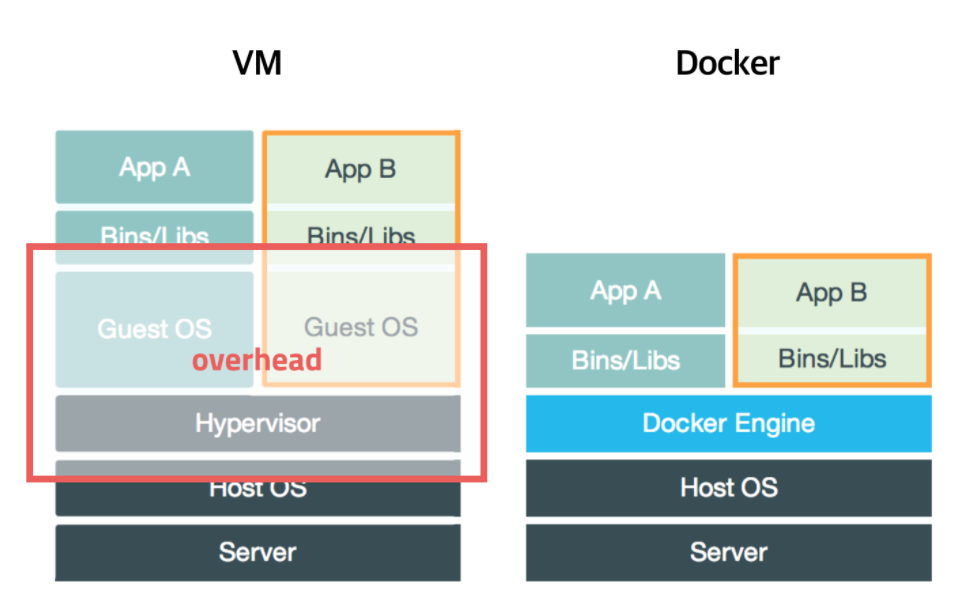
**Docker 란?**

리눅스 컨테이너 기반의 오픈소스 가상화 플랫폼

다양한 프로그램, 실행환경을 컨테이너로 추상화하고 동일 인터페이스를 제공하여 프로그램의 배포 및 관리를 단순하게 함

VM과 Docker의 차이



* 가상머신은 항상 게스트 OS를 설치해 Hypervisor를 통해 OS 및 커널이 통째로 가상화 되어 성능이 떨어짐
* Docker는 서버 운영을 위한 프로그램과 라이브러리를 따로 설치, 하드웨어를 가상화하는 hypervisor 이 없어 성능이 좋음

**컨테이너 란?**

격리된 공간에서 프로세스가 동작하는 가상화 기술

Hypervisor는 OS 및 커널이 통째로 가상화됨, Container는 filesystem이 가상화됨

**이미지 란?**

컨테이너에 필요한 파일과 설정 값 등을 저장한 것, 같은 이미지에서 여러 개의 컨테이너를 생성할 수 도 있음

이미지는 컨테이너를 실행하기 위한 모든 정보를 가지고 있어 서버를 추가하는 작업을 해도 이미지로 컨테이너를 생성 및 실행하면 됨

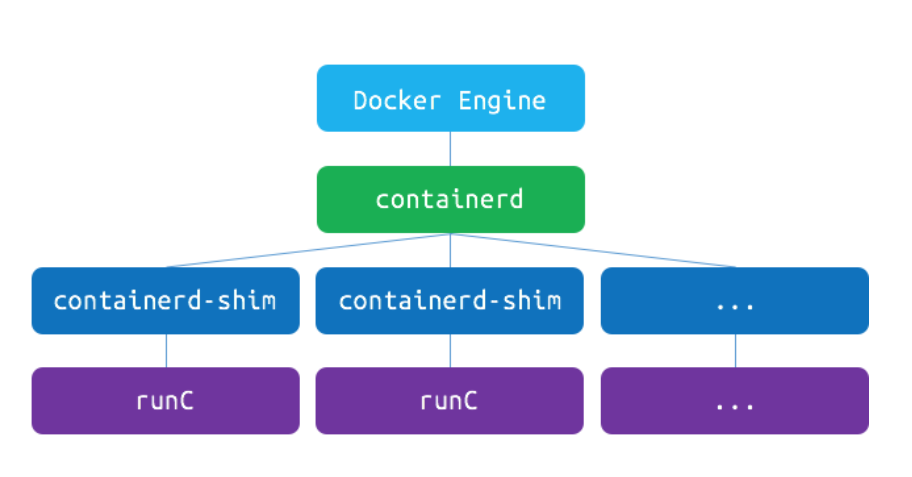
* 즉, 컨테이너는 이미지를 실행한 상태를 말하며 이미지를 여러 개의 컨테이너로 만들 수 있는 하나의 격리된 공간, Docker는 Dockerfile파일을 이용해 컨테이너를 구성할 수 있음
* Immutable Infrastructure : 이미지 기반 어플리케이션 배포, 한 번 설정하고 변경하지 않는 이미지 기반의 어플리케이션 배포 패러다임
* 관리가 편하고 확장성이 높다. 테스트가 간변해지며 개발 환경이 가벼워 진다.

**Docker Hub 란?**

Docker의 큰 이미지 용량을 Docker Hub를 통해 관리

**컨테이너 기술 및 작동 방식**

1. Namespace : VM에서 각 게스트 머신 별로 독립적인 공간을 제공하여 충돌이 없게 하는 기능인 namspaces 기능을 커널에 내장
2. cgroups (Control Groups) : 자원에 대한 제어를 가능하게 해주는 커널의 기능
3. Docker 구조



* Containerd-shim으로 나눈 각 어플리케이션들을 containerd에서 관리하고, 각 이미지, 네트워크 등의 관리는 Docker Engline에서 이루어진다. 각 독립적인 프로세스로 작동됨

**사용 이유**

* 소프트웨어 버전과 개별 구성과 관련된 세부 업무를 배포 및 관리가 수월해짐
* 게스트 OS를 설치하지 않아 성능이 좋고 메모리가 경량화된 방식이다.

**Docker 설치**

Sudo yum undate (업데이트)

Sudo yum install yum-utils device-mapper-persistent-data lvm2

(항상 최신버전이 아닐 수 있기 때문에 docker의 저장소를 생성해서 이곳에 설치한다.)

…………………………………////////////////////////////////////////////////////////////////////……………………………………

Sudo yum-config-manager –add-repo <https://download.docker.com/linux/centos/docker-ce.repo>

(저장소 추가)

Sudo yum install docker-ce (Docker CE(Community Edition) 최신 버전 설치)

Sudo systemctl start docker (docker 실행)

Sudo systemctl status docker (docker 상태)

Sudo systemctl enable docker (부팅 시 docker 자동 실행 설정)

Sudo usermod -aG docker $USER

(docker 그룹에 현재 접속중인 사용자 계정 추가, root 권한이 아니어도 docker명령어 실행 가능)

**Docker 사용법**

docker -v (현재 버전 확인)

sudo docker run hello-world (/hello 컨테이너 확인)

sudo docker ps -a (컨테이너 확인)