LESSON 1

도커의 개념

도커의 개념

Build-Ship-Run

DEVELOPER WORKFLOWS

Docker allows you to compose your application from microservices, without worrying about inconsistencies between development and production environment, and without locking into any platform or language.

REGISTRY SERVICES

Docker lets you design the entire cycle of application development, testing and distribution, and manage it with a consistent user interface.

MANAGEMENT

Docker offers you the ability to deploy scalable services, securely and reliably, on a wide variety of platforms.

DOCKER ENGINE

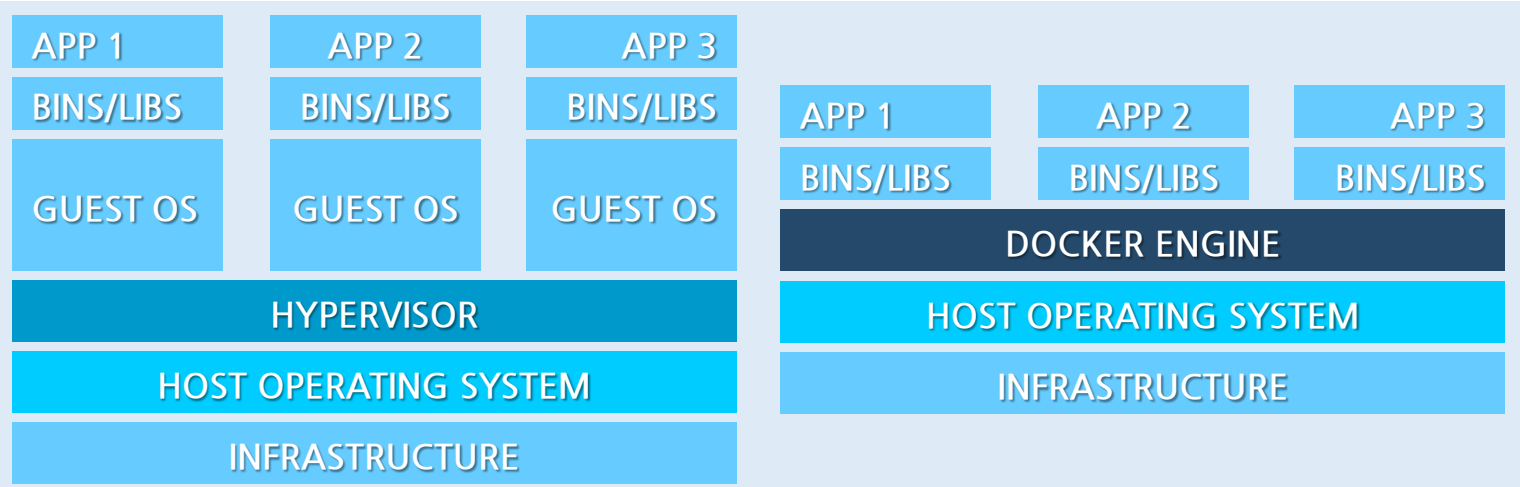
INFRASTRUCTURE

도커의 개념

컨테이너 기반 가상화

기존의 가상화와 다른 개념

- 하드웨어 가상화가 아닌 실행환경의 분리(isolation)
- 각 컨테이너간 영향을 분리



<하이퍼바이저 / 도커>

도커의 개념

도커의 성능

- 오버헤드가 5%이내

items	method	host	docker
CPU	sysbench	I	0.9931
memory	sysbench seq	I (r)	0.9999
		I (w)	0.9759
	sysbench rnd	I (r)	1.0056
		I (w)	0.9807
disk	dd	I	0.9716
network	iperf	I	**0.7889**

도커의 개념

도커의 특징

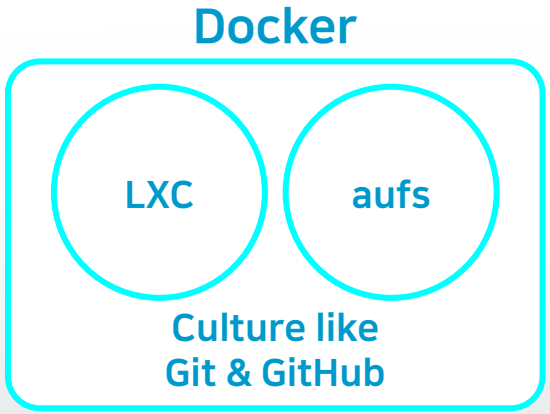
- 모든 컨테이너들이 동일 OS 커널 공유
  - 독립적인 스케줄링이나 CPU/메모리/디스크/네트워크를 가상화하지 않음

- 리눅스의 특수 기능(LXC)을 사용한 실행환경 격리를 응용
  - 리눅스에서만 사용 가능
    - 처음에는 우분투에서 현재 리눅스 배포판(fedora, RHEL, centos, …) 에서 사용 가능
  - 다른 OS(윈도우/OSX)에서는
    - 일반 하이퍼바이저(경량)가 있어야 함
  - 현재는 LXC -> Libcontainer를 사용해 리눅스 의존도를 줄이려 하고 있슴

도커의 개념

도커의 특징

- 시스템의 분리에는 Linux Containers (LXC)을,
- 파일 시스템은 Advanced multi layered unification filesystem (Aufs)를 사용
- 그리고 Git과 같은 이미지 버전컨트롤 시스템 도입



도커의 개념

도커의 특징

- 리눅스 컨테이너 가상화(LXC) 기반
- 2014/6 1.0 출시
- 구글과 같은 메이저 벤더들에 빠른 속도로 채택
- 현재 사실상의 표준(de facto standard)이 됨
- 하이퍼바이저와 달리 게스트OS란 계층이 없기 때문에 더 가볍고, 빠른 성능이 남
- 하이퍼바이저 기반의 우분투와 도커 기반의 우분투가 실제 설정/사용하는 방식이 상이
  - 환경변수 설정, 서비스 수행 방식, ….

도커의 개념

도커의 특징

- 구글에서 만든 Go라는 언어로 작성
- DotCloud
  - PaaS 공급 업체 DotCloud 가 PaaS의 벡엔드로 사용하는 컨테이너 기반의 가상화 소프트웨어를 오픈소스로 공개
  - 현재 도커(Docker Inc.)로 회사명 변경
- Docker 는 하나의 Linux 시스템에 여러 Linux 시스템 운영을 위한 소프트웨어

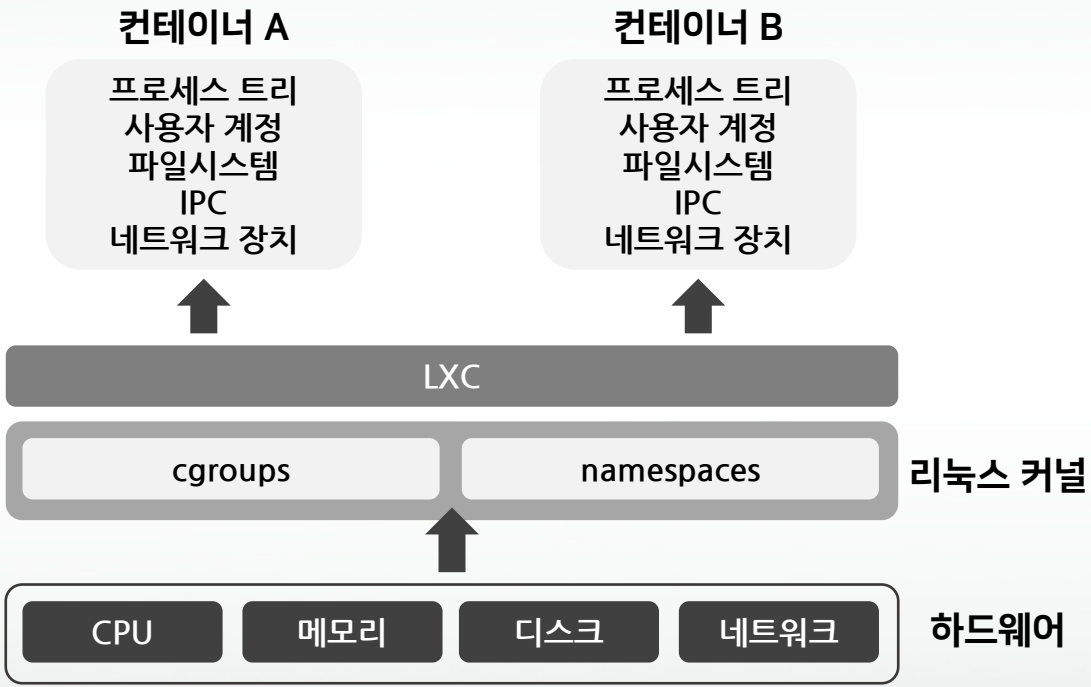
도커의 개념

LXC

- Linux Container
- 시스템레벨 가상화
- cgroups(control groups)
  - CPU, 메모리, 디스크, 네트워크
- Namespaces(Namespace Isolation)
  - 프로세스트리, 사용자계정, 파일시스템, IPC, …
  - 호스트와 별개의 공간설정
- chroot(change root) 명령어에서 발전
  - chroot jail
  - chroot 상의 폴더에서 외부 디렉토리 접근 안 됨

도커의 개념

LXC


컨테이너 A
프로세스 트리
사용자 계정
파일시스템
IPC
네트워크 장치
컨테이너 B
프로세스 트리
사용자 계정
파일시스템
IPC
네트워크 장치
LXC
cgroups
namespaces
리눅스 커널
CPU
메모리
디스크
네트워크
하드웨어

도커의 개념

Libcontainer

- 컨테이너 최적화 기술 LXC 외에
  리브컨테이너(libcontainer)란 별도의 실행 드라이버(exec driver)를 만들어, 특정 우분투 버전 외에 다양한 리눅스를 지원 가능

- 맥OS나 윈도서도 사용할 수 있는 가능성 생김

- native(libcontainer), lxc(LXC)

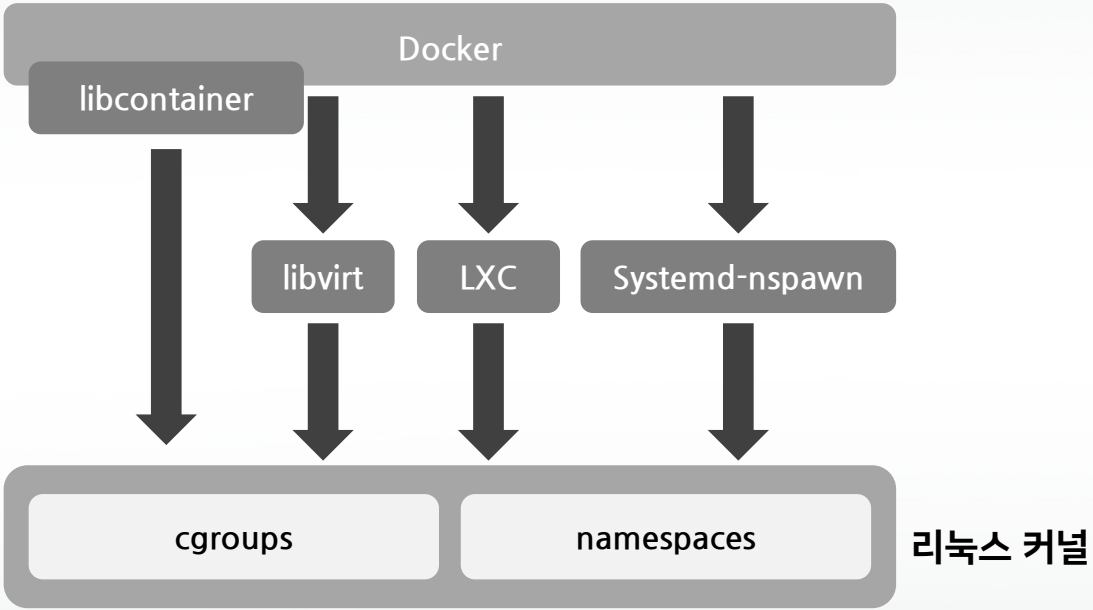
도커의 개념

도커의 구조

- 도커 =
  - LXC(cgroups+namespaces) or libcontainer +
  - AUFS(Advanced multi layered unification filesystem)
  - 이미지, 컨테이너 생성관리 +
  - 각종 부가기능

도커의 개념

도커의 구조


Docker
libcontainer
libvirt
LXC
Systemd-nspawn
cgroups
namespaces
리눅스 커널

도커의 개념

LXD

- 우분투를 만든 캐노니컬(canonical)에서 만든 컨테이너 솔루션
- 기존의 LXC에 보안 개념까지 추가
  - Secure by default
  - Unprivileged container
    - root가 아니어도 컨테이너 생성 가능
- 도커는 Application Container , LXD는 Machine Container
- LXD는 Container “Hypervisor”
- 경쟁기술이라기 보다는 보완관계(도커와 병행 가능)
  - KVM(Kernel Virtual Machine)을 경쟁기술로 간주

LESSON 2

설치

설치

도커 설치

- http://docker.com
- 요구사항
  - 윈도우 64비트 버전이상
  - 도커 툴박스(윈도우 8.1 이하)/도커 머신(윈도우 10 이상)
  - Boot2Docker vs. Docker Machine
- 다운로드 & 설치

설치

도커툴박스 vs. 도커머신

- Boot2docker(deprecated)
  - Tiny Core linux 기반의 경량 리눅스배포판 사용
  - 내부적으로 버추얼박스 지원
- Docker machine(new)
  - 가상호스트에 도커엔진 설치하는 툴
  - 버추얼박스, vmware 지원

LESSON 3

도커 사용법

도커 사용법

도커기반 우분투 설치

- 도커 이미지 검색(기본이 최신버전)
  - docker search ubuntu
- 우분투 이미지 다운로드
  - docker pull ubuntu
- 이미지 리스트 출력
  - docker images
- 컨테이너 생성
  - docker run --name=ubuntu ubuntu
- 컨테이너 접속
  - docker attach ubuntu
  - docker exec -it ubuntu bash
- 컨테이너 탈출
  - exit or 컨트롤-P-Q(컨테이너 정지하지 않고 나옴)

도커 사용법

도커 기본 명령어

- 도커 컨테이너 리스트
  - docker ps -a
- 도커 컨테이너 정지
  - docker stop ubuntu
- 도커 컨테이너 재시작
  - docker restart ubuntu
- 도커 컨테이너 삭제
  - docker rm ubuntu
  - docker rm -f ubuntu
  - docker kill ubuntu
- 도커 이미지 삭제
  - docker rmi ubuntu
  - docker images

도커 사용법

도커 명령어(***)

이미지 파일 생성

- docker save -o ubuntu_img.tar ubuntu

이미지 태그 지정

- docker tag 이미지ID ubuntu

이미지 압축/해제

- gzip ubuntu_img.tar / bzip2 ubuntu_img.tar
- gzip -d ubuntu_img.tar.gz / bzip2 -d ubuntu_img.tar.bz2

이미지 삭제

- docker rmi ubuntu

파일에서 이미지 로드

- docker load -i ubuntu_img.tar
- docker images
  - 이미지ID 확인

도커 사용법

도커 명령어

- search(검색) ubuntu
- pull(다운로드) ubuntu:latest
- run(이미지->컨테이너생성), exec(컨테이너 쉘명령어 수행)
- ps(프로세스 리스트)/attach(컨테이너 접속)
- stop/restart/kill/rm(컨테이너 삭제)/pause/unpause
- images(이미지리스트)/rmi(이미지삭제)
- commit(컨테이너->이미지생성)
- history(이미지 변경사항내역)
- diff(이미지와 컨테이너사이의 변경내역조회)
- inspect(컨테이너/이미지의 세부정보 조회)
- tag(이미지 새로운 태그지정)

도커 사용법

이미지 생성

도커 이미지 생성방법

- Dockerfile을 수행시켜 새로운 이미지 생성
  - 우분투 이미지에 Dockerfile 입력 후 수행
  - docker build --tag=hadoop-1.2.1 .
  - docker images
  - docker history hadoop-1.2.1

도커 사용법

Dockerfile - Hadoop 1.2

```
FROM ubuntu:latest
MAINTAINER Seokjae Ha <sjha72@gmail.com>

RUN apt-get update
RUN apt-get install nano
ENV TERM=xterm

RUN apt-get install -y openjdk-8-jdk

RUN apt-get install -y openssh-server
RUN mkdir /var/run/sshd

RUN echo 'root:kitri' | chpasswd

RUN sed -i 's/PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config
RUN sed -i 's/PermitRootLogin without-password/PermitRootLogin yes/' /etc/ssh/sshd_config

EXPOSE 22

CMD ["/usr/sbin/sshd", "-D"]

RUN wget https://archive.apache.org/dist/hadoop/common/hadoop-1.2.1/hadoop-1.2.1.tar.gz
RUN tar xvfz /hadoop-1.2.1.tar.gz
RUN rm /hadoop-1.2.1.tar.gz

ENV JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
ENV CLASSPATH=$JAVA_HOME/lib/*:.

ENV HADOOP_HOME=/hadoop-1.2.1
```

도커 사용법

Dockerfile - Hadoop 1.2

```
ENV HADOOP_HOME=/hadoop-1.2.1
ENV PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP_HOME/sbin

RUN cp $HADOOP_HOME/conf/hadoop-env.sh $HADOOP_HOME/conf/hadoop-env.sh.default
RUN sed -i 's/\# export JAVA_HOME=\/usr\/lib\/j2sdk1.5-sun/export JAVA_HOME=\/usr\/lib\/jvm\/java-8-openjdk-amd64/' /hadoop-1.2.1$

RUN cp $HADOOP_HOME/conf/core-site.xml $HADOOP_HOME/conf/core-site.xml.default
RUN cp $HADOOP_HOME/conf/hdfs-site.xml $HADOOP_HOME/conf/hdfs-site.xml.default
RUN cp $HADOOP_HOME/conf/mapred-site.xml $HADOOP_HOME/conf/mapred-site.xml.default

ADD core-site.xml   $HADOOP_HOME/conf/core-site.xml
ADD hdfs-site.xml   $HADOOP_HOME/conf/hdfs-site.xml
ADD mapred-site.xml $HADOOP_HOME/conf/mapred-site.xml

RUN mkdir /root/tmp

RUN ssh-keygen -t dsa -P "" -f ~/.ssh/id_dsa
RUN cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys

ENV USER ubuntu
RUN adduser --disabled-password --gecos "" $USER
RUN adduser $USER sudo
ADD authorized_keys /root/.ssh/authorized_keys
RUN chown $USER /root/.ssh/authorized_keys
RUN chown -R $USER:$USER /root/.ssh/authorized_keys
RUN chmod 700 /root/.ssh/authorized_keys
```

도커 사용법

core-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
   <name>fs.default.name</name>
   <value>hdfs://localhost:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/root/tmp</value>
  </property>
</configuration>
```

도커 사용법

hdfs-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
        <name>dfs.replication</name>
        <value>1</value>
</property>
</configuration>
```

도커 사용법

mapred-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
        <name>mapred.job.tracker</name>
        <value>localhost:9001</value>
</property>
</configuration>
```

도커 사용법

ssh_config

```
Host *
  UserKnownHostsFile /dev/null
  StrictHostKeyChecking no
  LogLevel quiet
  Port 22
```

도커 사용법

authorized_keys 복사

- ssh 자동로그인 설정

$ ssh-keygen -t dsa -P "" -f ~/.ssh/id_dsa

```
Generating public/private dsa key pair.
Created directory '/home/sjha/.ssh'.
Your identification has been saved in /home/sjha/.ssh/id_dsa.
Your public key has been saved in /home/sjha/.ssh/id_dsa.pub.
The key fingerprint is:
5f:36:aa:9c:9e:c8:0c:04:93:92:2e:53:7a:ae:cf:8d sjha@ubuntu-server
The key's randomart image is:
+--[ DSA 1024]----+
|          |
| . .        |
|o =         |
|.+ o        |
|+.. .  S  +  |
|.+ .   . + .  |
| . .   o   |
| o o + o +    |
|..E . +.*     |
+-----------------+
```

$ cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys

$ cp ~/.ssh/authrorized_keys ~

도커 사용법

도커 이미지 실행 및 예제 실행

컨테이너 실행 및 접속

```
$ docker run -d --name=hadoop hadoop-1.2.1
$ docker exec -it hadoop bash
```

네임노드 포맷 및 서버 실행

```
# hadoop namenode -format
# start-dfs.sh
# start-mapred.sh
# jps
```

하둡 예제 실행

```
# hadoop jar hadoop-examples-1.2.1.jar wordcount
    /input/README.txt /output
```