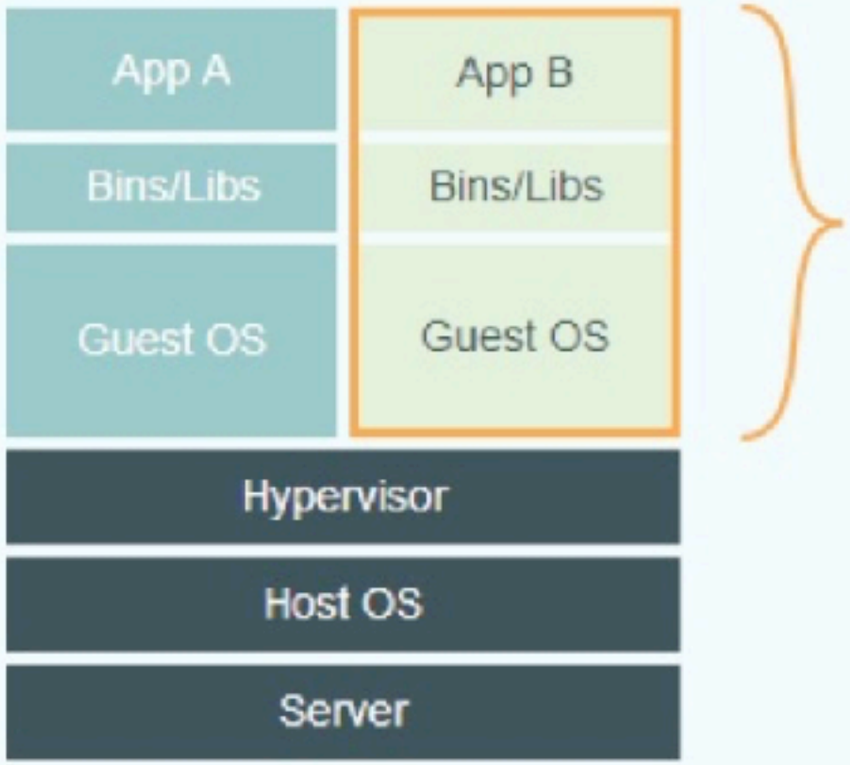
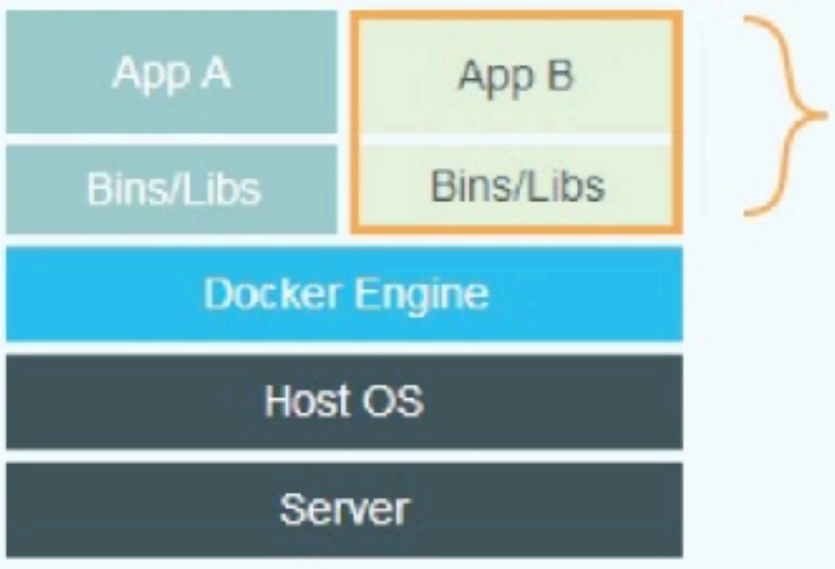
乐聚 Docker 使用分享

Docker 是什么?



Virtual Machines

Each virtualized application includes not only the application - which may be only 10s of MB - and the necessary binaries and libraries, but also an entire guest operating system - which may weigh 10s of GB.



Docker

The Docker Engine container comprises just the application and its dependencies. It runs as an isolated process in userspace on the host operating system, sharing the kernel with other containers. Thus, it enjoys the resource isolation and allocation benefits of VMs but is much more portable and efficient.

為什麼要使用 Docker?

- Docker 容器的啟動可以在秒級實作，這相比傳統的虛擬機方式要快得多。 其次，Docker 對系統資源的使用率很高，一台主機上可以同時執行數千個 Docker 容器。
- 更快速的交付和部署,一次建立或設定，可以在任意地方正常執行。

特性	容器	虛擬機
啟動	秒級	分鐘級
硬碟容量	一般為 MB	一般為 GB
效能	接近原生	比較慢
系統支援量	單機支援上千個容器	一般幾十個

基本概念

- Docker 映像檔,层的概念，类似 git 的 commit
- Docker 容器容器是從映像檔建立的執行實例。它可以被啟動、開始、停止、刪除。每個容器都是相互隔離的、保證安全的平台。可以把容器看做是一個簡易版的 Linux 環境（包括root使用者權限、程式空間、使用者空間和網路空間等）和在其中執行的應用程式
- 倉庫: 最大的公開倉庫是 Docker Hub，存放了數量龐大的映像檔供使用者下載。类似于 ubuntu 的 apt，apple 的 app store

docker images

```
$ sudo docker images
REPOSITORY          TAG       IMAGE ID      CREATED      VIRTUAL SIZE
ubuntu              12.04     74fe38d11401  4 weeks ago  209.6 MB
ubuntu              precise   74fe38d11401  4 weeks ago  209.6 MB
ubuntu              14.04     99ec81b80c55  4 weeks ago  266 MB
ubuntu              latest    99ec81b80c55  4 weeks ago  266 MB
ubuntu              trusty    99ec81b80c55  4 weeks ago  266 MB
```

容器基本操作

- Sudo docker run -t -I ubuntu:14.04 /bin/bash 其中, -t 選項讓Docker分配一個虛擬終端（pseudo-tty）並綁定到容器的標準輸入上, -i 則讓容器的標準輸入保持打開。
- Sudo docker ps －all 查看启动和停止的容器
- Docker start 用来启动关闭的容器
- 作为 deamon 运行 -d
- 进入一个已经在运行的容器 sudo docker exec -ti container_id /bin/bash

个性化镜像

- 传播，分享镜像
- 保存当前的镜像状态 sudo docker export container_id > ubuntu.tar
- 倒入找到的镜像及分享的镜像文件 cat ubuntu.tar | sudo docker import test/ubuntu:v1.0
- 删除无用的镜像 sudo docker rmi image

数据与程序分离

- 通过启动镜像的命令切换加载的数据源 sudo docker run -d -P —name web -v /src/webapp:/opt/webapp image name /bin/bash

- 通过操作数据目录来备份及恢复数据

容器互联 docker-compose

- 什么是 docker-compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a Compose file to configure your application’s services. Then, using a single command, you create and start all the services from your configuration. To learn more about all the features of Compose see the list of features.

```yaml
services:
  redis:
    restart: always
    image: sameersbn/redis:latest
    command:
    - --loglevel warning
    volumes:
    - ./docker/gitlab/redis:/var/lib/redis:Z

  postgresql:
    restart: always
    image: sameersbn/postgresql:9.5-1
    volumes:
    - ./docker/gitlab/postgresql:/var/lib/postgresql:Z
    environment:
    - DB_USER=gitlab
    - DB_PASS=password
    - DB_NAME=gitlabhq_production
    - DB_EXTENSION=pg_trgm

  gitlab:
    restart: always
    image: sameersbn/gitlab:9.2.5
    depends_on:
    - redis
    - postgresql
    ports:
    - "10088:80"
    - "10026:22"
    volumes:
    - ./docker/gitlab/gitlab:/home/git/data:Z
    environment:
    - DEBUG=false
```

Two suffixes :z or :Z can be added to the volume mount. These suffixes tell Docker to relabel file objects on the shared volumes. The 'z' option tells Docker that the volume content will be shared between containers. Docker will label the content with a shared content label. Shared volumes labels allow all containers to read/write content. The 'Z' option tells Docker to label the content with a private unshared label. Private volumes can only be used by the current container.

实践

- gitolite https://hub.docker.com/r/szyhf/gitolite-alpine/

- docker run -e SSH_KEY="$(cat ~/.ssh/id_rsa.pub)" elsdoerfer/gitolite

附錄二：常見倉庫介紹

- Ubuntu
- CentOS
- MySQL
- MongoDB
- Redis
- Nginx
- WordPress
- Node.js