Docker

A container is a running instance of an image

Install Docker

install virtualBox and Ubuntu

refer to: <https://docs.docker.com/linux/step_one/>

ctrl + alt + t打开终端

$ curl -fsSL https://get.docker.com/ | sh

reboot Ubuntu

$docker run helloworld

若无法connection

$sudo gedit /etc/default/docker

export http\_proxy=”<http://HOST:PORT/>”

或者命令行

export http\_proxy=http://165.225.96.34:10015/

export https\_proxy=https://165.225.96.34:10015/

$sudo service docker restart //restart the docker service

docker: 告诉操作系统使用docker程序

run：docker的子命令创建与运行Docker容器

hello-world: 告诉Docker将hello-world 镜像装入到容器里

Build your own image

1. Write a Dockerfile

FROM docker/whalesay:latest

RUN apt-get -y update && apt-get install -y fortunes

CMD /usr/games/fortune -a | cowsay

1. Build an image from your Dockerfile

$docker build – t docker-whale .

（Note: 若无法connection，改为

$docker build - -build-arg HTTP\_PROXY= <http://165.225.96.34:10015/> -t docker-whale .

1. Run your image

$docker run docker-whale

Tag & push your image to DockerHub

$ docker tag imageID qizhonglin/docker-whale:latest

$ docker login - - username=qizhonglin - - [email=qizhong.lin@philips.com](mailto:email=qizhong.lin@philips.com)

$ docker push qizhonglin/docker-whale

<第一本Docker书>

$docker info

该命令返回所有容器和镜像的数量，Docker使用的执行驱动和存储驱动以及Docker的基本配置

$docker run –it ubuntu /bin/bash

该命令提供Docker容器的创建到启动的功能。

- i 标志保证容器中STDIN是开启的

- t 标志告诉Docker为要创建的容器分配一个伪tty终端

ubuntu镜像告诉Docker基于ubuntu镜像来创建容器，该容器拥有自己的网络、IP地址、以及一个用来和宿主机进行通信的桥接网络接口

/bin/bash告诉Docker在新容器中运行的命令，在这里指启动一个bash shell

[root@de28...:/#](mailto:root@de28...:/#) apt-get updata && apt-get install vim

若无法连接网络，需要设置

export http\_proxy=http://165.225.96.34:10015/

export https\_proxy=https://165.225.96.34:10015/

$docker ps –a

查看当前系统中容器的列表

$docker run - - name bob\_the\_container - i - t ubuntu /bin/bash

--name: 指定容器名称，可以用容器的名称来替代容器ID,容器的命名必须是唯一的。

$docker start bob\_the\_container

启动容器

$docker attach bob\_the\_container

重新附着到该容器的会话上

$docker rmi - f imageID

or $docker rmi - f bob\_the\_container

除了交互式运行的容器(interactive container),我们也可以创建长期运行的容器（守护式容器daemonized container）没有交互式会话，非常适合运行应用程序和服务

$docker run - - name daemon\_dave -d ubuntu /bin/sh –c “while true; do echo hello world; sleep 1; done”

-d标识告诉Docker将容器入到后台运行

$docker ps

查看正在运行的容器

$docker logs –ft daemon\_dave

获取容器的日志

-f监控Docker的日志，这与tail –f 命令非常相信

$docker top daemon\_dave

查看容器内部运行的进程

$docker exec –d daemon\_dave touch /etc/new\_config\_file

-d: 标志表明运行一个后台进程

daemon\_dave: 指定的容器

touch /etc/new\_config\_file: 执行的命令

$docker exec -it daemon\_dave /bin/bash

$docker run - -restart=always –name daemon\_dave –d Ubuntu /bin/sh –c “while true;…”

restart=always标志表明无论容器的退出代码是什么，Docker都会自动重启该容器

restart=on-failure标志表明只有当容器的退出代码为非0值的时候，才会自动重启

$docker inspect daemon\_dave

该命令对容器进行详细的检查，然后返回其配置信息，包括名称、命令、网络配置以及很多有用的数据

$docker rm daemon\_dave

删除容器

$docker rm `docker ps –a –q`

删除所有容器

docker ps命令会列出现有的全部容器

-a标志代表列出所有容器包括运行和停止

-q标志则表示只需要返回容器的ID而不会返回容器的其他信息