1. Docker不是一种编程语言，并且也不是构建软件的框架。Docker是一个开源的Linux软件，用Golang编写，采用C/S架构，可以很好地运行在大多数操作系统中（通过虚拟机在Windows和Mac上使用Docker）。Docker包括一个命令行程序、一个后台守护进程，以及一组远程服务。
2. 硬件虚拟化（虚拟机）：创建时间长、资源开销大；Docker:不是一个虚拟化技术，也不提供容器技术。相反，它可以帮助使用已经内置到操作系统中的容器技术。运行Docker可以认为是在用户空间运行着两个程序---Dockers守护进程和Docker命令行接口(CLI)，每个Docker容器都是以Docker守护程序的紫禁城进行，并授权其在自身用户空间的存储子空间中运行。Linux的命名空间和cgroups管理着运行时的容器，而Docker采用另一套技术来为文件提供容器，
3. 镜像指的是以分层的、可以被LXC/libcontainer（内核虚拟化技术）理解的文件存储格式。把应用镜像从Docker仓库系在到本地机器上，以镜像为模板，在一个容器类虚拟机把这个应用启动，这个虚拟机叫做容器。，
4. 安装：windows下：到官网<https://www.docker.com/toolbox>下载Docker ToolBox，双击安装成功后会在桌面出现两个快捷图标，其中Kitematic是Docker图形化管理方式，Docker Quickstart是命令行管理方式。若报错则可能系统的虚拟化被禁止了，可以在任务管理器的性能一项中查看，并在开机BIOS中激活虚拟化。安装完成后可执行:

docker run hello-world 确认是否工作正常。

Ubuntu下：方法一：sudo apt-get install docker.io （版本较旧不推荐）

方法二: sudo apt-get install curl curl -sSL https://get.docker.com / | sh

安装完成后，通过 sudo service docker start 命令启动Docker的守护进程。因为docker默认是以root用户执行，因此可执行 sudo usermod -aG docker username 避免每次都要加上sudo。

1. 搭建个人博客网站：

（1）docker run –name db –env MYSQL\_ROOT\_PASSWORD=example -d mariadb

-d 在后台启动一个mariadb数据库（MySQL的一个分支）,数据库管理员root的密码设置为example； -name 起名为db 进行标识；

（2）docker run –name MyWordPress –link db:mysql -p 8080:80 -d wordpress

WordPress是把博客和个性化信息存储到数据库，所以需要通过–link跟数据库建立连接

WordPress是通过监听Apache（Web服务器）的80端口对外提供服务，但每台机器的80端口只有一个，如果被其他应用占用了，则可以通过-p 把80端口映射到任意端口。

完成后若是在windows下，则执行docker-machine.exe ip 来查看ip，然后在浏览器中输入<http://ip:8080>来访问WordPress；若是在ubuntu下则通过ifconfig来获取ip。

（3）安装GitLab：其优势是可以实现本地部署，搭建公司内部的版本控制系统，且提供几个免费的私有仓库。

我们使用sameersbn/docker-gitlab来搭建GitLab服务。它的运行环境由如下三部分组成：

启动postgresql:

docker run –name gitlab-psotgresql -d –env ‘DB\_NAME=gitlabhq\_production’ \

--env ‘DB\_USER=gitlab’ --env ‘DB\_PASS=password’ sameersbn/postgresql:9.4-12

启动redis:

docker run –name gitlab-redis -d sameersbn/redis:latest

启动gitlab:

docker run –name gitlab -d –link gitlab-postgresql:postgresql \

--link gitlab-redis:redisio –publish 10022:22 –publish 10080:80 \

--env ‘GITLAB\_PORT=10080’ –env ‘GITLAB\_SSH\_PORT=10022’ \

--env ‘GITLAB\_SECRETS\_DB\_KEY\_BASE=long-and-random-alpha-numeric-string’ \

Sanmeersbn/gitlab:8.4.4

测试GitLab: 通过ip:10080即可访问。默认用户名为root,密码为5iveL!fe

项目管理系统：Redmine

docker run –name=postgresql-redmine -d –env=’DB\_NAME=redmine\_production’ \

--env=’DB\_USER=redmine’ –env=’DB\_PASS=password’ sameersbn/postgresql:9.4-12

docker run –name=redmine -d –link=postgresql-redmine:postgresql –publish=10083:80

--env=’REDMINE\_PORT=10083’ sameersbn/redmine:3.2.0-4

测试：通过ip:10083即可访问，默认用户名密码均为admin

Docker基本概念和常用操作指令：三大基础组件：仓库、镜像、容器

1. Docker ps | awk ‘{print $2, $NF}’ 只列出IMAGE和NAMES两列
2. Docker指令的操作对象主要针对四个方面：

针对守护进程的系统资源的设置和全局信息的获取，如：docker info、docker deamon；

针对Docker仓库的查询\下载操作，比如：docker search、docker pull；

针对Docker镜像的查询、创建、删除操作,如：docker images、docker build；

针对Docker容器的查询、创建、开启、停止操作，如：docker ps、docker run。

Docker指令除了单挑使用外，还支持赋值、解析变量、嵌套等使用，如：

例1：获取容器的ID，并根据ID提交到仓库。

ID=$(docker run -d ubuntu echo hello world)

docker commit $ID helloworld

例2：删除所有停止运行的容器

Docker rm $(docker ps -a -q)

简单使用教程:

1. docker version
2. 查询镜像：docker search tutorial
3. 下载镜像：docker pull learn/tutorial
4. 创建并启动容器：docker run learn/tutorial echo “hello world”
5. 修改容器：docker run learn/tutorial apt-get install -y ping (注意：此时又新建了第二个容器)
6. 创建新镜像：在上一步已经安装了ping程序包，你可能想要保存这个变更，以便于以后启动荣其实不需要重复安装该程序包。Docker支持在原有镜像基础上，只提交增量修改部分新城一个新镜像。首先用 docker ps -l 找到安装过ping包的容器的ID号，再用docker commit ID learn/ping 把容器提交为新镜像learn/ping。
7. 使用新镜像：docker run learn/ping ping [www.docker.com](http://www.docker.com) 使用CTRL+C终止ping 命令
8. 查询容器信息：docker ps -a 可查询所有本机的容器（包括未运行的）

根据提示，我们可以使用容器的前3~4个字符来指定容器，执行docker inspect (容器ID前几个字符) 即可查看单个容器详细信息。

9）把新镜像上传仓库：先执行docker image 可查看本机的镜像列表，再执行

docker push learn/ping 即可镜像推送到Docker官方仓库

在heeps://hub.docker.com/account/signup/上注册一个Docker Hub账号，使用docker login登录后才可以做上传镜像的操作。

1. docker run 只在第一次运行时使用，将镜像放到容器中，以后再次启动这个容器时，只需要使用命令docker start 即可。

2. 然后使用下边的命令进入shell：

sudo docker exec -it “容器ID” bash

3. 将主机的文件复制到容器里：

sudo docker cp 主机目录 容器ID:容器目录

4. 停止所有的container，这样才能够删除其中的images：

docker stop $(docker ps -a -q)

如果想要删除所有container的话再加一个指令：

docker rm $(docker ps -a -q)

5.查看当前有些什么images

docker images

6.删除images，通过image的id来指定删除谁

docker rmi <image id>

想要删除untagged images，也就是那些id为<None>的image的话可以用

docker rmi $(docker images | grep "^<none>" | awk "{print $3}")

要删除全部image的话

docker rmi $(docker images -q)

7. docker创建自己的image

http://blog.csdn.net/wenwenxiong/article/details/46347205

8.centos安装maven

wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo

yum -y install apache-maven

9.docker caffe libdc1394 error: Failed to initialize libdc1394

解决方案：

ln /dev/null /dev/raw1394

docker ps和docker images查看容器和镜像，都可以加上-a选项

docker inspect 命令可查询容器的所有基本信息，通过-f 可以使用Golang的模板来提取指定部分的信息，即 docker inspect -f {{.State.Status}} MywordPress

docker logs MywordPress 查询日志，加上-f 可以试试打印最新的日志。

docker stats MywordPres 查询容器占用的系统资源

经常有登入Docker容器内部执行命令的需求，可以在容器中启动sshd服务来响应用户登录，但sshd方式存在进程开销和增加被攻击的风险。同时也违反Docker所倡导的“一个容器一个进程”的原则。因此可用 docker exec 容器名 容器内执行的命令

如果希望在容器内连续执行多条命令，可以上加上-it 参数，就相当于以root身份登入容器内，可以连续执行命令，执行完成后通过 exit 退出。

Docker Compose:允许用户在一个模板(YAML格式)中定义一组相关联的应用容器，这组容器会根据配置模板中的’--link’等参数，对启动优先级自动排序，简单执行一条 docker-compose up ,就可以把一个服务中的多个容器依次创建和启动。

Docker Compose安装方式如下：

sudo curl -L [https://github.com/docker/compose/release/download/1.6.0/docker-compose-`uname](https://github.com/docker/compose/release/download/1.6.0/docker-compose-%60uname) -s`-`uname -m` > /usr/local/bin/docker-compose

sudo chmod +x /sur/local/bin/docker-compose

我们看看如何使用Docker Compose 来管理WordPress项目

首先docker stop db MywordPress

接着创建一个项目文件夹~/wordpress,在文件夹下创建一个名为docker-compose.yml(默认名)的文件，后面运行时也可以用-f 选项指定配置文件，其内容如下：

wordpress:

image: wordpress

links:

-db:mysql

ports:

-8080:80

db:

image: mariadb

environment:

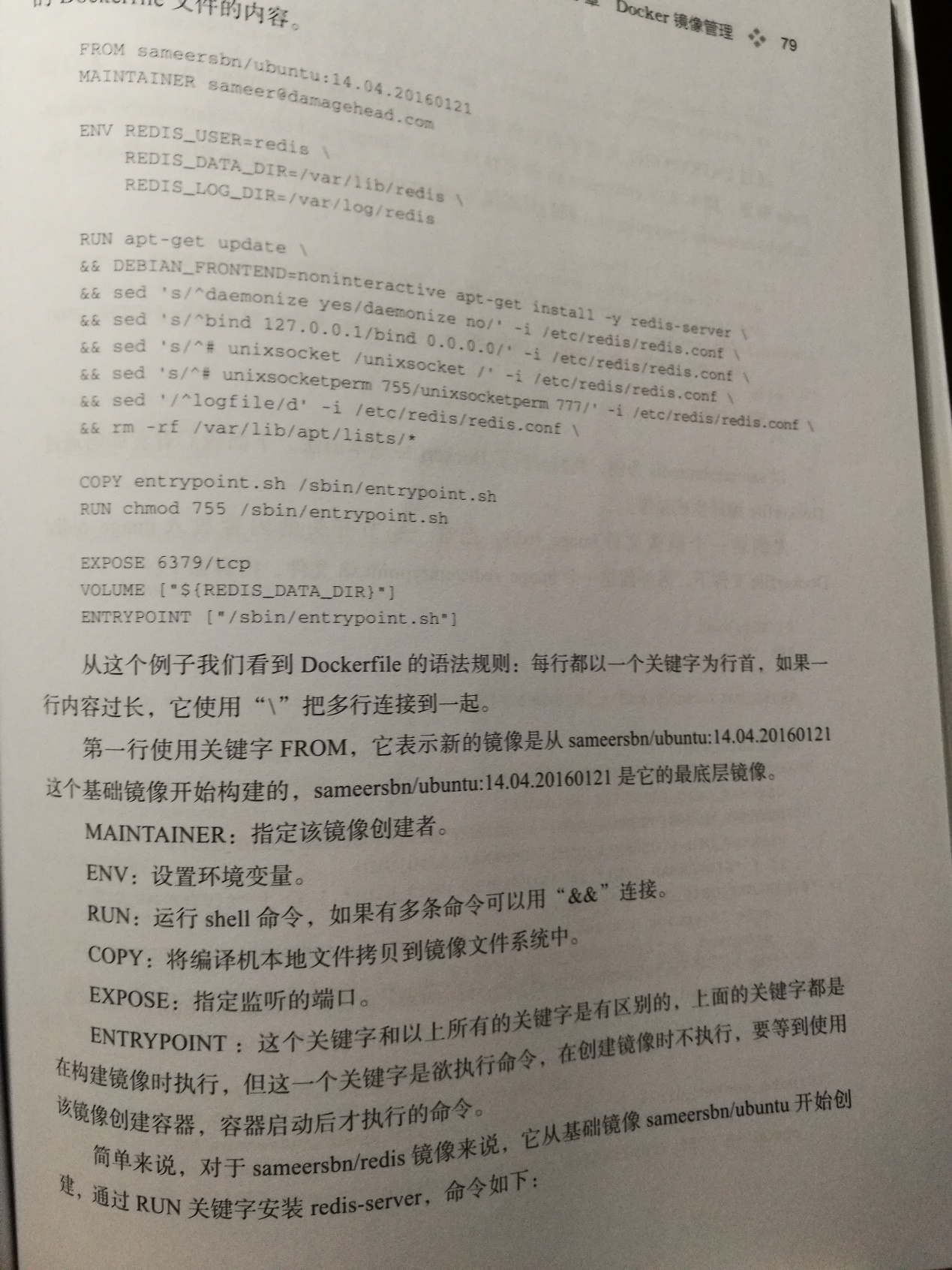
MYSQL\_ROOT\_PASSWORD: example

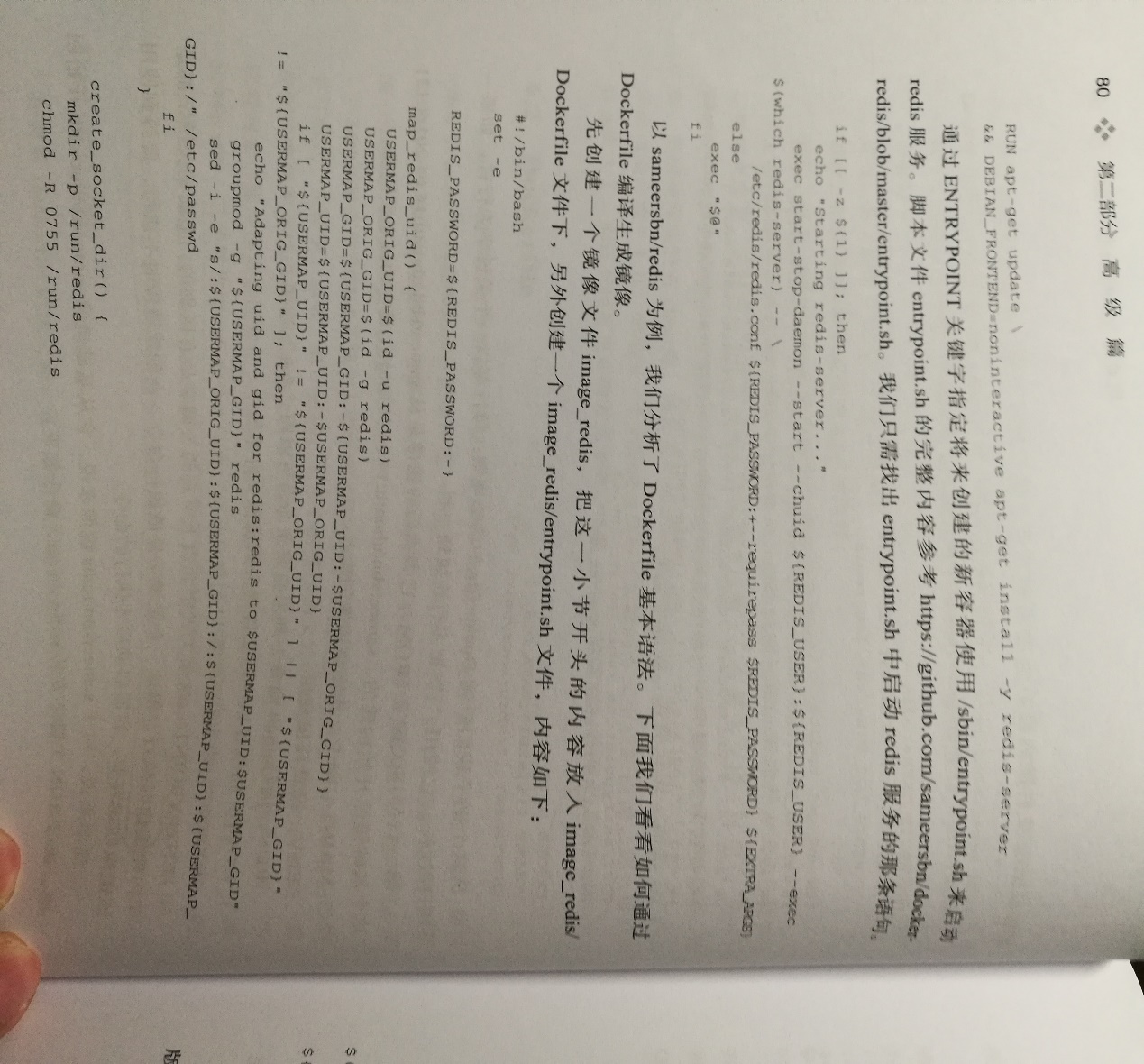
然后就可以执行 cd ~/wordpress && docker-compose up 来创建和启动WordPress服务。

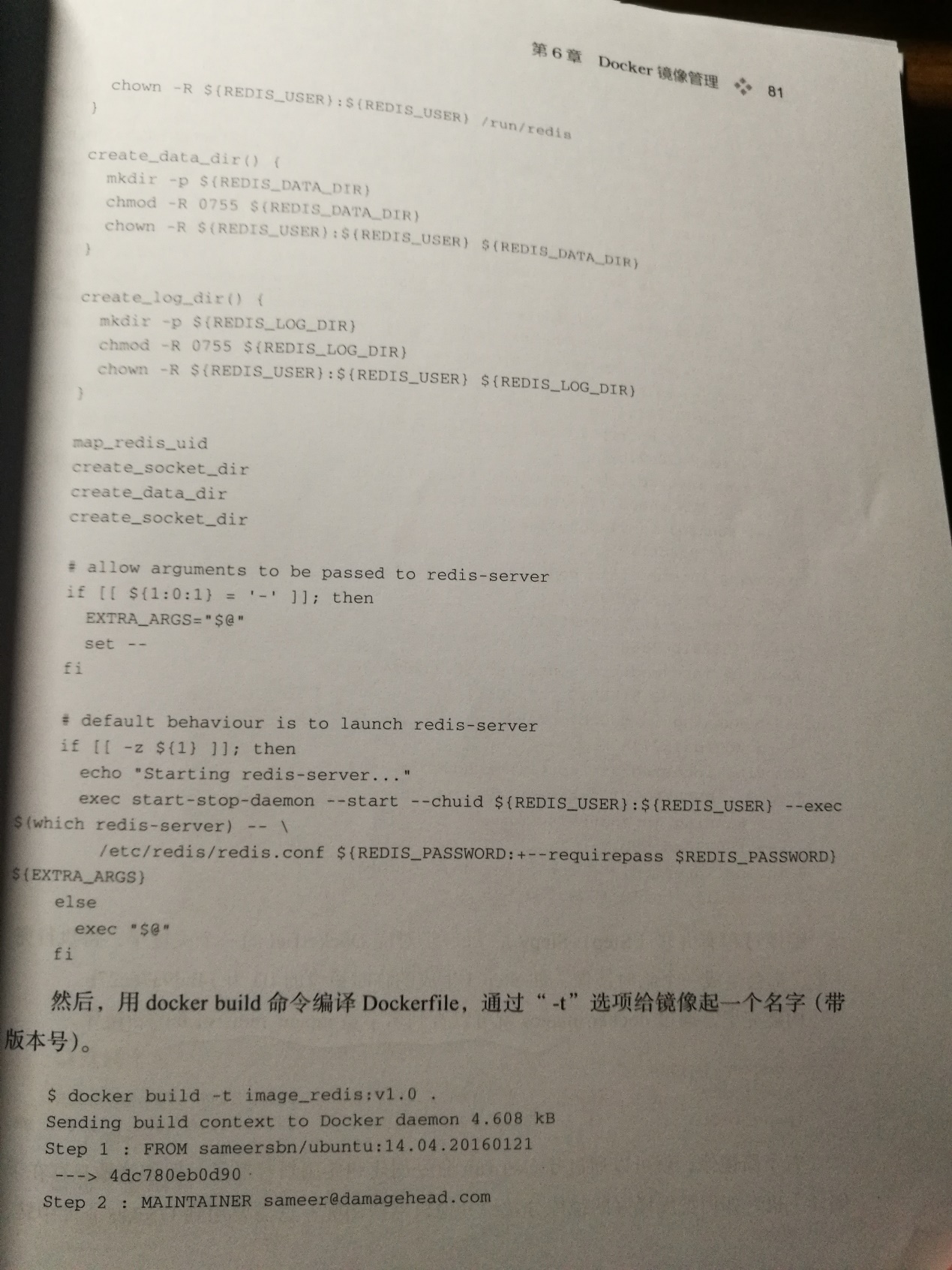
另外通过docker-compose start 和stop来运行和停止。

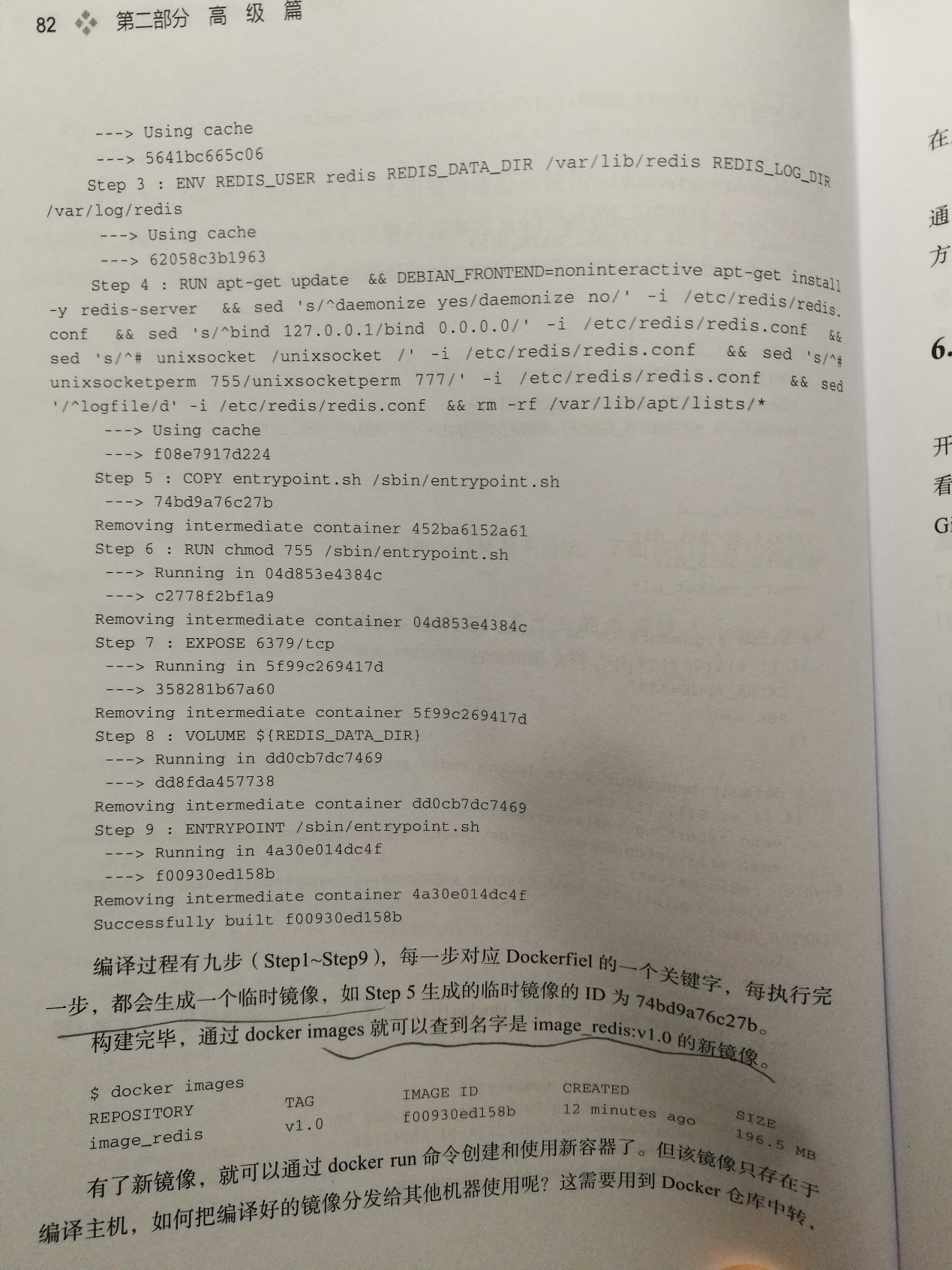
docker history [--tree]可查询镜像分了多少层，每一层具体做了什么操作。已有的分层只能读不能修改；上层镜像的优先级高于底层镜像。

DockerFile定制镜像







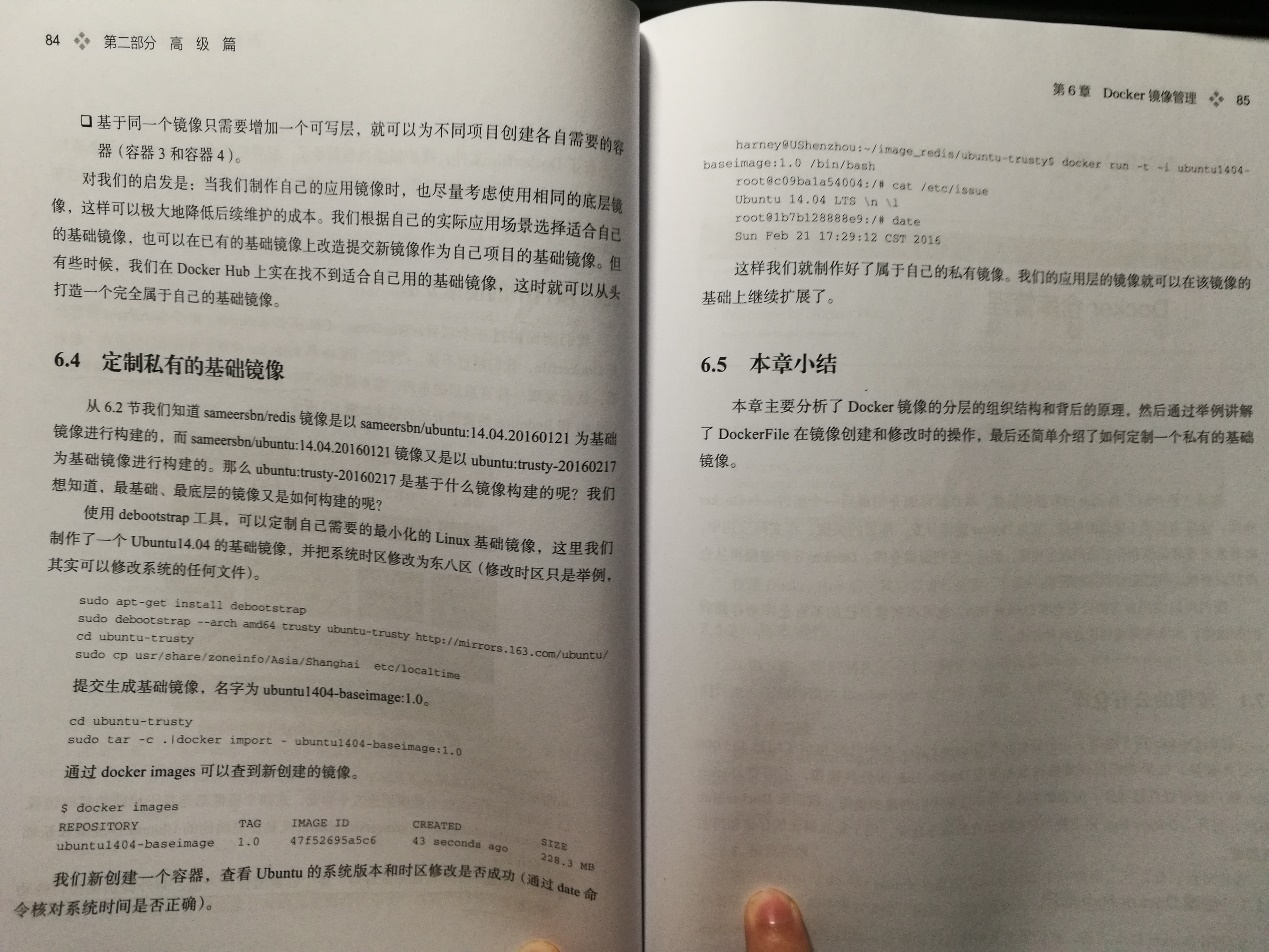


**Docker仓库管理**

1. 在hub.docker.com上完成注册
2. 在命令行通过docker login登录Hub
3. docker push 镜像名:tag 上传镜像到Hub
4. docker search 搜索镜像
5. docker pull 拉取镜像

**利用docker-registry组件构建自己的私有镜像仓库**

1. docker run -p 5000:5000 registry 这样Docker会自动拉取并运行registry镜像
2. 或者通过yum install docker-registry-y 以及 service docker-registry start来安装启动registry，默认其会监听5000端口，启动八个工作进程( netstat -ltnp | grep 5000 来查)
3. Registry没有提供安全认证，为此需用Nginx构建一个带认证功能的私有仓库，再配置SSL和CA



Docker的容器集群管理平台Kubernetes(k8s)、服务发现的键值存储系统Etcd，以及容器监控平台Cadvisor,其中后两者也是K8s的核心组件。

1. 安装Etcd: mkdir -p /home/install && cd /home/install

wget <https://github.com/coreos/etcd/releases/download/v0.4.6/etcd-v0.4.6-linux-amd64.tar.gz>

tar -zxvf etcd-\*.tar.gz

cd etcd-\*

cp etcd\* /bin/

/bin/etcd -version

启动Etcd服务，如有提供第三方管理需求，需在启动参数时添加”-cors=’\*’”参数，比较好的管理工具有etcd-browser

mkdir /data/etcd #etcd数据目录

.bin/etcd -name etcdserver -peer-addr 192.168.1.10:7001 -addr 192.168.1.10:4001 -data-dir /data/etcd -peer-bind-addr 0.0.0.0:7001 -bind-addr 0.0.0.0:4001 &

配置etcd服务防火墙

iptables -I INPUT -s 192.168.1.0/24 -p tcp –dport 4001 -j ACCEPT #4001为服务端口

iptables -I INPUT -s 192.168.1.0/24 -p tcp –dport 7001 -j ACCEPT #7001集群数据交互

Cadvisor是google用来分析运行中的Docker容器的资源占用及性能特性的工具。

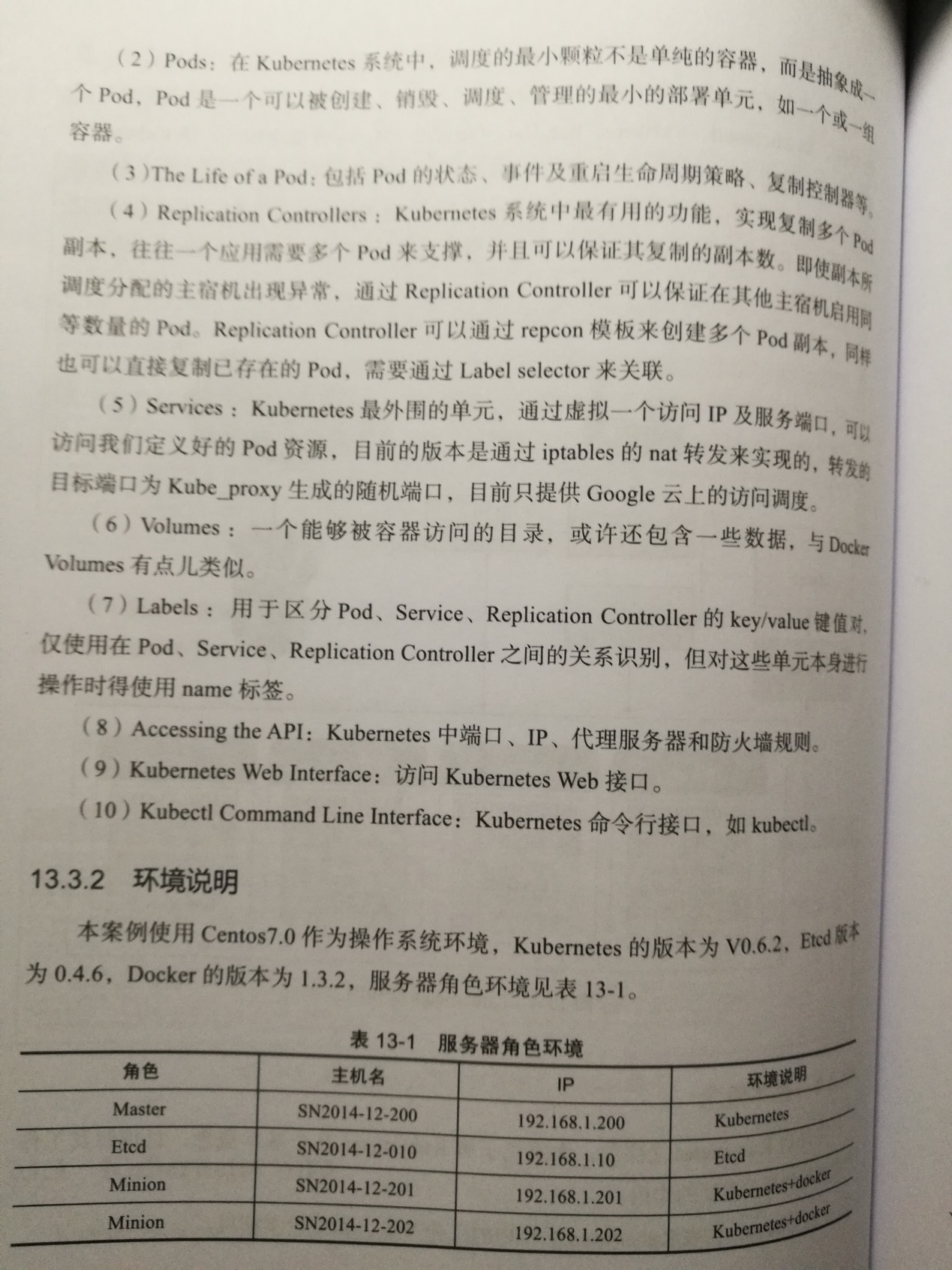
docker run –volume=/var/run:/var/run:rw –volume=/sys/fs/cgroup/:/sys/fs/cgroup:ro \

--volume=/var/lib/docker/:/var/lib/docker:ro –publish=8080:8080 –detach=true \ google/cadvisor

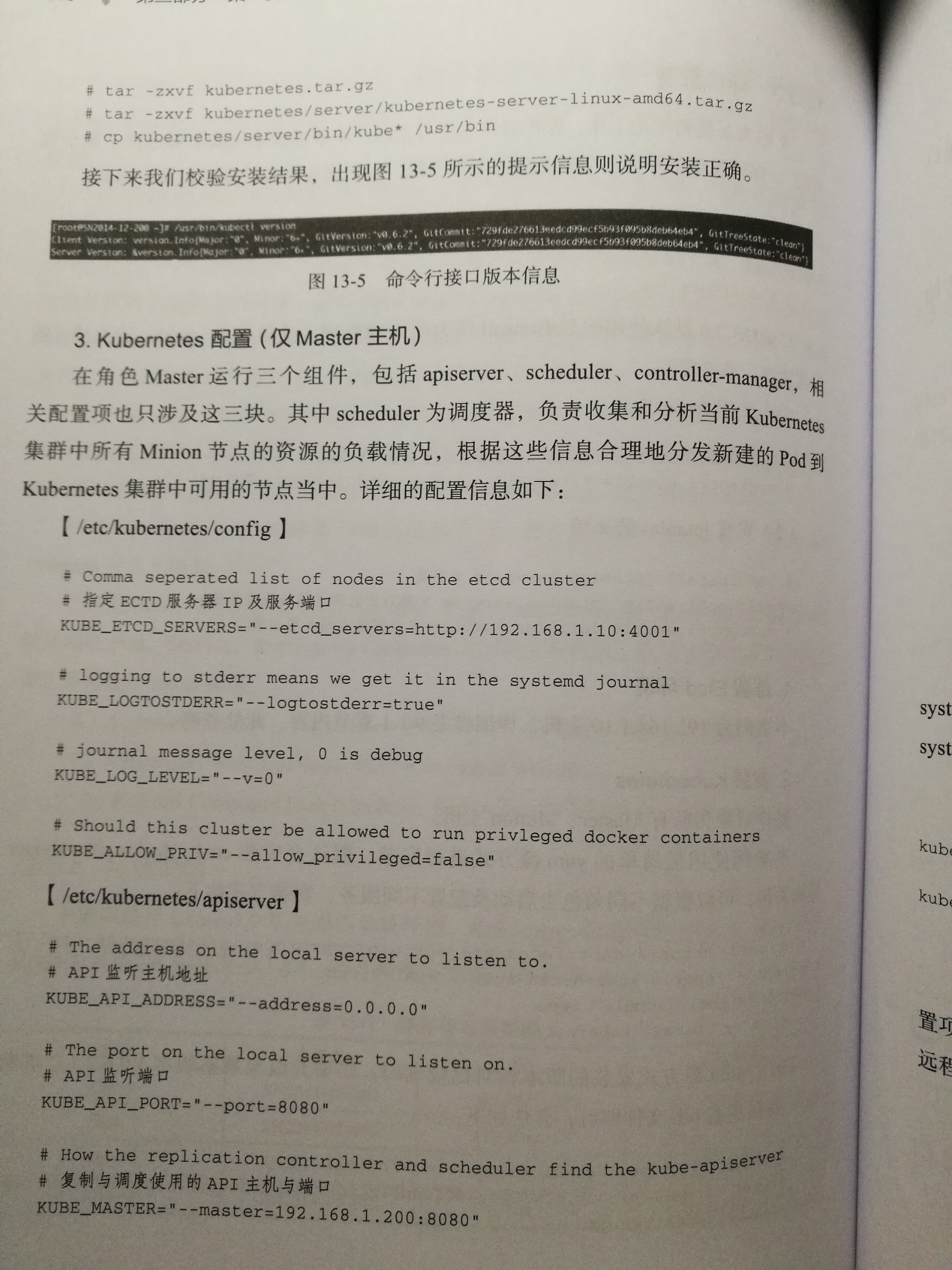
访问<http://宿主机IP:8080>出现界面说明成功

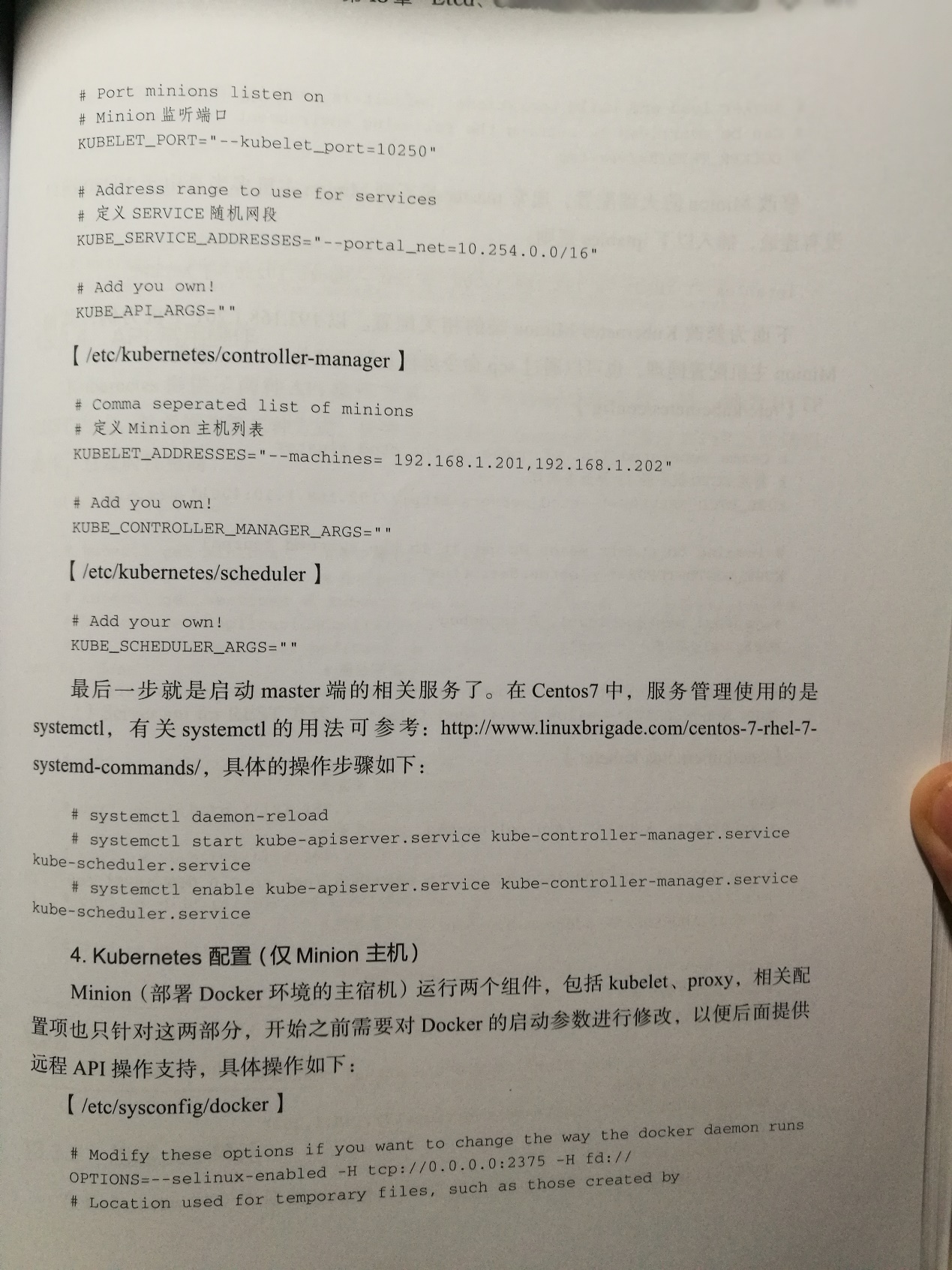
K8s基本概念

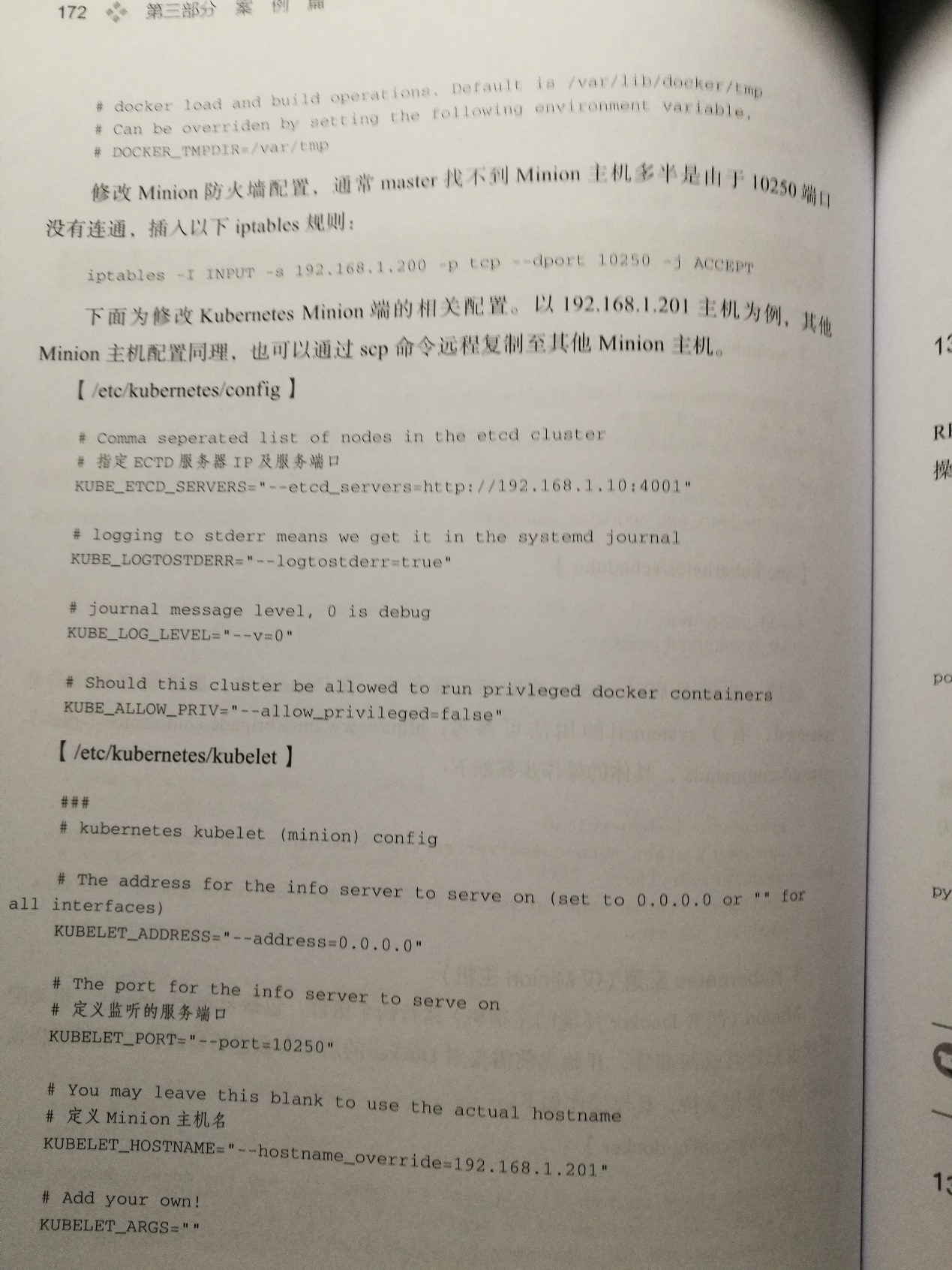
1. Nodes:代表k8s中的工作节点，如一台宿主机

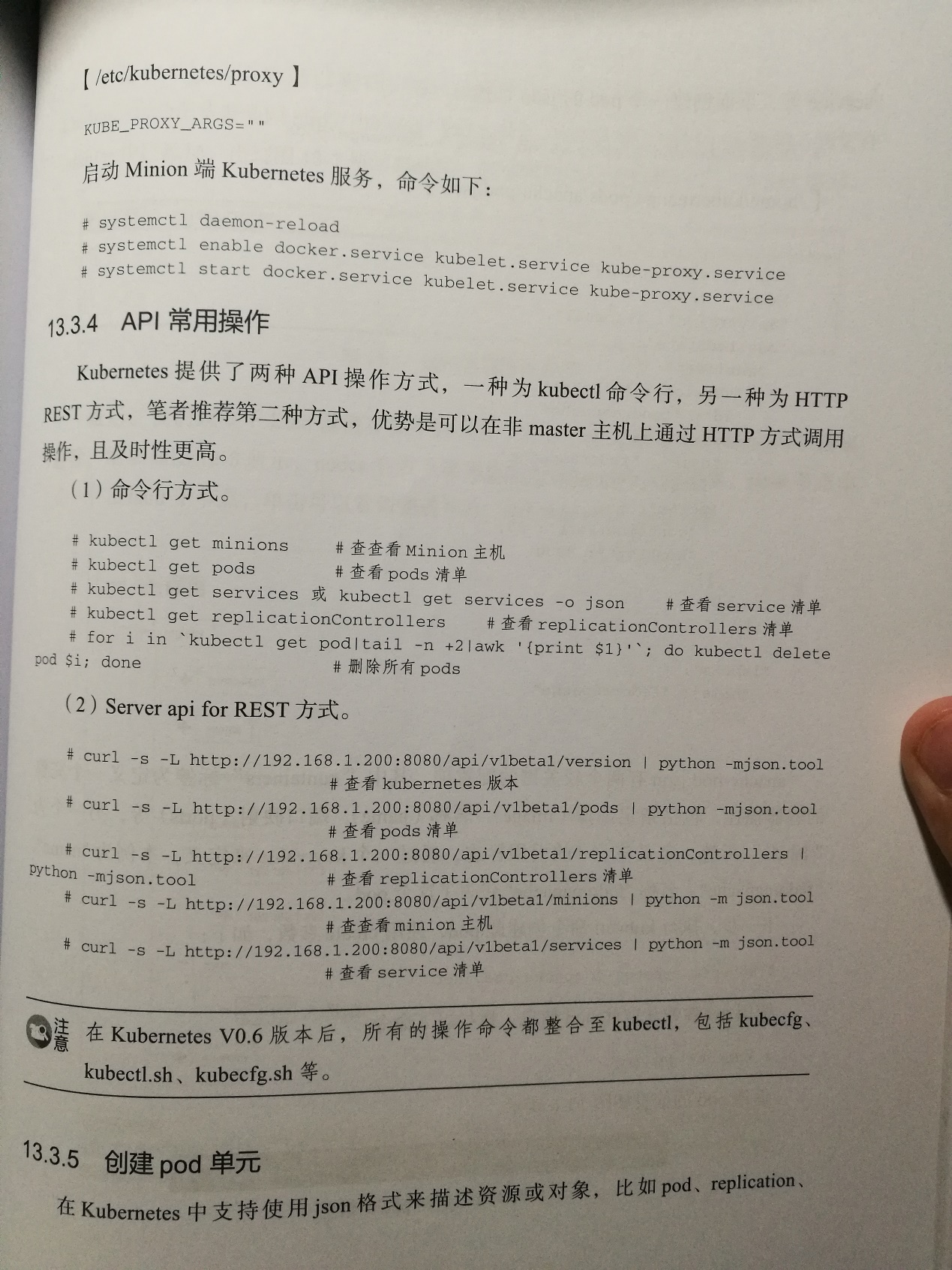












1.docker配置文件位置：/lib/systemd/system/nvidia-docker.service

2.Docker daemon日志的位置，根据系统不同各不相同。

\* Ubuntu - /var/log/upstart/docker.log

\* Boot2Docker - /var/log/docker.log

\* Debian GNU/Linux - /var/log/daemon.log

\* CentOS - /var/log/daemon.log | grep docker

\* Fedora - journalctl -u docker.service

\* Red Hat Enterprise Linux Server - /var/log/messages | grep docker