Linux 云计算集群架构师课程

北京学神科技有限公司

学神 IT 教育：从零基础到实战，从入门到精通！

联系方式：

学神 IT 教育官方网站: http://www.xuegod.cn

学神 IT 教育-PHP 技术交流 QQ 群: 196954821

学神 IT 教育-Linux 技术交流 QQ 群: 57873866

==========

咨询 QQ: 1514460659

学习一个服务的过程：

1、 此服务器的概述：名字，功能，特点，端口号
2、 安装
3、 配置文件的位置
4、 服务启动关闭脚本，查看端口
5、 此服务的使用方法
6、 修改配置文件，实战举例
7、 排错（从下到上，从内到外）

本节所讲内容：

- Docker 概述
- 部署 docker 容器虚拟化平台
- docker 镜像制作方法
- Container 容器端口映射

Dcoker 概述

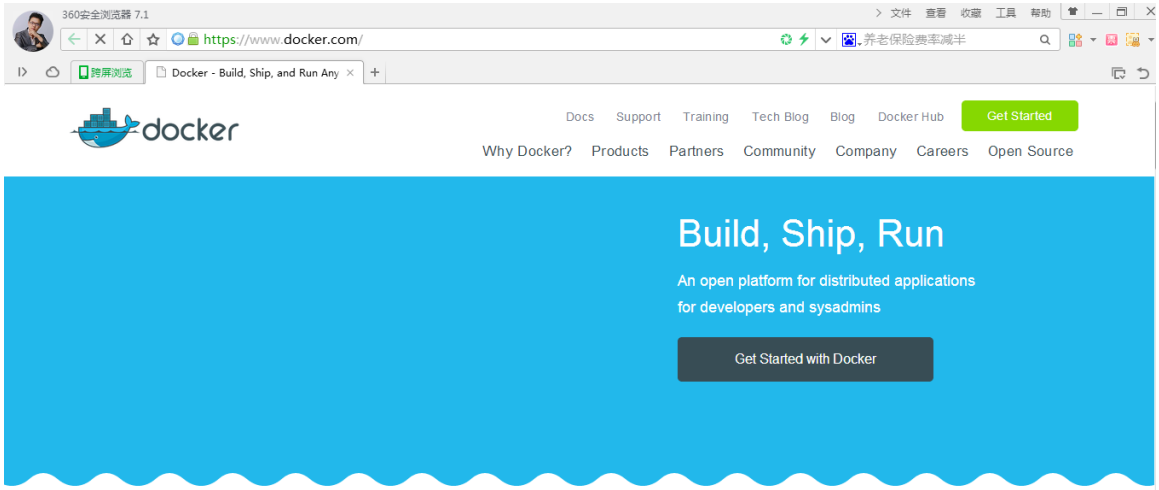
Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的容器中，然后发布到任何流行的 Linux 机器上，也可以实现虚拟化。容器是完全使用沙箱机制，相互之间不会有任何接口（类似 iPhone 的 app）。几乎没有性能开销,可以很容易地在机器和数据中心中运行。最重要的是,他们不依赖于任何语言、框架或包装系统。

Docker 是 dotCloud 公司开源的一个基于 LXC 的高级容器引擎，源代码托管在 Github 上，基于 go 语言并遵从 Apache2.0 协议开源。
Docker 让开发者可以打包他们的应用以及依赖包到一个可移植的 container 中，然后发布到任何流行的 Linux 机器上。

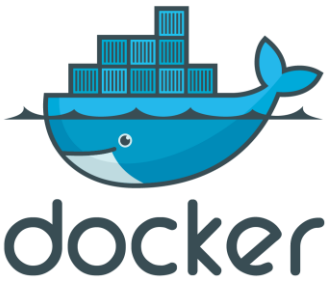
以前发布软件时，以二进制文件，如：office-2012.exe 。

扩展：
LXC 为 Linux Container 的简写。Linux Container 容器是一种内核虚拟化技术，可以提供轻量级的虚拟化，以便隔离进程和资源，而且不需要提供指令解释机制以及全虚拟化的其他复杂性。
LXC 主要通过来自 kernel 的 namespace 实现每个用户实例之间的相互隔离，通过 cgroup 实现对资源的配额和度量。

官方网站：
https://www.docker.com/



logo:



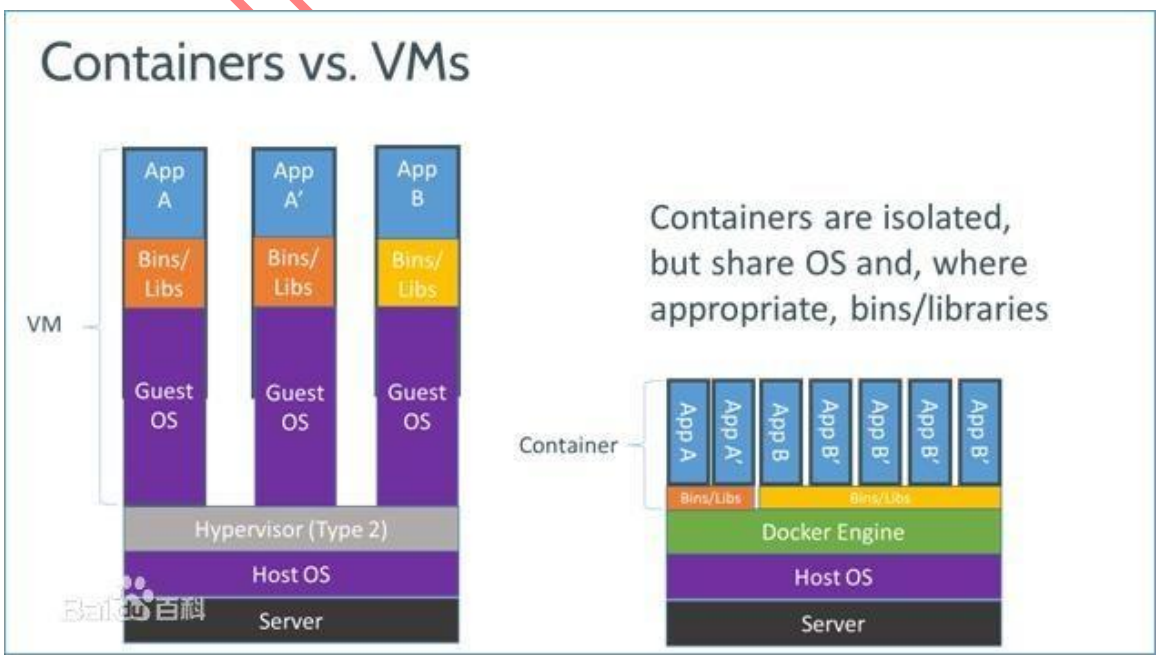
注：docker- engine 相当于鲸鱼，container 容器就是集装箱。

container ：集装箱，容器

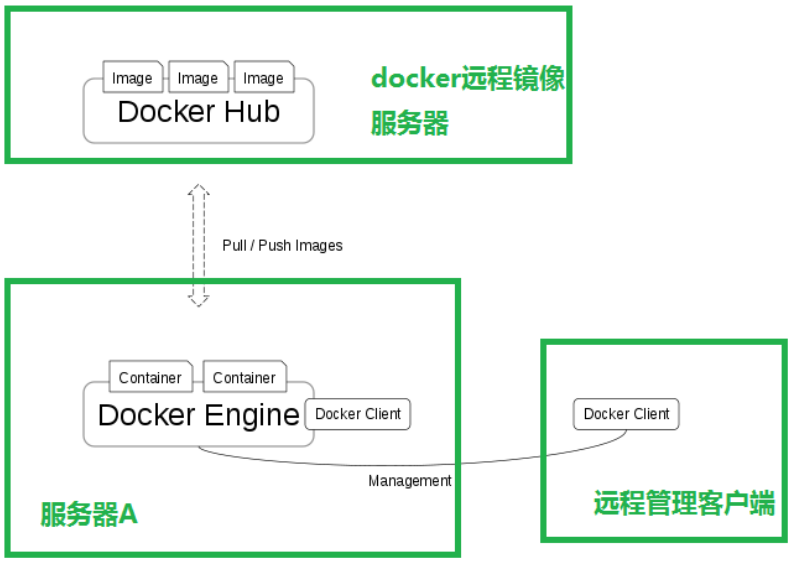
源代码下载：

https://github.com/docker/docker

docker 容器技术和虚拟机对比：



Docker 架构



工作流程：服务器 A 上运行 docker Engine 服务，在 docker Engine 上启动很多容器 container ， 从外网 Docker Hub 上把 image 操作系统镜像下载来，放到 container 容器运行。 这样一个虚拟机实例就运行起来了。

最后，通过 Docker client 对 docker 容器虚拟化平台进行控制。

Image 和 Container 的关系：image 可以理解为一个系统镜像，Container 是 Image 在运行时的一个状态。

如果拿虚拟机作一个比喻的话，Image 就是关机状态下的磁盘文件，Container 就是虚拟机运行时的磁盘文件，包括内存数据。

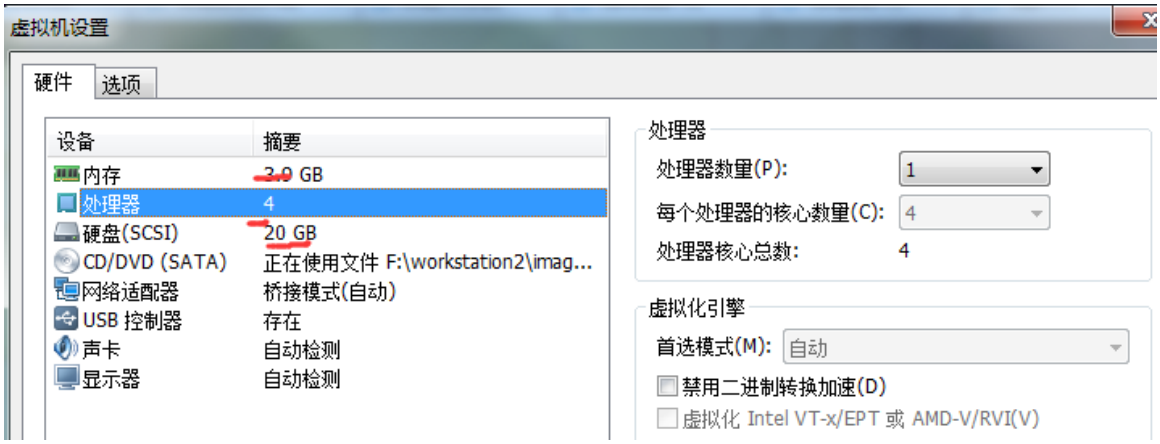
dockerhub：dockerhub 是 docker 官方的镜像存储站点，其中提供了很多常用的镜像供用户下载，如 ubuntu，centos 等系统镜像。通过 dockerhub 用户也可以发布自己的 docker 镜像，为此用户需要注册一个账号，在网站上创建一个 docker 仓库。

Docker 核心技术

1.Namespace — 实现 Container 的进程、网络、消息、文件系统和主机名的隔离。

2.Cgroup — 实现对资源的配额和度量。

注：Cgrout 的配额，就像 vmware 虚拟机中的配置参数：可以指定 cpu 个数，内存大小等



特性：

文件系统隔离：每个进程容器运行在一个完全独立的根文件系统里。

资源隔离：系统资源，像 CPU 和内存等可以分配到不同的容器中，使用 cgroup。

网络隔离：每个进程容器运行在自己的网络空间，虚拟接口和 IP 地址。

日志记录：Docker 将会收集和记录每个进程容器的标准流（stdout/stderr/stdin），用于实时检索或批量检索。

变更管理：容器文件系统的变更可以提交到新的镜像中，并可重复使用以创建更多的容器。无需使用模板或手动配置。

交互式 shell：Docker 可以分配一个虚拟终端并关联到任何容器的标准输入上，例如运行一个一次性交互 shell。

优点：

1.一些优势和 VM 一样，但不是所有都一样。

比 VM 小，比 VM 快，Docker 容器的尺寸减小相比整个虚拟机大大简化了分布到云和从云分发时间和开销。Docker 启动一个容器实例时间很短，一两秒就可以启动一个实例。

2.对于在笔记本电脑，数据中心的虚拟机，以及任何的云上，运行相同的没有变化的应用程序，IT 的发布速度更快。

Docker 是一个开放的平台，构建，发布和运行分布式应用程序。

Docker 使应用程序能够快速从组件组装和避免开发和生产环境之间的摩擦。

3.您可以在部署在公司局域网或云或虚拟机上使用它。

4.开发人员并不关心具体哪个 Linux 操作系统

使用 Docker，开发人员可以根据所有依赖关系构建相应的软件，针对他们所选择的操作系统。

然后，在部署时一切是完全一样的，因为一切都在 DockerImage 的容器在其上运行。

开发人员负责并且能够确保所有的相关性得到满足。

5.Google，微软，亚马逊，IBM 等都支持 Docker。

缺点局限性：

1.Docker 支持 Unix/Linux 操作系统，不支持 Windows 或 Mac（即使可以在其上安装，不过也是基于 Linux 虚拟机的）

2.Docker 用于应用程序时是最有用的，但并不包含数据。日志，跟踪和数据库等通常应放在 Docker 容器外。

实战 1：部署 docker 容器虚拟化平台

实验环境： RHEL7.2 64 位 IP： 192.18.1.63

配置 Docker YUM 源

配置 Docker YUM 源

```
[root@xuegod63 yum.repos.d]# vim /etc/yum.repos.d/docker.repo
[dockerrepo]
name=Docker Repository
baseurl=http://yum.dockerproject.org/repo/main/centos/7/
enabled=1
gpgcheck=0
```

开启路由转发功能，否则容器中的实例上不了网。

```
[root@xuegod63 ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
```

安装 docker-engine 软件包

```
[root@xuegod63 yum.repos.d]# yum -y install docker-engine
```

注：查看下载下来的软件包，可以保存到本地，方便后期使用

```
[root@xuegod63 packages]# pwd
/var/cache/yum/x86_64/7Server/dockerrepo/packages
[root@xuegod63 packages]# ls
docker-engine-1.10.2-1.el7.centos.x86_64.rpm
docker-engine-selinux-1.10.2-1.el7.centos.noarch.rpm
```

```
[root@xuegod63 ~]# systemctl start docker     #启动 docker 服务
[root@xuegod63 ~]# systemctl enable docker    #设置开机启动 docker 服务
```

```
[root@xuegod63 docker]# docker version     #显示 Docker 版本信息。
Client:
 Version:      1.10.2
 API version:  1.22
 Go version:   go1.5.3
 Git commit:   c3959b1
 Built:        Mon Feb 22 16:16:33 2016
 OS/Arch:      linux/amd64
Server:
 Version:      1.10.2
 API version:  1.22
 Go version:   go1.5.3
 Git commit:   c3959b1
 Built:        Mon Feb 22 16:16:33 2016
 OS/Arch:      linux/amd64
```

查看 docker 信息（确认服务运行）显示 Docker 系统信息，包括镜像和容器数。

[root@xuegod63 ~]# docker info
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: 1.10.2
Storage Driver: devicemapper
 Pool Name: docker-8:3-50861283-pool
 Pool Blocksize: 65.54 kB
 Base Device Size: 10.74 GB
 Backing Filesystem: xfs
 Data file: /dev/loop0

[root@xuegod63 ~]# docker search centos #从 Docker Hub 中搜索符合条件的镜像。

```
[root@xuegod63 ~]# docker search centos
NAME                              DESCRIPTION                                     STARS     OFFICIAL   AUTO
MATED
centos                            The official build of CentOS.                   1965      [OK]
jdeathe/centos-ssh                CentOS-6 6.7 x86_64 / CentOS-7 7.2.1511 x8...   15                   [OK]
million12/centos-supervisor       Base CentOS-7 with supervisord launcher, h...   9                    [OK]
blalor/centos                     Bare-bones base CentOS 6.5 image                8                    [OK]
nimmis/java-centos                This is docker images of CentOS 7 with dif...   7                    [OK]
torusware/speedus-centos          Always updated official CentOS docker imag...   7                    [OK]
centos/mariadb55-centos7                                                          3                    [OK]
nathonfowlie/centos-jre           Latest CentOS image with the JRE pre-insta...   3                    [OK]
nickistre/centos-lamp             LAMP on centos setup                            3                    [OK]
consol/sakuli-centos-xfce         Sakuli end-2-end testing and monitoring co...   2                    [OK]
yajo/centos-epel                  CentOS with EPEL and fully updated              1                    [OK]
layerworx/centos                  CentOS container with etcd, etcdctl, confd...   1                    [OK]
```

名字　　描述　　受欢迎成度　是否官方提供
如果 OFFICIAL 为[ok] ，说明可以放心使用。

方法 1：从公网 docker hub 拉取（下载）image　pull：拉
[root@xuegod63 ~]# docker pull centos #从 Docker Hub 中拉取或者更新指定镜像。
Using default tag: latest
latest: Pulling from library/centos
a3ed95caeb02: Pull complete
a07226856d92: Pull complete
Digest:
sha256:1272ae53bac7bf054dd209a0b4a8629bcc39526c2a767427c7639b630a224a9e
Status: Downloaded newer image for centos:latest

导入 image 方法二：
docker load -i centos-latest-docker-image.tar #-i " centos-latest-docker-image.tar " 指定载入的镜像归档。

查看 images 列表
[root@xuegod63 ~]# docker images #列出本地所有镜像。其中 [name] 对镜像名称进行关键词查询。
REPOSITORY　　TAG　　IMAGE ID　　CREATED　　SIZE

centos latest 0f0be3675ebb 11 days ago 196.6 MB

注：Docker 的镜像以及一些数据都是在/var/lib/docker 目录下

[root@xuegod63 docker]# du -sh /var/lib/docker/*

616K /var/lib/docker/containers

248M /var/lib/docker/devicemapper

732K /var/lib/docker/image

32K /var/lib/docker/network

4.0K /var/lib/docker/tmp

0 /var/lib/docker/trust

0 /var/lib/docker/volumes

实战 2： docker 平台基本使用方法

例 1：运行一个 container 并加载镜像 centos，运行起来这个实例后，在实例中执行 /bin/bash 命令

[root@xuegod63 ~]# docker run -i -t centos /bin/bash

#启动一个容器，在其中运行指定命令。

-i 以交互模式运行容器，通常与 -t 同时使用；

-t 为容器重新分配一个伪输入终端，通常与 -i 同时使用；

查看现在容器运行的 linux 环境：

[root@068fd8c70344 /]# ls

anaconda-post.log dev home lib64 media opt root sbin sys usr

bin etc lib lost+found mnt proc run srv tmp var

[root@f072b5ae7542 /]# cat /etc/redhat-release

CentOS Linux release 7.2.1511 (Core)

退出容器：

[root@f072b5ae7542 /]#exit

例 2：在 container 中启动一个长久运行的进程

[root@xuegod63 ~]# JOB=$(docker run -d centos /bin/sh -c "while true;do echo hello world; sleep 1; done")

#-d 后台运行容器，并返回容器 ID；

-c 待完成

[root@xuegod63 docker]# echo $JOB

40e2f9d9f620d11fbe0a395ac422bd02db451f39f244954ee2ac65c18f1a3758

从一个容器中取日志，查看标准输出或输入的内容

语法：docker logs 容器的 Name/ID

[root@xuegod63 ~]# docker logs $JOB

hello world

hello world

hello world

hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world

查看正在运行的容器：

[root@xuegod63 ~]# docker ps　#列出所有运行中容器。

```
[root@xuegod63 ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
c4a213627f1b        centos              "/bin/sh -c 'while tr"   3 minutes ago       Up 3 minutes                            sad_mclean
[root@xuegod63 ~]#
```

[root@xuegod63 ~]# docker ps -a　#-a 列出所有容器（包含沉睡/退出状态的容器）；

```
[root@xuegod63 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
c4a213627f1b        centos              "/bin/sh -c 'while tr"   6 minutes ago       Up 6 minutes                                    sad_mclean
c3117737525c        centos              "/bin/bash"              10 minutes ago      Exited (0) 9 minutes ago                        adoring_no
rthcutt
```

[root@xuegod63 ~]# docker images　#列出所有本地镜像

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
centos	latest	0f0be3675ebb	12 days ago	196.6 MB

例 3：杀死一个容器

查看要杀死容器的 ID：

[root@xuegod63 ~]# docker ps -a　#-a 列出所有容器（包含沉睡/退出状态的容器）；

```
[root@xuegod63 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
c4a213627f1b        centos              "/bin/sh -c 'while tr"   6 minutes ago       Up 6 minutes                                    sad_mclean
c3117737525c        centos              "/bin/bash"              10 minutes ago      Exited (0) 9 minutes ago                        adoring_no
rthcutt
```

杀死 ID 为 c4a213627f1b 的容器

[root@xuegod63 ~]# docker kill c4a213627f1b

c4a213627f1b

或

[root@xuegod63 ~]# docker kill $JOB　#　杀死一个容器

例 4：启动、停止、重启 container 容器

[root@xuegod63 ~]#　JOB=$(docker run -d centos /bin/sh -c "while true;do echo hello world; sleep 1; done")

查看容器：

```
[root@xuegod63 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
1a63ddea6571        centos              "/bin/sh -c 'while tr"   6 seconds ago       Up 4 seconds                                    adoring_
spence
c4a213627f1b        centos              "/bin/sh -c 'while tr"   11 minutes ago      Exited (137) 2 minutes ago                      sad_mcle
an
c3117737525c        centos              "/bin/bash"              15 minutes ago      Exited (0) 14 minutes ago                       adoring_
northcutt
```

[root@xuegod63 ~]# docker stop 1a63ddea6571　关闭容器

1a63ddea6571

查看：

```
[root@xuegod63 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
1a63ddea6571        centos              "/bin/sh -c 'while tr"   52 seconds ago      Exited (137) 4 seconds ago                      adoring_
spence
c4a213627f1b        centos              "/bin/sh -c 'while tr"   12 minutes ago      Exited (137) 3 minutes ago                      sad_mcle
an
c3117737525c        centos              "/bin/bash"              15 minutes ago      Exited (0) 15 minutes ago                       adoring_
northcutt
```

[root@xuegod63 ~]# docker start 1a63ddea6571

1a63ddea6571

```
[root@xuegod63 ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
1a63ddea6571        centos              "/bin/sh -c 'while tr"   About a minute ago  Up 4 seconds                                    adoring
_spence
c4a213627f1b        centos              "/bin/sh -c 'while tr"   12 minutes ago      Exited (137) 3 minutes ago                      sad_mcl
ean
c3117737525c        centos              "/bin/bash"              16 minutes ago      Exited (0) 15 minutes ago                       adoring
_northcutt
```

[root@xuegod63 ~]# docker restart 1a63ddea6571

1a63ddea6571

删除指定 container

[root@xuegod63 ~]# docker rm 1a63ddea6571

Failed to remove container (1a63ddea6571): Error response from daemon: Conflict, You cannot remove a running container. Stop the container before attempting removal or use -f

解决：你可以先把容器 1a63ddea6571 关闭，然后再删除或加-f 强制删除

[root@xuegod63 ~]# docker rm -f 1a63ddea6571

1a63ddea6571

实战 4：Docker Image 的制作两种方法

方法 1：docker commit　#保存 container 的当前状态到 image 后，然后生成对应的 image

方法 2：docker build　#使用 Dockerfile 自动化制作 image

方法 1：docker commit

创建一个新的 container 容器，并安装好 nmap 工具

[root@xuegod63 ~]# docker run -ti centos /bin/bash

[root@1d3563200047 /]# yum -y install nmap-ncat　#在 container 中安装 nmap-ncat 软件包

[root@1d3563200047 /]# exit

查看 images 列表

[root@xuegod63 ~]# docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
centos	latest	0f0be3675ebb	12 days ago	196.6 MB

注：当前只有一个 image centos

根据容器当前状态做一个 image 镜像：创建一个安装了 nmap-ncat 工具的 centos 镜像

提交 image 语法： docker commit <container 的 ID> <image_name>

例：

查看容器 ID：

```
[root@xuegod63 ~]# docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED          STATUS                      PORTS   NAMES
1d3563200047   centos    "/bin/bash"   10 minutes ago   Exited (0) 6 minutes ago            tender_joliot

[root@xuegod63 ~]# docker commit 1d3563200047 centos:nmap-ncat
sha256:e5917c01599c70d0680beeb35f6df98889dd22106399efd6907d956d8a943242
[root@xuegod63 ~]# docker images
REPOSITORY   TAG         IMAGE ID       CREATED          SIZE
centos       nmap-ncat   e5917c01599c   8 seconds ago    198.3 MB
centos       latest      0f0be3675ebb   12 days ago      196.6 MB
```

使用新创建的 nmap-ncat 镜像，启动一台容器：

```
[root@xuegod63 ~]# docker run -ti centos:nmap-ncat  /bin/bash
```

查看：

```
[root@f9ab13f299ee /]# rpm -qa nmap-ncat  #已经安装好 nmap-ncat 命令
nmap-ncat-6.40-7.el7.x86_64
```

注：说明基于 nmap-ncat 镜像的容器创建成功。

方法二：通过：docker build 创建一个基于 centos 的 httpd web 服务器镜像。

以下操作要在 docker 物理机上操作：

1、创建工作目录

```
[root@xuegod63 ~]#  mkdir /docker-build
[root@xuegod63 ~]# cd /docker-build
[root@xuegod63 docker-build]# touch Dockerfile
```

注： make 自动化编译时需要 Makefile 文件，自动化创建 docker 镜像时，需要 Dockerfile

2、编辑 Dockerfile

Dockerfile 用来创建一个自定义的 image,包含了用户指定的软件依赖等。

```
[root@xuegod63 docker-build]# vim Dockerfile
FROM centos
MAINTAINER userabc <userabc@gmail.com>
RUN yum -y install httpd
ADD start.sh /usr/local/bin/start.sh
ADD index.html /var/www/html/index.html
```

注释：

```
FROM centos    # FROM 基于本地哪个镜像，这里基于 centos 镜像
MAINTAINER userabc <userabc@gmail.com>   # MAINTAINER 镜像创建者
RUN yum -y install httpd    #RUN 安装软件用
ADD start.sh /usr/local/bin/start.sh
```

ADD index.html /var/www/html/index.html
ADD 将文件<src>拷贝到新产生的镜像的文件系统对应的路径<dest>。所有拷贝到新镜像中的文件和文件夹权限为 0755,uid 和 gid 为 0

创建 start.sh 脚本启动 httpd 服务
[root@xuegod63 docker-build]# echo "/usr/sbin/httpd -DFOREGROUND" > start.sh
注：/usr/sbin/httpd -DFOREGROUND 相当于执行了 systemctl start httpd
[root@xuegod63 docker-build]# chmod a+x start.sh

创建 index.html
echo "docker image build test" > index.html

使用命令 build 来创建新的 image
语法：docker build -t 父镜像名：自己定义的镜像名 Dockerfile 文件所在路径
-t :表示 tag，用于指定新的镜像名

[root@xuegod63 docker-build]# docker build -t centos:httpd .
注： . 表示当前目录。另外你的当前目录下要包含 Dockerfile,使用命令 build 来创建新的 image,并命名为 centos:httpd
Sending build context to Docker daemon 4.096 kB
Step 1 : FROM centos
 ---> 0f0be3675ebb
Step 2 : MAINTAINER userabc <userabc@gmail.com>
 ---> Using cache
 ---> 9d1cc5ad2a7b
Step 3 : RUN yum -y install httpd
。。。
Complete!
 ---> bce6b3f0a700
Removing intermediate container c9567092d67b
Step 4 : ADD start.sh /usr/local/bin/start.sh
 ---> 521463f9bbeb
Removing intermediate container 18b34849606d
Step 5 : ADD index.html /var/www/html/index.html
 ---> 585eb8e1d7ad
Removing intermediate container ecdbd06a3c1e
Successfully built 585eb8e1d7ad

查看 images 列表
[root@xuegod63 docker-build]# docker images

```
[root@xuegod63 docker-build]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
centos              httpd               585eb8e1d7ad        5 minutes ago       298.9 MB
centos              nmap                e5917c01599c        6 hours ago         198.3 MB
centos              latest              0f0be3675ebb        12 days ago         196.6 MB
```

Docker Image 的发布：

方法 1：Save Image To TarBall

方法 2：Push Image To Docker Hub

方法 1：Save Image To TarBall

保存 Image 到 tar 包

语法：docker save -o 导出的镜像名.tar 本地镜像名

例：

[root@xuegod63 docker-build]# docker save -o centos-httpd-docker-image.tar centos:httpd

[root@xuegod63 docker-build]# ls #查看导出成功

centos-httpd-docker-image.tar Dockerfile index.html start.sh

后期使用方法：

[root@xuegod63 docker-build]# docker load -i centos-httpd-docker-image.tar

方法 2：Push Image To Docker Hub 发布到外网

1、Signup on docker hub & create repo 注册一个帐号

https://hub.docker.com/

2、Login to docker hub

 # docker login -u userabc -p abc123 -e userab@gmail.com

3、Push image to docker hub

 # docker push centos:httpd

4、Pull image from docker hub

 # docker pull userabc/centos:httpd

用户名/镜像名

实战 5：Container 端口映射，使用新生成 centos:httpd 镜像，启动一个容器，然后容器中的 80 端口映射到 docker 物理上的 9000 端口上。

启动 container

[root@xuegod63 ~]# docker run -d -p 9000:80 centos:httpd /bin/sh -c /usr/local/bin/start.sh

87fadc0249a96736f588f16b7d3ad662ef3536a06d7a74115cd7c76546ed3a22

注： -p 9000:80 把容器中的 80 端口映射到物理机上的 9000 端口

在物理机上查看容易状态：

[root@xuegod63 docker-build]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

a9c3a7bf2054 centos:httpd "/bin/sh -c /usr/loca" About a minute ago Up About a minute 0.0.0.0:9000->80/tcp goofy_lovelace

查看物理机上开启的 9000 代理端口

[root@xuegod63 ~]# netstat -antup | grep 9000 #查看物理机上端口已经开启

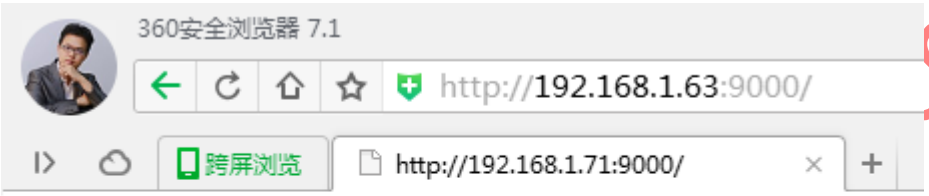
tcp6 0 0 :::9000 :::* LISTEN

25232/docker-proxy

[root@xuegod63 ~]# curl http://127.0.0.1:9000

docker image build test

或：http://192.168.1.63:9000/



docker image build test

实战 6：访问正在运行的 container

语法： docker exec -ti <container id | name> /bin/bash

查看正在运行的容器 ID：

[root@xuegod63 ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED

STATUS PORTS NAMES

87fadc0249a9 centos:httpd "/bin/sh -c /usr/loca" 4 minutes ago Up 4 minutes

0.0.0.0:9000->80/tcp elated_perlman

[root@xuegod63 ~]# docker exec -ti 87fadc0249a9 /bin/bash #进入容器

创建 test.html 文件

[root@87fadc0249a9 /]# echo xuegod > /var/www/html/test.html

[root@87fadc0249a9 /]#

测试：在物理机上查看新添加的 test.html 文件

[root@xuegod63 ~]# curl http://127.0.0.1:9000/test.html

xuegod

查看物理机和容器的网络：

查看容器的 IP：

[root@a9c3a7bf2054 /]# yum install net-tools

[root@a9c3a7bf2054 /]# ifconfig

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.17.0.4 netmask 255.255.0.0 broadcast 0.0.0.0

物理机的 IP：
[root@xuegod63 ~]# ifconfig
docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.17.0.1 netmask 255.255.0.0 broadcast 0.0.0.0

测试网络：
[root@xuegod63 ~]# ping 172.17.0.4
PING 172.17.0.4 (172.17.0.4) 56(84) bytes of data.
64 bytes from 172.17.0.4: icmp_seq=1 ttl=64 time=0.414 ms

配置容器 root 密码：
[root@a9c3a7bf2054 /]# echo 123456 | passwd --stdin root