Ubuntu Server 从入门到精通

第19章：容器

容器

有了新武器，并不表示旧武器没有用

- 有了原子弹为什么还要发展常规武器
  - 核威慑
- 你能干，就什么都让你来干
  - 乔丹得分
- 新都源自旧
  - 创新不等于打破一切旧的东西（WG）
  - 想要创新最大的困难是可有的现有技术太少

容器

有了新武器，并不表示旧武器没有用

- 虚拟化技术变革了数据中心技术
  - 一切都软件定义 SDN
  - 在一个物理机上运行多个VM，切割利用物理机资源
  - 最大的问题是分派给VM系统的资源浪费（1G内存的VM，APP仅用100M）
  - 虽可共享未使用资源，但效率损失较大
- 容器再次颠覆IT业界
  - 轻量的虚拟化环境（开销小），但不是我们通常意义认为的虚拟机服务器
  - 容器使用宿主机的CPU（VM有自己的CPU）
  - 容器共享宿主机的OS内核和只读库
  - 容器也可实现应用隔离（与VM相同又不同）
  - 可移植（便携）的软件实例，提高部署效率，改变开发方式

容器

有了新武器，并不表示旧武器没有用

- 什么是容器
  - 建议将容器看作一个文件系统（而非VM）
  - 在Linux内核里打造的轻便、接近裸机速度的虚拟环境
  - 容器包含一个与宿主一致的文件结构
    - U上的容器拥有一致的文件系统，看起来像是VM或物理机
  - 拷贝所有 Ubuntu中的文件，放到一个新的目录中
    - 独立运行二进制程序，而不运行实际的操作系统
    - 应用运行在这个独立的文件系统中，与宿主隔离
  - 容器只包含应用运行所必需的最小资源集
  - 一个容器倾向于只运行一个应用（也可运行多个）
  - 熟悉之后你会发现自己更喜欢容器，而不是VM
  - 云时代容器技术会得到更多的重视

容器

有了新武器，并不表示旧武器没有用

- 可移植性
  - 可移植性是容器的另一重大优势，是容器设计的核心技术
  - 保证在不同宿主系统上获得完全相同的开发运行环境
  - VM也可导入导出，但容器移植更简单
- 容器并非新技术
  - Docker 是目前最火的容器
  - LXC（Linux Container）

容器

有了新武器，并不表示旧武器没有用

- 虚拟机技术是不是完蛋了？
  - 目前虚拟化更成熟
  - 虚拟化拥有自己独特的优势
  - 生态已经非常成熟（产品、解决方案、管理工具）
  - 某些场景更适合使用虚拟化
  - 会与容器长期共存（多长的长期？）
- 并非所有应用都可运行于容器中
  - WEB应用或各种服务
  - 容器还出于飞速的进化发展阶段，特性很吸引人，但还远没有一统江湖

容器

有了新武器，并不表示旧武器没有用

虚拟机	容器
表现为硬件虚拟化	系统 / 应用虚拟化
重量	轻量
交付速度慢	事实交付、可扩展
性能开销大	接近裸机性能
全隔离更安全	进程级隔离
生态成熟	发展很快，尚未完善

容器

有了新武器，并不表示旧武器没有用

- Docker vs LXD/LXC
- Docker 目前最火（优秀的市场营销）
  - Docker 利用分层的方法实现容器化
  - 对容器做的所有修改都会产生一个新层，这些层将成为其他容器的依赖基础
- LXD 源于 LXC（Lex-C）
  - Docker 最早基于 LXC（所有容器技术的鼻祖）
  - LXC 基于控制组实现进程隔离（提高安全性）
  - LXD 最初由Canonical创建，曾经是 Ubuntu 独有（目前使用 snap 包管理）
  - LXD 是对 LXC 的增强提高
    - 支持快照、ZFS、迁移
  - 可认为是LXC在其上增加了额外的管理层和功能

容器

有了新武器，并不表示旧武器没有用

- Docker vs LXD/LXC
  - LXC 可视为机器容器，更接近VM
  - LXC 基于一个文件系统实现容器化，可以从宿主机系统访问
  - Docker 努力地将自己与虚拟机区分开来
  - Docker 可视为应用程序容器，提供运行应用程序所需的基础
  - Docker 每个任务都运行在单独的层中
- 该选择 Docker 还是 LXD/LXC
  - 都用，按应用的环境要求选择
  - Docker 可以在Linux、macOS、Windows上运行Docker容器
  - LXC/LXD 几乎可以运行在所有 Linux 发行版上
  - 使用容器服务运行容器，而非自建容器服务器（Amazon ECS）
  - Docker 商业、社区较成功（Docker Hub）

其他容器技术

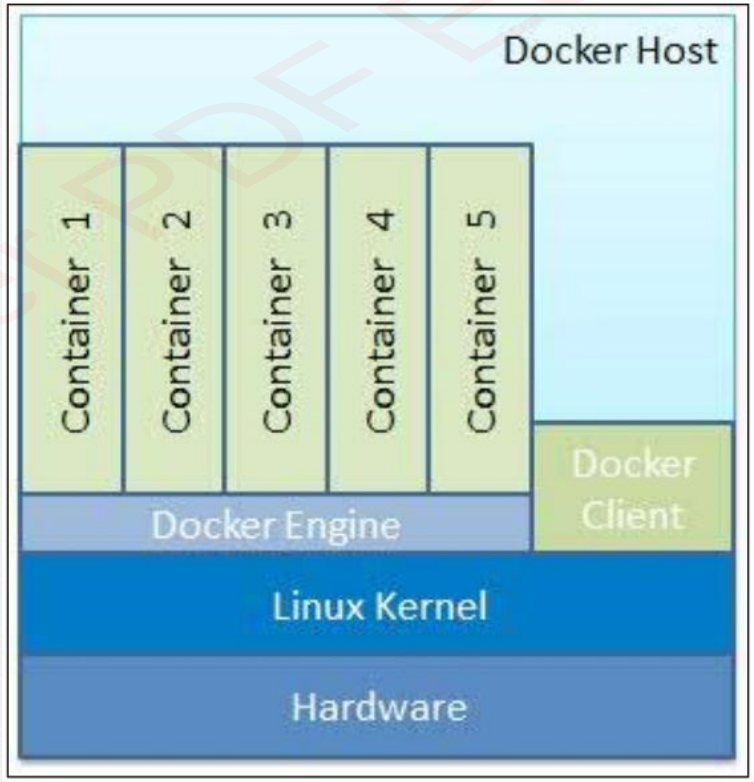
有了新武器，并不表示旧武器没有用

- OpenVZ：基于Linux内核的操作系统级虚拟化技术
  - 虚拟的系统示例称为
    - containers 容器
    - virtual private servers (VPSs)
    - virtual environments (VEs)
- FreeBSD jail
  - OS-Level 虚拟化技术
  - 将基于freebsd的系统划分为多个称为 **监狱** 的独立微型系统
- WPARs（The AIX Workload partitions）
  - 工作负载分区（OS-Level 虚拟化技术）
- Solaris Containers
  - Solaris Zones

Docker

有了新武器，并不表示旧武器没有用

- Docker 是一个开源的容器引擎
- Docker **容器**包含独立运行软件所需的一切
  - 二进制、库、配置文件、脚本、jar 包等
  - 容器之间完全隔离，各自独立的根文件目录
- Docker 容器有自己的进程空间和网络接口
- 组件
  - Docker engine：生成监视和管理所有容器
  - Docker Hub：映像库
  - Docker Client：管理工具



Docker

有了新武器，并不表示旧武器没有用

- Docker 引擎只可以直接运行在 Linux系统上
  - 利用Boot2Docker等适配器帮助，使用轻量级Linux vm可在Mac和微软系统上运行
- 安装Docker引擎
  - uname -m　　# 仅64位系统保障（32位系统可用）
  - sudo apt install docker.io
  - systemctl status docker
  - 一个老版本的Ubuntu包名为 docker
- 将当前用户加入 docker 组
  - sudo usermod -aG docker ${USER}
  - 重启系统

Docker

有了新武器，并不表示旧武器没有用

- ubuntu 官方仓库中不是 docker 的最新版本
- 安装依赖包
  - sudo apt install apt-transport-https ca-certificates curl software-properties-common
- 添加docker官方库 GPG key
  - curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
- 添加docker官方库更新源
  - sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
- 安装
  - sudo apt update && sudo apt install docker-ce

Docker

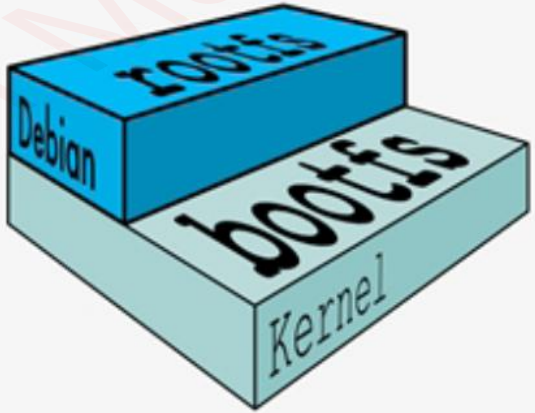
有了新武器，并不表示旧武器没有用

- 查看信息
  - docker version
  - docker info
- Docker registry
  - 应用程序仓库（基本linux映像、高级应用程序映像）
- 下载映像
  - docker pull busybox
- 查看映像
  - docker images
- 运行程序
  - docker run busybox echo "Hello World!"

名词解释

有了新武器，并不表示旧武器没有用

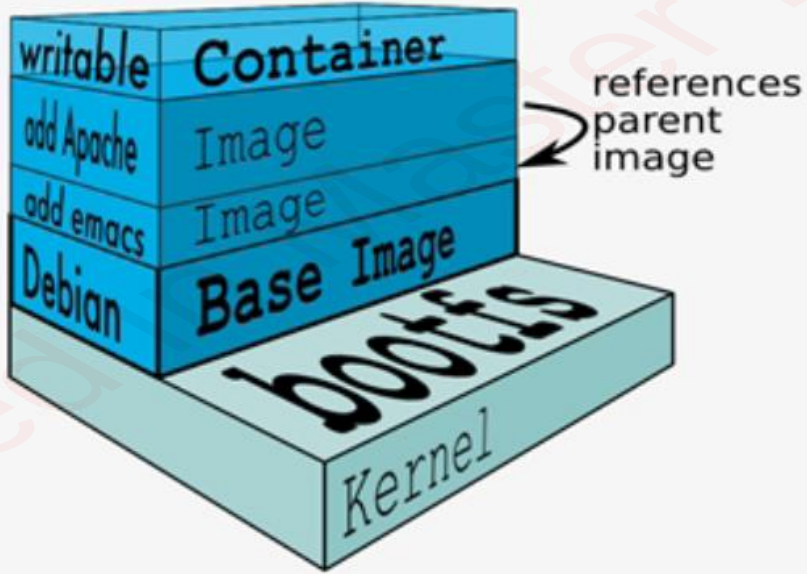
- Docker 映像
  - 组成应用程序运行所需的全部文件合集
  - 只读 / 读写 分层结构（每次对映像的修改都通过commit形成一个新层）
  - 容器层 是基于映像可读写的顶层
  - 所有的 Docker 映像都源自一个 基础映像（Debian / Ubuntu）
  - 其他的功能模块附加于基础映像形成最终程序



名词解释

有了新武器，并不表示旧武器没有用

- Docker images 是构建 Docker container 的基础
- Docker 层 / 容器层



名词解释

有了新武器，并不表示旧武器没有用

- 每个映像有唯一的ID（SHA256/6字节）
  - docker images
  - docker images --no-trunc
  - REPOSITORY：用户名 / 库名
  - TAG：标签（latest / 其他变种）
  - IMAGE ID：SHA256，默认只显示6字节
  - CREATED：创建日期
  - SIZE：映像大小
- 每个容器也有唯一的ID（SHA256/6字节）
  - docker run busybox echo "Hello World!"
  - docker run -t -i busybox:ubuntu-14.04
  - docker ps -a

Docker注册中心

有了新武器，并不表示旧武器没有用

- Docker Registry
  - 提供开放的映像注册、查找、访问、使用中央位置
  - 官方注册中心提供多重检查校验的高质量映像（可个人自建）
  - 注册用户可以发布自己定制的映像
- Repository
  - 映像在注册中心里的具体存储位置（命名空间）
  - 注册用户拥有自己的 库
- Docker Hub Registry
  - Docker 社区创建的官方映像库 / 第三方映像分享平台
  - docker search ubuntu（默认访问地址 index.docker.io）
  - docker pull ubuntu（数字签名保，异常告警）
  - docker pull registry.example.com/myapp（自建映像库）

Docker

有了新武器，并不表示旧武器没有用

- 搜索映像
  - docker search mysql
- 交互登陆
  - docker run --name aa -it ubuntu /bin/bash　　# 创建容器
  - Ctrl + D / exit　　# 停止、推出
  - docker ps -a --no-trunc
  - docker start 32a21c6613c4
  - (Ctrl + P) + (Ctrl + Q)　　# 保持运行
  - docker attach yuanfh
  - CONTAINER ID、IMAGE、COMMAND、CREATED、STATUS、
  - PORTS：端口映射（对外提供服务）
  - NAMES：自动随即生成的容器名（形容词_名词）

Docker

有了新武器，并不表示旧武器没有用

- 容器 与 映像的变化
  - docker diff f9e5e70fd58d
  - C：修改
  - A：增加
  - D：删除
- 日常管理命令
  - docker start f9e5e70fd58d
  - docker stop f9e5e70fd58d
  - docker restart f9e5e70fd58d
  - docker rename f9e5e70fd58d yuanfh
  - docker pause f9e5e70fd58d
  - docker unpause f9e5e70fd58d

Docker

有了新武器，并不表示旧武器没有用

- 创建映像
  - docker run -it ubuntu /bin/bash
  - touch {a,b,c}.txt
  - Ctrl + P / Ctrl + Q
  - docker commit f9e5e70fd58d yuanfh/ubuntu_txt:1.0
- 基于新映像创建容器
  - docker run -it yuanfh/ubuntu_txt /bin/bash
- 后台运行容器
  - docker run -d ubuntu /bin/bash -c "while true; do date; sleep 5; done"
- 查看日志
  - docker logs 1ea0c21b037e

Docker

有了新武器，并不表示旧武器没有用

- 运行 Apache 服务
  - docker run -dit -p 8080:80 ubuntu /bin/bash
  - docker attach dfb3e4dfb3e4
  - apt update && apt install apache2
  - Ctrl + P 、 Ctrl + Q
  - 宿主机浏览器访问 http://localhost:8080
- 服务自动启动
  - apt install nano
  - echo '/etc/init.d/apache2 start' >> /etc/bash.bashrc
- 创建新映像
  - docker commit dfb3e4dfb3e4 ubuntu/apache-server:1.0

Docker

有了新武器，并不表示旧武器没有用

- 使用Dockerfiles自动创建映像
  - Dockerfiles 文本文件，包含创建容器的指令
  - vi Dockerfile

```
FROM ubuntu
MAINTAINER Yuanfh <yfh131@sina.com>
RUN apt update; apt dist-upgrade -y
RUN apt install -y apache2 vim
RUN echo "/etc/init.d/apache2 start" >> /etc/bash.bashrc
```

  - docker build -t test/apache-server:1.0 .
- 基于映像创建容器
  - docker run -dit -p 8080:80 test/apache:1.0 /bin/bash

LXD

有了新武器，并不表示旧武器没有用

- 安装
  - sudo snap install lxd　　# sudo apt install lxd
  - sudo usermod -aG lxd <username>
  - 重启主机
- 初始化
  - lxd init
  - 回答问题（默认即可）
- 运行容器
  - lxc launch ubuntu:18.04 C01
  - 自动下载映像并创建、运行容器
  - lxd 命令针对 lxd 管理层，容器管理命令仍然使用 lxc

LXD

有了新武器，并不表示旧武器没有用

```
yyy@ubuntu:~$ lxd init
Do you want to configure a new storage pool (yes/no) [default=yes]?
Name of the new storage pool [default=default]:
Name of the storage backend to use (dir, btrfs, lvm) [default=btrfs]:
Create a new BTRFS pool (yes/no) [default=yes]?
Would you like to use an existing block device (yes/no) [default=no]?
Size in GB of the new loop device (1GB minimum) [default=15GB]: 2
Would you like LXD to be available over the network (yes/no) [default=no]?
Would you like stale cached images to be updated automatically (yes/no) [default=yes]?
Would you like to create a new network bridge (yes/no) [default=yes]?
What should the new bridge be called [default=lxdbr0]?
What IPv4 address should be used (CIDR subnet notation, “auto” or “none”) [default=auto]?
What IPv6 address should be used (CIDR subnet notation, “auto” or “none”) [default=auto]?
LXD has been successfully configured.
```

LXD

有了新武器，并不表示旧武器没有用

- 常用管理命令
  - lxc list # 列出容器
  - lxc start <container> # 启动容器
  - lxc stop <container> # 停止容器
  - lxc delete <container> # 删除容器
  - lxc image list # 查看映像
  - lxc image delete <img_name> # 删除映像
- 交互登陆容器
  - lxc exec C01 bash # root 帐号登陆
  - lxc exec C01 -- sudo --login --user ubuntu
    - Ubuntu 映像包含一个默认用户帐户 Ubuntu
  - lxc exec C01 -- apt update

LXD

有了新武器，并不表示旧武器没有用

- 特点
  - lxd 新部署容器过程比Docker更容易
  - 退出容器时不必以特定的方式退出
  - lxd 容器中没有层的概念
  - Docker中的层可以使部署更快（没有清理也可能感觉混乱)
- 退出
  - Ctrl + D / exit
- 随宿主机自动启动容器
  - lxc config set C01 boot.autostart 1

LXD

有了新武器，并不表示旧武器没有用

- 远程映像存储服务器
  - lxc remote list　　# 默认三个
  - images　　# 其他 Linux 发行版映像
  - ubuntu　　# 稳定版 Ubuntu 映像
  - ubuntu-daily　　# 每日最新的 ubuntu 映像
- 从远程复制映像
  - lxc launch images:debian/stretch test01
  - lxc image copy ubuntu:18.10 local: --alias u1810
    - 从 ubuntu 存储复制映像到 local 存储
- 创建容器
  - lxc launch u1810 C02
  - lxc list　　/　　lxc image list

LXD

有了新武器，并不表示旧武器没有用

- 上传下载文件
  - lxc file pull C01/etc/hosts .
  - lxc file push hosts C01/tmp/
- 映像自动更新间隔（小时）
  - lxc config set images.auto_update_interval 24
- 自动清除未使用的映像（天数）
  - lxc config set images.remote_cache_expiry 5
- 查看配置
  - lxc config show

LXD

有了新武器，并不表示旧武器没有用

- 安装 apache2 服务
  - lxc exec C01 Bash
  - apt update && sudo apt install apache2
  - ip addr show
  - curl 1.1.1.1
- 外网访问
  - 防火墙规则路由流量
  - 创建配置文件 DHCP 获取物理网络地址（类似于 VM）
  - 宿主机创建桥接网卡 br0

LXD

有了新武器，并不表示旧武器没有用

- 编辑配置文件
  - lxc network edit lxdbr0

```
description: External access profile
devices:
eth0:
name: eth0
nictype: bridged
parent: br0
type: nic
```

- lxc profile add -C 01 external
- lxc launch ubuntu:18.04 web01 -p default -p external
  - 多个配置文件，后者覆盖前者

Questions？