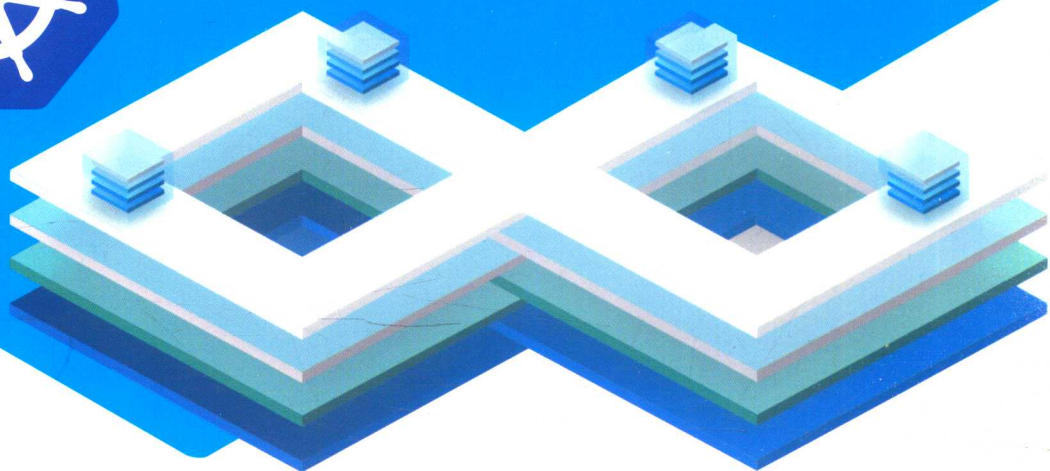


Docker是一个社区所有项目，它建立了容器化的行业标准；同时，世界上主要的云服务商和IT供应商都选择了Kubernetes作为默认的容器编排工具。这就要求所有的运维人员需要理解Docker，学会Kubernetes，这样才能更好地进行容器管理。

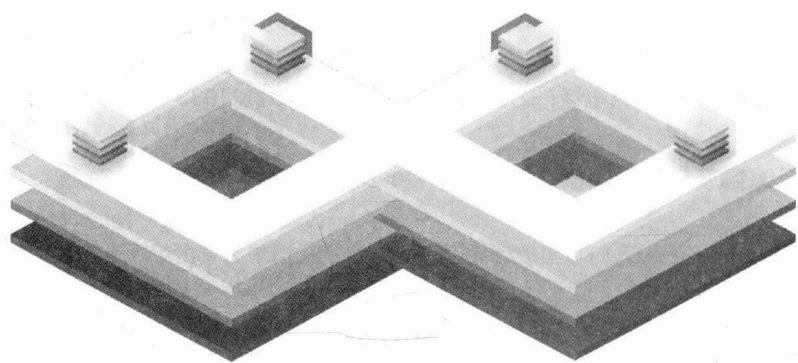


从Docker到Kubernetes 入门与实战

罗利民 著

清华大学出版社





从 Docker 到 Kubernetes

入门与实战



罗利民 著

清华大学出版社
北京

内 容 简 介

Docker 已经成为行业中最火爆的开源技术，没有之一。而 Kubernetes 的诞生，更是使得 Docker 如虎添翼。越来越多的人开始投入到 Docker 和 Kubernetes 的开发和运维中去。本书是一本为初学者量身定做的入门教材，适合对容器技术了解不多，没有太多的相关开发和运维经验，但是又想掌握 Kubernetes 技术的读者使用，帮助他们快速地进入这个领域。

本书分为两篇共 12 章，第一篇全面介绍 Docker，包括 Docker 的基础知识、在主流操作系统上安装 Docker 的方法以及 Docker 的基本管理操作；第二篇详细介绍 Kubernetes，主要包括 Kubernetes 基础知识、Kubernetes 的安装方法、Kubernetes 命令行管理工具、运行应用、访问应用、存储管理、软件包管理、网络管理以及 Kubernetes 的图形管理工具。

本书内容详尽、示例丰富，是广大 Docker 和 Kubernetes 初学者必备的参考书，同时也适合高等院校和培训学校计算机相关专业作为教材使用。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目（CIP）数据

从 Docker 到 Kubernetes 入门与实战/罗利民著. —北京：清华大学出版社，2019
ISBN 978-7-302-53481-5

I. ①从… II. ①罗… III. ①Linux 操作系统—程序设计 IV. ①TP316.85

中国版本图书馆 CIP 数据核字（2019）第 172247 号

责任编辑：夏毓彦

封面设计：王 翔

责任校对：闫秀华

责任印制：杨 艳

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>，<http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社总机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn

质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者：北京密云胶印厂

经 销：全国新华书店

开 本：190mm×260mm

印 张：15

字 数：384 千字

版 次：2019 年 9 月第 1 版

印 次：2019 年 9 月第 1 次印刷

定 价：69.00 元

产品编号：080824-01

前言

读懂本书

还在用虚拟机？是时候开始用 Docker 了

未来五年引领云计算发展的核心技术必然是容器技术。现在越来越多的 IT 公司逐渐抛弃传统的虚拟化技术，而转向轻量化的容器技术。

主流云服务商已经开始支持 Docker

云服务提供商，包括微软、IBM、Rackspace、Google 以及其他主要的 Linux 提供商如 Canonical 和 Red Hat，都开始支持 Docker。

Docker 和 Kubernetes 如何改变传统的虚拟化技术？

Docker 和 Kubernetes 给虚拟化技术带来了革命性的改变，给开发人员以及系统管理员提供了一个平台，使配置和部署分布式应用变得更加容易，让应用真正实现零宕机。

本书真的适合你吗？

本书帮你从传统的虚拟化技术过渡到 Docker，再过渡到 Kubernetes 时代；本书全面而又简洁地介绍了 Docker，轻松扫除初学者学习本书的障碍；本书从最简单的例子出发，逐步深入，使得读者能够在轻松愉快的过程中，学习到新的技术；本书摒弃了过多的理论介绍，突出了日常开发和运维必需的知识，可谓去芜存菁，取精用宏。

本书涉及的技术或框架

虚拟化	容器	Docker
Linux	Linux Container	Git
版本控制	Nginx	反向代理
防火墙	路由	OSI 七层网络模型
子网	网桥	集群
NFS	iSCSI	SAN
Hyper-V	VMware Workstation Pro	域名解析

本书涉及的示例和案例

- 第一个 Docker 应用：Hello world
- 在 Ubuntu 中体验 Docker
- 容器的创建、查看、启动、停止以及删除
- Docker 的网络模式
- 通过软件包管理工具安装 Kubernetes
- 通过源代码安装 Kubernetes
- kubeadm 部署 Kubernetes
- 管理 DaemonSet
- 在 Windows 中体验 Docker
- Docker 镜像的查找、下载、列举、删除、查看、构建以及标签管理
- 容器的互联
- 通过二进制文件安装 Kubernetes
- kubectl 管理资源对象
- 管理 Deployment
- 通过 Job 实现倒计时
- 管理服务
- 通过 NodePort 实现外部访问
- 存储卷管理
- 通过 Helm 管理应用
- kube-proxy 和 ClusterIP 实现外部访问
- 通过负载均衡实现外部访问
- 持久化存储卷管理
- 在 Kubernetes 集群中部署 Tomcat

本书特点

- (1) 本书不论是理论知识的介绍，还是实例的选择，都是从实际应用的角度出发，精心选择运维和开发过程中典型例子，讲解细致，分析透彻。
- (2) 深入浅出、轻松易学，以 Docker 和 Kubernetes 重要知识点为主线，激发读者的阅读兴趣，让读者能够真正学习到 Docker 和 Kubernetes 实用、前沿的技术。
- (3) 技术新颖、与时俱进，结合时下最热门的技术，如微服务、集群以及自动化运维等，让读者在学习 Docker 和 Kubernetes 的同时，扩大知识面，了解和掌握更多的、更先进的运维技术。
- (4) 贴近读者、贴近实际，大量成熟技巧和经验的介绍，帮助读者快速找到问题的最佳答案，及时解决运维和开发过程中遇到的问题。
- (5) 贴心提醒，本书根据需要在各章使用了很多“注意”“提示”等小提示，让读者可以在学习过程中更轻松的理解相关概念及知识点。
- (6) 本书汇集了作者大量的实战经验，不仅可以作为入门教程，还可以作为运维和开发的参考书。

本书读者

- IT 实施和运维工程师
- 软件开发工程师
- 对云服务技术感兴趣，并希望进一步学习的中高级技术人员
- 系统管理员

- 云端原生开发人员
- 想了解容器和 Kubernetes 技术的初学者
- 想从 Docker 转移到 Kubernetes 的技术人员

本书第 1~10 章由平顶山学院的罗利民创作，第 11~12 章由张春晓创作。

作 者
2019 年 5 月

目 录

第 1 章 全面认识 Docker.....	1
1.1 容器技术.....	1
1.1.1 什么是容器.....	1
1.1.2 容器与虚拟机之间的区别.....	3
1.1.3 容器究竟解决了什么问题.....	4
1.1.4 容器的优点.....	5
1.1.5 容器的缺点.....	6
1.1.6 容器的分类.....	7
1.2 Docker 技术.....	8
1.2.1 什么是 Docker.....	8
1.2.2 Docker 的由来.....	9
1.2.3 Docker 究竟是什么.....	9
1.3 Docker 的架构与组成.....	10
1.3.1 Docker 的架构.....	10
1.3.2 Docker 中应用系统的存在形式.....	12
1.4 为什么使用 Docker.....	12
1.4.1 Docker 的应用场景.....	12
1.4.2 Docker 可以解决哪些问题.....	13
1.4.3 Docker 的应用成本.....	13
第 2 章 初步体验 Docker.....	15
2.1 在 Windows 中安装 Docker.....	15
2.1.1 通过 Boot2Docker 体验 Docker.....	15
2.1.2 通过 Docker Desktop 体验 Docker.....	22
2.1.3 搭建第一个 Docker 应用: Hello world.....	25
2.2 在 Ubuntu 中安装 Docker.....	27
2.2.1 通过远程仓库安装 Docker.....	27

2.2.2	通过软件包安装 Docker	28
2.2.3	测试安装的结果	29
第 3 章	Docker 基本管理	30
3.1	镜像管理	30
3.1.1	查找镜像	30
3.1.2	下载镜像	31
3.1.3	列出本地镜像	32
3.1.4	删除镜像	32
3.1.5	查看镜像	32
3.1.6	构建镜像	35
3.1.7	镜像标签管理	37
3.2	容器管理	38
3.2.1	创建容器	38
3.2.2	查看容器	40
3.2.3	启动容器	41
3.2.4	停止容器	41
3.2.5	删除容器	42
3.3	网络管理	42
3.3.1	Docker 网络原理	42
3.3.2	网络模式	44
3.3.3	Docker 容器的互连	46
3.3.4	容器与外部网络的互连	47
第 4 章	Kubernetes 初步入门	49
4.1	Kubernetes 技术	49
4.1.1	什么是 Kubernetes	49
4.1.2	Kubernetes 的发展历史	49
4.1.3	为什么使用 Kubernetes	50
4.2	Kubernetes 重要概念	51
4.2.1	Cluster (集群)	51
4.2.2	Master (主控)	51
4.2.3	Node (节点)	52
4.2.4	Pod	53
4.2.5	服务	53
4.2.6	卷	54
4.2.7	命名空间	54

第 5 章 安装 Kubernetes	55
5.1 通过软件包管理工具安装 Kubernetes	55
5.1.1 软件包管理工具	55
5.1.2 节点规划	56
5.1.3 安装前准备	57
5.1.4 etcd 集群配置	57
5.1.5 Master 节点的配置	63
5.1.6 Node 节点的配置	65
5.1.7 配置网络	68
5.2 通过二进制文件安装 Kubernetes	69
5.2.1 安装前准备	69
5.2.2 部署 etcd	73
5.2.3 部署 flannel 网络	76
5.2.4 部署 Master 节点	77
5.2.5 部署 Node 节点	80
5.3 通过源代码安装 Kubernetes	83
5.3.1 本地二进制文件编译	83
5.3.2 Docker 镜像编译	84
第 6 章 Kubernetes 命令行工具	85
6.1 kubectl 的使用方法	85
6.1.1 kubectl 用法概述	85
6.1.2 kubectl 子命令	87
6.1.3 Kubernetes 资源对象类型	89
6.1.4 kubectl 输出格式	90
6.1.5 kubectl 命令举例	90
6.2 kubeadm 的使用方法	93
6.2.1 kubeadm 安装方法	94
6.2.2 kubeadm 基本语法	95
6.2.3 部署 Master 节点	95
6.2.4 部署 Node 节点	97
6.2.5 重置节点	97
第 7 章 运行应用	99
7.1 Deployment	99
7.1.1 什么是 Deployment	99
7.1.2 Deployment 与 ReplicaSet	100

7.1.3	运行 Deployment	100
7.1.4	使用配置文件	107
7.1.5	扩容和缩容	112
7.1.6	故障转移	114
7.1.7	通过标签控制 Pod 的位置	116
7.1.8	删除 Deployment	118
7.1.9	DaemonSet	118
7.2	Job	121
7.2.1	什么是 Job	121
7.2.2	Job 失败处理	123
7.2.3	Job 的并行执行	124
7.2.4	Job 的定时执行	125
第 8 章	通过服务访问应用	127
8.1	服务及其功能	127
8.1.1	服务基本概念	127
8.1.2	服务的功能原理	128
8.2	管理服务	129
8.2.1	创建服务	129
8.2.2	查看服务	132
8.2.3	删除服务	133
8.3	外部网络访问服务	133
8.3.1	kube-proxy 结合 ClusterIP	134
8.3.2	通过 NodePort 访问服务	135
8.3.3	通过负载均衡访问服务	137
8.4	通过 CoreDNS 访问应用	138
8.4.1	CoreDNS 简介	138
8.4.2	安装 CoreDNS	138
第 9 章	存储管理	147
9.1	存储卷	147
9.1.1	什么是存储卷	147
9.1.2	emptyDir 卷	148
9.1.3	hostPath 卷	151
9.1.4	NFS 卷	152
9.1.5	Secret 卷	153
9.1.6	iSCSI 卷	156

9.2	持久化存储卷	157
9.2.1	什么是持久化存储卷	157
9.2.2	持久化存储卷请求	157
9.2.3	持久化存储卷的生命周期	158
9.2.4	持久化存储卷静态绑定	159
9.2.5	持久化存储卷动态绑定	162
9.2.6	回收	167
第 10 章	Kubernetes 软件包管理	170
10.1	Helm	170
10.1.1	Helm 相关概念	170
10.1.2	Tiller	171
10.1.3	Chart	171
10.1.4	Repository	171
10.1.5	Release	171
10.2	安装 Helm	172
10.2.1	安装客户端	172
10.2.2	安装服务器端 Tiller	174
10.3	Chart 文件结构	176
10.4	使用 Helm	177
10.4.1	软件仓库的管理	177
10.4.2	查找 Chart	178
10.4.3	安装 Chart	180
10.4.4	查看已安装 Chart	183
10.4.5	删除 Release	183
第 11 章	Kubernetes 网络管理	185
11.1	Kubernetes 网络基础	185
11.1.1	Kubernetes 网络模型	185
11.1.2	命名空间	186
11.1.3	veth 网络接口	186
11.1.4	netfilter/iptables	187
11.1.5	网桥	187
11.1.6	路由	187
11.2	Kubernetes 网络实现	188
11.2.1	Docker 与 Kubernetes 网络比较	188
11.2.2	容器之间的通信	192

11.2.3	Pod 之间的通信	194
11.2.4	Pod 和服务之间的通信	197
11.3	Flannel	206
11.3.1	Flannel 简介	206
11.3.2	安装 Flannel	207
第 12 章	Kubernetes Dashboard	212
12.1	Kubernetes Dashboard 配置文件	212
12.1.1	Kubernetes 角色控制	212
12.1.2	kubernetes-dashboard.yaml	213
12.2	安装 Kubernetes Dashboard	218
12.2.1	官方安装方法	219
12.2.2	自定义安装方法	219
12.3	Dashboard 使用方法	222
12.3.1	Dashboard 概况	222
12.3.2	通过 Dashboard 创建资源	224
写在最后	226

第 1 章

◀ 全面认识 Docker ▶

随着计算机硬件、网络技术以及云计算的飞速发展和日益普及，Docker 已经成为当下业内讨论的焦点。Docker 号称要成为云服务的基石，并将互联网升级到下一代。由此可以看出，Docker 必然成为支撑整个互联网服务的主流技术之一。了解和掌握 Docker 是从业于互联网行业的必备条件之一。本章将初步介绍 Docker 的基础知识，使得读者能够从整体上理解 Docker 技术，为后面章节的学习打下基础。

本章主要涉及的知识点有：

- 什么是容器：了解容器技术的发展历史。
- 什么是 Docker：介绍 Docker 的由来，Docker 为互联网服务带来的便利以及 Docker 的具体含义。
- Docker 的架构与组成：主要介绍 Docker 的基本架构、组成部分、应用的存在形式以及内容变更的管理方法等。
- 为什么使用 Docker：主要介绍 Docker 具体应用场景、Docker 能够解决的问题以及应用成本等。

1.1 容器技术

目前，虚拟化技术已经成为一种被大家广泛认可的服务器硬件资源共享方式。容器技术的出现，为虚拟化技术带来了新的生机。容器技术已经引起了业内的广泛关注。通过应用容器技术，可以大大地提升工作效率。本节将对容器的基础知识进行系统地介绍，以便于读者学习后续的内容。

1.1.1 什么是容器

在介绍什么是容器之前，首先需要介绍另外一个概念，那就是虚拟化技术。尽管虚拟化技

术已经有了几十年的发展历史，但是由于受限于之前硬件的发展水平，例如，CPU 的性能、计算机的内存容量、硬盘的存储容量以及网卡的速率等，虚拟化技术并没有普及开来。但是，最近几年以来，计算机的各种硬件资源都得到了长足的发展。在这种情况下，某台计算机的计算资源可能会得不到充分地运用，从而造成资源浪费。于是，虚拟化技术也就逐渐普及开来。

简单地讲，所谓虚拟化是将计算机的各种硬件资源，例如 CPU、内存、磁盘以及网络等，都看作是一种资源池，系统管理员可以将这些资源池进行重新分配，提供给其他的虚拟计算机使用。对于管理员来说，底层物理硬件完全是透明的，即完全不用考虑不同的物理架构，在需要各种硬件资源的时候，只要从这个资源池中划出一部分即可。

从上面的描述可以得知，虚拟化技术至少给计算机行业带来了两个巨大的改变，其一就是解决了当前高性能计算机硬件的产能过剩问题，其二是可以把老旧的计算机硬件重新组合起来，作为一个整体的资源来使用。

目前，市场上面主流的虚拟化产品有 Linux 平台上面的 KVM、Xen、VMWare 以及 VirtualBox 等，运行在 Windows 平台上面的虚拟化产品主要有 Hyper V、VMWare 以及 VirtualBox 等。对于这些产品来说，其支持的宿主操作系统是非常广泛的，可以包括 Linux、OpenBSD、FreeBSD 以及各种 Windows 版本等。

在传统的虚拟化技术中，虚拟化系统会虚拟出一套完整的硬件基础设施，包括 CPU、内存、显卡、磁盘以及主板等。因此，所有的虚拟机之间是相互隔离的，每个虚拟机都不会受到其他虚拟机的影响，如同多台物理计算机一样。

尽管传统的虚拟化技术通过虚拟出一套完整的计算机硬件，实现了各个虚拟机之间的完全隔离，从而给用户带来了极大的灵活性，并降低了硬件成本。但是越来越多的用户发现，这种技术方案实际上同时也给自己制造了许多麻烦。例如，在这种环境中，每个虚拟机实例都需要运行客户端操作系统的完整副本以及其中包含的大量应用程序。从实际运行的角度来说，由此产生的沉重负载将会影响其工作效率及性能表现。

容器的诞生为虚拟化技术带来了革命性的变化。它既拥有虚拟化技术的灵活性，又避免了传统的虚拟化技术的上述缺点。

所谓容器，是一种轻量级的操作系统级虚拟化，可以让用户在一个资源隔离的进程中运行应用及其依赖项。运行应用程序所必需的组件都将打包成一个镜像并可以复用。执行镜像时，它运行在一个隔离环境中，并且不会共享宿主机的内存、CPU 以及磁盘，这就保证了容器内的进程不能监控容器外的任何进程。图 1-1 显示了容器的基本架构。

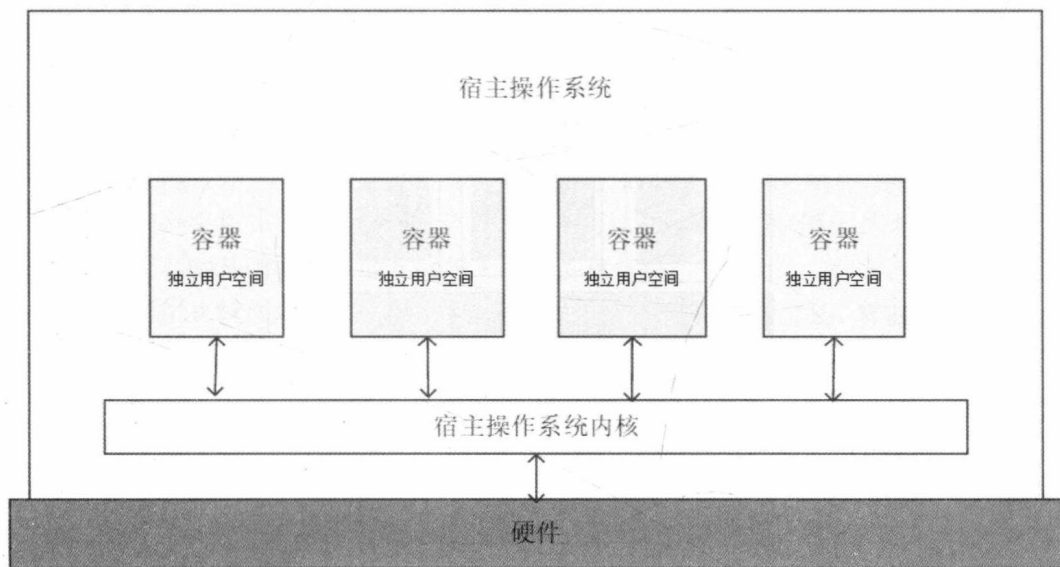


图 1-1 容器架构

1.1.2 容器与虚拟机之间的区别

在 1.1.1 小节中，已经简单介绍了容器与虚拟机的区别。在本小节中，再详细地进行介绍。实际上，与传统的虚拟机相比，容器有着明显的区别。

虚拟机管理系统通常需要为虚拟机虚拟出一套完整的硬件环境，此外，在虚拟机中，通常包含整个操作系统及其应用程序。从这些特点去看，虚拟机与真实的物理计算机非常相似。因为虚拟机包含完整的操作系统，所以虚拟机所占磁盘容量一般都比较大大，一般为几个 GB。若安装的软件比较多，则可以占用几十个 GB，甚至上百个 GB 的磁盘空间。虚拟机的启动相对也比较慢，一般需要数分钟的时间。

容器作为一种轻量级的虚拟化方案，其所占磁盘空间一般为几个 MB。在性能方面，与虚拟机相比，容器表现得更加出色，并且其启动速度非常快，一般只需要几秒的时间。

图 1-2 和图 1-3 显示了虚拟机和容器之间的区别。从图 1-2 可以看出，客户机和宿主机之间有个虚拟机管理器来管理虚拟机。每个虚拟机都有操作系统，应用程序运行在客户机操作系统中。从图 1-3 可以看出，宿主机和容器之间为容器引擎，容器并不包含操作系统，应用程序运行在容器中。

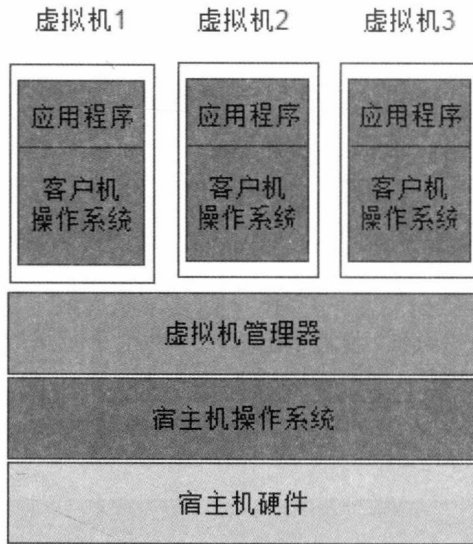


图 1-2 虚拟机

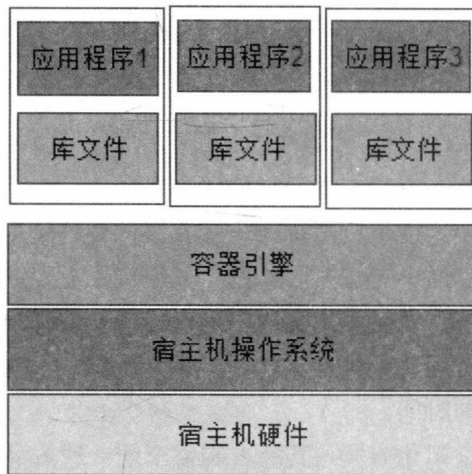


图 1-3 容器

1.1.3 容器究竟解决了什么问题

容器的产生为虚拟化技术带来革命性的变化。然而，许多人并不理解，容器的出现到底解决了什么问题？

在虚拟化系统中，大多数问题都是在应用系统的运行环境改变时才突显出来。例如，开发者在 Windows 操作系统环境中编写应用代码，但是实际使用的运行环境（也可以称为生产环境）却是 Linux 系统。在这种情况下，应用系统的某些功能就极有可能出现问题。也就是说，当开发时的软件环境和实际运行的软件环境不一样的时候，应用系统故障的几率就会大大增加。

Docker 创始人 Solomon Hykes（见图 1-4）曾经说过，“如果测试环境使用 Python 2.7，但是实际使用的运行环境使用 Python 3，那么一些奇怪的事情就会发生。或者我们依赖某个特定版本的 SSL 库，但是却安装了另外一个版本。或者在 Debian 上面运行测试环境，但是实际投入运行时环境是 RedHat，那么任何奇怪的事情都可能发生。”

除了运行环境之外，发生改变的还有可能是网络或者其他方面。例如，测试环境和实际投入运行的环境的网络拓扑可能不同，安全策略和存储也有可能不同。原因是为用户开发的应用系统最终需要在这些基础设施上面实际投入使用并运行。

当用户将应用系统部署在容器中之后，它们的迁移就变得非常容易。容器的初衷也就是将各种应用程序和它们所依赖的运行环境打包成标准的镜像文件，进而发布到不同的平台上去运行。这一点与现实生活中货物的运输非常相似。为了解决各种型号、规格、尺寸的货物在各种运输工具上进行运输的问题，我们发明了集装箱。把货物放进集装箱之后，物流公司只管运输集装箱就可以了，而不用再去关心集装箱里面的货物到底该如何包装、以及提供多大规格的包装箱。他们面对的就是一个个整齐划一的集装箱。而应用容器之后，部署人员面对的不再是具体的应用系统，不用再关心如何为应用系统准备运行环境以及依赖的其他组件，他们面对的就是一个个镜像，主要把镜像部署好就可以了。

从上面的描述可以得知，容器非常适合于在当前的云计算环境快速地迁移和部署应用系统。

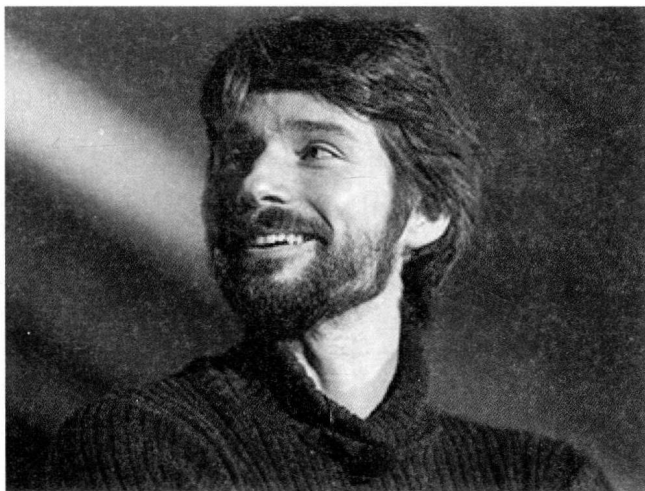


图 1-4 Docker 创始人 Solomon Hykes

1.1.4 容器的优点

容器是在传统的虚拟化基础上发展起来的，因此，容器必然会吸收传统的虚拟化的优点，并克服传统的虚拟化技术的缺点。所以说，容器的优点是非常明显的。下面对容器的优点进行详细介绍。概括地说，容器具有以下优点。