



Kubernetes in Action

中文版

[美] Marko Lukša 著

七牛云 七牛容器云团队 译

Kubernetes
in Action

Kubernetes in Action 中文版

[美] Marko Lukša 著

七牛云 七牛容器云团队 译

Kubernetes in Action

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书主要讲解如何在 Kubernetes 中部署分布式容器应用。本书开始部分概要介绍了 Docker 和 Kubernetes 的由来和发展，然后通过 Kubernetes 中部署一个应用程序，一点点增加功能，逐步加深我们对于 Kubernetes 架构的理解和操作的实践。在本书的后面部分，也可以学习一些高阶的主题，比如监控、调试及伸缩。

Kubernetes 是希腊文，意思是“舵手”，带领我们安全地到达未知水域。Kubernetes 这样的容器编排系统，会帮助我们妥善地管理分布式应用的部署结构和线上流量，高效地组织容器和服务。Kubernetes 作为数据中心操作系统，在设计软件系统时，能够尽量降低在底层网络和硬件设施上的负担。

Original English Language edition published by Manning Publications, USA. Copyright © 2017 by Manning Publications. Simplified Chinese-language edition copyright © 2019 by Publishing House of Electronics Industry. All rights reserved.

本书简体中文版专有出版权由 Manning Publications 授予电子工业出版社。未经许可，不得以任何方式复制或抄袭本书的任何部分。专有出版权受法律保护。

版权贸易合同登记号 图字：01-2018-5008

图书在版编目 (CIP) 数据

Kubernetes in Action 中文版 / (美) 马尔科·卢克沙著；七牛容器云团队译. —北京：电子工业出版社，2019.1

ISBN 978-7-121-34995-9

I. ①K… II. ①马… ②七… 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(2018)第206265号

责任编辑：张春雨

印 刷：北京京师印务有限公司

装 订：北京京师印务有限公司

出版发行：电子工业出版社

北京市海淀区万寿路173信箱 邮编：100036

开 本：787×980 1/16 印张：37.5 字数：888千字

版 次：2019年1月第1版

印 次：2019年1月第1次印刷

定 价：148.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010) 88254888, 88258888。

质量投诉请发邮件至 zltz@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819, faq@phei.com.cn。

推荐序

2013年，Docker 问世，由于其简练易用的使用范式，极大降低了容器技术的使用门槛，引爆了容器技术，一场轰轰烈烈的由容器带来的新计算革命开始了。

2014年，预见到了容器带来的革命性变化，七牛内部成立了一个新项目——QCOS，全称为 Qiniu Cloud Operating System（也可以理解为 Qiniu Cluster Operating System），目标是实现一个数据中心操作系统，让开发人员使用数据中心如同使用一台机器一样容易。在当时，Kubernetes 项目也刚刚开始，还在规格设计阶段。我们通读了 Kubernetes 的设计，决定自己干。这是一个非常疯狂的想法。促使我们决定自己干的原因有两点：一是我们存储也自己干，而且干成了，所以计算自己干，也不无成功的可能；二是 Kubernetes 刚开始，一切如果跟随 Kubernetes，那么我们做起事情来肯定束手束脚，没办法按照商业公司的推进速度来推进。

但是做着做着我们就发现，计算不像存储，计算问题是一个非常开放性的问题。而以容器为核心的计算系统，其复杂性也不同于以虚拟机（VM）为代表的计算系统，因为虚拟机（VM）是以虚拟一台机器为边界的，其问题域同样比较闭合。但是容器计算就是要打破机器的边界，让计算力在数据中心内自由调度。这个问题域涉及面非常广，除了常规的计算力调度、负载均衡、部署升级、服务发现、日志与监控，这些东西都要以全新的方式来解决。

实际上给数据中心做一个操作系统并不是什么新想法。20世纪80年代中期，

贝尔实验室就开启了一个名为 Plan9 的操作系统项目，目的就是做一个数据中心操作系统。参与这个操作系统的名单都大名鼎鼎：Rob Pike、Ken Thompson，等等。你没看错，就是今天创建了 Go 语言的那帮人。他们在 Plan9 项目解散后被 Google 抢了过去，换了一个思路继续做数据中心操作系统——从面向数据中心的语言开始：Go 语言就这样诞生了。而随着 Go 语言的流行，Docker、Kubernetes 接连诞生，继续续写着数据中心操作系统的梦想。

Kubernetes 诞生之初，虽然嘴里衔着 Google Borg 系统的金钥匙，但是同期竞争的项目还是比较多的，除了七牛自己内部发起的 QCOS 外，比较知名的还有 Docker Swam 和 Mesos。但是到了 2016 年，这场竞争越来越趋于 Kubernetes 一统天下。七牛内部 QCOS 项目也放弃了自研，将方向转向了 Kubernetes 阵营。

QCOS 项目对七牛有着特殊的意义，它是七牛业务多元化的开始。在此之前，七牛秉承专注做好一件事情，着眼于对象存储一个点，从单点切入，把单点做到极致的思路，获得了极佳的口碑，大量的移动互联网应用都选择了七牛作为它们的图片和视频托管的云服务提供商。选择做 QCOS，实际上是我们在打第二个根据地时，选择了一条极其艰难的道路。

今天我们 QCOS 团队（内部已更名为 KIRK 团队）发起了 *Kubernetes in Action* 一书的翻译，他们邀请我给译本作序，我脑子里不由自主想起了这段历史。选择基于 Kubernetes，是我们从商业上来说的务实选择，但是它并不代表放弃自主研发，只是把梦想暂时封存在心里。中国古话说，师夷长技以制夷，别人的好东西我们是要学习的，学好了我们才能完成从模仿到超越的过程。

Kubernetes 的背后，是一场新计算革命，是真正的云计算 2.0，我们期待更多有想法的开发者能够学习 Kubernetes，能够加入这场计算革命。也欢迎大家加入七牛。是为序。

许式伟 七牛云 CEO

译者序

早在 2011 年创业初期，七牛就决定使用 GoLang 作为主要开发语言，那时距离 GoLang 1.0 的正式发布还有将近一年的时间。当时我们就断定，在分布式时代，GoLang 这种语言必定会大放异彩。今天，七牛的绝大多数线上服务都是用 GoLang 实现的，Golang 帮助我们以最高的效率实现应用，快速响应客户需求。现在，我们也很幸运地看到当前最热门的开源项目如 Docker、Kubernetes 等，也都是基于 GoLang 来实现的。

就像早期拥抱 GoLang 一样，七牛也是 Docker 和 Kubernetes 技术的坚定拥抱者和践行者。早在 2014 年，我们就基于 Docker 自研了一套容器集群管理系统，用于图片、音视频转码应用实例的资源调度，这套系统在线上运行了好几年，现在正逐渐被 Kubernetes 替代。

Docker 和 Kubernetes 的出现，让我们发现了构建数据中心操作系统（DCOS）的可能性。Docker 的轻量级、Kubernetes 的灵活性和开放性，能让我们以 API 调用的方式来交付计算力（包括 CPU、内存、网络、磁盘等），能让业务应用摆脱传统资源交付方式和既往运维手段的束缚，更快捷地得到部署、升级和管理，业务迭代速度大大加快，资源利用率大幅提高。

早在几年前，我们在七牛就成立了专门的容器云团队，致力于打造更健壮、更易用的容器集群调度管理系统。现在，我们在七牛内部全面推广和应用 Kubernetes，

不仅把无状态服务运行在 Kubernetes 中，也把有状态服务比如数据库运行在 Kubernetes 中，正如使用 GoLang 提高了我们的开发效率一样，使用 Kubernetes 大大提高了我们的部署和运维效率。

在七牛，我们坚定地认为，Kubernetes 会成为下一个 Linux，但是管理的不再是单台机器，而是以 DCOS 的方式来管理整个数据中心。熟练地掌握和使用 Kubernetes，将成为每个前后端工程师的必备技能，Kubernetes 将成为发布前后端服务的标准途径。

这本书的翻译，我们集中了七牛容器云团队，以及其他七牛内部热心志愿者的力量，针对翻译的每个术语我们认真推敲，尽最大可能达到“信”“达”“雅”的程度。鉴于水平有限，难免有纰漏，请读者谅解。

希望这本书能带领你进入 Kubernetes 的世界。

参与本书翻译的七牛容器团队成员有：卢兴铭（致谢词、第 1 章）、李雪璨（第 2 章）、路涛（第 3 章）、孙讷敏（第 4 章）、刘岩（第 5 章）、孙毅飞（第 6 章）、林培裕（第 7 章）、周玉壁（第 8 章）、陈凯俊（第 9 章）、杨冠军（第 10 章）、张钦尧（第 11 章）、况永巧（第 12 章）、易弢（第 13 章）、王浩宇（第 14 章）、王雪瑞（第 15 章）、屈啸（第 16 章）、金鑫鑫（第 17 章）、陈忠杰（第 18 章）。由袁晓沛任翻译组组长，由李雪璨、马力、冯义勇负责审校。

袁晓沛 容器计算部技术总监

前 言

在 Red Hat 工作了几年之后，2014 年底，我被分配到一个叫 Cloud Enablement 的新团队。我们的任务是将公司的中间件系列产品引入基于 Kubernetes 开发的 OpenShift 容器平台。当时，Kubernetes 还在初始的 1.0 版本中，甚至还没有正式发布。

我们团队必须尽快了解 Kubernetes 的细节，以便能够充分利用 Kubernetes 的一切，为软件开发设定正确的方向。当遇到问题时，很难判断是我们出错了，还是仅仅是碰到了一个 Kubernetes 的 bug。

从那以后，Kubernetes 有了长足的发展，我对它的理解也有了很大的提升。当我第一次使用它的时候，大多数人甚至从未听说过 Kubernetes。现在，几乎每个软件工程师都知道它，Kubernetes 已经成为在云上和内部数据中心运行应用程序的增长最快和使用最广泛的方式之一。

在使用 Kubernetes 的第一个月，我写了一篇包含两部分的博客文章，介绍如何在 OpenShift/Kubernetes 中运行 JBoss WildFly 应用服务集群。当时，我从未想过一篇简单的博客文章会让曼宁出版社的人联系我，询问我是否愿意写一本关于 Kubernetes 的书。当然，我不能拒绝这样的提议，尽管我确信他们也会联系其他人，甚至最终会选择其他人。

经过一年半的写作和研究，完成了本书，这是一次很棒的经历。写一本关于一项技术的书是比使用更好的了解它的方法。随着我对 Kubernetes 了解的深入，以及

Kubernetes 本身的不断发展，我不断地回到之前写完的章节，添加更多的信息。我是一个完美主义者，所以永远不会对这本书感到绝对满意，但我很高兴听到 MEAP (Manning Early Access Program) 的许多读者觉得它是一本很好的学习 Kubernetes 的指南。

我的目的是让读者了解技术本身，并教会他们如何使用工具有效地在 Kubernetes 集群中开发和部署应用程序。本书的重点不在如何建立和维护一个高可用的 Kubernetes 集群，但本书的最后一部分告诉读者这样一个集群应该包含什么，这能让大家很容易地理解处理这个问题的额外资源。

希望你能享受阅读此书，并且让你学到如何能够充分利用 Kubernetes 系统的强大之处。

致 谢

在我开始写这本书之前，从来就没有想到将来会有这么多人参与进来，将一堆粗略的手稿整理成一本可以发行的书籍，我要感谢他们的帮助。

首先，我要感谢 Erin Twohey，是他指导了我写这本书。还有 Michael Stephens，从我开始写书的第一天直到后来一年半的时间里，他始终给予我信心，并激励着我继续做下去。

要感谢我的两位编辑。第一位编辑 Andrew Warren，是他协助我完成了此书的首章，迈出了艰难的第一步。第二位编辑 Elesha Hyde，庆幸有他接管 Andrew 的工作，和我并肩作战，直到完成本书的全部章节。我知道自己是一个难以沟通合作的人，但是他们表现出了极大的宽容与耐心。

我要感谢 Jeanne Boyarsky，他是我的第一个审稿人，在我写作期间，不断地审阅我的稿子，并给出很多有用的建议。后来又有了 Jeanne 和 Elesha，他们三人让这本书的品质达到了预期的美好。如今我收到了大量外部读者与评论者的好评，这都是他们的功劳。

要感谢我的技术校对 Antonio Magnaghi，以及所有的外部审稿人：Al Krinker、Alessandro Campeis、Alexander Myltsev、Csaba Sari、David DiMaria、Elias Rangel、Erisk Zelenka、Fabrizio Cucci、Jared Dunca、Keith Donaldson、Michael Bright、Paolo Antinori、Peter Perlepes 和 Tiklu Ganguly。在我心情糟糕到想要放弃写作时，

他们的鼓励继续温暖我前行。另一方面，他们建设性的想法帮助我完善了那些因为精力有限而匆匆拼凑的章节。他们还积极指出了那些晦涩难懂的章节，并给出了如何优化的建议。这期间他们也提出了一些很好的问题，让我及时地意识到自己的错误，并完善了手稿的初版。

我还需要感谢一些读者，他们在 Manning MEAP (Manning Early Access Program) 购买了这本书的早期版本，并通过在线论坛讨论或者直接联系我的方式，给出了很多建议，特别是 Vimal Kansal、Panoo Patierno 和 Roland Huß，他们发现了不少内容前后不一致和描述性错误。我还要感谢那些在 Manning 工作并参与过此书出版的所有人。我同样要感谢高中和大学期间的好友 Ales Justin，他引荐我到 Red Hat，结识了 Cloud Enablement 团队的同事。感谢他们，如果不是进入这家公司，进入这个团队，我不可能写这本书。

最后，我要感谢我的妻子与孩子，在过去的 18 个月，我把自己反锁在办公室里，忽略了对他们的陪伴，即便这样，他们表现出了对我的莫大理解与支持。

感谢他们所有人！

关于本书

本书旨在让你能够熟练使用 Kubernetes。它介绍了在 Kubernetes 中有效地开发和运行应用所需的几乎所有概念。

在深入研究 Kubernetes 之前，本书概述了 Docker 等容器技术，包括如何构建容器，以便即使以前没有使用过这些技术的读者也可以使用它们。然后，它会慢慢带你从基本概念到实现原理了解大部分的 Kubernetes 知识。

本书适合谁

本书主要关注应用开发人员，但也从操作的角度概述了应用的管理。它适合任何对在多服务器上运行和管理容器化应用感兴趣的人。

对于希望学习容器技术以及大规模的容器编排的人，无论是初学者还是高级软件工程师都将得到在 Kubernetes 环境中开发、容器化和运行应用所需的专业知识。

阅读本书不需要预先了解容器或 Kubernetes 技术。本书以渐进的方式展开主题，不会使用让非专家开发者难以理解的应用源代码。

读者至少应该具备编程、计算机网络和运行 Linux 基本命令的基础知识，并了解常用的计算机协议，如 HTTP 协议。

本书的组织方式：路线图

本书分为三个部分，涵盖 18 个章节。

第一部分简要地介绍 Docker 和 Kubernetes、如何设置 Kubernetes 集群，以及在集群中运行一个简单的应用。它包括两章：

- 第 1 章解释了什么是 Kubernetes、Kubernetes 的起源，以及它如何帮助解决当今大规模应用管理的问题。
- 第 2 章是关于如何构建容器镜像并在 Kubernetes 集群中运行的实践教程。还解释了如何运行本地单节点 Kubernetes 集群，以及在云上运行适当的多节点集群。

第二部分介绍了在 Kubernetes 中运行应用必须理解的关键概念。本章内容如下：

- 第 3 章介绍了 Kubernetes 的基本构建模块——pod，并解释了如何通过标签组织 pod 和其他 Kubernetes 对象。
- 第 4 章将向你介绍 Kubernetes 如何通过自动重启容器来保持应用程序的健康。还展示了如何正确地运行托管的 pod，水平伸缩它们，使它们能够抵抗集群节点的故障，并在未来或定期运行它们。
- 第 5 章介绍了 pod 如何向运行在集群内外的客户端暴露它们提供的服务，还展示了运行在集群中的 pod 是如何发现和访问集群内外的服务的。
- 第 6 章解释了在同一个 pod 中运行的多个容器如何共享文件，以及如何管理持久化存储并使得 pod 可以访问。
- 第 7 章介绍了如何将配置数据和敏感信息（如凭据）传递给运行在 pod 中的应用。
- 第 8 章描述了应用如何获得正在运行的 Kubernetes 环境的信息，以及如何通过与 Kubernetes 通信来更改集群的状态。
- 第 9 章介绍了 Deployment 的概念，并解释了在 Kubernetes 环境中运行和更新应用的正确方法。
- 第 10 章介绍了一种运行需要稳定的标识和状态的有状态应用的专门方法。

第三部分深入研究了 Kubernetes 集群的内部，介绍了一些额外的概念，并从更高的角度回顾了在前两部分中所学到的所有内容。这是最后一组章节：

- 第 11 章深入 Kubernetes 的底层，解释了组成 Kubernetes 集群的所有组件，以及每个组件的作用。它还解释了 pod 如何通过网络进行通信，以及服务如何跨多个 pod 形成负载平衡。
- 第 12 章解释了如何保护 Kubernetes API 服务器，以及通过扩展集群使用身份验证和授权。
- 第 13 章介绍了 pod 如何访问节点的资源，以及集群管理员如何防止 pod 访问

节点的资源。

- 第 14 章深入了解限制每个应用程序允许使用的计算资源，配置应用的 QoS (Quality of Service) 保证，以及监控各个应用的资源使用情况。还会介绍如何防止用户消耗太多资源。
- 第 15 章讨论了如何通过配置 Kubernetes 来自动伸缩应用运行的副本数，以及在当前集群节点数量不能接受任何新增应用时，如何对集群进行扩容。
- 第 16 章介绍了如何确保 pod 只被调度到特定的节点，或者如何防止它们被调度到其他节点。还介绍了如何确保 pod 被调度在一起，或者如何防止它们调度在一起。
- 第 17 章介绍了如何开发应用程序并部署在集群中。还介绍了如何配置开发和测试工作流来提高开发效率。
- 第 18 章介绍了如何使用自己的自定义对象扩展 Kubernetes，以及其他是如何开发并创建企业级应用平台的。

随着章节的深入，不仅可以了解单个构建 Kubernetes 的模块，还可以逐步增加对使用 `kubectl` 命令行工具的理解。

关于代码

虽然这本书没有包含很多实际的源代码，但是包含了许多 YAML 格式的 Kubernetes 资源清单，以及 shell 命令及其输出。所有这些都是用等宽字体显示的，以便与普通文本相互区分。

shell 命令大部分以粗体文字展示，以便与命令的输出清晰地区分，但为了表示强调有时只有命令的最重要的部分或是命令输出的一部分是粗体的。在大多数情况下，为了适应图书有限的版面空间，命令输出会被重新格式化。此外，由于 Kubernetes 的命令行工具 `kubectl` 不断发展更新，新版本输出的信息可能会比书中显示的更多。如果输出结果不完全一致，请不要感到困惑。

代码清单有时会包括续行标记 (↪) 表示一行文字延续到下一行。代码清单还可能包括注释，这些注释用于解释最重要的部分。

在文本段落包含一些常见的元素，如 Pod、ReplicationController、ReplicaSet、DaemonSet 等，为了避免代码字体的过度复杂，便于阅读，都以常规字体显示。在某些地方，大写字母开头的“Pod”表示 Pod 资源，小写则表示实际运行的容器组。

本书中的所有示例都使用谷歌 Kubernetes 引擎 (Google Kubernetes Engine)，在 Kubernetes 1.8 版本和 Minikube 的本地集群进行了运行测试。完整的源代码和 YAML 清单可以在 <https://github.com/luksa/kubernetes-in-action> 找到，或者从出版商的网站 www.manning.com/books/kubernetes-in-action 中下载。

其他线上资源

可以在下面的网址中找到许多额外的 Kubernetes 信息：

- Kubernetes 官方网站：<https://kubernetes.io>
- 经常发布有用信息的 Kubernetes 博客：<http://blog.kubernetes.io>
- Kubernetes 社区的 Slack 频道：<http://slack.k8s.io>
- Kubernetes 以及 Cloud Native Computing Foundation 的 YouTube 频道：
 - https://www.youtube.com/channel/UCZ2bu0qutTOM0tHYa_jkIwg
 - <https://www.youtube.com/channel/UCvqbFHwN-nwalWPjPUKpvTA>

为了获得对于单个话题更深入的理解或者贡献 Kubernetes 项目，可以查看 Kubernetes 特别兴趣小组（SIGs）：[https://github.com/kubernetes/kubernetes/wiki/Special-Interest-Groups-\(SIGs\)](https://github.com/kubernetes/kubernetes/wiki/Special-Interest-Groups-(SIGs))。

最后，由于 Kubernetes 是一个开源项目，Kubernetes 源码本身包含大量的信息。可以访问 Kubernetes 和相关的仓库：<https://github.com/kubernetes/>。

读者服务

轻松注册成为博文视点社区用户（www.broadview.com.cn），扫码直达本书页面。

- **提交勘误**：您对书中内容的修改意见可在 [提交勘误](#) 处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **交流互动**：在页面下方 [读者评论](#) 处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：<http://www.broadview.com.cn/34995>



关于作者



马尔科·卢克沙 (Marko Lukša) 是一位具有 20 年开发 Web 应用、ERP 系统、框架以及中间件软件经验的专业软件工程师。1985 年，六岁的他在父亲给他买的一台二手 ZX 光谱计算机 (ZX Spectrum computer) 上开始了编程的第一步。小学时，他是全国 Logo 语言编程大赛的冠军，他在参加编程夏令营时学习了 Pascal 编程。从那时起，他开始使用各种编程语言开发软件。

他从高中在万维网的早期就开始搭建动态网站。在斯洛文尼亚卢布尔雅那大学 (University of Ljubljana) 学习计算机科学期间，他在当地一家公司为医疗和电信行业开发软件。最终，他成为了 Red Hat 的员工，最初开发了谷歌应用程序引擎 (Google App Engine) API 的开源实现，该 API 底层使用了 Red Hat 的 JBoss 中间件产品。他还参与了 CDI/Weld、Infinispan/JBoss DataGrid 等项目。

自 2014 年底以来，他一直是 Red Hat Cloud Enablement 团队的一员，在该团队中，他的职责包括了解 Kubernetes 和相关技术的最新发展趋势，保证公司的软件中间件充分利用 Kubernetes 和 OpenShift 的特性。

关于封面插画

本书封面上的人物肖像名为“Divan 的成员”（Member of the Divan），Divan 是指土耳其国务委员会或理事机构。图片来自 1802 年 1 月 1 日出版的由伦敦老邦德街的威廉·米勒（William Miller）创作的一套奥斯曼帝国服装设计作品集。作品集的标题页丢失了，至今没有找到。作品集的目录用英文和法文标出了这些人物，每幅插图上都有创作的两位艺术家的名字，艺术家们应该会为自己的作品出现在 200 年之后的一本电脑编程书的封面感到惊讶吧。

作品集由 Manning 的一位编辑从位于曼哈顿西 26 街“车库”的一个古董跳蚤市场买的。卖家是一位居住在土耳其安卡拉的美国人，交易发生在他收拾摊子的时候。Manning 的编辑当时身上没有足够的现金，信用卡和支票都被卖家礼貌地拒绝了。随着那天晚上卖主飞回安卡拉，情况变得令人绝望。后来，通过一种类似握手的旧式口头协议，交易出现了转机。卖主提议用电汇的方式把钱汇给他，然后编辑拿着一张包含银行信息的纸和一叠照片走了出来。第二天我们把钱转了过去，我们仍然对这个陌生人的信任心存感激，印象深刻，让人想起很久以前发生的事情。Manning 以呈现两个世纪前丰富多彩的地区生活的书籍封面来颂扬计算机产业的独创性和首创性，并让像这本画册这样的古籍和藏品中的绘画作品重现生机。