



Kubernetes Advanced Practical

Kubernetes 进阶实战

马永亮 著

马哥教育CEO马哥（马永亮）撰写，实用性和专业性毋庸置疑

涵盖Kubernetes架构、部署、核心组件、扩缩容、存储与网络策略、安全、系统扩展等话题
Kubernetes主流知识点全覆盖，渐进式讲解，手把手示范，大量实操案例，随时动手验证



机械工业出版社
China Machine Press

云计算与虚拟化技术丛书



Kubernetes Advanced Practical

Kubernetes 进阶实战

马永亮 著



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

Kubernetes 进阶实战 / 马永亮著. —北京: 机械工业出版社, 2019.1
(云计算与虚拟化技术丛书)

ISBN 978-7-111-61445-6

I. K… II. 马… III. Linux 操作系统—程序设计 IV. TP316.85

中国版本图书馆 CIP 数据核字 (2018) 第 275406 号

Kubernetes 进阶实战

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 张志铭

责任校对: 殷虹

印刷: 北京市荣盛彩色印刷有限公司

版次: 2019 年 1 月第 1 版第 1 次印刷

开本: 186mm × 240mm 1/16

印张: 28.75

书号: ISBN 978-7-111-61445-6

定价: 109.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88379426 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzit@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

HZBOOKS | 华章IT | Information Technology



为什么要写这本书

作为置身于 IT 技术领域多年的实践者和教育者，我们一直盼望着行业迎来这样一个时刻：异构的 IT 基础设施环境所造成的开发和部署系统应用纷繁复杂的局面终于迎来了终结者，开发人员无须再考虑复杂多样的运行环境下软件程序的移植问题，运维人员不用再手动解决运行环境中组件间的依赖关系等，从而让各自的核心职责都回归到开发和保证系统稳定运行本身。终于，以 Docker 为首的容器技术为此带来了基础保障，并在容器编排技术的支撑下尘埃落定，甚至连 IT 管理者心心念念多年的 DevOps 文化运动也借此找到了易于落地的实现方案。于是，系统运行割据多年的局面终于将走向天下一统。

尽管距 Kubernetes 1.0 的发布不过三四年的光景，但其如今的影响力在 IT 技术领域完全算得上空前绝后，目前，一众大小公司都在使用或正筹划使用这一 IT 技术发展史上可能最为成功的开源项目。Linux 软件基金会的常务董事 Jim Zemlin 在 Google Cloud Next 17 大会上曾表示，Kubernetes 是“云时代的 Linux”。的确，Kubernetes 应该是开源世界有史以来迭代最快的项目，而且在几乎所有需要采用容器技术的场景里成为占统治地位的解决方案，其发展速度恐怕也仅有 Linux 内核项目可堪匹敌。2018 年 3 月，Kubernetes 成为 CNCF 旗下“毕业”的第一个项目，并荣获 2018 年 OSCON 最具影响力奖项。

目前，Kubernetes 保持着每年发布四个重要版本的节奏，版本的每次更新都会引入数个新特性。这种快速迭代的机制在为用户不断带来惊喜的同时，也给他们在学习和使用上造成了一些困扰：相关领域的可参考书籍仍不丰富，互联网上可以得到的众多文档并非源于同一个版本，以及厘清脉络拼凑成完整的知识框架所需的时间成本较大。因此，我在课程以及直接或间接参与生产或测试环境的交付之余便萌生了撰写一本 Kubernetes 入门、进阶与实战的书籍的想法，将自己学习和使用的经验总结、沉淀并分享给更多有此需求的技术同行，帮助大家快速找到入门路径，降低时间成本，并迅速投入测试和生产之用。

的确，在写作过程中，Kubernetes 这种快速迭代的机制，以及每每引入的新特性，在小惊喜之余带给笔者更多的却是真真切切的梦魇般的恐惧感：在一年多的写作时间里，许多章节几易其稿，却也依然无法确保能够涵盖即将成为核心功能的特性，于是沮丧感几度如影随形，直到自我安慰着“基础的核心特性基本不会发生大的变动，只要能帮助读者弄清楚 Kubernetes 系统的基础架构及核心工作逻辑就算工夫没有白费”之后方才释然。于是便有了这本力图尽量多地包罗 Kubernetes 系统目前主流特性及实践路径的入门和进阶之书、工具之书。

本书特色

本书致力于帮助容器编排技术的初级和中级用户循序渐进地理解与使用 Kubernetes 系统，因此本书的编写充分考虑到初学者进入新知识领域时的茫然，采用由浅入深、提纲挈领、再由点到面的方式讲解每一个知识细节。对于每个知识点，不仅介绍了其概念和用法，还分析了为什么要有这个概念，实现的方式是什么，背后的逻辑为何，等等，使读者不仅能知其然，还能知其所以然。

本书不仅要带领读者入门，更是一本可以随时动手加以验证的实践手册，而且对于部分重要的内容还会专门一步步地给出具体的实操案例，帮助读者在实践中升华对概念的理解。本书几乎涵盖了应用 Kubernetes 系统的所有主流知识点，它甚至可以作为计划考取 CKA 认证的读者的配套参考图书。

读者对象

- 云计算工程师
- 运维工程师
- 系统开发工程师
- 程序架构师
- 计划考取 CKA 认证的人员
- 其他对容器编排感兴趣的人员

如何阅读本书

阅读使用本书之前，读者需要具备 Docker 容器技术的基础使用能力。本书逻辑上共分为五大部分，15 章。

第一部分（第 1 ~ 2 章），介绍 Kubernetes 系统的基础概念及其基本应用。

第 1 章介绍容器编排系统出现的背景，以及 Kubernetes 系统的功能、特性、核心概念、

系统组件及应用模型。

第 2 章讲解 Kubernetes 的核心对象，以及直接使用命令管理资源对象的快速入门技巧。

第二部分（第 3 ~ 6 章），介绍核心资源类型及其应用。

第 3 章介绍资源管理模型、陈述式与声明式资源管理接口，并通过命令对比说明两种操作方式的不同之处。

第 4 章介绍 Pod 资源的常用配置、生命周期、存储状态和就绪状态检测，以及计算资源的需求及限制等。

第 5 章介绍 Pod 控制器资源类型，重点讲解了控制无状态应用的 ReplicaSet、Deployment、DaemonSet 控制器，并介绍了 Job 和 CronJob 控制器。

第 6 章介绍 Service 和 Ingress 资源类型，涵盖 Service 类型、功用及其实现，以及 Ingress 控制器、Ingress 资源的种类及其实现，并通过案例详细说明了 Ingress 资源的具体使用方式。

第三部分（第 7 ~ 9 章），介绍存储卷及 StatefulSet 控制器。

第 7 章主要介绍存储卷类型及常见存储卷的使用方式、PV 和 PVC 出现的原因及应用，以及存储类资源的应用和存储卷的动态供给。

第 8 章介绍使用一等资源类型 ConfigMap 和 Secret 为容器应用提供配置及敏感信息的方式。

第 9 章主要介绍有状态应用的 Pod 控制器资源 StatefulSet，包括基础应用、动态扩缩容及更新机制等。

第四部分（第 10 ~ 11 章），介绍安全相关的话题，主要涉及认证、授权、准入控制、网络模型与网络策略。

第 10 章重点讲解认证方式、Service Account 和 TLS 认证、授权插件类型及 RBAC，并于章节的最后介绍 LimitRanger、ResourceQuota 和 PodSecurityPolicy 三种类型的准入控制器及相关的资源类型。

第 11 章主要介绍网络插件基础及 flannel 的三种后端实现与应用、借助 Canal 插件实现网络策略的方式，以及 Calico 网络插件的基础使用。

第五部分（第 12 ~ 15 章），介绍 Kubernetes 系统的高级话题。

第 12 章介绍 Pod 资源的调度策略及高级调度方式的应用，包括节点亲和、Pod 资源亲和以及基于污点和容忍度的调度。

第 13 章介绍系统资源的扩展方式，包括自定义资源类型、自定义资源对象、自定义 API 及控制器、Master 节点的高可用、基于 Kubernetes 的 PaaS 系统等话题。

第 14 章介绍资源指标、自定义指标、监控系统及 HPA 控制器的应用。

第 15 章介绍简化应用管理的工具 Helm，并基于 Helm 介绍如何为 Kubernetes 系统提供统一的日志收集与管理工具栈 EFK。

有一定 Kubernetes 使用经验的读者可以挑选感兴趣的章节阅读。而对于初学者，建议从基础部分逐章阅读，但构建在 Kubernetes 系统之上的应用多数都要求读者能熟练使用相关领域的知识和技能，如果对某些内容的理解比较困难，那么可能是由于相关知识欠缺，建议读者通过其他资料补充学习相关知识后再阅读本书。编撰本书的主要意图是为初学者提供一个循序渐进的实操手册，不过，任何读者也都可以将它作为一本案头的工具书随时进行查阅。

排版约定

本书中所有的命令都附带了或长格式或短格式的命令提示符，以便读者区分文中正常使用的“#”和“\$”，命令提示格式为“~]#”或“~]\$”，较长的命令使用了“\”为续行符，且命令及其输出使用了有别于正文的字体。

更多内容和附带代码

本书相关的配置清单等都放置于 <https://github.com/ikubernetes/> 的相关仓库中，在实践中需要时可直接克隆至本地实验环境中使用。

勘误和支持

尽管进行授课及技术写作已十数年，但动笔著书尚属首次，考虑到排版印刷后的表述无可更改，整个写作过程战战兢兢、如履薄冰。进行每一个关键话题的表述之前都查阅了大量资料，并且反复斟酌，既期望能够将知识点清晰、准确地加以描述，也试图避免因自己的理解偏差而误导读者。尽管如此，由于笔者水平有限，加之编写时间仓促，书中难免存在不妥之处，恳请读者批评指正。如果读者有更多的宝贵意见，请通过邮箱 mage@mageedu.com 联系我，期待能够得到你们的真挚反馈，在技术之路上互勉共进。另外，本书的勘误将会发布在笔者的博客（<http://www.ilinux.io>）或本书专用的 GitHub 主页（<https://github.com/ikubernetes>）上，欢迎读者朋友们关注并留言讨论。

参考资料

对于具有不同知识基础和结构的读者来说，仅凭一本书的内容根本不足以获取所需的全部信息，大家还可以通过以下信息获取关于 Kubernetes 系统的更多资料，本书在写作期间也从这些参考资料中获得了很大的帮助。

- Kubernetes Documentation 和 Kubernetes API Reference，这是提供 Kubernetes 领域相关知识最全面、最深入和最准确的参考材料。
- 《Kubernetes in Action》，本书的谋篇布局及写作理念与此书不谋而合，因此笔者在本书中对许多概念的理解和验证也以此书为素材，并且在写作之时有多处概念的描述也借鉴了此书的内容。
- Red Hat Openshift Documentation，Red Hat 公司的产品文档规范、权威、细致且条理清晰，是不可多得的参考材料。
- The New Stack 的技术文章及调研报告是了解 Kubernetes 系统技术细节和行业应用现状及趋势的不可多得优秀资源。
- Bitnami 及 Heptio 站点上的博客文章提供了深入了解和学习 Kubernetes 系统某个特定技术细节的可靠资料。

另外，本书还大量借鉴了通过搜索引擎所获取到的不少技术文章和参考文档，在这里一并向这些书籍和文章的作者表示深深的谢意！

致谢

感谢 Kubernetes 社区创造性的劳动成果和辛苦付出，我们因此有了学习和使用如此优秀的开源系统的可能性，这也是本书得以编撰的基石。

感谢我的同事们在我写作期间给予的支持和理解，他们的努力让我拥有了得以放心写作的时间和精力。感谢提供了相关行业信息并促使我下定决心开始编写此书的张常安和张士杰先生，本书的写作过程中也得到了他们支援的宝贵素材。

感谢参加了我的课程（马哥教育）的学员朋友们，大家的学习热情及工作中源源不断反馈而来的信息与需求在不同程度上帮助我一直保持着对技术的追求和热忱，教学相长在此得到了充分的体现。

感谢机械工业出版社华章公司高婧雅女士对本书写作的悉心指导，以及对我本人的包容和理解。

最后要特别感谢我的家人，为写作这本书，我牺牲了很多陪伴他们的时间，也正因为他们在生活中的关怀和鼓励才使我能够踏踏实实地完成本书内容的编写。

马永亮

2018 年 10 月

目 录 *Contents*

前言

第 1 章 Kubernetes 系统基础 1

1.1 容器技术概述 1

1.1.1 容器技术的功用 2

1.1.2 容器简史 3

1.1.3 Docker 的功能限制 4

1.2 Kubernetes 概述 4

1.2.1 Kubernetes 简史 4

1.2.2 Kubernetes 特性 5

1.2.3 Kubernetes 概念和术语 6

1.3 Kubernetes 集群组件 10

1.3.1 Master 组件 10

1.3.2 Node 组件 12

1.3.3 核心附件 13

1.4 Kubernetes 网络模型基础 13

1.4.1 网络模型概述 13

1.4.2 集群上的网络通信 15

1.5 本章小结 16

第 2 章 Kubernetes 快速入门 17

2.1 Kubernetes 的核心对象 17

2.1.1 Pod 资源对象 18

2.1.2 Controller 19

2.1.3 Service 20

2.1.4 部署应用程序的主体过程 21

2.2 部署 Kubernetes 集群 22

2.2.1 kubeadm 部署工具 22

2.2.2 集群运行模式 24

2.2.3 准备用于实践操作的集群环境 25

2.2.4 获取集群环境相关的信息 26

2.3 kubectl 使用基础与示例 26

2.4 命令式容器应用编排 29

2.4.1 部署应用 (Pod) 30

2.4.2 探查 Pod 及应用详情 33

2.4.3 部署 Service 对象 36

2.4.4 扩容和缩容 38

2.4.5 修改及删除对象 40

2.5 本章小结 41

第 3 章 资源管理基础 42

3.1 资源对象及 API 群组 42

3.1.1 Kubernetes 的资源对象 43

3.1.2	资源及其在 API 中的组织形式	46	4.3.1	标签概述	75
3.1.3	访问 Kubernetes REST API	48	4.3.2	管理资源标签	77
3.2	对象类资源格式	49	4.3.3	标签选择器	78
3.2.1	资源配置清单	50	4.3.4	Pod 节点选择器 nodeSelector	79
3.2.2	metadata 嵌套字段	51	4.4	资源注解	80
3.2.3	spec 和 status 字段	52	4.4.1	查看资源注解	81
3.2.4	资源配置清单格式文档	53	4.4.2	管理资源注解	82
3.2.5	资源对象管理方式	54	4.5	Pod 对象的生命周期	82
3.3	kubectl 命令与资源管理	56	4.5.1	Pod 的相位	82
3.3.1	资源管理操作概述	56	4.5.2	Pod 的创建过程	83
3.3.2	kubectl 的基本用法	57	4.5.3	Pod 生命周期中的重要行为	84
3.4	管理名称空间资源	59	4.5.4	容器的重启策略	87
3.4.1	查看名称空间及其资源对象	60	4.5.5	Pod 的终止过程	87
3.4.2	管理 Namespace 资源	61	4.6	Pod 存活性探测	88
3.5	Pod 资源的基础管理操作	61	4.6.1	设置 exec 探针	89
3.5.1	陈述式对象配置管理方式	62	4.6.2	设置 HTTP 探针	90
3.5.2	声明式对象配置管理方式	64	4.6.3	设置 TCP 探针	92
3.6	本章小结	65	4.6.4	存活性探测行为属性	93
第 4 章	管理 Pod 资源对象	66	4.7	Pod 就绪性探测	94
4.1	容器与 Pod 资源对象	66	4.8	资源需求及资源限制	96
4.2	管理 Pod 对象的容器	68	4.8.1	资源需求	96
4.2.1	镜像及其获取策略	69	4.8.2	资源限制	98
4.2.2	暴露端口	70	4.8.3	容器的可见资源	99
4.2.3	自定义运行的容器化应用	71	4.8.4	Pod 的服务质量类别	100
4.2.4	环境变量	72	4.9	本章小结	101
4.2.5	共享节点的网络名称空间	73	第 5 章	Pod 控制器	103
4.2.6	设置 Pod 对象的安全上下文	74	5.1	关于 Pod 控制器	103
4.3	标签与标签选择器	75	5.1.1	Pod 控制器概述	104
			5.1.2	控制器与 Pod 对象	105
			5.1.3	Pod 模板资源	106

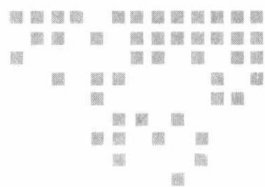
5.2	ReplicaSet 控制器	106	6.1.1	Service 资源概述	136
5.2.1	ReplicaSet 概述	107	6.1.2	虚拟 IP 和服务代理	138
5.2.2	创建 ReplicaSet	108	6.2	Service 资源的基础应用	140
5.2.3	ReplicaSet 管控下的 Pod 对象	109	6.2.1	创建 Service 资源	140
5.2.4	更新 ReplicaSet 控制器	111	6.2.2	向 Service 对象请求服务	141
5.2.5	删除 ReplicaSet 控制器资源	114	6.2.3	Service 会话粘性	142
5.3	Deployment 控制器	114	6.3	服务发现	143
5.3.1	创建 Deployment	115	6.3.1	服务发现概述	143
5.3.2	更新策略	116	6.3.2	服务发现方式: 环境变量	145
5.3.3	升级 Deployment	119	6.3.3	ClusterDNS 和服务发现	146
5.3.4	金丝雀发布	121	6.3.4	服务发现方式: DNS	146
5.3.5	回滚 Deployment 控制器下的应用发布	123	6.4	服务暴露	147
5.3.6	扩容和缩容	123	6.4.1	Service 类型	147
5.4	DaemonSet 控制器	124	6.4.2	NodePort 类型的 Service 资源	149
5.4.1	创建 DaemonSet 资源对象	124	6.4.3	LoadBalancer 类型的 Service 资源	150
5.4.2	更新 DaemonSet 对象	126	6.4.4	ExternalName Service	151
5.5	Job 控制器	127	6.5	Headless 类型的 Service 资源	152
5.5.1	创建 Job 对象	128	6.5.1	创建 Headless Service 资源	153
5.5.2	并行式 Job	129	6.5.2	Pod 资源发现	153
5.5.3	Job 扩容	130	6.6	Ingress 资源	154
5.5.4	删除 Job	130	6.6.1	Ingress 和 Ingress Controller	154
5.6	CronJob 控制器	131	6.6.2	创建 Ingress 资源	155
5.6.1	创建 CronJob 对象	131	6.6.3	Ingress 资源类型	157
5.6.2	CronJob 的控制机制	132	6.6.4	部署 Ingress 控制器 (Nginx)	159
5.7	ReplicationController	133	6.7	案例: 使用 Ingress 发布 tomcat	161
5.8	Pod 中断预算	133	6.7.1	准备名称空间	161
5.9	本章小结	134	6.7.2	部署 tomcat 实例	162
			6.7.3	创建 Service 资源	163
			6.7.4	创建 Ingress 资源	164
第 6 章	Service 和 Ingress	136			
6.1	Service 资源及其实现模型	136			

6.7.5 配置 TLS Ingress 资源	165	8.2 通过命令行参数配置容器应用	204
6.8 本章小结	168	8.3 利用环境变量配置容器应用	206
第 7 章 存储卷与数据持久化	169	8.4 应用程序配置管理及 ConfigMap 资源	208
7.1 存储卷概述	169	8.4.1 创建 ConfigMap 对象	209
7.1.1 Kubernetes 支持的存储卷类型	170	8.4.2 向 Pod 环境变量传递 ConfigMap 对象键值数据	212
7.1.2 存储卷的使用方式	171	8.4.3 ConfigMap 存储卷	215
7.2 临时存储卷	172	8.4.4 容器应用重载新配置	219
7.2.1 emptyDir 存储卷	172	8.4.5 使用 ConfigMap 资源的注意事项	220
7.2.2 gitRepo 存储卷	175	8.5 Secret 资源	221
7.3 节点存储卷 hostPath	176	8.5.1 Secret 概述	221
7.4 网络存储卷	178	8.5.2 创建 Secret 资源	222
7.4.1 NFS 存储卷	178	8.5.3 Secret 存储卷	224
7.4.2 RBD 存储卷	180	8.5.4 imagePullSecret 资源对象	225
7.4.3 GlusterFS 存储卷	182	8.6 本章小结	226
7.4.4 Cinder 存储卷	183	第 9 章 StatefulSet 控制器	227
7.5 持久存储卷	184	9.1 StatefulSet 概述	227
7.5.1 创建 PV	186	9.1.1 Stateful 应用和 Stateless 应用	227
7.5.2 创建 PVC	188	9.1.2 StatefulSet 控制器概述	228
7.5.3 在 Pod 中使用 PVC	190	9.1.3 StatefulSet 的特性	230
7.5.4 存储类	191	9.2 StatefulSet 基础应用	231
7.5.5 PV 和 PVC 的生命周期	194	9.2.1 创建 StatefulSet 对象	232
7.6 downwardAPI 存储卷	196	9.2.2 Pod 资源标识符及存储卷	234
7.6.1 环境变量式元数据注入	197	9.3 StatefulSet 资源扩缩容	237
7.6.2 存储卷式元数据注入	199	9.4 StatefulSet 资源升级	238
7.7 本章小结	201	9.4.1 滚动更新	238
第 8 章 配置容器应用: ConfigMap 和 Secret	202		
8.1 容器化应用配置方式	202		

9.4.2	暂存更新操作	239	10.4.5	面向用户的内建 ClusterRole	273	
9.4.3	金丝雀部署	240	10.4.6	其他的内建 ClusterRole 和 ClusterRoleBinding	274	
9.4.4	分段更新	241	10.5	Kubernetes Dashboard	275	
9.4.5	其他话题	241	10.5.1	部署 HTTPS 通信的 Dashboard	275	
9.5	案例: etcd 集群	242	10.5.2	配置 token 认证	277	
9.5.1	创建 Service 资源	242	10.5.3	配置 kubeconfig 认证	277	
9.5.2	etcd StatefulSet	243	10.6	准入控制器与应用示例	279	
9.6	本章小结	247	10.6.1	LimitRange 资源与 LimitRanger 准入控制器	279	
第 10 章 认证、授权与准入控制			248	10.6.2	ResourceQuota 资源与准入 控制器	281
10.1	访问控制概述	248	10.6.3	PodSecurityPolicy	283	
10.1.1	用户账户与用户组	249	10.7	本章小结	288	
10.1.2	认证、授权与准入控制 基础	250	第 11 章 网络模型与网络策略			
10.2	服务账户管理与应用	253	11.1	Kubernetes 网络模型及 CNI 插件	289	
10.2.1	Service Account 自动化	253	11.1.1	Docker 容器的网络模型	289	
10.2.2	创建服务账户	255	11.1.2	Kubernetes 网络模型	291	
10.2.3	调用 imagePullSecret 资源 对象	256	11.1.3	Pod 网络的实现方式	293	
10.3	X.509 数字证书认证	256	11.1.4	CNI 插件及其常见的实现	295	
10.3.1	Kubernetes 中的 SSL/TLS 认证	257	11.2	flannel 网络插件	297	
10.3.2	客户端配置文件 kubeconfig	259	11.2.1	flannel 的配置参数	297	
10.3.3	TLS bootstrapping 机制	262	11.2.2	VxLAN 后端和 direct routing	298	
10.4	基于角色的访问控制: RBAC	263	11.2.3	host-gw 后端	301	
10.4.1	RBAC 授权插件	264	11.3	网络策略	302	
10.4.2	Role 和 RoleBinding	266	11.3.1	网络策略概述	302	
10.4.3	ClusterRole 和 ClusterRoleBin- ding	269				
10.4.4	聚合型 ClusterRole	271				

11.3.2	部署 Canal 提供网络策略 功能	303	12.4.3	Pod 对象的容忍度	346
11.3.3	配置网络策略	305	12.4.4	问题节点标识	347
11.3.4	管控入站流量	306	12.5	Pod 优先级和抢占式调度	347
11.3.5	管控出站流量	308	12.6	本章小结	348
11.3.6	隔离名称空间	310	第 13 章 Kubernetes 系统扩展 349		
11.3.7	网络策略应用案例	311	13.1	自定义资源类型 (CRD)	349
11.4	Calico 网络插件	315	13.1.1	创建 CRD 对象	350
11.4.1	Calico 工作特性	316	13.1.2	自定义资源格式验证	351
11.4.2	Calico 系统架构	318	13.1.3	子资源	353
11.4.3	Calico 部署要点	320	13.1.4	使用资源类别	355
11.4.4	部署 Calico 提供网络服务和 网络策略	321	13.1.5	多版本支持	355
11.4.5	客户端工具 calicoctl	324	13.1.6	自定义控制器基础	356
11.5	本章小结	325	13.2	自定义 API Server	359
第 12 章 Pod 资源调度 326			13.2.1	自定义 API Server 概述	359
12.1	Kubernetes 调度器概述	326	13.2.2	APIService 对象	360
12.1.1	常用的预选策略	327	13.3	Kubernetes 集群高可用	361
12.1.2	常用的优选函数	330	13.3.1	etcd 高可用	362
12.2	节点亲和调度	332	13.3.2	Controller Manager 和 Scheduler 高可用	363
12.2.1	节点硬亲和性	332	13.4	Kubernetes 的部署模式	364
12.2.2	节点软亲和性	335	13.4.1	关键组件	365
12.3	Pod 资源亲和调度	337	13.4.2	常见的部署模式	366
12.3.1	位置拓扑	338	13.5	容器时代的 DevOps 概述	369
12.3.2	Pod 硬亲和调度	338	13.5.1	容器: DevOps 协作的 基础	369
12.3.3	Pod 软亲和调度	341	13.5.2	泛型端到端容器应用程序 生命周期 workflow	370
12.3.4	Pod 反亲和调度	342	13.5.3	基于 Kubernetes 的 DevOps	371
12.4	污点和容忍度	343	13.6	本章小结	372
12.4.1	定义污点和容忍度	344			
12.4.2	管理节点的污点	345			

第 14 章 资源指标及 HPA 控制器	373
14.1 资源监控及资源指标	373
14.1.1 资源监控及 Heapster	374
14.1.2 新一代监控架构	376
14.2 资源指标及其应用	378
14.2.1 部署 metrics-server	378
14.2.2 kubectl top 命令	380
14.3 自定义指标与 Prometheus	381
14.3.1 Prometheus 概述	382
14.3.2 部署 Prometheus 监控系统	384
14.3.3 自定义指标适配器 k8s- prometheus-adapter	388
14.4 自动弹性缩放	390
14.4.1 HPA 概述	390
14.4.2 HPA (v1) 控制器	391
14.4.3 HPA (v2) 控制器	393
14.5 本章小结	397
第 15 章 Helm 程序包管理器	398
15.1 Helm 基础	398
15.1.1 Helm 的核心术语	399
15.1.2 Helm 架构	400
15.1.3 安装 Helm Client	400
15.1.4 安装 Tiller server	401
15.1.5 Helm 快速入门	402
15.2 Helm Charts	405
15.2.1 Charts 文件组织结构	405
15.2.2 Chart.yaml 文件组织 格式	406
15.2.3 Charts 中的依赖关系	407
15.2.4 模板和值	408
15.2.5 其他需要说明的话题	409
15.2.6 自定义 Charts	410
15.3 Helm 实践：部署 EFK 日志 管理系统	415
15.3.1 ElasticSearch 集群	416
15.3.2 日志采集代理 fluentd	421
15.3.3 可视化组件 Kibana	422
15.4 本章小结	424
附录 A 部署 Kubernetes 集群	425
附录 B 部署 GlusterFS 及 Heketi	437



Kubernetes 系统基础

近十几年来，IT 领域新技术、新概念层出不穷，例如 DevOps、微服务（Microservice）、容器（Container）、云计算（Cloud Computing）和区块链（Blockchain）等，直有“乱花渐欲迷人眼”之势。另外，出于业务的需要，IT 应用模型也在不断地变革，例如，开发模式从瀑布式（Waterfall）到敏捷（Agile）再到精益（Lean），甚至是与 QA 和 Operations 融合的 DevOps，应用程序架构从单体（monolithic）模型到分层模型再到微服务，部署及打包方式从面向物理机到虚拟机再到容器，应用程序的基础架构从自建机房到托管再到云计算，等等，这些变革使得 IT 技术应用的效率大大提升，同时却以更低的成本交付更高质量的产品。

尤其是以 Docker 为代表的容器技术的出现，终结了 DevOps 中交付和部署环节因环境、配置及程序本身的不同而造成的动辄几种甚至十几种部署配置的困境，将它们统一在容器镜像（image）之上。如今，越来越多的企业或组织开始选择以镜像文件作为交付载体。容器镜像之内直接包含了应用程序及其依赖的系统环境、库、基础程序等，从而能够在容器引擎上直接运行。于是，IT 运维工程师（operator）无须关注开发应用程序的编程语言、环境配置等，甚至连业务逻辑本身也不必过多关注，而只需要掌握容器管理的单一工具链即可。

部署的复杂度虽然降低了，但以容器格式运行的应用程序间的协同却成了一个新的亟待解决的问题，这种需求在微服务架构中表现得尤为明显。结果，以 Kubernetes 为代表的容器编排系统应需而生。

1.1 容器技术概述

容器是一种轻量级、可移植、自包含的软件打包技术，它使得应用程序可以在几乎任