容器即服务

从零构建企业级容器集群

林帆 / 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

容器即服务

从零构建企业级容器集群

林帆 / 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书介绍了容器即服务的发展过程和主要技术，重点阐述当下主流的 SwarmKit、Kubernetes、Mesos 和 Rancher 开源容器集群方案，并探讨了容器技术在网络、存储、监控、日志等方面的运用场景和基础知识，以及该领域在近年来的一些新的发展方向。

本书适合一线架构师、开发者、运维人员以及技术管理者进行阅读。

图书在版编目（CIP）数据

容器即服务：从零构建企业级容器集群 / 林帆著. —北京：电子工业出版社，2018.4
ISBN 978-7-121-33276-0

Ⅰ. ①容… Ⅱ. ①林… 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（2017）第 309385 号

策划编辑：张春雨
责任编辑：牛 勇
印 刷：三河市鑫金马印装有限公司
装 订：三河市鑫金马印装有限公司
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：30 字数：578 千字
版 次：2018 年 4 月第 1 版
印 次：2018 年 4 月第 1 次印刷
定 价：99.00 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 zlts@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。

序

在这个日新月异的时代，每一位站在浪尖上的技术匠人，都不得不加紧步伐，追赶不断变化的趋势。与此相应的一个现象是，当一本技术类的书籍刚刚面市，它所讲述的内容就已经开始迅速过时。

这本书从 2016 年初开始筹备，由于种种原因拖沓了近两年终于完稿。在这段时间里：

- SwarmKit 诞生了，原先的 Swarm 技术栈光荣退役。
- Kubernetes 的版本从 1.0 一路更迭到 1.10，增加了无数新特性。
- Mesos 推出 Unified Container，曾经一度被看好的 Docker 集成器风光不再。
- Rancher 发布 2.0 版本，完全颠覆先前的用户体验设计。
- Docker 自家的 LinuxKit、阿里的 Pouch 这些底层开源技术在不断演进。

书还没写完，最初准备的材料有一大半都已经作废。

先前笔者写作《CoreOS 实践之路》一书时，同样是一边增加新章节，一边关注书里涉及软件的变化，对已有章节进行三番五次的补充修正，到完成时，许多地方都被大段大段地重写了。此次的《容器即服务：从零构建企业级容器集群》因为涉及方面较多，加上写作时间跨度较大，以至于维护其中的内容变化更加困难，经过数次截稿日的跳票，才费劲地将书中示例涉及的大部分软件更新到 2017 年中下旬的版本。

不过，本书写作的初衷并非在于介绍最新的工具。对于学习一门成熟的工具，最直接的方式莫过于阅读它的文档。但面对一个领域中众多的知识，入门者最容易迷失的地方在于缺少一条主线。本书一方面希望为容器集群及其周边的领域勾勒一幅入门的蓝图，另一方面则是点出一些在文档中没有讲清但实际很容易迷惑用户的大坑小洼，对于细节和扩展的内容则以参考链接的形式提供。

如今的容器技术正在处于百花齐放的时期，当我们讨论到容器，很多时候已不是单纯地在说某种内核虚拟化技术，而是在谈服务集群、任务调度，以及 Cloud Native 和微服务。与此同时，容器平台相关的应用场景也越来越丰富，大规模容器化部署的运用逐渐从少数大型企业发展到许多中型和创业企业里。作为现代产品发布模式的重塑者，容器技术以及它所提倡的基础设施即代码交付思想，对每位一线架构师、开发者、运维人员乃至技术管

理者的工作带来的影响，都不容小觑。本书截取了一些具有当下时代特征的技术剪影，提供给读者品味。

在编写内容时，本书尽量以通用的容器技术作为背景，而非限定于特定的容器产品（比如 Docker）。但在一些具体的例子方面，均采用了当前最主流的 Docker 容器作为讲解示例。

由于写作周期较长，加之作者个人的经验所限，书中难免存在一些阐述不当和错误的地方。本书的勘误表发布在博文视点官方网站 http://www.broadview.com.cn/33276，恳请各位读者通过此页面提交勘误或发邮件到 linfan.china@gmail.com 予以指正。

最后，感谢在过去两年中不断督促和鼓励我完成写作的张春雨以及负责了整本书编辑的吴倩雪，没有你们的努力，这本书肯定无法按时出版。感谢将我养育成材的父母以及我的爱人杨斌清，你们默默的支持使我得以静下心来认真地完成这部作品。同样感谢每一位开源代码的贡献者，正是开源推动了技术的革命，才使“旧时王谢堂前燕”，如今“飞入寻常百姓家”。我亦是一名普通的技术匠人，且当少一些浮躁，多一些沉淀，借以此书自勉。

林　帆

2017 年 12 月 25 日

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/33276*

目　录

第 1 部分　基础概念

第 1 章　容器集群综述 …… 2

1.1　虚拟化与容器 …… 2

1.1.1　计算资源虚拟化 …… 2

1.1.2　容器技术的本质 …… 4

1.1.3　基于容器的软件交付 …… 13

1.2　容器集群与分布式服务 …… 16

1.2.1　微服务架构 …… 16

1.2.2　容器集群生态圈 …… 18

1.3　容器即服务 …… 26

1.3.1　从基础设施到平台 …… 26

1.3.2　数据中心操作系统 …… 29

1.4　本章小结 …… 31

第 2 部分　解决方案

第 2 章　SwarmKit 集群解决方案 …… 35

2.1　开源容器集群方案 …… 35

2.1.1　容器社区的“四朵金花” …… 35

2.1.2　经典 Swarm、SwarmKit 和 Swarm Mode …… 36

2.2　使用 SwarmKit …… 37

2.2.1　SwarmKit 综述 …… 37

2.2.2　创建 SwarmKit 集群 …… 40

2.2.3 在 SwarmKit 集群上运行服务……43
2.2.4 SwarmKit 集群的其他功能……45
2.3 Docker Swarm Mode……45
2.3.1 Swarm Mode 综述……45
2.3.2 集群的创建与销毁……46
2.3.3 节点管理……48
2.3.4 服务管理……51
2.3.5 服务编排……56
2.3.6 应用栈的管理……63
2.3.7 外置配置和密文管理……66
2.4 Swarm Mode 的图形界面……69
2.4.1 Swarm Mode UI 现状……69
2.4.2 Portainer……71
2.5 本章小结……74
第 3 章 Kubernetes 集群解决方案……75
3.1 Kubernetes 集群概述……75
3.1.1 Kubernetes 项目的起源……75
3.1.2 Kubernetes 的结构……76
3.1.3 基本概念……78
3.2 部署 Kubernetes 集群……82
3.2.1 使用 Minikube……82
3.2.2 使用 kubeadm……83
3.2.3 理解 Kubernetes 集群的部署过程……87
3.2.4 验证集群可用性……96
3.3 使用 Kubernetes……98
3.3.1 通过 Kubernetes 部署服务……98
3.3.2 服务的在线更新和回滚……103
3.3.3 单次任务、定时任务和全局服务……109
3.3.4 持久化存储……113

3.3.5 配置存储……116
3.3.6 管理有状态的服务……122
3.3.7 健康检查……126
3.3.8 提供对外服务……127
3.3.9 多租户隔离和配额……131
3.3.10 集群的节点管理……135
3.4 Kubernetes 包管理工具 Helm……137
3.4.1 Helm 简介……137
3.4.2 使用 Helm 管理服务……137
3.4.3 自定义 Chart……142
3.4.4 Chart 仓库……146
3.5 本章小结……147
第 4 章 Mesos 集群解决方案……148
4.1 Mesos 和 DC/OS 概述……148
4.1.1 Mesos 项目的起源……148
4.1.2 Mesos 的结构……149
4.1.3 Mesos 的内部构成……151
4.1.4 DC/OS 数据中心操作系统……152
4.2 部署 Mesos 集群……153
4.2.1 部署 ZooKeeper……153
4.2.2 部署 Mesos……157
4.2.3 启动 Master 节点……158
4.2.4 添加 Agent 节点……161
4.2.5 Mesos 服务的启动参数……164
4.3 使用 Marathon 管理服务……170
4.3.1 部署 Marathon……170
4.3.2 添加一个应用……172
4.3.3 使用 DC/OS 命令行工具……177
4.3.4 使用 Docker 容器……177

4.3.5　使用 Unified Container ····179
4.3.6　持久化卷存储 ····182
4.3.7　Marathon-LB 负载均衡 ····184
4.3.8　Mesos-DNS 域名服务 ····188
4.3.9　服务依赖和编组 ····191
4.3.10　应用升级 ····194
4.3.11　调度约束 ····199
4.3.12　健康检查 ····201
4.4　使用 Chronos ····203
4.4.1　部署 Chronos ····203
4.4.2　定时表达式 ····204
4.4.3　创建定时任务 ····205
4.4.4　定时任务的依赖 ····208
4.5　更多的 Mesos 服务框架 ····209
4.5.1　Mesos 服务框架的本质 ····209
4.5.2　编写自己的 Mesos 服务框架 ····211
4.5.3　其他常见服务框架 ····216
4.6　DC/OS ····218
4.6.1　DC/OS 简介 ····218
4.6.2　部署 DC/OS ····219
4.6.3　DC/OS 的操作 ····228
4.6.4　DC/OS 命令行工具 ····230
4.6.5　DC/OS 的应用仓库 ····231
4.7　本章小结 ····234
第 5 章　Rancher 集群解决方案 ····235
5.1　Rancher 集群概述 ····235
5.1.1　Rancher 项目的起源 ····235
5.1.2　Rancher 的结构 ····236
5.1.3　相关概念 ····237

5.2 构建 Rancher 集群……239
5.2.1 部署 Server 节点……239
5.2.2 Server 节点的高可用部署方式……240
5.2.3 添加 Agent 节点……241
5.3 Rancher 的服务管理……243
5.3.1 使用 Rancher Web UI 创建服务……243
5.3.2 从容器……245
5.3.3 特殊类型的服务……247
5.3.4 使用应用商店……251
5.3.5 服务编排……252
5.3.6 服务的升级和回滚……254
5.4 Rancher 使用进阶……256
5.4.1 Rancher 的标签……256
5.4.2 调度选项……257
5.4.3 服务健康检查……258
5.4.4 Rancher 的元数据服务……260
5.4.5 Rancher 的 DNS 服务……262
5.4.6 使用私有镜像仓库……263
5.4.7 Rancher 的 Secret 服务……264
5.4.8 在应用商店添加自定义应用……265
5.5 Rancher 的命令行工具……268
5.5.1 配置 Rancher 命令行工具……268
5.5.2 命令工具的基本使用……270
5.5.3 通过命令行进行服务编排……273
5.5.4 通过命令行进行服务升级……273
5.6 使用 Rancher 安装 Kubernetes……274
5.6.1 Rancher 的环境管理……274
5.6.2 在 Rancher 中添加 Kubernetes 环境……276
5.6.3 在 Rancher 中使用 Kubernetes……279

5.7 本章小结……282

第 3 部分 技术周边

第 6 章 容器集群的网络和存储……284

6.1 容器网络……284

6.1.1 容器网络标准……284

6.1.2 本地网络……288

6.1.3 跨节点网络……293

6.1.4 使用 Docker 内置的 Overlay 类型网络……300

6.1.5 构建基于 Flannel 的覆盖网络……301

6.1.6 构建基于 Calico 的 BGP 路由网络……306

6.2 容器存储……310

6.2.1 容器实例和镜像的存储……310

6.2.2 容器卷的存储……312

6.2.3 容器卷存储标准……316

6.2.4 基于 NFS 的卷存储……317

6.2.5 基于 Ceph 的卷存储……320

6.2.6 使用公有云存储……330

6.3 本章小结……332

第 7 章 容器服务的基础设施……333

7.1 集群性能监控……333

7.1.1 常见的开源性能监控方案……333

7.1.2 基于 TICK Stack 的性能监控……335

7.1.3 TICK Stack 的部署和使用……336

7.1.4 基于 Prometheus 的性能监控……341

7.1.5 Prometheus 的部署……343

7.1.6 Prometheus 的使用……353

7.2 集群日志管理……361

7.2.1 常见的开源日志管理方案……361

7.2.2 基于 Elastic Stack 的日志管理……363
7.2.3 基于 Fluentd 的日志管理……372
7.3 服务发现……377
7.3.1 常见的服务发现方案……377
7.3.2 Etcd……379
7.3.3 Consul……390
7.4 镜像仓库……398
7.4.1 容器镜像仓库概述……398
7.4.2 Registry……399
7.4.3 Harbor……405
7.5 本章小结……412
第 8 章 容器技术新风向……413
8.1 安全的集群操作系统：Container Linux……413
8.1.1 Container Linux 概述……413
8.1.2 Container Linux 的部署……416
8.1.3 Container Linux 的使用……418
8.2 基于容器的操作系统：RancherOS……419
8.2.1 RancherOS 概述……419
8.2.2 部署 RancherOS……421
8.2.3 RancherOS 的使用……422
8.2.4 使用 ros 工具管理系统……424
8.3 容器式的虚拟机：Hyper……429
8.3.1 Hyper 概述……429
8.3.2 部署 Hyper……430
8.3.3 Hyper 的使用……431
8.4 虚拟机式的容器：LXD……434
8.4.1 LXD 概述……434
8.4.2 LXD 的安装和使用……435
8.4.3 服务热迁移……440

8.5　容器与虚拟机的统一：Rkt……442
8.5.1　Rkt 概述……442
8.5.2　Rkt 的安装和使用……444
8.6　企业级定制容器：Pouch……450
8.6.1　Pouch 概述……450
8.6.2　Pouch 的开源生态……453
8.6.3　体验 Pouch……455
8.7　微内核操作系统：Unikernel……458
8.7.1　Unikernel 概述……458
8.7.2　Unikernel 的发展……460
8.7.3　体验 Unikernel……462
8.8　本章小结……465

1

第 1 部分　基础概念

容器即服务的核心在于容器技术，它的流行与近年来大规模、分布式、无状态服务的发展趋势密切相关。容器归根到底只是一系列内核特性的组合，以及基于此的许多实践理念，这些看似简单的概念改变了软件交付的面貌，在此之上又变化出许多新颖的套路。万变不离其宗，唯有谙熟其中门道，才能在学习的过程中拨云见日、融会贯通。

↘ 容器集群综述

第1章　容器集群综述

容器集群并不是许多容器的简单堆积，而是以容器技术为基础的包含部署、调度、网络、存储等方面的有机整体。在容器集群之上可以构建更高层的服务系统，如动态伸缩的任务队列服务、企业级的业务平台、分布式的数据计算服务等。作为底层计算资源和上层业务服务的黏合剂，以按需使用的方式提供基于容器的云端运行环境的平台，形成了一种具有独特价值的服务，这类场景被称为容器即服务。

这一章将从虚拟化和容器说起，介绍容器集群以及容器即服务平台的一些运用场景。

1.1　虚拟化与容器

1.1.1　计算资源虚拟化

虚拟化在现代计算机领域的使用相当广泛，它通过将真实的硬件抽象为软件可控的逻辑单元，使得昂贵的硬件资源能够按需、按量分配，以达到减少浪费、实现硬件利用率最大化的目的。计算资源的虚拟化是虚拟化领域里比较重要的一个分支，这里的计算资源主要指的是 CPU、内存、硬盘等与计算机运算直接相关的硬件资源。在很长的一段时间里，计算资源的虚拟化始终是各类虚拟机技术争夺的热土。从最初 IBM 和 Sun 等公司主导的早期 CPU 虚拟化实现，到 VMware、Xen、KVM、QEMU 等虚拟化或半虚拟化技术的成熟，就经历了 40 余年的发展过程。

与此同时，另一种虚拟化技术的分支也在缓慢发展。这类虚拟化技术不依赖与硬件相关的特性，而是在系统内核的层次之上，将进程运行的上下文环境加以限制和隔离。最初它们并不被视为虚拟化方法，比如在 Unix 系统中引入的 chroot 工具仅仅是将特定进程的文件上下文锁定在特定目录中，制造出在同个系统里模拟多个隔离的系统目录的效果。随后，在 FreeBSD 4.0 中出现的 Jails 和前 SWsoft 公司（现已更名为 Parallels）开发的基于 Windows

系统的 Virtuozzo（睿拓）等技术在 chroot 的基础上增加了进程空间和网络空间的隔离，更好地实现了进程之间互不干扰地共享硬件资源的目的。这种虚拟化方式就是最早的容器技术雏形。

随后不久，IBM、Sun 和惠普等老牌虚拟机和操作系统公司也纷纷进入不依赖特定 CPU 和硬件支持的虚拟化技术阵营，分别在自家的操作系统里推出相应的产品，例如运行在 Solaris 系统的 Zones、运行在 IBM AIX 小型机系统的 WPARs 以及运行在惠普服务器系统 HP-UX 的 SRP Containers 等。在这段时期里，特别值得一提的是在开源 GNU/Linux 系统上实现的操作系统级虚拟化服务：Linux-VServer。

开源社区的介入使得这类虚拟化技术迅速发展，并被应用到更多的领域中。然而作为社区产品，由于参与人员众多、早期目标定位不明确，Linux-VServer 的配置细节很复杂，加上文档十分混乱，因此当时只有对 Linux 内核有一定了解的用户才能驾驭它。这种状态一直持续到 2005 年，这一年，曾经设计了 Virtuozzo 的 SWsoft 公司开始在 Linux 系统上开发一款全新的开源虚拟化产品：OpenVZ[①]。这款采用了 GNU/GPL 协议开源的软件很好地改善了 Linux 系统上进行虚拟化隔离的使用体验，并为 SWsoft 公司带来了可观的收入。同一时期还诞生了一个对虚拟化技术影响颇远的概念：VPS（Virtual Private Server，虚拟专用服务器）。VPS 技术是指将一台服务器分割成多个逻辑上的虚拟专享服务器。每个 VPS 都可分配独立公网 IP 地址、独立操作系统、独立硬盘空间、独立内存和 CPU 资源。这种虚拟主机间的隔离服务为用户和应用程序模拟出“独占使用计算资源”的体验。OpenVZ 理所当然地成为了当时虚拟化技术的代表之一，与 Xen、KVM 并列成为 VPS 提供商首选的虚拟化实施方案。

此时的 Linux 系统级虚拟化技术已经逐渐成熟，然而对代码质量颇为严苛的 Linux 内核团队并没有采纳 Linux-Vserver 或 OpenVZ 提交的内核补丁，而是在 Linux 2.6 的内核中重新设计了 Namespace 和 CGroup 等功能，实现了更加灵活的可组合式虚拟化能力。随后在 2008 年，Linux 社区就出现了基于内核隔离能力设计的 LXC（Linux Containers）虚拟化项目，它充分利用 Linux 内核的 Namespace 隔离能力和 CGroup（Control Groups，控制组）控制能力实现了操作系统内核层面上的虚拟化，不再需要修改内核代码，大大降低了使用该项技术的门槛。此后的几年里，又相继出现了 linux-utils 和 systemd-nspawn[②]等 Linux 内核虚拟化工具。Linux 系统开始在内核虚拟化技术的演进过程中发挥越来越重要的作用，如图 1-1 所示。

① https://wiki.openvz.org/History

② https://www.freedesktop.org/software/systemd/man/systemd-nspawn.html