Broadview®
www.broadview.com.cn

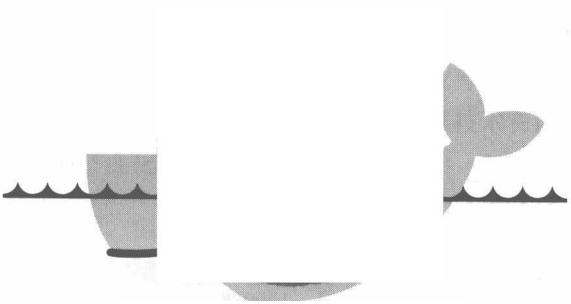
Docker
微服务架构实战

蒋彪◎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

Docker
微服务架构实战

蒋彪◎著



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

内 容 简 介

微服务与 Docker 是近年来分布式大规模服务架构中两个主流的技术趋势，本书主要介绍中小型企业在架构落地过程中柔性地切入微服务和 Docker 虚拟化的各种方法。

书中主要介绍了微服务架构的各种技术选型、微服务拆分的各项原则、传统应用向微服务架构过渡的方法论、Docker 技术原理、Docker 跨主机通信选型、Docker 与 DevOps 的整合方法等要点，同时简单介绍了利用 Rancher搭建 Docker 容器云平台的快速解决方案，非常适合云计算从业人员阅读、学习。

图书在版编目（CIP）数据

Docker 微服务架构实战 /蒋彪著. —北京：电子工业出版社，2018.11
ISBN 978-7-121-35033-7

Ⅰ. ①D… Ⅱ. ①蒋… 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 207873 号

策划编辑：孙奇俏
责任编辑：张春雨
印　　刷：北京京科印刷有限公司
装　　订：北京京科印刷有限公司
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：17.75　字数：365 千字
版　　次：2018 年 11 月第 1 版
印　　次：2018 年 11 月第 1 次印刷
定　　价：69.00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

推荐序

在这个技术日新月异的时代，每一位技术从业者都不得不加紧步伐，追赶不断变化的技术趋势。

从单体架构到分布式服务架构，到 SOA 架构，再到微服务架构以及最新的服务网格，架构演进的步伐在业务爆发式增长的背景下从未停滞过。微服务、服务网格和容器技术俨然已经成为了当下最热门的技术话题。如今的微服务、服务网格和容器技术已经不再局限于理论，而是成为了势不可当的技术潮流。

微服务的核心思想就是将整个业务系统拆分为相对独立的业务模块，并强调各个微服务都可以独立开发、独立测试、独立部署、独立运行、独立运维。这种松耦合的灵活特性正是目前业界所有人所盼望的，因此微服务架构在较短的时间内便在很多大型互联网企业被落地实践，成为解决复杂应用的一把利器。比微服务更进一步的服务网格更是将传统服务治理框架中的服务注册、服务发现、服务熔断、服务降级、服务限流这些和业务无关的内容抽离出来，下沉为底层服务，让研发的重心更集中在业务逻辑的实现上。而容器技术在此过程中发挥了“推波助澜”的作用，加速了微服务的产业化步伐。

微服务看似简单、容易理解，但其实际落地过程却困难重重。很多已经实施或者计划实施微服务的公司在服务划分、服务测试、服务运维、服务安全以及人员组织结构调整等方面缺乏经验，为此付出了很大的代价也没能实际感受到微服务架构的灵活性。

很多时候，我们都非常希望在落地微服务的过程中有一盏明灯来为我们指引方向。如果你和你所在的团队正在经历或即将经历这样的架构演进过程，那么相信本书会是你的那盏指路明灯，带领你快速步入微服务架构时代。

纵览全书，内容循序渐进，概念清晰明了，由浅入深，从易到难，技术描述有点有面，清晰地为读者呈现了一幅包括微服务、服务网格、容器技术原理及落地实践在内的全景图。更难能可贵的是，全书采用了原理结合实战的行文思路，融合了作者在中小型企业推广实施微服务

的亲身经验，是一本理论架构完整，又包含典型实践案例的好书。

无论你是开发工程师、测试工程师、运维工程师、架构师，还是技术经理，相信你都会通过本书接触到微服务和容器领域最热门的技术以及最清晰的工程实践流程，快速建立微服务生态圈的全局知识体系。

茹炳晟

eBay 中国研发中心技术主管

2018 年 9 月于上海

前言

Docker 与微服务是什么

到底什么是架构师？一名架构师的职责应该是什么？

很多年前，我作为一名架构初学者，便了解了架构师要做的事情。从方法论上来看，就是“识别利益相关者”“梳理各个层次的利益传递”和“根据利益线索设计有伸缩性的架构”。

同样，以架构师的视角来看，到底什么是微服务呢？作为架构师，我所看到的微服务，是在团队人数井喷，产品迭代周期太快，系统中的技术负债过剩导致不能将鸡蛋放到同一个摇摇欲坠的篮子，投资人对产品功能特性提出夸张、多变要求的大背景下，在投资人、经理层、研发人员、测试人员、运维人员等各个利益相关者的逼迫下，被迫地将系统从并发、扩展、易维护等维度进行的拆分。

按照微服务的思想将系统拆分之后，我们发现，拆分得越厉害，系统的信息熵就越大，因此有了系统的拆分，就要有拆分之后的治理。

比如，如何保证拆分之后应用的注册发现？如何保证拆分之后服务的熔断限流监控？如何保证服务之间底层通信的可靠性和序列化？

为此，各种各样的服务治理以及分布式服务协同框架便出现了。

那么是不是有了微服务的拆分再加上微服务的治理就足够了呢？对于开发人员而言也许足够了，但是对于运维人员而言，远远不够。拆分使得系统的健壮性大大提高，但是也使系统的发布和运维异常复杂。试想一下，如何在一个分布式的环境下令几万个微服务快速启停？如何实现几万个微服务的管理、编排与自动扩容？

为了解决这些问题，Docker 以及与 Docker 关联的各种容器编排技术便出现了。

因此，如果我们站在终点回望起点，就会知道为什么会有微服务，又为什么会有 Docker，

乃至为什么会有今天的 Service Mesh。我们可以说，这些技术的产生一脉相承，它们的根本目的都是解决系统复杂度井喷之后的系统架构问题；从学术上来说，这些技术都是在和复杂分布式架构中的信息熵进行对抗；从工程上来说，这些技术是当今时代背景催生出的必然产物。

为什么写此书

作为一名技术人员，比起知道技术的用法，更重要的是要知道如何使这项技术切合公司现有的技术栈、业务线，如何解决公司的痛点，如何在各个部门的协同下将技术实际落地。

一项技术再好，再优秀，在实际落地中也要“削足适履”，要根据公司的产品线走，跟着公司的技术现状确定如何应用。毕竟绝大多数公司的第一要务是生存与盈利，能够专职供养一支底层研究技术团队的公司少之又少，更何况是对 Docker 这样的“底之又底”的技术栈。

对于一名架构师而言，当你决定引入 Docker 的时候，你要知道，你在公司内引入的不是一个简单的技术框架、一个中间件、一个自动化工具，而是一种思想、一个云平台、一个新的开发习惯。

对于 Docker 的引入，从入门到落地，需要架构师、运维工程师、网络工程师、研发人员、测试工程师，乃至公司高层的全面支持和协作。把 Docker 用到极致，就是在建设自己的云平台和 PaaS，对于这种体量的研究，若没有一个全面的规划和路线图是无法完成的。

同时，对于 Docker 的引入者而言，需要学习包括 Docker 底层的原理、Docker 的网络模型、Docker 的容器隔离、Docker 下的自动发布与自动运维，乃至常被很多人挂在嘴边的 Docker 容器编排、Docker 与微服务、Docker 与 DevOps 在内的方方面面的知识。

上述所有的知识，若能够集中在一个人身上，实在是难得。毕竟，在 IT 行业，每个人都有自己的知识短板。

甚至我可以负责任地说，在很多公司中，若想找到一支能够搞定底层到上层方方面面建设的专业团队，都是很困难的。

举个例子，我见过很多 Docker 实际应用的场景，在一些中型场景中，通常直接用 bridge 端口来转发通信，为什么会这样？因为很多公司虽然不缺少优秀的程序员，但是缺少网络方面的架构师，不懂也不敢尝试复杂的网络模式。再比如，我亲眼见到有的项目在落地的时候，采用 Docker 挂载卷轴的方式发布代码，为什么？-因为这些公司缺少专门的人来维护 CI/CD 平台。

正是因为有这些困难，所以笔者将自己在中小型企业中推广微服务和 Docker 的亲身经历和

亲眼所见总结出来，从技术选型和架构切入的层面进行梳理，希望通过这本书将这些好的经验传递给需要的读者，帮助更多的人。

本书内容

本书共包括三个部分，共计 13 章，每部分及每章的简要内容如下。

第一部分 Docker 与微服务基础

第 1 章 微服务架构概述

本章介绍了微服务具体是什么，为什么要使用微服务，微服务的架构设计原则，以及从单体到微服务的演进过程。

第 2 章 微服务中的技术选型

本章主要介绍了微服务架构中的各种技术选型，包括服务治理、服务网关、服务发现、请求链路追踪等。

第 3 章 Service Mesh

本章主要介绍了目前非常前沿的一个微服务架构 Service Mesh，讲述其定义、优势、发展历程以及未来可能的发展趋势。

第 4 章 Docker 技术简介

本章主要介绍了 Docker 技术是什么，Docker 与传统 VM 的区别，Docker 的作用和优势，Docker 生态圈的现状与发展，以及微服务与 Docker 的联系等。

第二部分 Docker 架构与生态

第 5 章 Docker 技术架构

本章介绍了 Docker 的进程模型，Docker 在宿主机上的进程特征，Docker 容器的底层实现机制，以及 Docker 在运行时的技术模型。

第 6 章 Docker 逻辑架构

本章介绍了 Docker 中各个核心组件之间的逻辑架构关系，讲解了各个组件之间是如何进行耦合的，最后简单介绍了开源仓库 Harbor 的原理和部署方式。

第 7 章 Docker 网络架构

本章介绍了 Docker 在单机环境下的网络通信模式，包括 Bridge、Host、Container、None 等，同时介绍了 Docker 在跨主机集群环境下的常见网络通信模式，比如 Flannel、L2-FLAT 等。

第 8 章 Docker 安全架构

本章介绍了 Docker 技术中常见的安全问题，比如 Docker 自身隔离机制不足导致的安全缺陷、Docker 镜像被注入恶意代码导致的安全缺陷等，同时提出了一系列的 Docker 安全检查基线，最后介绍了几种常见的 Docker 安全工具。

第 9 章 Docker 与 DevOps

本章介绍了 Docker 中的代码如何挂载，如何在 Docker 容器中实现服务注册发现，同时介绍了 Dockerfile 的写法和规范，以及如何在 Docker 中收集日志、实施监控，如何与 CI/CD 平台整合等。

第 10 章 容器编排

本章介绍了容器编排技术的概念、用途和具体用法，列举了常见的几种容器编排技术，比如 Swarm、Kubernetes 和 Mesos 等，对它们进行了比较，简单介绍了它们的部署安装及使用方法。

第三部分 Docker 落地之路

第 11 章 企业级 Docker 容器云架构

本章介绍了在企业级环境下如何进行整体的 Docker 架构设计，同时介绍了围绕着 Docker 建立企业级容器云平台架构的方法。

第 12 章 基于 Rancher 的容器云管理平台

本章介绍了基于开源 PaaS 软件 Rancher 建立企业级容器云平台的各类方案，以及其中各种技术难点的解决方法。

第 13 章 微服务与 Docker 化实战

本章介绍了如何在某条实际产品线的 Docker 化落地过程中进行平滑切入，如何打通容器网络和物理网络，以及如何将产品线整合进 Rancher 的容器云平台。

致谢

在本书的写作过程中，我得到了大量的帮助，其中有些来自我的同事，有些来自我的领导，有些来自网络博客中的真知灼见。

感谢电子工业出版社博文视点的孙奇俏老师对我的帮助和指正。作为一名理工科出身的程序员，我语言贫乏，词不达意，如果没有孙老师的悉心指教，我想我绝对不可能完成这本书。

我还要感谢我的家人，尤其是我可爱的女儿蒋文婷，她美丽的笑容是我不断努力工作的最大动力，在这里，我想将这本书送给我亲爱的女儿，祝她在人生的成长道路上，常与真理和知识为伴。

蒋彪

2018 年 7 月于南京

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/35033

目录

part one

第一部分

01

在这一部分中，我们将介绍一些与微服务、Docker、DevOps 相关的基础知识，包括什么是微服务、微服务的起源与发展、为什么要使用微服务等。

同时，本部分还将介绍微服务架构中常见的痛点、每个痛点对应的技术选型、微服务架构中为什么要用 Docker、Docker 是什么、Docker 的发展等内容。

通过这一部分的介绍，读者能够快速对这些主流技术有一个宏观的了解。

Docker 与微服务基础

第1章

微服务架构概述

1.1 什么是微服务

我们先来看一下微服务之父 Martin 先生给微服务下的定义。

> The microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. These services are built around business capabilities and independently deployable by fully automated deployment machinery. There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies.

简单翻译一下，如 Martin 所言，将一个单体应用拆分成一组微小的服务组件，每个微小的服务组件运行在自己的进程上，组件之间通过如 RESTful API 这样的轻量级机制进行交互， 这些服务以业务能力为核心，用自动化部署机制独立部署，另外，这些服务可以用不同语言研发，用不同技术来存储数据。以上便是微服务架构的特点。

或者以笔者自己的理解来看，所谓的微服务架构特征如下。

- 在分布式的环境中，将单体应用拆分成一组边界隔离、互相依赖的服务进程单元。
- 微服务之间的通信机制更轻量，包括但并不限于 RESTful。
- 微服务能够按照业务边界分割，自动化部署，自动化发布。
- 微服务的开发语言不限，落地数据形式也不限。