中兴通讯
技术丛书

HZ BOOKS
华章IT

OpenStack CI/CD 原理与实践

OPENSTACK CI/CD: PRINCIPLE AND PRACTICE

董文娟 尚小冬 张军◎著

由中兴通讯OPNFV开源团队撰写，团队对社区的贡献位居全球前3

从系统管理员的角度阐述OpenStack CI/CD系统的组成、架构和原理，是中兴技术团队在OpenStack CI/CD领域的实践经验总结

机械工业出版社
China Machine Press

中兴通讯
技术丛书

OpenStack CI/CD 原理与实践

OPENSTACK CI/CD: PRINCIPLE AND PRACTICE

董文娟 尚小冬 张军◎著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

OpenStack CI/CD：原理与实践 / 董文娟，尚小冬，张军著．—北京：机械工业出版社，2018.10
（中兴通讯技术丛书）

ISBN 978-7-111-61191-2

I. O… II. ①董… ②尚… ③张… III. 计算机网络 IV. TP393

中国版本图书馆 CIP 数据核字（2018）第 237998 号

OpenStack CI/CD：原理与实践

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：李 艺 责任校对：殷 虹
印 刷：北京诚信伟业印刷有限公司 版 次：2019 年 1 月第 1 版第 1 次印刷
开 本：186mm×240mm 1/16 印 张：18.25
书 号：ISBN 978-7-111-61191-2 定 价：69.00 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

华章IT
HZBOOKS | Information Technology

Foreword 推荐序一

2018 年 7 月 12 日，我正在美国波特兰出差，很荣幸收到张军的邀请，为他们团队最新撰写的书写序。与 OpenStack 相关的书籍在国内已经有很多了，而且 OpenStack 在国内的发展也进入了一个成熟期，但是特别讲 OpenStack CI/CD 或者定制化 CI/CD 系统的书籍，市场上鲜有耳闻。恰逢此时，7 月 14 日，美国商务部正式发布公告，宣布解除对中兴的拒绝令。中兴通讯也在官方微博上发布一条振奋人心的信息："满怀信心再出发！"。相信此刻中兴和上万员工的心情是激动的，我们所有人心情也是激动的。反思这件事情，我们的教训是深刻的，没有技术的自主创新和自主可控，就意味着将来可能要被别人牵着鼻子掐着脖子。因此，在我们专注的软件领域，我们要坚定不移地积极地投入开源事业。

言归正传，认识张军是通过 OPNFV（Open Platform Network Function Virtualization）社区，他是中兴通讯在 OPNFV 技术指导委员会的代表，在我的印象中，他是一个十分严谨的人。当他把本书草稿的电子版发给我后，我粗略地看了一遍。这本书是他和他的团队对 OpenStack CI/CD 系统各个组件开发和研究多年的成果，也是团队多年实践和经验的总结。除了 OpenStack 系统，现在很多软件发行版比如 Linux、Android、OPNFV，甚至是边缘计算项目 StarlingX 都可以充分利用 CI/CD 系统。正如他在书中所提到的，OpenStack CI 的框架目前只用于 OpenStack 社区基础设施的管理，了解前台的 OpenStack 的人很多，但是由于 CI/CD 是在后台的，接触和了解后面 CI/CD 的开发者却很少。所以，对于一个要全面掌握系统的开发人员来说，了解和熟悉 CI/CD 也是十分必要的。而且，CI/CD 的适用范围很广，定制化非常高，熟悉了 OpenStack CI/CD，则一通而百通。

书中除了介绍 OpenStack CI/CD 中的关键技术和组件之外，还有一大特色是分享了中兴团队在社区 CI/CD 工具集上的实践经验，同时指导读者如何定制化和修改 OpenStack CI/CD 系统，包括剪裁和扩展，即告诉读者如何以一通百。这一点是非常有益的，因为针对不同内部使用需求，针对不同项目需求，是不能直接将 OpenStack CI/CD 拿来使用的，在这点上

我深有体会。举个例子，我现在在英特尔公司从事边缘计算和 StarlingX 发行版的开发工作，第一个艰巨任务就是要编译和管理 StarlingX 的众多依赖包，并快速搭建起 StarlingX 的 CI/CD 体系。目前这项任务由以前负责 Linux 开发的墨西哥团队执行，同时该任务也是该边缘计算其他所有功能开发的前提和依赖。它不是简单 OpenStack CI/CD 的拷贝，在这条路上我们也费了些周折。如果墨西哥团队能了解 OpenStack CI/CD，并掌握 CI/CD 的定制化，以后在任何项目的 CI/CD 搭建道路上就可以少走很多弯路，相信本书对需要根据项目搭建并定制 CI/CD 系统的读者有很大的帮助。

另外，开源的持续集成 / 部署平台 Zuul 项目最初是为 OpenStack CI 测试开发的，后来被许多不同的组织所贡献和使用。在今年 5 月份的 OpenStack Vancouver Summit 上，OpenStack 基金会宣布 Zuul 项目发布第 3 版，并且正式成为由 OpenStack 基金会托管的第三个独立项目。这一点也足以说明 CI/CD 系统对一个项目的重要性。需要了解 Zuul 项目的读者，在本书中也能找到相应的答案。

最后，感谢张军和他的团队为 OpenStack 社区，为开源软件社区贡献了这样一本好书，希望众多 OpenStack 的开源项目的开发人员、运维人员，以及爱好者们能在书中找到它的价值，并获得帮助。

王庆博士，英特尔开源技术中心网络和存储开发经理，OpenStack 基金会个人独立董事

Forword 推荐序二

读了董文娟等同事创作的这本书，我感到非常高兴，并由衷感谢创作团队的杰出贡献！

云计算、大数据、人工智能是21世纪发展的基石和助推器，它们将共同促进未来生产力的提升。电信设备在往NFV、云原生演进的过程中越来越多地使用和借鉴了IT思想和工具链。这些新技术的应用，加速了整个产业链的发展。5G就是典型的综合应用场景，云原生、微服务和HTTP接口规范，奠定了5G蓬勃发展的基础。中兴通讯在云计算领域深耕多年，致力于利用先进的技术，研发更快速度、更大容量、更高安全、更具弹性、更低成本的云计算基础设施。OpenStack已经成为开源云计算领域的事实标准，其社区拥有一套完整的、标准化的、自动化的持续集成测试平台，它不仅适用于开源社区本身，也有助于电信产品的研发。本书记录了中兴通讯在这个领域的探索、研究和实践，对于构建自动化的测试系统具有很好的借鉴意义，对于公司数字化转型起到了基础支撑作用。

中兴通讯非常认同和重视开源社区，且致力于建立开放合作的生态环境。公司加入了全球多个开源的社区，成为其中的关键伙伴，比如Apache、CNCF、OpenStack和OPNFV等。中兴通讯在NFV领域技术的发展，离不开与社区的合作。中兴通讯作为最负社会责任的高科技企业之一，我们非常愿意将我们的知识、经验和服务分享到社区，回馈到社会。

本书创作团队是公司在开源社区合作的典范，是一个优秀的自组织、自管理、自激励的开放合作的敏捷团队，他们内通外联与社区合作，共同推动NFV和开源技术的发展演进。本书是团队在OpenStack CI/CD领域多年的研究和实践总结，很乐意将这本优秀的著作分享给大家！

最后再次感谢创作团队的重大贡献，并欢迎各位读者开启精彩的阅读之旅！

施嵘，中兴通讯无线研究院院长

前　言 *Preface*

本书由来

过去十年，中国电信业快速发展，语音和数据业务需求极大提升，尤其当以安卓、iPhone 为代表的智能手机推出后，数据业务的增长速度远超预期。近三年，移动数据业务每年以大约 90% 的复合增长率增长[⊖]，这对设备商交付新版本的速度提出了更高的要求。中兴通讯在 2013 年就启动了无线接入网项目移植虚拟化技术预研，希望实现软硬件解耦以缩短版本交付周期，满足运营商业务飞速发展的需求。目前，这些虚拟化研究成果已经成为 5G 的基础技术标准。

虚拟化预研项目不仅涉及电信网元本身的适配，还涉及开发云操作系统和制定操作管理规范、接口标准。而云操作系统是虚拟化的关键技术之一，综合考虑云操作系统开源社区影响力、项目热度、扩展性、可维护性、业界认可度等因素，项目团队选择 OpenStack 作为云操作系统的开源解决方案。但电信领域常见的需求，如网口绑定、高速转发、巨页（Huge Page）、CPU 绑核等功能在 OpenStack 社区尚不支持，如何把虚拟化的电信网元（Virtual Network Function）运行在 OpenStack 上，并提供高性能服务，是项目面临的巨大挑战。在此背景下，笔者所在的团队开始深入参与到 OpenStack 开源社区的开发工作中，推动社区一起解决这些关键技术问题。

计算、存储和网络是 OpenStack 虚拟层三大基础设施服务，其中存储（Cinder）项目涉及厂商的硬件集成。Cinder 项目制定了一套 API 接口规范，厂商的存储设备若需要集成，需向社区提交遵守该接口规范的驱动程序，并搭建一套 CI 系统来验证，该 CI 系统需要遵守社区第三方 CI 规范[⊜]。

图 0-1 是一个简化的第三方 CI 系统架构图，厂商需要准备的有：

⊖ http://www.chyxx.com/industry/201708/546504.html。

⊜ https://docs.openstack.org/infra/system-config/third_party.html。

- 第三方 CI 系统，即本书所要讲述的 OpenStack CI 系统；该系统连接厂商的存储硬件，并与社区的 Gerrit 系统[⊖]连接，监控 stream-events 事件流；当社区 Gerrit 有新的变更和补丁提交时，将变更代码与厂商提供的硬件环境进行集成测试并向社区 Gerrit 反馈测试结果；
- 本地 OpenStack 资源池，为第三方 CI 系统运行提供虚拟机资源；
- 存储设备，用于对变更代码进行验证。

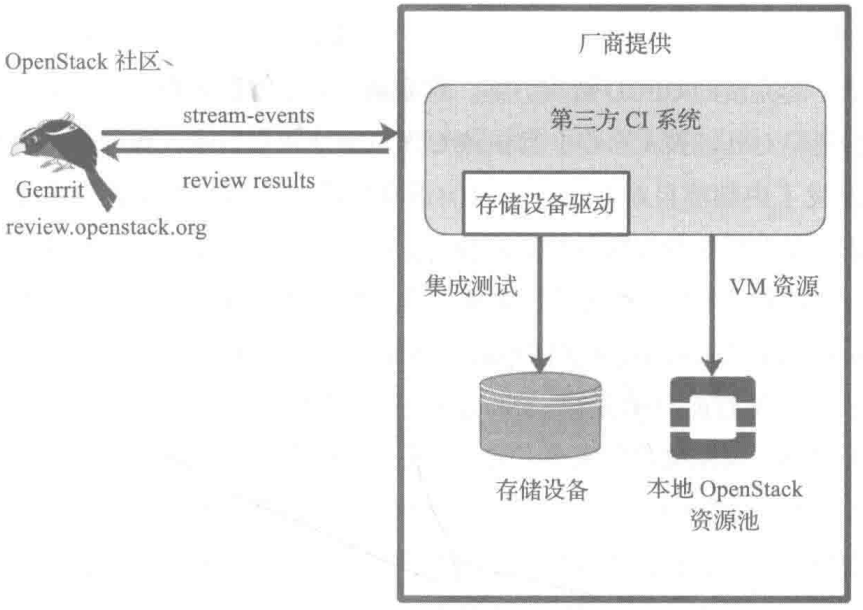


图 0-1　第三方 CI 架构图

通过这种机制，Cinder 项目可以及时发现变更代码的兼容性问题，厂商也能及时发现变更代码对设备驱动的影响。所有第三方 CI 系统的执行结果都会显示在社区 Gerrit 系统每个代码变更的评审记录中。

OpenStack CI 系统提供了一套动态调度测试任务的方法，极大提升了开发效率，具体优点如下：

- 提高 CI 系统的并发性，开发人员的多个变更测试任务不需要在环境排队等待执行，只要本地 OpenStack 资源池资源足够多，就可以并行执行所有测试任务，这可以显著减少开发人员的等待时间；
- 提高测试任务的可管理性，使用 YAML 文件描述测试任务，易阅读、评审，可进行版本管理，容易在不同团队之间复制和重用；

⊖ https://review.openstack.org。

- 可定制虚拟机基础镜像，如支持不同的操作系统、不同的软件栈配置，增加了在项目实际使用中的适用性和灵活性；
- 提高测试任务的准确性，每次测试任务都在相同的环境和配置下执行，不存在静态环境中测试任务执行后对软件、配置文件等清理不完整而导致测试任务执行不一致的问题；
- 可复用云环境，统一运维资源，减少维护开销并提高资源利用率。

这套 CI 系统不仅适用于 OpenStack 社区组件研发，也适用于电信网元的开发。基于此，笔者所在的团队潜心研究 OpenStack CI 这套系统，结合自身需求，增强了 CD 的方案和实践，最终形成一套完整的 CI/CD 解决方案。这套解决方案在团队得到广泛应用，无论是内部的维护工作，还是对外的技术合作，都由该 CI/CD 系统提供技术支撑。公司数字化转型战略和敏捷实践触发了内部项目对 CI/CD 技术的重视，越来越多的团队渴望获得和提升这方面的技能。所以，笔者所在的团队作为开源技术的先锋，在给内部进行培训和技术转移的过程中，诞生了把对该系统的经验、实践汇总整理并出版成书的想法。

了解 OpenStack 的人很多，但对 OpenStack CI 系统有实际应用经验的人很少，主要是因为 OpenStack CI 系统目前只用于 OpenStack 社区基础设施的管理。国内 DevOps 思想和技术在最近两年才得到大规模接受和实践，大家对了解 CI/CD 相关技术的需求非常高。笔者很荣幸把这套适用性广、定制性强和并发性高的 CI 系统介绍给大家，同时回馈社区，推广和普及这些基础设施技术。目前，国内外几乎没有该套系统的完整描述，因其组件众多，且搭建该系统需要投入大量的人力物力（笔者团队搭建该系统花费了近两个月的时间，这还是在使用社区已有安装脚本进行最简安装的情况下）。从目前已经接入的第三方 CI[㊀]环境来看，国内搭建 OpenStack CI 系统的公司凤毛麟角，说明在 OpenStack 技术如此普及的情况下，国内公司对这套 OpenStack CI 系统仍然知之甚少。希望本书能让读者少走弯路，事半功倍[㊁]。

本书结构

本书从逻辑上可分为四部分：

- 第一部分（第 1 章和第 2 章）对 DevOps 的发展、文化和转型进行了简单说明，并引入本书的重点内容 OpenStack CI/CD 说明其对 DevOps 转型的重要性。
- 第二部分（第 3 ～ 9 章）以系统管理员视角对 OpenStack CI/CD 中的每一项关键技术进行详细的分析和阐述，这些关键技术包括：

㊀ https://wiki.openstack.org/wiki/ThirdPartySystems。

㊁ 本书使用 reStructuredText 语言编写，团队因采用该工具而在格式排版方面节省了大量时间精力。

- 版本控制（Git）与代码评审（Gerrit）系统，开源社区最常用的代码管理和代码评审工具，Gerrit 系统与本书介绍的持续集成系统和门控系统进行密切配合，共同完成整个开发流程。
- 持续集成系统（Jenkins），最为流行的一款开源持续集成工具，用于代替人工进行重复的持续集成工作，同时也为促进持续交付提供技术支撑；该章还介绍了 JJB（Jenkins Job Builder）工具，通过 YAML 文件来定义 Jenkins 任务，提升 Jenkins 任务的开发和运维效率。
- 门控系统（Zuul），确保主线稳定，仅当与变更相关的所有测试都通过后，才会将变更合入相应的目标分支中。它保障了测试任务获得最大程度的并行，且在发生错误的情况下，自动回滚；它支持指定项目间变更依赖关系，解决跨项目集成问题，致力于守护项目及关联项目。
- 资源管理系统（Nodepool），通过在 OpenStack 资源池中动态创建虚拟机并作为 Jenkins Slave 节点添加到 Jenkins 的资源池中，确保不同的测试之间互不影响；在测试结束后，删除使用过的虚拟机，保证对每个测试任务都能提供一致的测试环境；此外，Nodepool 还集成了镜像制作功能，支持测试环境的定制开发。
- 日志服务器，对所有测试任务提供统一的日志存储服务，以便回溯测试故障。
- 日志分析系统，将测试过程中的测试日志导入 Elasticsearch 中，便于出现问题时，快速检查和分析故障。
- 公共组件，重点介绍了支持 OpenStack CI 系统组件之间进行协作和通信的系统，包括任务分发（Gearman）、消息队列（ZeroMQ）和分布式协调服务（ZooKeeper）等组件。

- 第三部分（第 10 章）描述团队在社区工具和经验的基础上的经典实践，以及如何进行裁剪、扩展和定制化修改以满足团队需求。
- 第四部分（第 11 章）探讨当前解决方案中存在的不足和可行的优化方案，并描述了社区当前正在经历的变化和未来的演进路线。

尽管本书各章节功能相对独立，笔者建议顺序阅读第 3 ～ 5 章，因为这些章节大量使用了前序章节的基本概念和术语；其他章节可以根据需要进行翻阅。

本书读者

本书面向基础设施管理团队、DevOps 团队或致力于了解 CI/CD 的技术人员，可作为学习 OpenStack 社区 CI/CD 基础设施解决方案的教材，协助你高效搭建社区第三方 CI 系统。

勘误与交流

笔者团队在编写本书的过程中虽已经进行了大量的内部评审和检验工作，但因时间紧、精力有限，难免会出现一些错漏，敬请广大专家和读者批评、指正。你可以把与本书相关的意见或建议发送到电子邮箱 openzero@aliyun.com 中。

本书中团队实践和示例代码都可以在 GitHub[⊖]上进行下载，读者可以尝试使用并与我们进行探讨。

致谢

本书的创作团队是一支拥有丰富专业技术经验的团队，团队成员有十年以上电信领域的工作经验、四年以上开源社区技术研究和贡献经验。本书是团队智慧和多年经验的结晶。使用本书介绍的技术和经验，希望可以帮助你的团队高效管理公司内部的基础设施，节省大量人力资源，并提供快速而稳定的服务。

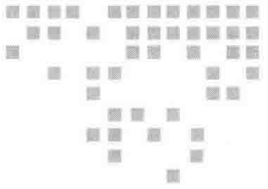
本书由创作团队在业余时间编写，占用了团队成员很多家庭生活的时间，每位成员都付出了极大的努力并克服了诸多困难，对此表示感谢。感谢所有团队成员不厌其烦，一次又一次地评审修订，以认真严谨的态度验证了本书所有的代码实践。感谢董文娟在本书编写过程中有力地组织、协调和推进，没有你辛勤的付出，本书不可能顺利完成。

在此，还要感谢我的领导潘峰、王前和施嵘，没有你们的信任和支持，不可能有本书的诞生。感谢中兴学院的闫林院长对本书的指导和建议。感谢 OpenStack 基础设施团队设计出如此优秀的工具和系统，感谢你们在日常工作中及时答疑解惑。感谢机械工业出版社的杨福川和李艺编辑的信任和支持，及时更正了本书中较多的排版问题。

张　军

⊖ https://github.com/openzero

Contents 目　录



第 1 章 Chapter 1

DevOps

什么是 DevOps? 为什么 DevOps 会受到越来越多的关注?

“DevOps”是英文单词“Development”和“Operation”的组合，即开发和运维的结合。目前 DevOps 并没有权威的定义，但得到大部分人认可的是，DevOps 已经成为一种文化价值观和实践方法。DevOps 价值观的呈现和实践并不依赖于特定的软件，但通过合适的软件工具链的支撑可以更容易实现这一价值观。

DevOps 使得业务可以更快地运营，更快地应对变化。2017 年 DevOps 现状调查报告[⊖]显示，工作于 DevOps 团队的人员比例达到 27%，相比于 2014 年，增加了一倍。DevOps 能够缩短上市时间，提升产品质量并增加营收，从而受到越来越多的公司关注并进行实践。

1.1 DevOps 简介

谈到 DevOps，就不得不回顾一下软件开发模型（Software Development Model）[⊜]的历史。软件开发模型是指软件开发全部的过程、活动和任务的结构框架。它包括需求、设计、编码和测试等阶段，有时也包括维护阶段。软件开发模型能清晰、直观地表达软件开发全过程，明确要完成的主要活动和任务。本章不会系统介绍曾经出现的所有软件开发模型，只选取其中较典型的瀑布模型、迭代模型、敏捷模型进行说明，以及如何演进到 DevOps。

⊖ https://ask.qcloudimg.com/raw/yehe-133f028/dorjc09gaa.pdf。

⊜ https://baike.baidu.com/item/ 软件开发模型。