Docker实践

作者：伊恩·米尔(Ian Miell)

目　　录

版权信息

书名：Docker实践

ISBN：978-7-115-47458-2

- 著　　[美] 伊恩 • 米尔（Ian Miell）

  [美] 艾丹 • 霍布森 • 塞耶斯（Aidan Hobson Sayers）

  译　　吴佳兴　梁晓勇　黄博文　杨　锐

  责任编辑　杨海玲

- 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号

  邮编　100164　电子邮件　315@ptpress.com.cn

  网址　http://www.ptpress.com.cn

- 读者服务热线：(010)81055410

  反盗版热线：(010)81055315

版权声明

内容提要

本书由浅入深地讲解了Docker的相关内容，涵盖从开发环境到DevOps流水线，再一路到生产环境的整个落地过程以及相关的实用技巧。书中介绍Docker的核心概念和架构，以及将Docker和开发环境有机、高效地结合起来的方法，包括用作轻量级的虚拟机以及构建和宿主机编排、配置管理、精简镜像等。不仅如此，本书还通过“问题/解决方案/讨论”的形式，将“Docker如何融入DevOps流水线”“如何在生产环境落地”等一系列难题拆解成101个相关的实用技巧，为读者提供解决方案以及一些细节和技巧方面的实践经验。阅读本书，读者将学到的不只是Docker，还包括持续集成、持续交付、构建和镜像管理、容器编排等相关领域的一线生产经验。本书编写时一些案例参考的Docker版本是Docker 1.9。

本书要求读者具备一定的容器管理和运维的基础知识，适合想要将Docker投入实践的相关技术人员阅读，尤其适合具有中高级DevOps和运维背景的读者阅读。

译者简介

吴佳兴，毕业于华东理工大学计算机系，目前是bilibili基础平台研发团队的一员，主要研究方向有CI\CD、监控和运维自动化、基于容器的PaaS平台建设、微服务架构等。2014年年底有幸加入DockOne社区，作为译者，利用闲暇时间为社区贡献一些微薄的力量。个人博客devopstarter.info。欢迎邮件联系（wjx_colstu@hotmail.com）。

黄博文，ThoughtWorks资深软件工程师/咨询师，担任过开发、测试、运维、技术经理等角色，在国内外多家企业做过技术教练以及技术咨询，拥有丰富的敏捷团队工作经验。目前专注于DevOps技术及云端架构，在搭建持续集成及部署平台、自动化构建基础设施、虚拟化环境、云端运维等方面有着丰富的经验。拥有AWS解决方案架构师以及开发者证书。个人博客为www.huangbowen.net，个人邮箱为huangbowen521@gmail.com。译作有《Effective JavaScript》《Html5和CSS3响应式设计指南》《C#多线程编程实战》《面向对象的思考过程》《基础设施即代码》等。

杨锐，前ThoughtWorks咨询师，DevOps领域持续关注者，任职期间曾任某海外大型项目DevOps工程师，对其持续交付、基础设施即代码、流水线即代码等方面进行了持续推动，对云计算、容器化、持续交付等有一定经验。现供职美团点评。

梁晓勇，毕业于厦门大学，现任某互联网金融公司架构师，DockOne社区编外人员。长期奋战在技术研发第一线，在网络管理、技术开发、架构设计等方面略有心得。热爱互联网技术，积极投身开源社区，对Docker等容器技术具有浓厚兴趣。欢迎邮件联系（sunlxy@yahoo.com）。

序

可能是鄙人的拙见，不过Docker真的相当了不起。

在不久以前应用程序还是一些庞大的单体软件，独自运行在一堆钢铁和硅块上。这样的情况持续了很多年，它们拒绝改变，不思进取。对于那些想要快速前行的组织而言，这的确是一个难题，因此虚拟机的崛起也就不足为奇了。应用程序不必再和这些硬件捆绑在一起，这使得一切可以更快更替，也更加灵活。

但是，虚拟机本身也很复杂。它们在其他的机器上模拟出整台计算机，而这台虚拟计算机仍然是异常复杂并且需要管理的。再者，因为虚拟机更小而且更容易被创建，所以围绕着它们也就产生了更多需要管理的东西。

我们到底该如何管理所有的复杂性呢？通过配置管理可以办到，当然，这又是另外一个极其复杂的系统。

Docker则采取了一种截然不同的做法。如果用户将软件放到一个容器里，它就会将应用程序本身的复杂度与基础设施相隔离开来，这使得基础设施本身可以变得更加简单，而应用程序也更加易于装配。基于这样的组织效率，与虚拟机相比，技术速度和执行效率都有了巨大的飞跃。容器的启动是毫秒级的，而不是分钟级的。内存是共享的，而不是预先分配的。这使得用户的应用程序能够以更低的成本运行，同样也意味着他可以按照想要的方式来架设应用，而不用再受缓慢的、不太灵活的基础设施的制约。

在我第一次见到Solomon Hykes（即“Docker之父”）谈论Docker并将其与装载集装箱做类比的时候，我便意识到他正在做的是一件了不起的大事。全球航运业建立标准之前的混乱状态可以很好地用来类比容器出现以前管理软件的复杂状态。Solomon的观点非常有说服力，我甚至为此开了一家公司，专门围绕Docker构建工具，这家公司最终被Docker公司收购了，并且最终变成了我们现在所熟知的Docker Compose。

初次见到Ian是在伦敦我们组织的某个Docker聚会（meetup）上。当时，我们坚称：“Docker还没有为生产环境做好准备，请不要在生产环境里使用它！”然而，Ian是何许人也，他无视这个看似明智的建议，一意孤行，非要让它运行在生产环境里。那时，他和Aidan一起为OpenBet公司工作，当我看到他们用我们当时的代码处理过的货币数额时，真的让我感到有点儿懵。

Ian和Aidan发现，他们在使用Docker的过程中收获的价值超越了其测试阶段使用它时伴随而来的不便之处。他们在很早的时候便用上了这一技术，因此，对怎样更好地运用它也有自己独到的见解。他们在OpenBet内部构建的工具指出了Docker本身缺失的一些东西，而我们之间私下的一些闲聊也实际影响了我们所采用的设计和产品方向。

Docker自Ian和Aidan第一次使用以来，已然有了一些长足的进步，到如今已有成千上万的组织在用它来解决遇到的真实问题，如更快地装配软件、管理惊人的复杂性、提高基础设施的效率、解决“这个在我机器上运行得好好的”问题等。这促使我们在构建、部署和管理软件方式上的巨大转变，并且围绕着它正在形成一套全新的工具及理念。容器化的美好前景的确是让人兴奋的，但是它同我们之前所习惯的方式也迥然不同。

对读者而言，恐怕很难从这本书中看到怎样由此及彼，但是这本书中涵盖了大量如何应用Docker去解决自己当下遇到的种种问题的实用建议。遵循这些建议，用户的组织将能够快速前行。同时或许更重要的是，构建和部署应用的过程将会变得更有意思。

Ben Firshman

Docker公司产品管理总监、Docker Compose联合创始人