O'REILLY

Docker Cookbook

Docker经典实例（影印版）

東南大學出版社

Sébastien Goasguen 著

Docker经典实例（影印版）

Docker Cookbook

Sébastien Goasguen 著

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

O'Reilly Media, Inc.授权东南大学出版社出版

南京 东南大学出版社

图书在版编目(CIP)数据

Docker 经典实例:英文/(美)塞巴斯蒂安·戈阿冈(sèbastien Goasguen)著. —影印本. —南京:东南大学出版社,2017.1

书名原文:Docker Cookbook

ISBN 978-7-5641-6899-5

Ⅰ.①D… Ⅱ.①塞… Ⅲ.①Linux 操作系统—程序设计—英文 Ⅳ.①TP316.85

中国版本图书馆 CIP 数据核字(2016)第 319378 号

图字:10-2015-247 号

Docker 经典实例(影印版)

出版发行:东南大学出版社

地　　址:南京四牌楼 2 号　　邮编:210096

出 版 人:江建中

网　　址:http://www.seupress.com

电子邮件:press@seupress.com

印　　刷:常州市武进第三印刷有限公司

开　　本:787 毫米×980 毫米　16 开本

印　　张:23

字　　数:450 千字

版　　次:2017 年 1 月第 1 版

印　　次:2017 年 1 月第 1 次印刷

书　　号:ISBN 978-7-5641-6899-5

定　　价:79.00 元

本社图书若有印装质量问题,请直接与营销部联系。电话(传真):025-83791830

Praise for *Docker Cookbook*

Starting with Docker is one thing, but really grasping the concept of it is another. It requires a sound understanding. This cookbook helped us tremendously in implementing Docker in the application landscapes we run for our customers.

—Arjan Eriks, Cloud Computing Services Director, Schuberg Philis

This is a whirlwind tour of the ever-expanding collection of tools and platforms that work with Docker, with specific and practical examples. As the core functionality of Docker becomes even more of an industry standard through the efforts of the Open Container Initiative, expect to see the this ecosystem continue to expand rapidly. Sébastien's book can give any practitioner the solid grounding they need to keep up with this pace of change.

—Chip Childers, Vice President of Technology, Cloud Foundry Foundation

Sébastien does a terrific job of encapsulating Docker best practices and introductory material for the novice user across networking, image management, configuration, and very fast moving orchestration and scheduling ecosystem including Kubernetes and Mesos/Marathon.

—Patrick Reilly, CEO, Kismatic

Preface

Why I Wrote This Book

I have been working on clouds at the IaaS layer for over 10 years. With Amazon AWS, Google GCE, and Microsoft Azure now providing large-scale cloud services for several years, it is fair to say that getting access to a server has never been this easy and this quick. The real value to me has been the availability of an API to access these services. We can now program to create an infrastructure and program to deploy an application. These programmable layers help us reach a higher level of automation, which for a business translates in faster time to market, more innovation, and better user service.

However, application packaging, configuration, and composition of services in a distributed environment has not progressed much despite a lot of work in configuration management and orchestration. Deploying and running a distributed application at scale and in a fault-tolerant manner is still hard.

I was not crazy about Docker until I tried it and understood what it brings to the table. Docker primarily brings a new user experience to Linux containers. It is not about full virtualization versus containers; it is about the ease of packaging and running an application. Once you start using Docker and enjoy this new experience, the side effect is that you will also start thinking automatically about composition and clustering.

Containers help us think more in terms of *functional isolation*, which in turn forces us to decompose our applications before stitching them back together for a distributed world.

How This Book Is Organized

This cookbook contains 10 chapters. Each chapter is composed of *recipes* written in the standard O'Reilly recipe format (Problem, Solution, Discussion). You can read

this book from front to back or pick up a specific chapter/recipe. Each recipe is independent of the others, but when concepts from another recipe are needed, appropriate references are provided.

- Chapter 1 goes through several Docker installation scenarios, including using *Docker Machine*. It then presents the basic Docker commands to manage containers, mount data volumes, link containers, and so on. At the end of this chapter, you will have a working Docker host and you will have started multiple containers as well as understood the life cycle of containers.
- Chapter 2 introduces the *Dockerfile* and *Docker Hub* and shows how to build/tag/commit an image. The chapter also shows how to run your own Docker registry and set up automated builds. At the end of this chapter, you will know how to create Docker images and share them privately or publicly and have a basic foundation to build continuous delivery pipelines.
- Chapter 3 explains the networking mechanisms in Docker. You will learn how to get containers' IP addresses and how to expose a container service on a specific host port. You will also learn about linking containers, and how to use nondefault networking configurations. This chapter contains a few recipes that provide a deeper understanding of networking containers. Concepts such as network namespaces, using an OVS bridge, and GRE tunnels are presented to lay a strong foundation for container networking. Finally, you will also learn about more advanced networking setups and tools, such as Weave, Flannel, and the currently experimental Docker Network feature.
- Chapter 4 goes through configuration of the Docker daemon, especially security settings and remote access to the Docker API. It also covers a few basic problems, like compiling Docker from source, running its test suite, and using a new Docker binary. A few recipes provide better insight on Linux namespaces and their use in containers.
- Chapter 5 introduces the new container management platform from Google. Kubernetes provides a way to deploy multicontainer applications on a distributed cluster. In addition, it provides an automated way to expose services and create replicas of containers. The chapter shows how to deploy Kubernetes on your own infrastructure, starting with a local Vagrant cluster and subsequently on a set of machines started in the cloud. I then present the key aspects of Kubernetes: *pods*, *services*, and *replication controllers*.
- Chapter 6 covers four new Linux distributions that are optimized to run containers: CoreOS (*https://coreos.com*), Project Atomic (*http://www.projectatomic.io*), Ubuntu Core (*http://www.ubuntu.com/cloud/tools/snappy*), and RancherOS (*http://rancher.com/rancher-os/*). These new distributions provide just enough operating system to run and orchestrate Docker containers. Recipes cover installation and access to machines that use these distributions. This chapter also

introduces tools that are used with these distributions to ease container orchestration (e.g., `etcd`, `fleet`, `systemd`).

- One of Docker's strengths is its booming ecosystem. Chapter 7 introduces several tools that have been created over the last 18 months and that leverage Docker to ease application deployment, continuous integration, service discovery, and orchestration. As an example, you will find recipes about Docker Compose, Docker Swarm, Mesos, Rancher, and Weavescope.
- The Docker daemon can be installed on a developer local machine. However, with cloud computing providing easy access to on-demand servers, it is fair to say that a lot of container-based applications will be deployed in the cloud. Chapter 8 presents recipes to show how to access a Docker host on Amazon AWS (*http://aws.amazon.com*), Google GCE (*https://cloud.google.com/compute/*), and Microsoft Azure (*http://azure.microsoft.com/en-us/*). The chapter also introduces two new cloud services that use Docker: the AWS Elastic Container Service (ECS (*http://aws.amazon.com/ecs/*)) and the Google Container Engine (*https://cloud.google.com/container-engine/*).
- Chapter 9 addresses concerns about application monitoring when using containers. Monitoring and visibility of the infrastructure and the application have been a huge focus in the DevOps community. As Docker becomes more pervasive as a development and operational mechanism, lessons learned need to be applied to container-based applications.
- Chapter 10 presents end-to-end application deployment scenarios on both single hosts and clusters. While some basic application deployments are presented in earlier chapters, the recipes presented here are closer to a production deployment setup. This is a more in-depth chapter that puts you on the path toward designing more-complex microservices.

Technology You Need to Understand

This intermediate-level book requires a minimum understanding of a few development and system administration concepts. Before diving into the book, you might want to review the following:

Bash (Unix shell)
: This is the default Unix shell on Linux and OS X. Familiarity with the Unix shell, such as editing files, setting file permissions, moving files around the filesystems, user privileges, and some basic shell programming will be beneficial. If you don't know the Linux shell in general, consult books such as Cameron Newham's *Learning the Bash Shell* or Carl Albing, JP Vossen, and Cameron Newham's *Bash Cookbook*, both from O'Reilly.

Package management

The tools in this book often have multiple dependencies that need to be met by installing some packages. Knowledge of the package management on your machine is therefore required. It could be *apt* on Ubuntu/Debian systems, *yum* on CentOS/RHEL systems, *port* or *brew* on OS X. Whatever it is, make sure that you know how to install, upgrade, and remove packages.

Git

Git has established itself as the standard for distributed version control. If you are already familiar with CVS and SVN, but have not yet used Git, you should. *Version Control with Git* by Jon Loeliger and Matthew McCullough (O'Reilly) is a good start. Together with Git, the GitHub (*http://github.com*) website is a great resource to get started with a hosted repository of your own. To learn GitHub, try *http://training.github.com* and the associated interactive tutorial (*http://try.github.io*).

Python

In addition to programming with C/C++ or Java, I always encourage students to pick up a scripting language of their choice. Perl used to rule the world, while these days, Ruby and Go seem to be prevalent. I use Python. Most examples in this book use Python, but there are a few examples with Ruby, and one even uses Clojure. O'Reilly offers an extensive collection of books on Python, including *Introducing Python* by Bill Lubanovic, *Programming Python* by Mark Lutz, and *Python Cookbook* by David Beazley and Brian K. Jones.

Vagrant

Vagrant has become one of the great tools for DevOps engineers to build and manage their virtual environments. It is best suited for testing and quickly provisioning virtual machines locally, but also has several plug-ins to connect to public cloud providers. This book uses Vagrant to quickly deploy a virtual machine instance that acts as a `docker` host. You might want to read *Vagrant: Up and Running* from the author of Vagrant itself, Mitchell Hashimoto.

Go

Docker is written in Go. Over the last couple of years, Go has established itself as the new programming language of choice in many start-ups. This cookbook is not about Go programming, but it shows how to compile a few Go projects. Some minimal understanding of how to set up a Go workspace will be handy. If you want to know more, the *"Introduction to Go Programming" video training course* is a good start.

Online Content

Code examples, Vagrantfiles, and other scripts used in this book are available at GitHub (*https://github.com/how2dock/docbook*). You can clone this repository, go to the relevant chapter and recipe, and use the code as is. For example, to start an Ubuntu 14.04 virtual machine using Vagrant and install Docker:

```
$ git clone https://github.com/how2dock/docbook.git
$ cd dockbook/ch01/ubuntu14.04/
$ vagrant up
```

The examples in this repo are not made to represent optimized setups. They give you the basic minimum required to run the examples in the recipes.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
: Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context.

This element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

Safari® Books Online

Safari Books Online is an on-demand digital library that delivers expert content in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of plans and pricing for enterprise, government, education, and individuals.

Members have access to thousands of books, training videos, and prepublication manuscripts in one fully searchable database from publishers like O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and hundreds more. For more information about Safari Books Online, please visit us online.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at *http://bit.ly/docker-ckbk*.

To comment or ask technical questions about this book, send email to *bookquestions@oreilly.com*.

For more information about our books, courses, conferences, and news, see our website at *http://www.oreilly.com*.

Find us on Facebook: *http://facebook.com/oreilly*

Follow us on Twitter: *http://twitter.com/oreillymedia*

Watch us on YouTube: *http://www.youtube.com/oreillymedia*

Acknowledgments

Writing this book turned out to be an eight-month project. During that time, I read countless blogs and documentation about Docker, and tested everything, and then retested and tested again. Of course, I would like to thank my wife and kids, who gave me time on weekends and at night to write this book. I would also like to thank Brian Anderson, who kept on pushing me, encouraging me, and thanks to regular check-ups, kept me on time and on target. The book would not be complete without additions from Fintan Ryan, Eugene Yakubovich, Joe Beda, and Pini Riznik. Those four guys helped me generate valuable content that will help many readers. Many thanks also go to Patrick Debois and John Willis for their early review of the books, which provided encouragement and valuable feedback to make the book even better. The thorough reviews from Ksenia Burlachenko and Carlos Sanchez helped me fix a good number of issues that will help all readers; many thanks to the two of them. Special thanks to Funs Kessen, who has been a great sounding board on networking and application design and who never turned down my many stupid questions. Finally, many thanks to the early-release readers, especially Olivier Boudry, who were willing to read the book with incomplete content, typos, bad grammar, and a few mistakes; without their corrections and comments, the book would not be what it is now.

Table of Contents

CHAPTER 1

Getting Started with Docker

1.0 Introduction

Getting started with Docker is straightforward. The core of Docker is made of the Docker engine, a single-host software daemon that allows you to create and manage containers. Before diving into using Docker, you need to install the Docker engine on a host, either your desktop, laptop, or a server.

The first recipes in this chapter go through the installation steps to get Docker running in your server. The official Docker documentation covers almost all cases of operating systems. Here we cover Ubuntu 14.04 (Recipe 1.1), CentOS 6.5 (Recipe 1.2) and CentOS 7 (Recipe 1.3). If you want to use Vagrant, Recipe 1.4 is for you.

We also show how to install Docker on Raspberry Pi (Recipe 1.5) to present an installation for ARM processors. For Windows and OS X hosts you can use the Docker toolbox, which packages several Docker utilities in addition to the Docker engine (see Recipe 1.6). The Docker toolbox uses a virtual machine running via VirtualBox to act as a Docker host. This machine is called boot2docker. While using boot2docker is now deprecated in favor of the Docker toolbox, we still present a Docker installation using boot2docker in Recipe 1.7.

To round up the installation recipes, we introduce `docker-machine`, a Docker utility that lets you starts a machine in the public cloud of your choice and automatically configures it to be used with your local Docker client. Recipe 1.9 shows you how to do it with the Digital Ocean cloud.

Once you have installed Docker on your favorite target, you are ready to explore the basic commands necessary to create and manage containers. Recipe 1.11 shows you the first steps to run a container, while Recipe 1.13 walks you through the standard life cycle of a container, creating, starting, stopping, killing, and removing containers.