THE EXPERT'S VOICE® IN OPEN SOURCE

# Pro Python System Administration

*Learn to manage and monitor your network, web servers, and databases with Python*

Rytis Sileika

Apress®

# Table of Contents

# Pro Python System Administration

Rytis Sileika

Apress®

**Pro Python System Administration**



ISBN-13 (pbk): 978-1-4302-2605-5
ISBN-13 (electronic): 978-1-4302-2606-2

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1



President and Publisher: Paul Manning
Lead Editors: Duncan Parkes and Michelle Lowman
Technical Reviewer: Patrick Engebretson
Editorial Board: Clay Andres, Steve Anglin, Mark Beckner, Ewan Buckingham, Gary Cornell, Jonathan Gennick, Jonathan Hassell, Michelle Lowman, Matthew Moodie, Duncan Parkes, Jeffrey Pepper, Frank Pohlmann, Douglas Pundick, Ben Reno Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh
Coordinating Editors: Mary Tobin and Jennifer L. Blackwell
Copy Editors: Jim Compton, Heather Lang and Marilyn Smith
Compositor: Lynn L'Heureux
Indexer: Julie Grady
Artist: April Milne
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media, LLC., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licens are also available for most titles. For more information, reference our Special Bulk Sales-eBook Licensing web page at www.apress.com/info/bulksales.



The source code for this book is available to readers at www.apress.com.

*I want to dedicate this book to my family—my wife Evelina and daughters Gabija and Milda*

# About the Author

**Rytis Sileika** has over twelve years of experience in the system administration field. Since obtaining his bachelor of science degree in computer science from Kaunas University of Technology, he's been specializing in system integration and deployment automation. His areas of interest and expertise are UNIX-based operating system management and automation tool development. Rytis is also a RedHat Certified Engineer. He lives with his wife and two daughters in London, United Kingdom. His nonprofessional interests are traveling, hiking, and photography.

# About the Technical Reviewer

**Dr. Patrick Engebreston** obtained his doctor of science degree with a specialization in information assurance from Dakota State University. He currently serves as an assistant professor of computer an network security and works as a senior penetration tester for security firm in the Midwest. His research interests include penetration testing, intrusion detection, exploitation, malware, and programming. He teaches courses in security, C programming, and Python. When not hacking or teaching, Dr. Engebretson spends every waking minute with his wife Lori and his two beautiful girls Maggie and Molly.

# Acknowledgments

I’d like to express my gratitude to everyone at Apress involved in the development and production of this book. First, I want to thank Duncan Parkes, who helped a lot with the initial proposal, set the general shape and structure of the book, and got the whole project moving forward.

Many thanks go to Michelle Lowman and Dr. Patrick Engebretson for correcting all technical and logical mistakes as well as providing valuable tips.

I would also like to thank Jennifer Blackwell and Mary Tobin for keeping the project and my writing on schedule and gently reminding me about the approaching deadlines.

Last but not least, I’d like to thank the Python development community and Guido van Rossum for creating such a nice and elegant programming language.

# Introduction

The scope of the system administrator role has changed dramatically over the years. The number of systems supported by a single engineer has also increased. As such, it is impractical to handcraft each installation, and there is a need to automate as many tasks as possible. The structure of systems varies from organization to organization, therefore system administrators must be able to create their own management tools. Historically, the most popular programming languages for these tasks were UNIX shell and Perl. They served their purpose well, and I doubt they will ever cease to exist. However, the complexity of current systems requires new tools, and the Python programming language is one of them.

Python is an object oriented programming language suitable for developing large-scale applications. Its syntax and structure make is very easy to read, so much so that the language is sometimes referred to as “executable pseudocode.” The Python interpreter allows for interactive execution, so in some situations, you can use it instead of a standard UNIX shell. Although Python is primarily an object-oriented language, you can easily adopt it for procedural and functional styles of programming. Given all that, Python makes a perfect fit as a new language for implementing system administration applications. There are a large number of Linux system utilities already written in Python, such as the Yum package manager and Anaconda, the Linux installation program.

# Prerequisites for This Book

This book is about using the Python programming language to solve specific system administration tasks. We will look at the four distinctive system administration areas: network management, web server and web application management, database system management, and system monitoring. Although I will explain most of the technologies used in this book in detail, bear in mind that the ma goal of this book is to show you the practical application of the Python libraries to solve rather specific issues. Therefore, I'm assuming that you are a seasoned system administrator.

As we go along with the examples, you will be asked to install additional packages and libraries. I most cases, I provide the commands and instructions to perform these tasks on a Fedora system, but you should be ready to adopt these instructions to the Linux distribution that you are going to use. Most of the examples work without many modifications on a recent OS X release (10.6.X) too.

I also assume that you have a background in the Python programming language. I will be focusing on introducing the specific libraries that are used in system administration tasks as well as some less known or less-often-discussed language functionality, such as the generator functions or the class internal methods, but the basic language syntax is not explained. If you want to refresh your Python skills I would recommend *Beginning Python: From Novice to Professional, Second Edition* by Magn Lie Hetland (Apress, 2008).

All examples presented in this book assume the Python version 2.6 and will not work correctly wit the latest Python 3 without additional modifications. Most of the examples rely on additional module that have not yet been ported to this version of Python.

> **Note** Because of the line length limitations of the printed page, some lines of the code had to b split into two lines. This is indicated by the backslash character (\) at the end of the split lin When you use the code examples, you can either leave the structure as it is (i.e., with the wrappe lines), or you can join the two lines together, in which case you'll have to remove the backsla character from the code.

# Structure of This Book

This book contains 13 chapters, and each chapter solves a distinctive problem. Some examples span multiple chapters, but even then, each chapter deals with a specific aspect of the particular problem.

In addition to the chapters, several other organizational layers span this book. First, I grouped the chapters by the problem type. Chapters 1 to 4 deal with network management issues; Chapters 5 to 7 talk about the Apache web server and web application management; Chapters 8 to 11 are dedicated to monitoring and statistical calculations; and finally, Chapters 12 and 13 focus on database management issues.

Second, I am maintaining a common pattern in all chapters. I start with the problem statement and then move on to gather requirements and through the design phase before going into the implementation section.

Third, each chapter focuses on one or more technologies and the Python libraries that provide the language interface to the particular technology. Examples of such technologies could be the SOAP protocol, application plug-in architecture, or cloud computing concepts.

# *Chapter 1: Reading and Collecting Performance Data Using SNMP*

Most network attached devices expose the internal counters via the Simple Network Management Protocol (SNMP). This chapter explains basic SNMP principles and the data structure. We then look at the Python libraries that provide the interface to SNMP-enabled devices. We also investigate the Round Robin database, which is the *de facto* standard for storing the statistical data. Finally, we'll look at the Jinja2 template framework, which allows us to generate simple web pages.

# *Chapter 2: Managing Devices Using the SOAP API*

Complicated tasks, such as managing the device configuration, cannot be easily done by using SNMP because the protocol is too simplistic. Therefore, advanced devices, such as the Citrix Netscaler load balanacers, provide the SOAP API interface to the device management system. In this chapter, we'll investigate the SOAP API structure and the libraries that enable the SOAP-based communication from the Python programming language. We'll also look at the basic logging functionality using the built-in libraries.

# *Chapter 3: Creating a Web Application for IP Address Accountancy*

In this chapter, we will build a web application that maintains the list of the assigned IP addresses an the address ranges. We will learn how to create web application using the Django framework. I'll show you the way Django application should be structured, how to create and configure the application settings and the URL structure. We'll also investigate how to deploy the Django application using the Apache web server.

# *Chapter 4: Integrating the IP Address Application with DHCP*

This chapter expands on the previous chapter, and we will implement the DHCP address range support. We will also look at some advanced Django programming techniques such as customizing th response MIME type as well as serving AJAX calls.

# *Chapter 5: Maintaining a List of Virtual Hosts in an Apache Configuration File*

This is another Django application that we are going to develop, but this time, our focus will be the Django administration interface. While building the Apache configuration management application, you'll learn how to customize the default Django administration interface with your own views and functions.

# *Chapter 6: Gathering and Presenting Statistical Data from Apache Log Files*

In this chapter, our goal is to build an application that parses and analyses the Apache web server log files. Instead of taking the straightforward but inflexible approach of building a monolithic application, we'll look at the design principles of building plug-in based applications. You will learn how to use the object and class type discovery functions and how to perform a dynamic module loading.

# *Chapter 7: Performing Complex Searches and Reporting on Application Log Files*

This chapter also deals with the log file parsing, but this time I'll show you how to parse complex, multiline log file entries. We are going to investigate the functionality of the open source log file parser tool called Exctractor, which you can download from http://exctractor.sourceforge.net/.

# *Chapter 8: A Web Site Availability Check Script for Nagios*

Nagios is one of the most popular open source monitoring systems, because its modular structure allows users to implement their own check scripts and thus customize the monitoring tool to their needs. In this chapter, we are going to create two scripts that check the functionality of a web site. We're going to investigate how to use the Beautiful Soup HTML parsing library to extract the information from the HTML web pages.

# *Chapter 9: Management and Monitoring Subsystem*

This chapter starts a three chapter series in which we'll build a complete monitoring system. The goa of this chapter is not to replace mature monitoring systems such as Nagios or Zenoss but to show you the basic principles of the distributed application programming. We'll look at database design principles such as data normalization. We're also going to investigate how to implement the communication mechanisms between network services using the RPC calls.

# *Chapter 10: Remote Monitoring Agents*

This is the second chapter in the series where we'll implement the remote monitoring agent components. In this chapter, I also describe how to decouple the application from its configuration using the ConfigParser module.

# *Chapter 11: Statistics Gathering and Reporting*

This is the last part of the monitoring series, where I'll show you how to perform basic statistical analysis on the collected performance data. We're going to use scientific libraries—NumPy to perform the calculations and matplotlib to create the graphs. You'll learn how to find which performance readings fall into the comfort zone and how to calculate the boundaries of that zone. We'll also do the basic trend detection, which provides a good insight for the capacity planning.