TURING 图灵程序设计丛书

# Working with Unix Processes

# 理解Unix进程

【加】 Jesse Storimer 著　门佳 译

- 唯一一本针对Web开发人员的Unix编程书籍
- 无需借助C语言即可玩转Unix进程
- 自主编写并调试高效服务器

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

# Working with Unix Processes

# ——理解——Unix进程

【加】Jesse Storimer 著　门佳 译

人民邮电出版社
北　京

**图书在版编目（CIP）数据**

理解Unix进程 /（加）斯托里默（Storimer, J.）著；门佳译. -- 北京 ：人民邮电出版社，2013.6
（图灵程序设计丛书）
书名原文：Working with Unix Processes
ISBN 978-7-115-31689-9

Ⅰ. ①理… Ⅱ. ①斯… ②门… Ⅲ. ①UNIX操作系统
Ⅳ. ①TP316.81

中国版本图书馆CIP数据核字(2013)第079983号

## 内 容 提 要

本书从Unix编程的基础概念着手，采用循序渐进的方法，详细介绍了Unix进程的内部工作原理。本书提供的许多简单而强大的技术，能够帮助Web开发人员深入了解Unix系统的并发性、守护进程、生成进程（spawning process）与信号等。同时，读者也可以使用这些技术和方法编写并调试自己的服务器。此外，本书附录部分也涉及了一些流行的Ruby项目，让读者进一步了解如何巧妙运用Unix进程。

本书适合Unix程序员、Web开发人员阅读。

图灵程序设计丛书

**理解Unix进程**

◆ 著　[加] Jesse Storimer
译　门　佳
责任编辑　丁晓昀
执行编辑　陈婷婷

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街14号
邮编　100061　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京铭成印刷有限公司印刷

开本：880×1230　1/32

◆ 印张：4
字数：97千字　2013年6月第1版
印数：1－3 500册　2013年6月北京第1次印刷

著作权合同登记号　图字：01-2012-7762号

ISBN 978-7-115-31689-9

定价：29.00元

**读者服务热线：(010)51095186转604　印装质量热线：(010)67129223**

**反盗版热线：(010)67171154**

# 版权声明

# 致　　谢

我要向几位了不起的人致以深深的谢意，他们阅读了本书初稿，让我知道如何宣传好此书，在需要时给我动力，在各方面都给予我莫大的帮助。他们是 Sam Storry，Jesse Kaunisviita，以及 Marc-André Cournoyer。

我也要向妻子和女儿表达无限的感激，你们不仅帮助我度过写书这段飘忽不定的日程安排，而且时刻陪伴着我，为我提供各种参考意见。没有你们的爱和支持，我无法做到这一切。是你们让这一切更有意义！

# 目　录

第 1 章

# 引　言

从孩提时起，只要一有机会我就会坐在计算机前。倒不是为了写程序，而是为这台神奇的机器所能做的事而着迷。于是我成为了一名使用 ICQ、Winamp 和 Napster 的计算机用户。

长大后，我将更多的时间耗在了计算机的电子游戏上。起初是第一人称射击游戏，后来大部分时间都在玩即时战略游戏。再后来，我发现这些游戏竟然可以在线玩了！我年轻的时候就是个“computer guy”（计算机小子）：知道如何使用计算机，但对于计算机的工作原理却一无所知。

之所以告诉你我的经历，是因为想让你知道我并不是什么神童。我没有在七岁的时候自学 Basic 语言编程，而当我开始学编程时，也没能反客为主去指点老师并纠正他的错误。

直到大二那年，我才真正爱上了编程这种活动。也许有人会说我这是大器晚成，可是我觉得自己其实比你想象的还要普通。

尽管出于编程本身而热爱编程，但我对于计算机工作原理的理解仍然不够深入。如果那时候你告诉我说我所有的代码都在一个进程中运行，我肯定会将信将疑地对你另眼相看了。

幸运的是，我在附近一家互联网初创公司谋到一份不错的差事。这让我有机会在真正的生产系统上从事一些编程工作，给我带来了全新的变化，让我有理由去学习事物究竟是如何运作的。

在这个高流量的生产系统上工作时，我碰到的问题越来越复杂。随着流量和资源需求的增长，我们不得不梳理软件的方方面面来调试和修复未解决的问题。单靠浏览应用程序的代码并不能让我们洞悉程序运作的全景。

我们的应用程序面对的东西可是不少：防火墙、负载均衡器、反向代理，还有 http 缓存。除了应用程序之外也还有很多层：作业队列、数据库服务器，以及统计收集器。每一个应用的组成部分都不尽相同，这本书也不会去教你所有的这些细节。

本书将讲解 Unix 进程方面所有你需要知道的知识，保证会增进你对应用程序任何一部分组件的理解。

托程序调试的福，我不得不深入研究了一些采用 Unix 编程概念的 Ruby 项目，例如 Resque 和 Unicorn。正是这两个项目引导我开始用 Ruby 进行 Unix 编程。

对工作原理有了深入的了解之后，我不仅可以更快地理解并诊断出现的问题，还能够对那些单靠查看代码依然让人摸不着头绪的难题进行排查。

凭借在这些项目中学到的技术，我甚至想到了一些更快、更有效的新方法来解决手头上的问题。好了，关于我的事儿已经说得够多了，让我们开始漫游 Unix 仙境吧。

第 2 章

# 基础知识

本章介绍了全书涉及的重要概念的相关知识。在深入主要章节之前，建议你先阅读该部分内容。

## 2.1 干嘛要在意？

自 1970 年起，Unix 编程模型就已经以某种形式存在了。当时，Unix 在贝尔实验室闪亮问世，随之一起诞生的还有 C 程序设计语言，或曰 C 程序设计框架。在随后的数十载中，作为一款可靠、安全和稳定的操作系统，Unix 经受住了时间的考验。

Unix 编程理念及技术并非一时之风，亦不是新近流行的编程语言。它们已超越了编程语言。无论你是使用 C、C++、Ruby、Python、JavaScript、Haskell，还是自己钟意的其他语言，这些技术都有用武之地。

Unix 编程模型已经存在了数十载之久，且大部分都没有改变。过去 40 年里，聪慧的程序员们一直在 Unix 编程模型中借助多种编程语言来解决各类难题，在接下来的 40 年里，他们仍将一如既往。

## 2.2 驾驭神力!

现在我要提醒你，本书所讲述的概念和技术将赐予你强大的力量。有了它，你能够编写出新的软件，理解现有的复杂软件，甚至能将你的职业生涯提升到新的高度。

请记住：能力越大，责任也越大。在阅读的过程中，我会详细告诉你如何获取这种能力，并避开各种陷阱。

## 2.3 概述

本书并非参考手册，它更像是一份攻略。因为每一章内容都是基于之前的章节，所以为了最有效地利用本书，你应该按章节顺序逐一读下去。读完全书后，你可以利用章节名来查找有关信息，温故而知新。

书中包含了大量的代码示例。我强烈建议你在 Ruby 解释器中逐一运行这些代码。自己动手调试有助于更深入地理解概念。

一旦读完全书并把玩过那些示例，你肯定想要接触一些更具深度的实际项目。那时，你可以看看书中提到的 Spyglass 项目。

Spyglass 是一个专门为本书编写的 Web 服务器，旨在传授 Unix 编程概念。它采用了你在这里学习到的各种概念，并展示了如何将其应用于真实的项目之中。有关详细介绍请参阅本书最后一章。

## 2.4 系统调用

要理解系统调用，首先需要了解 Unix 系统的组成，具体来说就是用户空间（userland）和内核。

Unix 系统内核位于计算机硬件之上，它是与硬件交互的中介。这些交互包括通过文件系统进行读/写、在网络上发送数据、分配内存，以及通过扬声器播放音频。鉴于它这些强大的能力，程序不可以直接访问内核，所有的通信都是通过系统调用来完成的。

系统调用为内核和用户空间搭建了桥梁。它规定了程序与计算机硬件之间所允许发生的一切交互。

所有的程序都运行在用户空间。就算是不借助系统调用，你的用户空间程序仍旧能做不少事情：数学运算、字符串操作，以及使用逻辑语句控制程序流程。不过我敢说，如果你打算让程序做些有意思的事情，那就还是得通过系统调用来使用内核。

如果你是一名 C 程序员，那么这些内容你可能已经驾轻就熟了。系统调用可谓是 C 编程的核心。

我估计你像我一样毫无 C 编程经验。你学习的是高级语言编程。在你学会将数据写入文件系统的时候，对于使用了哪些系统调用却并不知情。

总地来说，系统调用允许你的用户空间程序通过内核间接地与计算机硬件进行交互。在本书接下来的章节中，我们会考查一些常见的系统调用。

## 2.5 命名法，wtf(2)

学习 Unix 编程的障碍之一就是不知道从哪里查找合适的文档。想不想听点出人意料的回答？其实所有东西都包含在 Unix 手册页（manpages）中了。如果你在用 Unix 系统，那么这些东西都已经在你的计算机里了。

要是你以前压根就没有用过手册页，那么你可以在终端中输入命令 `man man` 进行查看。

不错吧？嗯，还算凑合吧。系统调用 API 的手册页在以下两种情况中是很不错的资源。

(1) 你是一名想知道如何使用特定系统调用的 C 程序员。
(2) 你想搞明白某个系统调用的用途。

这里假设我们都不是 C 程序员，所以第一种情况就没什么用了，但是第二种情况却是相当受用。

你在整本书中都会看到类似于 select(2)这样的引用。它告诉你可以从哪里找到特定系统调用的手册页。你可能有所不知，Unix 手册页包含了很多节（section）。

以下是 FreeBSD 和 Linux 系统手册页中最常用的节：

- 节 1：一般命令
- 节 2：系统调用
- 节 3：C 库函数

❑ 节 4：特殊文件

因此，节 1 专门用于一般命令，也就是 shell 命令。如果我希望你参考 `find` 命令的手册页，我就会这样写：find(1)。这就是说在手册页的节 1 你可以找到 `find` 的使用说明。

如果我希望你参考 `getpid` 系统调用的手册页，我就会这样写：getpid(2)。这就是说在手册页的节 2 你可以找到 `getpid` 的使用说明。

> 为什么手册页还要分这么多节？因为一个命令可能在不止一节中出现，也就是说它既可以作为shell命令，也可以作为系统调用。
>
> 例如stat(1)和stat(2)。

要查看手册页的其他节，你可以像这样在命令行上输入：

```
$ man 2 getpid
$ man 3 malloc
$ man find # 等同于 man 1 find
```

这种记法并非本书所独创，而是大家参照手册页时通用的一套惯例[①]。所以你最好还是从现在开始学习并适应它。

## 2.6 进程：Unix 之本

进程乃 Unix 系统的基石。为什么要这么说？因为所有的代码都是在进程中执行的。

① http://en.wikipedia.org/wiki/Man_page#Usage