The Linux Programmer's Toolbox

Linux 开发工具箱

——项目开发的最有效途径

(美) John Fusco 著
贾严磊 董西广 王在奇 译

PEARSON Education
清华大学出版社

Linux 开发工具箱

——项目开发的最有效途径

(美) John Fusco 著

贾严磊 董西广 译

王在奇

清华大学出版社

北 京

北京市版权局著作权合同登记号 图字：01-2007-2720

图书在版编目(CIP)数据

Linux 开发工具箱——项目开发的最有效途径/(美) 法斯克(Fusco, J.) 著；贾严磊，董西广，王在奇 译.
—北京：清华大学出版社，2008.9
书名原文：The Linux Programmer's Toolbox
ISBN 978-7-302-17786-9
Ⅰ.L… Ⅱ.①法…②贾… ③董… ④王… Ⅲ.Linux 操作系统—程序设计 Ⅳ.TP316.89
中国版本图书馆 CIP 数据核字(2007)第 078423 号

责任编辑：王 军 徐燕萍
装帧设计：孔祥丰
责任校对：成凤进
责任印制：李红英
出版发行：清华大学出版社 地 址：北京清华大学学研大厦 A 座
http://www.tup.com.cn 邮 编：100084
社 总 机：010-62770175 邮 购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京四季青印刷厂
装 订 者：三河市新茂装订有限公司
经 销：全国新华书店
开 本：185×230 印 张：30.75 字 数：670 千字
版 次：2008 年 9 月第 1 版 印 次：2008 年 9 月第1 次印刷
印 数：1～4000
定 价：58.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：(010)62770177 转 3103 产品编号：025482－01

序

如果您已经掌握了使用 Linux 的基本操作，比如会使用 ls、grep、find 和 sort 这样的命令，而且还是一名 C 或 C++程序员，知道如何使用 Linux 系统调用，就应该明白除了能使用鼠标“单击”之外，Linux 还能给自己的生活带来更多的方便——只是现在还不知道怎样做到而已，所以您会问自己：“我该如何去做呢？”

本书会帮您回答这个问题。作者以其广博的知识，向那些非 Linux 初学者展示了如何攀越下一阶段并走向精通的求知之路。

从用于调试和性能分析的命令行工具，到/proc 目录下的文件列表，本书将会向您介绍这些Linux 行家里手所熟练使用的各种工具，从而使您在日常生活中更方便、更有效地使用 Linux。

本书除了告诉您很多的“是什么”(什么工具，什么选项，什么文件)，还会告诉您“为什么”。本书会解释它们的运行原理，使您能知其然并知其所以然，最终将 Linux(还有 UNIX)的精华了然于胸。

本书是献给 Linux 程序员的一席饕餮大餐，希望您能用心品味，至少我是这么做的。

开始享受吧！

丛书主编

Arnold Robbins

前　言

Linux 从不缺乏工具，但其中的许多工具都是从 UNIX 那里继承而来的，而这些以两个字符命名的工具总令人匪夷所思，并勾起人们对最早通过纸带打孔来编写程序的那段艰辛岁月的回忆：那时，程序员总是试图用最少的字符给变量命名以节省纸带上的存储空间。值得高兴的是，那些日子已经一去不复返了，只留下了这些历史的见证。

许多这些早期遗留下来的工具现在依然非常有用，而且大部分还相当专业。每个工具虽然可能只执行一个任务，但却能完成得非常漂亮。非常专业的工具往往都有很多选项，以满足用户更具体的要求。假设您是第一次使用 grep，而且知道什么是正则表达式，但或许您还不能熟练掌握正则表达式的句法(别担心，其他人也没有掌握)，这没关系，因为很好地使用 grep 工具不一定非要成为精通正则表达式的高手。

从这本书中我希望您知道，有很多工具只要能够使用就可以了，没有必要精通它们。不必在有效地使用它们之前花费大量的时间去阅读帮助手册。我希望您能发现一些自己也许并不熟悉的新工具。本书中提到的一些工具有些是较早版本的，有些是最新版本的。所有这些工具都非常有用。随着对每个工具的深入学习，您会发现它的更多用途。

在本书中使用的“工具”一词是广义的。对我而言，创造工具与使用工具同等重要，所以本书包含了各种 API 用法的介绍，这在其他书籍的细节中通常没有涉及过。另外，该书还提供了 Linux 内核工作机制的一些背景知识，这将有助于您对一些工具的理解。本书从一个独特角度——用户角度来介绍内核，您将会找到足够的信息来理解内核为每个进程设置的底层规则，不必为此而去读内核源代码。

本书中没有拼凑的 man 页面或拼接成文本的其他文档。GNU 和 Linux 开发人员做了大量工作将他们做的工作文档化，但对于不熟练的用户来说，很难找到那些文档。与其花时间看那些过时的再版文档，不如了解一些查找最新文档的新方法。

虽然 GUN/Linux 文档非常丰富，但往往其可读性不强。如果为了熟悉一种工具而去读 10 000 字的帮助文档，却仍然对该工具的功能和用法毫无头绪，这显然是徒劳的。这就是本书要解决的问题。本书将解释怎样使用一种工具，以及为什么要使用它，并尽可能提供简明扼要的例子，您可以亲自输入并修改，以加强对这些工具和 Linux 本身的理解。

本书提到的所有工具都是免费使用的。大多数是由标准 Linux 版本所提供的，对于那些未提供的工具，书中已经给出了它们的 URL，您可以自行下载。

这些资料会被尽可能的设置得读起来更有趣味性。

本书读者对象

本书写给那些希望成为更有效率的，并且对 Linux 编程环境有更深入理解的中高级 Linux 程序员。如果您是一位有经验的 Windows 程序员，且在 Linux 环境下游刃有余，那么这本书也同样适合您。

非程序员读者也可以从中受益，因为这里涉及到的许多工具和主题不只局限于编程应用。如果您是一位系统管理员，或者只是一位 Linux 爱好者，那么这本书同样也有您需要的信息。

本书目的

我编写本书的目的是将它作为我曾为 *Linux Journal* 所撰写的一篇名为“Ten Command Every Linux Developer Should Know”的后续篇章。文章的灵感来源于我本人作为一名 Linux 程序员的经验。在我的日常工作中，我努力做到了每天花一些时间学习新知识，即使这意味着当前工作的短暂停滞，但我却从中受益颇多。让我感到惊讶的是，有很多次我学习了一种工具但最终却证明这种工具是没有用的，这促使我后来试图寻找立刻使用它的方法。这也成为我不断学习的强大动力。希望您通过学习本书，能够以我为范例来加强对基本规则技巧的掌握。

学习这些知识纯粹是一种乐趣。如果您跟我一样，以 Linux 学习为乐，那么促使自己学习更多的知识是没有问题的。因为 Linux 是开源码，您可以有机会了解到它所有的内部工作机制，而这对于像 Windows 一样的封闭源代码环境是不可能的。本书会介绍一些可以免费使用的资源来帮助您获取更多的知识。

如何阅读本书

每章的内容都可独立成章。后面章节的学习可能会用到前面章节的一些背景知识。只要可能，书中会交叉引用一些可以帮助您查找必要背景信息的资料。

示例是最好的老师，所以本书尽可能地提供简单的例子。建议您能够尝试实践这些例子。

本书结构

第 1 章　开源工具的下载和安装：包括发行开源码使用的机制，讨论了不同版本使用的多种软件包格式以及它们各自的优缺点，并介绍了一些维护软件包的工具以及使用方法。

第 2 章　从源代码构建：涉及到构建开源项目的基本知识，介绍了一些用来创建软件的工具以及合并工具，本章还提供了一些熟练使用 make 的方法和技巧。还演示了如何配置同 GNU 的 autoconf 工具一同发行的项目，以使用户可以定制它们来满足要求。最后，讲解了许多程序员经常误解的构建阶段，并说明了一些很可能碰到的错误和警告，以及如何解释它们。

第 3 章　查找帮助：介绍了您可能不太了解的 Linux 版本中的各种文件格式，还介绍了用来获取这些格式的工具，并讨论了使用它们的有效方法。

第 4 章　编辑和保存源文件：讨论了程序员使用的各种文本编辑器以及它们各自的优缺点，并介绍了每个程序员都应注重的并用来衡量编辑器优劣的一些特性。本章也涉及到版本控制的基本知识，这对于软件项目管理是非常关键的技术。

第 5 章　开发者必备内核知识：从用户角度来介绍内核。本章中提供了理解 Linux 系统的工作机制所需的必备背景知识，并介绍了几个允许用户查看代码如何影响内核的工具。

第 6 章　进程：重点讲解进程、进程的特性以及如何管理进程。本章介绍了引入工具以及理解它们为什么有用所需的大量背景知识。另外，本章还介绍了几个编程 API，可以用来创建自己的工具。

第 7 章　进程通信：介绍了进程间通信(IPC)的一些概念。本章包含了第 8 章所需的大部分背景知识。对于每个 IPC 机制，介绍了需要在运行实例环境下使用它的 API。

第 8 章　使用 shell 命令调试 IPC：介绍了一些用来调试使用 IPC 的应用程序的工具。基于第 7 章中的内容，有助于解释这些工具的输出信息，这些信息是难以理解的。

第 9 章　性能优化：介绍了衡量系统性能及个人应用程序性能的工具。本章通过一些示例说明编程设计是如何影响性能的，还讨论了多核处理器所特有的几个性能问题。

第 10 章　调试：介绍了一些可以用来调试应用程序的工具和技术。并介绍了一些包括 Valgrind 和 Electric Fence 在内的开源内存调试工具，而且深入讲解了 gdb 的性能以及如何有效地使用它。

目　录

Chapter 1

开源工具的下载和安装

1.1 简介

本章介绍了自由软件不同的发行版本，以及怎样熟练掌握它们和找到它们的途径，并详细地说明了存档文件和包文件，以及用于管理它们的最常见的工具命令。

不明来源的软件可能存在风险，这里也涉及到一些安全方面的问题，您应该有意识地采取一些措施来保护自己。本章介绍了验证和信任机制的概念，说明了如何将它们应用到安全机制中，并介绍了当无法通过验证时，如何查看软件包和存档文件。

最后，介绍了管理软件包的工具以及有效使用它们的方法。

1.2 什么是开放源码

开源是指自由软件的行业术语，由 Open Source Initiative[1] (OSI，开放源码促进会、开放源码组织)首创。这个组织由 Richard Stallman 创建，旨在推动以 GNU 项目为源的自由软件原则。它的目标之一是反对那些消极的墨守成规的自由软件，促进源代码的免费共享。

起先，很多商家不敢使用开源软件。毋庸置疑，一些大型软件公司的市场部门曾对它做出改动。俗语说："天下没有免费的午餐"，一些人担心这种许可证(像 GNU 公共许可证)如同一个病毒，如果他们在项目中使用了开源软件，那么他们的项目也要将源代码公之于众。

值得庆幸的是，大多数这样的担心慢慢减少。许多大型开发商在自己的项目中自由使用并改进开放源代码，有些甚至全部基于开源软件。精灵现身于世了。

1.3 开放源码的意义

对于大多数人来说，开源软件只不过是许多自由使用的高质量软件罢了。遗憾的是，现在使用的许多软件并不都是高质量的，但却是必不可少的一部分。好的项目理念日益繁茂，欣欣向荣；差的却日渐枯萎，郁郁而终。获取开源软件有些像摘水果：要靠经验才能知道它是否成熟。

优胜劣汰的自然选择规律无处不在。在源码水平上，选择具有不同特征的代码(基于补丁程序)以插入最优的代码。作为一个消费者，会选择若干项目来下载，以增强项目的生命力。没人会为无人使用的项目开发代码，下载的越少越不能吸引开发人员，下载的越多意味着有更多的开发人员使用它，进而意味着能可以从中选择更多的代码，从而得到更好的代码。有时选择一个开发项目就像一场赌博，是拿时间和为之付出的努力作赌注，不可避免会做出让人后悔的选择，但要记住："这就是过程"。

对于一些人来说，不清楚自己获得的是什么或许也是一种乐趣，就和打开生日礼物时一样。对其他人来说，这非常麻烦，而且浪费时间。假如要寻找能够即装即用的现成软件，这也有用户所需要的开源软件——只是数量有限而已。幸运的是，因特网上有很多资源可以帮助用户做出不错的选择。

1.3.1 搜索工具

在网上开始搜索之前，首先要看一下 CD 的发行版本。假如是从套装 CD 或 DVD 安装的 Linux，那么一定有许多未安装的工具。大多数发行的 CD 版本比默认安装的版本多一些软件，举例来说，当安装操作系统时，会提示选择安装种类，这可能会导致用户随意地安装软件包，

[1] www.opensource.org。

系统主要是基于“工作站”或“服务器”的角度。

可以手动地从安装光盘寻找原始软件包，然后将它们添加到安装软件中。缺点是软件包并没有按特定的顺序进行分类，所以必须知道所要找到的软件名称。一些发行版本采用图形界面将软件包分类，以帮助用户选择要安装的软件。

假如不知道要查找的具体名称，可以使用因特网。一些网站为开源软件提供了类似票据交换场所的服务，www.freshmeat.net 就是其中一员。在那里会发现软件是分门别类的，所以很容易就可以找到所需的软件。例如，我在著书过程中，从 Freshmeat 上搜索“word processor”这个词，结果找到了 71 个相关的内容。可想而知，从 71 个不同的结果中选择是非常困难的。

Freshmeat 允许对结果进行筛选以缩小选择范围。我的搜索结果中包含除了 Linux 和处于开发阶段的项目外的多种操作系统。所以我选择将搜寻范围限定在由 Linux 支持的成熟的使用开放式系统互连参考模型(OSI)的开源认证的那些项目中(Freshmeat 的搜索结果中也包括商业软件)，这样就减少到只有 12 项了，更容易搜索。给定的 word processor 这个术语具有宽泛的解释，更进一步地查找表明仍有一些项目不是我们要找的。再次过滤之后，我才找到一些著名的高质量的项目，如 AbiWord，还有一些我以前从来没听说过的。但仍有一些没有出现的，如 OpenOffice，我也将其写入这本书中。后来证明，我找不到 OpenOffice 是因为它是属于 Office/Business::Office Suites，而不是 word processor。这件事使我明白，如果没找到想要的信息，不是因为不存在，而是因为没找到，所以继续寻找肯定会找到。

1.3.2　版本格式

既然已经找到了所感兴趣的软件，那么可能会有更多的选择。成熟的项目通常提供一种或多种版本的准备安装的软件包，只有较少的仅在存档文件中提供源代码和二进制文件。通常这对用户来说是个很好的提示。下载软件包可能就像是买新车，不必知道它是怎样工作的，只需要转动钥匙，然后起动即可。相反，下载源码或二进制文件的存档文件就像买旧车，如果对车有所了解，那么会好一些；否则，就无所适从了。

通常，当一个项目提供一个安装包时，标志着这个项目已经成熟，也标志着该项目的发行周期日趋稳定。如果这个项目每周都发布新的版本，那它很可能不用担心打包的问题。一个软件包文件，如果运气好的话，也许能安装并运行它，但即使是一辆新车，也可能偶尔有瑕疵。

存档文件是软件包文件的一类，它适用于基于 Linux 的项目，通常是存档的 tar 文件。存档文件是存档成单个文件的文件集合，它使用像 tar 命令一样的存档工具。这些文件经常使用 gzip 程序命令进行存档以节省空间，通常作为 tar 文件或 tarballs 被提及。

tar 文件是项目发布源码的首选格式，它们容易创建和使用，而且每个程序员也都熟悉 tar 程序。有时会发现 tar 文件中包含二进制可执行代码，这对打包来说是个快捷但却不理想的选

择，应该避免使用这种方法，除非清楚地知道自己在做什么。总的来说，tar 文件适用于对编程和系统管理有所了解的人群。

1.4 存档文件

在下载和安装开源软件的过程中，可能会遇到各种各样的存档文件。存档文件是包含其他文件集合的文件。如果您是一个 Windows 用户，肯定非常熟悉占主导地位的 Windows 的存档文件 PKZip。Linux 的存档效用函数与之非常类似，而与 PKZip 不同的是，它们不包括压缩。Linux 的存档工具专注于存档，而将压缩交给另一种工具(典型的是 gzip 或 bzip2)，这就是 UNIX 的思想。当然，因为在 Linux 下可以选择多种存档文件，但作为开源的使用者，必须取之所用。所以即使您可能只遇到 tar 文件，但至少也应知道其他工具也是可用的。

存档工具除了要保存文件名及数据外，还有一些特殊的要求。除了文件的路径名及数据之外，存档工具还需要保存文件的元数据。元数据(metadata)包括文件的所有者、组别，以及其他属性(如读/写/执行权限)。存档文件记录了所有诸如此类的信息，比如从文件系统中删除文件，然后从存档文件中无损地恢复。假如您存档了一个可执行文件，然后把它从文件系统中删除了，那么当要恢复时，它应该仍是可执行的。在 Windows 中，文件名通过它的扩展名(如.exe)表明它是否是可执行文件。而 Linux 使用元数据表明文件是否是可执行的，所以要保存用存档文件存储的这些数据。

在 Linux 中最常用的存档工具如表 1-1 所示。到目前为止，最流行的存档格式就是 tar。tar 是 tape archive 的缩写，它来源于磁带备份功能。现在，tar 已经成为用来存档批量文件的一种最常见的通用工具。使用 tar 工具，有时会遇到 cpio，它使用一种截然不同的语法来完成同样的功能。POSIX(可移植 UNIX 操作系统接口标准)标准存档效用函数 pax 可以识别 tar 文件、cpio 文件或者它自己的格式文件。我从未见过 pax 格式的发行版本，提到它只是为了论述的完整性。

最后值得提及的存档工具是 ar，它常用来创建软件开发所使用的目标代码库，但也可用来创建 Debian 版本的软件包文件。

表 1-1 常用的存档工具

工　具	说　明
tar	最流行的工具
cpio	RPM 格式内部使用的工具；应用不广泛
ar	Debian 软件包内部使用的工具，只有软件开发库使用。ar 文件没有路径信息

也可以找到一些用来处理由 PKZip 创建的 zip 文件的工具，以及一些不常见的压缩存档工具，如 lha。然而，Linux 开源程序不会以这些格式发行。如果见到.zip 存档文件，它也一定是为微软操作系统准备的。

在大多数情况下，对于每种版本格式，需要知道两点：如何查询存档文件的内容以及如何从存档文件中提取文件。Linux 存档文件更注重基本功能，它并不像 Windows 的存档文件，有各种危险的附加功能。所以，从存档文件中查询和提取文件大体上是安全的，尤其当自己不是超级用户的时候。在提取文件之前查询存档文件是个明智的选择，这样，就不会不小心修改了系统中的同名文件。

1.4.1　识别存档文件

当从网上下载了一个存档文件时，它很可能因节省带宽而被压缩了。有许多存档文件的命名规则，其中的一部分如表 1-2 所示。

表 1-2　部分存档工具命名规则

扩　展　名	类　　型
.tar	tar 存档文件，未压缩
.tar.gz.tgz	tar 存档文件，使用 gzip 压缩
.tar.bz2	tar 存档文件，使用 bzip2 压缩
.tar.Z. taz	tar 存档文件，使用 UNIX 的 compress 命令压缩
.ar .a	ar 存档文件，通常只用于软件开发
.cpio	cpio 存档文件，未压缩

如果不确定文件的存档类型，就记住使用 file 命令。当文件名不能提供什么信息时，使用这个工具能够确定文件类型。当网络浏览器或其他工具使得文件名无法识别时，它也是非常有用的，例如，假设有一个存档的 tar 文件，文件名为 foo.x，文件名不能反映文件的内容，那么试一下下面的命令：

```
$ file foo.x
foo.x: gzip compressed data, from UNIX, max compression
```

现在可以知道它是用 gzip 存档的，但仍然不确定它是否是 tar 文件，可以用 gzip 将其解压缩，然后再次使用 file 命令，或者使用该命令的-z 选项。

```
$ file -z foo.x
Foo.x: tar archive (gzip compressed data, from UNIX, max compression)
```

现在可以确定它是 tar 文件了。