

[Linux]
实·用·技·术·丛·书

Linux与 X Window 系统基础

邓增涛 黄超 编著
童寿彬 审校



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

Linux 实用技术丛书

Linux 与 X Window 系统基础

邓增涛 黄 超 编著
童寿彬 审校

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

Linux 和 X Window 与 DOS 和 Windows 属于不同类型的系统。DOS 和 Windows 属于典型的商业系统,而 Linux 和 X Window 系统则属于开放和自由的系统,它们之间有许多本质的区别。这些区别使 Windows 用户在转向学习 Linux 时会遇到一定的困难。本书针对开放系统的特点,从系统总体出发,重点说明各种 Linux 发行版本的共性部分,其中包括系统发展史、系统各组成部分的相互关系、内核的主要功能、文件系统、主要的编辑工具、命令行界面、网络和图形用户界面等,特别对 X Window 系统进行了深入的说明。

本书适合初步掌握 Linux 安装、配置和使用的读者,帮助他们深入掌握 Linux 和 X Window 系统的总体概念和系统各组成部分之间的相互联系。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,翻版必究。

图书在版编目(CIP)数据

Linux 与 X Window 系统基础/邓增涛等编著. —北京:电子工业出版社,2002.1

Linux 实用技术丛书

ISBN 7-5053-7305-6

I. L… II. 邓… III. ①Linux 操作系统 ②Linux 操作系统—软件包, X Window IV. TP316.81

中国版本图书馆 CIP 数据核字(2001)第 089748 号

丛 书 名: Linux 实用技术丛书

书 名: Linux 与 X Window 系统基础

编 著 者: 邓增涛 黄 超

审 校 者: 童寿彬

责任编辑: 毛兆余

排版制作: 电子工业出版社计算机排版室

印 刷 者: 中国科学院印刷厂

装 订 者: 三河市路通装订厂

出版发行: 电子工业出版社 <http://www.phei.com.cn>

北京市海淀区万寿路 173 信箱 邮编 100036

经 销: 各地新华书店

开 本: 787×1092 1/16 印张: 29.25 字数: 748.8 千字

版 次: 2002 年 1 月第 1 版 2002 年 1 月第 1 次印刷

书 号: ISBN 7-5053-7305-6
TP·4195

印 数: 3 000 册 定价: 48.00 元

NJ5222/06

凡购买电子工业出版社的图书,如有缺页、倒页、脱页、所附磁盘或光盘有问题者,请向购买书店调换;
若书店售缺,请与本社发行部联系调换。电话 68279077

前 言

Linux 和 X Window 与 DOS 和 Windows 属于不同类型的系统。DOS 和 Windows 属于典型的商业系统,全部由微软公司独家开发,并且系统组成部分是单一的。Linux 属于开放和自由的系统,它不仅有许多不同的发行版本,而且系统组成部分也是多样化的。这种差异在用户界面(命令行界面和图形用户界面)上显得更为突出。这些使 Windows 用户在转向学习 Linux 时会遇到一定困难。

本书针对开放系统的特点,从系统总体出发,重点说明各种 Linux 发行版本的共性部分,其中包括系统发展史、系统各组成部分的相互关系、内核的主要功能、文件系统、主要的编辑工具、命令行界面、网络和图形用户界面等;特别对 Linux 系统的图形用户界面——X Window 系统,作了详细的说明。

第 1 章说明 UNIX 系统、GNU 系统和 Linux 内核的发展史和三者之间的关系,以及 UNIX 类操作系统的总体框图,特别是 X Window 系统和 UNIX 类操作系统之间的关系。

第 2 章说明硬件、内核和系统软件之间的关系,内核的主要功能。内容包括:掌握 Linux 内核所必要的基本知识、内存管理、进程调度、进程间通信、外围设备互连、中断和中断处理以及设备驱动程序。重点使读者对内核的功能有全面的了解。

Linux 的强大功能是通过 Shell 发挥出来的。Shell 既是命令解释程序,又是编程语言。它既可以通过单独的合作实现简单的功能,又可以通过命令的组合实现复杂的功能。事实上,它相当于第 4 代语言。一条 Shell 命令完成的功能相当于 10 行 C++ 语句或者 100 行 C 语句完成的功能。它可以实现自动管理、复杂的信息提取和简单的数据库管理功能。因此,可以用 Shell 快速建立应用软件的雏型。掌握和用好 Shell 是充分发挥 Linux 功能的关键。Linux 的命令由 Shell 本身和各种实用程序提供,我们用第 3 章说明一些基本命令的使用。用第 6、第 7 和第 8 章说明 Shell 的基础知识、Shell 编程、各种主要的 Shell 和它们之间的差异。

Linux 的文件系统有它自己的特色。通过虚拟文件系统,可以将多达 32 种不同的文件系统有机地组合在单一的目录树中。这是它适合于异种机联网的基础。我们用第 4 和第 5 两章说明虚拟文件系统各种主要的文件系统、目录树结构以及它们是如何在磁盘中实现的。

建立、修改和删除文件是由各种编辑软件实现的,它们是编程必不可少的工具。第 10 章综合说明 3 种编辑软件的使用,其中包括流编辑程序 sed,屏幕编辑软件 vi 和 Emacs。

TCP/IP 是 UNIX 类操作系统内核的功能,这使得它们在联网能力上比其他操作系统占有优势。Linux 使用了最快的 TCP/IP 网络驱动程序,这使得它的联网能力比同类操作系统更胜一筹。第 9 章介绍 Linux 的联网能力。

作为 UNIX 类操作系统共同的图形用户界面,X Window 系统建立在 TCP/IP 基础之上。它按照客户机/服务器模式工作。服务器和客户程序(X 应用程序)是分离的,它们既可以在同一台机器上工作,也可以通过网络在不同的机器上工作。不仅如此,窗口管理程序和桌面环境又是单独的客户程序。这些使 X Window 系统比 Windows 更复杂,更灵活,功能更强大。本书用第 11 到第 16 章的内容来说明 X Window 系统,其中包括 X Window 系统的基础知识,X 服务器

软件(XFree86)及其配置,两种主要桌面环境 KDE 和 GNOME,KDE 应用程序,用 X 资源定制客户程序和定制窗口管理程序。

本书适合从 Windows 转向学习 Linux 系统的读者。在初步掌握 Linux 系统的安装、配置和使用后,帮助读者从系统总体出发,深入掌握和运用 Linux 和 X Window 系统。

本书主要由邓增涛、黄超编写。在编写过程中,得到了陈琪、李晓、黄成、杨德昌、周兆确、叶青、方可燕、黎加佳、章海、徐进民、邓增荣、王筑、黎洁、陈菊周、王涛、黄正华等同志的大力支持,他们在资料的搜集与整理方面做了大量工作,在此对他们表示衷心的感谢。

作 者

目 录

第 1 章 UNIX,GNU 与 Linux	(1)
1.1 UNIX 系统	(1)
1.2 GNU 系统.....	(5)
1.3 Linux 系统	(9)
1.4 小结	(12)
第 2 章 Linux 内核的主要功能.....	(14)
2.1 基础知识	(14)
2.2 内存管理	(20)
2.3 进程	(30)
2.4 进程间通信机制	(40)
2.5 外围设备互连	(46)
2.6 中断和中断处理	(54)
2.7 设备驱动程序	(56)
2.8 模块	(68)
2.9 Linux 内核源程序	(71)
2.10 小结	(74)
第 3 章 基本命令和主要实用程序	(75)
3.1 改变工作目录	(75)
3.2 建立和移动	(76)
3.3 ls 命令	(81)
3.4 重命名	(88)
3.5 复制和删除	(88)
3.6 查找和排序	(93)
3.7 格式化	(97)
3.8 网络命令	(98)
3.9 小技巧	(107)
3.10 其他	(110)
3.11 小结	(118)
第 4 章 文件系统	(119)
4.1 概述	(119)
4.2 文件系统管理	(122)
4.3 第二代扩展文件系统(EXT2)	(128)
4.4 虚拟文件系统(VFS).....	(134)
4.5 缓存区	(139)

4.6 小结	(141)
第5章 目录结构与磁盘系统	(142)
5.1 概述	(142)
5.2 目录树	(143)
5.3 文件类型	(147)
5.4 文件属性	(148)
5.5 磁盘及其他存储介质	(151)
5.6 分配磁盘空间	(157)
5.7 管理磁盘	(158)
5.8 无文件系统的磁盘	(159)
5.9 小结	(159)
第6章 Shell 简介	(160)
6.1 Shell 基础知识	(160)
6.2 Shell Script(脚本)知识简介	(169)
6.3 Bash	(170)
6.4 小结	(188)
第7章 Csh 和 Tcsh	(189)
7.1 Csh	(189)
7.2 Tcsh 简介	(196)
7.3 小结	(201)
第8章 Shell 之间的比较	(202)
8.1 选择 Shell 的决定因素	(202)
8.2 Bourne Shell	(203)
8.3 POSIX Shell	(204)
8.4 C Shell	(204)
8.5 Korn Shell	(205)
8.6 TC Shell	(206)
8.7 Bourne Again Shell	(206)
8.8 Z Shell	(207)
8.9 小结	(208)
第9章 Linux 网络系统	(209)
9.1 网络的拓扑结构	(209)
9.2 TCP/IP 网络简介	(209)
9.3 IP 层	(213)
9.4 地址解析协议(ARP)	(216)
9.5 TCP/IP 传输层	(218)
9.6 INET Socket 层	(219)
9.7 BSD Socket 接口	(222)
9.8 检查 TCP/IP 配置	(224)
9.9 网络文件系统	(224)

9.10	网络分析工具	(225)
9.11	小结	(226)
第 10 章	正文编辑	(227)
10.1	流编辑 Sed	(227)
10.2	屏幕编辑器 Vi	(228)
10.3	Emacs	(235)
10.4	Emacs 在线帮助	(238)
10.5	Emacs 的组合环境	(244)
10.6	与 Emacs 有关的议题	(249)
10.7	如何退出 Emacs	(254)
10.8	如何载入文件与储存文件	(255)
10.9	Emacs 的基本编辑指令	(257)
10.10	Emacs 的基础编辑指令	(263)
10.11	Emacs 高级编辑指令	(270)
10.12	Emacs 的其他相关事项	(273)
10.13	小结	(275)
第 11 章	X Window 系统	(276)
11.1	概述	(276)
11.2	Client/Server 架构	(278)
11.3	窗口管理器	(282)
11.4	图形用户界面	(285)
11.5	客户程序之间的通信	(287)
11.6	X 的特征	(288)
11.7	小结	(289)
第 12 章	X Server	(291)
12.1	X Server ——显示器控制软件	(291)
12.2	X11R5 和 Xfree86	(294)
12.3	各种 X Server	(294)
12.4	设置 X Window 系统	(296)
12.5	Xf86Config 使用说明	(299)
12.6	XFree86 帮助	(314)
12.7	Turto Linux 的配置工具 TurboXCfg 使用	(317)
12.8	小结	(319)
第 13 章	桌面环境	(320)
13.1	桌面环境简介	(320)
13.2	KDE 介绍	(321)
13.3	Gnome 介绍	(324)
13.4	小结	(330)
第 14 章	KDE 应用软件	(331)
14.1	文件处理	(331)

14.2	网络应用程序	(349)
14.3	图像和多媒体	(360)
14.4	管理系统	(366)
14.5	其他实用软件	(388)
14.6	小结	(396)
第 15 章	使用资源定制客户程序	(397)
15.1	客户程序之间的通信	(397)
15.2	定制客户程序的资源机制	(398)
15.3	资源说明的使用	(401)
15.4	Xt 如何使用资源	(406)
15.5	其他客户程序如何利用资源	(408)
15.6	保存资源说明	(409)
15.7	在 Server 上保存预设值(xrdb)	(412)
15.8	一些常见错误	(413)
15.9	小结	(415)
第 16 章	定制窗口管理器	(416)
16.1	窗口管理器和应用程序之间的关系	(416)
16.2	使用 fvwm	(417)
16.3	定制 fvwm	(420)
16.4	fvwm 2 的主要命令	(424)
16.5	配置文件	(431)
16.6	小结	(446)
附录 A	GNU GENERAL PUBLIC LICENSE	(448)
附录 B	GNU 通用公共许可证 (GPL)	(455)

第 1 章 UNIX,GNU 与 Linux

Linux 有两种不同的含义。从严格的技术定义讲,它指的是 Linux 操作系统的内核;然而,人们已经习惯用它来表示以 Linux 内核为基础的、全部使用自由软件组成的 UNIX 类操作系统或各种 Linux 发行版本。

Linux 的迅速发展有它的背景,它和 UNIX 操作系统和自由软件运动的长期发展分不开。Linux 内核继承 UNIX 两大分支(UNIX System V 和 BSD UNIX)的主要特征,并进一步增加了新的特征。

Linux 内核和自由软件基金会的 GNU 项目所长期开发的大量系统软件有机地结合在一起,形成了完整的和全部用自由软件组成的操作系统。人们已经习惯将它称作 Linux 操作系统。但是,将它称为 GNU/Linux 似乎更为合理。

Linux 最初是为 IBM PC 及其兼容机开发的操作系统,现在可以将它用在许多不同的硬件平台上。

Linux 以它的高效、稳定和优异的联网能力为特色,可以使用灵活和功能强大的 X Window 系统以及 KDE 或 GNOME 桌面环境。

为了让读者对 Linux 系统有较为深入和全面的了解,本章将介绍 UNIX 系统、GNU 系统和 Linux 内核之间的关系以及它们的发展过程。

1.1 UNIX 系统

本节将介绍硬件、操作系统和应用软件之间的关系,UNIX 系统漫长的发展过程,以及各种 UNIX 类操作系统。

1.1.1 硬件、操作系统和应用软件之间的关系

计算机通过在它上面运行各种应用软件为人们提供不同类型的服务。

现代计算机系统是复杂的,个人计算机(PC)也不例外。它由许多部件组成。除了中央处理器和内存外,计算机还可以连接各种各样的外部设备。许多计算机和网络连接在一起。直接在硬件系统上编写应用软件是困难的。所以,计算机需要有操作系统。操作系统用来管理系统中的各种部件,将复杂的计算机硬件接口转化为相对简单的、统一的编程接口以简化应用软件的开发。操作系统是计算机硬件和应用软件的中介。

个人计算机原本使用的操作系统是 DOS 和 Windows。

个人计算机硬件的发展速度非常快。自 Intel 386 出现后,个人计算机的体系结构有很大的改变。它的能力和结构足以支持在它上面运行 UNIX 类操作系统。而且,在 UNIX 类操作系统取代 DOS 和 Windows 后,更能充分发挥新的硬件体系结构的能力。当时,这两种操作系统在功能上的差距是很大的,即使在今天仍然具有明显的差距。因而,在 PC 上发展 UNIX 类操作系统具有巨大的市场吸引力。人们相继推出了在 PC 上使用的商业的和自由的 UNIX 类操作系统版本,Linux 只是自由的 UNIX 类操作系统之一。

UNIX 类操作系统比 DOS 和 Windows 复杂,这一点也应该充分肯定。

UNIX 类操作系统本身大体可以划分为两大部分:操作系统内核和系统软件。其中,系统软件又包括各种命令解释程序、各种编译软件和大量的实用程序。

UNIX 操作系统给用户两种界面:由命令解释程序 Shell 提供的命令行界面和由 X Window 系统提供的图形用户界面。与此相对应,UNIX 操作系统中有直接建立在系统软件基础上的应用软件,以及建立在 X Window 系统上的应用软件。为了使 X Window 系统具有良好的可移植性,人们将它设计成不依赖于硬件和操作系统。也就是说,并不将 X Window 系统深植于操作系统内部。从这种意义上讲,X Window 系统本身只是 UNIX 操作系统的应用软件。

将上面的内容综合在一起,可以得到如图 1.1 所示的硬件、UNIX 类操作系统和应用软件之间的关系框图。

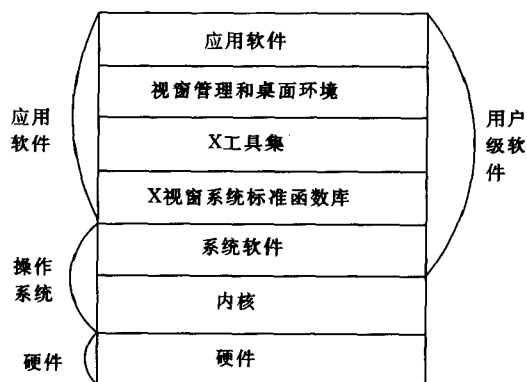


图 1.1 应用软件、操作系统和硬件的关系

UNIX 类操作系统的内核受到严格的保护。一个运行中的程序(也就是进程)有两种执行模式,即内核(或系统)模式和用户(或正常)模式。程序通常在用户模式下执行,但当执行系统调用时就进入内核模式。与此相对应,出现了另一个名词:用户级程序(或软件)。它的范围应该包括系统程序和应用程序两部分。

虽然常常将 UNIX 的 Shell 程序称为命令解释程序,但它并不只是简单的命令解释程序,它还是一种强有力的编程语言。它称得上是第 4 代语言。它和 UNIX 的大量实用程序结合在一起,能够充分发挥出 UNIX 系统的强大威力。有人说 UNIX 的能力是通过 Shell 释放出来的。仔细想来,这样说一点也不过分。

UNIX 是一个开放系统,许多人参加了它的开发工作。指导 UNIX 开发的哲学思想是后人站在前人的肩膀上前进。如果前面已经有的,就尽可能利用它,并不断去增加新的功能。在这种思想指导下,经过长期的发展,累积了大量的实用程序。

作为第 4 代语言的 Shell,它不仅可以使用传统的编程方法,更重要的是它可以使用组合法,将各种实用程序的功能有机地组合在一起,使它们完成复杂的功能。甚至可以使用组合法快速实现各种应用软件的雏型,再对它进一步进行改进、提高或更新。这就是所谓的快速样板法。一行 Shell 命令能够抵上 10 行 C++,甚至 100 行 C 语言,足可以看出 Shell 的效率。与 DOS 和 Windows 之间的关系不同的是,即使使用了 X Window 系统,也不能完全离开 Shell。

X Window 系统为 UNIX 系统的图形用户界面提供了基础结构,它和其他视窗系统有很大的不同。它使用了客户机/服务器体系结构。客户程序、窗口管理程序和桌面管理程序都是

独立的应用程序。这些内容将在第 11 章中再做详细介绍。

1.1.2 UNIX 系统的发展

UNIX 在国外经历了漫长的发展过程,因而在学术界和商业界打下了深厚的基础。

1970 年,UNIX 诞生于 Bell 实验室。那时,Bell 实验室与 AT&T 公司仍是一家人。最初的 UNIX 版本用汇编语言写成,以后用 C 语言改写。开始,UNIX 只在 AT&T 内部和部分学校使用。7 年后,UNIX 才开始成为商业产品。

20 世纪 70 年代末,UNIX 在加利福尼亚大学的伯克利(Berkeley)分校得到很大的发展,这就是著名的 BSD UNIX 系列。UNIX 在这个著名学府里的发展一直延续到 1992 年。那时,伯克利大学决定不再发展 UNIX。BSD UNIX 的最后版本是 BSD 4.4。

从 20 世纪 80 年代起,除 AT&T 及伯克利以外,对 UNIX 进行重大改进的另一个机构是卡内基梅隆大学(Carnegie-Mellon Unverisry, CMU)的 Mach 项目。现在,在多处理机、小型电脑以及超级电脑采用的操作系统中,很多都跟 Mach 有关。例如,NeXT 就是其中之一(根据可靠的说法,NeXT 是 UNIX 中的极品)。值得一谈的是 Mach 2.5 还含有 AT&T UNIX 的纯正血统,但 Mach 3.0 就没有 AT&T UNIX 的源代码了。CMU 把 Mach 原本拥有的 UNIX 贵族血统彻底换成了民家姑娘,从而成为一个人人都可以亲近的操作系统。这样,采用 Mach 3.0 为内核的操作系统也就没有 AT&T 许可证限制了。自由软件基金会正在发展的 GNU Hurd 就是建立在 Mach 3.0 基础上的操作系统内核。

在冷落几年后,AT&T 又开始在 UNIX 领域重显身手。从 1983 年 System III 诞生起,接着是 System IV,一直发展到 System V。现在许多 UNIX 分支是在 System V 基础上发展而来的。至于 System V 本身的改进则从 System VR2(R2 表示第 2 次发行)起,直到 System VR4。最后的发行版是 UNIX SVR4 或简称为 SVR4。SVR4 与 SVR3 有很大的不同,因为 SRV4 引进了 BSD UNIX 的特色,也就是将 SVR3 以及 BSD4.2 融合在一起。以上是发生在 20 世纪 80 年代的故事。

20 世纪 80 年代初,出现了 IBM 个人电脑。两年后,跟着推出了 PC 版本的 UNIX。这个市场几乎一年比一年好,美国 SCO UNIX 公司是这一行的佼佼者。

20 世纪 80 年代中,出现了工作站。这些运行 UNIX 的工作站将 UNIX 带入更为广阔的应用领域,尤其在图形处理、数据库应用以及网络通信等方面。在 20 世纪 90 年代,工作站和 UNIX 的 PC 版本得到市场青睐,SUN 公司更是独领风骚。与此同时,超级电脑的 UNIX 版本也同样受宠。这一方面的例子有 Cray、Fujitsu、Convex 等公司的产品。

进入 20 世纪 90 年代后,UNIX 工作站与 UNIX 超级电脑相结合,进军原本属于大型电脑市场的商业应用领域,而且明显取得了成功。

在 20 世纪 90 年代,UNIX 已被广泛应用。在不同环境下都有 UNIX。从 PC、工作站、小型电脑到超级电脑中都能找到它的踪影。从用户的角度看,只要熟悉一种环境下的 UNIX,走到哪里都不会觉得陌生。

从前 UNIX 是 AT&T 的注册商标。AT&T 成立 UNIX System Lab,Inc. (USL)子公司后,UNIX 的注册版权便转移到 USL。由于 USL 还是 AT&T 的子公司,所以 AT&T UNIX 仍然是一般人对 UNIX 最常用的全称。这个事实一直维持到 1993 年春天,到 AT&T 把 USL 卖给 Novell 公司时为止。从此 AT&T UNIX 这个字眼只存在于历史,或者只是概念上的名词而已。

到 1994 年春天,Novell 公司放弃了 UNIX 商标,将它授与英国的名为 X/Open 的标准化

组织。这一举动使所有符合 X/Open 所制定规范的操作系统，都可以冠上 UNIX 名称。不过 Novell 公司仍然保有原 UNIX 系统源代码的著作权。

1.1.3 各种 UNIX 系统版本

经过长期的发展，建立了很多不同的 UNIX 版本。它们的共同特点是多用户和多任务系统，即在同一时刻可以支持多个用户执行多项任务。UNIX 的一个重要分支来源于的贝尔实验室；另一个重要分支来源于加利福尼亚大学的伯克利(Berkeley)分校。Berkeley UNIX 的正规名称是 BSD UNIX。BSD 是 Berkeley Software Distribution 的缩写。

UNIX 虽然有多种版本，实际上它们或基于 BSD，或基于 AT&T 的 System V，或者基于两者之上。下面给出一份不同的 UNIX 系统列表，参见表 1-1。从表中我们看到现在已经有不止一种自由的 UNIX 类操作系统。

表 1-1 各种 UNIX 系统

UNIX 的名称	公司或组织、机构名称
386BSD	Internet 自由提供
AIX	IBM
A/UX	Apple
BSD	加利福尼亚大学的伯克利分校
BSD-LITE	加利福尼亚大学的伯克利分校
Goherent	BSDI
Dynix	Scquent
FreeBSD	Internet 自由提供
HP-UX	HP
Hurd(GNU)	FSF
Interactive	Graphics
Linux	Internet 自由提供
Mach	Carnegie Mellon
Minix	Andy Anenbaum
MKS Toolkit	Mortice Ker
NetBSD	Internet 自由提供
Nextstep	Next
OSF/1	DEC
SCO UNIX	Sarcta Cruz Operation
Solaris	Sun Microsystem
SunOS	Sun Microsystem
System V UNIX	PC 机上的各种版本
Unicos	Cray Research
UNIXware	Novell
OpenBSD	Internet 自由提供

从前面的说明可以看出，国外在开发 UNIX 类操作系统方面做了大量的工作。UNIX 类操作系统对学术界和商业界都具有广泛和深刻的影响。正因为这样，自由软件基金会的 Richard

Stallman 和开发 Linux 内核的 Linus Torvalds 才不谋而合地共同选择了 UNIX 类操作系统作为他们的发展方向,最后走到了一起。他们的成果相结合,形成了今天 Linux 操作系统的辉煌。

1.2 GNU 系统

自由软件包括很大的范围,Richard Stallman 和他领导的自由软件基金会(FSF)所开发的 GNU 系统只是其中的一部分。但是,Richard Stallman 和 FSF 所倡导的自由软件理念促进了自由软件运动的蓬勃展开,促使自由软件向完整的系统和纵深方向发展,而且在如何利用版权法保护自由软件及其衍生作品继续保持自由状态方面也起到了不可磨灭的作用。

1.2.1 GNU 系统简介

Richard Stallman 于 1984 年创立了 GNU 项目。该项目的目标是全部用自由软件建立完整的、自由的 UNIX 类操作系统。GNU 宣言(Manifesto)事先宣布了开发自由的 UNIX 类操作系统的目标,并把它称为 GNU 操作系统。GNU 的名称来自 GNU's Not UNIX 的双重缩写。它表示 GNU 操作系统并不是传统意义上的 UNIX 操作系统,它是全部用自由软件建立的新 UNIX 类操作系统。

GNU 操作系统或 GNU 系统现在使用 Linux 内核,并且得到了广泛的应用。这一系统通常被称为 Linux 系统,但是将它称为 GNU/Linux 更为确切。

一个操作系统不能只有内核,在它上面还要能运行其他程序。即使在 20 世纪 70 年代,一个操作系统也必须包括命令解释程序、汇编程序、编译程序、正文编辑程序和邮递程序等。其他的操作系统是这样,GNU 操作系统也是这样。

开发完整的系统是一项巨大的工程。为了将它付诸实施,Richard Stallman 决定采用已经存在的自由软件,继续充分发挥它们的作用。例如,开始采用 TeX 作为主要的排版系统,以后又采用 X Window 系统作为 GNU 视窗系统的基础。由于这项决定,GNU 系统并不完全是 GNU 软件的集合,也包括了非 GNU 软件。它包括了其他人和其他项目为他们自己用途所开发的软件。之所以能够这样做,就是因为它们也属于自由软件。所以,GNU 系统是由 GNU 软件和非 GNU 软件组成的系统。

GNU 系统的长远目标还包括开发应用软件,开发拖放图标的桌面环境和游戏等。在 X Window 系统基础上建立的拖放图标的桌面环境(GNOME)功能非常强大,其中包括提供数据库的接口等。目前它们已经得到使用。

1.2.2 自由软件基金会

1984 年 1 月,Richard Stallman 辞去了 MIT 的工作,开始从事 GNU 项目软件的开发工作。他认为离开 MIT 是必要的,这样做将使得 GNU 操作系统作为自由软件发行时不会受到 MIT 的干预。如果他继续作为 MIT 的职员,MIT 可以要求占有这项成果,并可以强加于他们自己的发行条款,甚至将它转为专有的软件包。他不希望做大量的工作而无助于实现原来的软件共享目标。

Richard Stallman 首先开发文本编辑软件 GNU Emacs,并在 1985 年初开始交付使用。随着人们使用 Emacs 的兴趣不断增长,其他人开始参加到 GNU 项目中来。在 1985 年建立了自由软件基金会(Free Software Foundation,FSF)。

FSF 致力于消除对计算机软件使用的种种限制。他们在所有的计算领域促进自由软件的开发和使用,以此来促进共享软件目标的实现,特别是促进 GNU 自由操作系统的开发。FSF 开发新的自由软件,使它成为协调一致的不需要使用任何专有软件的完整系统。这就是人们所说的由 Richard Stallman 推动的自由软件运动。

FSF 是免税的开发自由软件的团体。FSF 的经费部分来自捐赠,主要经济来源是从销售自由软件、文档的副本和提供服务得到。GNU 项目的目标是建立完整的操作系统来筹集基金。

1.2.3 自由软件赋予用户的自由

自由软件(free software)名称中的自由指的是自由而不是价格。它包括用户在运行、复制、发布、研究、修改和改进程序方面的种种自由。具体讲,它牵涉到用户在下列 4 方面的自由:

- 为任何目的运行程序的自由(0 级自由);
- 研究程序如何工作,并将程序用在他们自己的需要中的自由(1 级自由)。取得源代码是做到这一点的先决条件;
- 发行程序的副本,以帮助周围的人们的自由(2 级自由);
- 改进程序,并向公众发行程序的改进版本使软件用户团体受益的自由(3 级自由)。取得源代码是做到这一点的先决条件。

如果一个程序的用户拥有所有以上各方面的自由,这个程序就是自由软件。这样,用户就拥有在任何地方、向任何人重新发行程序副本的自由,不管是否对它进行过修改,不管是赠送的还是对发行版本收取一定费用的都可以。自由表示用户不必为这些行动去请求得到许可或者为得到许可而付出代价。

用户同样拥有修改自由软件,并将它私下用在自己的工作或游戏中的自由。用户甚至不必提到它的存在。如果用户真地公开了修改的结果,也不必专门或者以特定方式去通知其他任何人。

由于自由软件强调的是自由而不是价格,销售程序的副本和自由的精神并不矛盾。事实上,销售程序副本的自由是重要的。将自由软件收集在 CD-ROM 上出售是自由软件团体为自由软件发展筹集资金的重要手段。

不管以什么方式取得自由软件,用户始终应该具有复制、修改和重新发行软件的自由。

为了使修改自由软件和公布自由软件的修改版本变得有实际意义,需要有发行规则来保证用户始终能够得到这些自由。这种发行规则是可以接受的,只要它和上面提到的必不可少的自由不相冲突。

自由软件基金会的 Copyleft 规则是一种重新发行程序的规则。它不允许用户在重新发行程序时对其他用户增加额外的限制,即拒绝其他人得到上面提到的必不可少的自由。这种规则与必不可少的自由不相矛盾。相反,它被用来保护用户的必不可少的自由。借助自由软件的发行规则可以防止有人将 GNU 软件转化为专有软件。

介绍一下 Copyleft 规则和 GNU 通用公共许可证(GPL)。

将程序变为自由软件的最简单办法是放弃版权,使它变为公共域软件。但是这样做不能防止有人将它变为专有软件。

Copyleft 规则使用了版权法,也就是用版权来确保自由软件继续保持自由状态。但它的用法和商业的用法完全相反。在通常情况下,公司用版权法保护专有软件,剥夺用户使用专有软

件的自由。但是,Copyleft 使用版权法作为确保自由软件及其衍生作品继续保持自由状态的手段。

Copyleft 规则的具体要求是:给自由软件的用户以运行程序,复制程序,修改程序和发行程序的版本修改自由,但不允许用户在重新发行程序时增加他自己的限制。这样做保证了得到自由软件副本的任何人拥有使用软件的必不可少的自由,使这些成为不可剥夺的权利。

要保证程序的用户得到自由,将程序的修改版本继续保持自由状态是非常重要的。作为有效的 Copyleft 规则,必须使自由软件的修改版本同样保持自由。这就保证:当将基于自由软件的衍生作品公开发表时,它可以继续为软件用户团体所使用。当程序员们自愿改进 GNU 软件时,正是 Copyleft 规则防止了他们的雇主利用这些改进将程序作为他们的专有软件。

X Window 系统是一个反面的例子。一些公司通常将它进行一些局部修改,将它移植到它们自己的系统和硬件上作为公司的专有产品。这些修改工作量和 X Window 系统本身的巨大开发工作量相比往往是微不足道的。如果以进行了修改为理由拒绝给用户以自由,这就太容易将自由软件转化为专有软件了。

一个有关的问题是:将自由程序和非自由程序结合在一起。这样的组合不可避免地会导致失去自由。非自由程序没有自由,将导致整个程序失去自由。允许这样做无异于打开足以使自由软件之船沉没的大缺口。因此,对 Copyleft 规则极为重要的要求是堵住这个漏洞。任何增加到受 Copyleft 规则保护的程序中的任何程序,或者与受 Copyleft 规则保护的程序相结合的任何程序必须做到这样:更大的组合程序必须同样是自由的,并受到 Copyleft 规则保护。

是否使用 Copyleft 规则来保护自由软件的自由,这是在自由软件团体中最主要的争论之一。但是,Copyleft 规则是当前将程序的修改版本继续保持自由状态的惟一有效办法。

Copyleft 规则的具体实现之一是 GNU 通用公共许可证(GPL),它用来保护大多数 GNU 软件。自由软件基金会还有其他类型的 Copyleft 规则。如 GNU 手册同样受到 Copyleft 规则保护。但是它使用了简单得多的 Copyleft 规则,因为对手册并不需要使用像 GPL 那样复杂的条款。对函数库的保护则使用库通用公共许可证(LGPL)。

在国内,对 GPL 已经有多种翻译文本。在 GNU 组织的站点上有其他语言的译文,也有来自台湾地区的中译文本。作为法律文件,只有英文文本才是正式有效的文本。

1.2.4 GNU 系统和 Linux 内核的关系

为了让读者了解 FSF 对 GNU 系统和 Linux 内核之关系的看法,我们将 Richard Stallman 的观点摘译如下:

许多计算机用户每天运行着 GNU 系统的变体,而并不认识到它的存在。由于某些原因,今天广泛使用的 GNU 系统经常被大家称为“Linux”,许多用户并不知道它与 GNU 系统联系的紧密程度。

确实有一个 Linux,它是操作系统的内核,而且人们正在使用它。但是,用户不可能只使用内核本身。内核只在作为整个操作系统的一部分时才有用。Linux 内核通常与 GNU 操作系统结合在一起使用。系统的基础是 GNU 系统,Linux 则扮演系统内核的角色。许多用户并不注意两者之间的区分,而把整个系统称为“Linux”。这不利于对 GNU 的理解。

程序员知道 Linux 是内核。但是他们经常听到将整个系统称为 Linux。例如,许多人相信:一旦 Linus Torvalds 完成了内核的编写任务后,他的朋友们就去收集其他自由软件,而并不意识到建立自由 UNIX 类操作系统所需要的大多数软件已经存在。

无疑他们发现的是 GNU 系统。因为 GNU 项目从 1984 年就开始着手建立这样的系统,这些可供使用的自由软件加起来就是一个完整的系统。GNU 宣言(Manifesto)预先宣布了开发自由的 UNIX 类操作系统的目标。在最初的 GNU 项目公告中同样发表了 GNU 系统的部分原始计划。在 Linus Torvalds 写成 Linux 内核时,GNU 系统(除内核外)几乎已经完成。

大多数自由软件项目有为特定任务开发特定程序的目标。例如,Linus Torvalds 的目标,是开发 UNIX 类操作系统的内核(Linux);Donald Knuth 的目标是写文本排版系统(TeX);而 Bob Scheifler 设定的目标是开发视窗系统(X Window)。用来自特定项目的特定程序来衡量这种项目的贡献是很自然的事。

如果试图用这种方式来衡量 GNU 的贡献,会得到什么结论?发行 CD-ROM 的厂商发现:在他们的 Linux 发行版本中,GNU 软件占最大的份额,大约占源代码总量的 28%,其中包括形成系统必不可少的组成部分。Linux 内核只占源代码总量的 3%。如果在由他人主要编写程序的基础上为系统命名的话,最合适的选择自然应该是 GNU。

但是,FSF 并不认为这是考虑问题的正确方法。GNU 项目过去不是,现在也不是开发特定软件包的项目。它不是开发 C 编译软件的项目,虽然 FSF 已经这样做了。它也不是开发文本编辑软件的项目,虽然 FSF 也开发了这样的软件。GNU 项目的目标是开发完整和自由的 UNIX 类操作系统。

许多人对系统中的自由软件作出了主要贡献,他们都应该受到尊敬。这是一个系统,而不只是一些有用程序的集合,因为 GNU 项目的目标就是要建立一个系统。FSF 列出了建立完全自由的系统所需要的程序清单,而且系统地编写了这些软件,或者找人去写清单上的程序。FSF 编写了重要而并不令人兴奋的 GNU 的重要组成部分,如汇编和连接软件,因为在系统中不可能没有它们。在一个完整的系统中不仅仅需要编程工具,例如 Bash 命令解释程序、PostScript 解释程序 Ghostscript 和 GNU C 库也同样重要。在 20 世纪 90 年代初期,FSF 已经将除了内核以外的整个系统整合在一起,FSF 也在开发称为 GNU Hurd 的内核,它建立在 Mach 的基础之上。开发这个内核的困难比 FSF 想象的要大,但 FSF 仍然在继续完成它。

很幸运,我们不必等待 Hurd 内核的完成,因为 Linux 内核已经能够正常进行工作。当 Linus Torvalds 写完 Linux 内核后,他填补了系统最后的缝隙。人们然后能够将 Linux 内核和 GNU 系统组合在一起形成完全自由的系统:基于 Linux 内核的 GNU 系统(或者将它简称为 GNU/Linux 系统)。

将它们组合在一起听起来很简单,但实际上并不是一件轻松的工作。GNU C 库需要做重大的改变。集成一个完整的系统作为发行版本也是一项大任务,它还有安装和引导等问题需要解决。开发不同的系统发行版本也有很多工作要做。

在 GNU 系统以外,另一个项目也独立完成了一个自由的 UNIX 类操作系统。这个系统的名称叫 BSD,它由加利福尼亚大学的伯克利(Berkeley)分校完成。BSD 开发人员从 GNU 项目得到启发,偶尔也得到 GNU 激进分子的鼓励,将他们的成果作为自由软件向社会奉献。但是,他们的实际工作很少和 GNU 项目重复。今天的 BSD 系统使用一些 GNU 软件,就像 GNU 系统和它的变体也使用一些 BSD 软件一样。从整体上讲,它们是独立开发的不同系统。目前存在的自由操作系统几乎都是 GNU 系统的变体或者是 BSD 系统的一种。

GNU 项目支持 GNU/Linux 系统以及 GNU 系统,有时有资金支持。FSF 用资金支持对 GNU C 库与 Linux 内核有关的扩充,所以它们现在能够很好地结合在一起。最新的 GNU/Linux 系统免费使用当前库的最新发行版本。FSF 同样对 Debian GNU/Linux 的早期开发给以