LINUX

- 全面的基础知识，帮助读者完成 Linux 系统下内存、CPU、磁盘、网络及音频设备的访问及管理
- 案例为指导，每一个知识点中都实现了一个应用案例，读者可以针对每一个知识点进行实例编程演练
- 紧扣技术前沿，所采用的开发平台为新的 Linux 内核，开发工具为新的 GCC

高级程序设计（第三版）

◆ 杨宗德 吕光宏 刘雍 编著

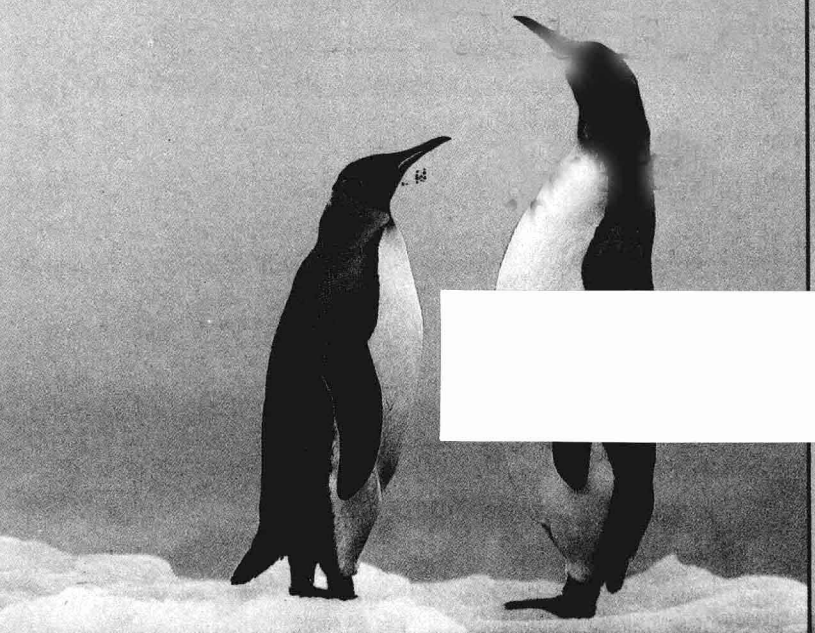
Linux

人民邮电出版社
POSTS & TELECOM PRESS

LINUX

高级程序设计（第三版）

◆ 杨宗德 吕光宏 刘雍 编著



Linux

人民邮电出版社
北京

图书在版编目（CIP）数据

Linux高级程序设计 : 第3版 / 杨宗德, 吕光宏, 刘雍编著. -- 3版. -- 北京 : 人民邮电出版社, 2012.11
ISBN 978-7-115-29290-2

Ⅰ. ①L… Ⅱ. ①杨… ②吕… ③刘… Ⅲ. ①Linux操作系统—程序设计 Ⅳ. ①TP316.89

中国版本图书馆CIP数据核字(2012)第207497号

内 容 提 要

本书围绕 Linux 操作系统“一切都是文件”的特点，讲述了 Linux 操作系统下应用层“一段执行单元（进程）对系统资源（CPU 资源、各类文件资源）的管理”。详细介绍了 Linux 系统编程环境及编程工具（GCC/Makefile/GDB）、文件管理（文件属性控制、ANSI 以及 POSIX 标准下文件读写操作、终端编程）、进程管理（创建、退出、执行、等待、属性控制）、进程间通信（管道、消息队列、共享内存）、进程间同步机制（信号量）、进程间异步机制（信号）、线程管理（创建、退出、取消等以及属性控制）、线程间同步（互斥锁、读写锁、条件变量）、线程与信号以及 BSD socket 编程中的 TCP、UDP、原始套接口、网络服务器应用开发等内容，并对 Linux 系统下的音频应用程序开发做了讲解。

本书内容丰富、紧扣应用，适合从事 Linux 下 C 应用编程的人员阅读，也适合从事嵌入式 Linux 开发的人员阅读。

Linux 高级程序设计（第三版）

◆ 编　　著　杨宗德　吕光宏　刘　雍
责任编辑　张　涛

◆ 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
中国铁道出版社印刷厂印刷

◆ 开本：787×1092　1/16
印张：31
字数：953 千字　　2012 年 11 月第 3 版
印数：10 801 – 14 300 册　　2012 年 11 月北京第 1 次印刷

ISBN 978-7-115-29290-2

定价：59.00 元

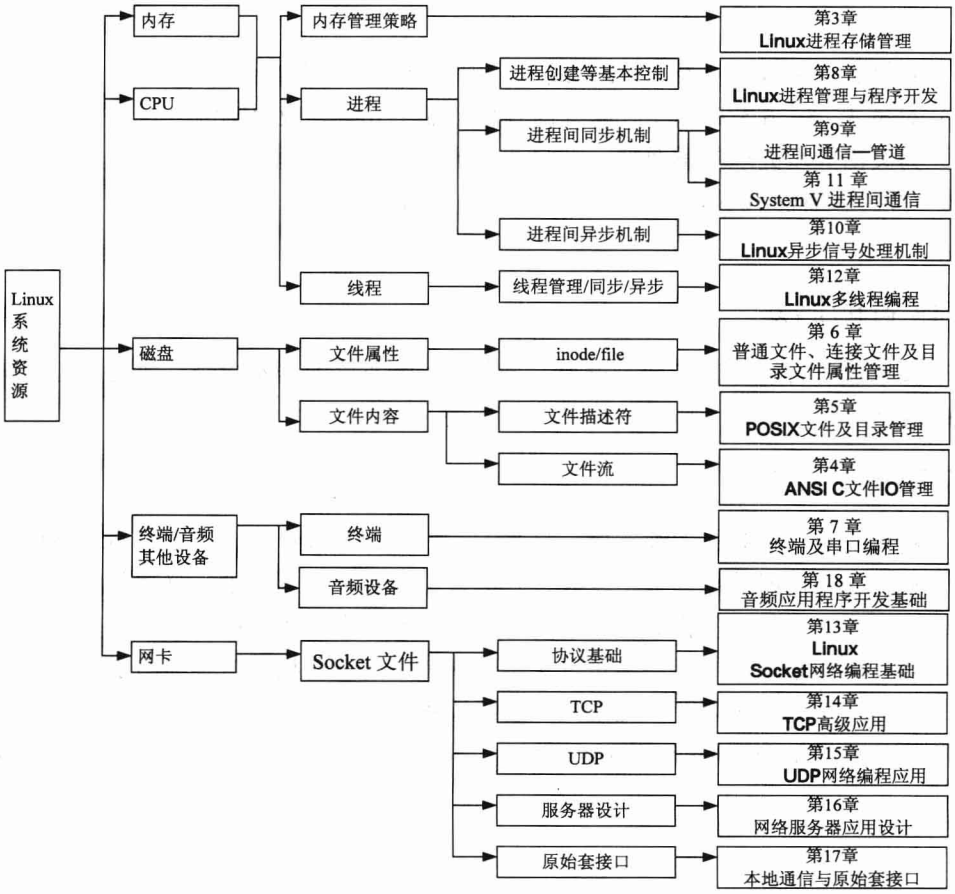
读者服务热线：(010)67132692　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

Linux 应用开发是目前最为广泛的软件开发内容之一，同时也是从事 Linux 内核及驱动开发的基础。《Linux 高级程序设计》一书经过两次出版，收到了大量的读者来信，对本书提出了各种意见和建议，同时，随着技术的更新，新技术、新应用不断涌现，综合各方面的考虑，笔者做了大量的修订工作，推出了第三版。

本书主要内容

本书围绕 Linux 操作系统“一切都是文件”的特点，讲述了 Linux 操作系统下应用层“一段执行单元（进程）对系统资源（CPU、内存、磁盘、外设、网络设备）的管理”，重点介绍了进程、线程、文件属性访问、文件内容访问、Socket、终端和音频设备编程等重要概念。整个内容安排如下图所示，完全涵盖了应用开发中涉及的所有内容。



本书主要特点

（1）内容丰富。本书是作者多年计算机教学及工程经验总结，整合了 Linux 应用编程的绝大多数知识点，几乎涵盖了 Linux 操作系统下 C 应用编程的所有内容，包括工具使用及环境设置、文件及文件管理、进程及进程管理、进程间通信、线程及线程管理、线程通信、网络及网络应用编程等知识点。

（2）循序渐进。本书在写作思路上避开了大量理论的介绍，按知识体系介绍→应用函数分析→应用案例开发的写作顺序，让读者在掌握具体知识点的同时可以掌握实例的具体实现。

（3）案例指导。本书中所有调用函数及引用都标出具体的出处（在 Linux 操作系统中的文件位置），读者可以一目了然地知道对应函数及类型的定义过程。另外，本书遵循案例教学思想，每一个知识点都讲解一个应用程序，且所有代码都在教学实践过程中调试通过，读者可以直接使用。

（4）紧扣应用。本书所列代码和实例都来源于具体的应用程序。

本书修订说明

这次修订在第二版的基础上增加了大量的应用案例及新的知识体系。

（1）新增加第 7 章终端编程，第 17 章本地通信及原始套接口内容以及第 18 章音频编程内容。

（2）对第 2 章编程工具及第 12 章线程编程章节做了适当的合并和删简。

（3）对第 4、5、6 章磁盘文件管理内容新增加了 tree 等应用案例。

（4）为突出异步处理的重要性，专门使用第 10 章介绍进程的异步信息处理。

（5）对网络编程知识体系进行了重新整编，根据应用协议更新了第 13、14、15、16、17 章内容。

对读者的假定

本书要求读者有较好的 C 语言基础，熟悉 Linux 系统的基本命令。如果读者对操作系统或 Linux 内核有一定了解，则更容易学习本书。

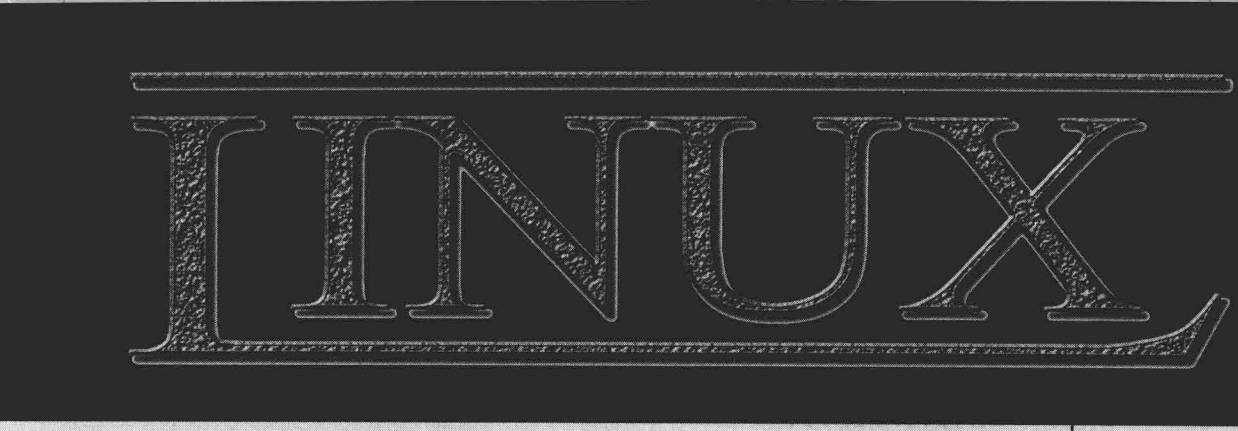
本书编写工作

本书所有内容由杨宗德主编完成，吕光宏、刘雍、邓玉春、曾庆华参与本书相关代码的编写及审稿工作。同时感谢何伟、张兵、刘兆宏、季建华、刘福刚、赵文革、黄弦等老师对本书的编写提出了大量宝贵指导意见，另外感谢石昀、朱元斌、钱文杰、陈功杰、汪洪、刘超、钟晓媛、刘梨平、石霞等同学试读了本书初稿，为本书相关内容提出了宝贵意见。由于时间仓促，本书难免有疏忽和不足之处，恳请读者批评赐教。

源程序下载地址为：www.ptpress.com.cn。联系邮箱为：zhangtao@ptpress.com.cn 。

编　者

目　录



第1章 Linux下C语言开发环境

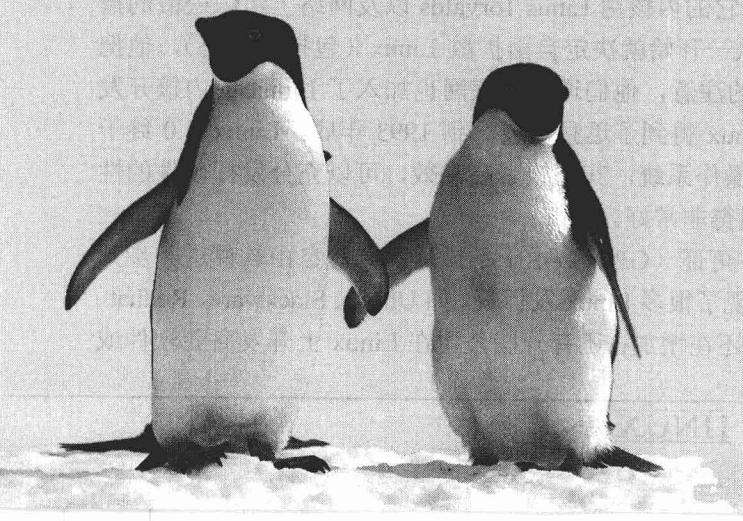
Linux 应用程序开发平台有别于 Windows 应用程序开发平台，因此在介绍具体编程内容之前，本书第 1、2 章主要介绍 Linux 操作系统下 C 语言程序的开发环境和开发工具。

本章主要介绍 Linux 下 C 语言开发环境，包括一些基本概念和基本编程环境。本章第 1 节主要对 Linux 操作系统及其相关术语进行了简要介绍。

本章第 2 节主要介绍 Linux 操作系统下编程基本概念以及如何获得 Linux 下的帮助文件，包括 Linux 操作系统下 C 语言库文件标准以及系统调用的基本概念。

本章第 3 节主要介绍 Linux 部分常用工具，包括文件打包工具、查找搜索工具，熟练使用这些命令或工具在编程时能够很好地提高效率。

本章第 4 节为读者展示了 GNU 编码规范和 Linux 内核编码规范。读者在学习 Linux 编程之前应养成良好的编码规范，这样不仅能增强代码的可读性，还能减少代码维护的工作量，提高代码的可扩展性。



1.1 Linux 操作系统简介

1.1.1 Linux 操作系统简介

UNIX 操作系统于 1969 年由 Ken Thompson 在 AT& T 贝尔实验室的一台 DEC PDP-7 计算机上实现。后来 Ken Thompson 和 Dennis Ritchie 使用 C 语言对整个系统进行了再加工和编写，使得 UNIX 能够很容易地移植到其他硬件的计算机上。由于此时 AT&T 还没有把 UNIX 作为它的正式商品，因此研究人员只是在实验室内部使用并完善它。正是由于 UNIX 是被作为研究项目，其他科研机构和大学的计算机研究人员也希望能得到这个系统，以便进行自己的研究。AT&T 采用分发许可证的方法，大学和研究机构仅仅需要很少的费用就能获得 UNIX 的源代码以进行研究。UNIX 的源代码被散发到各个大学，一方面使得科研人员能够根据需要改进系统，或者将其移植到其他的硬件环境中去；另一方面培养了大量懂得 UNIX 使用和编程的学生，这使 UNIX 的使用更为普及。

到了 20 世纪 70 年代末，在 UNIX 发展到版本 6 之后，AT&T 认识到了 UNIX 的价值，并成立了 UNIX 系统实验室（UNIX System Lab，USL）来继续发展 UNIX。因此一方面 AT&T 继续发展内部使用的 UNIX 版本 7，一方面由 USL 开发对外正式发行的 UNIX 版本，同时 AT&T 也宣布对 UNIX 产品拥有所有权。几乎在同时，加州大学伯克利分校计算机系统研究小组（CSRG）借助 UNIX 对操作系统进行了研究，他们对 UNIX 进行的改进相当多，增加了很多当时非常先进的特性，包括更好的内存管理、快速且健壮的文件系统等，大部分原有的源代码都被重写，很多其他的 UNIX 使用者，包括其他大学和商业机构，都希望能得到经 CSRG 改进的 UNIX 系统。因此 CSRG 的研究人员把他们的 UNIX 组成一个完整的 UNIX 系统——BSD UNIX（Berkeley Software Distribution）向外发行。

与此同时，AT&T 的 UNIX 系统实验室也在不断改进他们的商用 UNIX 版本，直到他们吸收了 BSD UNIX 中已有的各种先进特性，并结合其本身的特点，推出了 UNIX System V 版本。从此以后，BSD UNIX 和 UNIX System V 形成了当今 UNIX 的两大主流，现代的 UNIX 版本大部分都是这两个版本的衍生产品：IBM 的 AIX4.0、HP/UX11 和 SCO 的 UNIXWare 等属于 System V，而 Minix、freeBSD、NetBSD、OpenBSD 等属于 BSD UNIX。

Linux 由 UNIX 操作系统发展而来，它的内核由 Linus Torvalds 以及网络上组织松散的黑客队伍一起从零开始编写而成。Linux 从一开始就决定自由扩散 Linux（包括源代码），他把源代码发布在网上，随即就引起爱好者的注意，他们通过互联网也加入了 Linux 的内核开发工作。一大批高水平程序员的加入使 Linux 得到了迅猛发展。到 1993 年底，Linux 1.0 终于诞生。Linux 1.0 已经是一个功能完备的操作系统，其内核紧凑高效，可以充分发挥硬件的性能，在 4MB 内存的 80386 机器上也表现得非常好。

Linux 加入 GNU 并遵循通用公共许可证（GPL），由于不排斥商家对自由软件进一步开发，故而使 Linux 开始了又一次飞跃，出现了很多 Linux 发行版，如 Ubutu、Slackware、Redhat、TurboLinux、OpenLinux 等十多种，而且还在增加；还有一些公司在 Linux 上开发商业软件或

把其他 UNIX 平台的软件移植到 Linux 上来。如今很多 IT 界的大腕如 IBM、Intel、Oracle、Infomix、Sysbase、Netscape、Novell 等都宣布支持 Linux。商家的加盟弥补了纯自由软件的不足，扫清了发展障碍，Linux 得以迅速普及。

Linux 操作系统具有以下特点。

- Linux 具备现代一切功能完整的 UNIX 系统所具备的全部特征，其中包括真正的多任务、虚拟内存、共享库、需求装载、优秀的内存管理以及 TCP/IP 网络支持等。
- Linux 的发行遵守 GNU 的通用公共许可证（GPL）。
- 在原代码级上兼容绝大部分的 UNIX 标准（如 IEEE POSIX，System V，BSD），遵从 POSIX 规范。读者可以在网络上获得关于这一内容的更多信息。

1.1.2 GNU/Linux 简介

GNU 工程（GNU 是“GNU's Not UNIX”首字母缩写语）开始于 1984 年，旨在发展一款类 UNIX 且为自由软件的完整操作系统：GNU 系统。更精确地说，各种使用 Linux 作为内核的 GNU 操作系统应该被称为 GNU/Linux 系统。

GNU 工程开发了大量用于 UNIX 的自由软件工具和类 UNIX 操作系统，例如 Linux。虽然有许多组织和个人都对 Linux 的发展作出了帮助，但自由软件基金会依然是最大的单个贡献者。它不仅仅创造了绝大部分在 Linux 中使用的工具，还为 Linux 的存在提供了理论和社会基础。

为保证 GNU 软件可以自由地“使用、复制、修改和发布”，所有 GNU 软件都遵循无条件授权所有权利给任何人的协议条款——GNU 通用公共许可证（GNU General Public License，GPL）。

1985 年 Richard Stallman 创立的自由软件基金会（FSF，Free Software Foundation）为 GNU 计划提供了技术、法律以及财政支持。尽管 GNU 计划大部分时候是由个人自愿无偿贡献，但 FSF 有时还是会聘请程序员帮助编写。当 GNU 计划开始逐渐获得成功时，一些商业公司开始介入开发和技术支持。

到了 1990 年，GNU 计划已经开发出的软件包括了一个功能强大的文字编辑器 Emacs、C 语言编译器 GCC 以及大部分 UNIX 系统的程序库和工具。唯一依然没有完成的重要组件就是操作系统的内核（称为 HURD）。

1.1.3 相关术语介绍

1. POSIX 及其重要地位

POSIX 表示可移植操作系统接口（Portable Operating System Interface，缩写为 POSIX 是为了读音更像 UNIX）。它由电气和电子工程师协会（Institute of Electrical and Electronics Engineers，简称为 IEEE）开发，可以提高类 UNIX 环境下应用程序的可移植性。然而，POSIX 并不局限于 UNIX，许多其他的操作系统，例如 DEC OpenVMS 和 Microsoft Windows NT，都支持 POSIX 标准，尤其是 IEEE STD.1003.1-1990（1995 年修订）和 POSIX.1。POSIX.1 给操作系统提供了源代码级别的 C 语言应用编程接口（API），例如读写文件 read/write。POSIX.1 已经被国际标准化组织（International Standards Organization，缩写为 ISO）所接受，被命名

为 ISO/IEC9945-1:1990 标准。虽然某些部分还处在开发过程中，但是 POSIX 现在已经发展成为一个庞大的标准族。

2．GNU 和 Linux 的关系

GNU 项目已经开发了许多高质量的编程工具，包括 emacs 编辑器、著名的 GNU C 和 C++ 编译器（gcc 和 g++），这些编译器可以在任何计算机系统上运行。所有的 GNU 软件和派生工作均使用 GNU 通用公共许可证（GPL）。GPL 允许软件作者拥有软件版权，但要授予其他任何人以合法复制、发行和修改软件的权利。

Linux 中使用了许多 GNU 工具，用于实现 POSIX.2 标准的工具几乎都是 GNU 项目开发的。Linux 内核、GNU 工具以及其他的一些自由软件组成了人们常说的 Linux，包括 C 语言编译器和其他开发工具及函数库、X Window 窗口系统、各种应用软件（包括字处理软件、图像处理软件等）、其他各种 Internet 软件（包括 FTP 服务器、WWW 服务器）、关系数据库管理系统等。

3．通用公共许可证（General Public License，GPL）

GPL 的文本保存在 Linux 系统的不同目录中的 COPYING 文件里。例如，键入“cd/usr/doc/ghostscript*”，然后再键入“more COPYING”可查看 GPL 的内容。GPL 和软件是否免费无关，它的主要目标是保证软件对所有用户来说是自由的。GPL 通过如下途径实现这一目标。

（1）要求软件以源代码的形式发布，并规定任何用户都能够以源代码的形式将软件复制或发布给其他用户。

（2）提醒每个用户，对于该软件不提供任何形式的担保。

（3）如果用户的软件使用了受 GPL 保护的软件的任何一部分，该软件都会成为 GPL 软件，也就是说必须随应用程序一起发布源代码。

（4）GPL 并不排斥对自由软件进行商业性质的包装和发行，也不限制在自由软件的基础上打包发行其他非自由软件。

遵照 GPL 的软件并不是可以任意传播的，这些软件通常都有正式的版权，GPL 在发布软件或者复制软件时都会声明限制条件。但是，从用户的角度考虑，这些根本不能算是限制条件，相反，用户只会从中受益，因为它可以确保用户获得源代码。

尽管 Linux 内核也属于 GPL 范畴，但 GPL 并不适用于通过系统调用而使用内核服务的应用程序，通常把这种应用程序看作是内核的正常使用。假如准备以二进制的形式发布应用程序（像大多数商业软件那样），则必须确保自己的程序未使用 GPL 保护的任何软件。如果软件通过库函数调用而且使用了其他软件，则不必受此限制。大多数函数库受另一种 GNU 公共许可证，即 LGPL 的保护，下面将会介绍。

4．LGPL（Libraray General Public License）

GNU LGPL（GNU 程序库通用公共许可证）的内容包括在 COPYING.LIB 文件中。如果安装了内核源程序，在任意一个源程序的目录下都可以找到 COPYING.LIB 文件的一个复制。

LGPL 允许在自己的应用程序中使用程序库，即使不公开自己的源代码。但是，LGPL 还规定，用户必须能够获得在应用程序中使用的程序库的源代码，并且允许用户对这些程序库进行修改。

大多数 Linux 程序库，包括 C 程序库（libc.a）都属于 LGPL 范畴。因此，如果在 Linux 环境下，使用 GCC 编译器建立自己的应用程序，程序所连接的多数程序库是受 LGPL 保护的。如果想以二进制的形式发布自己的应用程序，则必须注意遵循 LGPL 有关规定。

遵循 LGPL 的一种方法是，随应用程序一起发布目标代码，并发布这些目标程序和受 LGPL 保护的、更新的 Linux 程序库连接起来的 makefile 文件。

遵循 LGPL 的另一种方法是使用动态连接。使用动态连接时，即使程序在运行中调用函数库中的函数，应用程序本身和函数库也是不同的实体。通过动态连接，用户可以直接使用更新后的函数库，而不用对应用程序重新连接。

1.2 Linux 开发初步

1.2.1 Linux 下 C 程序标准

在 Linux 操作系统下进行 C 程序开发的标准主要有两个：ANSI C 标准和 POSIX 标准。

ANSI C 标准是 ANSI（美国国家标准局）于 1989 年制定的 C 语言标准，后来被 ISO（国际标准化组织）接受为标准，因此也称为 ISO C。

POSIX 标准是最初由 IEEE 开发的标准族，部分已经被 ISO 接受为国际标准。

1. ANSI C

ANSI C 的目标是为各种操作系统上的 C 程序提供可移植性保证(例如 Linux 与 Windows 之间)，而不仅仅限于类 UNIX 系统。该标准不仅定义了 C 编程语言的语法和语义，而且还定义了一个标准库。ISO C 标准定义的头文件如表 1-1 所示。

表 1-1　　ISO C 标准定义的头文件

头 文 件	说　明	头 文 件	说　明
<assert.h>	验证程序断言	<signal.h>	信号
<complex.h>	支持复数算术运算	<stdarg.h>	可变参数表
<ctype.h>	字符类型	<stdbool.h>	布尔类型和值
<errno.h>	出错码	<stddef.h>	标准定义
<fenv.h>	浮点环境	<stdint.h>	整型
<float.h>	浮点常量	<stdio.h>	标准 I/O 库
<inttypes.h>	整型格式转换	<stdlib.h>	实用程序库函数
<iso646.h>	替代关系操作符宏	<string.h>	字符串操作
<limits.h>	实现常量	<tgmath.h>	通用类型数学宏
<locale.h>	局部类别	<time.h>	时间和日期
<math.h>	数学常量	<wchar.h>	扩展的多字节和宽字符支持
<setjmp.h>	非局部 goto	<wctype.h>	宽字符分类和映射支持

2. POSIX 标准

POSIX.1 和 POSIX.2 分别定义了兼容操作系统的 C 语言系统接口以及工具标准。Posix 是类 UNIX 系统都遵循的标准，例如 Ubuntu 下和 RedHat 下没有代码不需要移植可直接编译后执行，而在 Linux 下基于 Posix 标准完成的代码就无法在Windows 下直接编译并执行。表 1-2 所示为 26 项 POSIX 标准定义的头文件，表 1-3 所示为 26 项 POSIX 标准定义的 XSI 扩展头文件，表 1-4 所示为 8 项 POSIX 标准定义的可选头文件。

表 1-2　　26 项 POSIX 标准定义的头文件

头 文 件	说　明	头 文 件	说　明
<dirent.h>	目录项	<arpa/inet.h>	Internet 定义
<fcntl.h>	文件控制	<net/if..h>	套接字本地接口
<fnmatch.h>	文件名匹配类型	<netinet/in.h>	Internet 地址族
<glob.h>	路径名模式匹配类型	<netinet/tcp.h>	传输控制协议定义
<grp.h>	组文件	<sys/mman.h>	内存管理声明
<netdb.h>	网络数据库操作	<sys/select.h>	select 函数
<pwd.h>	口令文件	<sys/socket.h>	套接字接口
<regex.h>	正则表达式	<sys/stat.h>	文件状态
<tar.h>	tar 归档值	<sys/times.h>	进程时间
<termios.h>	终端 I/O	<sys/types.h>	基本系统数据类型
<unistd.h>	符号常量	<sys/un.h>	UNIX 域套接字定义
<utime.h>	文件时间	<sys/utsname.h>	系统名
<wordexp.h>	字扩展类型	<sys/wait.h>	进程控制

表 1-3　　26 项 POSIX 标准定义的 XSI 扩展头文件

头 文 件	说　明	头 文 件	说　明
<cpio.h>	cpio 归档值	<syslog.h>	系统出错日志记录
<dlfcn.h>	动态连接	<ucontext.h>	用户上下文
<fmtmsg.h>	消息显示结构	<ulimit.h>	用户限制
<ftw.h>	文件树漫游	<utmpx.h>	用户账户数据库
<iconv.h>	代码集转换实用程序	<sys/ipc.h>	IPC
<langinfo.h>	语言信息常量	<sys/msg.h>	消息队列
<libgen.h>	模式匹配函数定义	<sys/resource.h>	资源操作
<monetary.h>	货币类型	<sys/sem.h>	信号量
<ndbm.h>	数据库操作	<sys/shm.h>	共享存储
<nl_types.h>	消息类别	<sys/statvfs.h>	文件系统信息
<poll.h>	轮询函数	<sys/time.h>	时间类型
<search.h>	搜索表	<sys/timeb.h>	附加的日期和时间定义
<strings.h>	字符串操作	<sys/uio.h>	矢量 I/O 操作