Linux

Nguyễn Quản Bá Hồng*

Ngày 5 tháng 7 năm 2025

Tóm tắt nội dung

This text is a part of the series *Some Topics in Advanced STEM & Beyond*:
URL: https://nqbh.github.io/advanced_STEM/.
Latest version:

- .
  PDF: URL: .pdf.
  TEX: URL: .tex.
- .
  PDF: URL: .pdf.
  TEX: URL: .tex.

Mục lục

1 Linux Programming

1.1 [Per21]. Jack-Benny Persson. Linux System Programming Techniques: Become a Proficient Linux System Programmer Using Expert Recipes & Techniques

- Amazon review. Find solutions to all your problems related to Linux system programming using practical recipes for developing your own system programs.

  Key features.

  - Develop a deeper understanding of how Linux system programming works
  - Gain hands-on experience of working with different Linux projects with help of practical examples
  - Learn how to develop your own programs for Linux

- About Author. Jack-Benny Persson is a consultant & author based in Sweden. He has written several books about Linux & programming. His passion for Linux & other Unix-like systems started as a hobby > 20 years ago. Since then, he has spent most of his spare time reading about Linux, tinkering with Linux servers, & writing about Linux administration. Today he has his own IT & media company in Sweden that focuses in Linux. Jack-Benny holds an Advanced Higher Vocational Education Diploma as a Linux system specialist. He has also studied electronics, networking, & security.

- Preface. Linux system programming is all about developing system programs for Linux OS. Linux is world's most popular open-source OS & runs on everything from big servers to small Internet of Things (IoT) devices. Knowing how to write system programs for Linux will enable you to extend OS & connect it with other programs & systems.

  – Lập trình hệ thống Linux là tất cả về việc phát triển các chương trình hệ thống cho HĐH Linux. Linux là HĐH nguồn mở phổ biến nhất thế giới & chạy trên mọi thứ từ máy chủ lớn đến các thiết bị Internet vạn vật (IoT) nhỏ. Biết cách viết chương trình hệ thống cho Linux sẽ cho phép bạn mở rộng HĐH & kết nối nó với các chương trình & hệ thống khác.

*A scientist- & creative artist wannabe, a mathematics & computer science lecturer of Department of Artificial Intelligence & Data Science (AIDS), School of Technology (SOT), UMT Trường Đại học Quản lý & Công nghệ TP.HCM, Hồ Chí Minh City, Việt Nam.
E-mail: nguyenquanbahong@gmail.com & hong.nguyenquanba@umt.edu.vn. Website: https://nqbh.github.io/. GitHub: https://github.com/NQBH.

Start by learning how to make our programs easy to script & easy to interact with other programs. When write system programs for Linux, should always strive to make them small & do 1 thing only – & do it well. This is 1 of key concepts in Linux: to create small programs that can exchange data with each other in simple ways.

– Bắt đầu bằng cách học cách làm cho chương trình của chúng ta dễ viết kịch bản & dễ tương tác với các chương trình khác. Khi viết chương trình hệ thống cho Linux, hãy luôn cố gắng làm cho chúng nhỏ & chỉ làm 1 việc – & làm tốt. Đây là 1 trong những khái niệm chính trong Linux: tạo các chương trình nhỏ có thể trao đổi dữ liệu với nhau theo những cách đơn giản.

Take a deep dive into C & look at how compiler works, what linker does, how to write Makefiles, & much more.

– Tìm hiểu sâu hơn về C & xem trình biên dịch hoạt động như thế nào, trình liên kết thực hiện chức năng gì, cách viết Makefile, & nhiều hơn nữa.

Learn all about forking & daemons. Also create our own daemon. Then put our daemon under `systemd`'s control. This will enable us to start, stop, & restart daemon using built-in Linux tools.

– Tìm hiểu tất cả về forking & daemon. Cũng như tạo daemon của riêng chúng ta. Sau đó, đặt daemon của chúng ta dưới sự kiểm soát của `systemd`. Điều này sẽ cho phép chúng ta khởi động, dừng, & khởi động lại daemon bằng các công cụ Linux tích hợp.

Learn how to make our processes exchange information using different kinds of *Inter-Process Communication (IPC)*. Also take a look at how to write threaded programs.

– Tìm hiểu cách thực hiện các quy trình trao đổi thông tin bằng các loại Giao tiếp giữa các quy trình (IPC) khác nhau. Ngoài ra, hãy xem cách viết các chương trình có luồng.

At end of this book, cover how to debug our programs using *GNU Debugger (GDB)* & Valgrind. By end of this book, able to write a wide variety of system programs for Linux – everything from filters to daemons.

– Vào cuối cuốn sách này, sẽ trình bày cách gỡ lỗi chương trình của chúng tôi bằng *GNU Debugger (GDB)* & Valgrind. Vào cuối cuốn sách này, có thể viết nhiều chương trình hệ thống khác nhau cho Linux – mọi thứ từ bộ lọc đến daemon.

- Who this book is for. This book is intended for anyone who wants to develop system programs for Linux & wants to have a deep understanding of Linux system. Anyone facing any issues related to a particular part of Linux system programming & looking for some specific recipes or solutions can take advantage of this book.

  – Cuốn sách này dành cho ai. Cuốn sách này dành cho bất kỳ ai muốn phát triển chương trình hệ thống cho Linux & muốn hiểu sâu về hệ thống Linux. Bất kỳ ai gặp phải bất kỳ vấn đề nào liên quan đến một phần cụ thể của lập trình hệ thống Linux & đang tìm kiếm một số công thức hoặc giải pháp cụ thể đều có thể tận dụng cuốn sách này.
- What this book covers. p. 2+++
- To get most out of this book. To get most out of this book, need a basic understanding of Linux, some basic commands, be familiar with moving around filesystem, & installing new programs. It would help if you also have a basic understanding of programming, preferably C language.

  – Để tận dụng tối đa cuốn sách này, bạn cần có hiểu biết cơ bản về Linux, một số lệnh cơ bản, quen thuộc với việc di chuyển hệ thống tệp, & cài đặt chương trình mới. Sẽ hữu ích nếu bạn cũng có hiểu biết cơ bản về lập trình, tốt nhất là ngôn ngữ C.

  Will need a Linux computer with root access – either via `su` or `sudo` – to complete all recipes. Also need to install GCC compiler, Make tool, GDB, Valgrind, & some others smaller tools. Particular Linux distribution doesn't matter that much. There are installation instructions in book for these programs for Debian, Ubuntu, CentOS, Fedora, & Red Hat.

  – Sẽ cần một máy tính Linux có quyền truy cập root – thông qua `su` hoặc `sudo` – để hoàn thành tất cả các công thức. Cũng cần cài đặt trình biên dịch GCC, công cụ Make, GDB, Valgrind, & một số công cụ nhỏ hơn khác. Bản phân phối Linux cụ thể không quan trọng lắm. Có hướng dẫn cài đặt trong sách cho các chương trình này dành cho Debian, Ubuntu, CentOS, Fedora, & Red Hat.

  Can download example code files for this book from Github at https://github.com/PacktPublishing/Linux-System-Programmi
- Conventions used. There are a number of text conventions used throughout this book. `Code in text`: Indicates code words in text, directories, filenames, file extensions, pathnames, dummy URLs, user input, & so on, e.g., Copy `libprime.so.1` to `/usr/local/lib`." Any command-line input or output is written as follows:

  ```
  $> mkdir cube
  $> cd cube
  ```

  In numbered listings, command-line input is set in bold. `$>` characters indicate prompt & aren't sth you should write, an example of a numbered listing:

  ```
  $> ./a.out
  Hello, world!
  ```

  Long command lines that don't fit on a single line are broken up using \ character. This is same character as you use to break along lines in Linux shell. Line under it has a > character to indicate: line is a continuation of prev line. > character is not sth you should write; Linux shell will automatically put this character on a new line where last line was broken up with a \ character, e.g.:

```
$> ./exist.sh /asdf &> /dev/null; \
> if [ $? -eq 3 ]; then echo "That doesn't exist"; fi
That doesn't exist
```

- 1. Getting Necessary Tools & Writing 1st Linux Programs.
- 2. Making Programs Easy to Script.
- 3. Diving Deep into C in Linux. Time to take an in-depth look at C programming in Linux. Learn more about compiler, 4 stages from source code to binary program, how to use Make tool, & differences between system calls & a standard library functions. Also take a look at some essential header files when it comes to Linux, & look at some C & Portable Operating System Interface (POSIX) standards. C is tightly integrated with Linux, & mastering C will help you understand Linux.

  – Đã đến lúc xem xét sâu hơn về lập trình C trong Linux. Tìm hiểu thêm về trình biên dịch, 4 giai đoạn từ mã nguồn đến chương trình nhị phân, cách sử dụng công cụ Make, & sự khác biệt giữa các lệnh gọi hệ thống & các hàm thư viện chuẩn. Ngoài ra, hãy xem một số tệp tiêu đề cần thiết khi nói đến Linux, & xem một số tiêu chuẩn C & Giao diện hệ điều hành di động (POSIX). C được tích hợp chặt chẽ với Linux, & thành thạo C sẽ giúp bạn hiểu Linux.

  In this chap, develop both programs & libraries for Linux. Also write both a generic Makefile & more advanced ones for more significant projects. While doing this, also learn about different C standards, why they matter, & how they affect your programs.

  – Trong chương này, hãy phát triển cả chương trình & thư viện cho Linux. Cũng viết cả Makefile chung & các Makefile nâng cao hơn cho các dự án quan trọng hơn. Trong khi thực hiện việc này, hãy tìm hiểu về các tiêu chuẩn C khác nhau, lý do tại sao chúng quan trọng, & cách chúng ảnh hưởng đến chương trình của bạn.

  This chapter will cover the following recipes:

  - Linking against libraries using GNU Compiler Collection (GCC)
  - Changing C standards
  - Using system calls
  - Understand when not to use them
  - Getting information about Linux- & Unix-specific header files
  - Defining feature test macros
  - Looking at 4 stages of compilation
  - Compiling with Make
  - Writing a generic Makefile with GCC options
  - Writing a simple Makefile
  - Writing a more advanced Makefile
- 4. Handling Errors in Programs.
- 5. Working with Filo I/O & Filesystem Operations.
- 6. Spawning Processes & Using Job Control.
- 7. Using `systemd` to Handle Your Daemons.
- 8. Creating Shared Libraries.
- 9. Terminal I/O & Changing Terminal Behavior.
- 10. Using Different Kinds of IPC.
- 11. Using Threads in Your Programs.
- 12. Debugging Your Programs.

2 Miscellaneous

Tài liệu

[Per21] Jack-Benny Persson. "Linux System Programming Techniques: Become a Proficient Linux System Programmer Using Expert Recipes & Techniques". In: (2021), p. 432.