

Preface

Introduction

This book describes the systems programming interface to the Solaris operating system. Despite the word "Solaris" in the title, this book is suitable for programmers of any UNIX or UNIX-like operating system (that being said, some of the features we describe *are* specific to Solaris).

Like most operating systems, Solaris provides a huge number of services to programs. Examples include: opening a file, reading that file, allocating memory, getting the current time, starting a new program, and so on. We describe many public interfaces, but unfortunately, despite our best intentions, we can't cover *everything*. (A book that tried to cover everything would be so big as to be unwieldy, and would likely never be finished!)

Because the use of undocumented interfaces is hindersome to writing portable programs, we do not describe them. Also, the use of private, undocumented interfaces voids Sun's Application Certification, since these interfaces can change without notice from release to release.

The functions we describe in this text (nearly 540 of them) are documented in Sections 2 and 3 of the Solaris *Reference Manual Collection*. However, those manual pages do not provide background material and full examples: that is what this book provides.

Audience

This book can be used both as a tutorial for novice and intermediate programmers, and as a reference for experienced programmers. It is also suitable for use as a text for undergraduate or graduate level programming courses.

The book assumes that the reader has some experience with C programming (although not necessarily on the UNIX platform), and has at least a passing familiarity of UNIX concepts like interacting with the shell, editing programs with a text editor, and pipelines.

Although it isn't mandatory, access to a Solaris system on which the examples can be tried is highly recommended.

Organization of the Book

This book is divided into six parts:

- Part 1, *Introduction*. This part contains introductory and historical background information. It consists of two chapters:
 - Chapter 1, *Introduction*, is an overview and introduction to many basic UNIX programming concepts and terminology.
 - Chapter 2, *A Brief History of Solaris*, contains a description of how Solaris has evolved over the years, and it also describes the various standards to which Solaris complies.
- Part 2, *Fundamental Topics*. This part describes the interfaces and topics essential to UNIX programming, and consists of seven chapters:
 - Chapter 3, *Utility Functions*, describes most of the utility functions provided for manipulating character classes, character strings, and byte arrays. It also talks about dynamic memory, temporary files, parsing command line arguments, error reporting, and suspending a process. Many readers will be familiar with most of this material, but it provides a common starting point from which we can build.
 - Chapter 4, *Basic File I/O*, describes low-level file I/O—how to open and close files, how to read and write them, and how to change the current file offset. It discusses file sharing, and the buffering that programs using these functions must perform for themselves.
 - Chapter 5, *The Standard I/O Library*, describes the high-level I/O functions that are provided by the standard I/O library.
 - Chapter 6, *Date and Time Operations*, describes how to get and set the system clock, and the various functions for converting time to and from various formats.
 - Chapter 7, *Users and Groups*, describes the format of the user, group, and password files and how to obtain information from them. It also explains how to determine who is currently logged into the system, and when a user last logged in or out.
 - Chapter 8, *System Information and Resource Limits*, describes how to get and set various system and user resource limits, such as the amount of

CPU time a process may consume, the maximum number of open files per process, and information such as the system's hostname, how much memory is installed, and so on.

- Chapter 9, *Secure C Programming*, describes several common security flaws in programs (e.g., buffer overflows) and presents some tips for writing programs that are secure by design.
- Part 3, *Input/Output*. This part describes the I/O facilities offered by Solaris and consists of four chapters:
 - Chapter 10, *Files and Directories*, describes the features of files and directories, and the function available to manipulate them. It describes concepts such as the various file types; file permissions; resolving symbolic links; creating, reading, and removing directories; as well as set-user-ID and set-group-ID programs.
 - Chapter 11, *Working With File Systems*, explains how to read the on-disk data structures for file systems, how to mount and unmount them, and how to read the mounted file system table.
 - Chapter 12, *Terminal I/O*, talks about terminal I/O, including special input characters, and about examining and modifying terminal attributes. It also describes the functions used to manipulate serial lines (e.g., changing the baud rate or number of bits per character).
 - Chapter 13, *Advanced I/O*, covers more advanced I/O topics, such as record locking, nonblocking, multiplexed, and asynchronous I/O. It also discusses memory mapped files, access control lists, and extended file attributes.
- Part 4, *Processes and Process Control*. This part is about processes and how to control them. It consists of five chapters:
 - Chapter 14, *The Environment of a UNIX Process*, describes the environment in which a UNIX process runs. It also describes process start-up and termination.
 - Chapter 15, *Process Control*, describes how to create new processes and how to start another program running. It also discusses how to wait for the termination status of a process, how to avoid race conditions, and process accounting.
 - Chapter 16, *Process Relationships*, explains the concepts of sessions, process groups, and controlling terminals.
 - Chapter 17, *Signals*, describes the concepts of reliable and unreliable signals, including how to send them, catch them, block them, and ignore them.
 - Chapter 18, *Daemon Processes*, describes the characteristics of a daemon and shows how a process can become a daemon. It also describes the

facilities for logging messages and how to start only one copy of a daemon.

- Part 5, *Interprocess Communication*. This part describes the facilities by which processes may communicate with each other. It consists of four chapters:
 - Chapter 19, *Interprocess Communication Using Pipes and FIFOs*, describes the oldest, and possibly most oft used, methods of interprocess communication: pipes and FIFOs. It also describes the differences between iterative and concurrent servers.
 - Chapter 20, *The System V Interprocess Communication Facility*, describes System V message queues, semaphore sets, and shared memory segments.
 - Chapter 21, *Advanced Interprocess Communication*, describes the concepts of passing file descriptors to other processes, both related and unrelated. It also describes how to attach a detach a pipe to or from a pathname in the file system.
 - Chapter 22, *Doors*, describes Solaris doors, which facilitate fast remote procedure calls between processes on the same host.
- Part 6, *Pseudo Terminals*. This part describes pseudo terminals and consists of just one chapter:
 - Chapter 23, *Pseudo Terminals*, describes what pseudo terminals are and shows how to create them. From that, it shows how to execute other programs via a pseudo terminal, which is useful for scripting programs that would otherwise have to be run interactively.

There are also five appendices:

- Appendix A, *An Internationalization and Localization Primer*, provides a brief introduction to the subjects of internationalization and localization. That is, it describes the concepts of locales and how to write programs that are portable to different languages and regions.
- Appendix B, *The BSD Source Compatibility Package*, very briefly describes the BSD Source Compatibility Package, which is a transitional tool designed to enable programs written to the BSD-based SunOS 4.x APIs to be compiled.
- Appendix C, *Function Summary*, presents a summary of all the functions we describe in this text. As well as showing a function prototype for each function (including any return values), the availability of each function—by Solaris version and standard—is also tabulated.
- Appendix D, *Miscellaneous Source Code*, shows the source code for the header file we include in most of the example programs. It also presents the source code of the library functions we developed for use in the example programs.

- Appendix E, *Solutions to Selected Exercises*, shows possible solutions to many of the numerous end of chapter exercises.

To aid in use as a reference, a thorough index is provided, along with a comprehensive bibliography. To help those reading the book in a random order, numerous cross references to related topics are provided throughout the text.

Source Code and Errata Availability

The source code for all the examples presented in the text is available from the author's home page (the URL is shown at the end of this preface). The best way to learn the interfaces and techniques described in this book is to take these programs and modify them. Actually writing code, perhaps using these examples as a starting point, is the only way to fully understand these concepts and techniques.

All of the examples in this text have been tested on several systems (including a SPARCstation 20, a SPARCstation Voyager, a couple of Ultra 1s, a Sun Blade 100, and an Ultra 60), running various versions of Solaris. They were compiled using version 5.4 of Sun's C compiler, and version 2.95.2 of `gcc`. The examples have also been tested on a number of systems running Solaris x86.

A current errata for this book is available from the author's home page.

Acknowledgements

I am indebted to my family for their love and support over the last three years. Thank you Jenny and Judge (our canine child) for putting up with the long and weird hours that go into writing a book. Now that it is finally finished, they can have their husband and daddy back. I am also indebted to other members of my family and friends for their support and encouragement while writing this book: thank you everyone!

My thanks to the following technical reviewers, who provided invaluable feedback catching lots of errors, pointing out areas that needed more detailed explanation, and suggesting different wording, presentation, and code: Philip Brown, Alan Coopersmith, Casper Dik, Stefaan Eeckels, Peter Baer Galvin, Alexander Gelfenbain, Anthony Mandic, Chris Morgan, David Robinson, and especially Dragan Cvetković, who reviewed the entire manuscript.

The following people took time from their busy schedules to answer my (sometimes *many*) email questions, all of which helped improve the accuracy and presentation of the text: Dave Butenhof, Dennis Clarke, Alan Coopersmith, Casper Dik, Darren Dunham, Bill Fenner, Markus Gyger, Zhishun Alex Liu, Darren Moffat, Jim Moore, Alec Muffet, Greg Onufer, David Robinson, Karl Schendel, Andy Tanenbaum, and Tony Walton.

Special thanks to W. Richard Stevens for his inspiration and encouragement when this book was just a twinkle in the author's eyes. Thanks too to Michael Slaughter from Addison-Wesley for making the first contact and for being the catalyst that made all this happen.

My thanks to Bill Moffitt and Karen Hill for arranging for me to join the Solaris 9 Beta program. Thanks also to Sun Microsystems for supplying a license for the SunONE Studio 7 Compiler Collection.

Last, but by no means least, many thanks to the wonderful staff at Prentice Hall: Kathleen Caren, Raquel Kaplan, and especially my editor, Greg Doench; their seemingly infinite patience with my increases in page count, and many slips of the due date (not to mention, letting me do things "my way") is very much appreciated. (The author, being originally from England and currently living in Canada, uses British spelling and punctuation.)

Colophon

This book continues the time-honoured tradition of writing *real* UNIX books using `vi` and `troff`. I produced camera-ready PostScript using James Clark's excellent `groff` package on a Sun Ultra 60 running build 60 of Solaris 10. I typed in all 417,259 words using the `vi` editor, created the 84 illustrations using the `gpic` program, produced the 91 tables using the `gtbl` program, performed all the indexing (using a set of `awk` scripts written by Jon Bentley and Brian Kernighan), and did the final page layout. My own script, `c2ms`, the `expand` program, the `n1` utility, and one of my `sed` scripts were used to include the 12,619 lines (in 281 programs) of C source code in the book.

I welcome email from any readers with comments, suggestions, or bug fixes.

Kelowna, British Columbia
July 2004

Rich Teer
rich.teer@rite-group.com
<http://www.rite-group.com/rich>