About This Document

The purpose of this document is to provide fundamental high-level information about the OS X core operating-system architecture. It also provides background for system programmers and developers of device drivers, file systems, and network extensions. In addition, it goes into detail about topics of interest to kernel programmers as a whole.

This is *not* a document on drivers. It covers device drivers at a high level only. It does, however, cover some areas of interest to driver writers, such as crossing the user-kernel boundary. If you are writing device drivers, you should primarily read the document *IOKit Fundamentals*, but you may still find this document helpful as background reading.

Who Should Read This Document

This document has a wide and diverse audience—specifically, the set of potential system software developers for OS X, including the following sorts of developers:

- device-driver writers
- network-extension writers
- file-system writers
- developers of software that modifies file system data on-the-fly
- system programmers familiar with BSD, Linux, and similar operating systems
- developers who want to learn about kernel programming

If you fall into one of these categories, you may find this document helpful. It is important to stress the care needed when writing code that resides in the kernel, however, as noted in Keep Out.

Road Map

The goal of this document is to describe the various major components of OS X at a conceptual level, then provide more detailed programming information for developers working in each major area. It is divided into several parts.

The first part is a kernel programming overview, which discusses programming guidelines that apply to all aspects of kernel programming. This includes issues such as security, SMP safety, style, performance, and the OS X kernel architecture as a whole. This part contains the chapters Keep Out, Kernel Architecture Overview, The Early Boot Process, Security Considerations, Performance Considerations, and Kernel Programming Style.

The next part describes Mach and the bootstrap task, including information about IPC, bootstrap contexts, ports and port rights, and so on. This includes the chapters Mach Overview, Memory and Virtual Memory, Mach Scheduling and Thread Interfaces, and Bootstrap Contexts.

The third part describes the I/O Kit and BSD. The I/O Kit is described at only a high level, since it is primarily of interest to driver developers. The BSD subsystem is covered in more detail, including descriptions of BSD networking and file systems. This includes the chapters I/O Kit Overview, BSD Overview, File Systems Overview, and Network Architecture.

The fourth part describes kernel services, including boundary crossings, synchronization, queues, clocks, timers, shutdown hooks, and boot option handling. This includes the chapters Boundary Crossings,

Synchronization Primitives, and Miscellaneous Kernel Services.

The fifth part explains how to build and debug the kernel and kernel extensions. This includes the chapters Kernel Extension Overview and Building and Debugging Kernels.

Each part begins with an overview chapter or chapters, followed by chapters that address particular areas of interest.

The document ends with a glossary of terms used throughout the preceding chapters as well as a bibliography which provides numerous pointers to other reference materials.

Glossary terms are highlighted in bold when first used. While most terms are defined when they first appear, the definitions are all in the glossary for convenience. If a term seems familiar, it probably means what you think it does. If it's unfamiliar, check the glossary. In any case, all readers may want to skim through the glossary, in case there are subtle differences between OS X usage and that of other operating systems.

The goal of this document is very broad, providing a firm grounding in the fundamentals of OS X kernel programming for developers from many backgrounds. Due to the complex nature of kernel programming and limitations on the length of this document, however, it is not always possible to provide introductory material for developers who do not have at least some background in their area of interest. It is also not possible to cover every detail of certain parts of the kernel. If you run into problems, you should join the appropriate Darwin discussion list and ask questions. You can find the lists at http://www.lists.apple.com/.

For this reason, the bibliography contains high-level references that should help familiarize you with some of the basic concepts that you need to understand fully the material in this document.

This document is, to a degree, a reference document. The introductory sections should be easily read, and we recommend that you do so in order to gain a general understanding of each topic. Likewise, the first part of each chapter, and in many cases, of sections within chapters, will be tailored to providing a general understanding of individual topics. However, you should not plan to read this document cover to cover, but rather, take note of topics of interest so that you can refer back to them when the need arises.

Other Apple Publications

This document, *Kernel Programming*, is part of the Apple Reference Library. Be sure to read the first document in the series, *Mac Technology Overview*, if you are not familiar with OS X.

You can obtain other documents from the Apple Developer Documentation website at http://developer.apple.com/documentation.

Mach API Reference

If you plan to do extensive work inside the OS X kernel, you may find it convenient to have a complete Mach API reference, since this document only documents the most common and useful portions of the Mach API. In order to better understand certain interfaces, it may also be helpful to study the implementations that led up to those used in OS X, particularly to fill in gaps in understanding of the fundamental principles of the implementation.

OS X is based on the Mach 3.0 microkernel, designed by Carnegie Mellon University, and later adapted to the Power Macintosh by Apple and the Open Software Foundation Research Institute (now part of Silicomp). This was known as osfmk, and was part of MkLinux (http://www.mklinux.org). Later, this and code from OSF's commercial development efforts were incorporated into Darwin's kernel. Throughout this evolutionary process, the Mach APIs used in OS X diverged in many ways from the original CMU Mach 3 APIs.

You may find older versions of the Mach source code interesting, both to satisfy historical curiosity and to avoid remaking mistakes made in earlier implementations. MkLinux maintains an active CVS repository with their recent versions of Mach kernel source code. Older versions can be obtained through various Internet sites. You can also find CMU Mach white papers by searching for Mach on the CMU computer science department's website (http://www.cs.cmu.edu), along with various source code samples.

Up-to-date versions of the Mach 3 APIs that OS X provides are described in the Mach API reference in the kernel sources. The kernel sources can be found in the xnu project on http://kernel.macosforge.org/.

Information on the Web

Apple maintains several websites where developers can go for general and technical information on OS X.

- Apple Developer Connection: Developer Documentation (http://developer.apple.com/documentation). Features the same documentation that is installed on OS X, except that often the documentation is more up-to-date. Also includes legacy documentation.
- Apple Developer Connection: OS X (http://developer.apple.com/devcenter/mac/). Offers SDKs, release notes, product notes and news, and other resources and information related to OS X.
- AppleCare Tech Info Library (http://www.apple.com/support/). Contains technical articles, tutorials, FAQs, technical notes, and other information.