前言

本书(简称 CS:APP)的主要读者是计算机科学家、计算机工程师，以及那些想通过学习计算机系统的内在运作而能够写出更好程序的人。

我们的目的是解释所有计算机系统的本质概念，并向你展示这些概念是如何实实在在地影响应用程序的正确性、性能和实用性的。其他的系统类书籍都是从构建者的角度来写的，讲述如何实现硬件或系统软件，包括操作系统、编译器和网络接口。而本书是从程序员的角度来写的，讲述应用程序员如何能够利用系统知识来编写出更好的程序。当然，学习一个计算机系统应该做些什么，是学习如何构建一个计算机系统的很好的出发点，所以，对于希望继续学习系统软硬件实现的人来说，本书也是一本很有价值的介绍性读物。大多数系统书籍还倾向于重点关注系统的某一个方面，比如：硬件架构、操作系统、编译器或者网络。本书则以程序员的视角统一覆盖了上述所有方面的内容。

如果你研究和领会了这本书里的概念，你将开始成为极少数的“牛人”，这些“牛人”知道事情是如何运作的，也知道当事情出现故障时如何修复。你写的程序将能够更好地利用操作系统和系统软件提供的功能，对各种操作条件和运行时参数都能正确操作，运行起来更快，并能避免出现使程序容易受到网络攻击的缺陷。同时，你也要做好更深入探究的准备，研究像编译器、计算机体系结构、操作系统、嵌入式系统、网络互联和网络安全这样的高级题目。

读者应具备的背景知识

本书的重点是执行 x86-64 机器代码的系统。对英特尔及其竞争对手而言，x86-64 是他们自 1978 年起，以 8086 微处理器为代表，不断进化的最新成果。按照英特尔微处理器产品线的命名规则，这类微处理器俗称为“x86”。随着半导体技术的演进，单芯片上集成了更多的晶体管，这些处理器的计算能力和内存容量有了很大的增长。在这个过程中，它们从处理 16 位字，发展到引入 IA32 处理器处理 32 位字，再到最近的 x86-64 处理 64 位字。

我们考虑的是这些机器如何在 Linux 操作系统上运行 C 语言程序。Linux 是众多继承自最初由贝尔实验室开发的 Unix 的操作系统中的一种。这类操作系统的其他成员包括 Solaris、FreeBSD 和 MacOS X。近年来，