第 11 章　关于 CPU 虚拟化的总结对话

教授：那么，同学，你学到了什么？

学生：教授，这似乎是一个既定答案的问题。我想你只想让我说“是的，我学到了”。

教授：确实。但这也还是一个诚实的问题。来吧，让教授休息一下，好吗？

学生：好的，好的。我想我确实学到了一些知识。首先，我了解了操作系统如何虚拟化 CPU。为了理解这一点，我必须了解一些重要的机制（mechanism）：陷阱和陷阱处理程序，时钟中断以及操作系统和硬件在进程间切换时如何谨慎地保存和恢复状态。

教授：很好，很好！

学生：虽然所有这些交互似乎有点复杂，但我怎样才能学到更多的内容？

教授：好的，这是一个很好的问题。我认为没有办法可以替代动手。仅阅读这些内容并不能给你正确的理解。做课堂项目，我敢保证，它会对你有所帮助。

学生：听起来不错。我还能告诉你什么？

教授：那么，你在寻求理解操作系统的基本机制时，是否了解了操作系统的哲学？

学生：嗯……我想是的。似乎操作系统相当偏执。它希望确保控制机器。虽然它希望程序能够尽可能高效地运行 [因此也是受限直接执行（limited direct execution）背后的全部逻辑]，但操作系统也希望能够对错误或恶意的程序说“啊！别那么快，我的朋友”。偏执狂全天控制，并且确保操作系统控制机器。也许这就是我们将操作系统视为资源管理器的原因。

教授：是的，听起来你开始融会贯通了！干得漂亮！

学生：谢谢。

教授：那些机制之上的策略呢？有什么有趣的经验吗？

学生：当然能从中学到一些经验。也许有点明显，但明显也可以是很好。比如将短工作提升到队列前面的想法：自从有一次我在商店买一些口香糖，我就知道这是一个好主意，而且我面前的那个人有一张无法支付的信用卡。我要说的是，他不是“短工作”。

教授：这听起来对那个可怜的家伙有点过分。还有什么吗？

学生：好吧，你可以建立一个聪明的调度程序，试图既像 SJF 又像 RR——MLFQ 相当漂亮。构建真正的调度程序似乎很难。

教授：的确如此。这就是对使用哪个调度程序至今仍有争议的原因。例如，请参阅 CFS、BFS 和 O（1）调度程序之间的 Linux 战斗。不，我不会说出 BFS 的全名。

学生：我不会要求你说！这些策略战争看起来好像可以永远持续下去，真的有一个正确的答案吗？

教授：可能没有。毕竟，即使我们自己的度量指标也不一致。如果你的调度程序周转时间好，那么在响应时间就会很糟糕，反之亦然。正如 Lampson 说的，也许目标不是找到最好的解决方案，而是为了避免灾难。

学生：这有点令人沮丧。

教授：好的工程可以这样。它也可以令人振奋！这只是你的观点，真的。我个人认为，务实是一件好事，实用主义者意识到并非所有问题都有简洁明了的解决方案。你还喜欢什么？

学生：我非常喜欢操控调度程序的概念。我下次在亚马逊的 EC2 服务上运行一项工作时，看起来这可能是需要考虑的事情。也许我可以从其他一些毫无戒心的（更重要的是，对操作系统一无所知的）客户那里窃取一些时间周期！

教授：看起来我可能创造了一个“怪物”！你知道，我可不想被人称为弗兰肯斯坦教授。

学生：但你不就是这样想的吗？让我们对某件事感到兴奋，这样我们就会自己对它进行研究？点燃火，仅此而已？

教授：我想是的。但我不认为这会成功！