OS 论文阅读（1）

刘炜
2014210878

May 31, 2015

1 Arrakis: The Operating System is the Control Plane [1]

1.1 思路

这篇文章是 OSDI'14 的 Best Paper。描述的场景是数据中心使用的 OS 的处理 I/O 的性能跟不上硬件的读写处理速度。因此，作者思考的解决方法是将 Kernel 解放出来，将 I/O 直接面向应用，进而减小了 Kernel 的 overhead。

具体来说，OS Kernel 与 I/O 相关的处理可以分为 API，复用，命名，资源限制，访问控制，I/O 调度，I/O 处理，复制和保护这几类。解放 Kernel，就是将这些工作中与数据访问相关的，移出 Kernel，只保留控制相关的功能。

复用，保护和 I/O 调度这部分的工作，作者将它们移到硬件上利用硬件虚拟化来实现。

1. SR-IOV: 实现多个虚拟机共享同一个物理机下进行高速 I/O 的技术，这里可以用来实现复用。
2. IOMMU: 通过地址转换控制设备使用的用户级虚拟内存
3. 分组过滤，逻辑磁盘：保证访问的 I/O 都是合格的
4. 限速器，分组调度：进行 I/O 调度

如此，则 Kernel 的工作主要变成了命名和访问控制，将硬件直接面向应用，就没有文件系统的抽象了。因此，需要 Kernel 在配置数据部分时实现访问控制，并利用 Kernel 中的虚拟文件系统提供命名的服务给应用。

剩下的 API 和 I/O 处理被移到了应用层。作者将这些功能封装成链接库供引用调用直接访问硬件。最后的对比测试部分，延迟有 2 ～ 5 倍的优化，吞吐率有约 9 倍的优化。

1.2 讨论

之前的相关工作的侧重点都在优化传统的文件系统上，通过实现新的文件系统来减少 overhead，本文的亮点在于跳出了文件系统的框框，引入硬件虚拟化的技术来实现 I/O 的各种操作，只利用文件系统来实现命名的一些功能。换个角度，这篇文章可以认为是为硬件虚拟化找到了新的应用场景，这一点上是很创造性的工作。

1.3 体会

这篇文章开始部分研究了网络协议栈，存储栈和应用栈的性能测试，发现数据访问中通用的共享式的 I/O 速度很慢，因此开始考虑将控制通路和数据通路分离，解放 kernel 来实现。idea 的实际应用背景很强，工作也非常扎实，非常值得学习。

2 Ironclad Apps: End-to-End Security via Automated Full-System Verification [2]

2.1 思路

这是一篇验证系统的文章，之前对于验证系统也不是很了解，所以专门选了一篇这样的文章进行学习。

Ironclad 这个系统主要由三部分组成：延迟启动，可信计算和软件验证。文章主要讨论的部分是软件验证。主要的创新点在于使用了 Dafny 语言来书写具体的规范，而 Dafny 语言是原生支持 Z3 求解器的，因此可以快速底层的证明细节。然后作者写了些自动化的工具将 Dafny 的代码转成 BoogieX86 这种可验证的汇编语言，因而可以在汇编语言的层面通过 Boogie 验证器实现整个系统的验证。最大的贡献在于效率，之前的系统普遍的 overhead 大于 25：1, 而 Ironclad 的系统大概只有 4.8：1，虽然离完全应用还有一段距离，但进步已经很明显了。

2.2 讨论

这篇文章号称通过程序验证能解决种种的安全问题，我并不认同。因为从规则的来源看，一方面是编程手册，一方面是 RFC。而我们知道 RFC 不是强制性的规定要求，而且 RFC 很多只是现有实现方案的总结，只能说是没有以往已经报过的漏洞，对于规则本身是否存在逻辑漏洞，就要打上一个问号。验证系统真要保证完备性，复杂规则本身的正确性证明是不可或缺的一部分。

2.3 体会

验证系统的工作需要的底层工作的积累非常多，可能只是不大的改进，能把系统性能做到大幅度的提高就已经是非常大的贡献了。

References

[1] S. Peter et.al, "Arrakis: The Operating System is the Control Plane", In OSDI'2014.

[2] C. Hawblitzel et.al, "Ironclad Apps: End-to-End Security via Automated Full-System Verification", In OSDI'2014