天津大学

本科生毕业设计（论文）开题报告

题目：基于 RISC -V 的 Type-1 Hypervisor 的设计与实现

学　　院　智能与计算学部

专　　业　计算机科学与技术

年　　级　2019 级

姓　　名　齐呈祥

学　　号　3019244189

指导教师　李罡

一、课题的来源及意义

课题的来源：

虚拟化是一个在复杂计算系统和体系结构中被广泛应用的技术。今天我们所认为的大部分云系统都是在数据中心强大的服务器上的虚拟机中相互隔离运行的大量应用程序。这些应用的虚拟化特性十分灵活，可以抽象出特定的硬件模式，因此许多软件可以直接运行在许多系统中。

虚拟化技术的研究来源于对计算机系统性能和效率的不断追求，它可以解决传统物理系统存在的资源浪费、管理复杂、难以扩展等问题。

RISC -V 作为新兴的指令集架构，它的虚拟化技术仍然处于初始阶段，因此研究 RISC -V 的虚拟化技术是很有意义的，它可以促进 RISC -V 处理器架构在服务器领域的发展与应用，因为服务器需要一个稳定的虚拟化软件生态环境；也可以帮助增加 RISC -V 处理器架构的兼容性和灵活性，通过 RISC -V hypervisor 可以支持多种类型的操作系统，例如 Linux、Windows 以及 FreeBSD 等；除此之外，RISC -V hypervisor 可以利用 RISC -V 处理器架构的开放性和可扩展性，实现更高效和更安全的虚拟化机制，例如使用硬件辅助或者微内核技术。除此之外，较小规模的虚拟化也被广泛应用于隔离嵌入式平台上的应用程序，一个例子是 Siemens Jailhouse，它是开源的且可以公开获取。

本次课题来源于我个人对 RISC -V 指令集体系以及虚拟化技术的兴趣以及希望为 RISC -V 指令体系架构的生态做一份自己的贡献。

课题的意义：

本课题将围绕 RISC -V Type-1 hypervisor 进行 ，目标是基于 RISC -V H 扩展设计并实现一款高性能稳定运行的 Type-1 hypervisor。当前国内外学术界和工业界对于 RISC -V Type-1 hypervisor 的研究较少，因此该课题研究有助于促进推动 RISC -V 开源和创新的软硬件生态系统的建立；同时，对于 RISC -V Type-1 hypervisor 的研究有助于为操作系统提供更好的隔离环境。相比于其他移植了 RISC -V 架构的 hypervisor 来说，本课题使用 Rust 语言进行构建，具有内存安全、抽象性强等特性，更便于维护与调试；同时，本课题计划对性能进行调优，相比于现有的 RISC -V 虚拟化软件性能更强。

二、国内外发展状况

虚拟化的概念和技术已经存在了几十年，但在当今的互联和基于云的环境中更加相关。我们可以找到 Gold berg 和 Popek 的论文已经列举出了系统要支持虚拟化功能的一般性需求。

RISC -V 是一个开源的指令集架构，近年来发展迅猛，从一个学术性的新型指令集架构逐步发展为在工业界中广泛被使用的指令集系统。人们对其他架

技术应用到 RISC -V 产生了更广泛的兴趣, 例如 ARM 64 和 X86-64 有它们规范的虚拟化方式，最初的虚拟化并非如此，在 RISC -V 虚拟化草案出现之前，人们在 RISC -V 在虚拟化方面也有一些探索，例如 RVirt 和 diosix 依赖于特权级模式下的陷入和模拟，软件全部接管虚拟化，不需要特殊硬件的支持。

在 2021 年 11 月底, RISC -V H 扩展获得批准并正式采用进入特权架构规范。它定义了可以在核心中实现的硬件功能，以减少虚拟化开销并简化管理程序的实现。

同时国内外学术界和工业界都对 RISC -V 的进展有所研究：

国外在 RISC -V hypervisor 方面的研究主要在规范制定、硬件实现和软件生态等方面。目前，RISC -V 基金会已经发布了最新版本的 hypervisor 扩展(H-extension V1.0)规范，并有多个项目正在开发支持该规范的硬件核心。例如，由 Bruno Sa 和 Hose Martins 等人在 Rocket chip core 上扩展了 RISC -V Hypervisor 的功能并成功运行了虚拟化软件；同时在今年该团队尝试在 RISC -V CAV6 上面支持硬件虚拟化并设计了一个名为 "G-Stage Translation Lookaside Buffer(GTLB L2 TLB)" 来缓解虚拟化性能开销，并且在 FPGA 进行性能评估，取得了良好的性能提升。

同时，也有一些开源的 hypervispr 软件被移植到了 RISC -V 平台上，如 Bao、Xvisor、KVM 等。

除此之外，Rivos 公司最近也在开发一个 salus 系统，salus 是一个使用了 RISC -V H 扩展的虚拟机监控器，用于可信执行环境的开发。salus 在 M 模式下管理所有异常和中断；在 U 模式运行一个安全操作系统, 提供 TEE 服务和接口；在 VS 模式下运行一个主操作系统，通常是 Linux，负责调度、内存分配、设备驱动和虚拟机管理；在 VU 模式下运行多个客户虚拟机，每个虚拟机都有自己的 TEE 实例。

国内在 RISC -V hypervisor 方面的研究较少，但也有一些值得关注的动态，例如，清华大学计算机系与阿里云合作开发了基于 Rocket Chip 核心的 hypervisor 原型系统；中科院计算所与华为合作开发了基于 U54-MC 核心的 hypervisor 原型系统；浙江大学计算机系与紫光展锐合作开发了基于 Xuantie-910 核心的 hypervisor 原型系统。

三、研究目标、研究内容与研究方法

研究目标:

本课题的研究目标是探索 RISC -V 虚拟化的特性以及构建 RISC -V hypervisor 的一般性方法（包括全虚拟化以及 H 扩展辅助虚拟化）。

研究内容:

- 探索如何使用软件虚拟化（即全虚拟化）以及硬件虚拟化（即使用 H 扩展的硬件）在 RISC -V 平台上构建 type-1 hypervisor。
- 与传统虚拟化方法相比，探索 RISC -V H 扩展的特性。
- 探索 Rust 作为系统语言在系统开发过程中的优缺点。
- 设计实现可以在 RISC -V 平台上工作的软件虚拟化以及硬件虚拟化 hypervisor，并将其运行起来 minikernel。
- 继续扩展 hypervisor，可以支持更多外设与特性，运行更加复杂的操作系统，最终目标是可以运行 Linux。

研究方法:

在本课题的研究中为了方便 hypervisor 的迭代与开发，计划使用 QEMU virt 模拟器作为开发工具用于模拟 RISC -V 开发板。同时，本课题基于迭代式开发与测试驱动开发的软件工程思想进行开发，即首先对于某功能构建测试程序，随后在项目中对于测试进行开发，最终可以通过所有测试用例；随后再迭代下一个特性，并对该特性构建测试用例，以此类推。

四、进度安排

- 2022.12.15 - 2023.01.15 学习虚拟化技术，调研 RISC -V 虚拟化在学术界和工业界的进展，阅读 RISC -V H 扩展 1.0 标准，阅读现有的 hypervisor 开源项目。
- 2023.01.15 - 2023.02.15 使用 Rust 构建一个依赖于特权级陷入和模拟的全虚拟化的 Type-1 hypervisor，名为 hypocaust[13]，完成特权级指令陷入与模拟，影子页表的构建，影子页表与客户页表的同步以及设备的模拟与透传，并运行起 minikernel。
- 2023.02.15 - 2023.03.15 完成开题报告与任务书；使用 Rust 语言编写一个使用 H 扩展辅助虚拟化的 hypervisor，名为 hypocaust-2[14]，完成 SBI call 处理，两阶段页表翻译，以及中断转发和设备的透传或模拟，运行起 minikernel 和 rCore-Tutorial-v3。

- 2023.03.15-2023.04.15 继续扩展 hypocaust-2，为其支持 RISC -V IOMMU 驱动实现，支持更多设备虚拟化，运行起 xv6-riscv，争取运行起 Linux。同时完成文献翻译与中期汇报。
- 2023.04.15-2023.05.15 为 hypocaust-2 做性能测试，将其与 RVirt、Xvisor 等项目进行性能比较。
- 2023.05.15-2023.06.15 完成撰写毕业论文。

五、研究或方案的可行性分析

关于实现 RISC -V hypervisor 研究的可行性分析：

- 技术复杂性：开发 hypervisor 是一项复杂且技术上具有挑战性的任务，需要虚拟化，系统架构和底层代码编程的的知识。对于实现者来说需要在这些领域具有强大的技术背景，并愿意投入学习相关技能所需的时间和精力。
- 资源可用性：开发 RISC -V hypervisor 需要访问专门的硬件平台和软件工具，本科生可能不易获取这些工具，所幸我们可以使用 QEMU 代替。
- 项目范围：开发功能齐全的 hypervisor 对于本科毕业项目来说可能过于复杂。因此，专注于 hypervisor 某些部分，例如实施特定的虚拟化技术或提高现有 hypervisor 的性能。
- 项目时间：作为本科毕业设计，本科毕业项目通常有一个有限的时间表，因此需要规划好时间，在规定时间内完成项目设计。

六、主要参考文献

[1] Seshadri A, Luk M, Qu N, et al. SecVisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity OSes[C]//Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles. 2007: 335-350.

[2] Popek G J, Goldberg R P. Formal requirements for virtualizable third generation architectures[J]. Communications of the ACM, 1974, 17(7): 412-421.

[3] Palicherla A, Zhang T, Porter D E. Teaching virtualization by building a hypervisor[C]//Proceedings of the 46th ACM Technical Symposium on Computer Science Education. 2015: 424-429.

[4] Bauman E, Ayoade G, Lin Z. A survey on hypervisor-based monitoring: approaches, applications, and evolutions[J]. ACM Computing Surveys (CSUR), 2015, 48(1): 1-33.

[5] Andrew Waterman, Krste Asanovi , John Hauser, SiFive Inc., CS Division, EECS Department, University of California, Berkeley, et al. The RISC-V instruction set manual[J]. Volume I: Privileged Architecutre, version, 2021,12.

[6] Sá B, Martins J, Pinto S. A first look at RISC-V virtualization from an embedded systems perspective[J]. IEEE Transactions on Computers, 2021, 71(9): 2177-2190.

[7] Sá B, Valente L, Martins J, et al. CVA6 RISC-V Virtualization: Architecture, Microarchitecture, and Design Space Exploration[J]. arXiv preprint arXiv:2302.02969, 2023.

[8] Martins J, Tavares A, Solieri M, et al. Bao: A lightweight static partitioning hypervisor for modern multi-core embedded systems[C]//Workshop on Next Generation Real-Time Embedded Systems (NG-RES 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[9] Adams K, Agesen O. A comparison of software and hardware techniques for x86 virtualization[J]. ACM Sigplan Notices, 2006, 41(11): 2-13.

[10] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[J]. ACM SIGOPS operating systems review, 2003, 37(5): 164-177.

[11] Kivity A, Kamay Y, Laor D, et al. kvm: the Linux virtual machine monitor[C]//Proceedings of the Linux symposium. 2007, 1(8): 225-230.

[13] Sugerman J, Venkitachalam G, Lim B H. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor[C]//USENIX Annual Technical Conference, General Track. 2001: 1-14.

[14] Ben-Yehuda M, Day M D, Dubitzky Z, et al. The Turtles Project: Design and Implementation of Nested Virtualization[C]//Osdi. 2010, 10: 423-436.

[15] Dautenhahn N, Kasampalis T, Dietz W, et al. Nested kernel: An operating system architecture for intra-kernel privilege separation[C]//Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems. 2015: 191-206.

选题是否合适： 是☑ 否☐

课题能否实现： 能☑ 不能☐

指导教师（签字） 李罡

2023 年 3 月 4 日

选题是否合适：　是☑　否☐

课题能否实现：　能☑　不能☐

审题小组组长（签字） 

2023 年 3 月 4 日