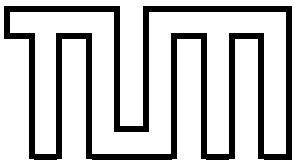


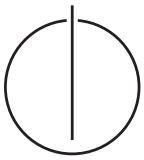
SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

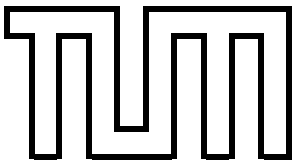
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Effects of Linux VFIO for User Space I/O

Adrian Simon Würth





SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

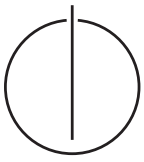
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Effects of Linux VFIO for User Space I/O

Effekt von Linux VFIO auf User Space E/A

Author:	Adrian Simon Würth
Supervisor:	Prof. Dr. Thomas Neumann
Advisor:	Simon Ellmann, M.Sc.
Submission Date:	August 15, 2024



I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Munich, July 9, 2024 Adrian Simon Würth

Abstract

The research into direct memory access in the userspace has increased over the latest years especially in the field of virtualization. The main reason for this is the need for high performance and low latency in virtualized environments. The hardware that enables this is called the IOMMU. Through the IOMMU, the guest operating system can directly access the hardware without the need for the hypervisor to intervene. This technology is not bound to only virtualization, but can also be used for high-performance drivers. In the past, the IOMMU was only accessible through the kernel, but with the introduction of the *Intel VT-d* and *AMD-Vi* extensions, the IOMMU can now be accessed from the userspace. The VFIO framework and the IOMMUFD user API build the foundation for this. We aim to achieve the same performance and low latency as directly mapping the hardware into the guest operating system, while increasing the security of the system.

Contents

1 Introduction

The IOMMU was first implemented as a way to increase address space for 32-bit devices. After almost every device was using 64-bit addresses, the IOMMUs purpose shifted to providing a layer of isolation between devices and the CPU. [BY+07] The isolation layer prevents devices from accessing memory that they are not allowed to access. An additional benefit of the IOMMU is that root privileges are not needed to access memory. We demonstrate the effects of using the IOMMU by implementing it on the NVMe driver vroom. [Pir24] We use Rust to implement the driver and the userspace application as it offers high performance and memory safety without garbage collection. The IOMMU used in this thesis is an Intel IOMMU using the VT-d extension.

2 Background

2.1 vroom

vroom currently uses hugepages and locks them using mlock to prevent the Kernel from swapping them out. This only enables the use of 2MiB hugepages and up, which can be disadvantageous for certain applications that would benefit from smaller page sizes. By using the IOMMU, this way of handling memory becomes redundant, as the IOMMU often supports smaller page sizes.

2.2 IOMMU

The IOMMU works similarly to the MMU in the CPU, mapping physical addresses to virtual addresses. The IOMMU does this by performing page table walks, through which a physcial address is translated to a virtual address. To improve performance on the IOMMU, an IOTLB is used to cache translations.

2.3 VFIO

VFIO is a framework that enables the use of the IOMMU from userspace.

2.4 IOMMUFD

IOMMUFD is a user API for exposing DMA to userspace. It allows management of I/O address spaces (IOAS), enabling mapping/unmapping user space memory on the IOMMU. It can be used in combination with VFIO to provide a more user-friendly interface.

3 Related Work

3.1 SPDK

3.2 IOMMUFD

4 Implementation

4.1 VFIO

4.1.1 Mapping DMA

4.1.2 Regions

4.1.3 Groups

4.2 IOMMUFD

IOMMUFD has only been recently added to the Linux Kernel. E.g. Debian 12 does not support it.

4.2.1 CDev

The device file descriptor, which was previously attained with `VFIO_GROUP_GET_DEVICE_FD` can now easily be opened under the path /dev/vfio/devices/vfioX.

4.2.2 IOAS

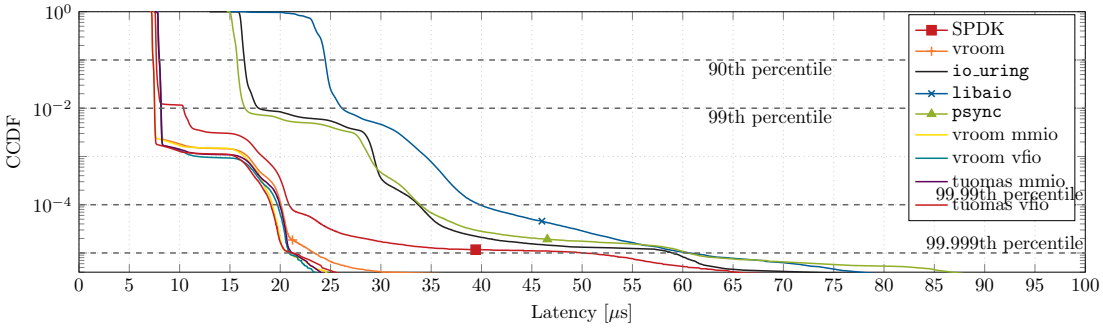
5 Evaluation

5.1 Setup

5.2 Latency

5.3 IOTLB

5.4 Mapping



(a) Random write

Figure 5.1: Tail latencies

6 Conclusion

List of Figures

List of Tables

Listings

Bibliography

[BY+07] M. Ben-Yehuda, J. Xenidis, M. Ostrowski, K. Rister, A. Bruemmer, and L. van Doorn. “The price of safety: Evaluating IOMMU performance.” In: *Ottawa Linux Symposium (OLS)* (Jan. 2007), p. 13.

[Pir24] T. Pirhonen. “Writing an NVMe Driver in Rust.” BA thesis. Technical University of Munich, 2024.