Design of SR-IOV Driver to Mitigate Performance Degradation for Lightweight Hypervisor

Keisuke Iida*
The University of Tokyo

Takaaki Fukai
National Institute of Advanced Industrial Science and Technology

Takahiro Hirofuchi
National Institute of Advanced Industrial Science and Technology

Takeshi Matsuya
Keio University

Lightweight hypervisor (LH) enables to achieve OS independent function by virtualization while minimize the code size of its hypervisor and performance degradation caused by virtualization. LH is designed to keep Trusted Computed Base (TCB) of whole system and virtualization overhead low as possible by eliminating the complex tasks of hypervisor such as vCPU scheduling, device virtualization, interrupt emulation.

In LH, there are some situations where the hypervisor wants to control the device. For example, to migrate guest, LH wants to use network card [1]. However, it is difficult because all devices assigned directly to the guest.

Single-Root I/O Virtualization (SR-IOV) is a good way to achieve this without device virtualization by software. SR-IOV is hardware virtualization feature which partition physical PCI function (PF) into independent lightweight PCI functions (VF). In SR-IOV architecture, VF driver in the guest OS is allowed to do data processing such as packet processing, but hardware configuration must be delegated to SR-IOV PF driver in the hypervisor from VF driver. To notify configuration delegation from VF to PF, SR-IOV use MSI-X interrupt.

Unfortunately, the dependency of the SR-IOV on the interrupt results in a performance overhead. CPU that supports hardware-assisted virtualization provide interrupt trap mechanisms that enable hypervisor to detect the interrupts. However, these mechanisms typically do not support selective trap, meaning that when the mechanism is enabled, all of the interrupts are trapped and the hypervisor is involved. Therefore, the PF-VF communication that relies on the interrupts in the hypervisor having to intercept all interrupts, which is known to cause significant performance degradation under I/O-intensive workloads. [2].

Posted Interrupt (PI) technology allows delivering specified interrupts to guests directly. However, this technology is a complex hardware mechanism that not only supports virtualization of CPU, but also supports virtualization of interrupt controllers and IOMMU, and hardware vendors' support for this technology varies. In addition, the implementation of the hypervisor for controlling posted interrupt mechanism tends to be complex due to the need to control these hardware components.

We propose interrupt-less SR-IOV driver, that detects processing requests from VF to the PF driver without using interrupts. Our proposed design allows the LH to pass through all interrupts to the guest, so be able to eliminate VM-Exits caused by external interrupts. It expects to mitigate performance degradation by SR-IOV driver. We noted that all of notifications to the PF by interrupts are not latency sensitive, therefore we use the monitor thread to process for new notifications asynchronously. This thread is executed when VM-Exit occurs for other reasons such as for handling sensitive instructions. Therefore, this avoids additional VM-exits for the PF driver. The advantage of this design is the simplicity of implementation and lower CPU consumption.

We implemented and evaluated interrupt-less SR-IOV Driver for 10GbE SR-IOV NIC (Intel X710) on BitVisor [3] based hypervisor. Compared to hypervisor with a classical SR-IOV driver which intercepts every interrupts, latency was improved by 8.5% and throughput was improved by 11.2% when the guest CPU is fully loaded.

Acknowledgments

This work was supported by JST, CREST Grant Number JPMJCR22M3, Japan.

References

[1] Takaaki Fukai et al. 2021. Live Migration in Bare-Metal Clouds. *IEEE Transactions on Cloud Computing* 9, 1 (2021), 226–239.

[2] Abel Gordon et al. 2012. ELI: bare-metal performance for I/O virtualization. In *Proceedings of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2012, London, UK, March 3-7, 2012*, Tim Harris and Michael L. Scott (Eds.). ACM, 411–422.

[3] Takahiro Shinagawa et al. 2009. BitVisor: A Thin Hypervisor for Enforcing i/o Device Security. In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE '09)*. Association for Computing Machinery, 121–130.

*This work was carried out while author was at National Institute of Advanced Industrial Science and Technology.