Rethinking IOMMU Address Translation

Muli Ben-Yehuda, Nadav Amit, Ben-Ami Yassour, Assaf Schuster, Dan Tsafrir

In a Nutshell

• IOMMUs translate and validate device accesses to main memory
• IOMMUs provide protection from misbehaving or malicious devices
• IOMMUs are used extensively on bare metal and in virtualized systems
• IOMMU translation overhead—caused by both software and hardware—can have a large effect on overall system performance and efficiency

Our main contribution is a combined software and hardware approach to IOMMU translation based on IOTLB injection and page table avoidance

Map/Unmap/DMA Read/DMA Write Traces

Map, unmap, and DMA reads and writes generated by an e1000 NIC when running netperf [Amit10]

IOMMU Translation Overhead

IOMMU translation overhead with different protection strategies [Amit11]. Strict protection requires 85% CPU vs. 30% for no IOMMU (×2.5)

Methodology

• Use IOMMU emulation for generating map/unmap/DMA traces
• Rethink translation based on workloads

Related Work

• Translation is expensive [Ben-Yehuda07]
• IOMMU protection strategies [Willman08]
• On-demand mapping [Yassour10]
• Exploring IOTLB design [Amit10]
• Skip, don’t walk (the page table) [Barr10]
• vIOMMU: IOMMU emulation [Amit11]

Results (Work in Progress)

IOTLB miss rates for different IOTLB configurations (lower miss rate is better). left shows e1000 with netperf, right shows SCSI controller with bonnie++. MPRE is mapping prefetch, which behaves similarly to IOTLB injection. Preliminary results and full experimental setup are available in [Amit10]