Contribution ID: **42** Type: **not specified**

# Interrupt Message Store: A scalable interrupt mechanism for the cloud

*Tuesday 10 September 2019 15:45 (45 minutes)*

With virtualization being the key to the success of cloud computing, Intel's
introduction of the Scalable IO Virtualization (SIOV) aims to further the cause by shifting the creation of assignable virtual devices from hardware to a more software assisted approach. Using SIOV, a device resource can be mapped directly to guest or other user space drivers for near native DMA (Direct Memory Access) performance. This flexible composition of direct assignable devices a.k.a. Assignable Device Interfaces (ADIs) is device specific and light weight, thus making them highly scalable. SIOV enables simpler device designs by unchaining hardware from costly PCI standard and can help address limitations associated with direct device assignment.

Until now, message signaled interrupts (MSI and MSI-X) were the de facto standard for device interrupt mechanism and could support up to 2048 interrupts per device. But now with SIOV, there is a need to support a large number of ADIs (>2048), through a matching scalable interrupt management mechanism to service these ADIs.

Interrupt message storage (IMS) is conceived as a scalable albeit device specific interrupt mechanism to meet such a demand. It allows non-PCI standard storage and enumeration of MSI address/data pair to reduce hardware overhead and achieve scalability. The size, location, and storage format for IMS is device-specific; some devices may implement IMS as on-device storage, while other devices may implement IMS in host memory.

Also, one of the limitations with the current Linux MSI-X code is that PCIe device MSI-x enablement and allocation is static. i.e. device driver gets only one chance to enable MSI-X vectors, usually during probe. With IMS, we aim to make the vector negotiation with the OS dynamic, deferring vector allocation to post probe phase, where actual demand information is available.

Through this presentation, the audience can view the internals of the complex
and ever evolving Linux interrupt subsystem and understand how IMS can fit into the maze of interrupt domains, chips, remapping etc. Also, an initial IMS Linux implementation will be presented with highlights on some of the unique implementation challenges. We will conclude by demonstrating a test case using the SIOV enabled device as an example for a complete view of IMS in a scalable virtualization environment.

## I agree to abide by the anti-harassment policy

Yes

## I confirm that I am already registered for LPC 2019

**Primary author:** DEY, Megha

**Presenter:** DEY, Megha

**Session Classification:** Kernel Summit Track

**Track Classification:** Kernel Summit talk