**What is the Problem?**

- Using traditional OS’s in the cloud—see RaaS poster nearby—is expensive.

**Today’s Operating Systems**

- Today’s operating systems are inefficient ⇒ need better sys. software.

**Traditional OS Structure**

- Traditional operating systems were designed to share I/O devices.

**Machine Virtualization**

- SR-IOV devices can be shared by multiple contexts.

**Benefits of nom**

- All applications bypass the kernel completely on the I/O path.
- Small, simple, and secure kernel.
- Applications customize their I/O stacks to fit their needs.
- Applications adapt to changing costs of different resources quickly.

**The nom Operating System**

- The nom kernel provides every application with direct access to its own devices using architectural support for machine virtualization.

**A Packet’s Progress**

**Traditional:**

- App1
- Network Stack
- Device Driver

**Exokernel:**

- App1
- Network Stack
- Device Driver

**nom:**

- App1
- Network Stack
- Device Driver

**nom is Work in Progress**

- Runs on x86-64 bare-metal and QEMU
- SMP support
- Intel, Mellanox SR-IOV devices
- PIO using iopl/VMCS exception bitmap
- MMIO using page-table mapping
- DMA using IOMMUs
- Direct interrupt injection [Gordon12]

**Related Work**

- Exokernel: [Engler95], [Kaashoek97], [Ganger02]
- Virtual machine device assignment: [LeVasseur04], [Ben-Yehuda06], [Gordon12]
- Userspace I/O, in particular VIA, Quadrics, and Infiniband.

**Current Research Projects**

- How should applications adapt to changing resource availability?
- What is the difference between an OS and a hypervisor?
- What is the difference between an application and a virtual machine?
- Are SR-IOV devices secure?