NoHype: Virtualized Cloud Infrastructure without the Virtualization

Mehdi Nasef

Universidad de Cantabria

November 2023

Introduction

Introduction

- Virtualization only in software (With no help from hardware)

Introduction

- Virtualization only in software (With no help from hardware)
- Virtualization in software with help from hardware

Introduction

- Virtualization only in software (With no help from hardware)
- Virtualization in software with help from hardware
- NoHype: Virtualization only in hardware

Motivation: Security concerns

Motivation: Security concerns

- A successful attack on the hypervisor gives access to all VMs

Motivation: Security concerns

- A successful attack on the hypervisor gives access to all VMs
- Large attack surface: Too many VM-Hypervisor interactions

Motivation: Security concerns

- A successful attack on the hypervisor gives access to all VMs
- Large attack surface: Too many VM-Hypervisor interactions
- Securing the Hypervisor is becoming more and more difficult

Threat model

Threat model

- Cloud provider is not malicious

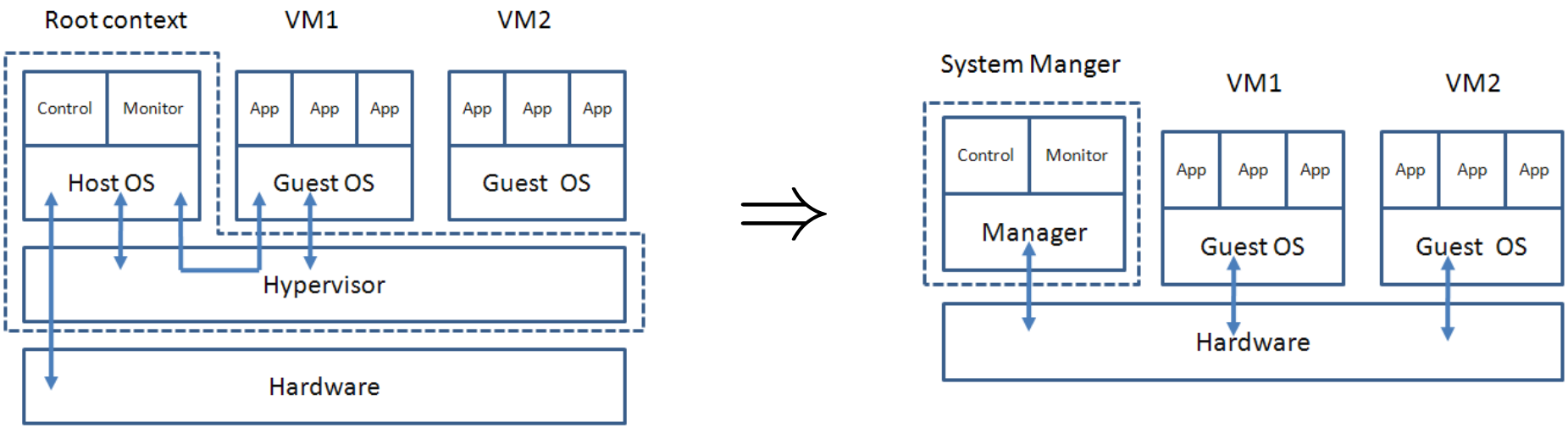
Threat model

- Cloud provider is not malicious
- Guest operating systems are not trusted

Threat model

- Cloud provider is not malicious
- Guest operating systems are not trusted
- Cloud management software is assumed to be secure
  - However, it is out of the scope of the paper.

NoHype: Doing hypervisor functionality in hardware



NoHype: Doing hypervisor functionality in hardware

Controlling access to CPU

- Currently, VMs are scheduled like processes in a regular OS

NoHype: Doing hypervisor functionality in hardware

Controlling access to CPU

- Currently, VMs are scheduled like processes in a regular OS
- NoHype dedicates a core to only one VM

NoHype: Doing hypervisor functionality in hardware

Controlling access to CPU

- Currently, VMs are scheduled like processes in a regular OS
- NoHype dedicates a core to only one VM
  - It runs uninterrupted on it

NoHype: Doing hypervisor functionality in hardware

Controlling access to CPU

- Currently, VMs are scheduled like processes in a regular OS
- NoHype dedicates a core to only one VM
  - It runs uninterrupted on it
  - Multi-tenancy is still possible in many-core CPUs

NoHype: Doing hypervisor functionality in hardware

Controlling access to CPU

- Currently, VMs are scheduled like processes in a regular OS
- NoHype dedicates a core to only one VM
  - It runs uninterrupted on it
  - Multi-tenancy is still possible in many-core CPUs
  - VMs are more isolated this way

NoHype: Doing hypervisor functionality in hardware

Memory management

Hardware enforced memory partitioning:

NoHype: Doing hypervisor functionality in hardware

Memory management

Hardware enforced memory partitioning:

- Each VM has a dedicated share of physical memory

NoHype: Doing hypervisor functionality in hardware

Memory management

Hardware enforced memory partitioning:

- Each VM has a dedicated share of physical memory
- It gets mapped to its guest physical space when started

NoHype: Doing hypervisor functionality in hardware

Memory management

Hardware enforced memory partitioning:

- Each VM has a dedicated share of physical memory
- It gets mapped to its guest physical space when started
- The guest OS manages it as it would do without virtualization

NoHype: Doing hypervisor functionality in hardware

Memory management

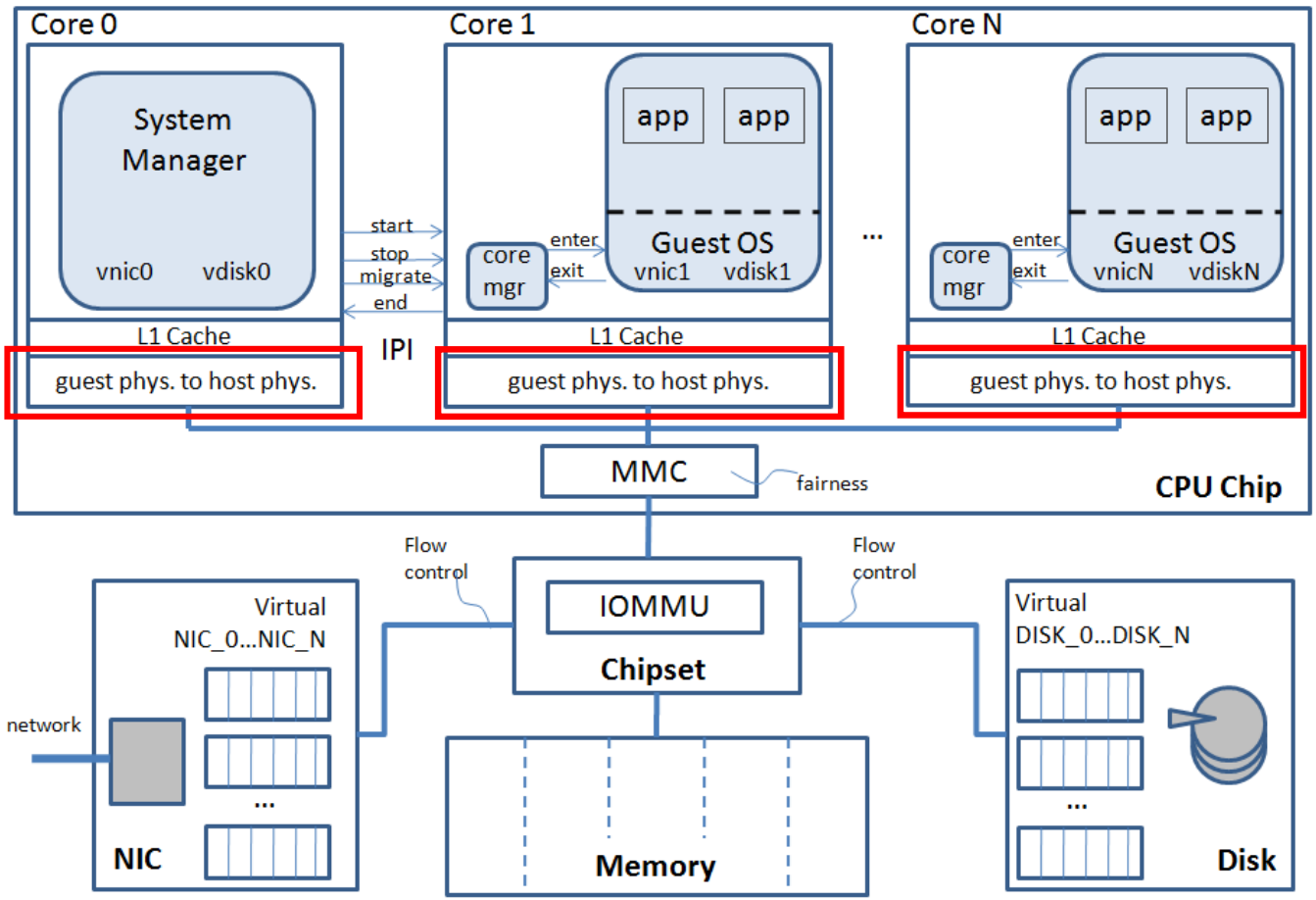
Hardware enforced memory partitioning:

- Each VM has a dedicated share of physical memory
- It gets mapped to its guest physical space when started
- The guest OS manages it as it would do without virtualization
- Hardware support is already present on current machines: Extended Page Tables

NoHype: Doing hypervisor functionality in hardware

Memory management

Hardware enforced memory partitioning:



NoHype: Doing hypervisor functionality in hardware

Memory management

Memory bandwidth fairness:

NoHype: Doing hypervisor functionality in hardware

Memory management

Memory bandwidth fairness:

- Prevent VMs from hogging memory bandwidth

NoHype: Doing hypervisor functionality in hardware

Memory management

Memory bandwidth fairness:

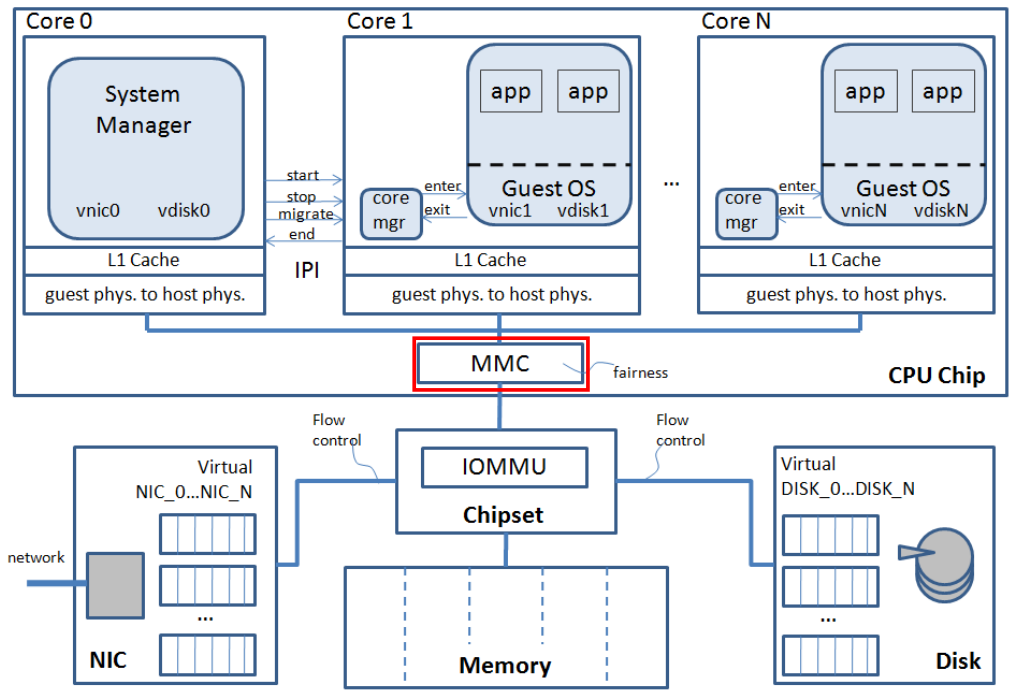
- Prevent VMs from hogging memory bandwidth
- Use a fair memory request scheduling algorithm

NoHype: Doing hypervisor functionality in hardware

Memory management

Memory bandwidth fairness:

- Prevent VMs from hogging memory bandwidth
- Use a fair memory request scheduling algorithm
- It is not supported by current hardware



NoHype: Doing hypervisor functionality in hardware

IO devices

NoHype: Doing hypervisor functionality in hardware

IO devices

- Dedicated actual or hardware virtual IO devices per VM (direct access)

NoHype: Doing hypervisor functionality in hardware

IO devices

- Dedicated actual or hardware virtual IO devices per VM (direct access)
- Device interfaces are mapped to the VM's guest physical space

NoHype: Doing hypervisor functionality in hardware

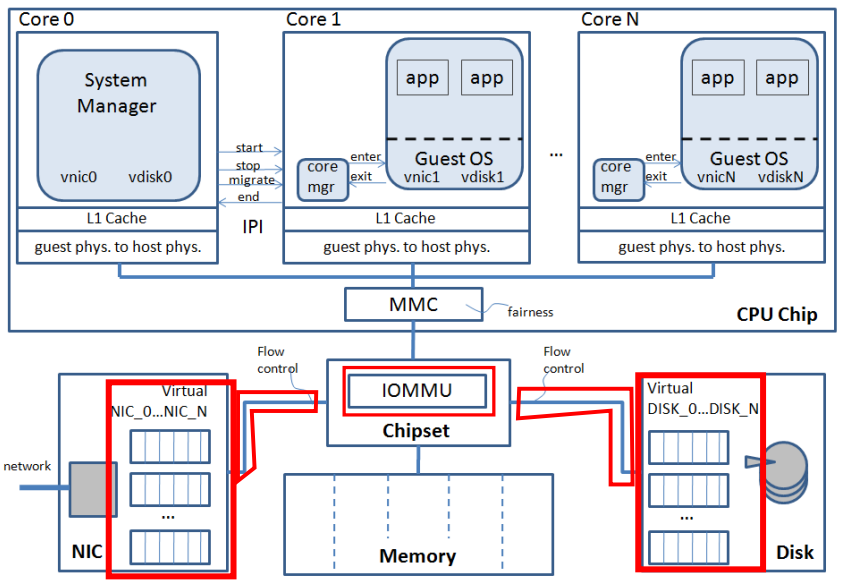
IO devices

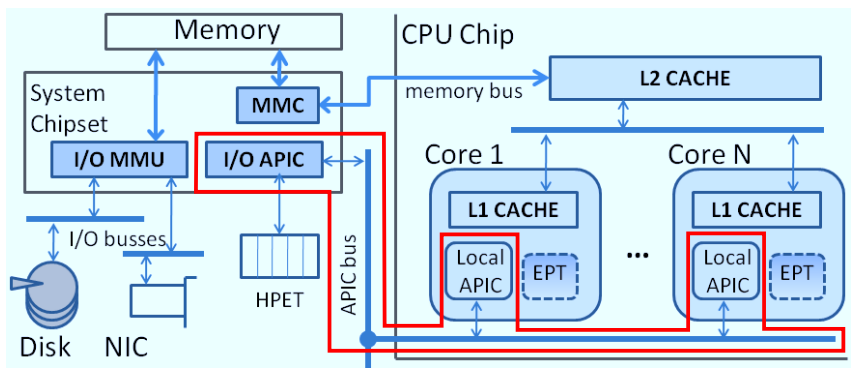
- Dedicated actual or hardware virtual IO devices per VM (direct access)
- Device interfaces are mapped to the VM's guest physical space
- controll the rate of IO operations: PCIe flow control

NoHype: Doing hypervisor functionality in hardware

IO devices

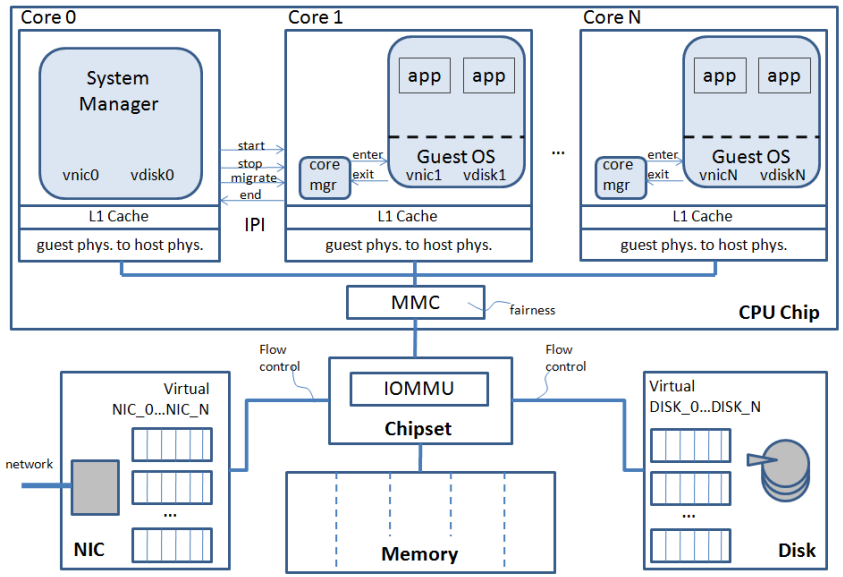
- Dedicated actual or hardware virtual IO devices per VM (direct access)
- Device interfaces are mapped to the VM's guest physical space
- controll the rate of IO operations: PCIe flow control
- hardware support: IOMMU, APIC, MSI, SR-IOV





Start, stop and abort a VM

Some software is still needed for these functions



Start a VM

1. Cloud manager notifies system manager to start a VM
2. System manager maps the memory and disk of the to-be-assigned VM into the system manager's space
3. System manager downloads the disk image and stores it on a local disk and zeroes out the memory
4. After initialization, system manager un-maps the memory and disk from its space
5. System manager issues a 'start' IPI to the core
6. Core manager initializes memory and I/O mapping
7. Core manager exits to guest OS, starting the guest OS execution

Stop a VM

1. Cloud manager notifies system manager to stop a VM
2. System manager issues a 'stop' IPI to the core running the target VM
3. Core manager optionally saves the disk image of the VM, then clears the disk and memory
4. Core manager un-maps memory and I/O
5. Core manager puts the current core into idle state

Abort a VM

1. Enter core manager on an illegal operation
2. Core manager sends an 'end' IPI to the system manager
3. Core manager optionally clears disk and memory
4. Core manager un-maps memory and I/O
5. Core manager puts the current core into idle state
6. System manager notifies cloud manager

Conclusions

Thank you for your attention!