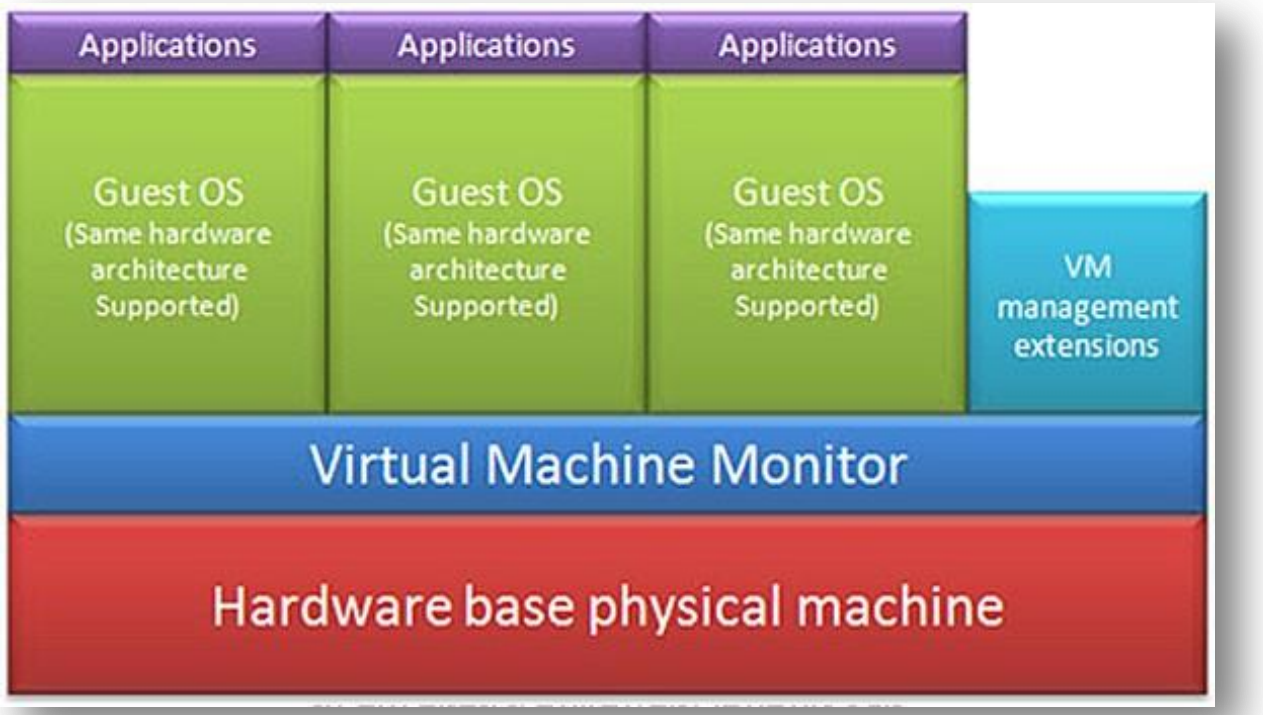
Xen I/O Overview

Xen I/O Overview

- Xen is a popular open-source x86 virtual machine monitor
  - full-virtualization
  - para-virtualization
- para-virtualization as a more efficient and lower overhead mode of virtualizations

Virtualization Approaches

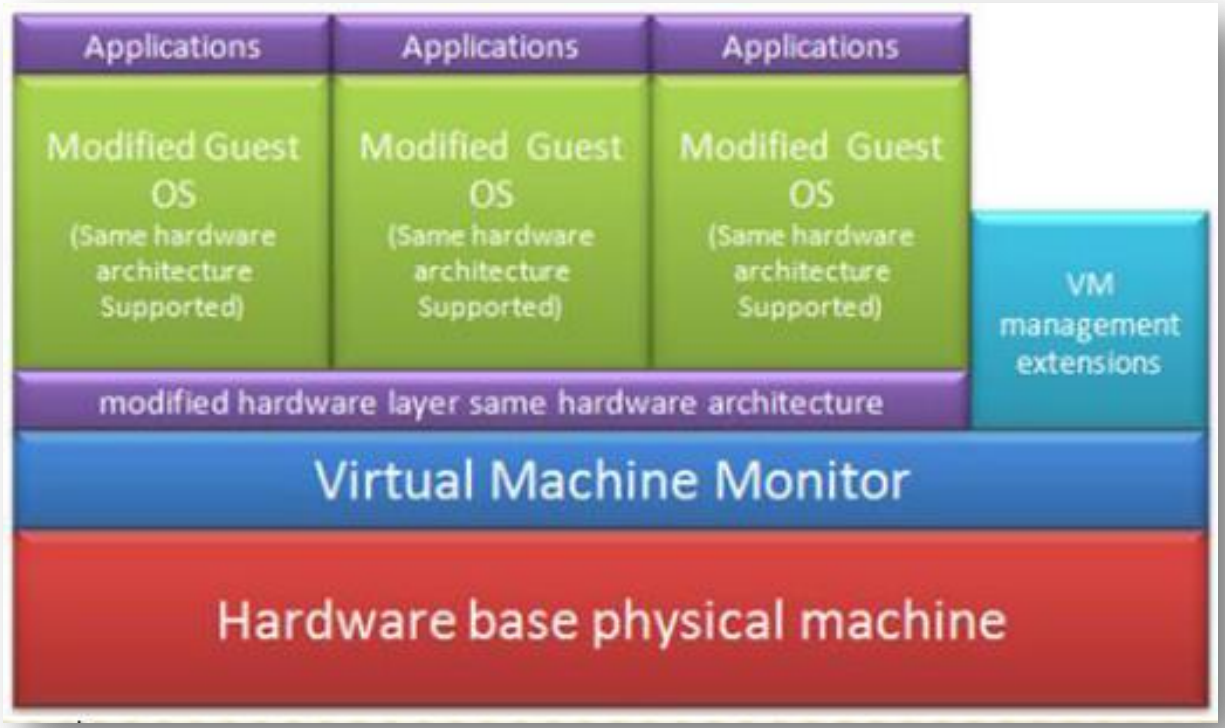
- Full-Virtualization



Pros	**Need not to modify guest OS**
Cons	**Significant performance hit**

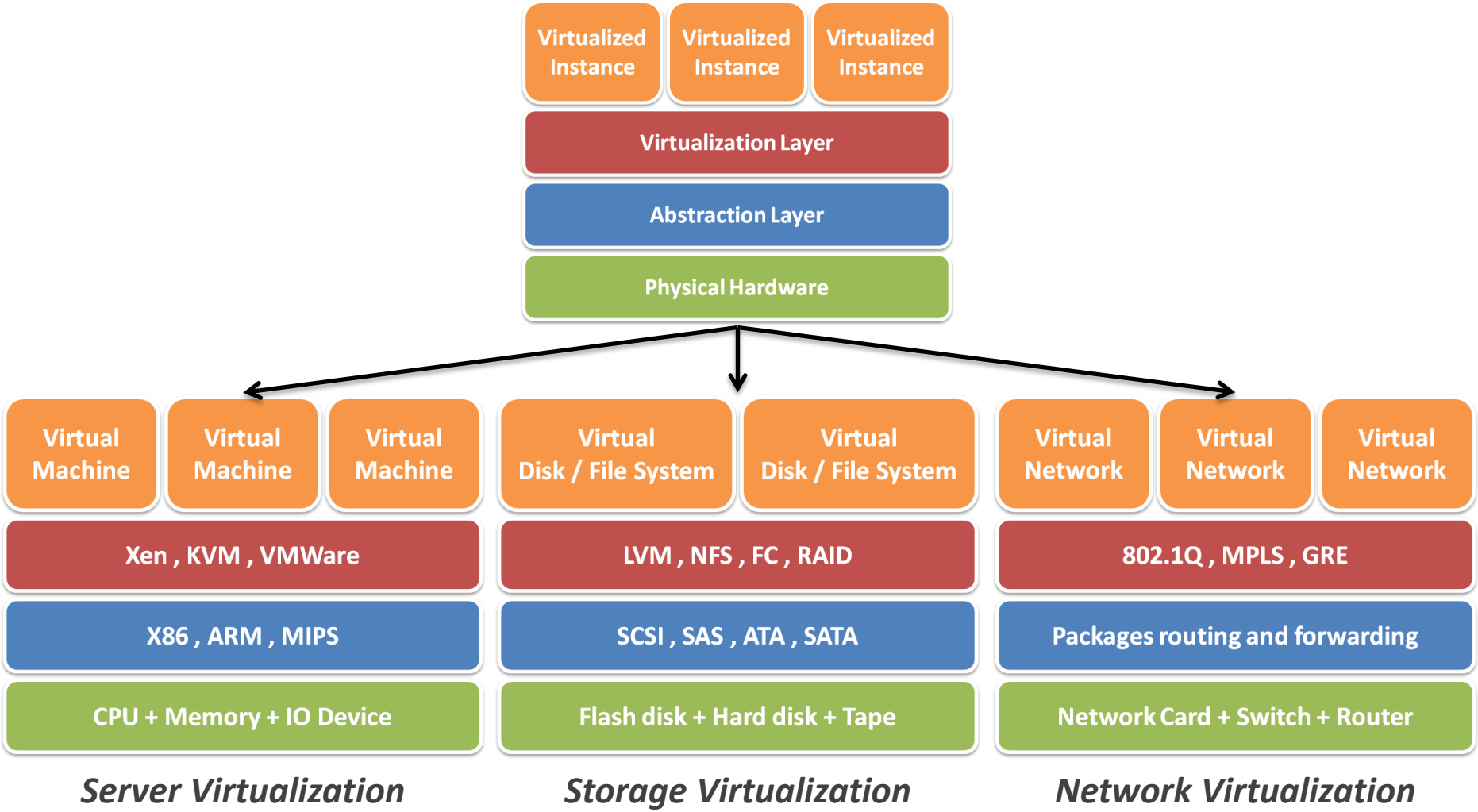
Virtualization Approaches

- Para-Virtualization



Pros	**Light weight and high performance**
Cons	**Require modification of guest OS**

- Virtualization techniques



Xen I/O Overview

- In para-virtualization I/O mode
  - Xen VMM layer uses asynchronous hypercall mechanism to deliver virtual interrupts
  - Notifications among domains via event channel

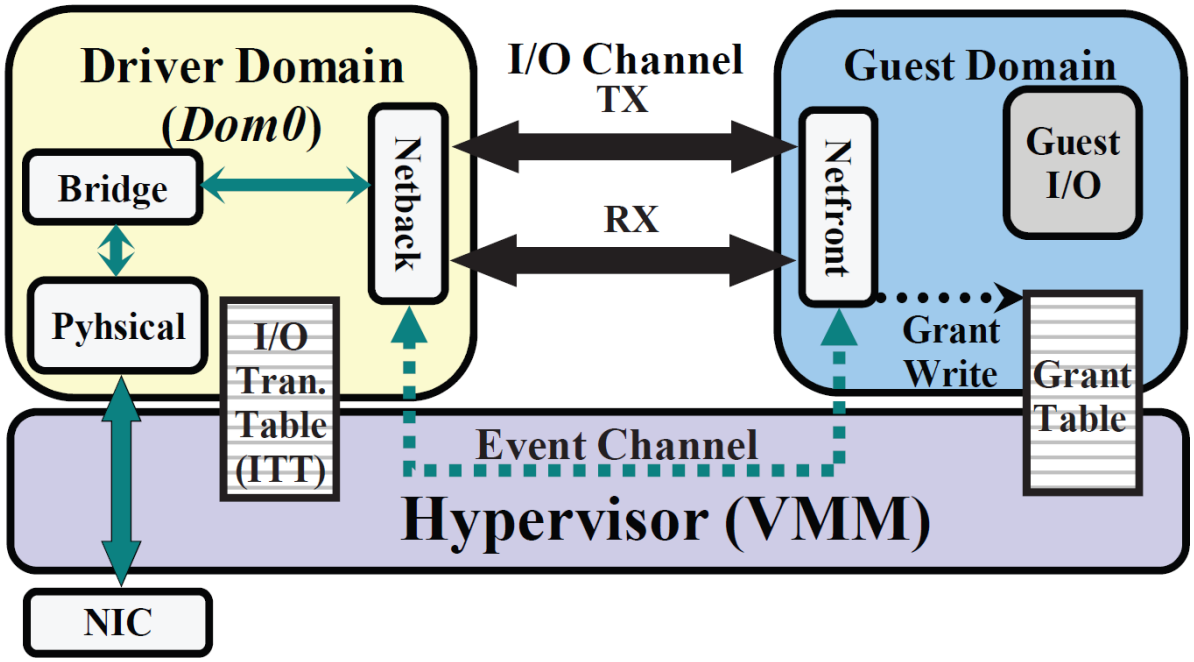
What's Virtual Machine Monitor (VMM) ?

VMM or **Hypervisor** is the software layer providing the virtualization

Xen I/O Overview

- A privileged domain called Dom0
  - Driver domain hosting unmodified Linux drivers
  - Access to hardware devices
  - Performs I/O operations on behalf of unprivileged guest domains

Xen I/O Architecture.



- The logical components of the latest Xen I/O model

Xen I/O Overview

- Virtual network interface in guest domain is called netfront
- In Dom0, netback is a counterpart for netfront
- Netfront and netback use a bidirection ring of asynchronous requests to exchange data
- The bridge in Dom0 handles the packets from NIC and performs the software based routine

Xen I/O Overview

- Receive packet by the NIC (RX)
  - Raises an interrupt to the upper layer
  - Hypervisor (VMM) handles the interrupt first
- Hypervisor determine whether Dom0 has the access to the real hardware

Xen I/O Overview

- Receiving the interrupt, Dom0 starts to process the network packet
  - Removes the packet from NIC
  - Sends the packet to the bridge
  - Bridge de-multiplexes the packet and delivers it to the appropriate netback interface

Xen I/O Overview

- Netback raises a hypercall to hypervisor, requesting an unused memory page
- Hypervisor notifies to grant a page to keep the memory allocation balanced
- Netback copies the received data to granted page in guest domain
- Guest domain receives the packet as if it comes directly from NIC

Xen I/O Overview

- Applied to send a packet on the send path (TX), explicit memory page
- Ownership of physical page is transferred instead of the real page

Xen I/O Overview

- Protect the I/O buffer in guest domain's memory
- Share the I/O buffer with Dom0 properly
- Hypervisor creates a unique grant table for each guest domain
  - Only can be accessed by hypervisor and guest domain

Xen I/O Overview

- Guest domain initializes an associated memory I/O buffer shared with Dom0
- Dom0 requests to exchange the I/O data with guest domain
- guest domain invokes a grant, then a entry in grant table with three key information
  - (1) Valid memory page address
  - (2) Dom0's id
  - (3) Operation permission(read-only for transmission or read-write for reception)

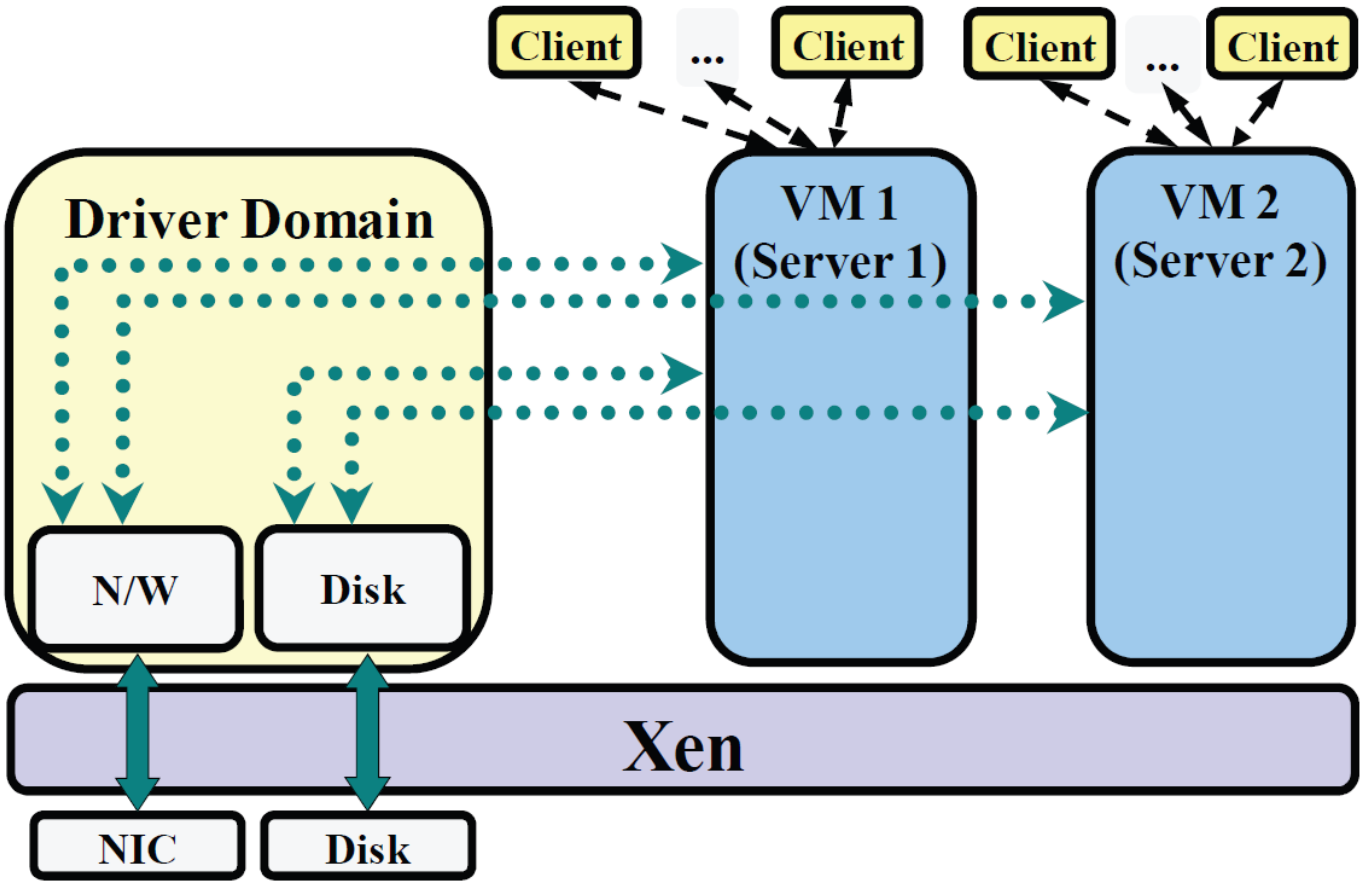
Xen I/O Overview

- guest domain issues a hypercall through hypervisor to Dom0
  - Pass a indexes the correct entry in grant table
  - Dom0 can find the grant entry
  - Hypervisor validate Dom0 to perform the I/O operation in I/O buffer

Xen I/O Overview

- When I/O exchange is accomplished or guest domain wants to repurpose the granted page
  - Another hypercall is issued to hypervisor
- Hypervisor synchronizes access permission between Dom0 and guest domain
- If Dom0 gives up the granted page
  - guest domain revokes the grant by another simple writing operation

Testbed Architechure



- Logical components of virtualized cloud environments

Bin packing problem

- Bin packing problem is a combinatorial NP-hard problem
- Definition:
  - n objects of different
  - Objects size is $a_1, a_2, .., a_n$
  - m bins
  - Each of volume *V*
- Objects packed into bins, and minimizes the number of bins used

Bin packing problem

- Best-fit algorithm
  - Initially all bins are empty
  - start with bins k = 0 and item i = 1
  - Consider all bins j = 1,…, k, place item i in the bin that has must accordance residual capacity

Bin packing problem

- First Fit Algorithm
  - Initially all bins are empty
  - start with bins k = 0 and item i = 1
  - Consider all bins j = 1,…, k, place item i in the first bin that has sufficient residual capacity
  - If there is no such bin increment k and repeat until item n is assigned

Bin packing problem

- Next Fit Algorithm
  - Initially all bins are empty
  - start with bin j = 1 and item i = 1
  - If bin j has residual capacity for item i, assign item i to bin j, and consider item i + 1
  - Otherwise consider bin j + 1 and item i.