the features for VMs:

not influence each other

support different OSs

little performance penalty for virtualization

\tip: difference between containers and vms: vms can support guest OS on it. containers do not run independent OS

the drawbacks of different users in a single machine:

no performance isolation

high demand for administration configuration among different applications

the thought in XEN -> provide multiplexing in lower-level hardware to mitigate the interactions. With overhead for supporting different OSs.

the drawbacks for full virtualization:

* sometimes hard to implement for the characteristics of the framework
* should offer the real machines to guests so that they can implement some related features

the decision for XEN: \*\*paravirtualization\*\*

\tip: the drawback of paravirtualization: each time for accommodation for a new guest OS, Xen has to be modified.

## the virtual machine interface

### memory

* guest OSs are responsible for allocating and managing the hardware paget ables
* Xen exists at the top of every address space

virtual memory virtualization

guest OS -> allocate a page -> register it with Xen -> further authentication request should be given to the hypervisor.

segmentation virtualization

similar to above with request validating to the hardware segment descriptor tables.

### CPU

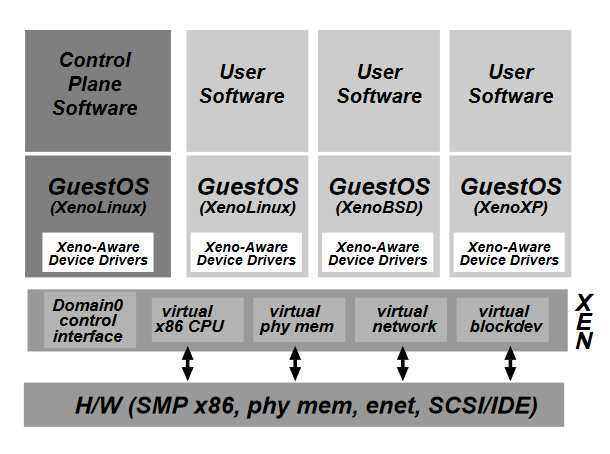
privilege -> the guest OS should be less privileged than th hypervisor but more privilege than the applications.

exception of system call: a fast exception handler which accesses directly by the processor

page fault exception should be delivered to Xen.

### I/O

Xen exposes a set of clean and simple device abstractions



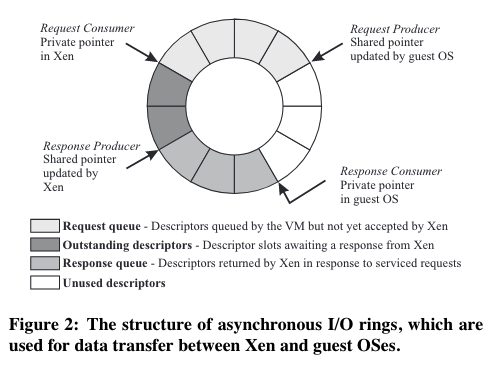
control interface: create/terminate other domains, control their associated scheduling parameters, physical memory allocations and the access they are given to the machine’s physical disks and network services.

## detailed design

control between Xen and other domains: hypercalls and events(asynchronized)

\tip: when a privileged instruction is sent from the user-mode, the hyperviser will trap it and handle it as an exception, and distinguish whether it’s sent from an app or the guest OS.

I/O



scheduling: borrowed virtual time scheduling algorithm

time: real time, virtual time and wall-clock time

\tip: Xen’s solution to page fault in x86 -> every time guest OS wants to change thepage table, send a hypercall to the hypervisor.(has to change the guest OS’s code)

VMWare’s solution: multi-layer page tables with virtual, logical, physical address