AHCI

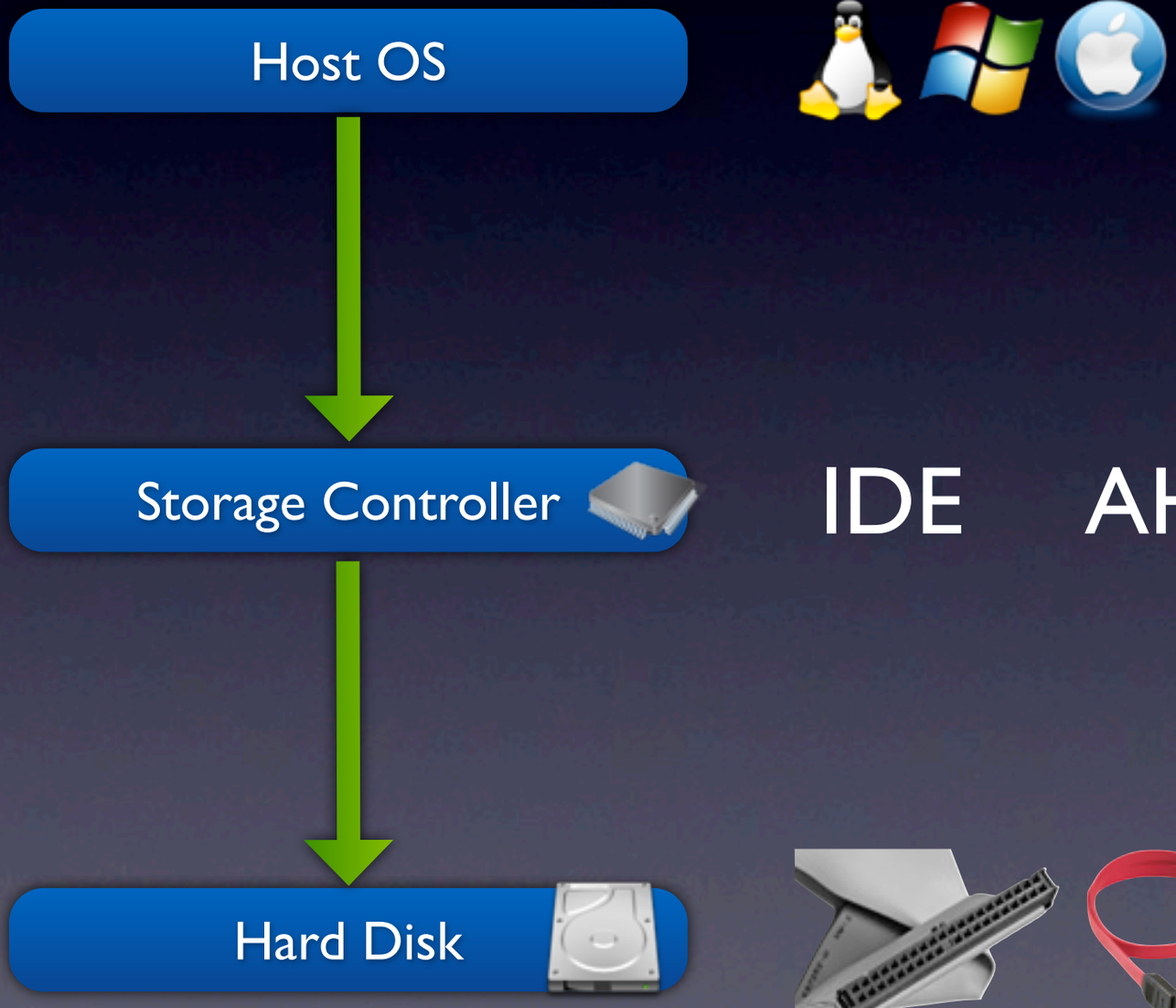
Doing storage right

About Me

- Alexander Graf
- KVM and Qemu developer
  - Server class PowerPC KVM port
  - S390x Qemu guest support
  - x86 Mac OS X in KVM
  - Nested SVM
  - Xenner
  - ...

Storage

Naming

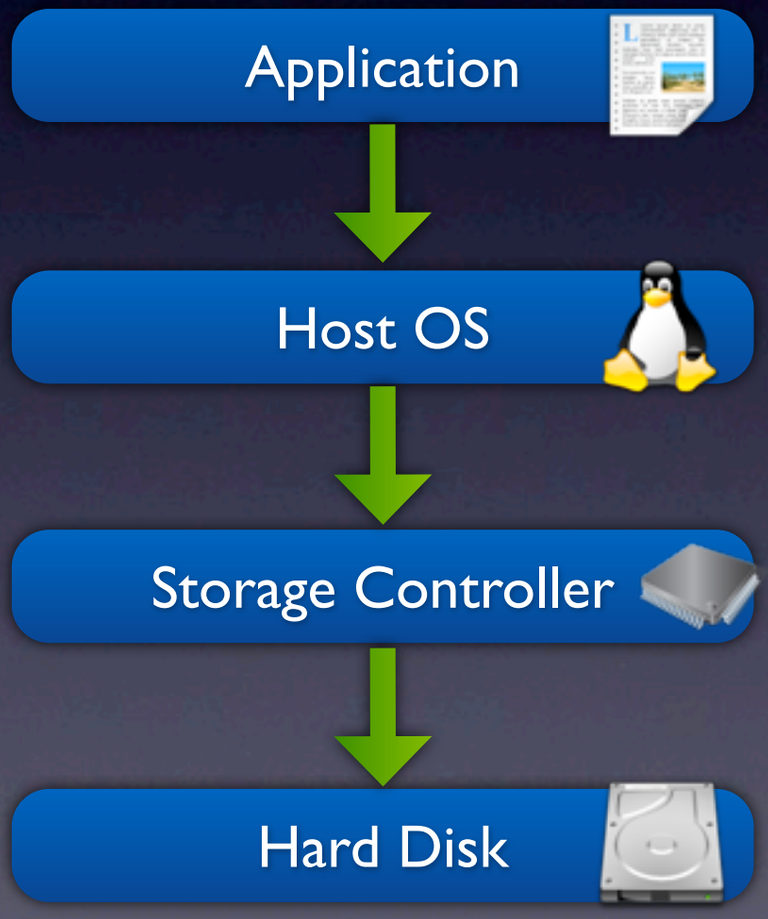


IDE AHCI virtio-blk

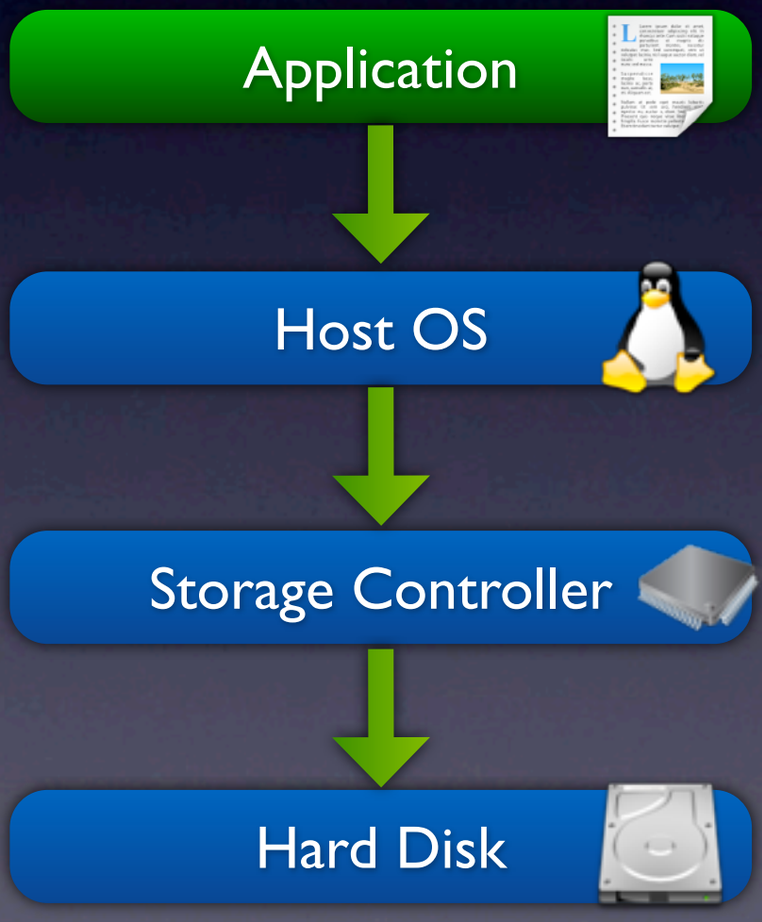
PATA

SATA

Storage

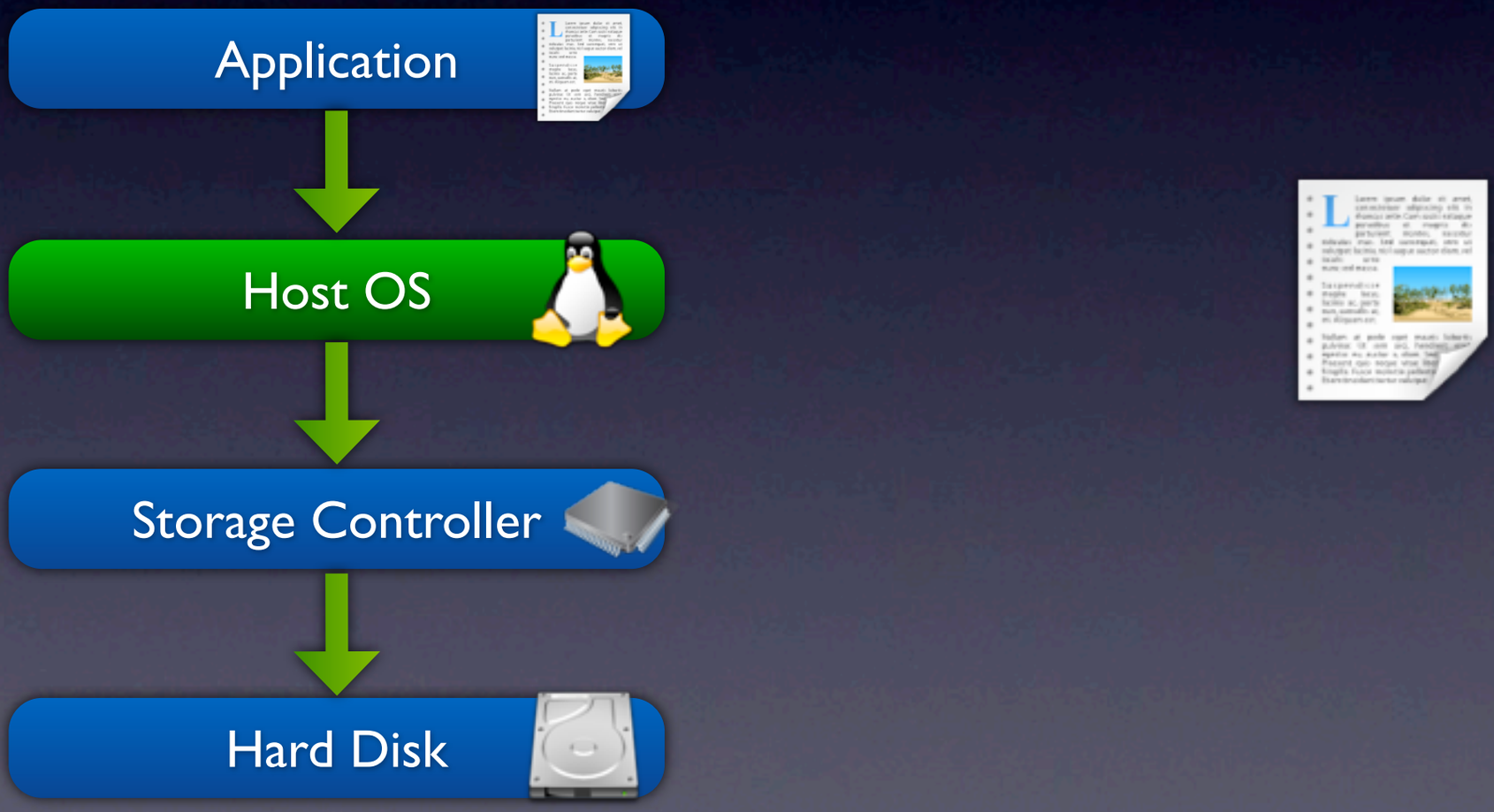


Storage



write( )

Storage

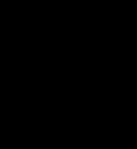


Storage

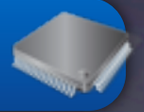
RAM

Application

Host OS

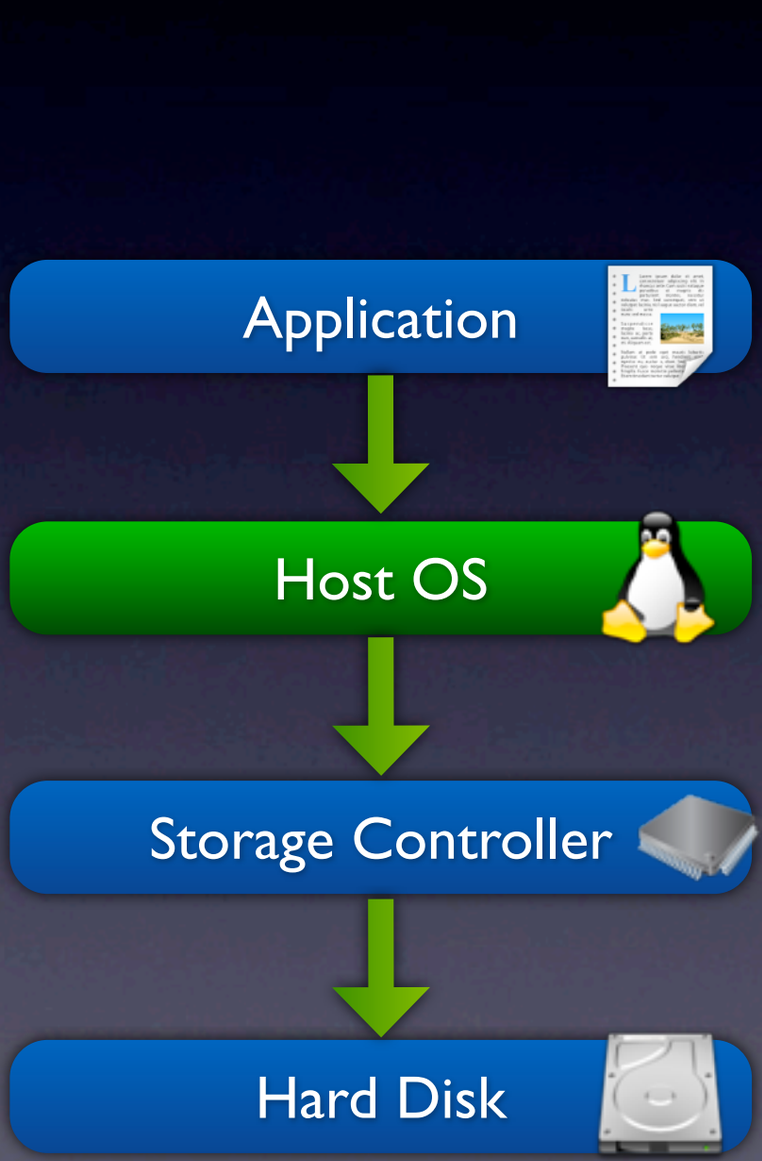


Storage Controller

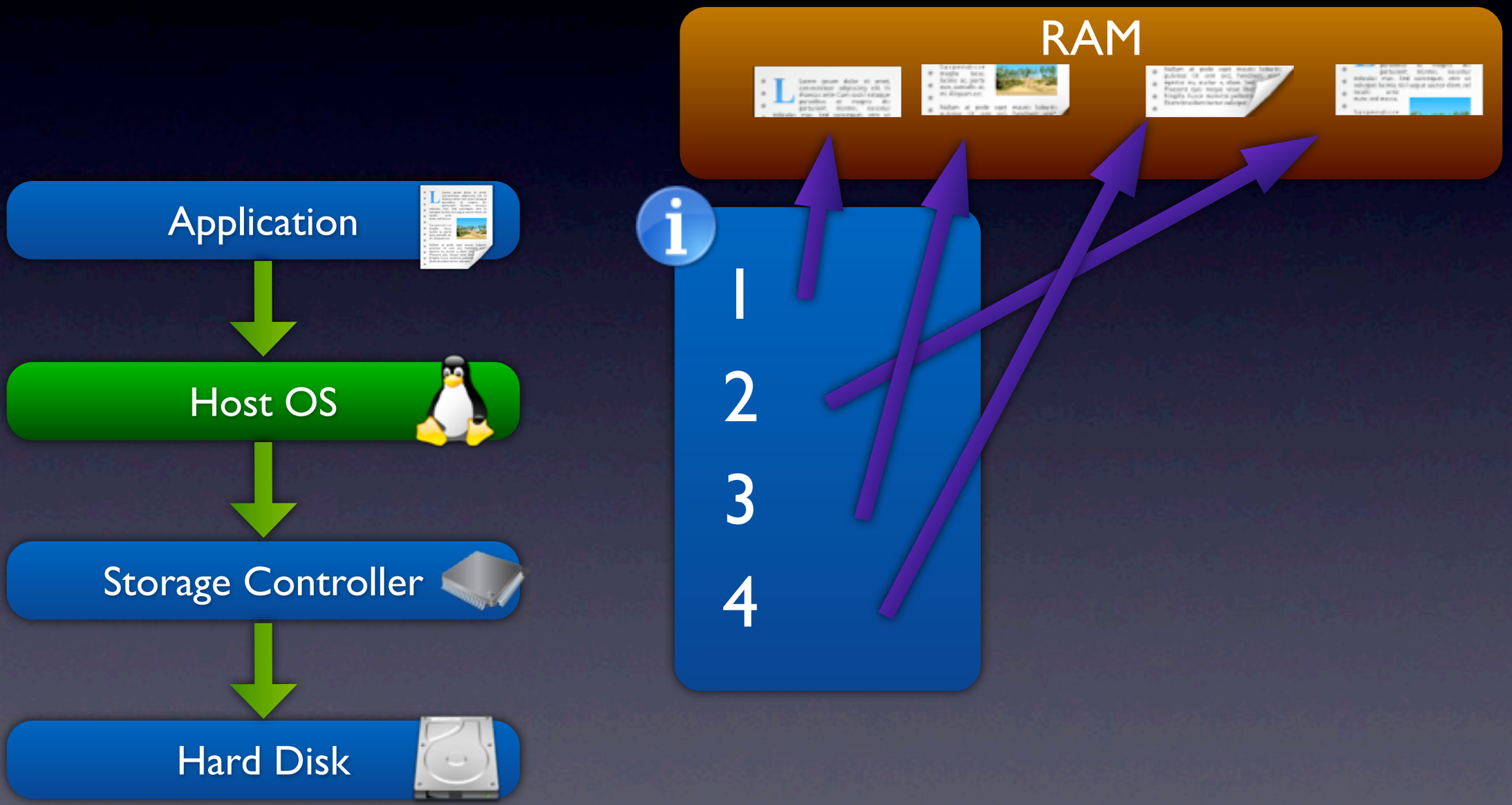


Hard Disk

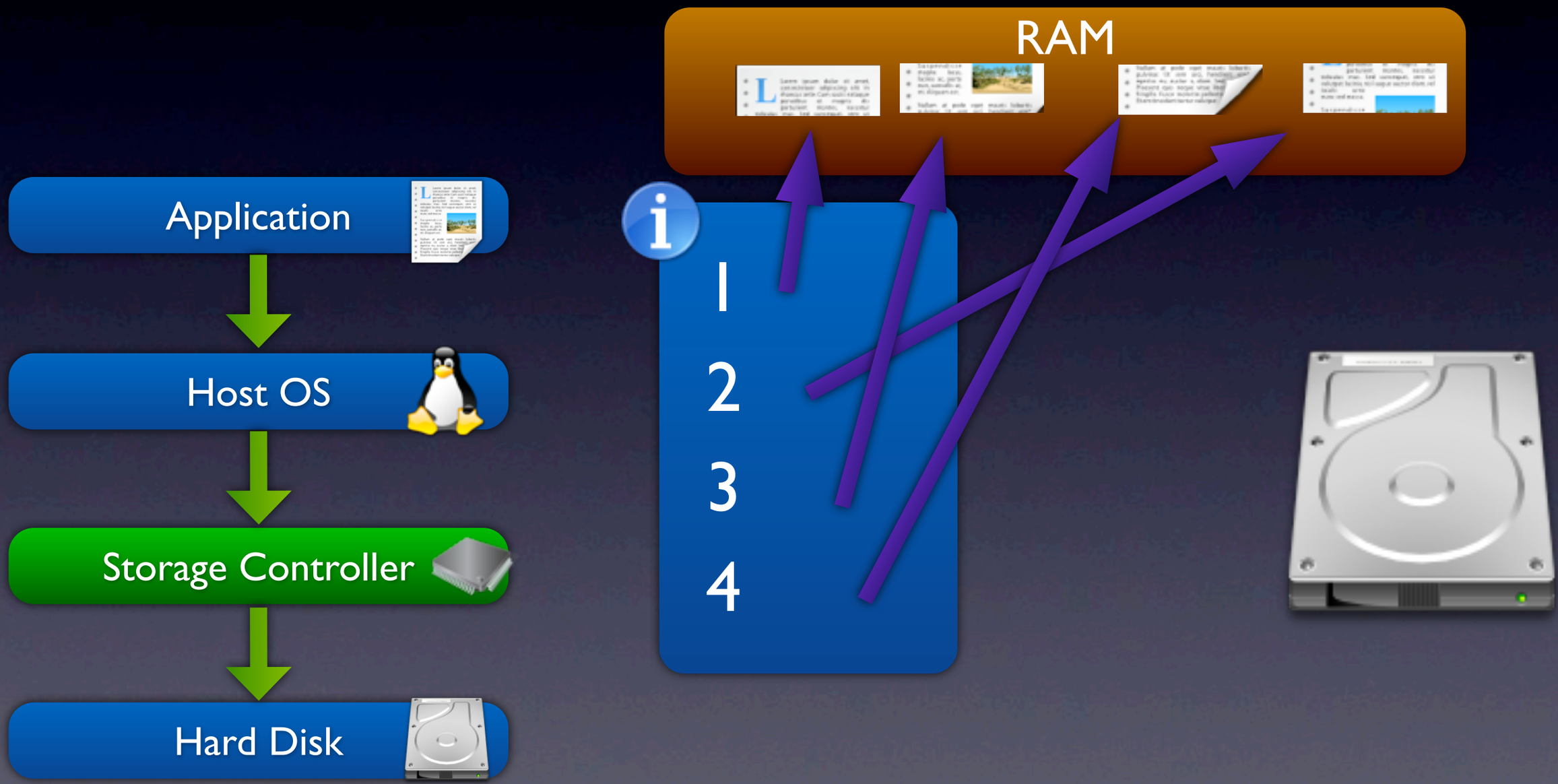
Storage



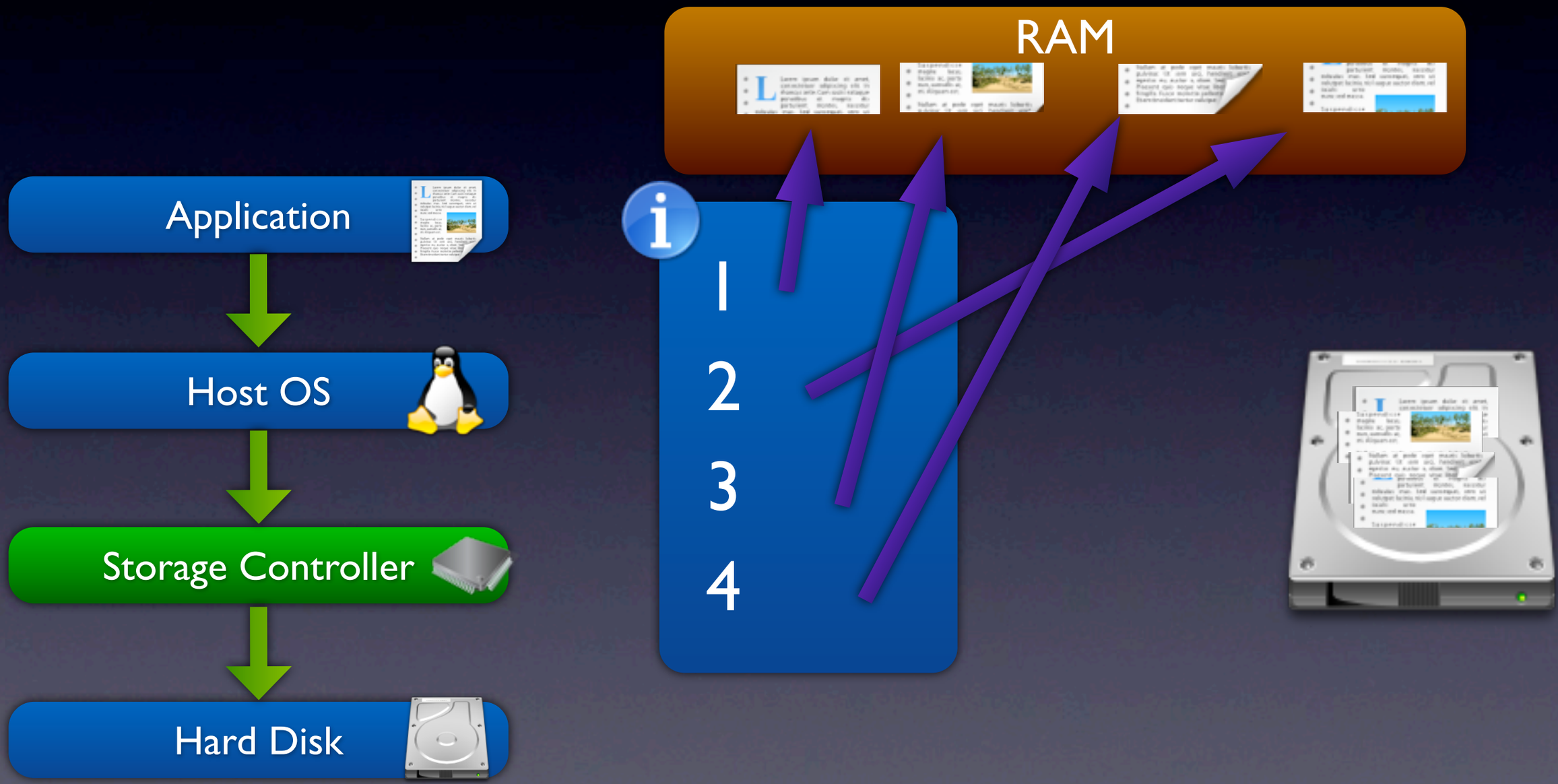
Storage



Storage



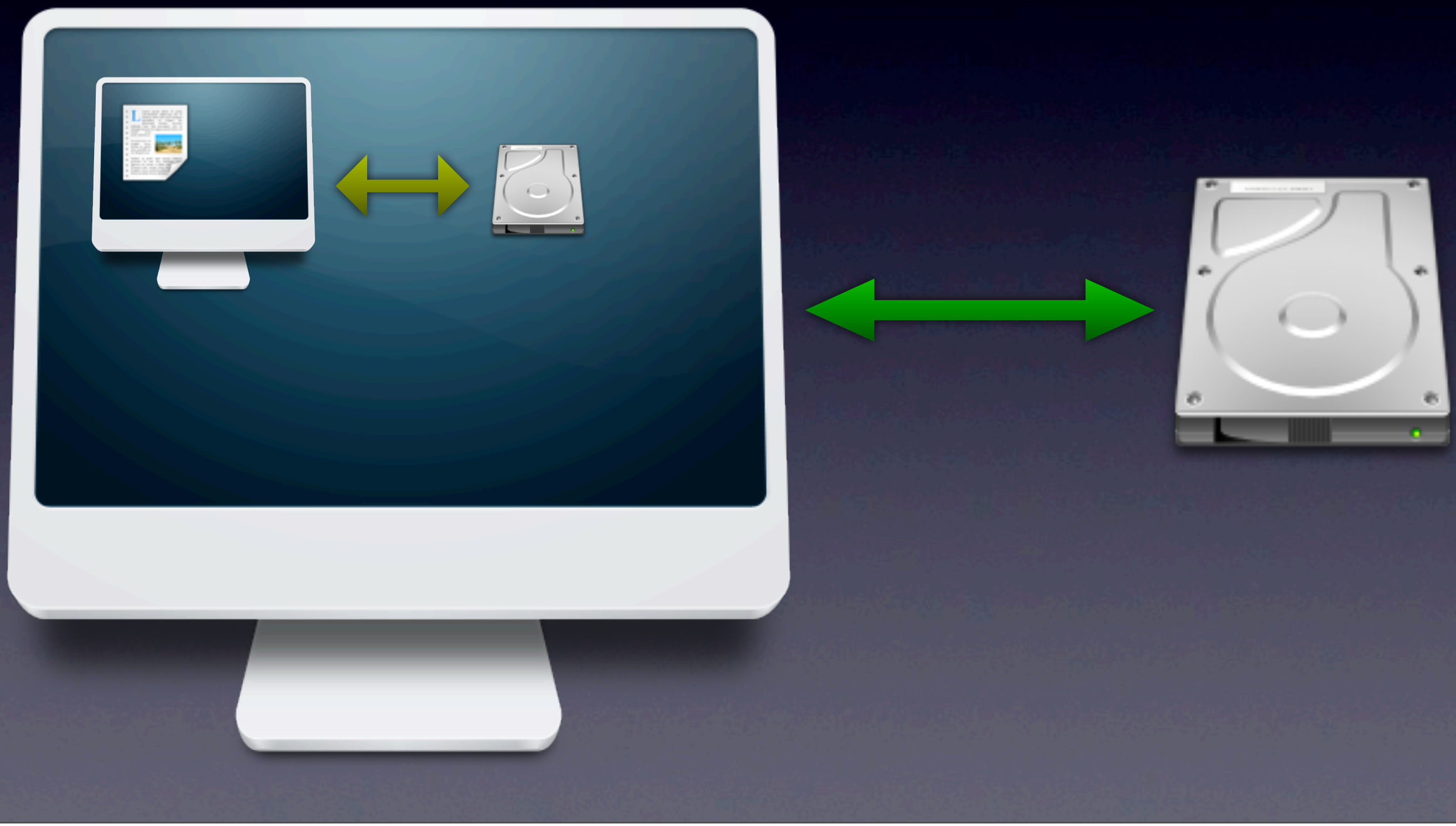
Storage



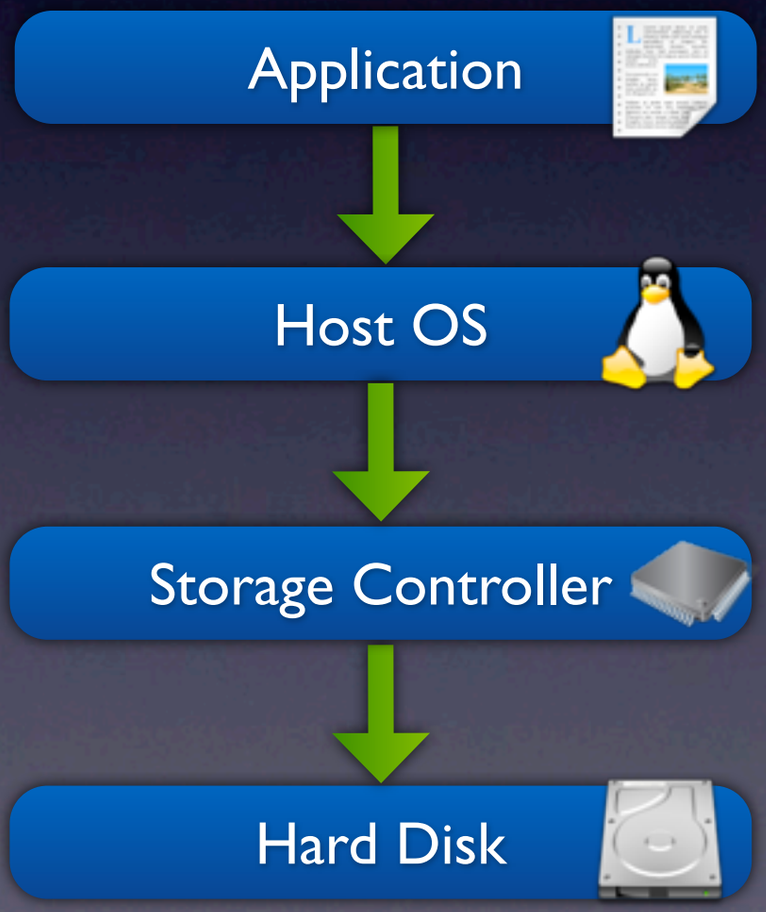
Virtualization

Virtualization

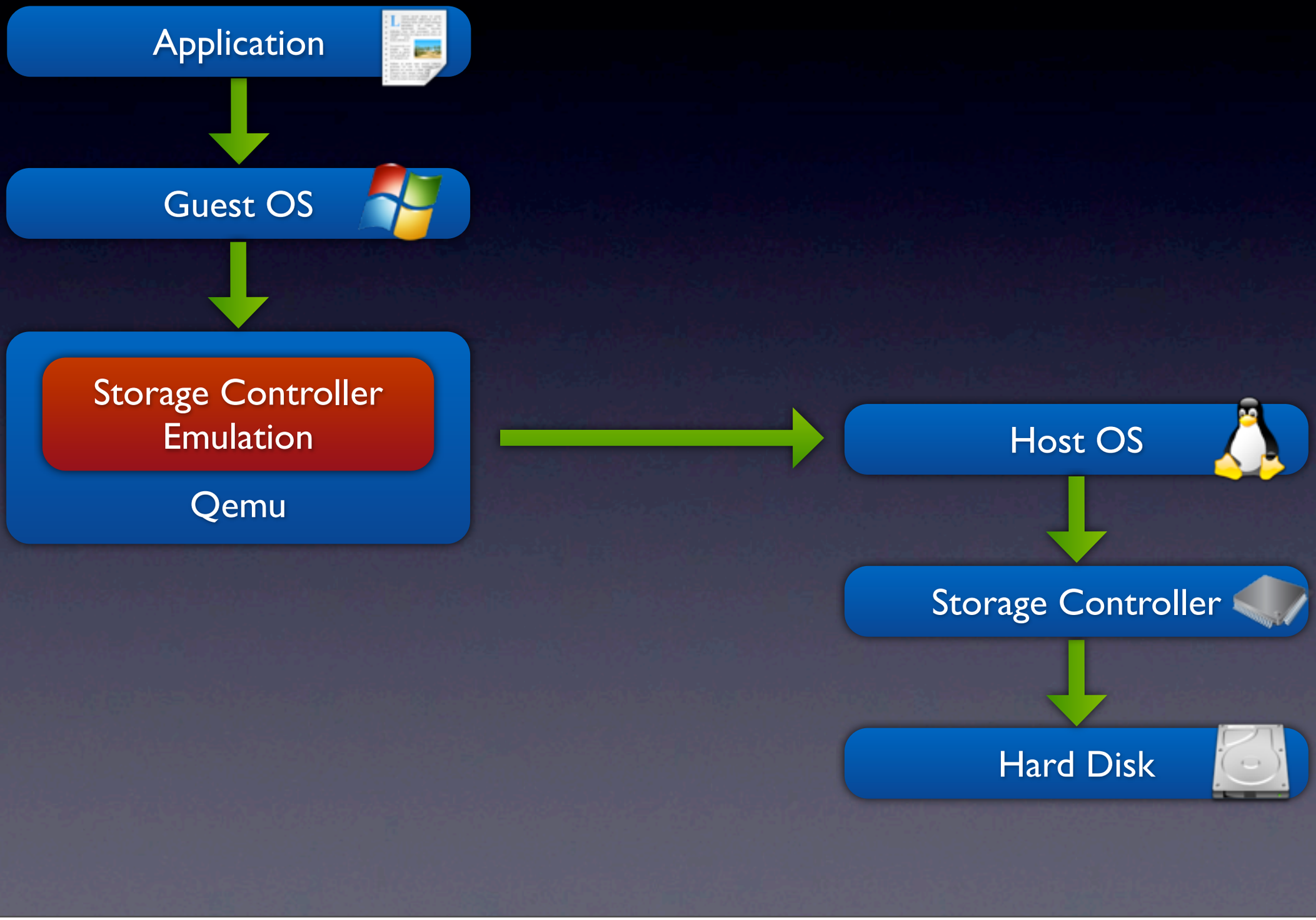
Virtual Storage



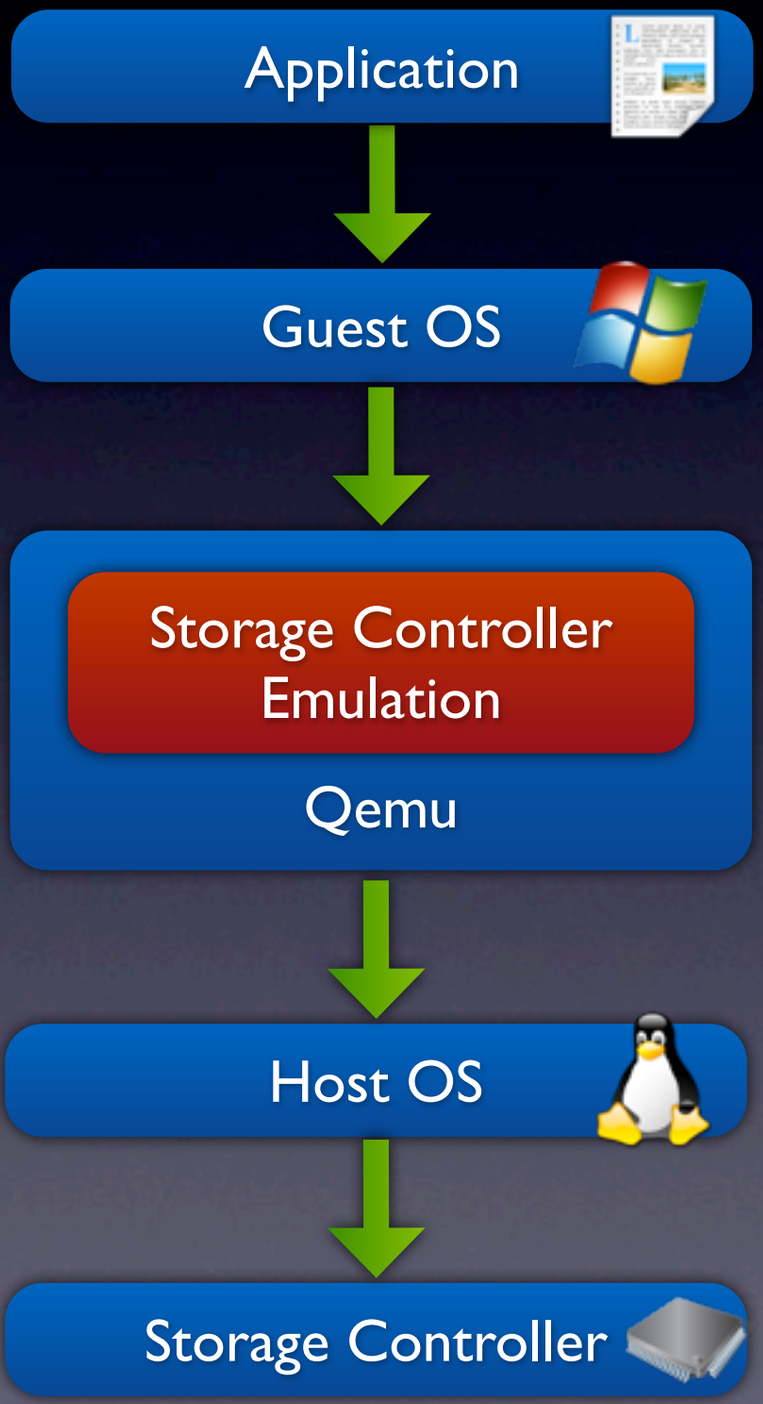
Virtualized Storage



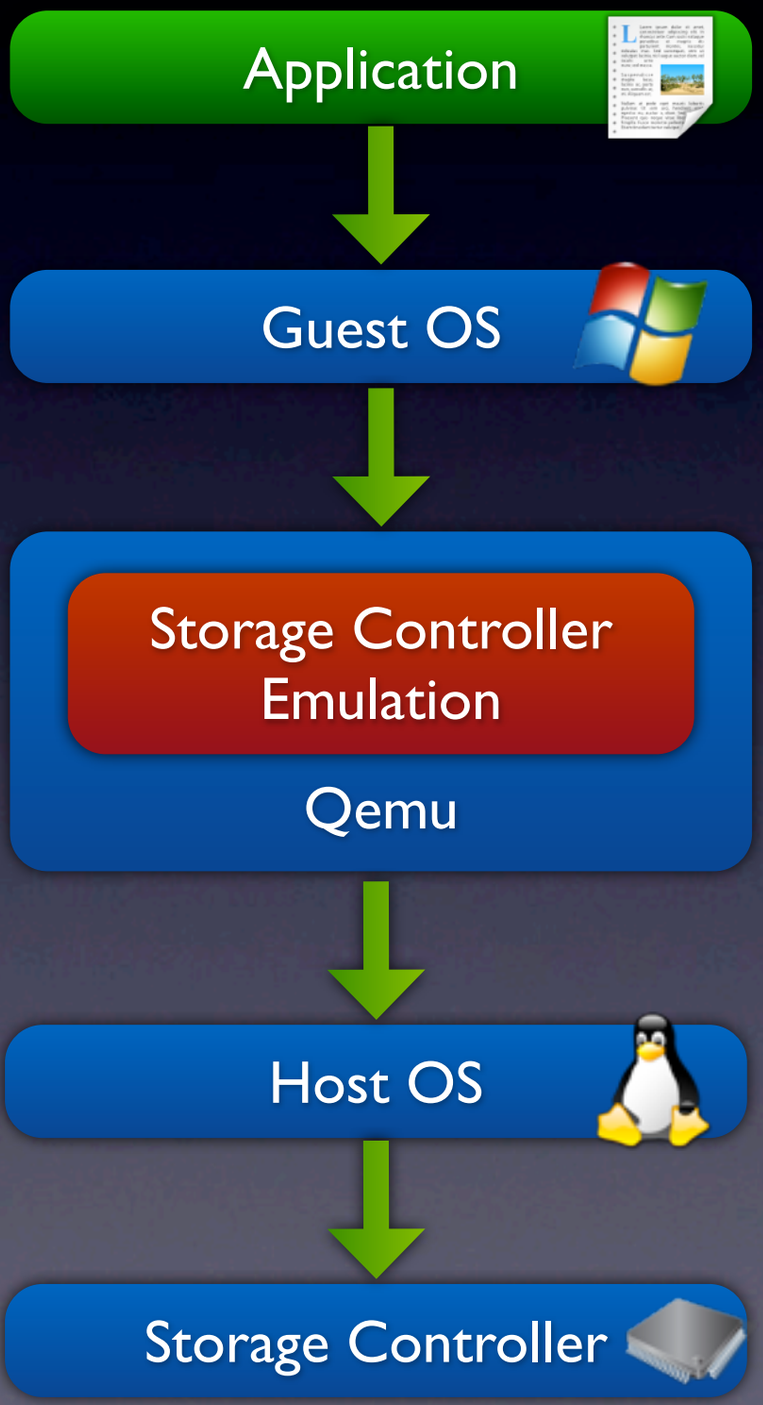
Virtualized Storage



Virtualized Storage

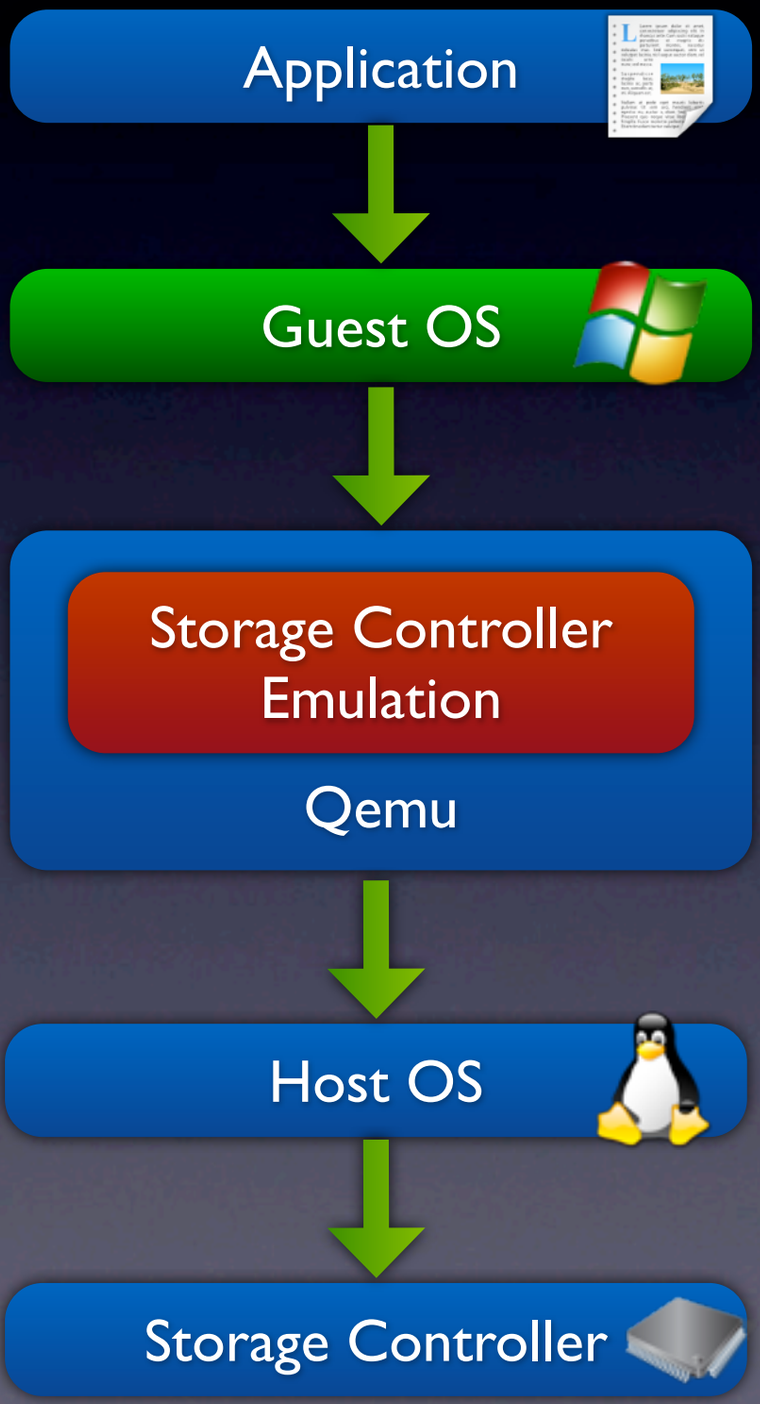


Virtualized Storage

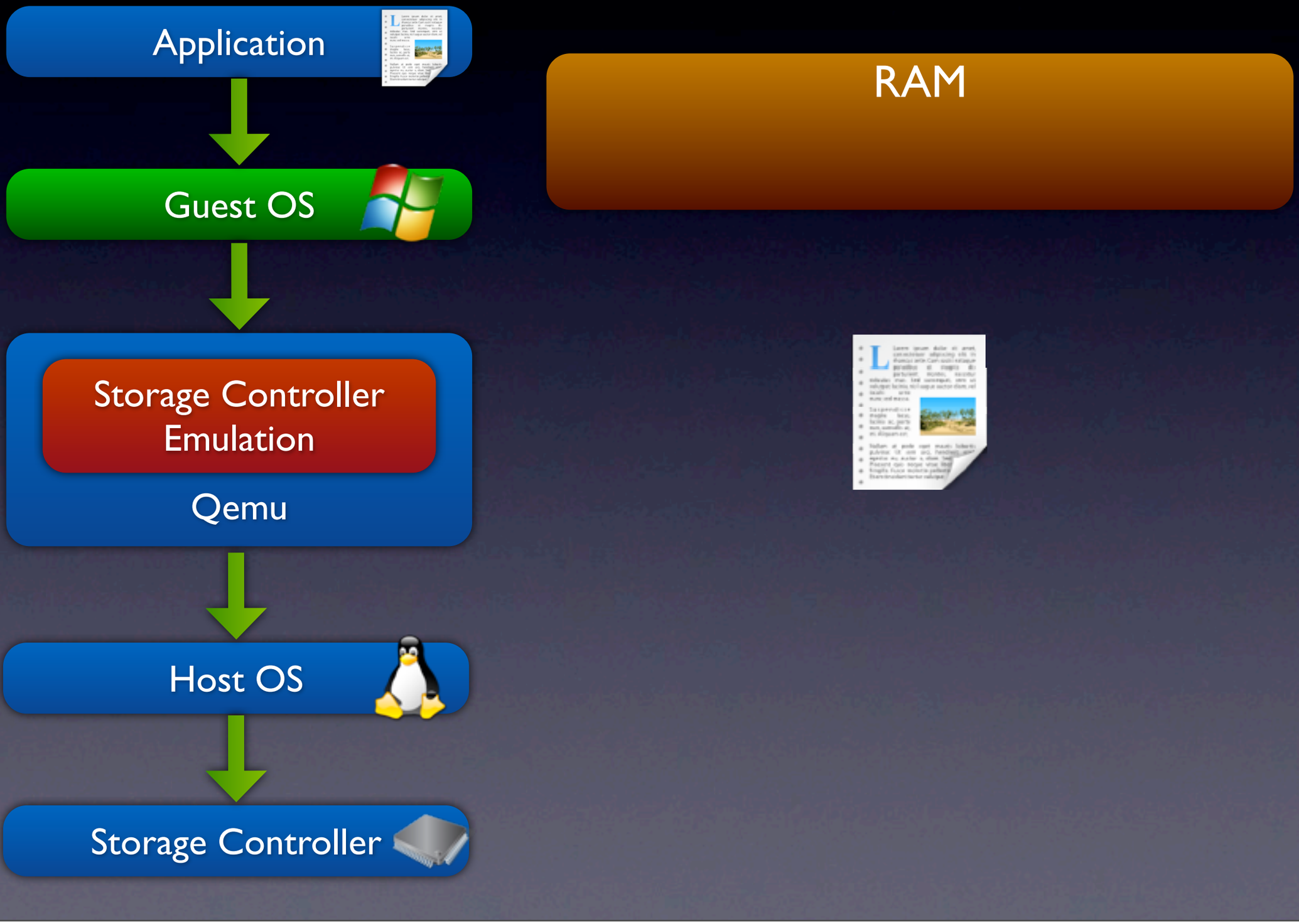


write( )

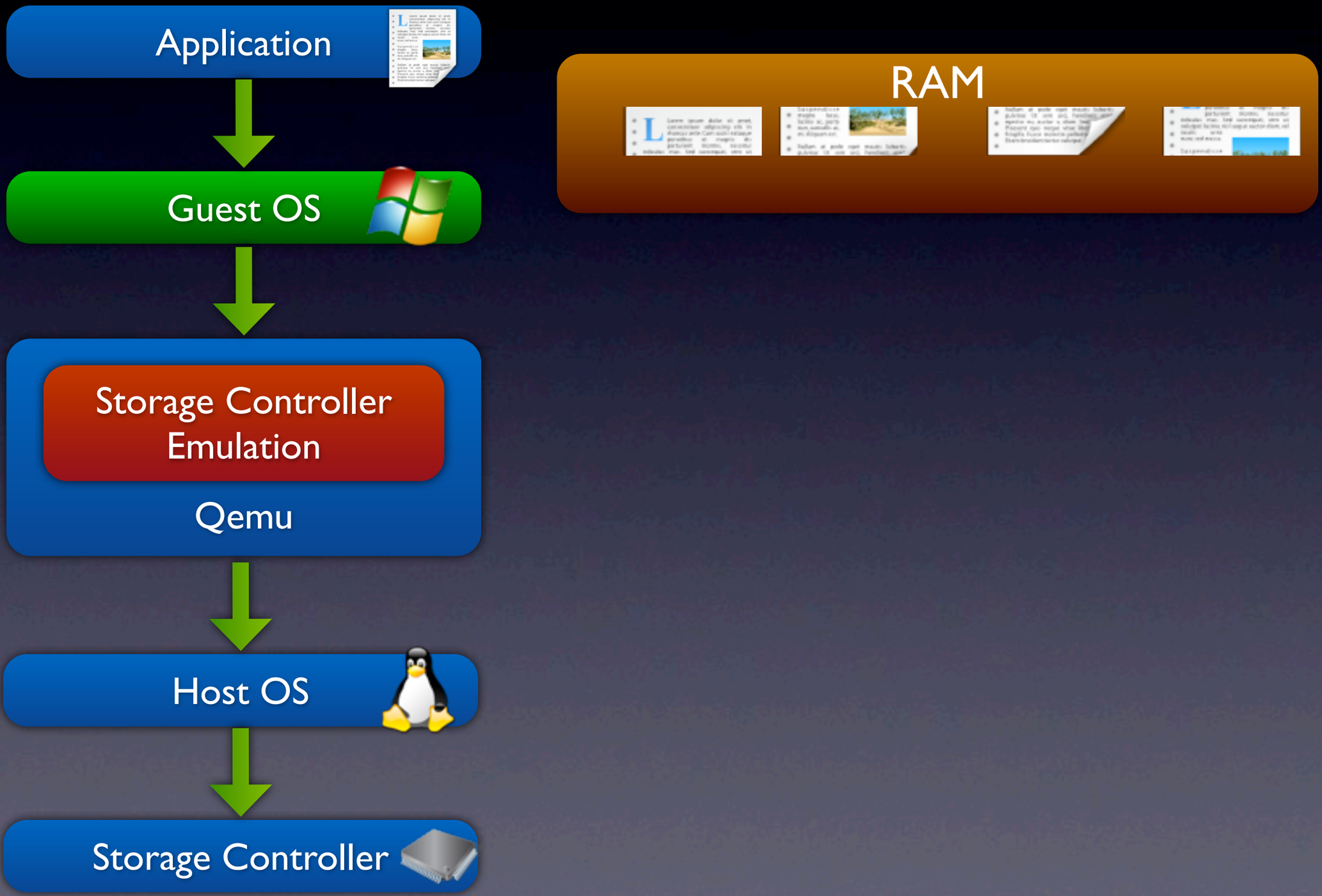
Virtualized Storage



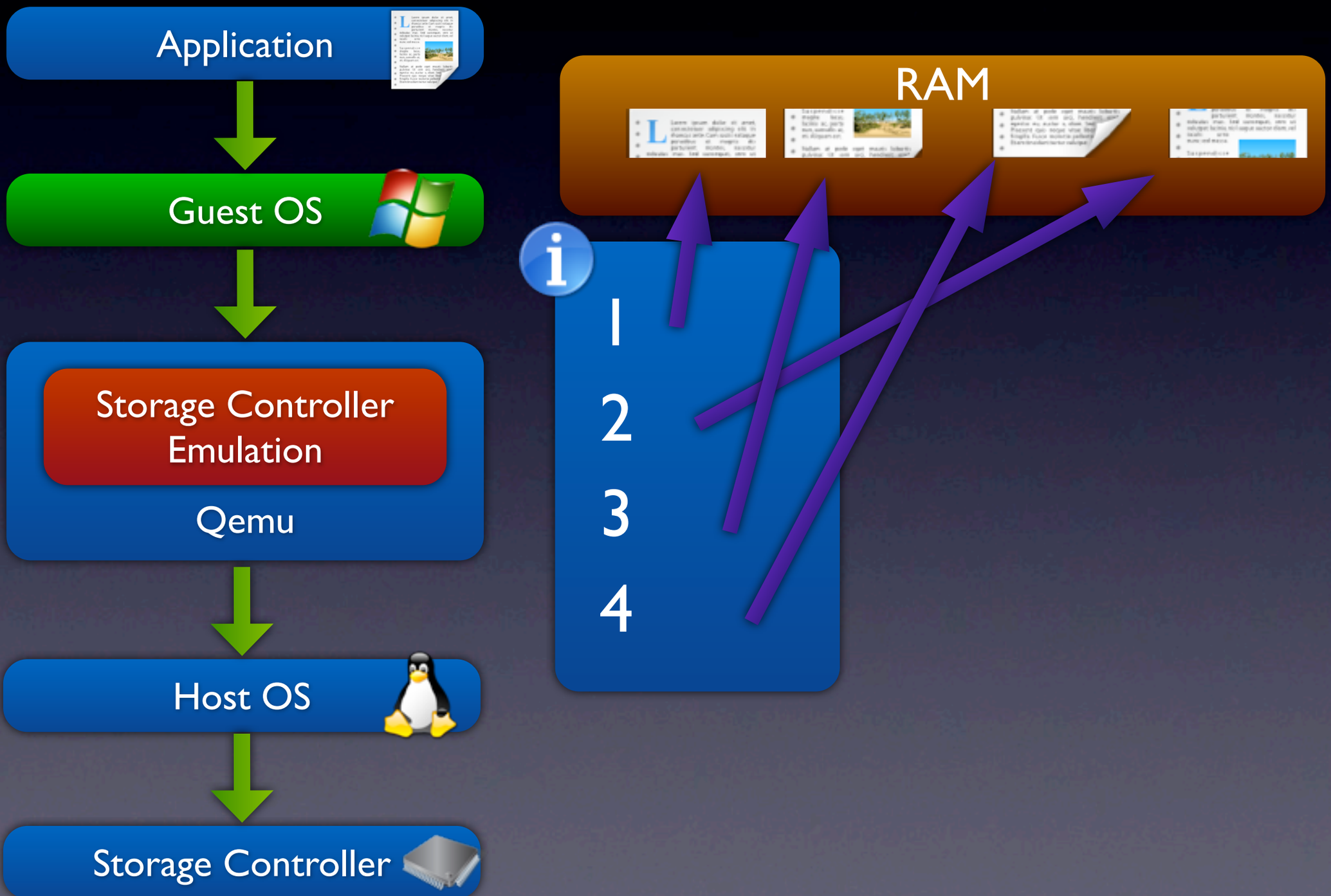
Virtualized Storage



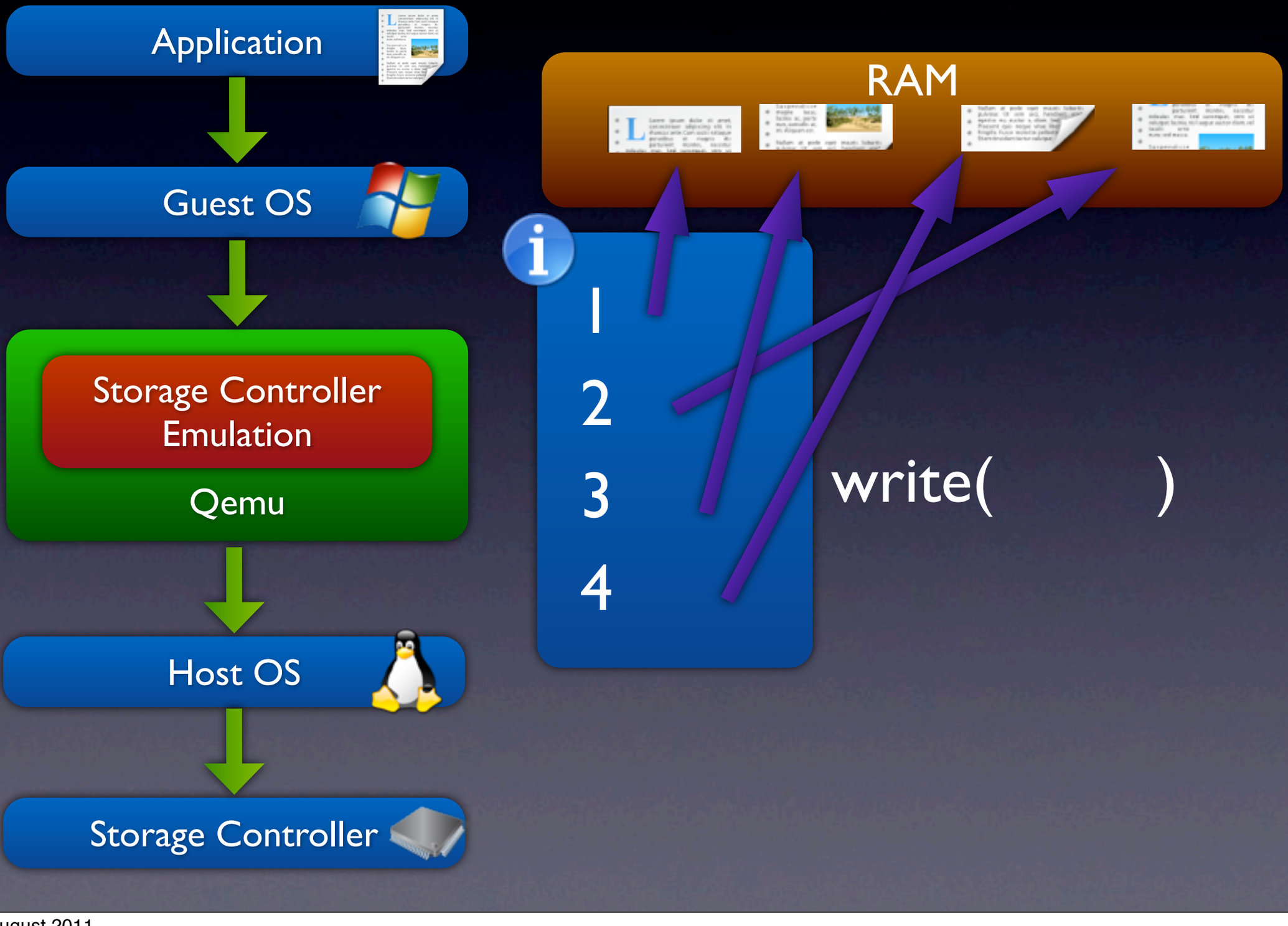
Virtualized Storage



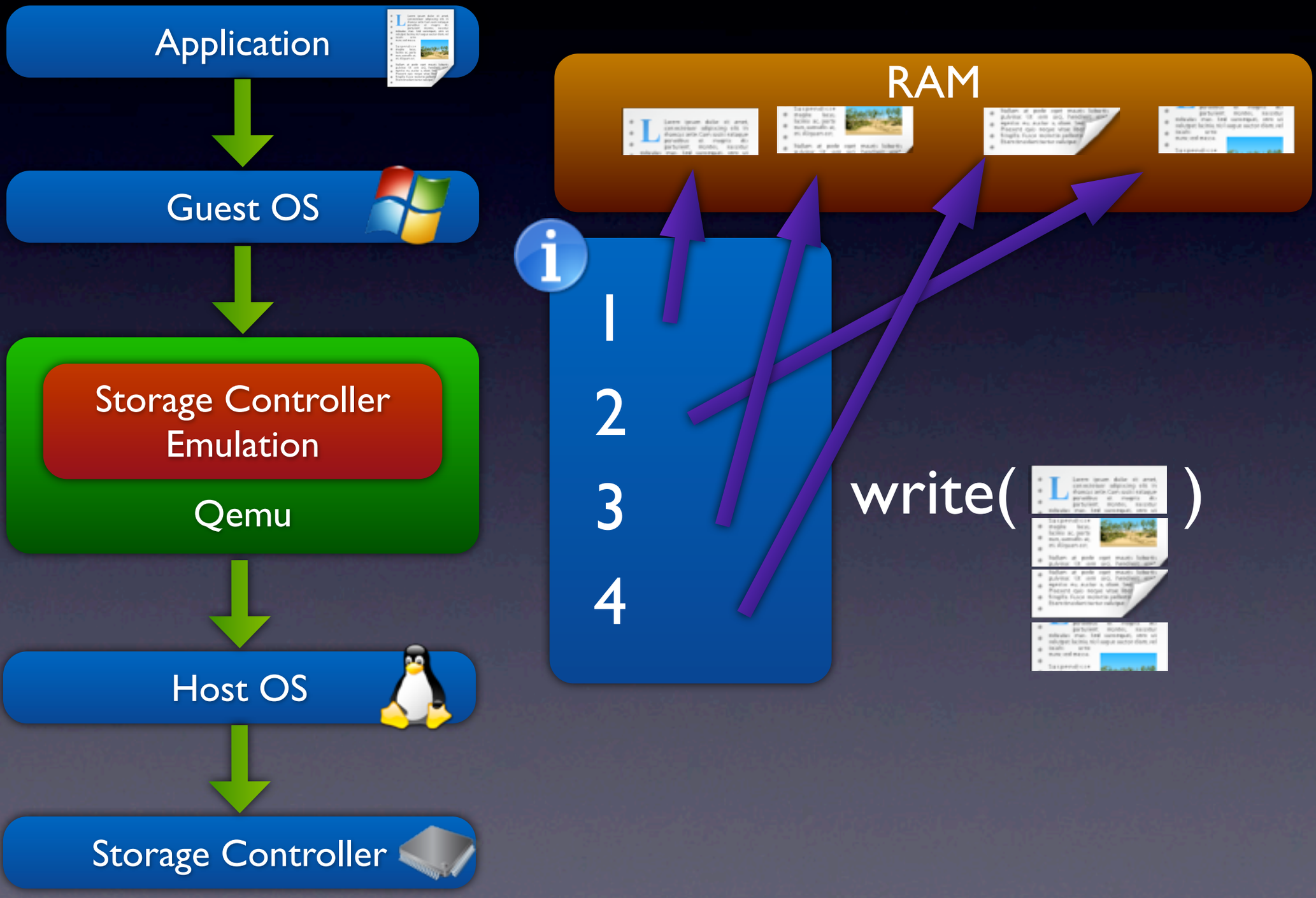
Virtualized Storage



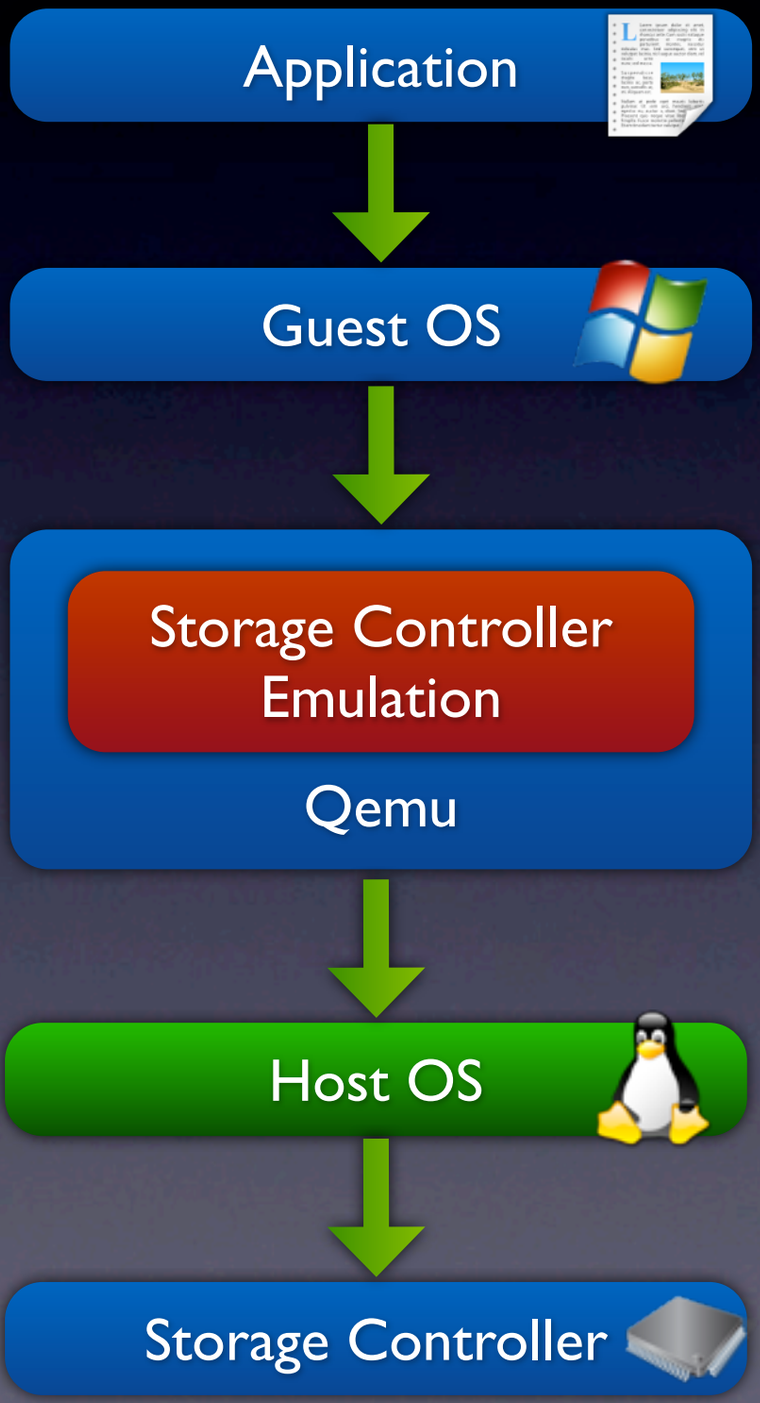
Virtualized Storage

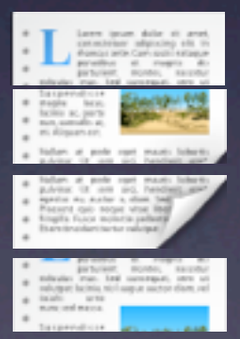


Virtualized Storage

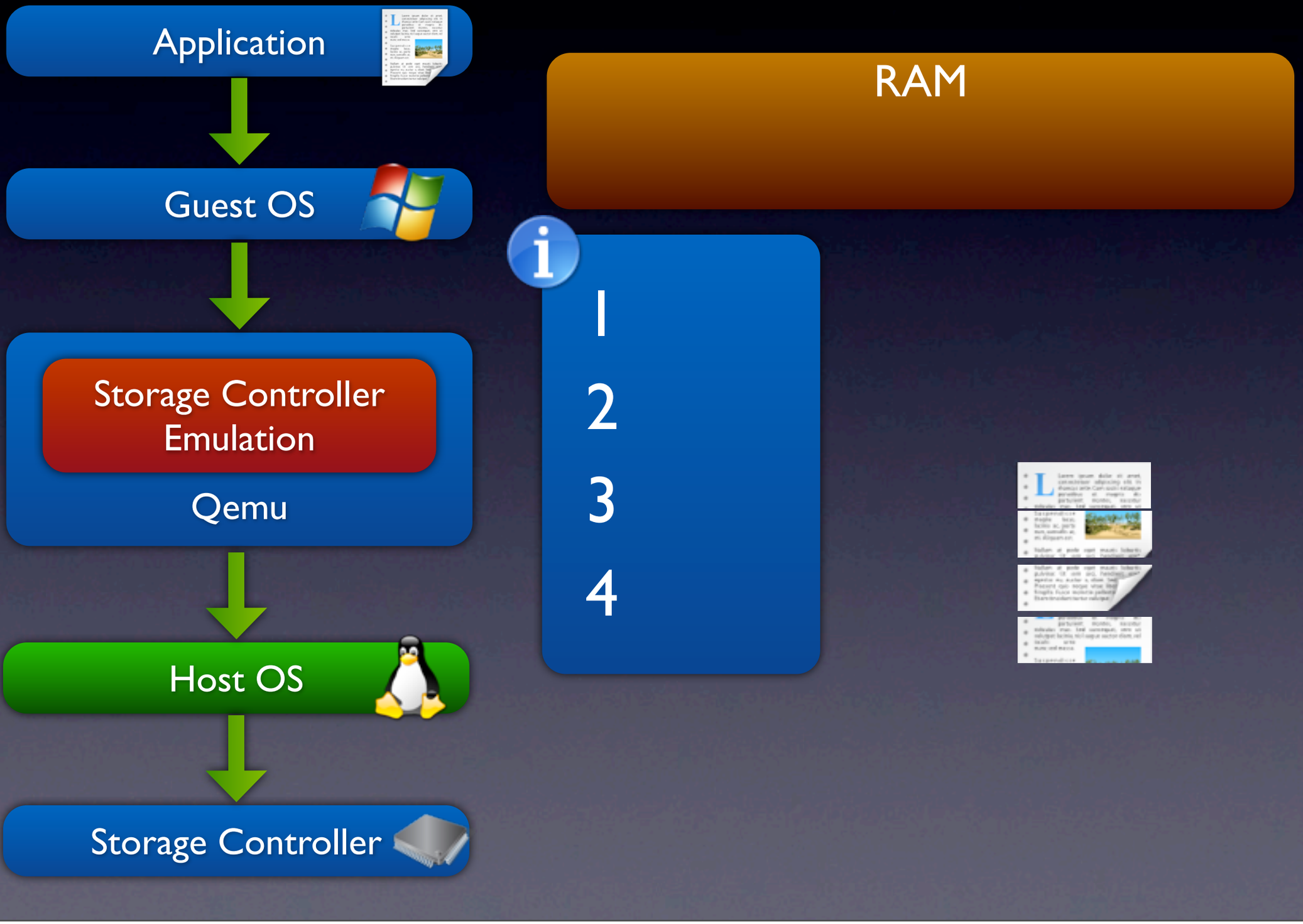


Virtualized Storage

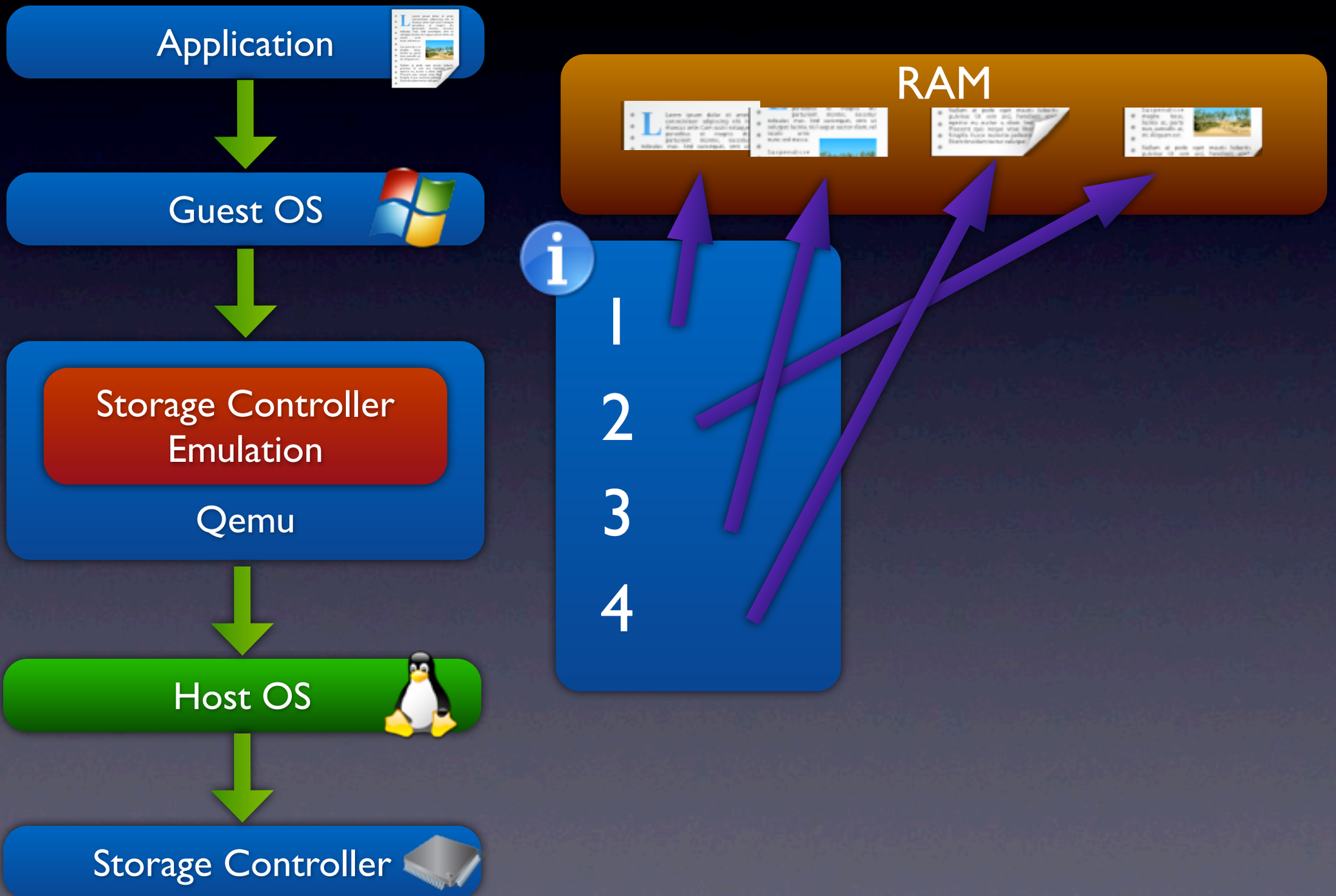




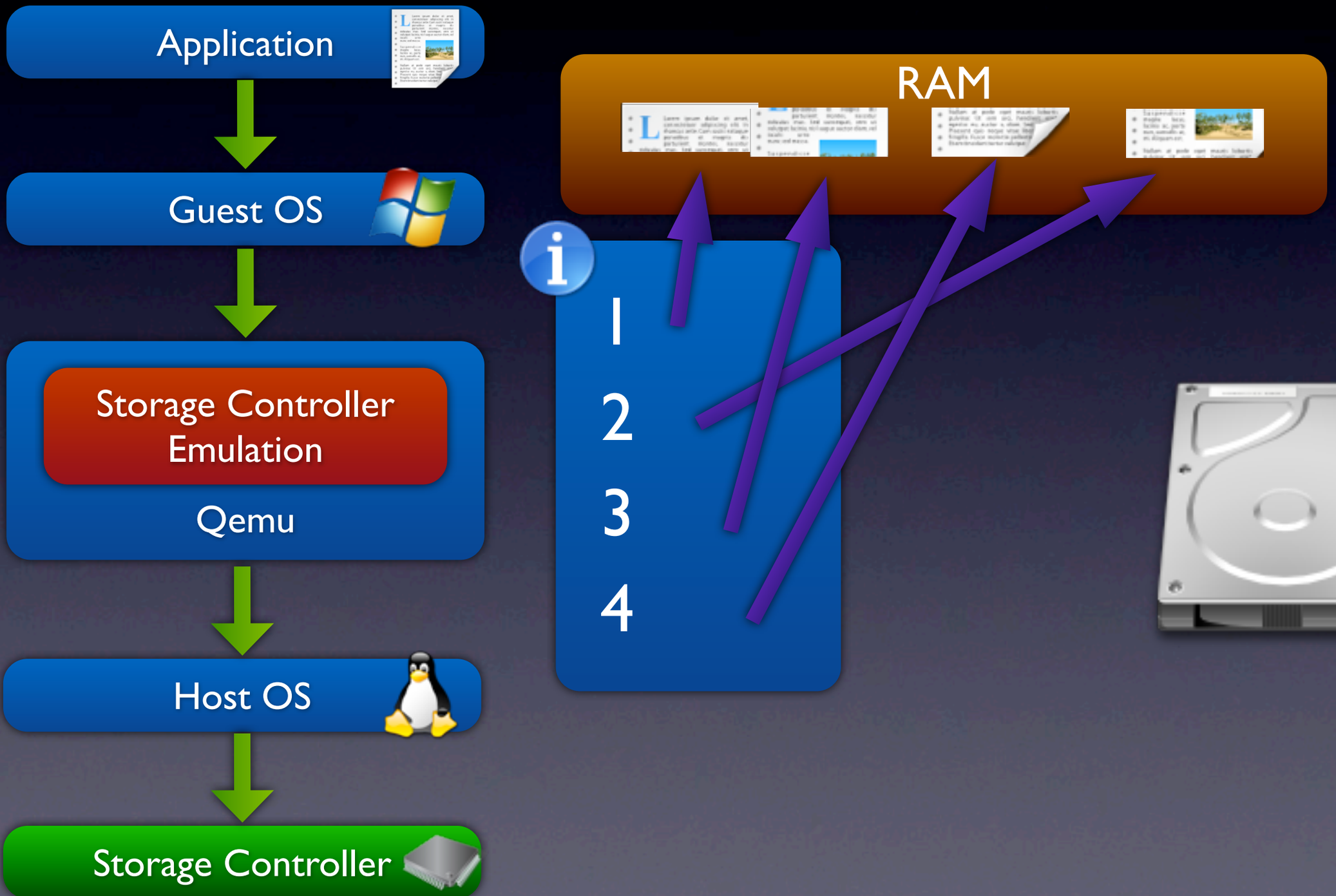
Virtualized Storage



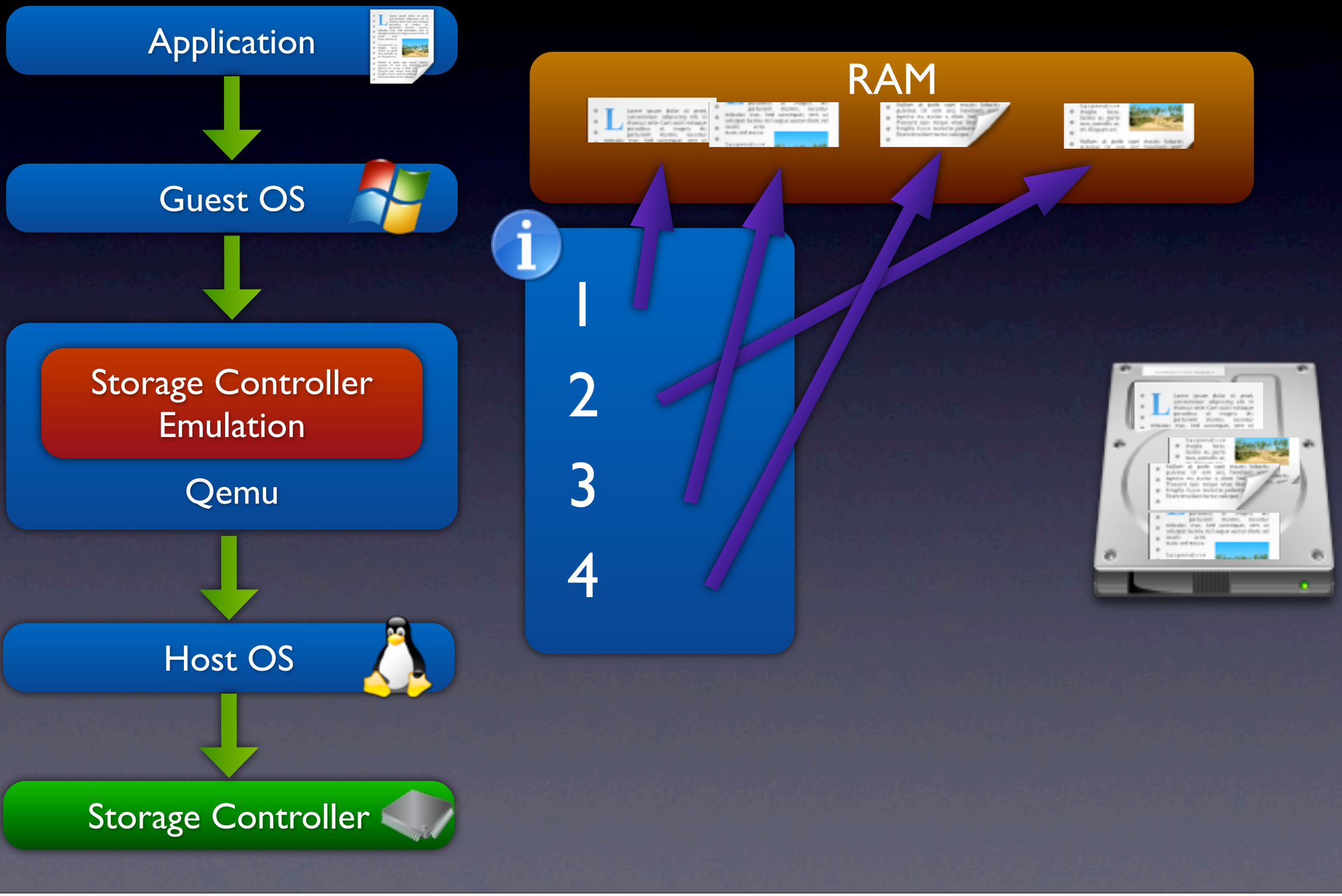
Virtualized Storage



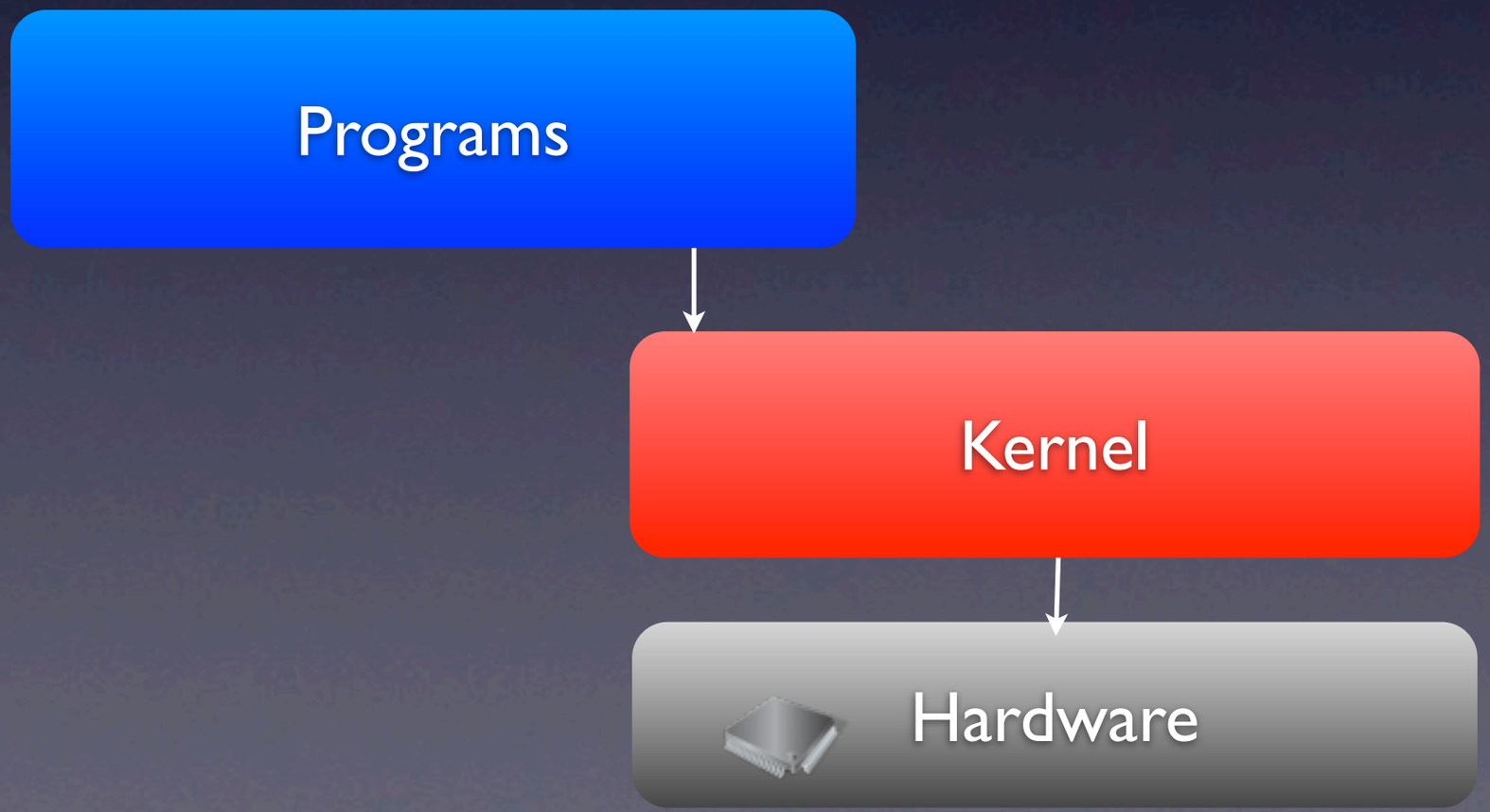
Virtualized Storage



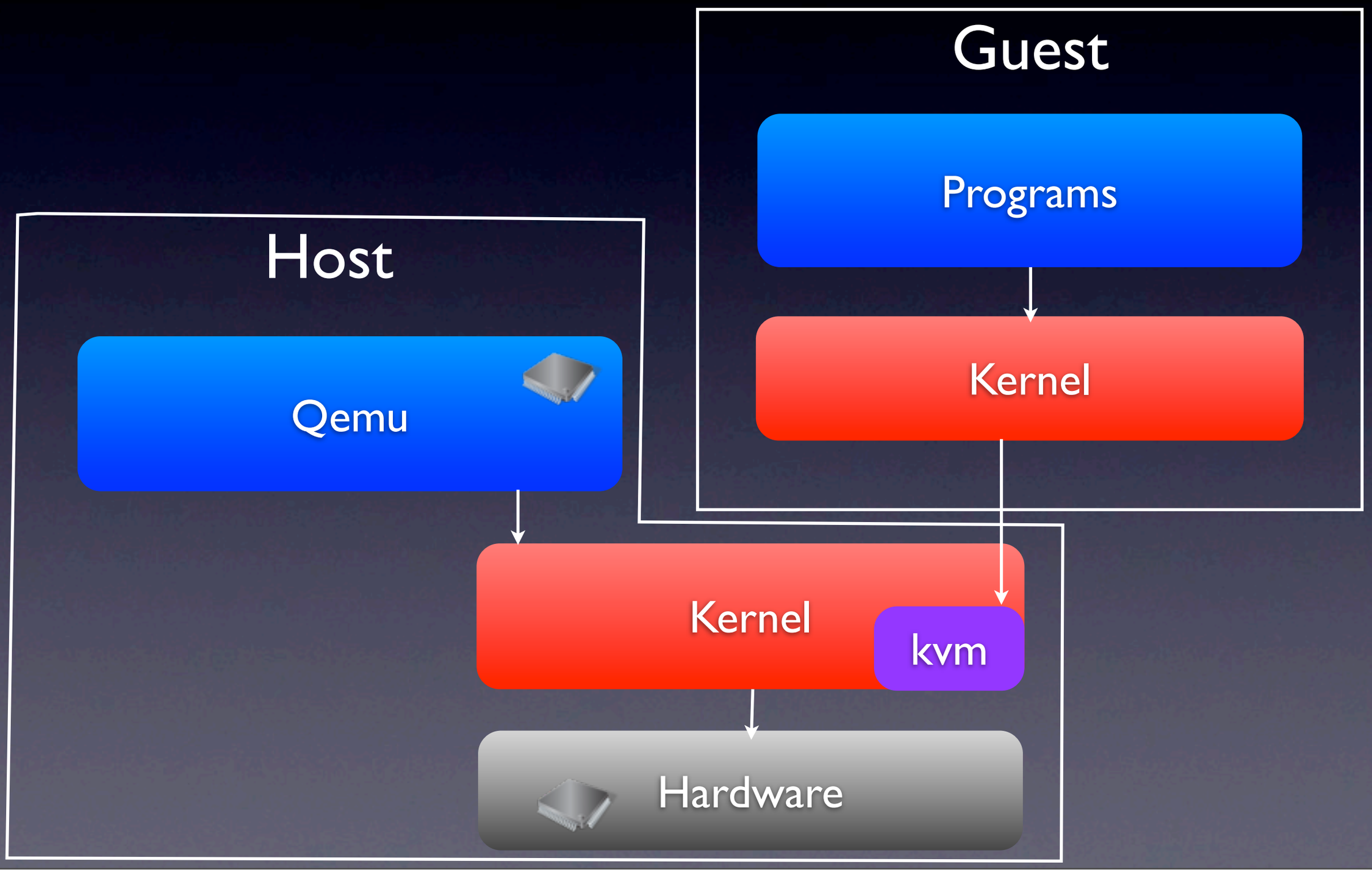
Virtualized Storage



Device Access

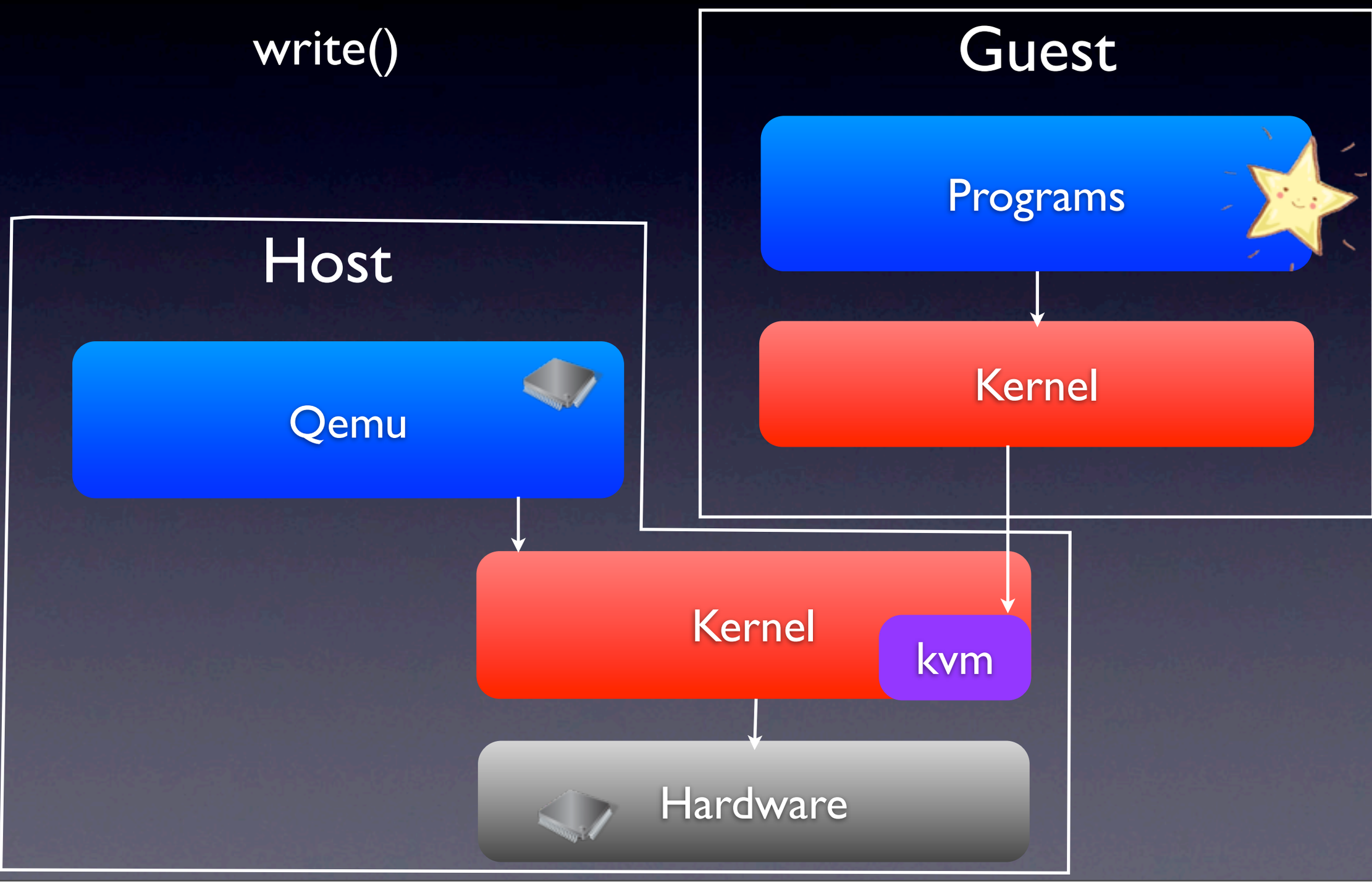


Device Access



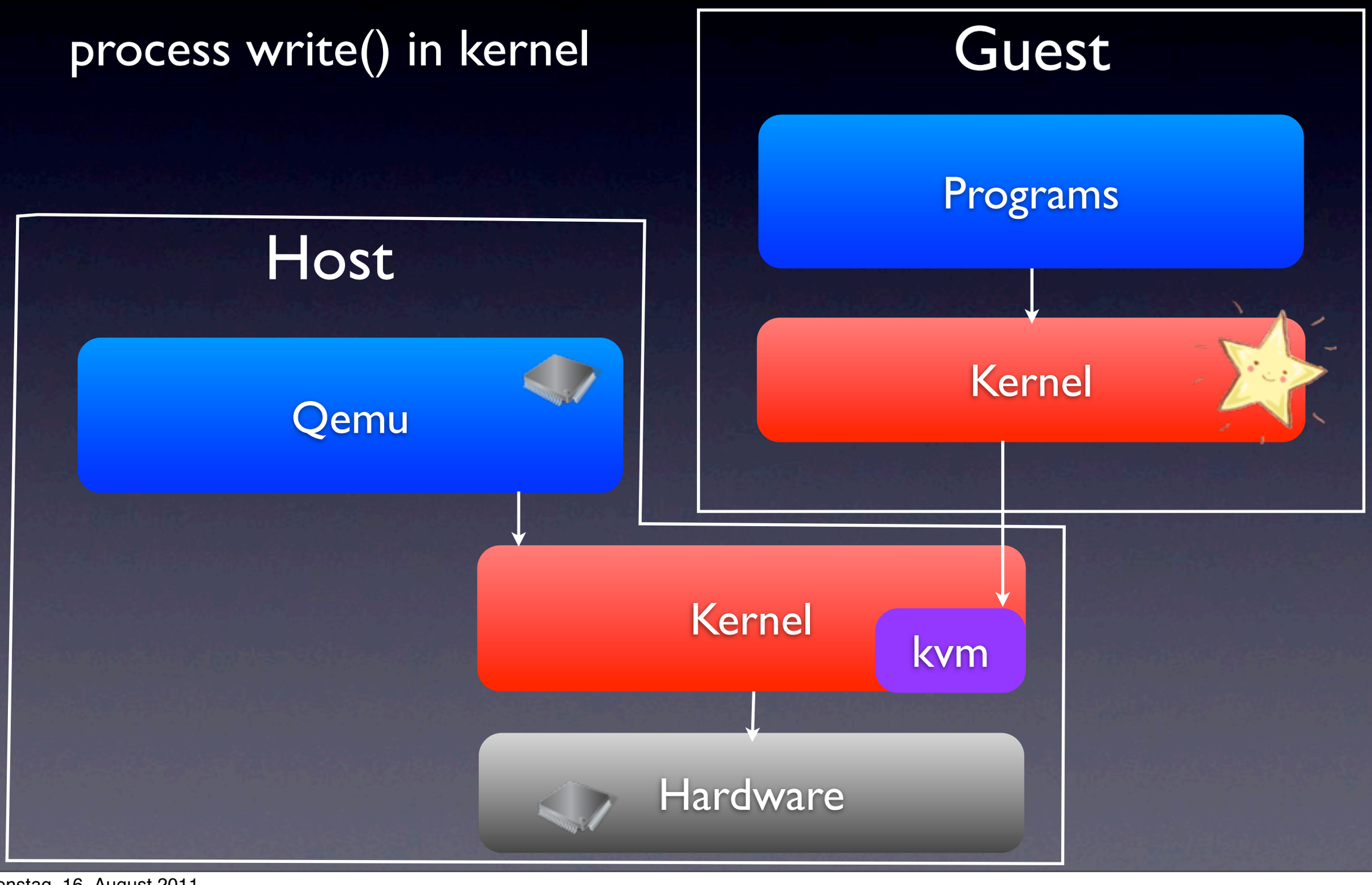
Device Access

write()



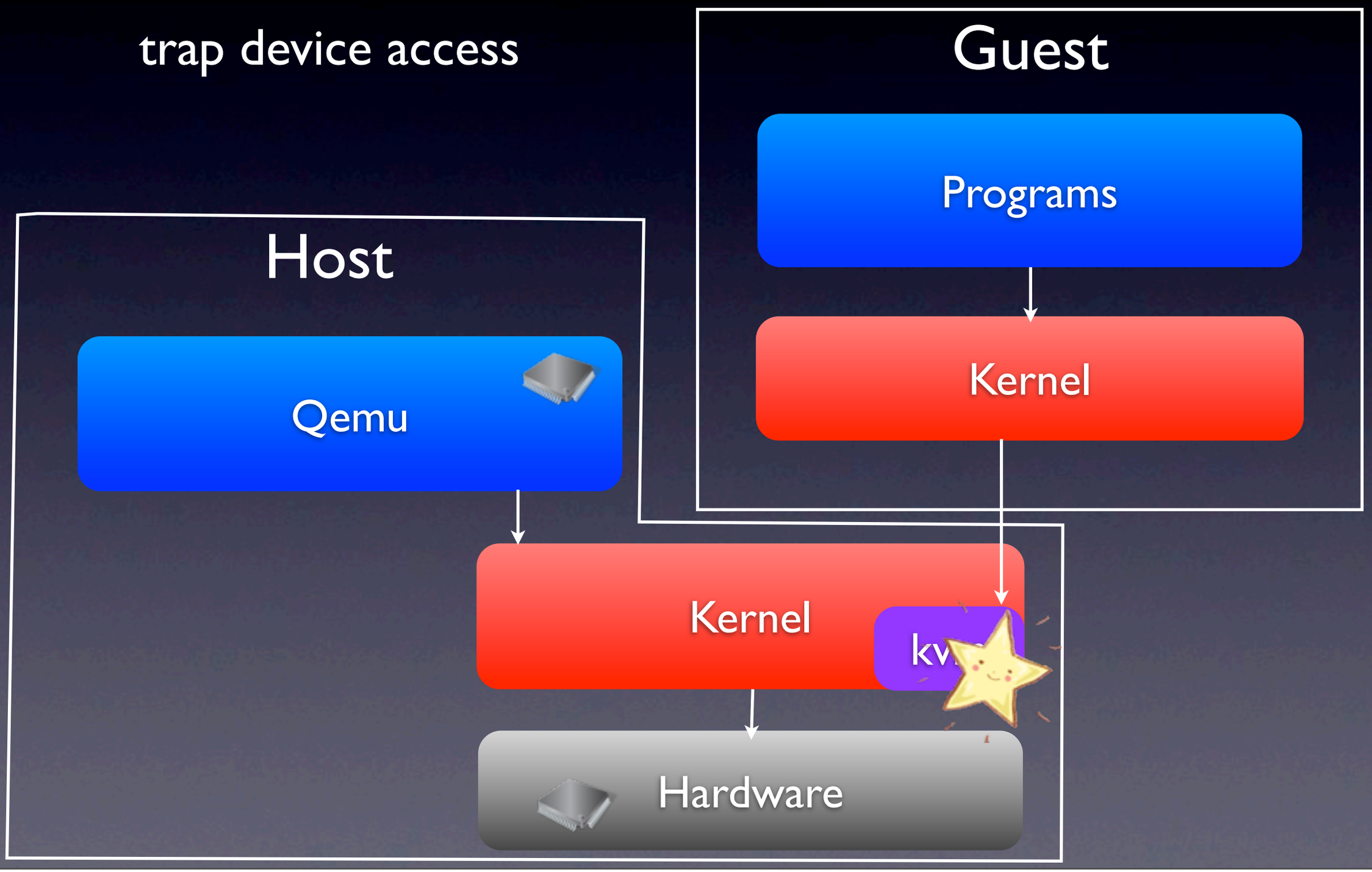
Device Access

process write() in kernel



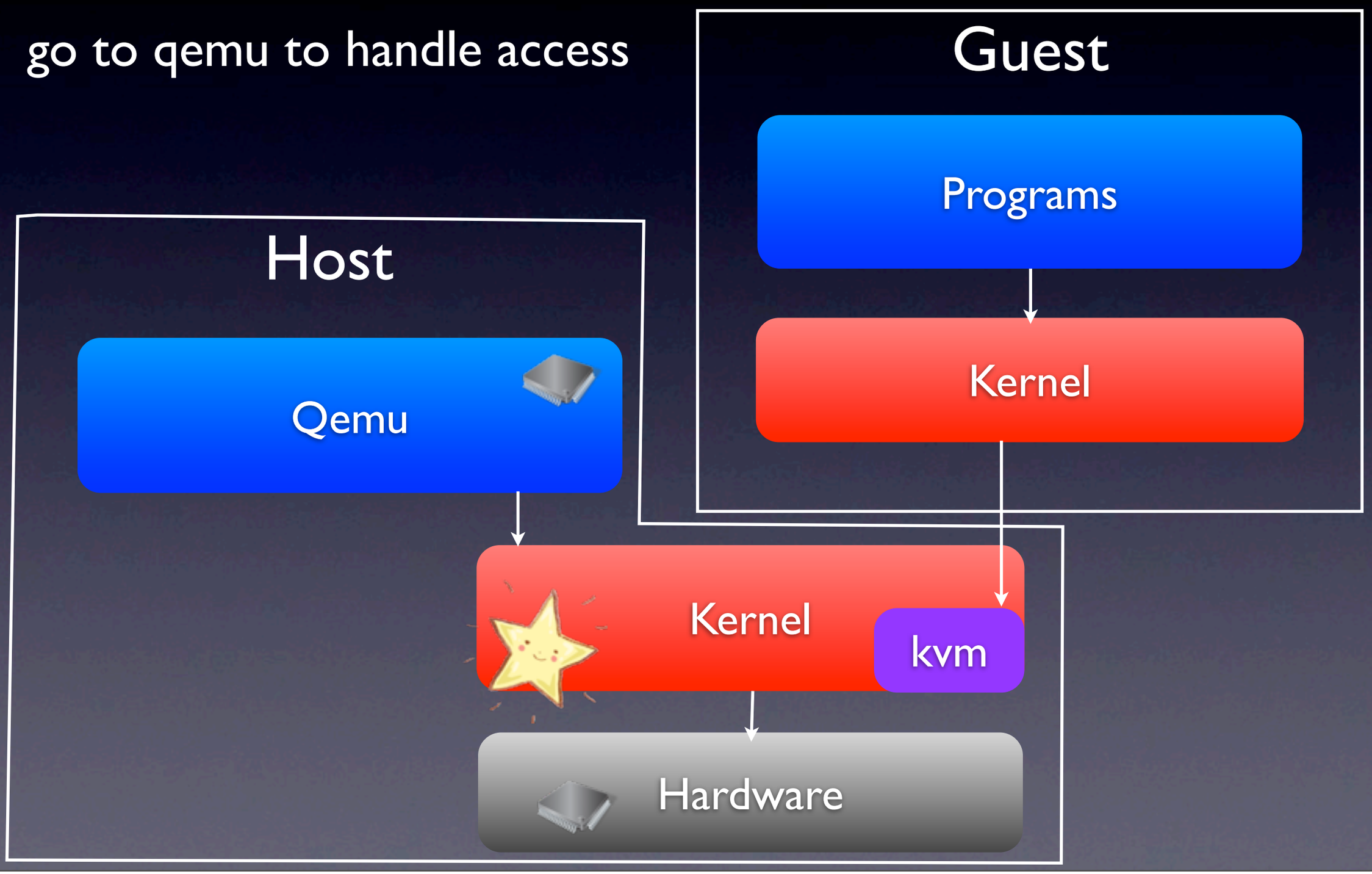
Device Access

trap device access



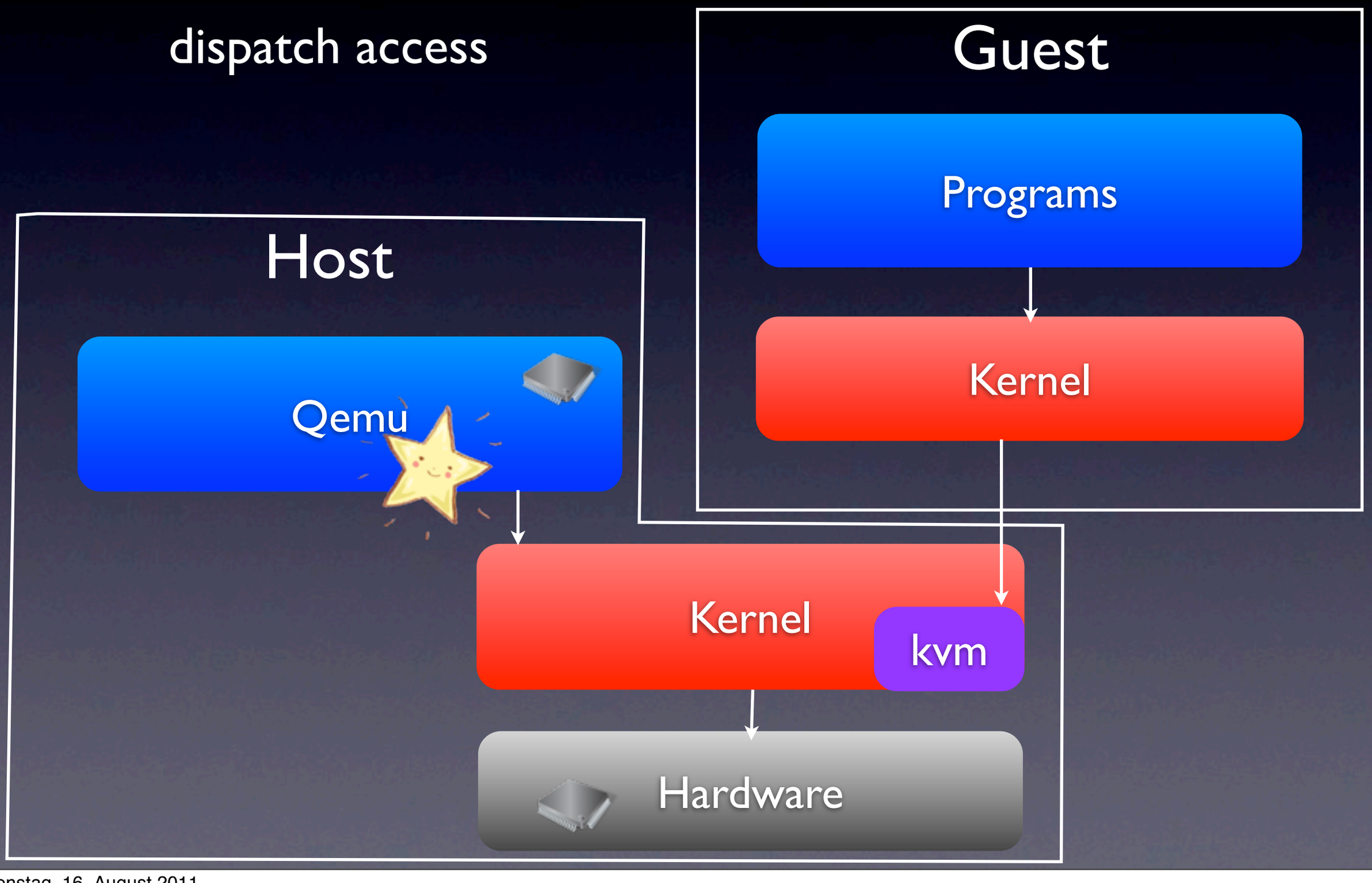
Device Access

go to qemu to handle access



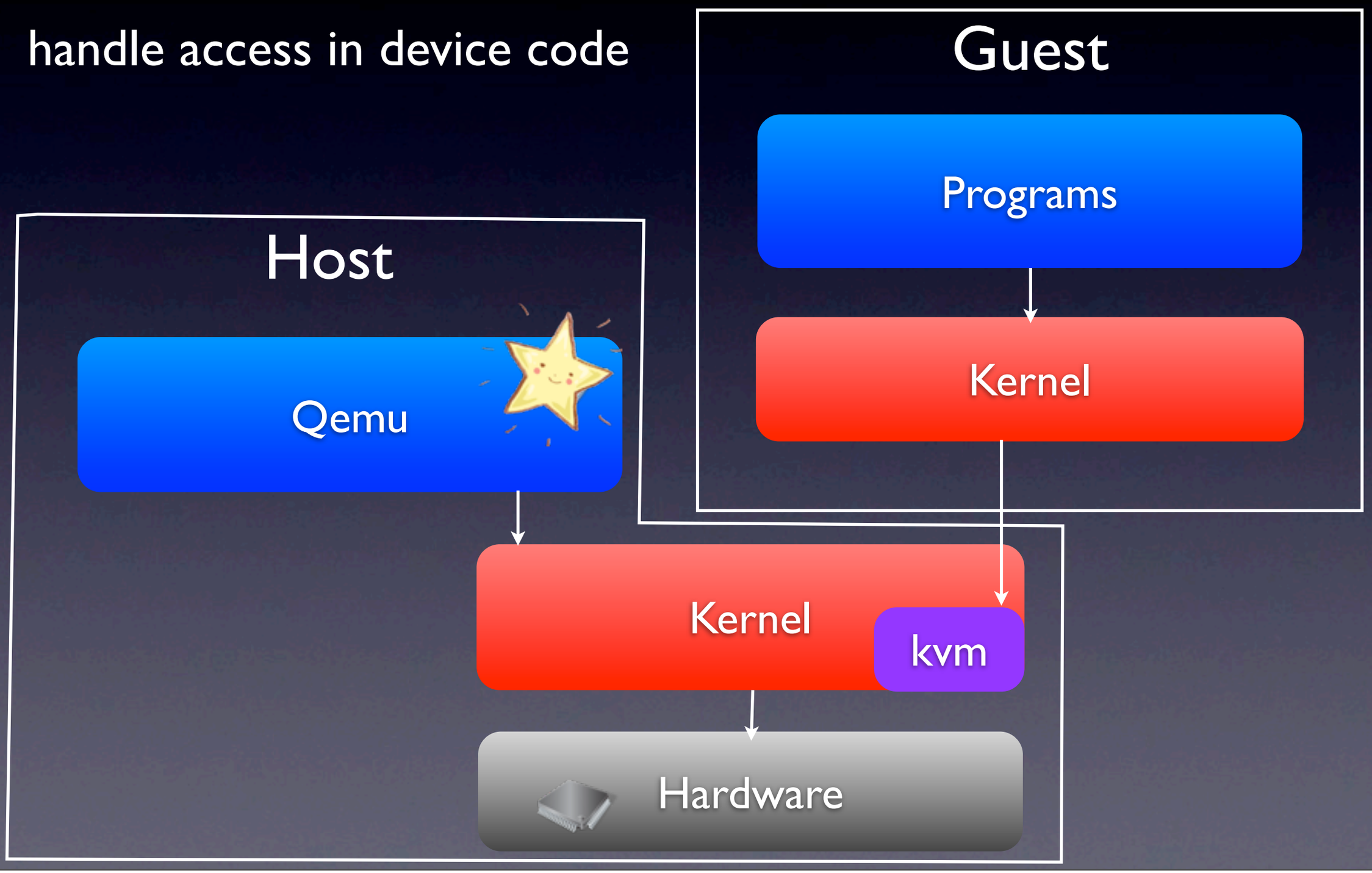
Device Access

dispatch access



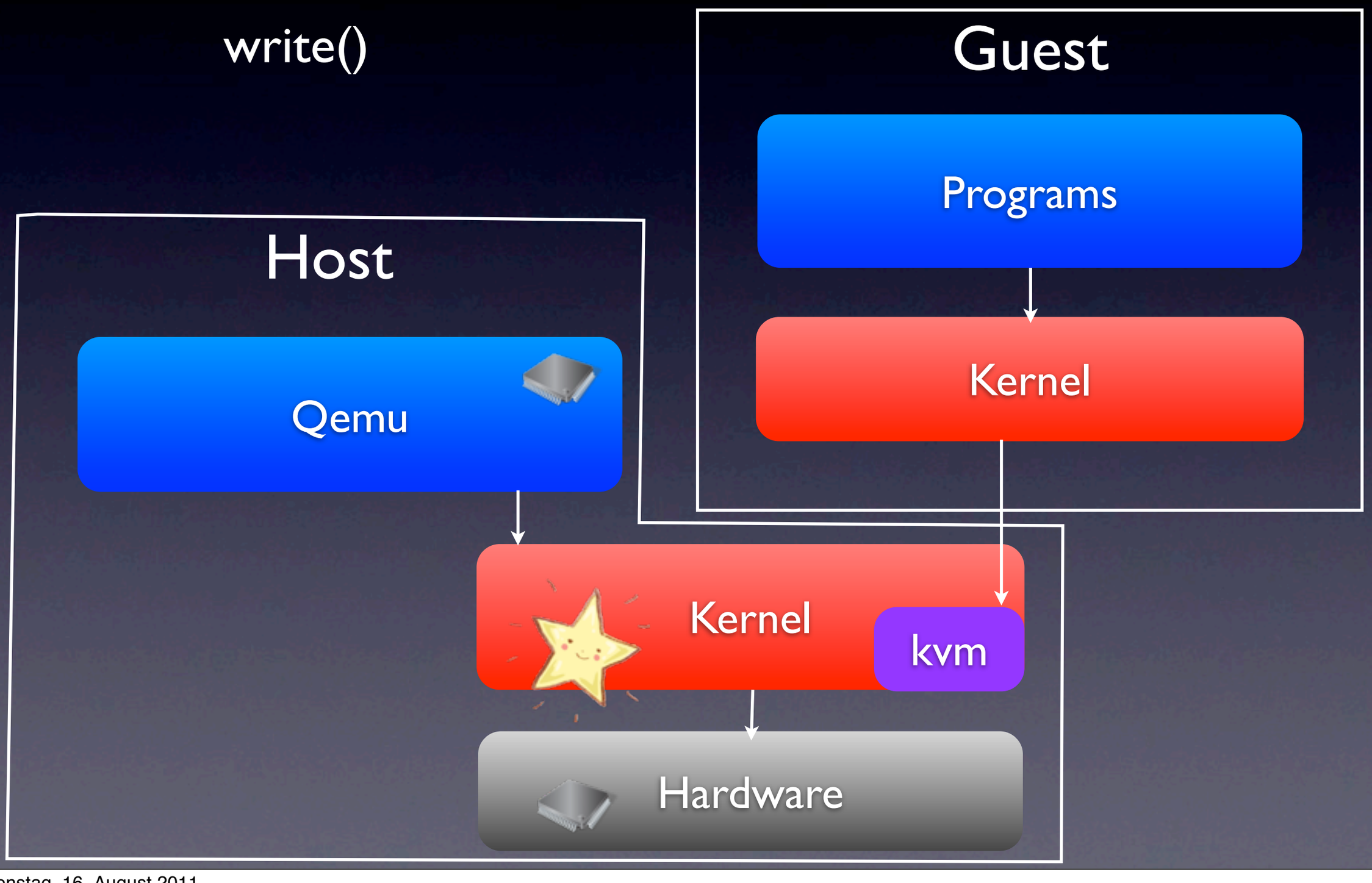
Device Access

handle access in device code



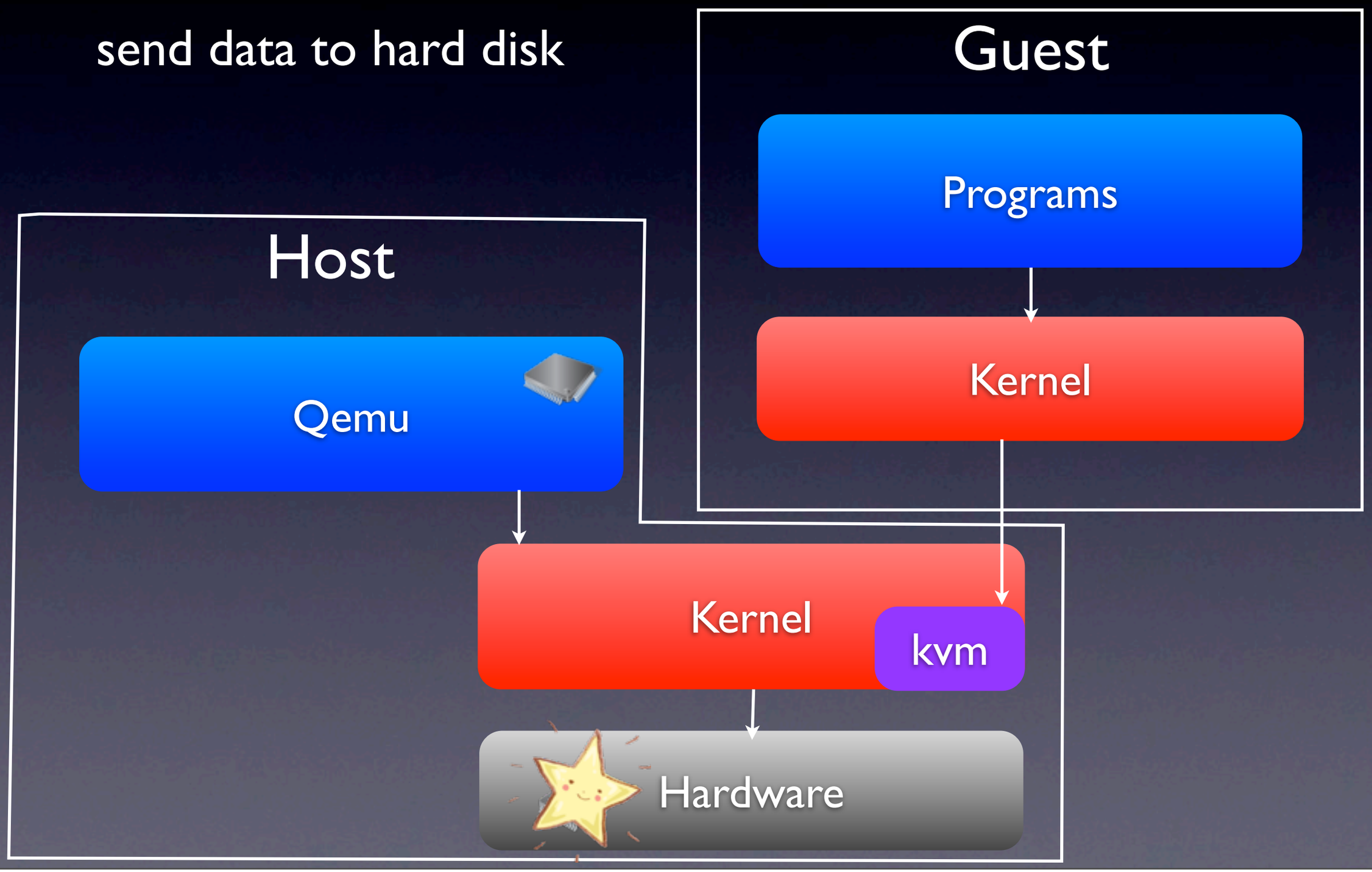
Device Access

write()



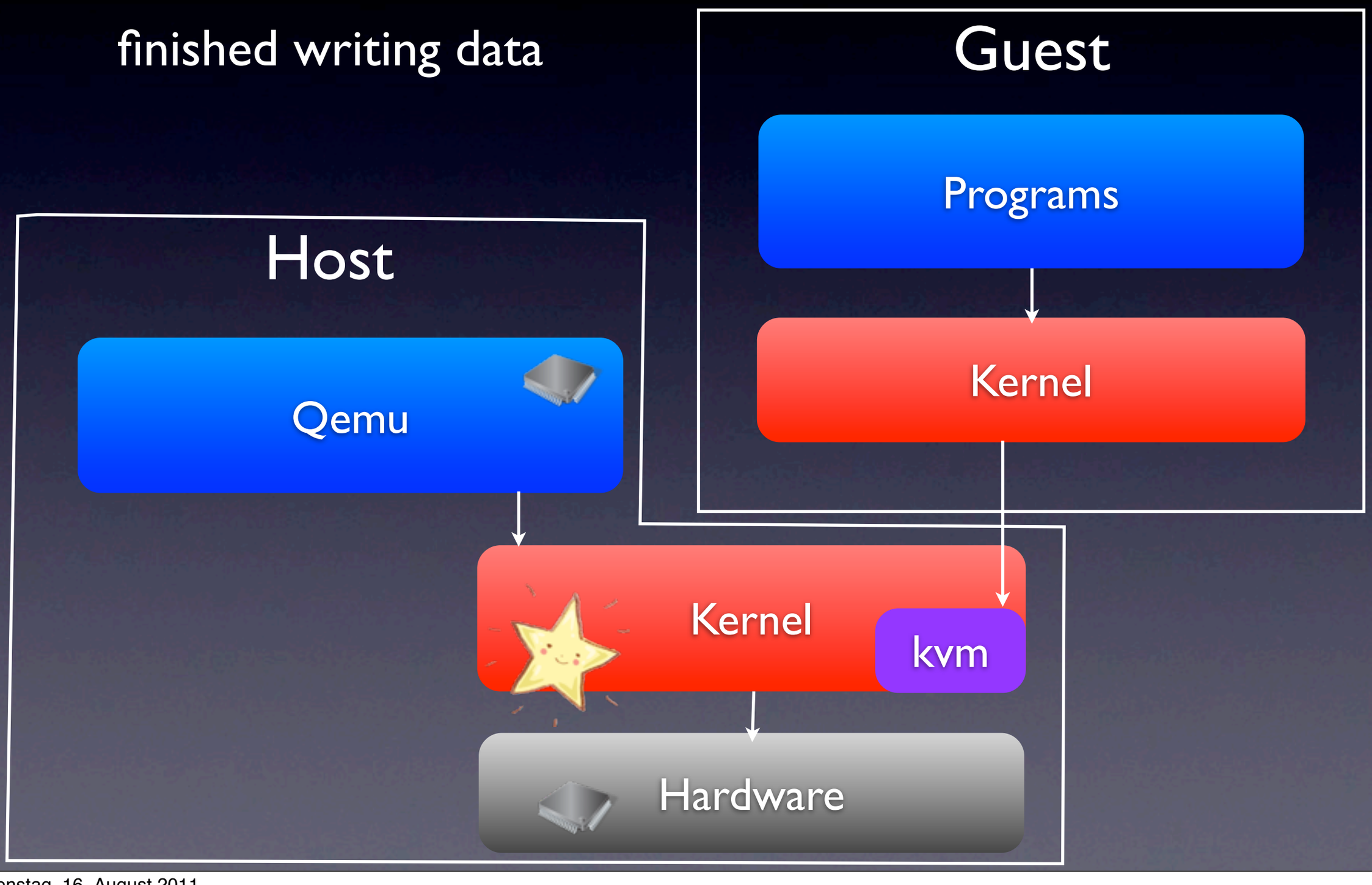
Device Access

send data to hard disk



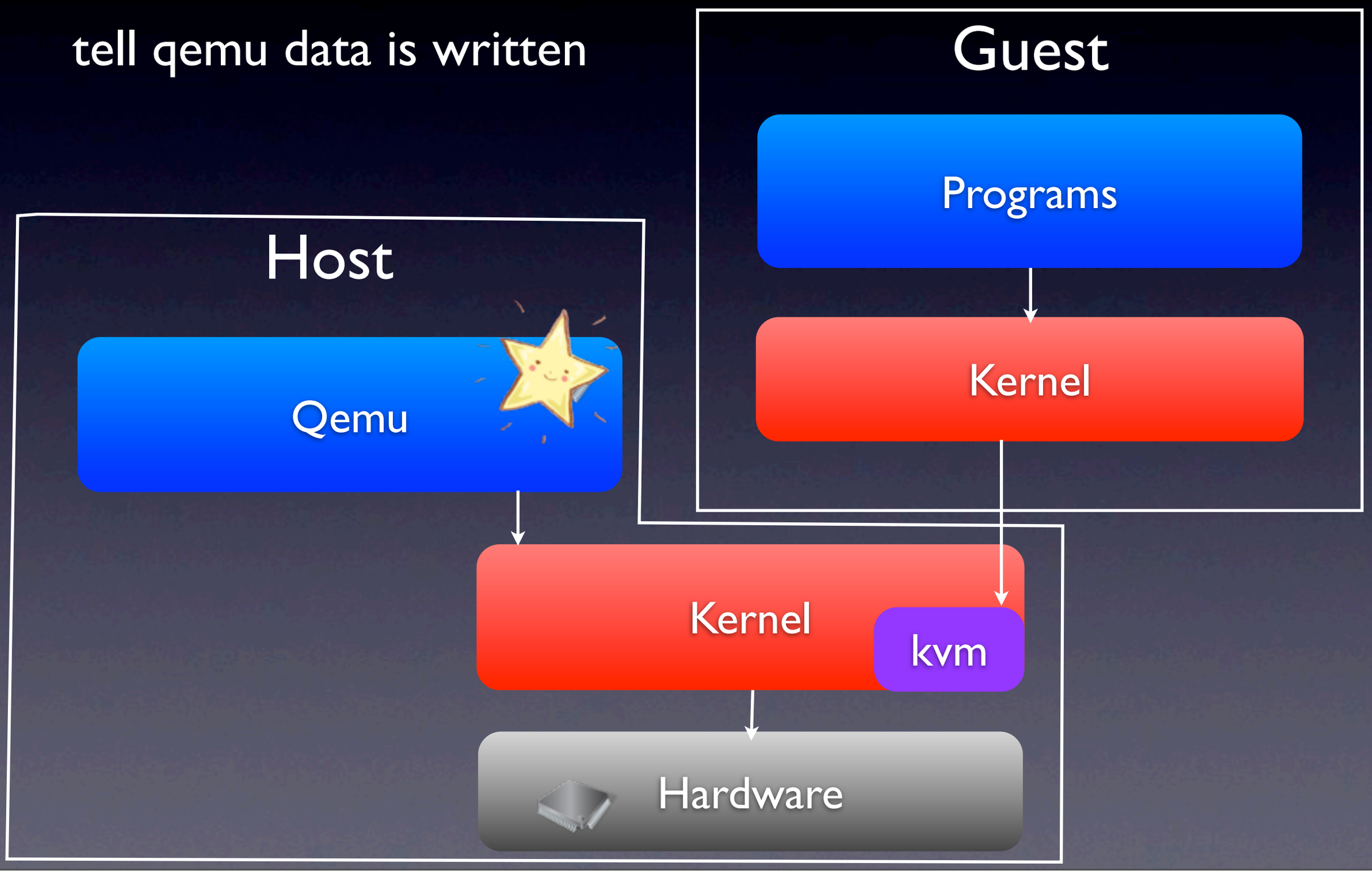
Device Access

finished writing data



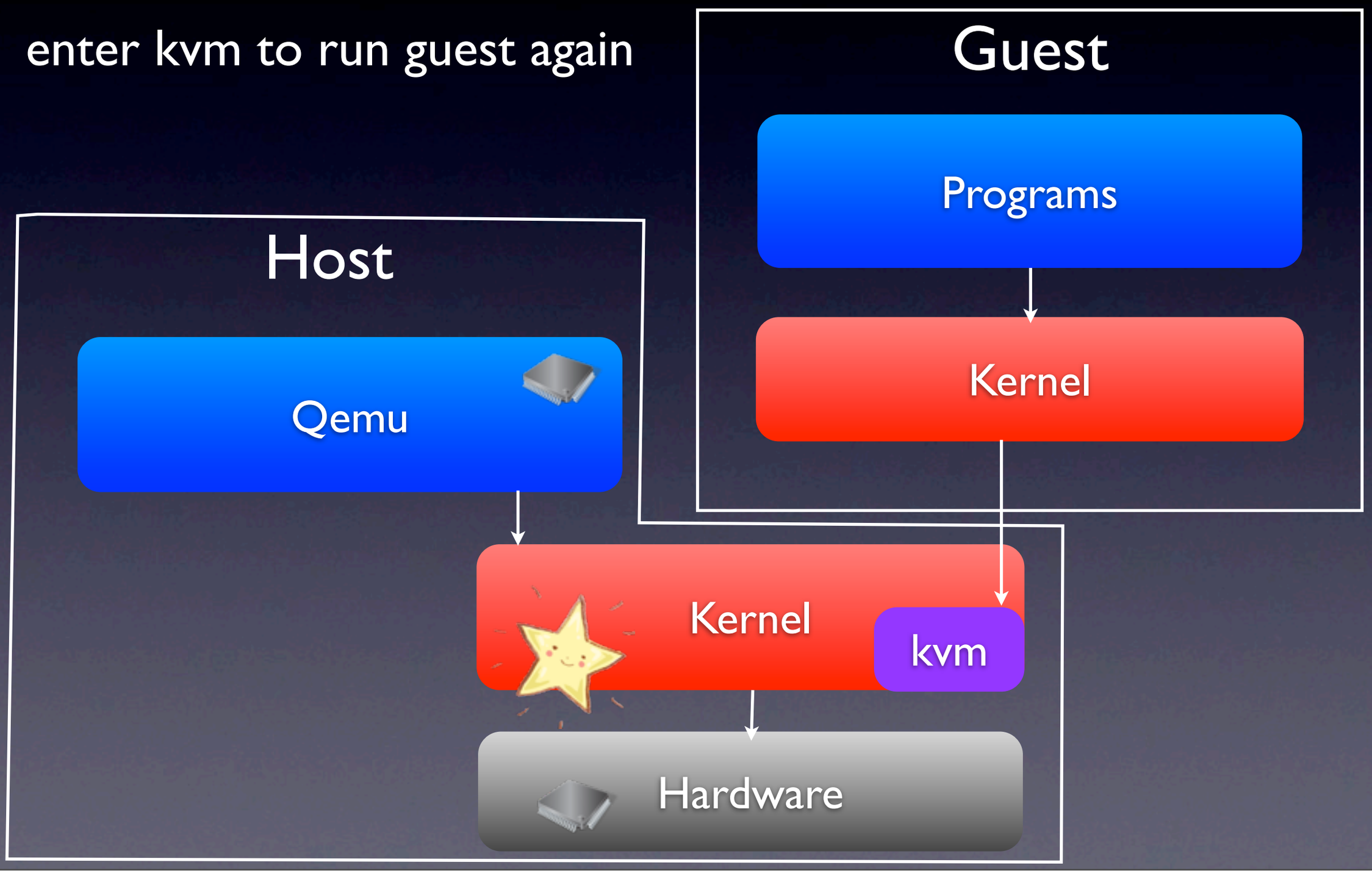
Device Access

tell qemu data is written



Device Access

enter kvm to run guest again



Device Access

execute guest

Guest

Programs

Kernel

Host

Qemu

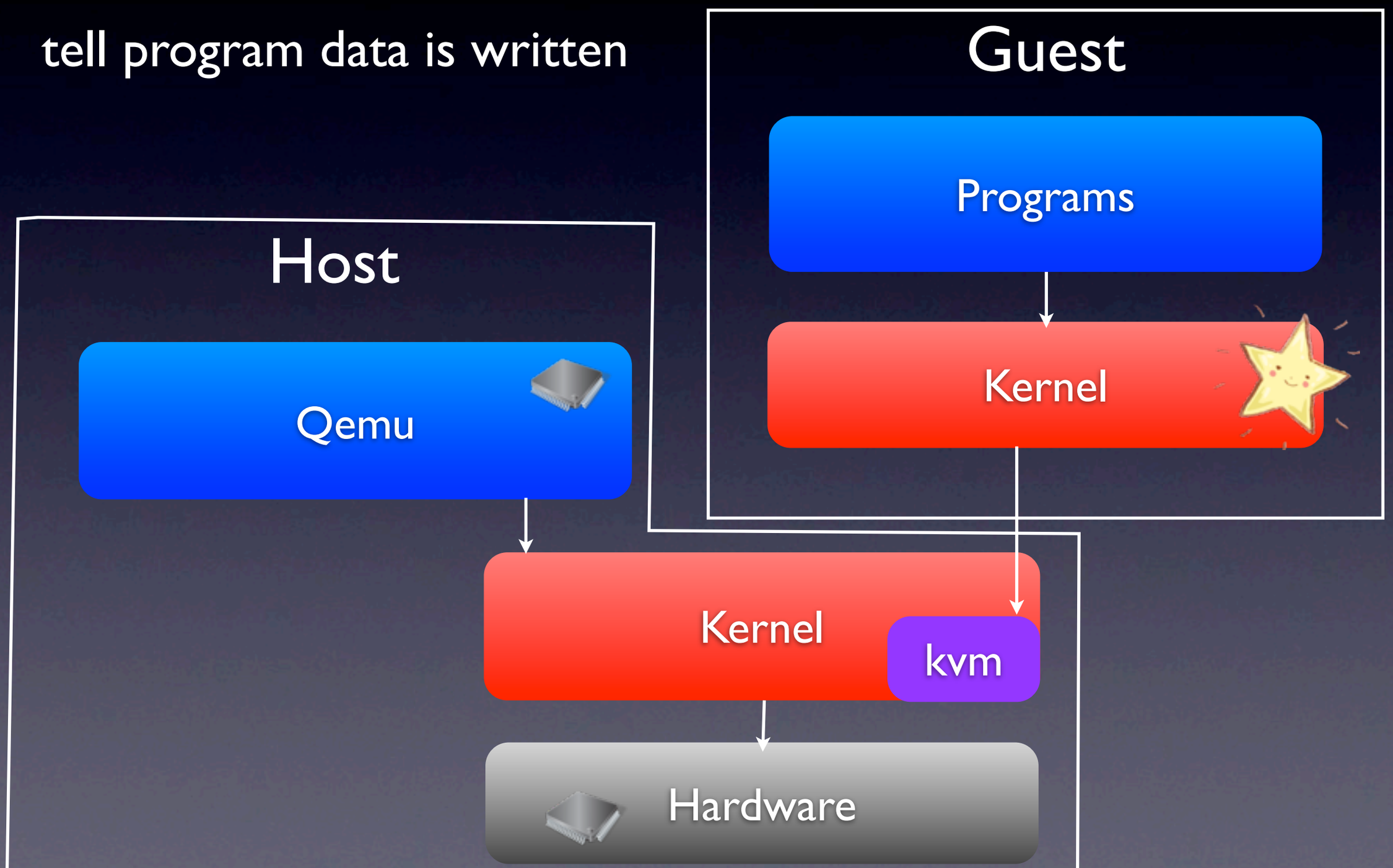
Kernel

kv

Hardware

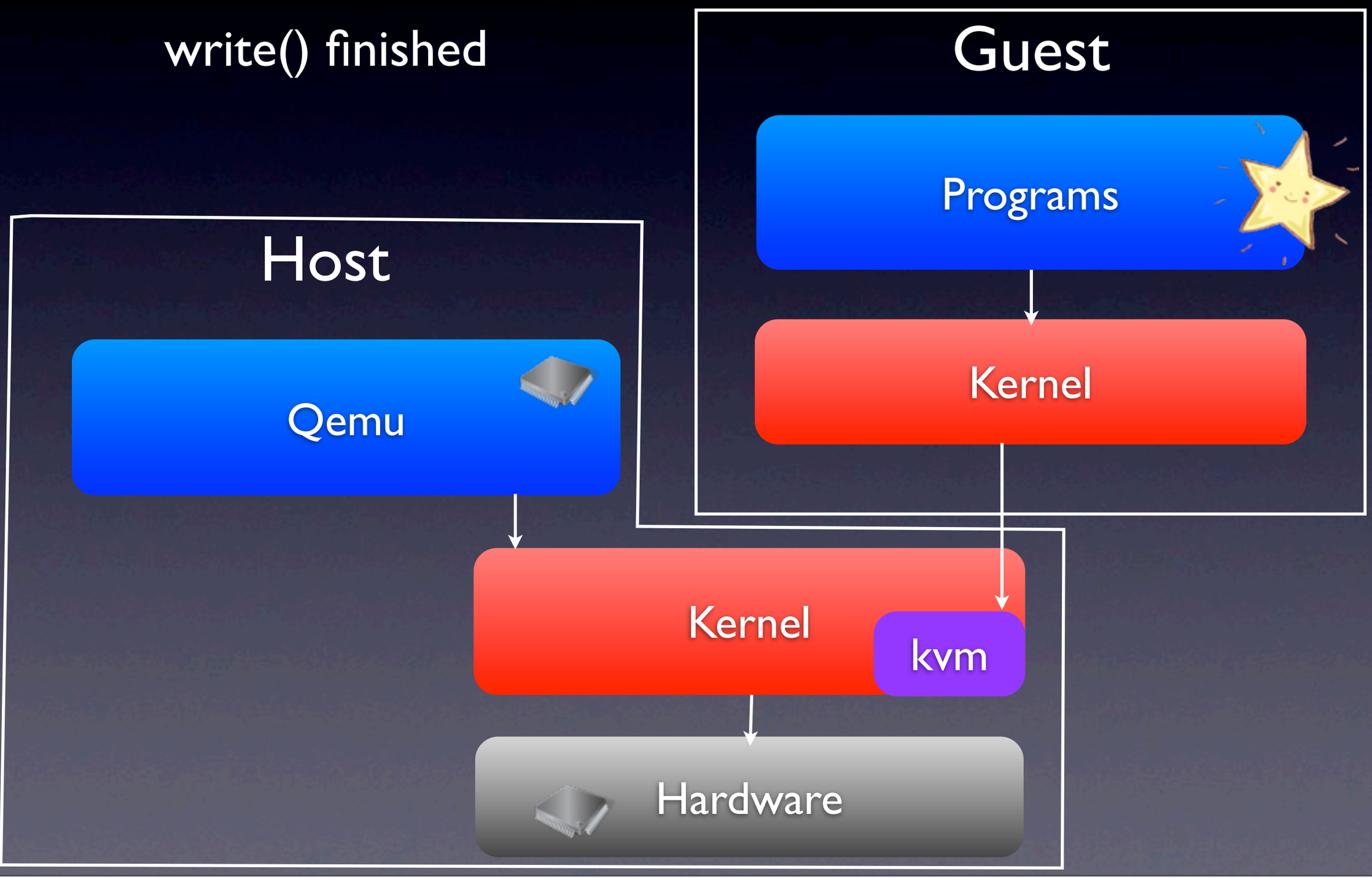
Device Access

tell program data is written

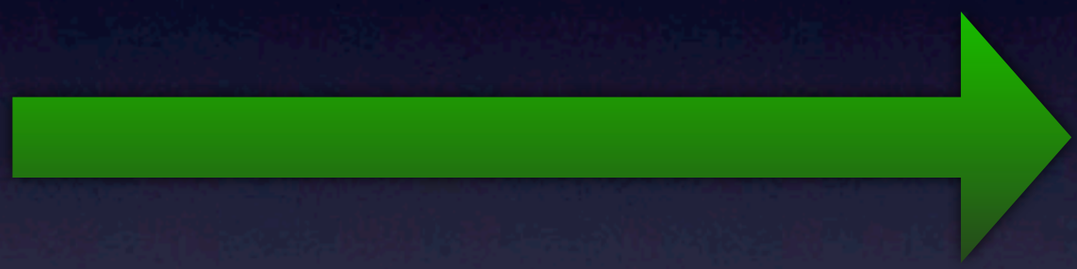


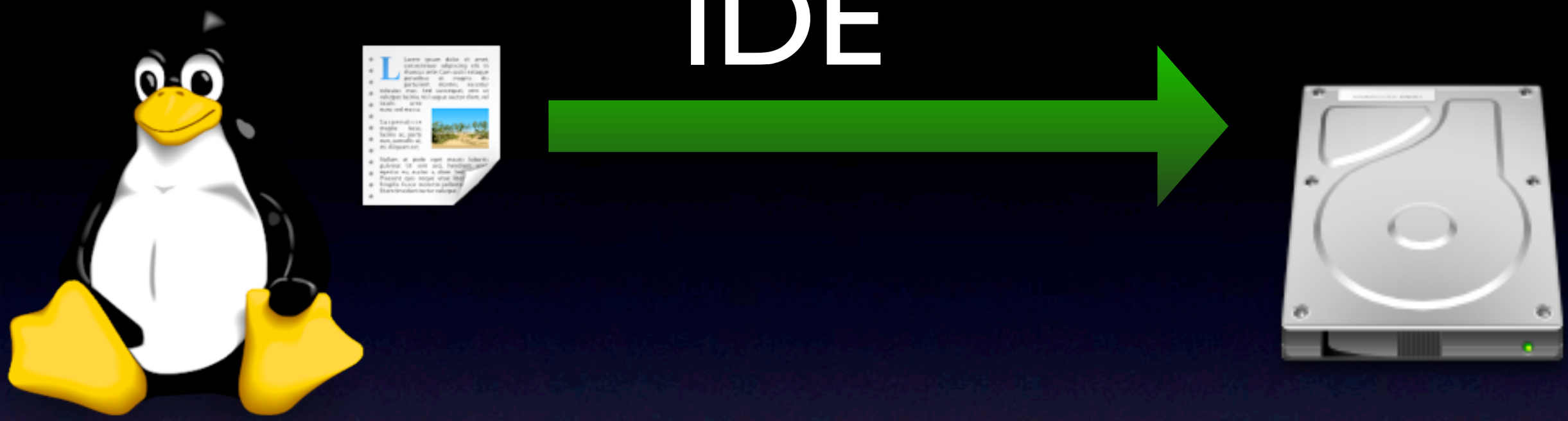
Device Access

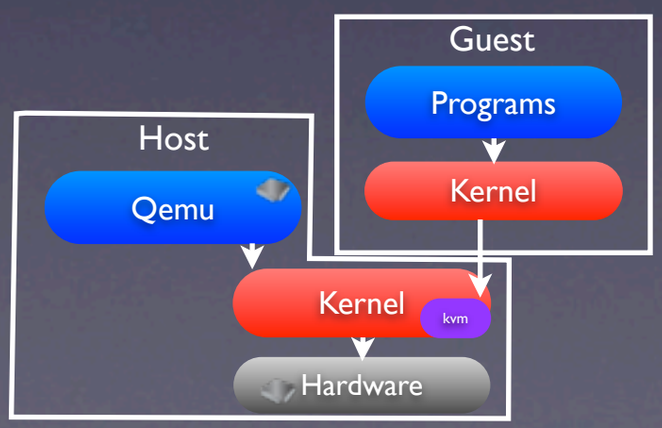
write() finished



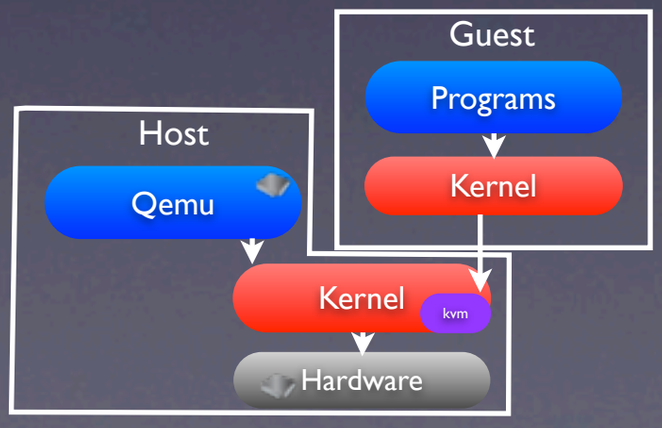
IDE




IDE


Guest
Programs
Kernel
Host
Qemu
Kernel
kvm
Hardware

IDE



IDE

set feature

set nsector

set sector

set cylinder

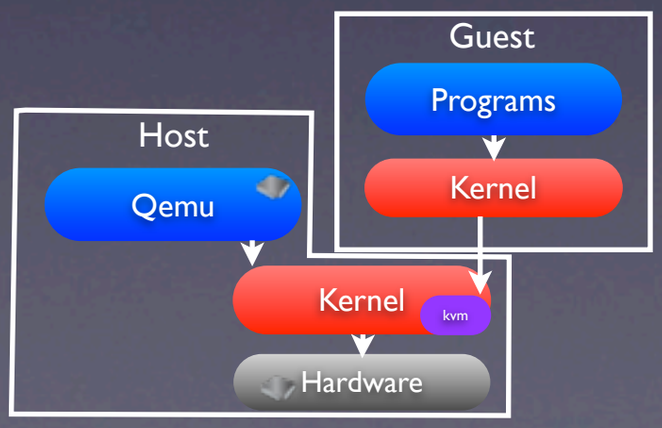
set hcylinder

set drive

issue command = DMA WRITE

set DMA SG list address

set DMA command = START



AHCI

AHCI

START PROCESSING fis n

virtio-blk

virtio-blk

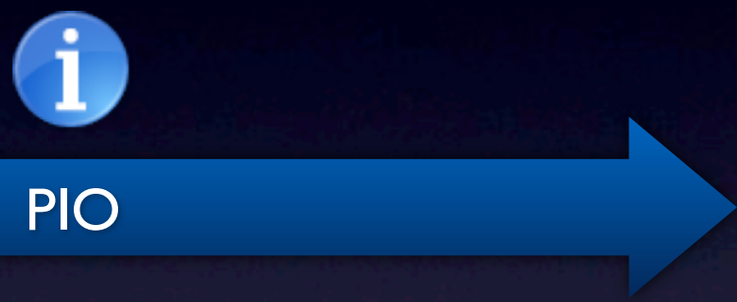
kick

Communication

PIO

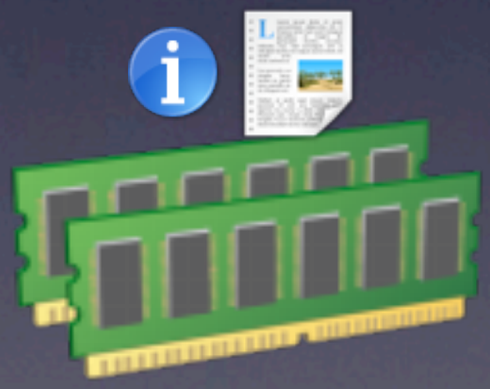
old IDE

Communication



PIO

IDE

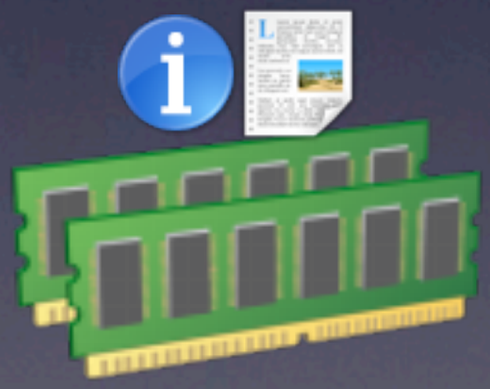


DMA

Communication

MMIO

AHCI

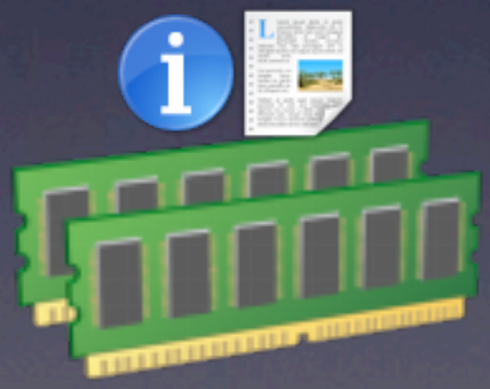


DMA

Communication

MMIO

virtio-blk



Multiple Requests

write

done

write

done

write

done

write

done

IDE

Multiple Requests

write

write

write

write

AHCI
(NCQ)

done

done

done

done

Multiple Requests

write

write

write

write

done

done

done

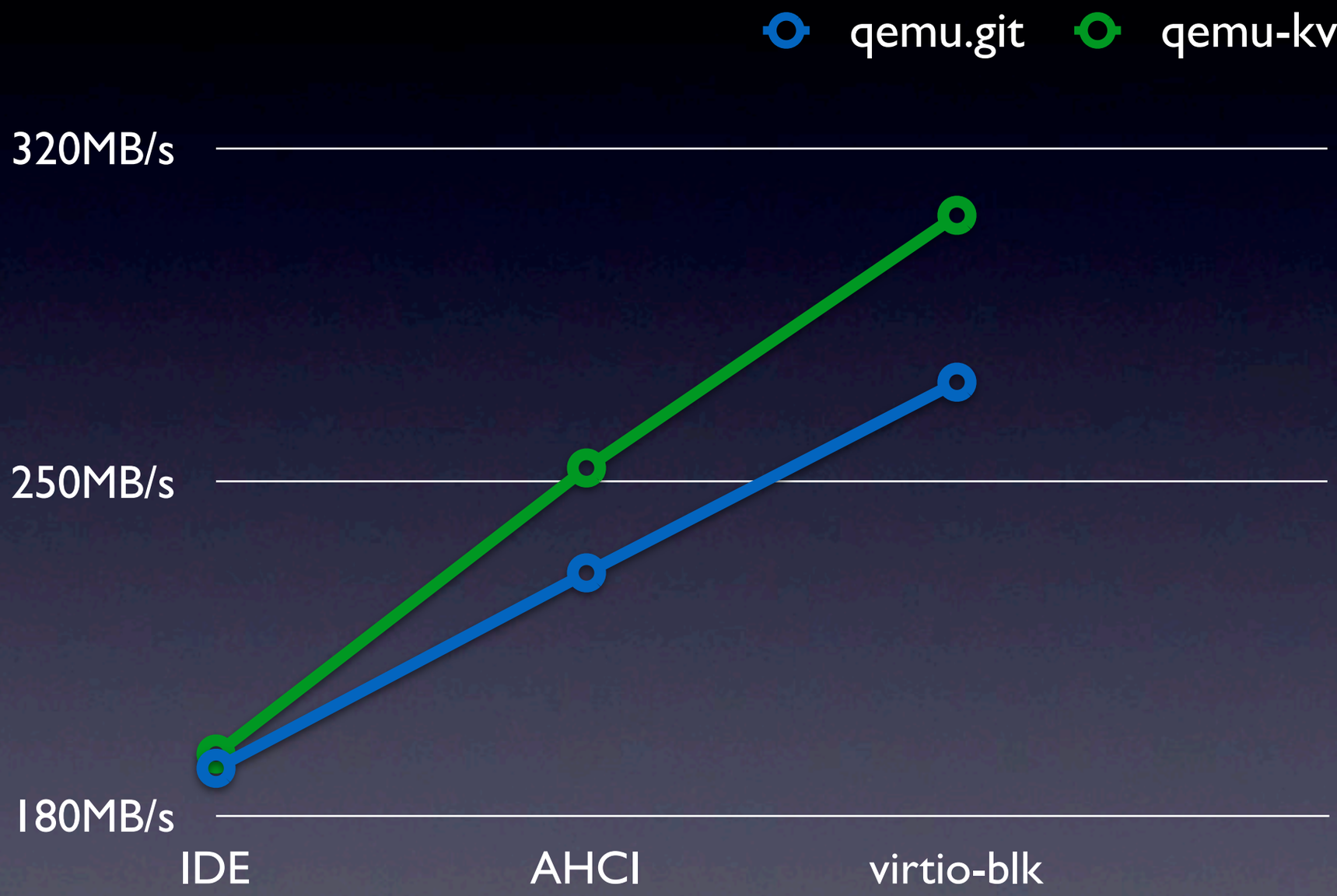
done

virtio-blk

Guest OS Support

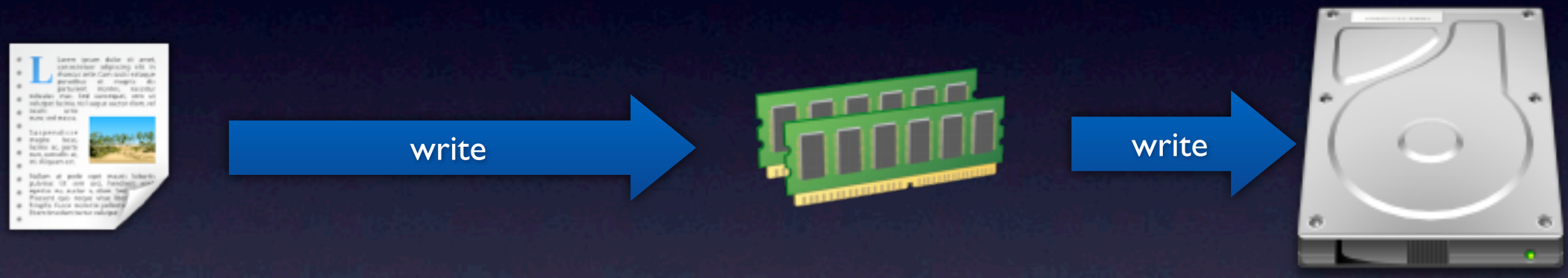
	IDE	AHCI	Virtio-blk
Linux	✔	✔	✔
Windows XP	✔	✗	external
Windows Vista	✔	✔	external
Mac OS X	✗	✔	✗
BSD	✔	✔	✗

Speed



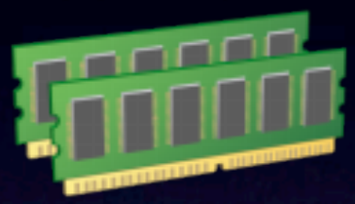
(host) qemu-img create /dev/shm/test.raw 20G
(guest) dd if=/dev/Xda of=/dev/null bs=1M count=1000 iflag=direct
Benchmark ran on Intel Atom z530

Caching

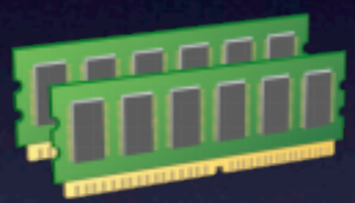


Caching

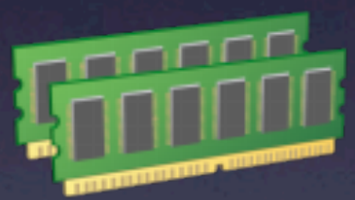
write



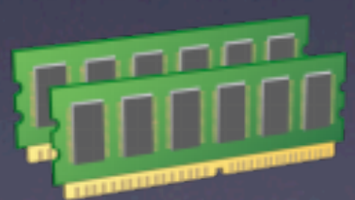
write



write

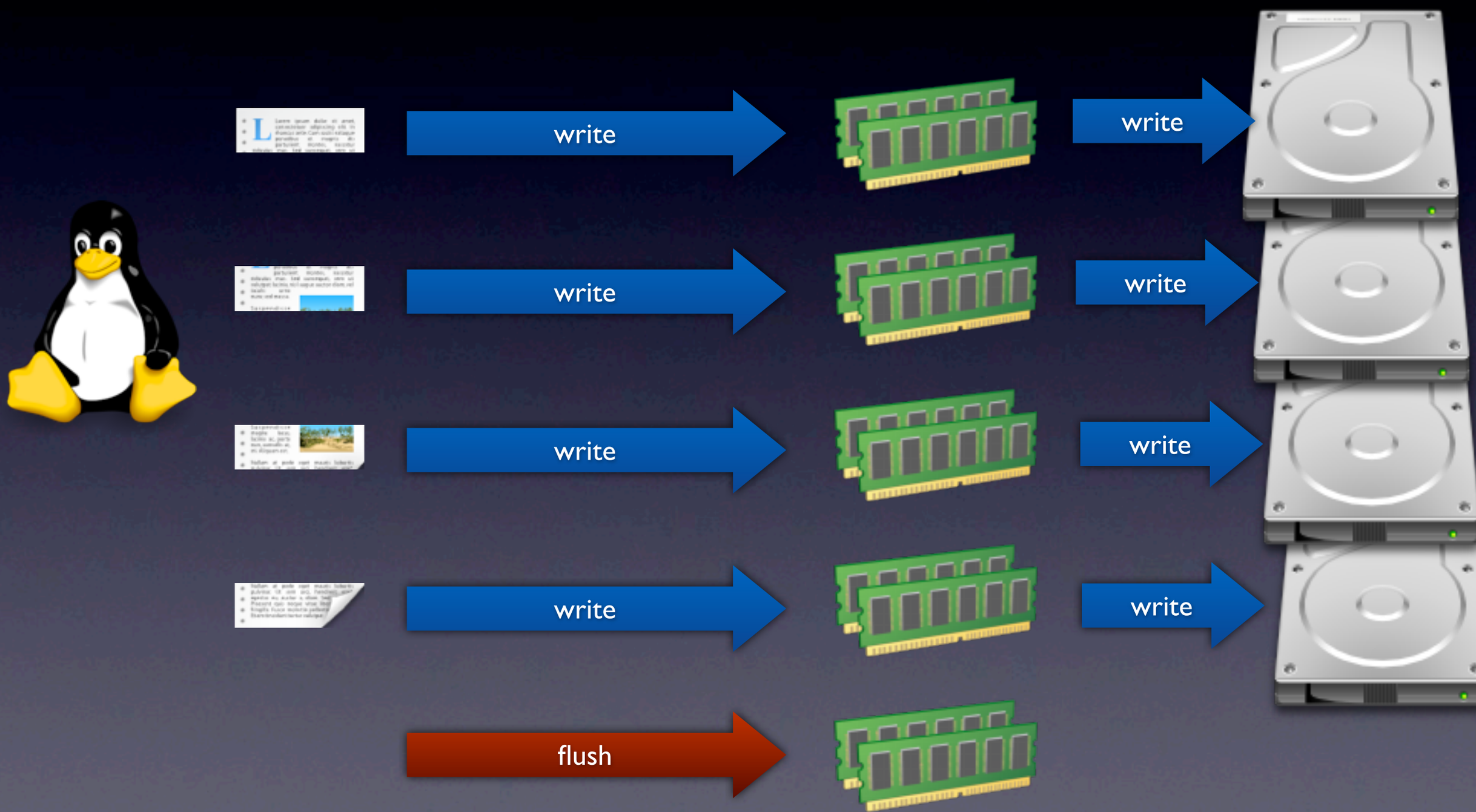


write



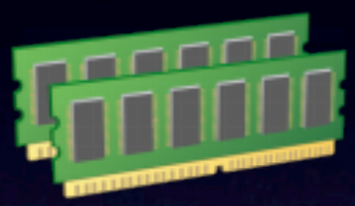
write

cache=writethrough

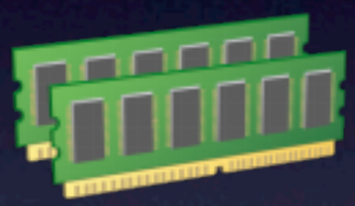


cache=writeback

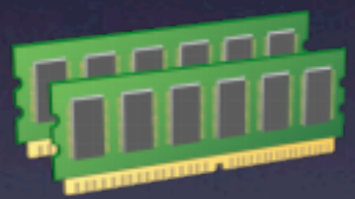
write



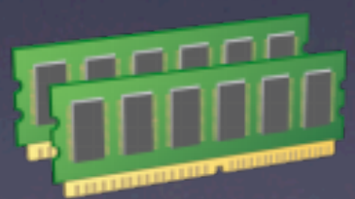
write



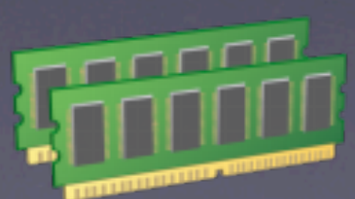
write



write



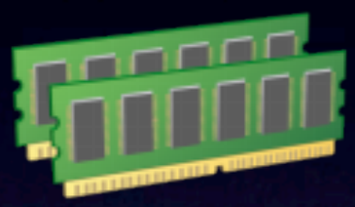
flush



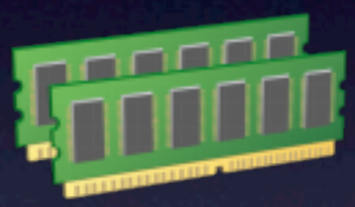
write

cache=unsafe

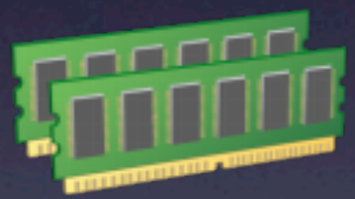
write



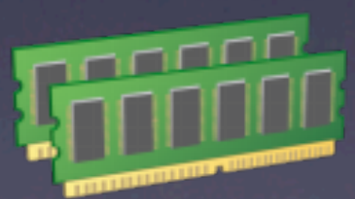
write



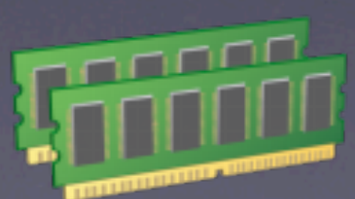
write



write



flush



cache=none

write

write

write

flush

Why AHCI

- Faster than IDE
- Compatible with most OSs
- Compatible with all modern devices (CD-ROMs)
- Only need to develop one side

Future ideas

- Default adapter in -M q35
- MSI-X

Questions?

蟻が十