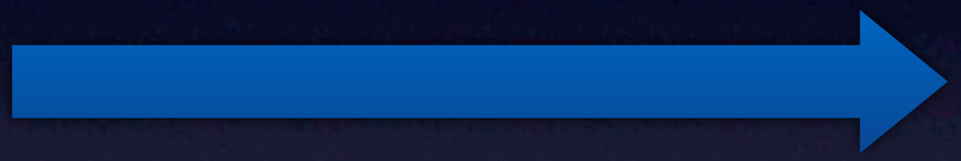
Partial Device Port Acceleration

About Me

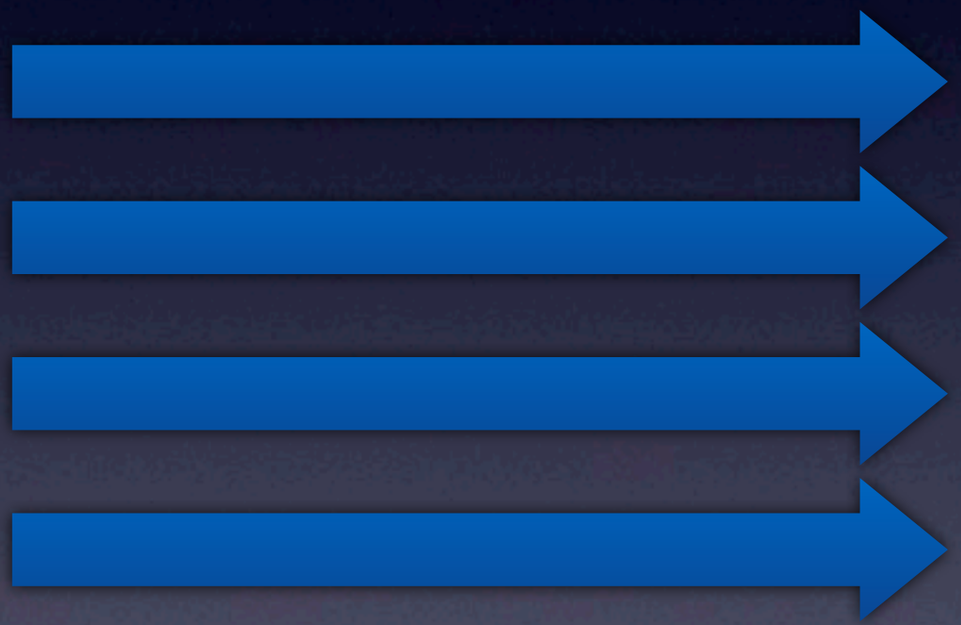
- Alexander Graf
- KVM and Qemu developer
  - Server class PowerPC KVM port
  - S390x Qemu guest support
  - x86 Mac OS X in KVM
  - Nested SVM
  - Xenner
  - ...

Disk performance

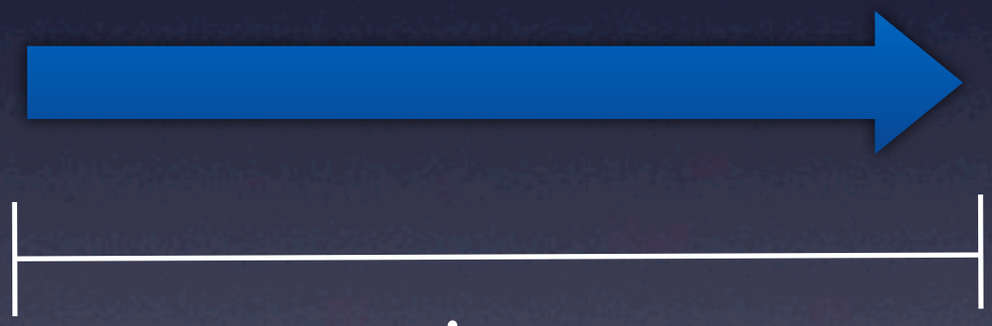
Disk performance



Disk performance



Disk performance



Disk performance

- Overhead split into 2 categories
  - Cost of parallel transfers
  - Cost of single transfers
- Looking at single transfers today

Disk performance

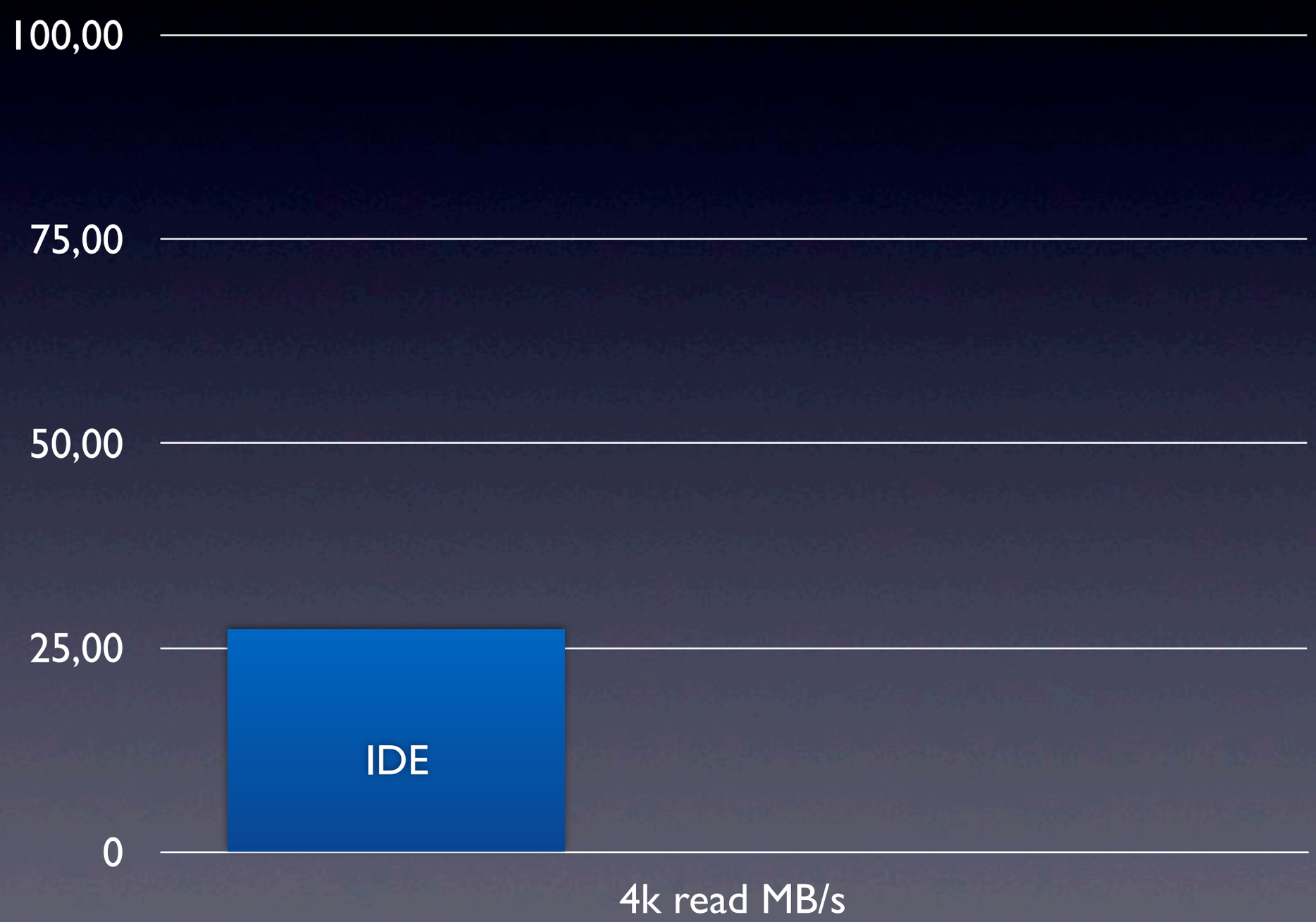
- No prefetch
- No caching
- One request at a time

Disk performance

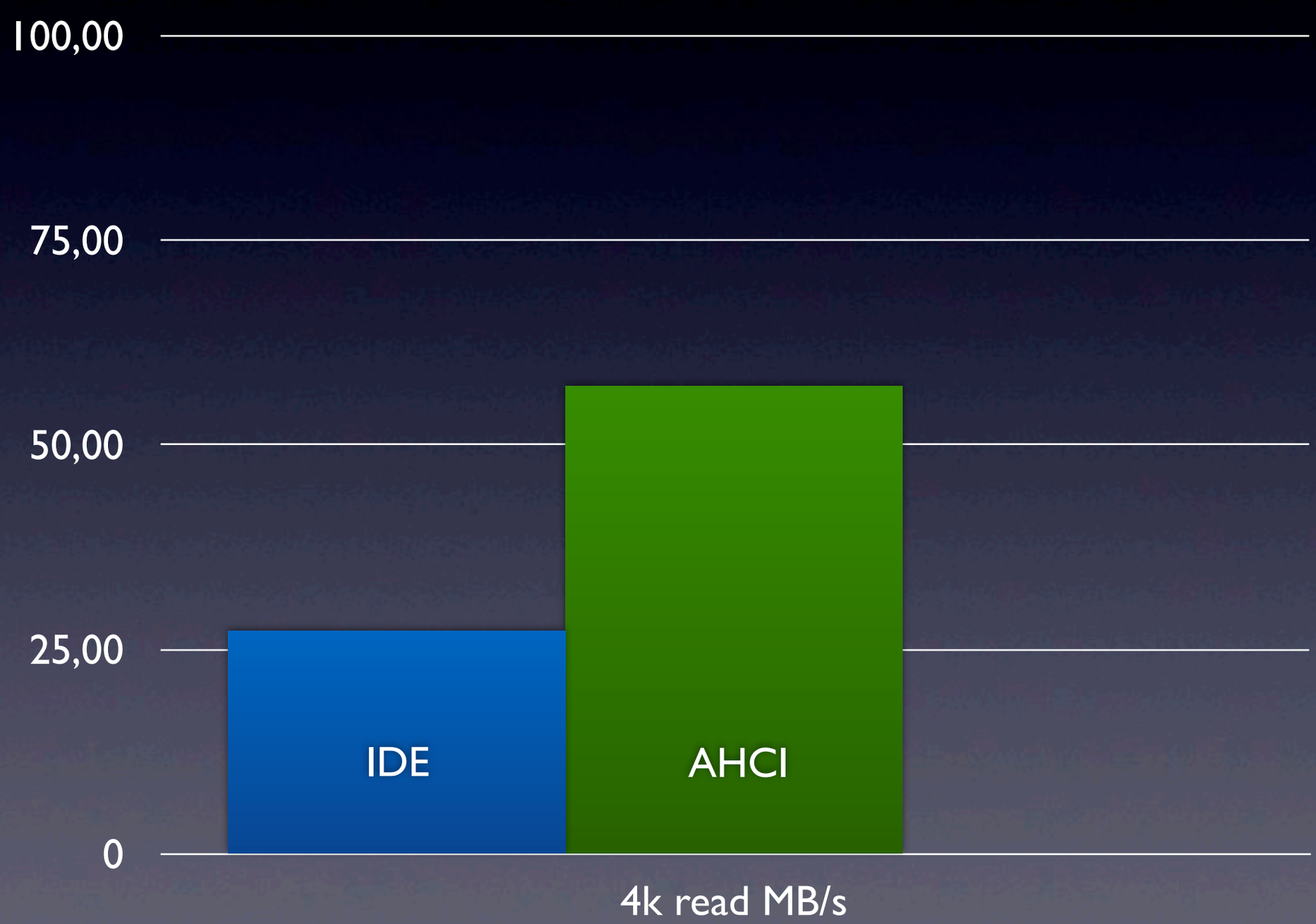
- No prefetch
- No caching
- One request at a time

```
$ dd if=/dev/xxx of=/dev/null bs=4k iflag=direct
```

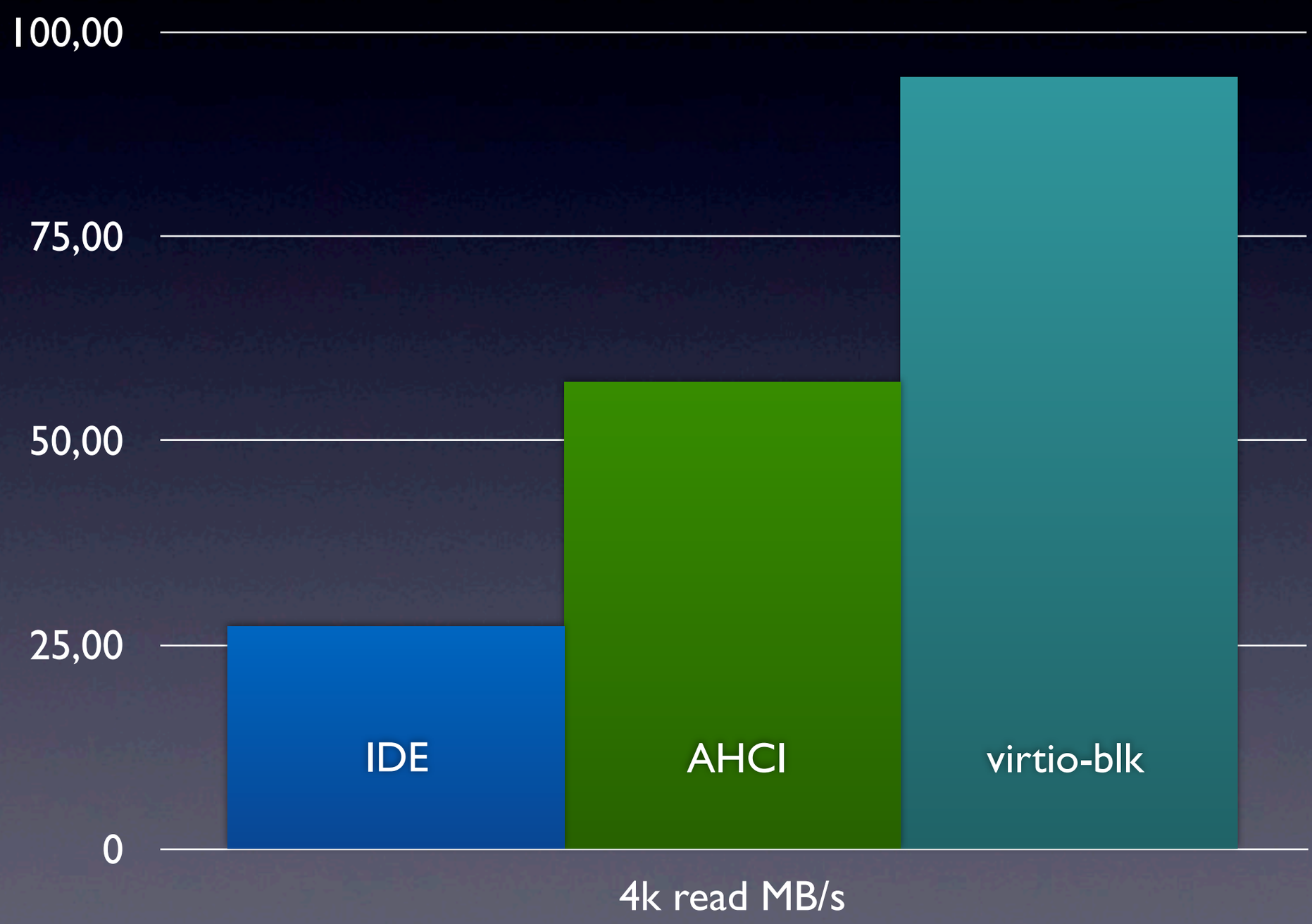
Disk performance



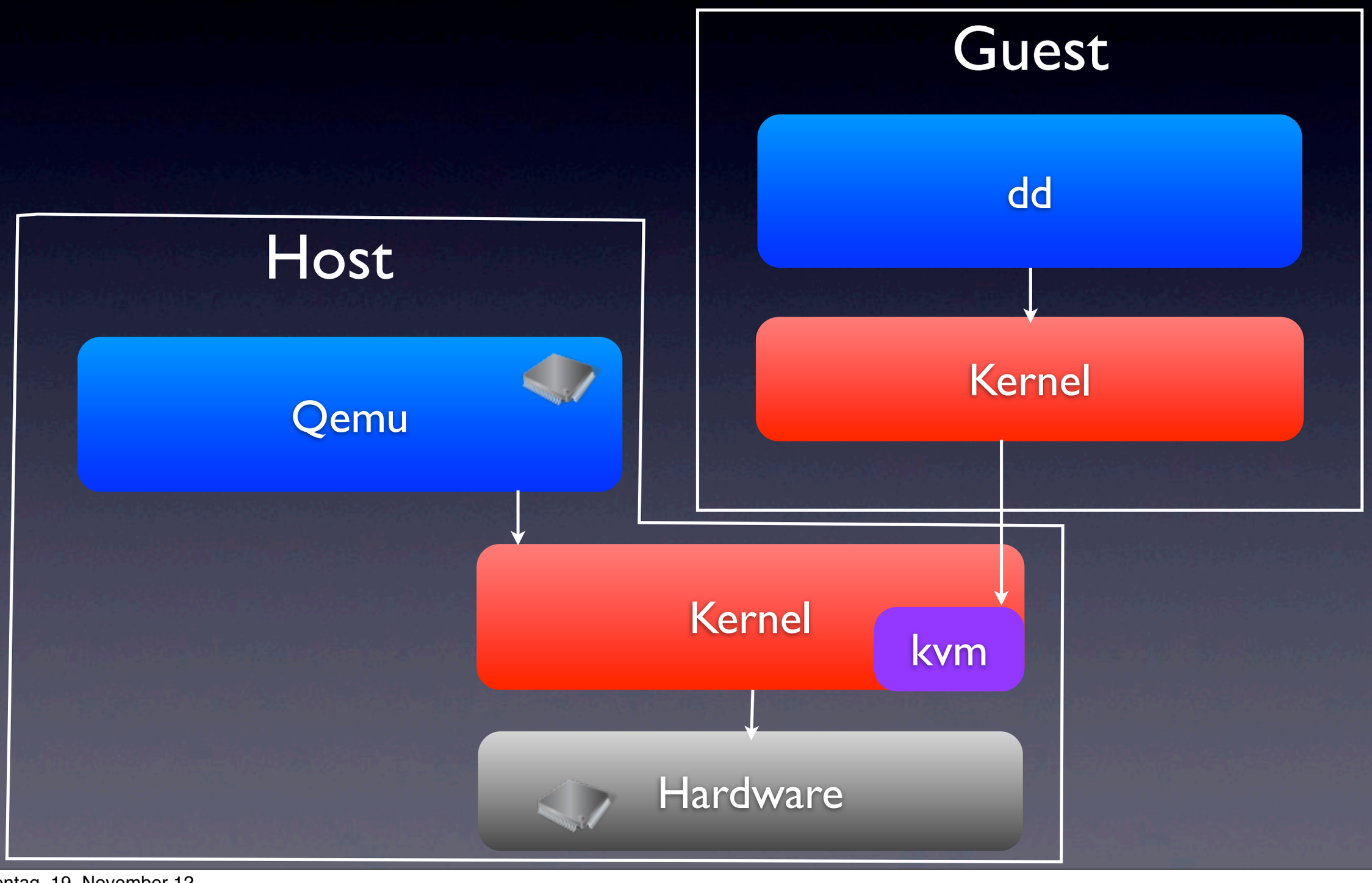
Disk performance



Disk performance

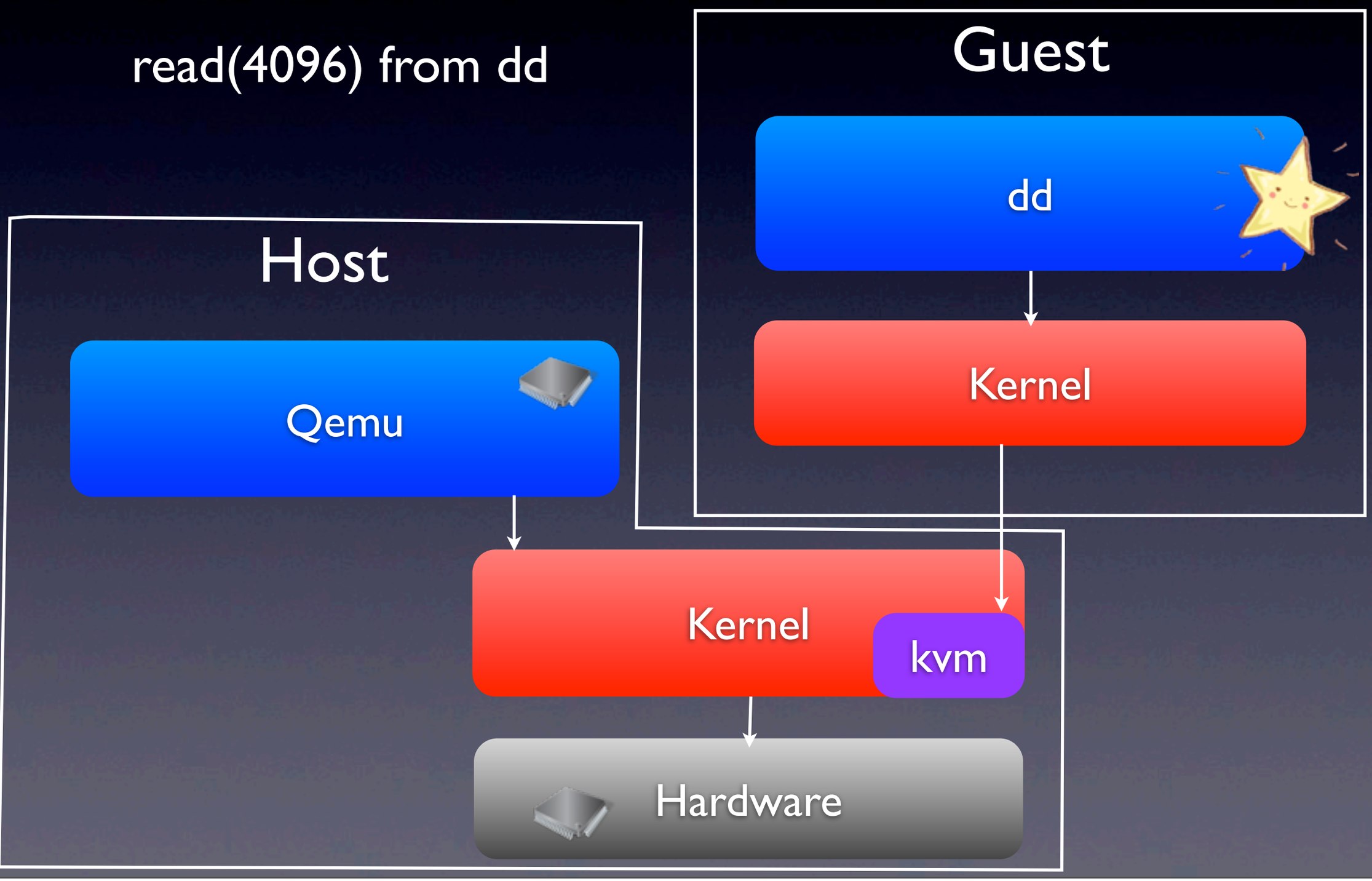


Device Access



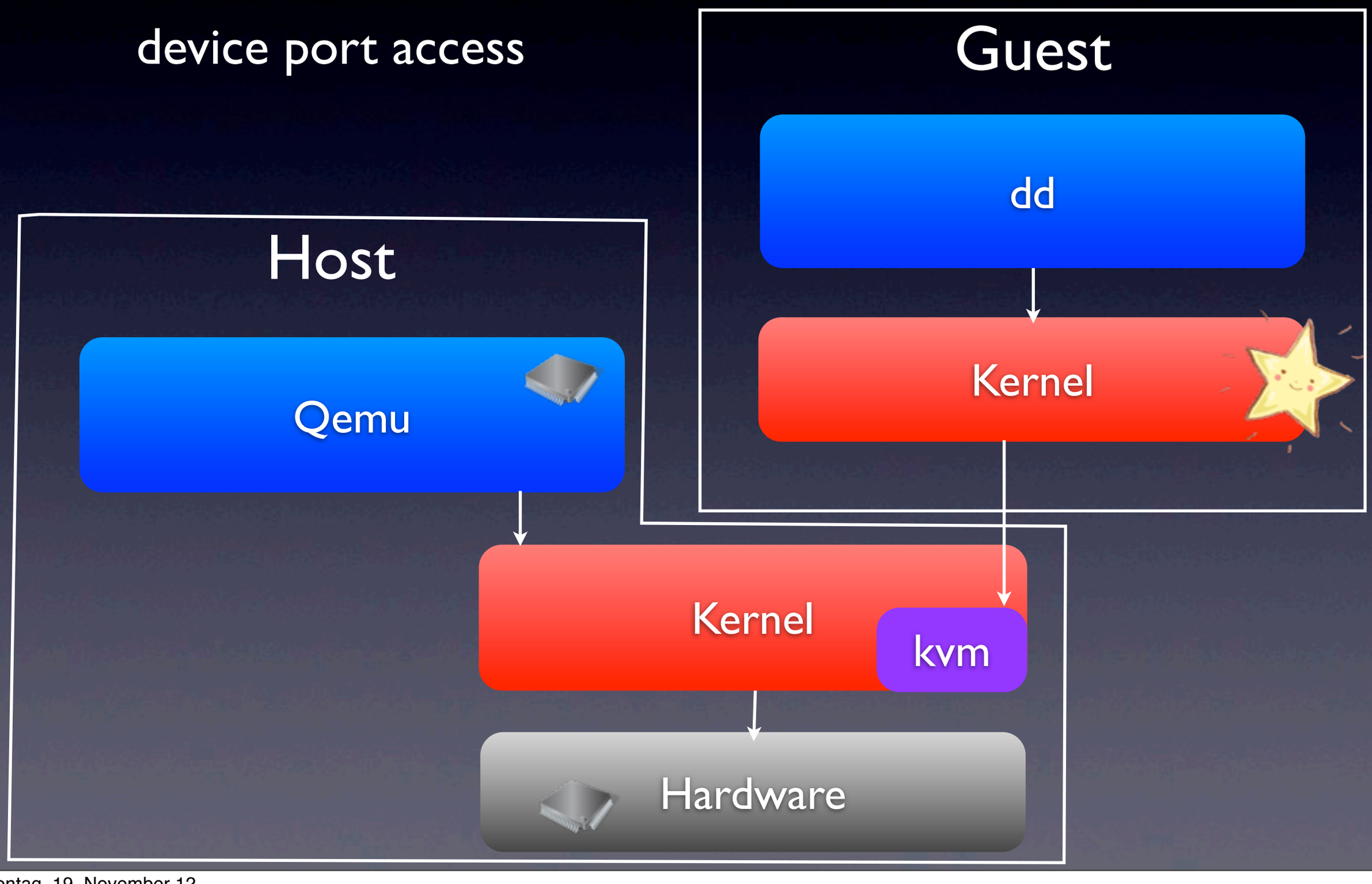
Device Access

read(4096) from dd



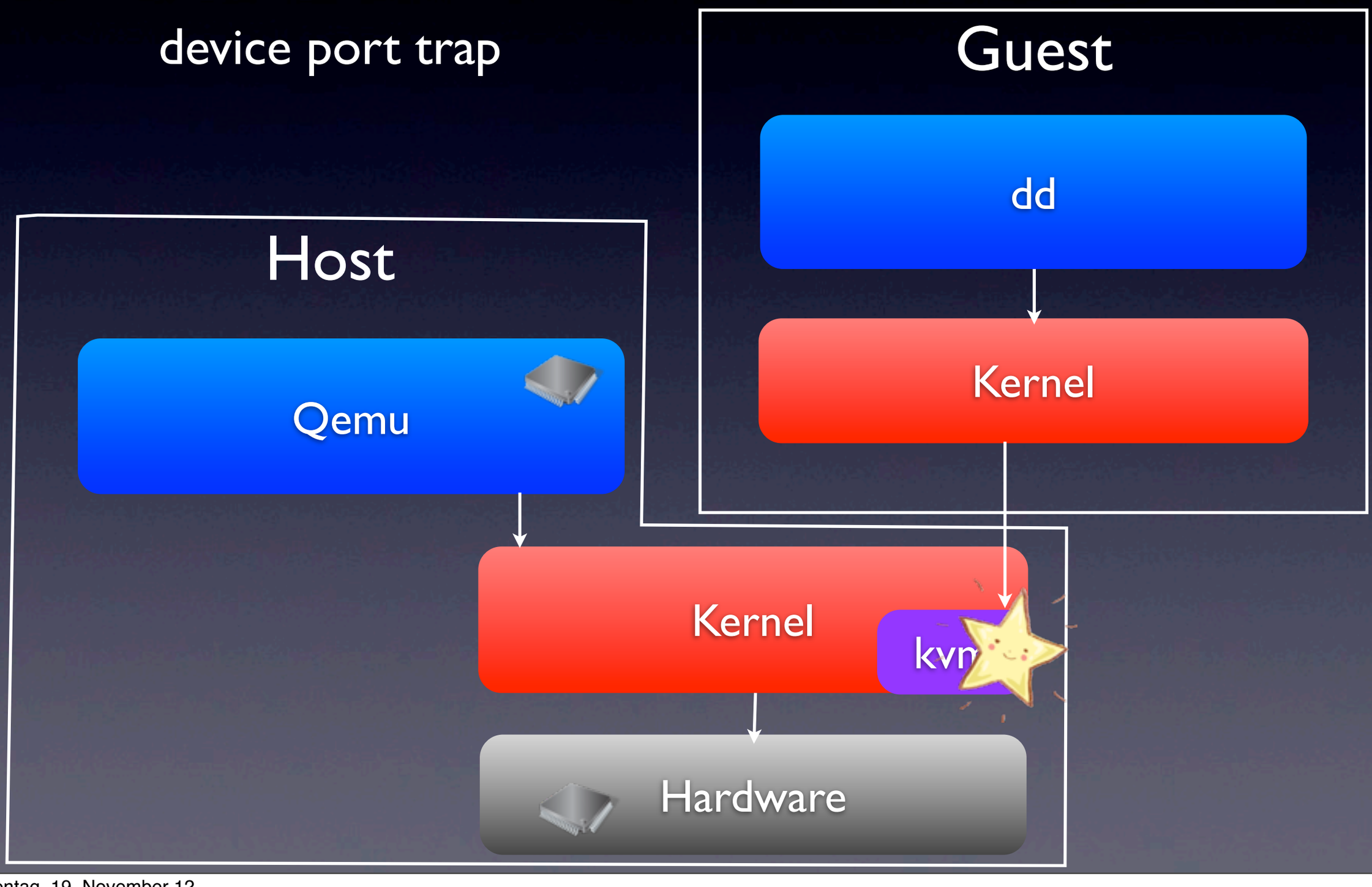
Device Access

device port access



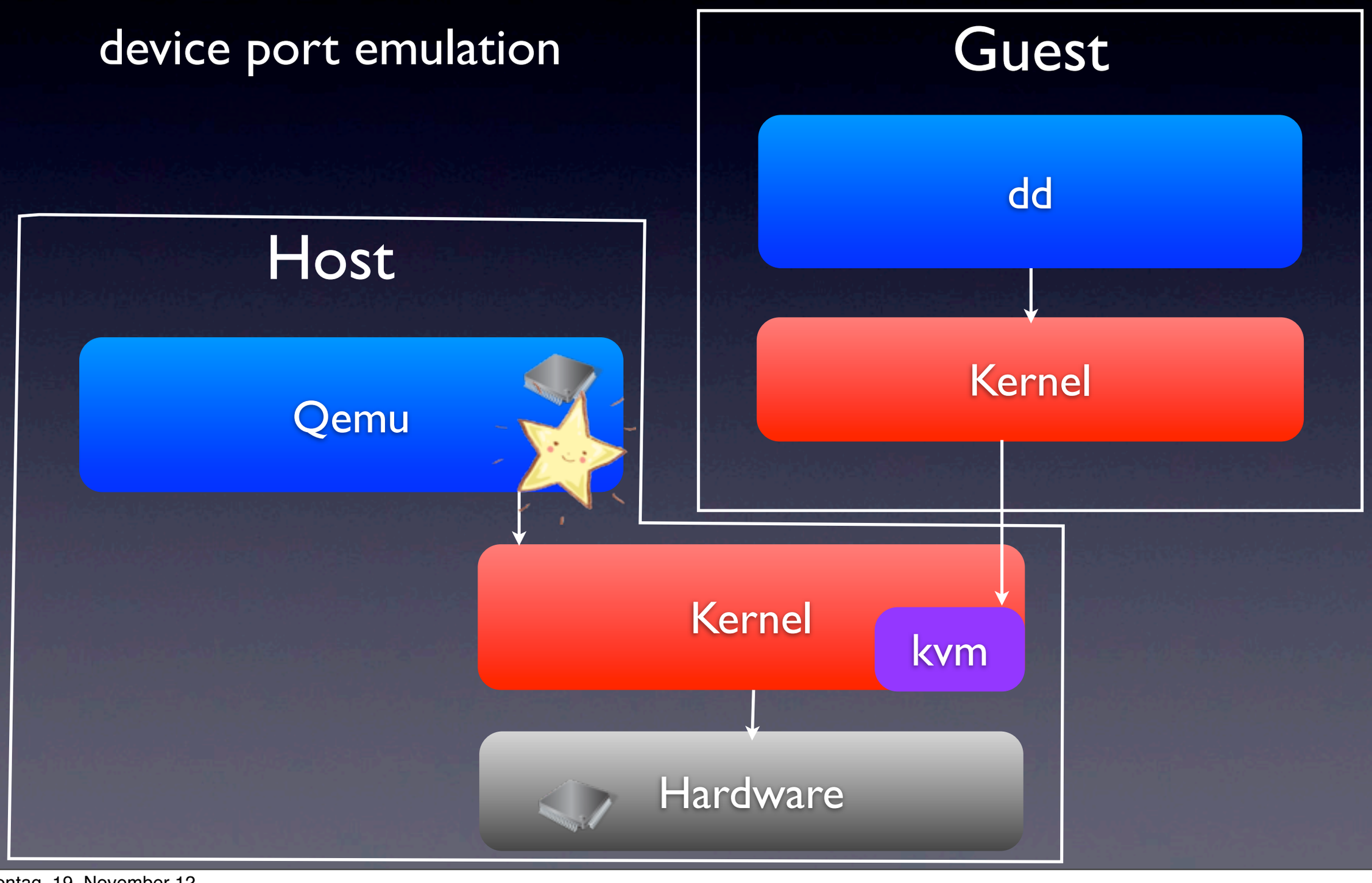
Device Access

device port trap

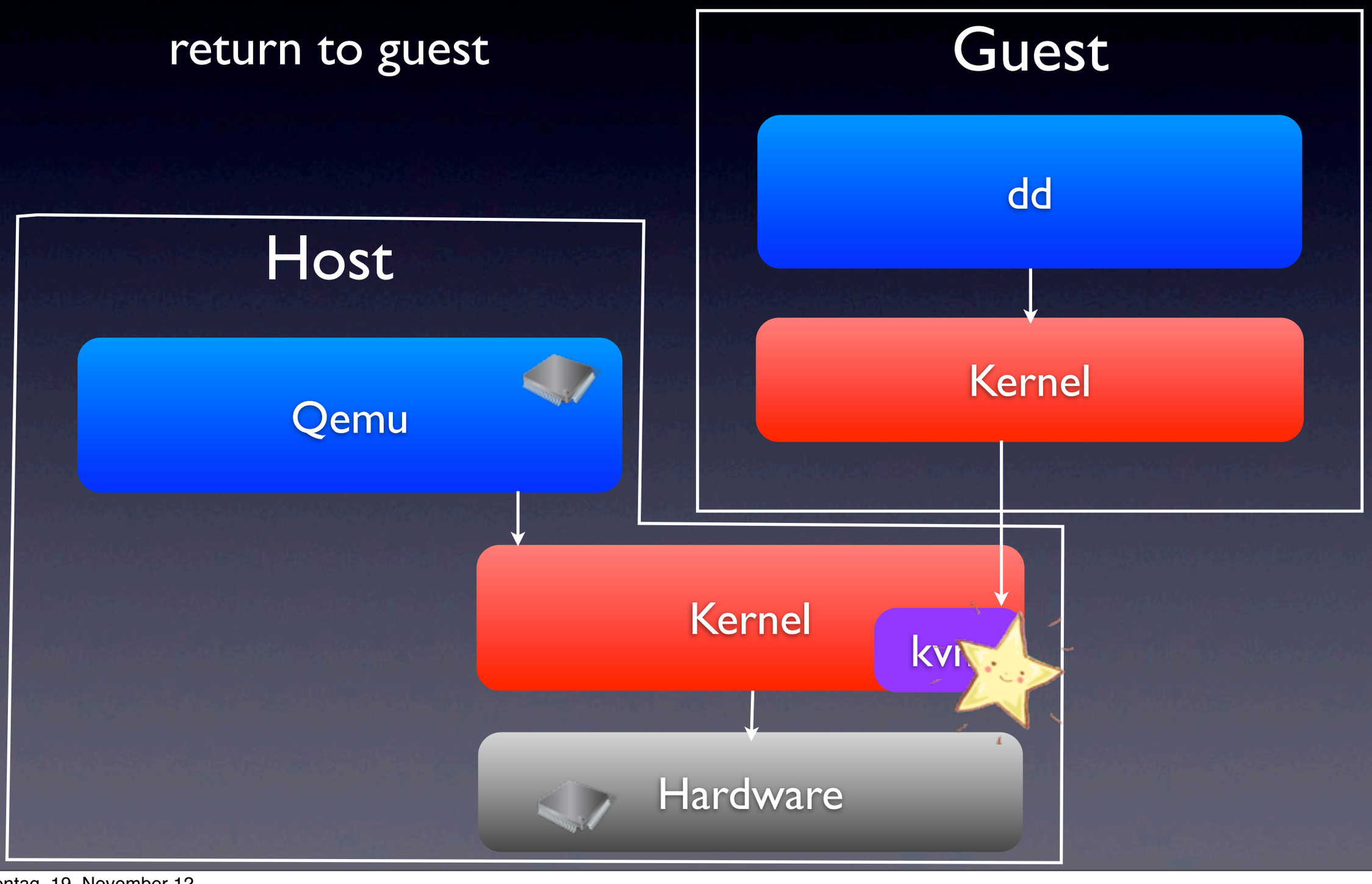


Device Access

device port emulation

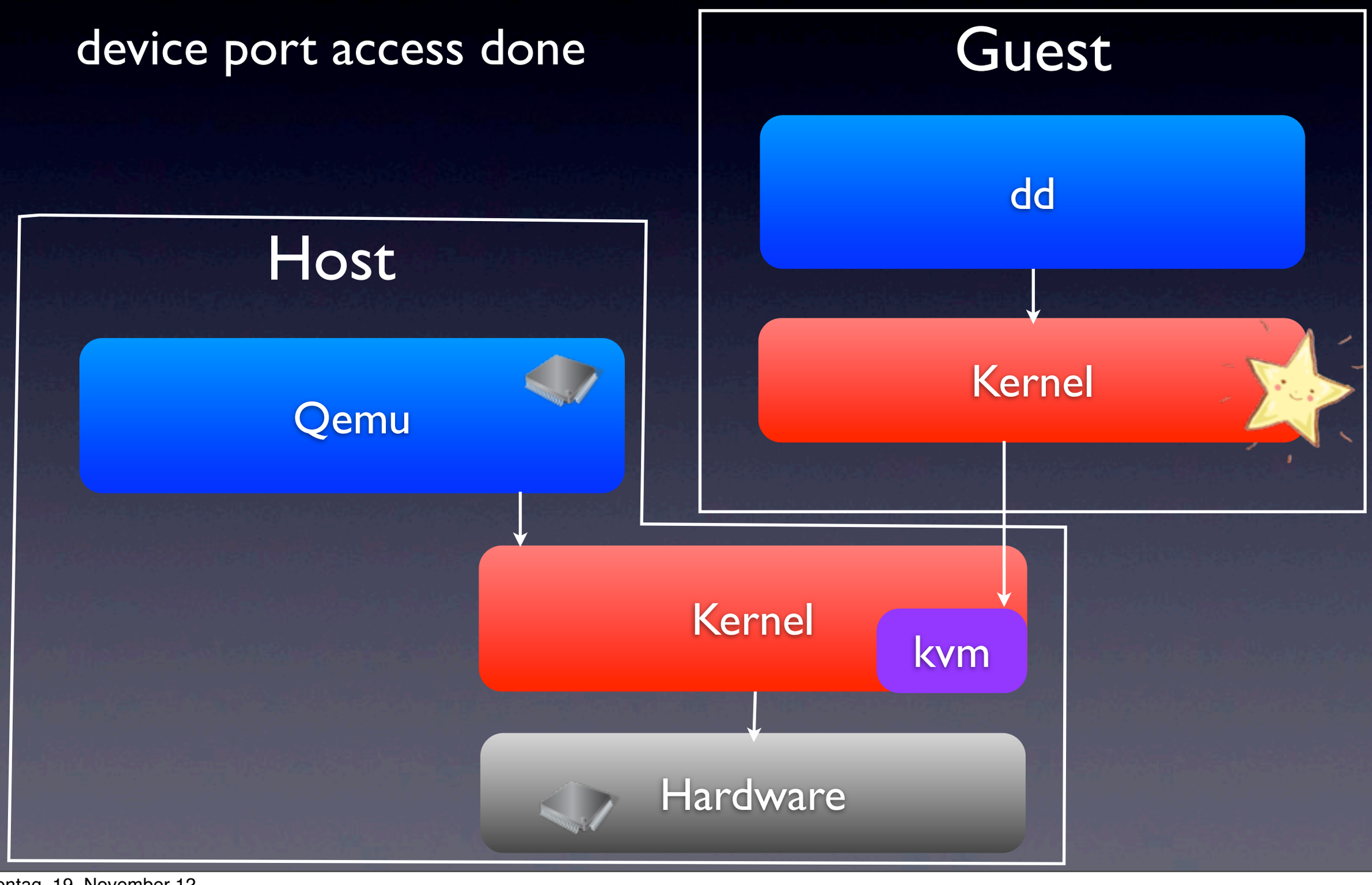


Device Access



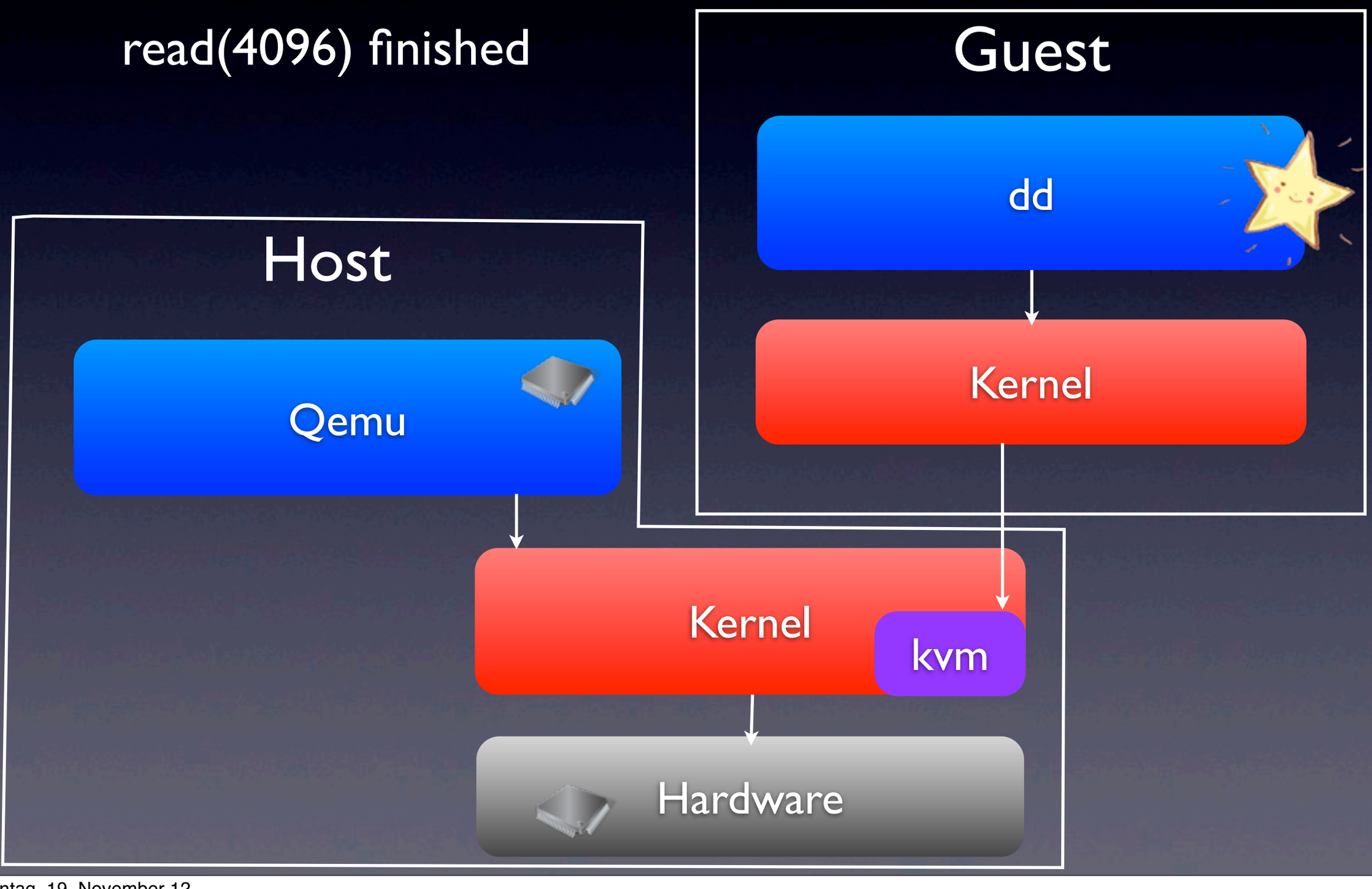
Device Access

device port access done

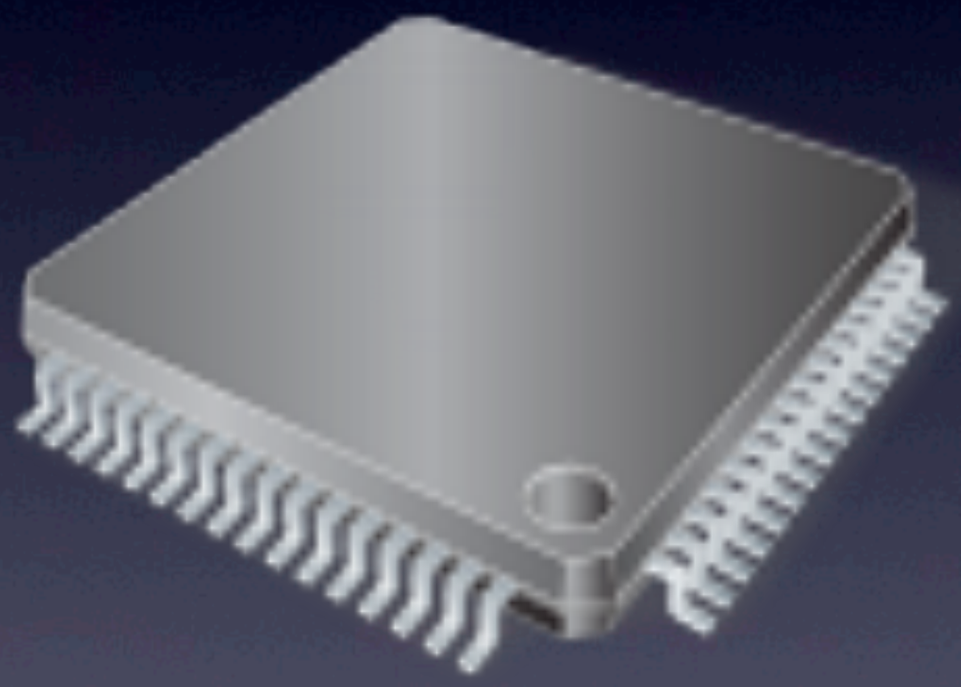


Device Access

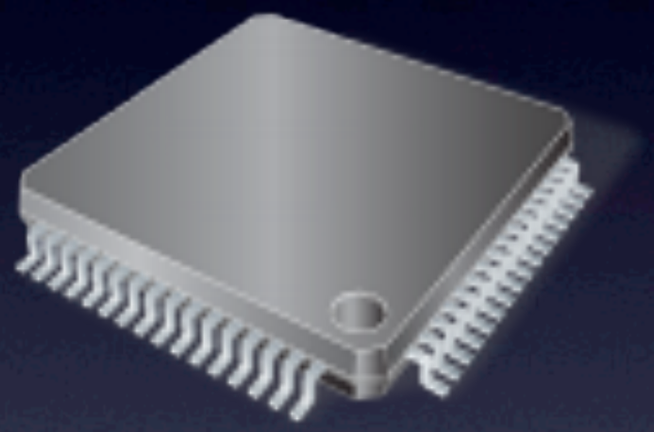
read(4096) finished



IDE



IDE

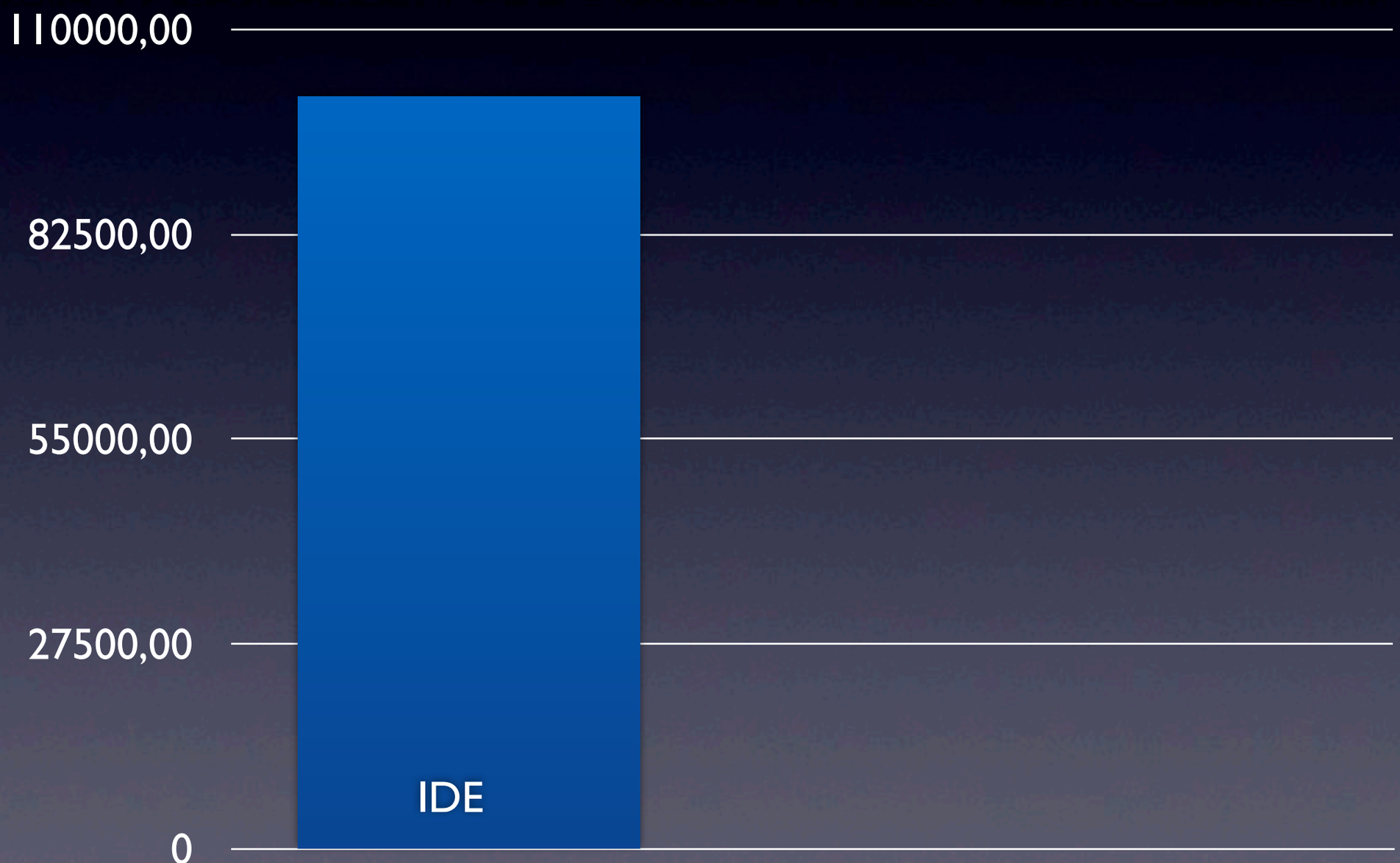


LBA

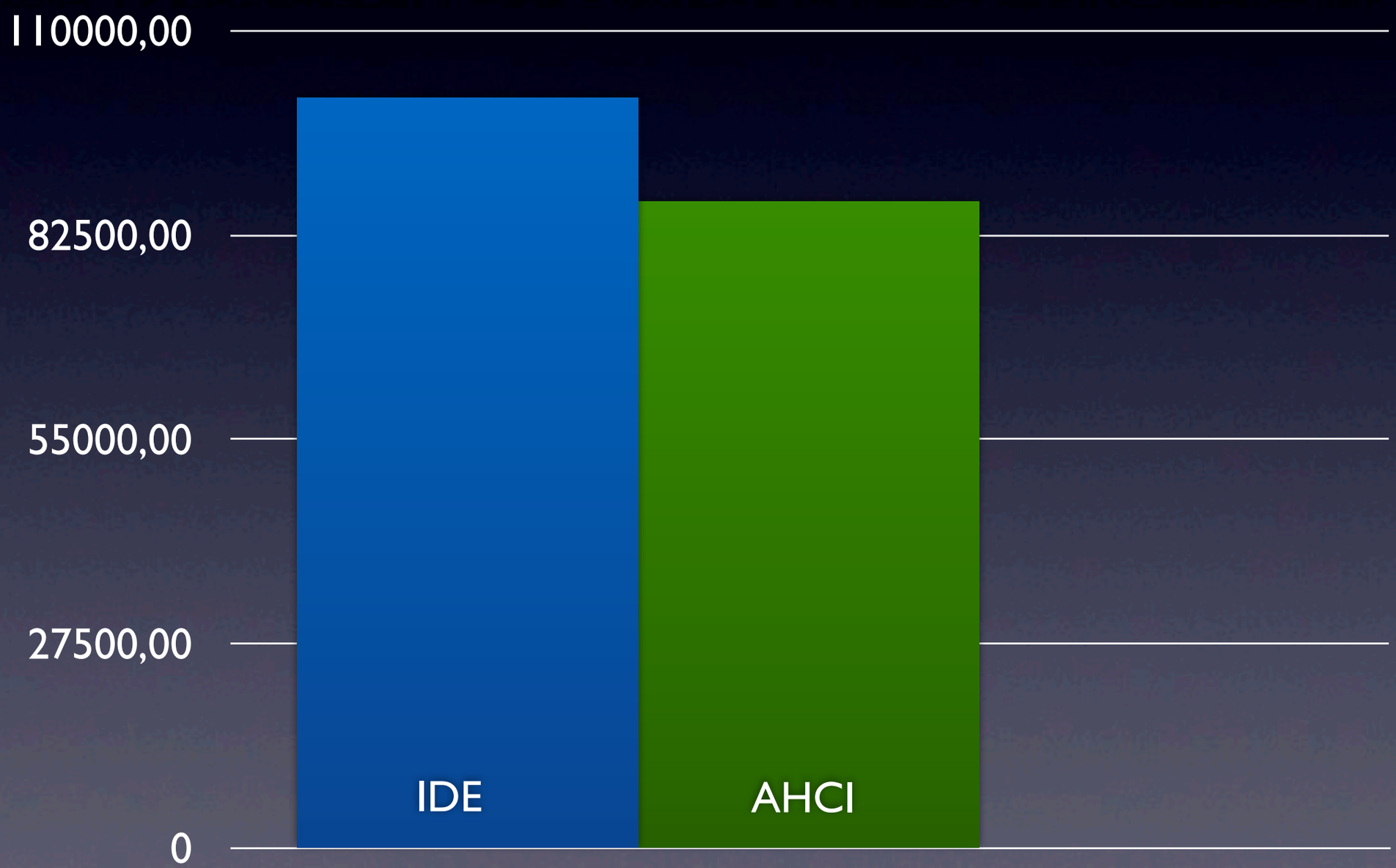
feature | nsector | sector | lcyl | hcyl | select | cmd

BMDMA cmd | BMDMA status | B B B BMaddr

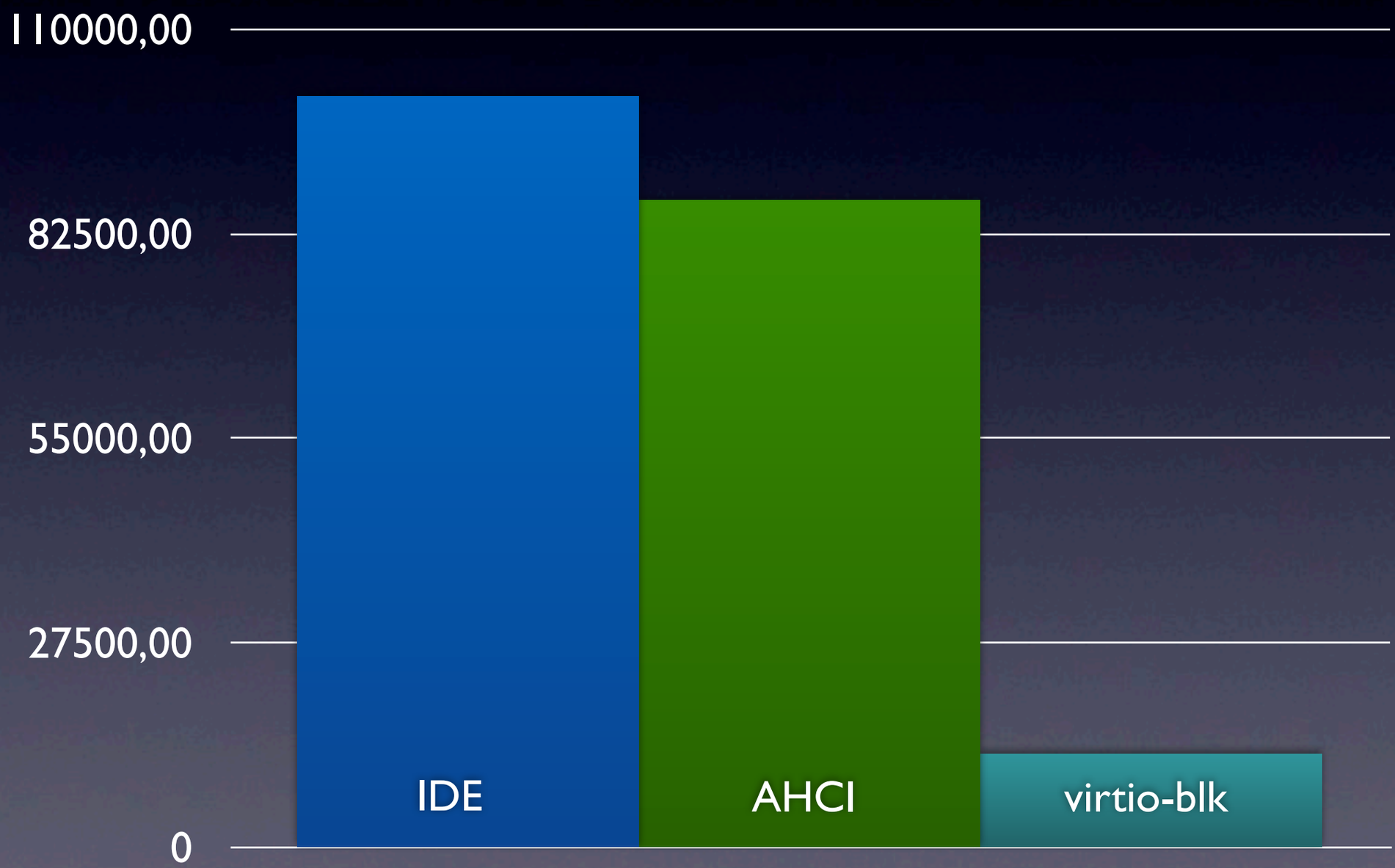
Exits per second



Exits per second



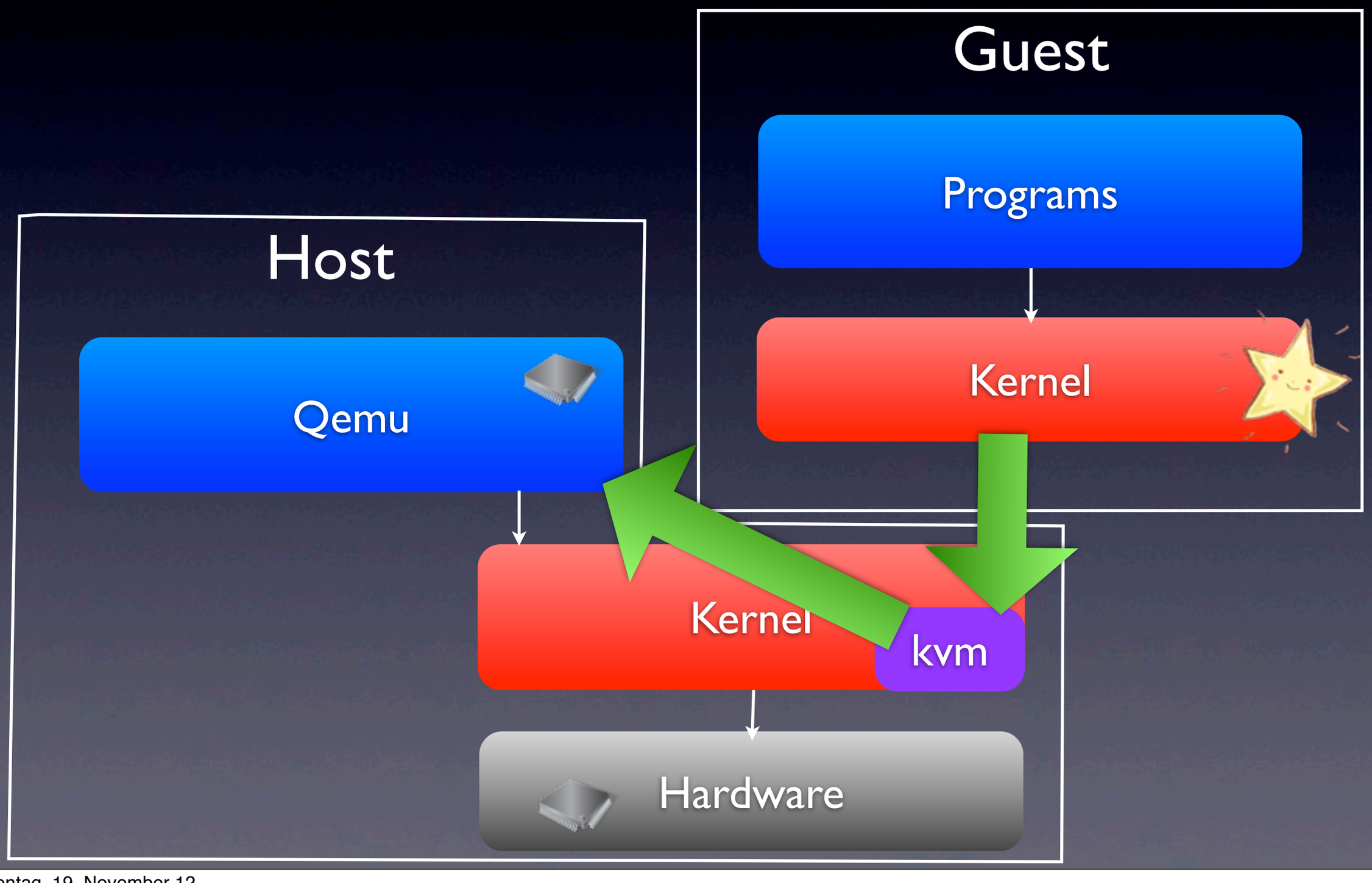
Exits per second



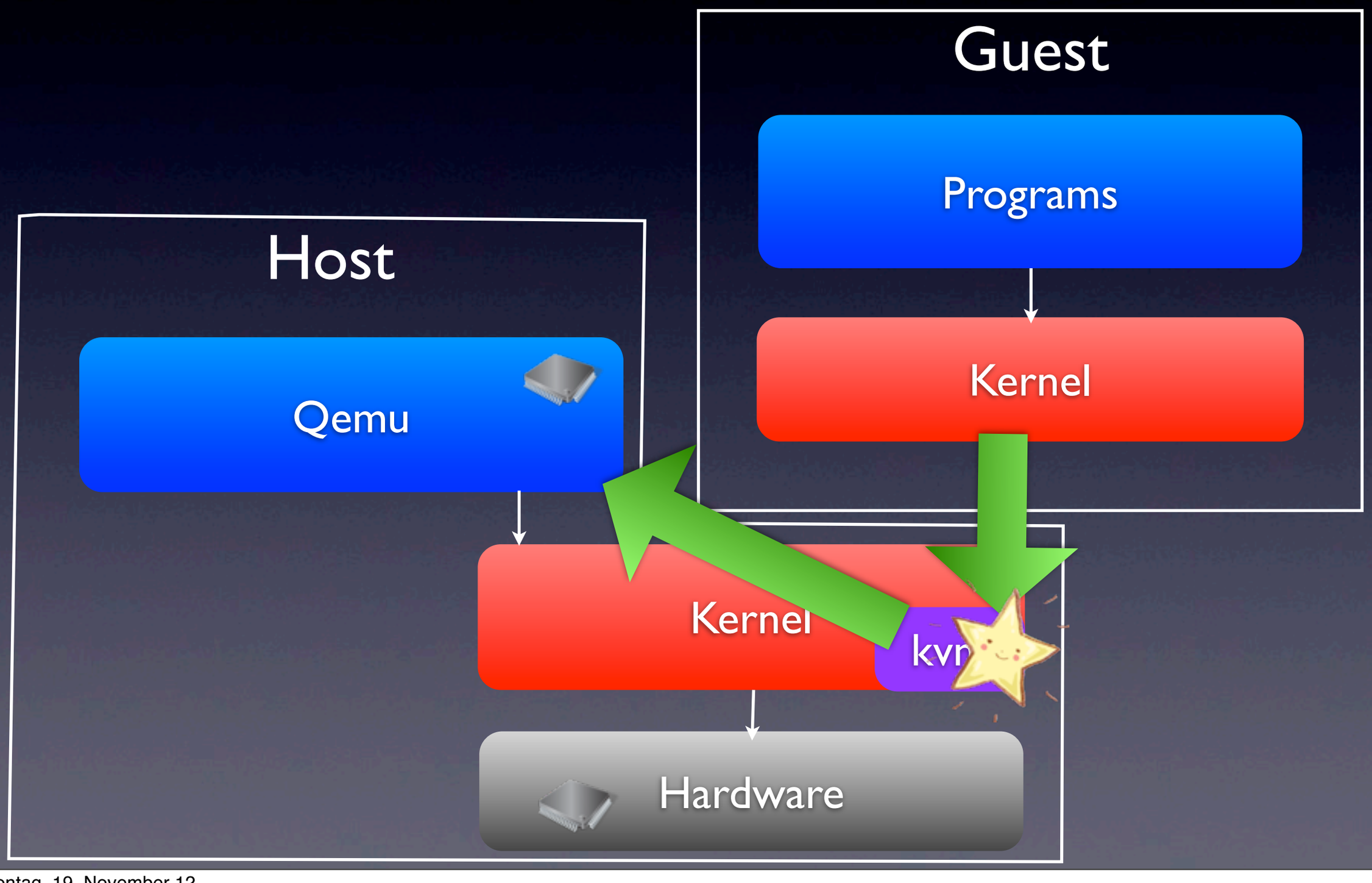
Exits per second

- Device port emulation is slow
- How to speed it up?

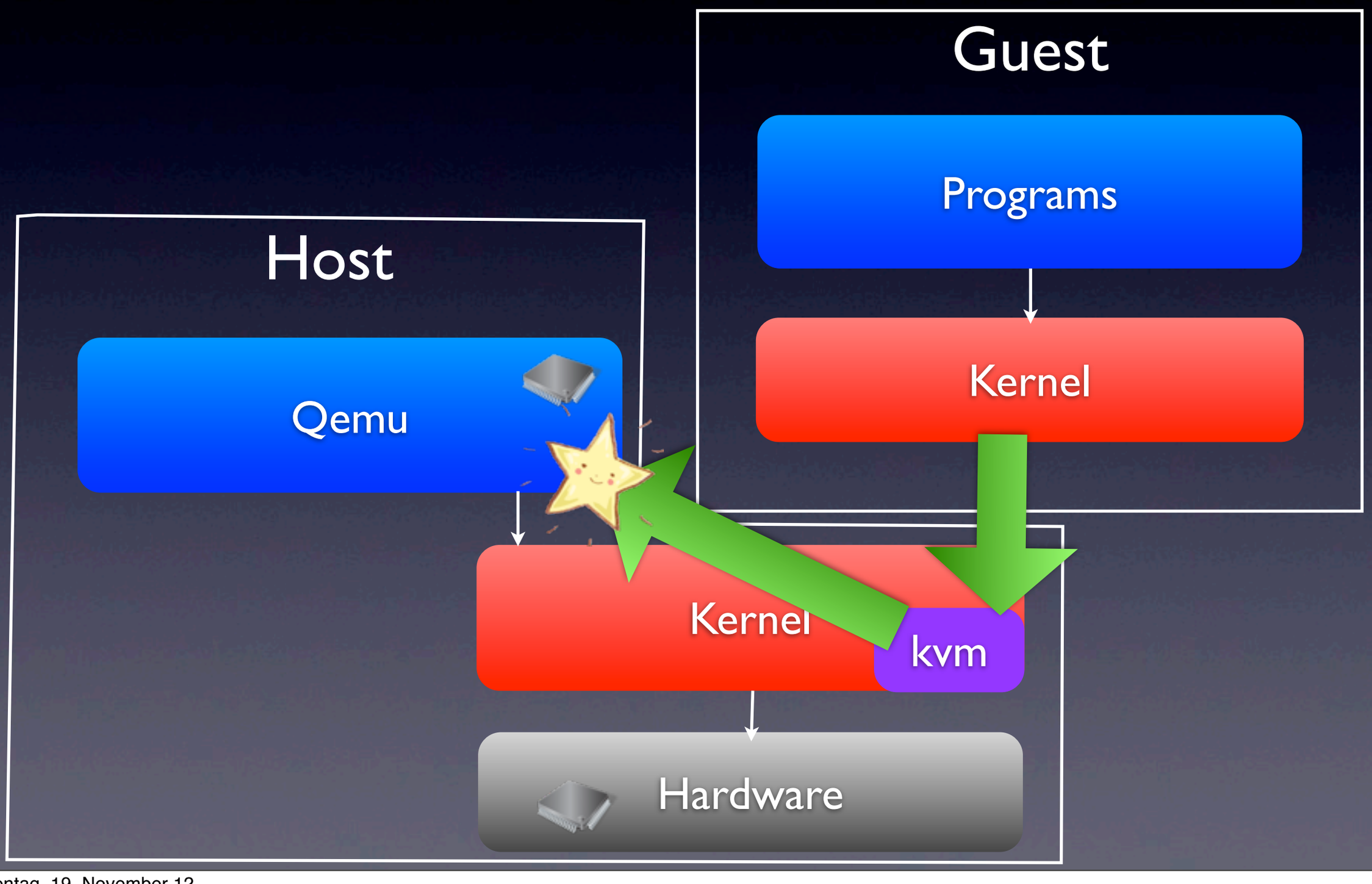
Device Access



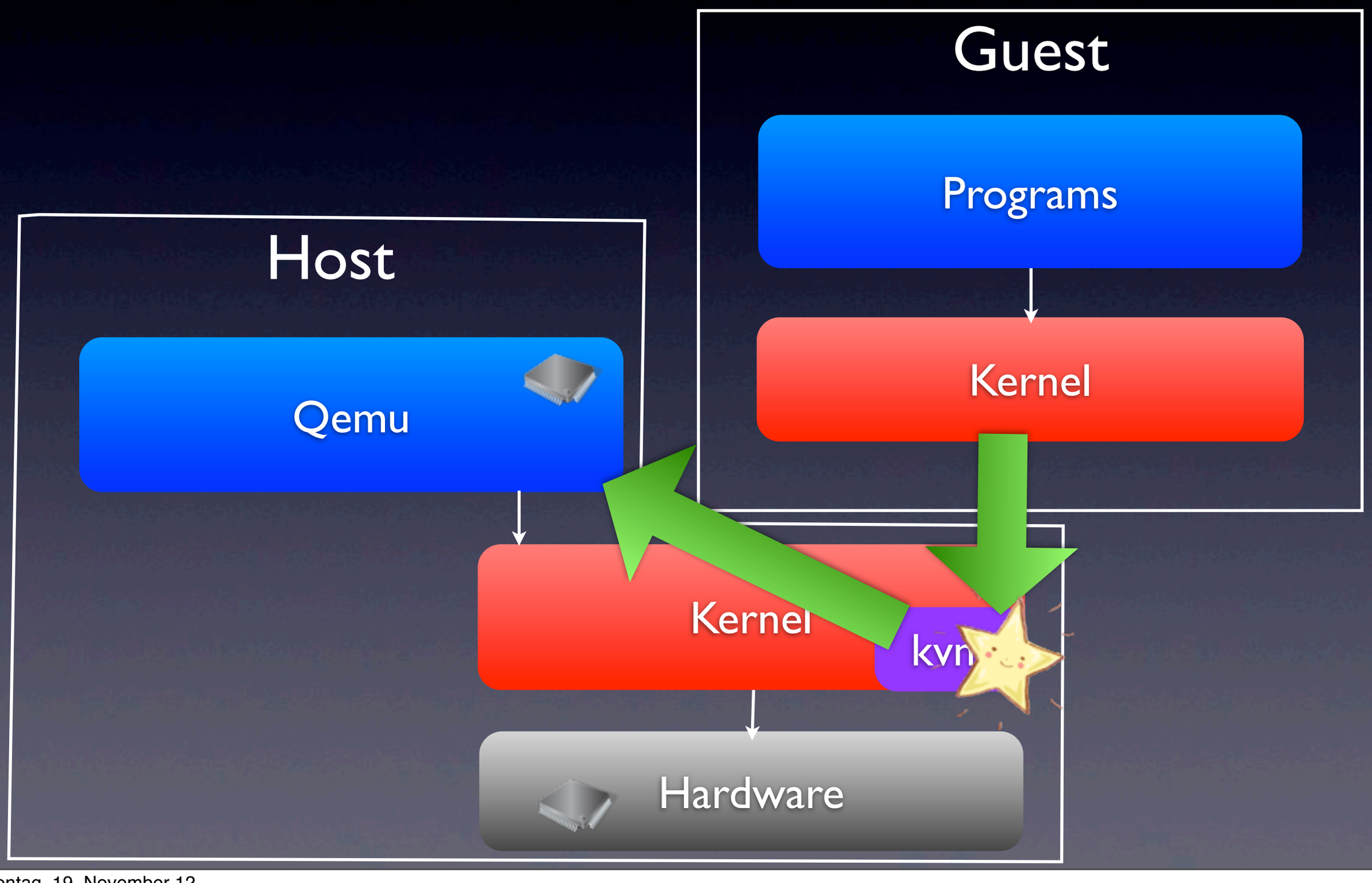
Device Access



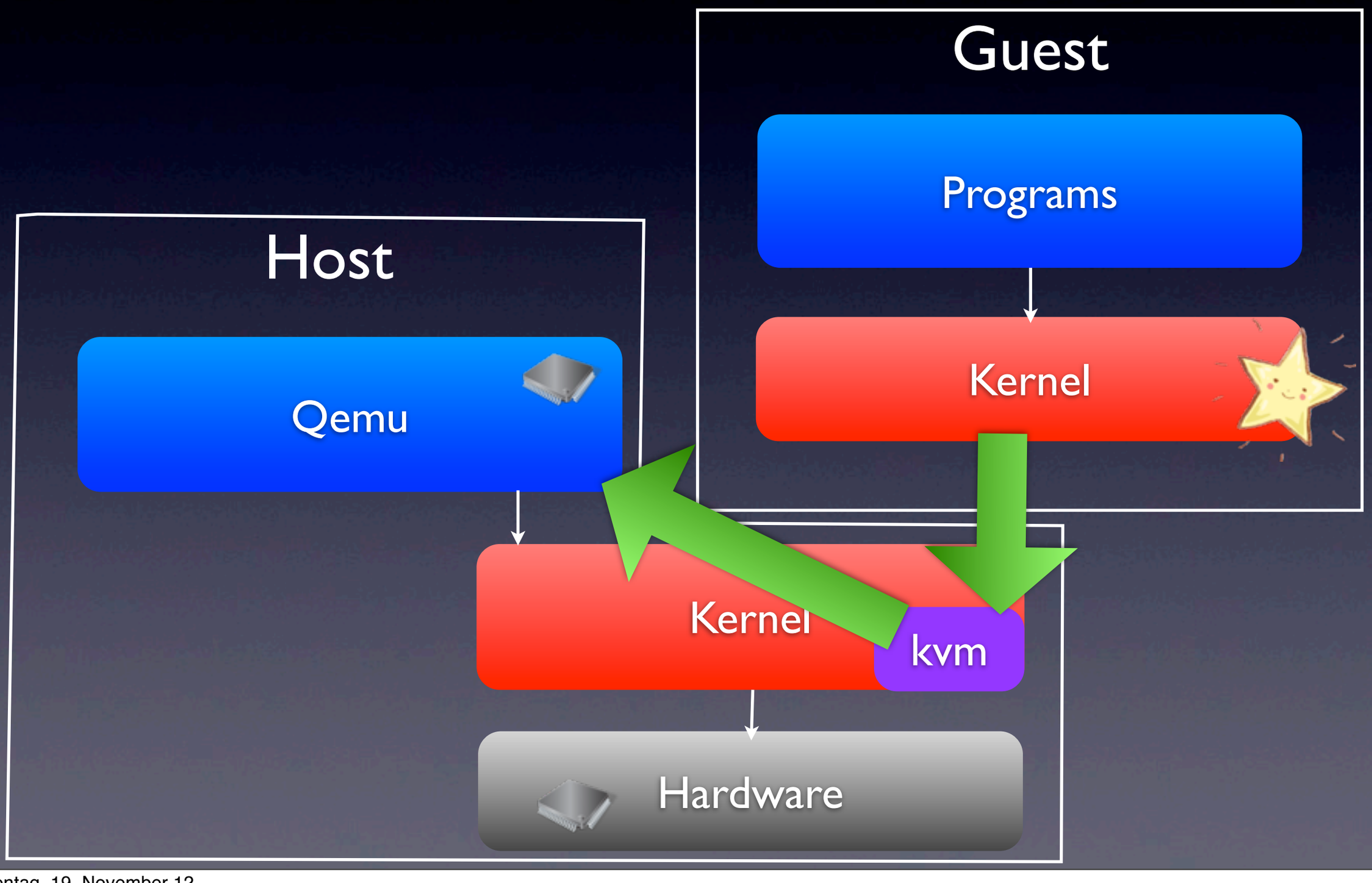
Device Access



Device Access



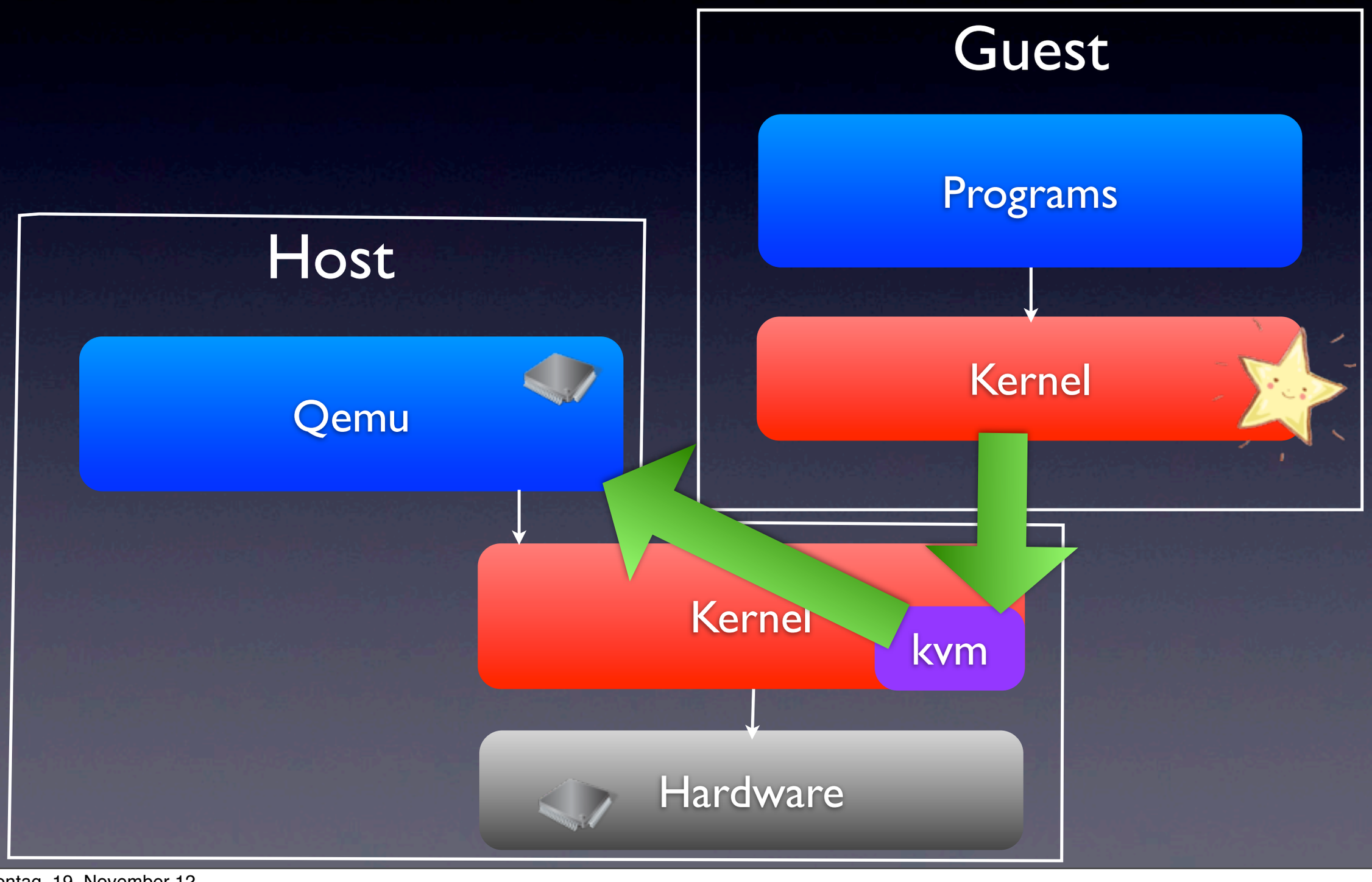
Device Access



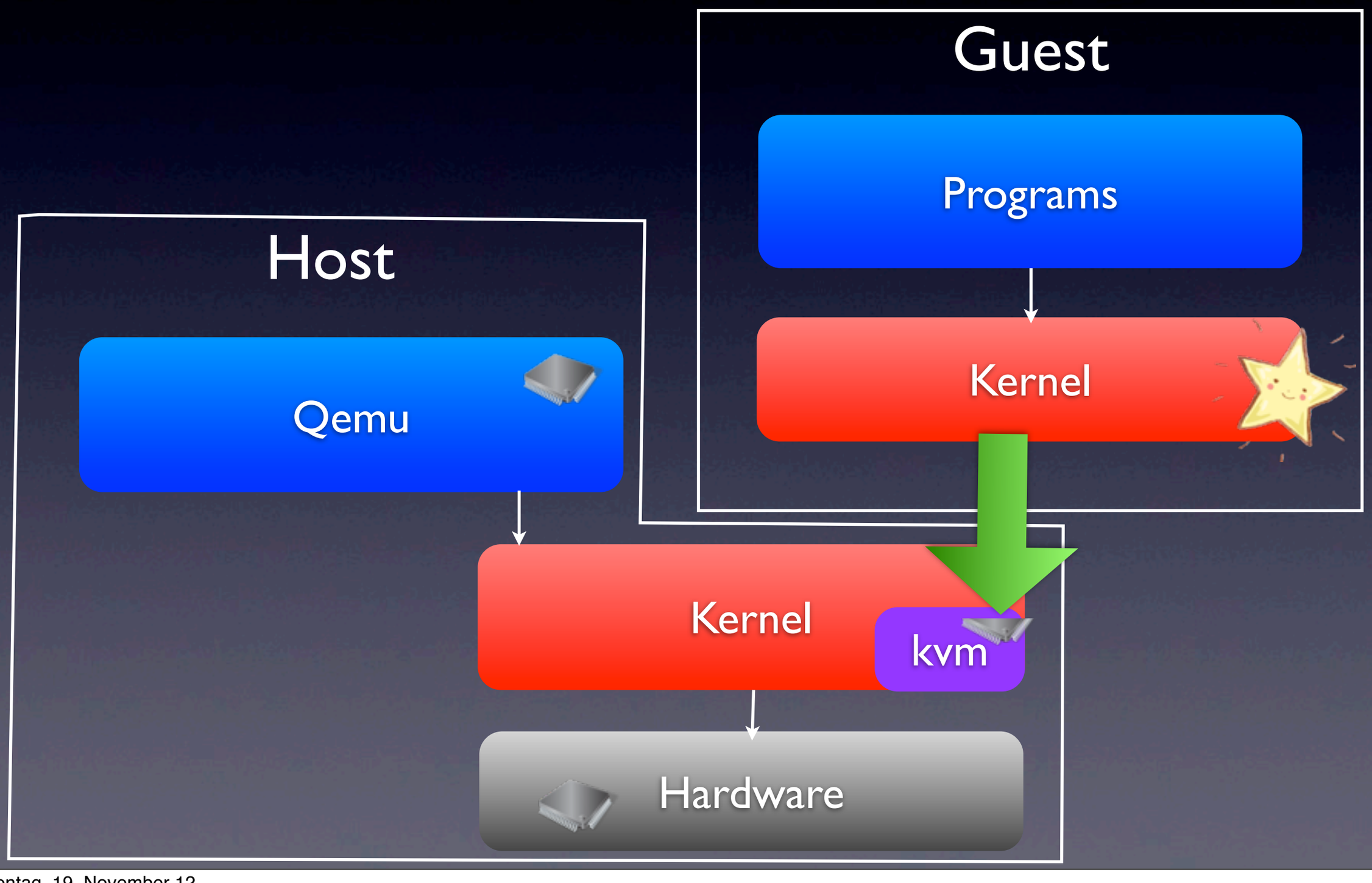
Device Access

What if we make this path shorter?

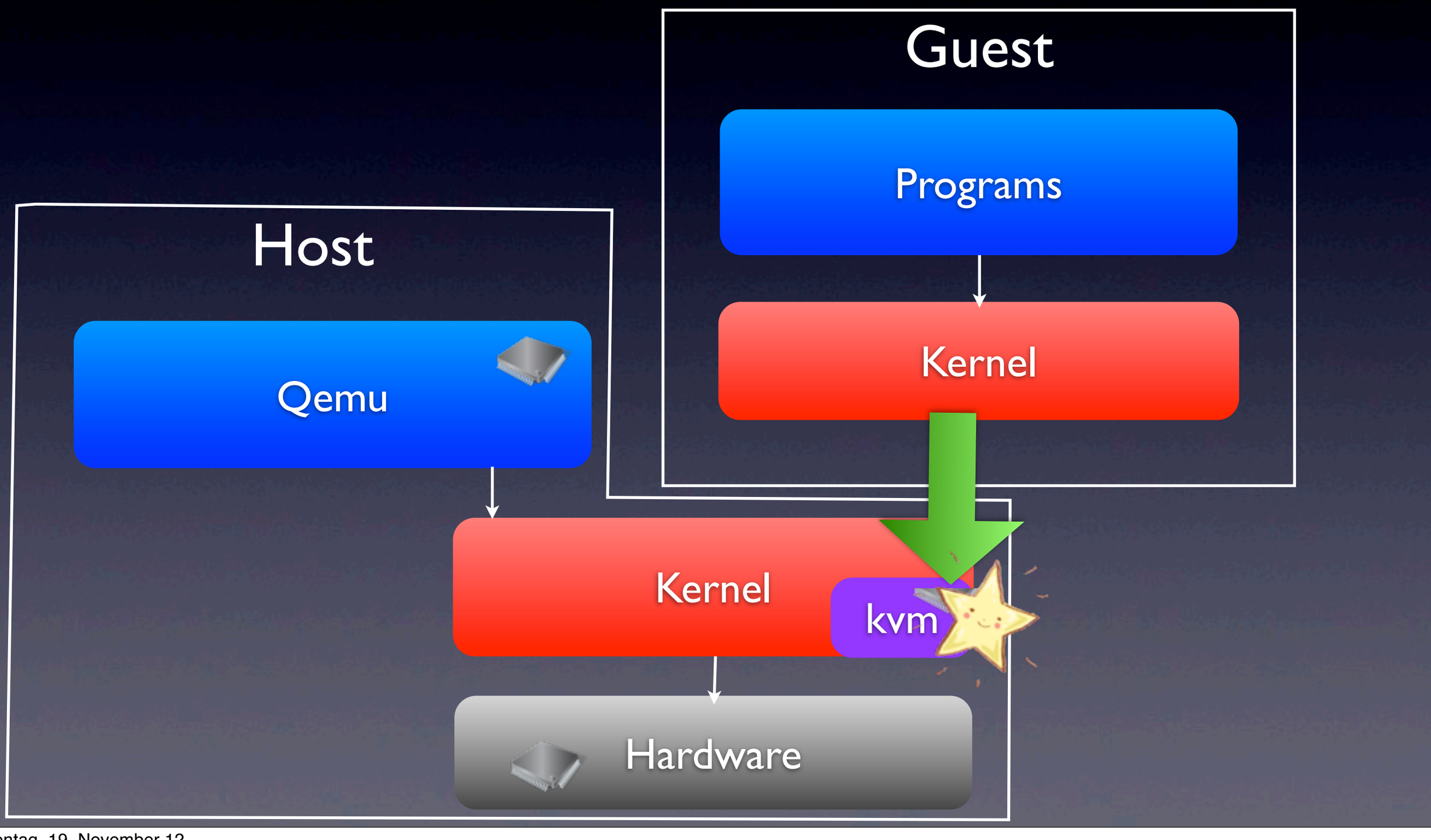
Device Access



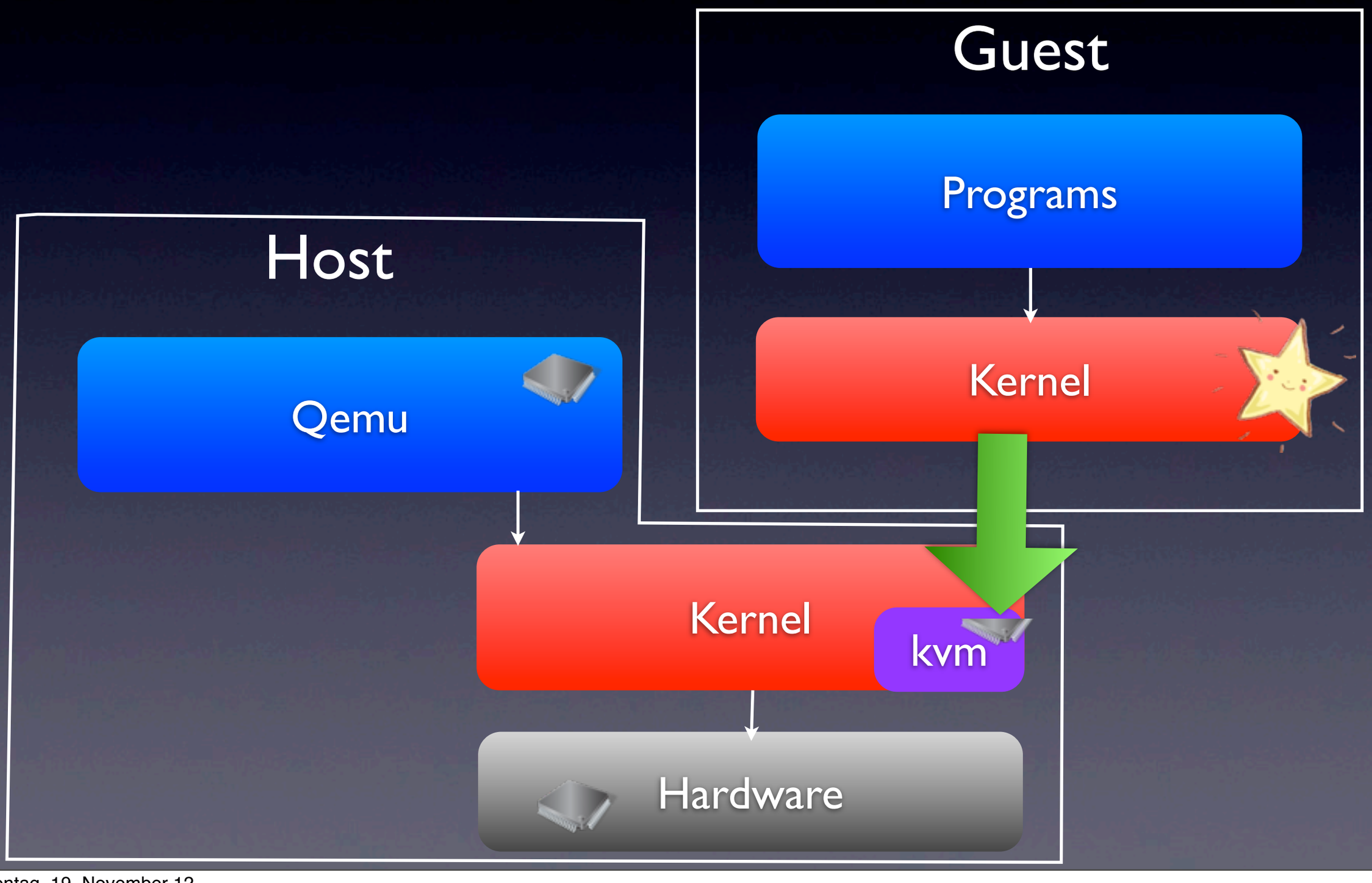
Device Access



Device Access



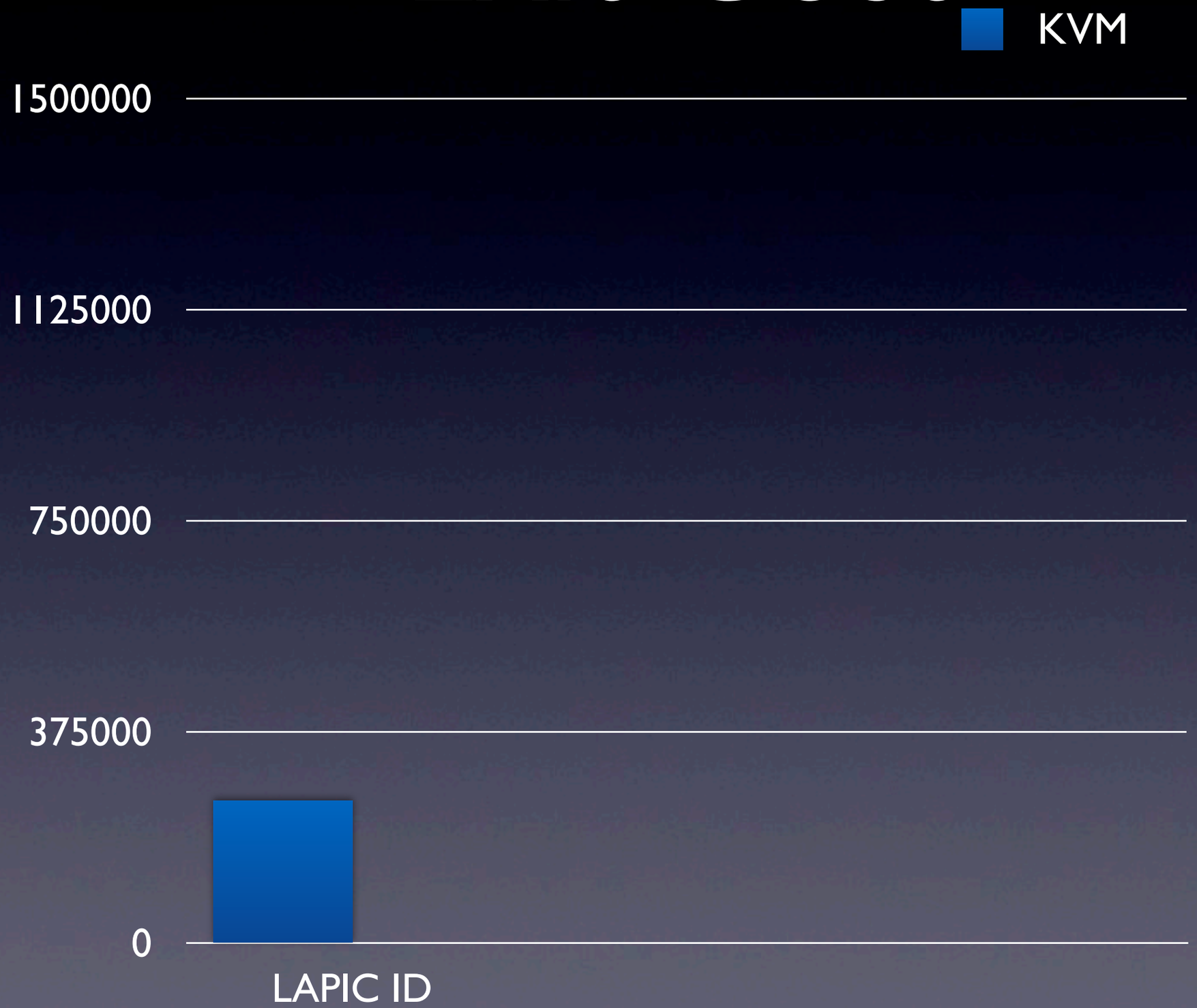
Device Access



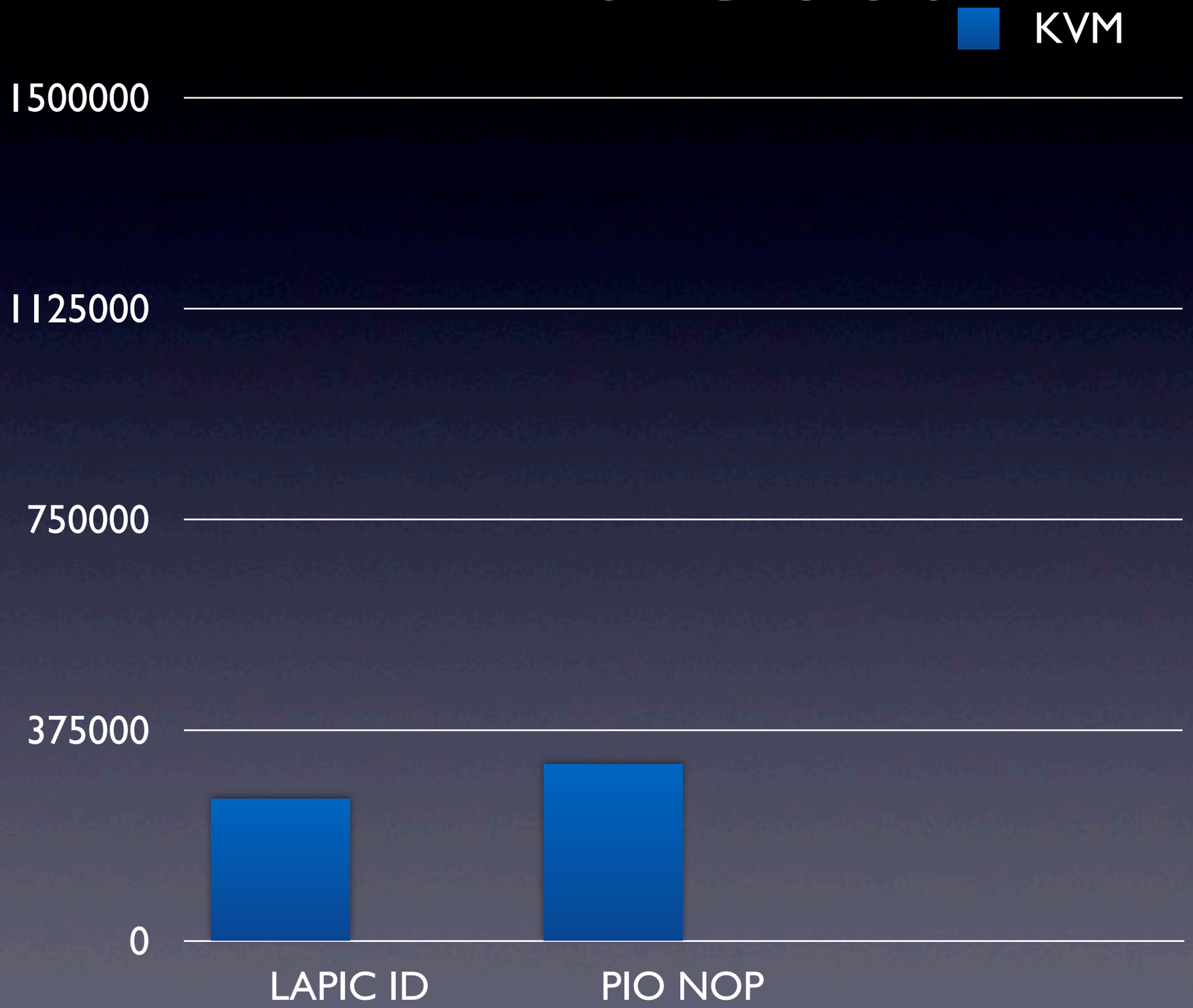
Exit Cost

Handle registers in KVM = speed?

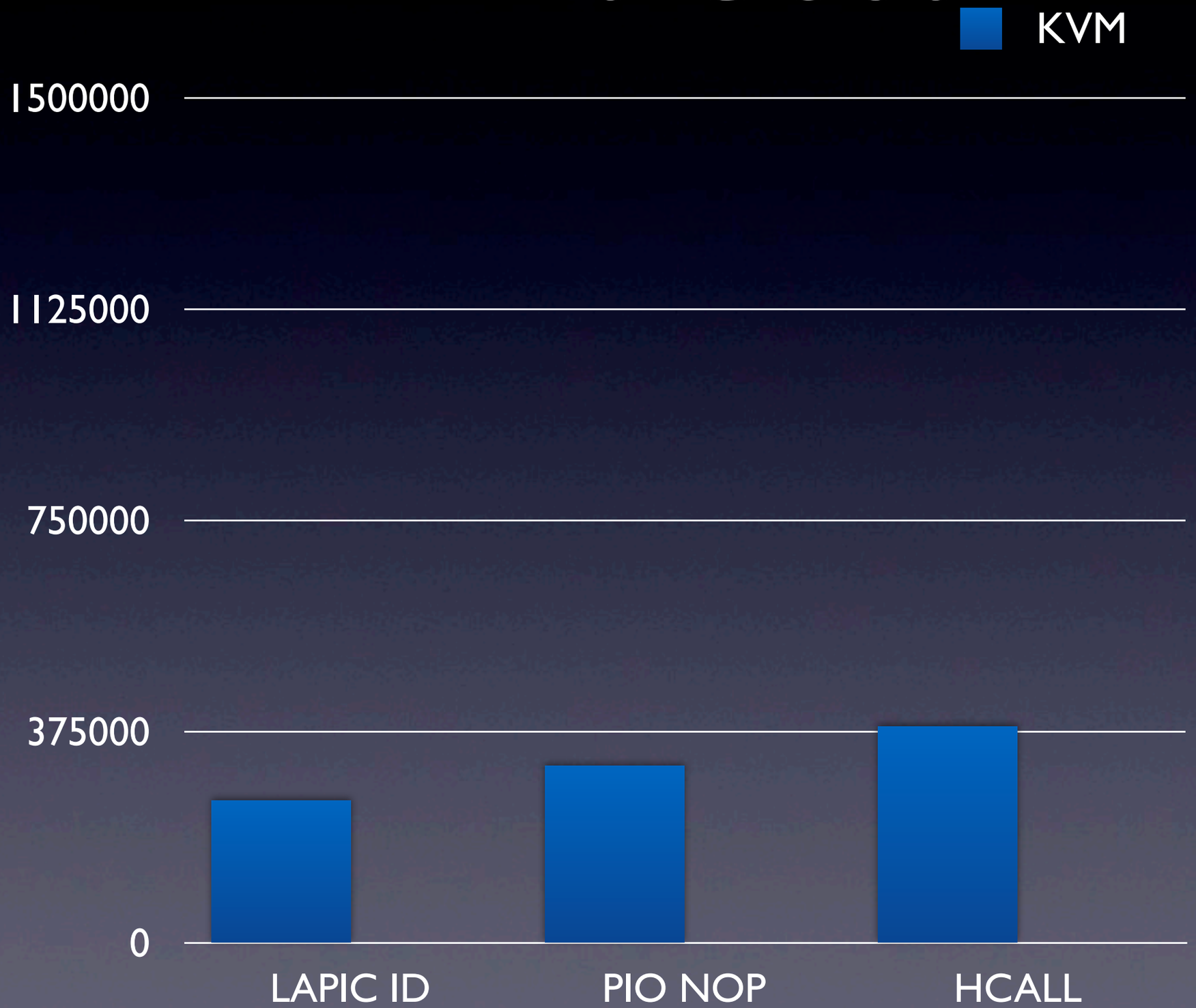
Exit Cost



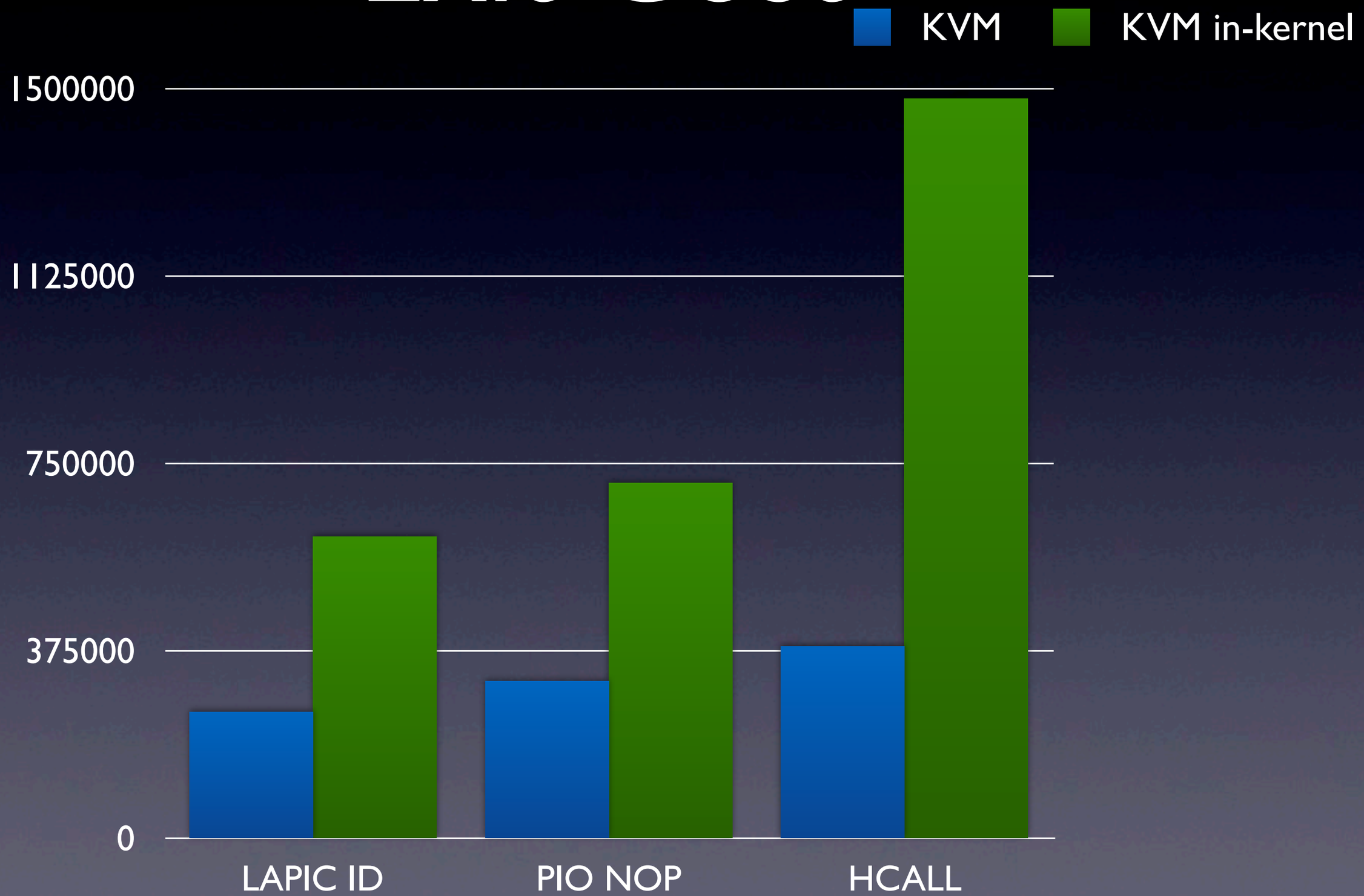
Exit Cost



Exit Cost



Exit Cost


KVM
KVM in-kernel
1500000
1125000
750000
375000
0
LAPIC ID
PIO NOP
HCALL

Exit Cost

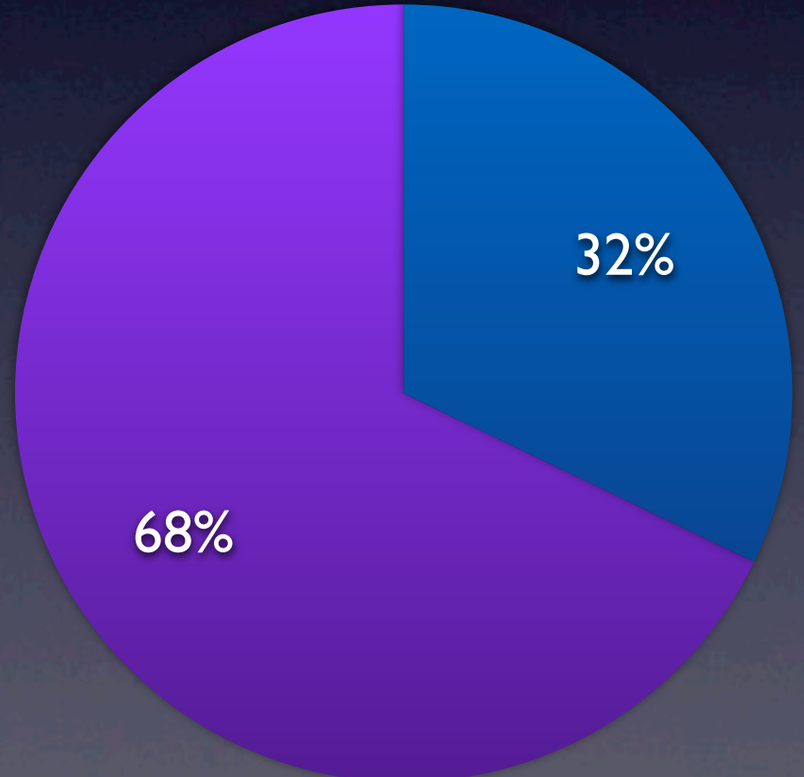
- Handling registers in QEMU doesn't scale
- Handling registers in KVM is faster

Exit Cost

- Is it worth optimizing?

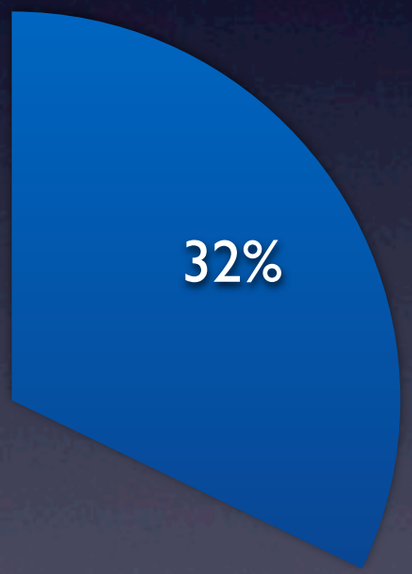
IDE Cost

- Handling Port Exits
- Other

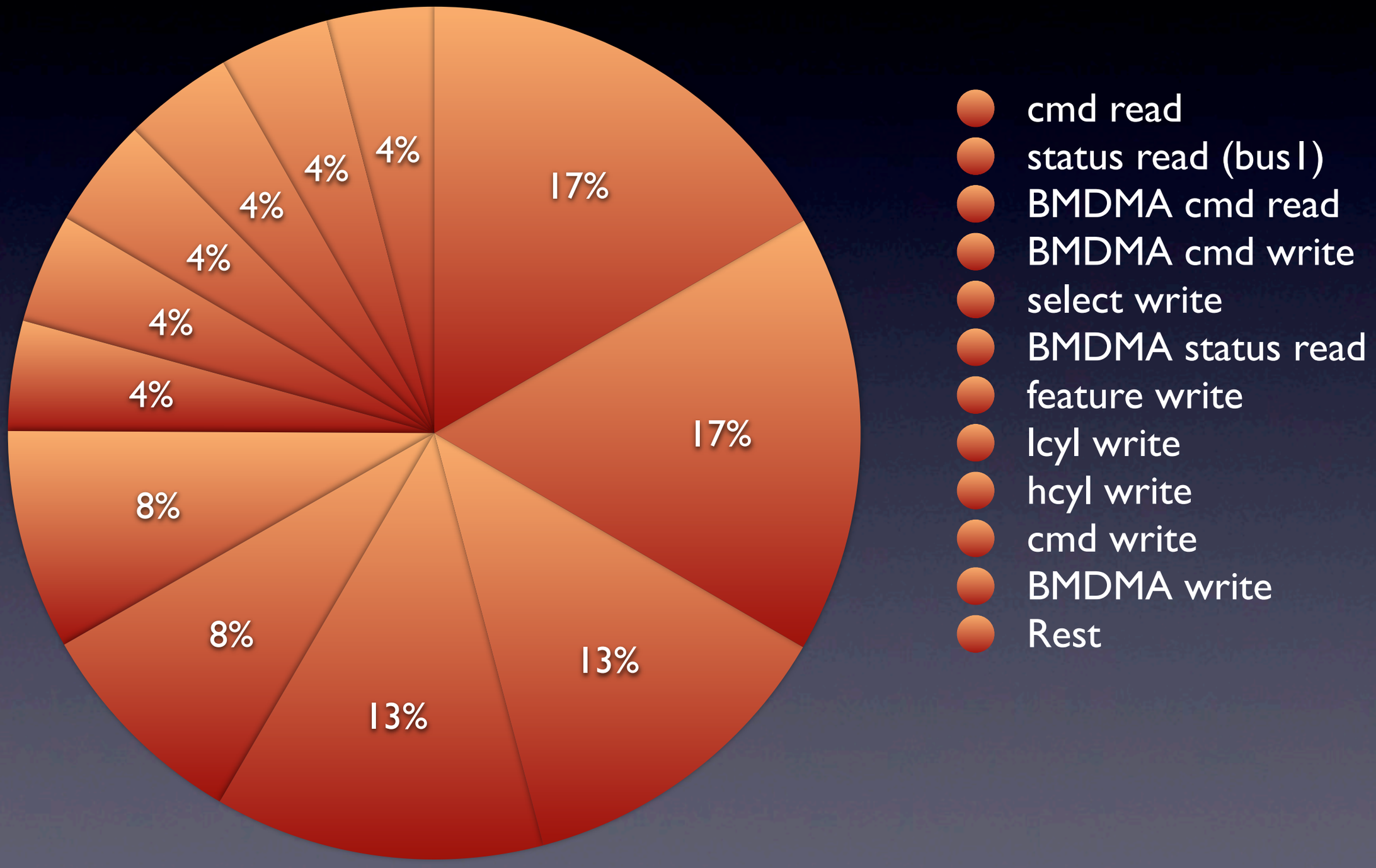


IDE Cost

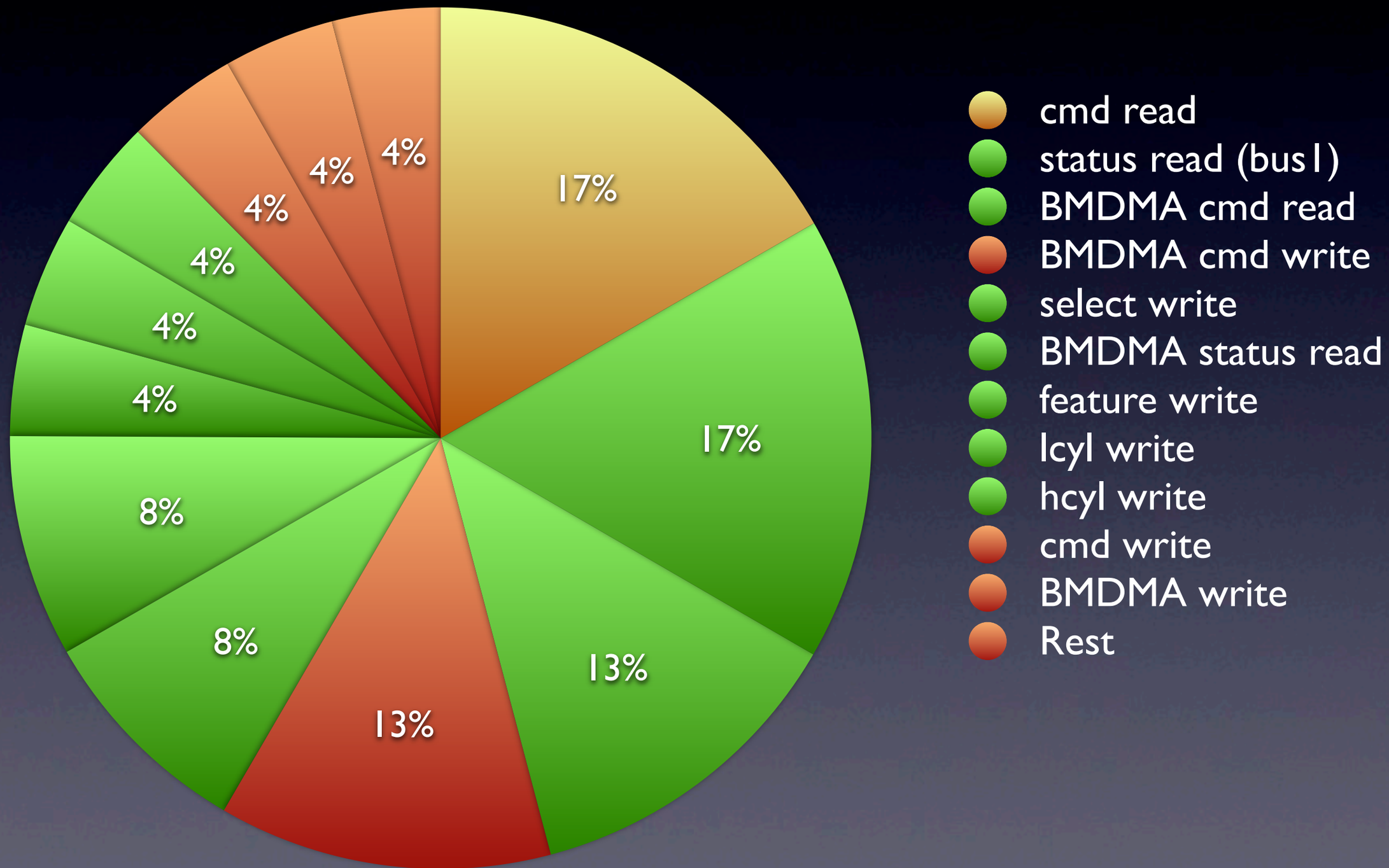
- Handling Port Exits



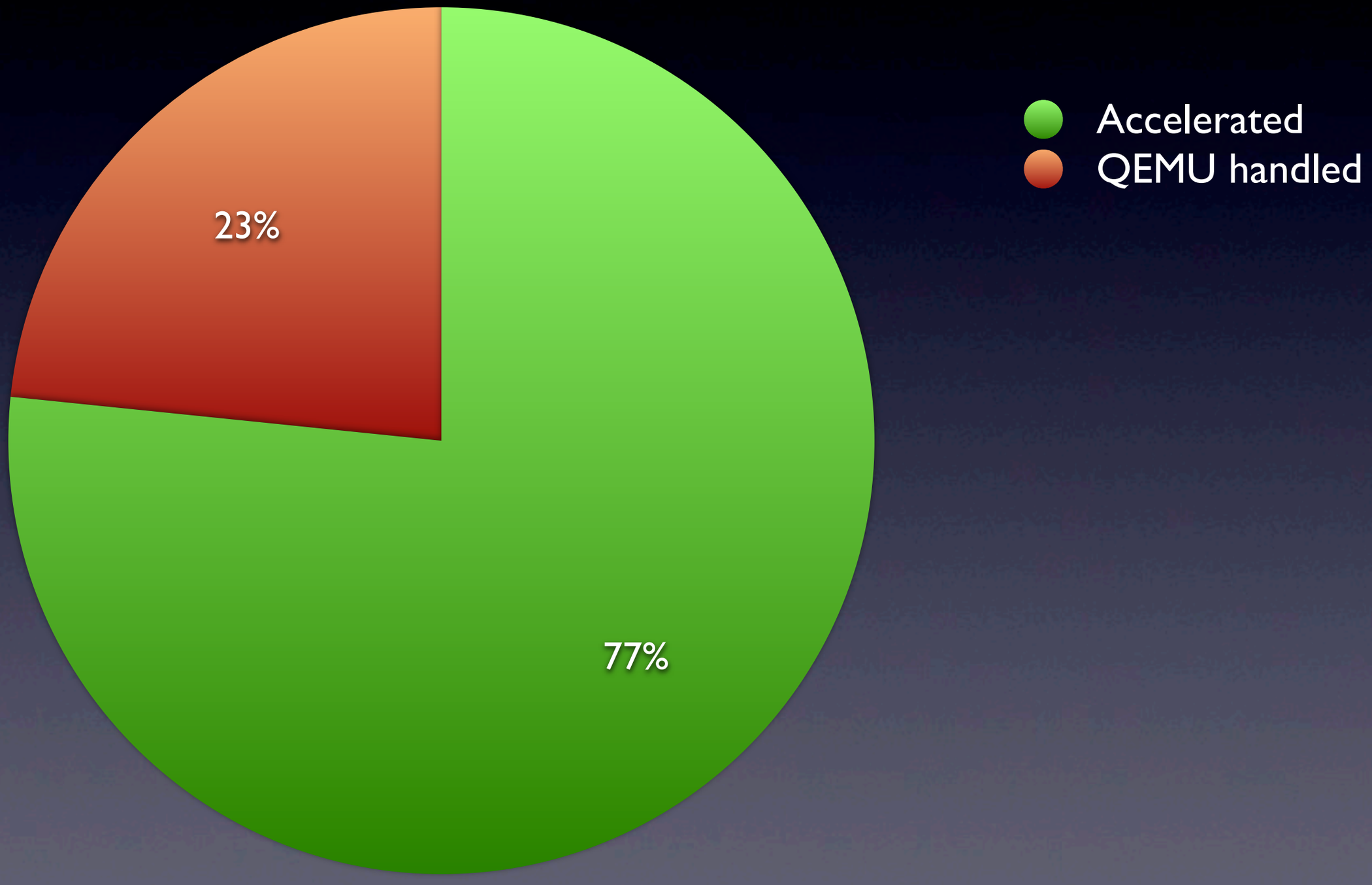
IDE PIO Access



IDE PIO Access

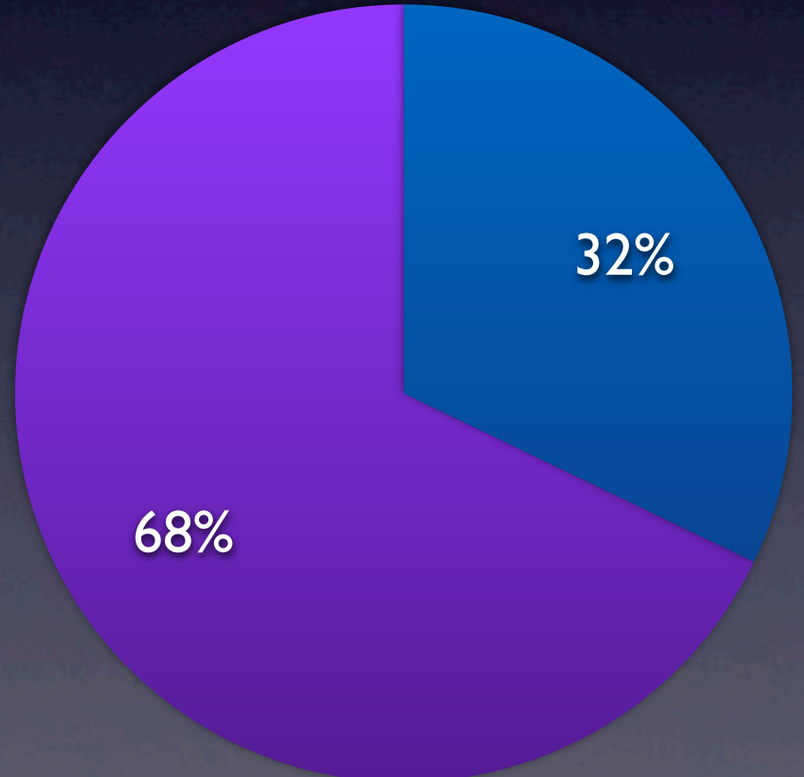


IDE PIO Access



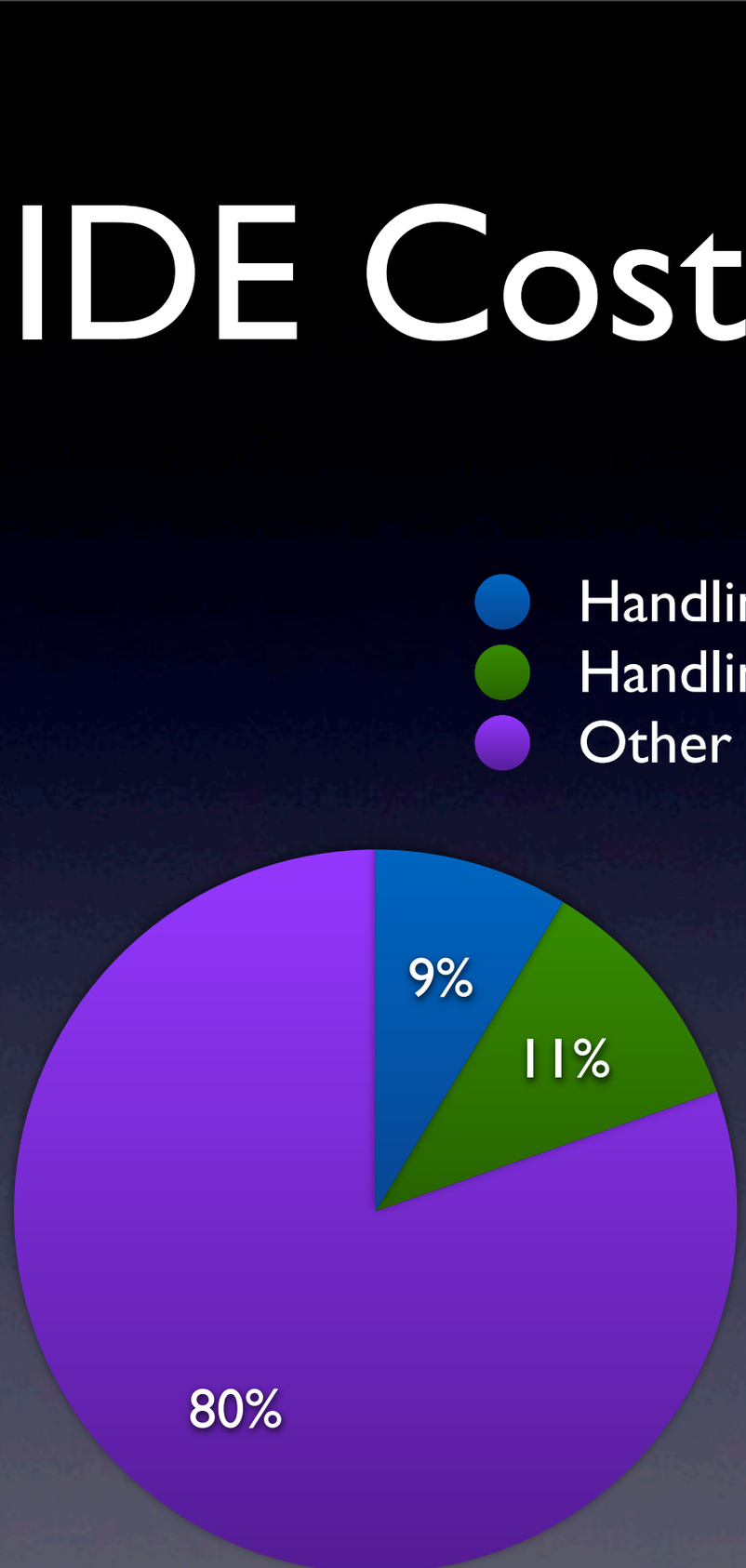
IDE Cost

- Handling Port Exits
- Other

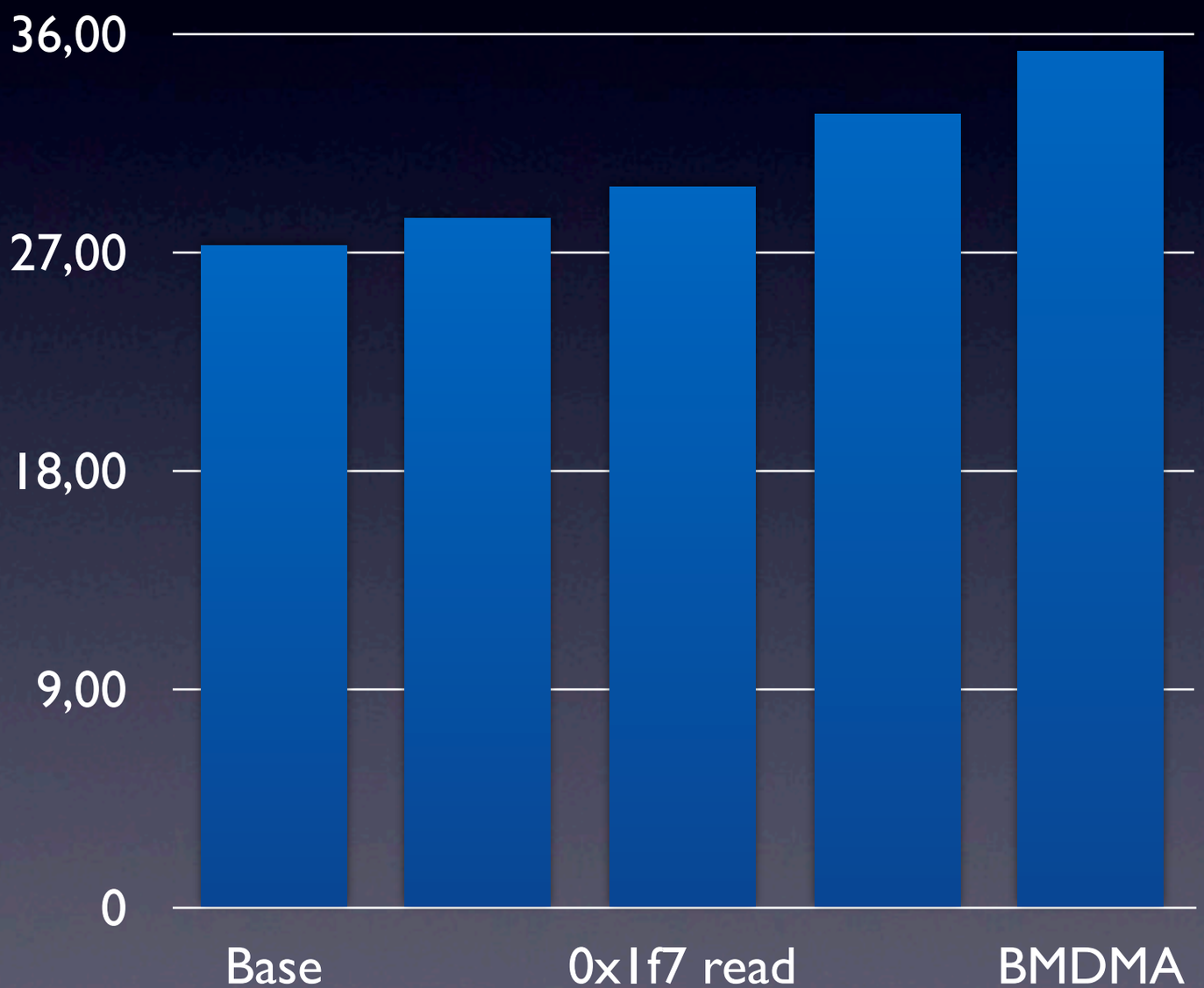


IDE Cost

- Handling Port Exits (QEMU)
- Handling Port Exits (KVM)
- Other

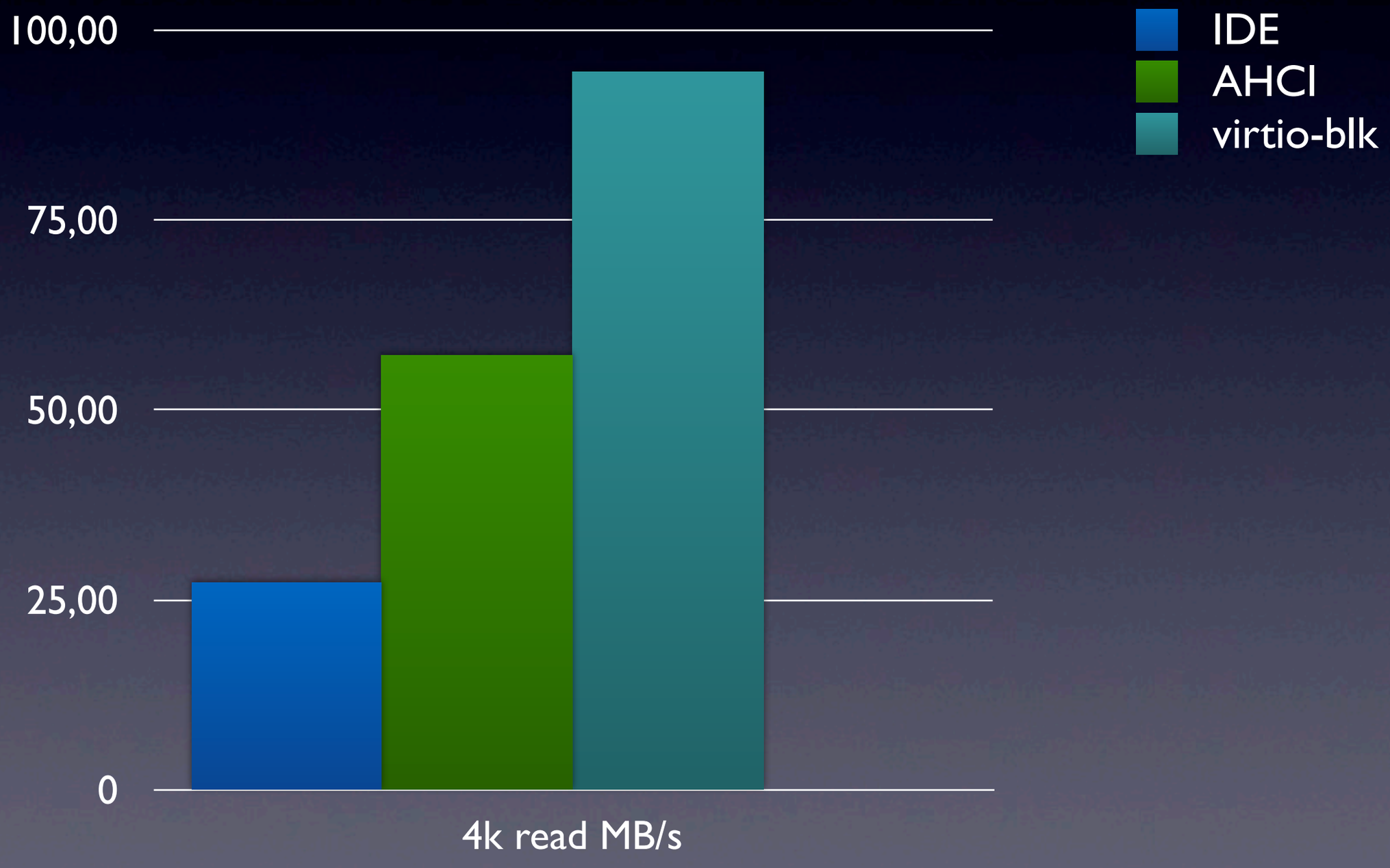


IDE PIO Access

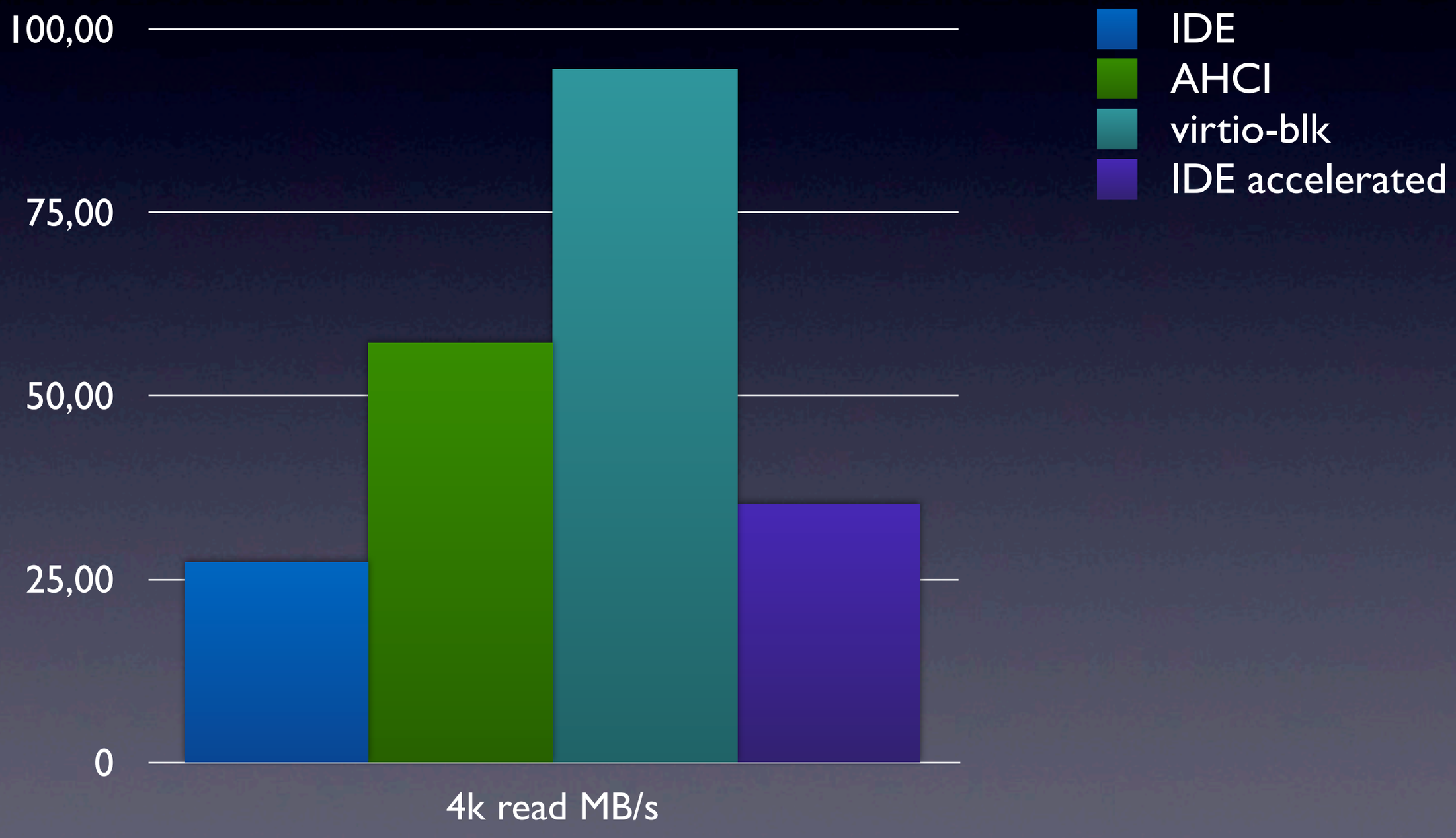


29% improvement

Disk performance



Disk performance



Problems

- Locking
- 2 code paths for the same logic

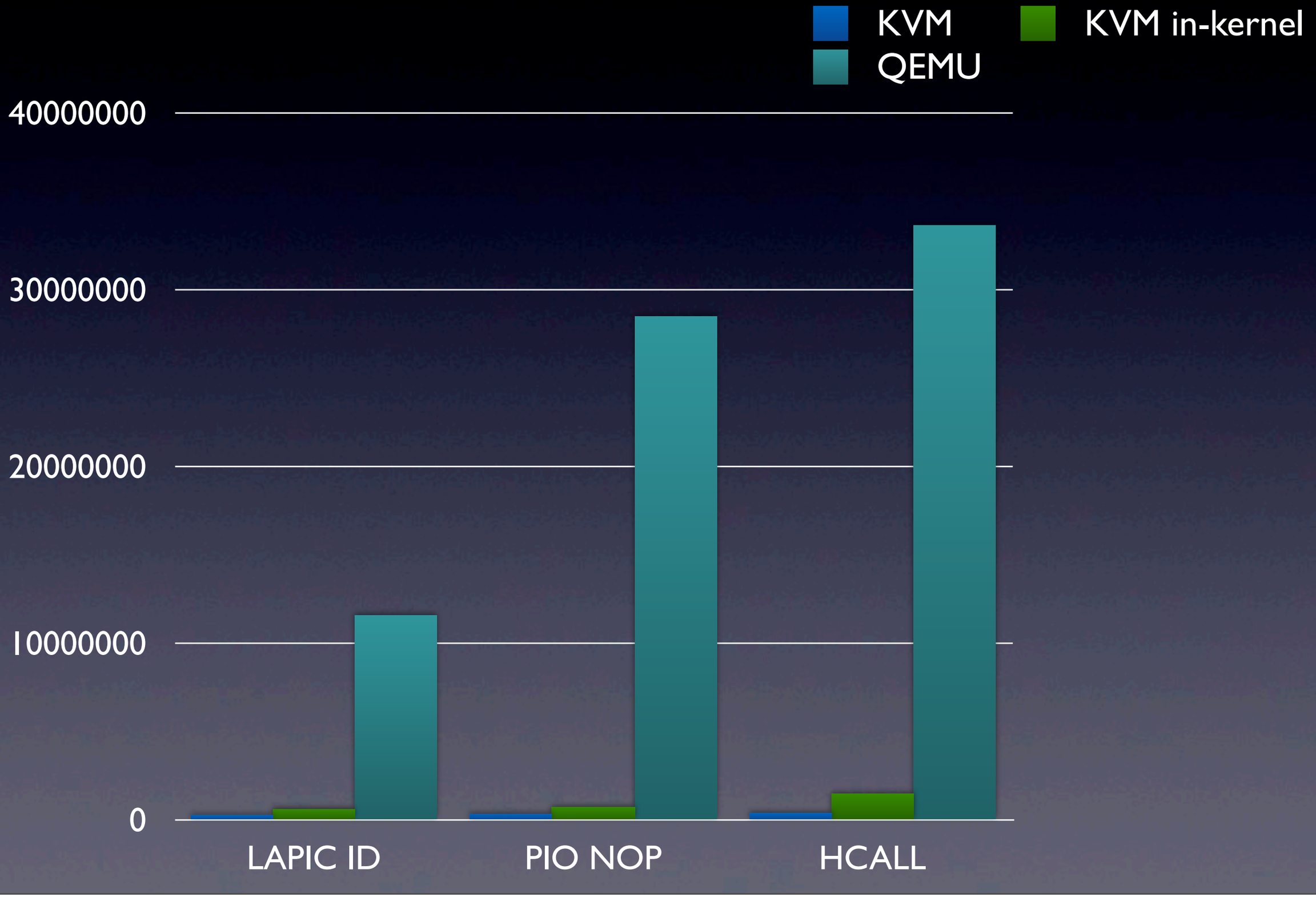
Conclusions

- Consider this when designing locks in QEMU
- Need a flexible framework
- Need to fix the other 80% in IDE
- Check for other devices that could benefit
- Idea: Implement PIO hcall for vhost

Questions

Thank You

Exit Cost


KVM
KVM in-kernel
QEMU
40000000
30000000
20000000
10000000
0
LAPIC ID
PIO NOP
HCALL

AHCI MMIO Access