Integrating KVM with Linux

Avi Kivity
August 10, 2010

Agenda

To Integrate or Not To Integrate?

Issues

Suspend/resume

Preempt notifiers

MMU notifiers

Events

User return notifiers

Lazy FPU

Conclusions

Advantages and Disdvantages

- Historically, zero impact made the merge into Linux 2.6.20 quick and painless
- Special hardware has special needs
- Performance and functionality
- kvm-kmod

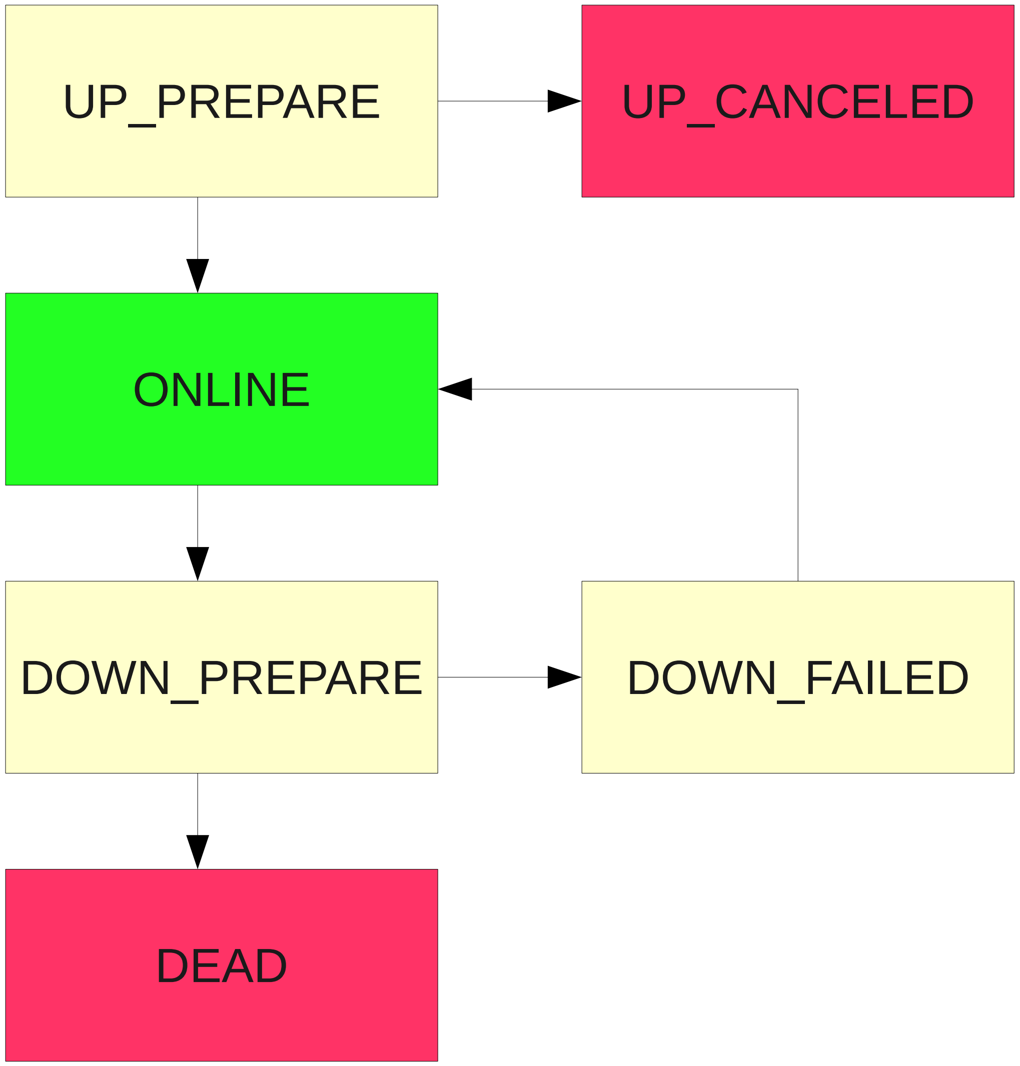
Issues

- Blend in with the rest of the infrastructure
- Zero impact when not configured
- Minor impact when configured and unused
- Find more users
- Coordinating multiple staging trees

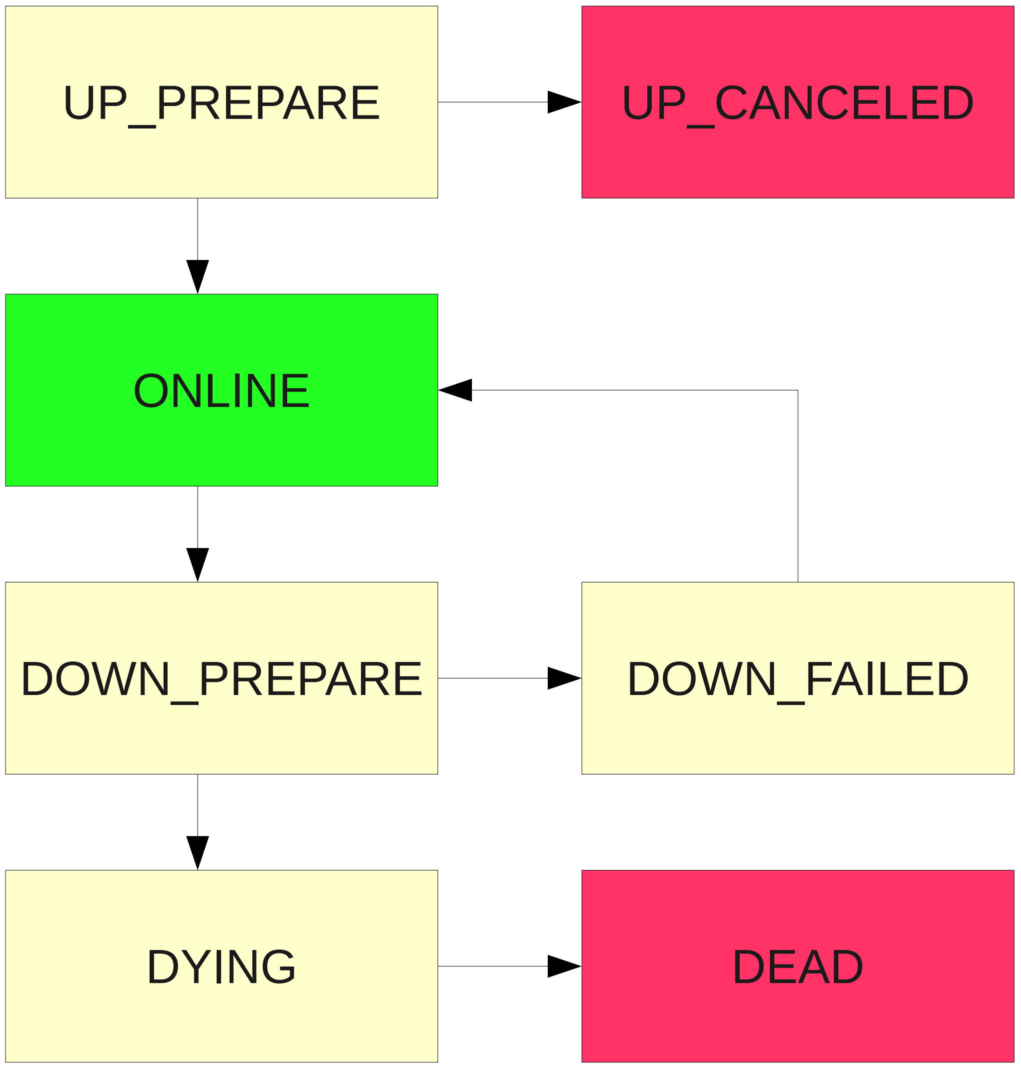
Suspend/resume, cpu hotplug

- Linux does power management for us
- But... VMX has on-core registers that Linux doesn't know about!

Death or a processor



Death or a processor



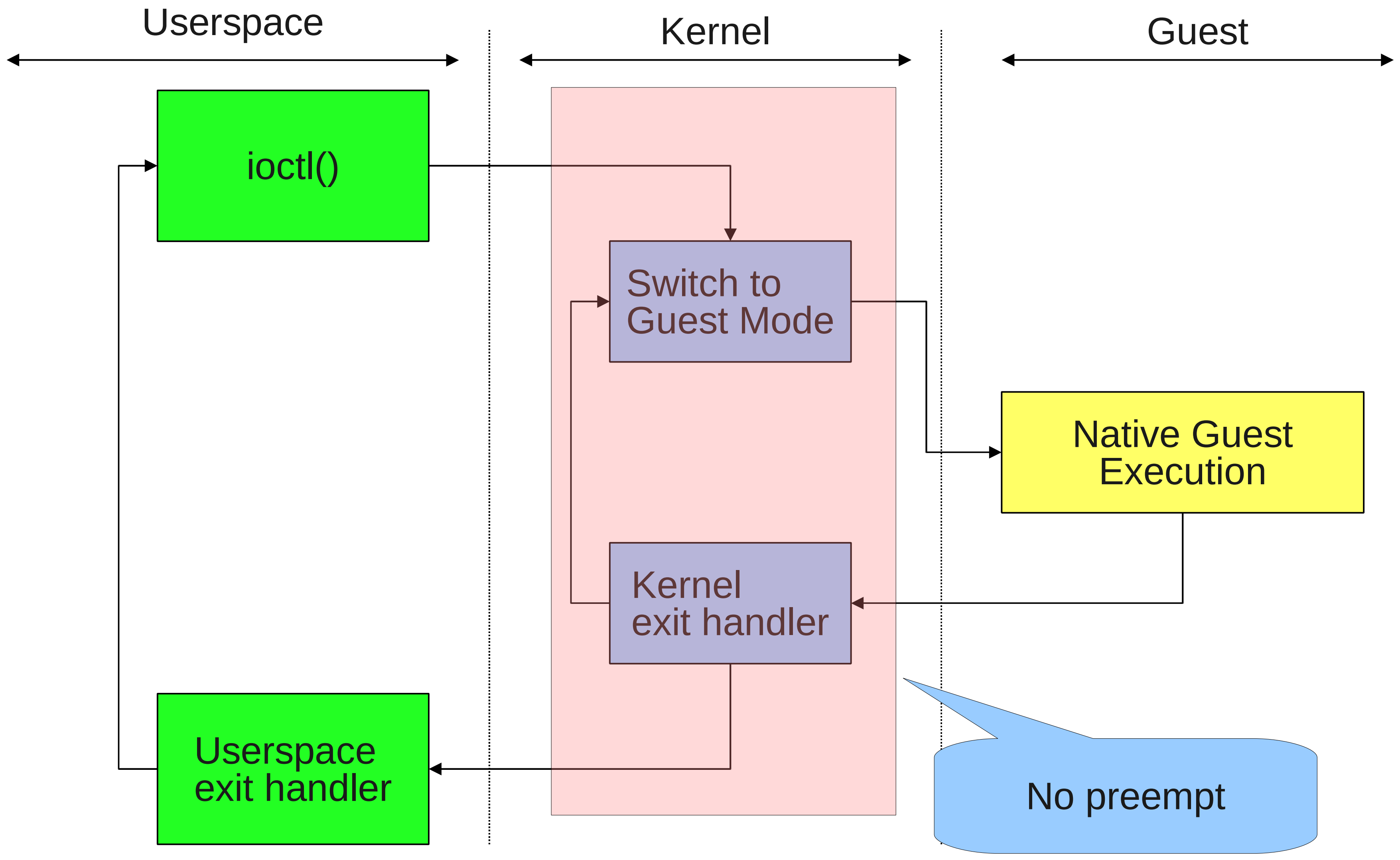
Suspend/resume, cpu hotplug

- Merged 2.6.23

Preempt notifiers

- Lightweight exits: host kernel runs with some guest state loaded
  - VMPTR
  - FPU
  - SYSCALL MSRs
- Task switch will clobber these register
- Initial solution: preemption disabled
  - Low QoS
  - Cannot allocate/swap/etc

Control flow



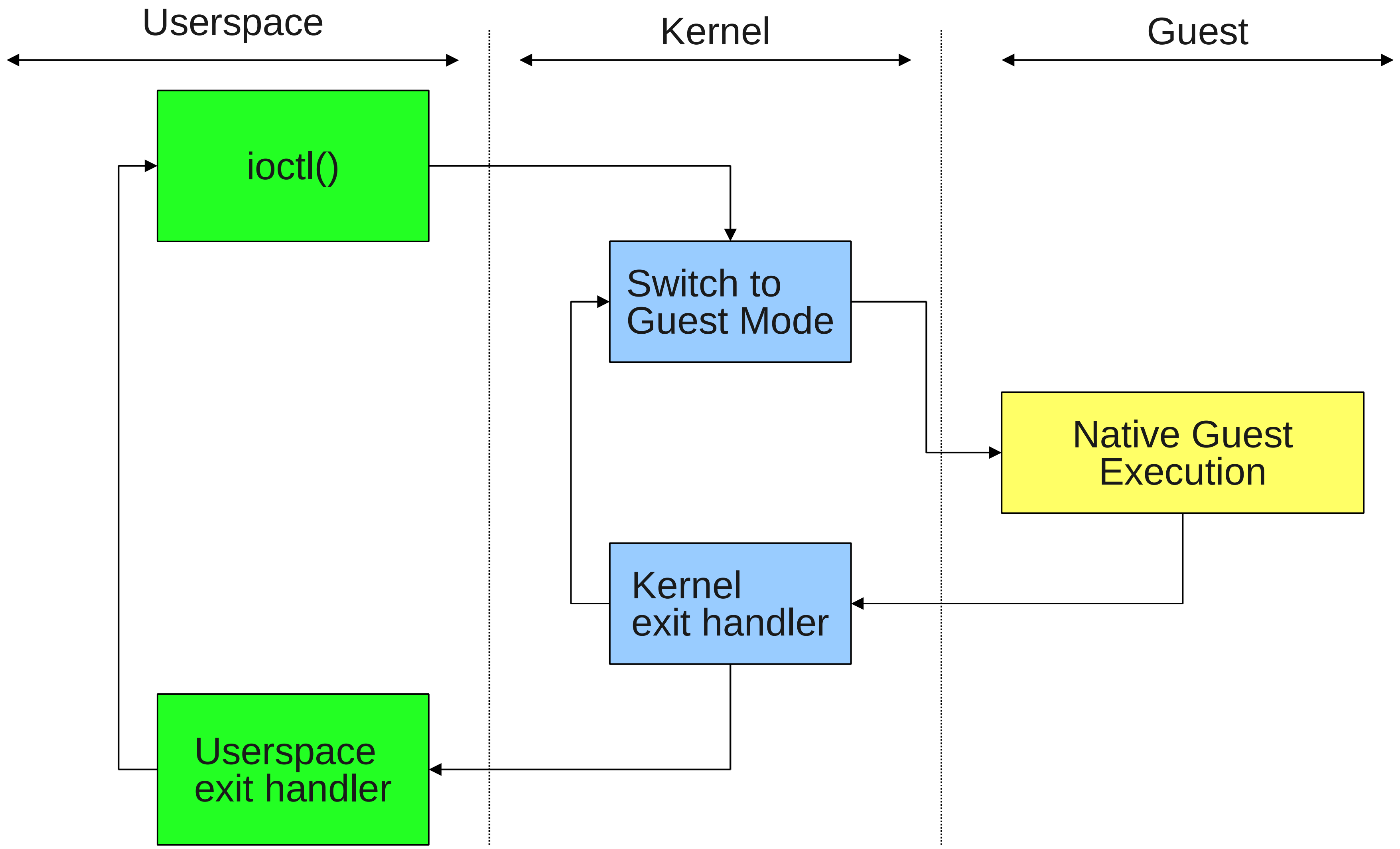
Preempt notifiers

- Task can ask schedulers for callbacks
  - sched_out() - save guest state, load host state
  - sched_in() - load guest state
- Kernel code no runs with preemption enabled
- Allocation, page-in allowed
- Merged in 2.6.23

Preempt notifiers – additional users

- Pending work for concurrency managed work queues
- New work threads spawned when a work thread blocks
- Can also be used for modular perf events

Control flow



MMU notifiers

- The K in KVM...
- Two way synchronization between Linux page tables and KVM shadow page tables
  - Linux updates a PTE (page out, recency scan, COW), then updates KVM
  - Hardware updates KVM shadow PTE, transferred to Linux page tables
- Also used for SGI's GRU, XPMEM
- May also be used for PCI ATS – device assignment
- Merged in 2.6.27

Events

- Native I/O model is synchronous
  - Guest issues I/O instruction
  - KVM emulates
  - Exits to userspace for fulfillment
  - Guest is blocked while userspace processes I/O
- Good model for lightweight processing
  - No context switch overhead
  - Cache hot
- Interrupts asynchronous, but driven from userspace

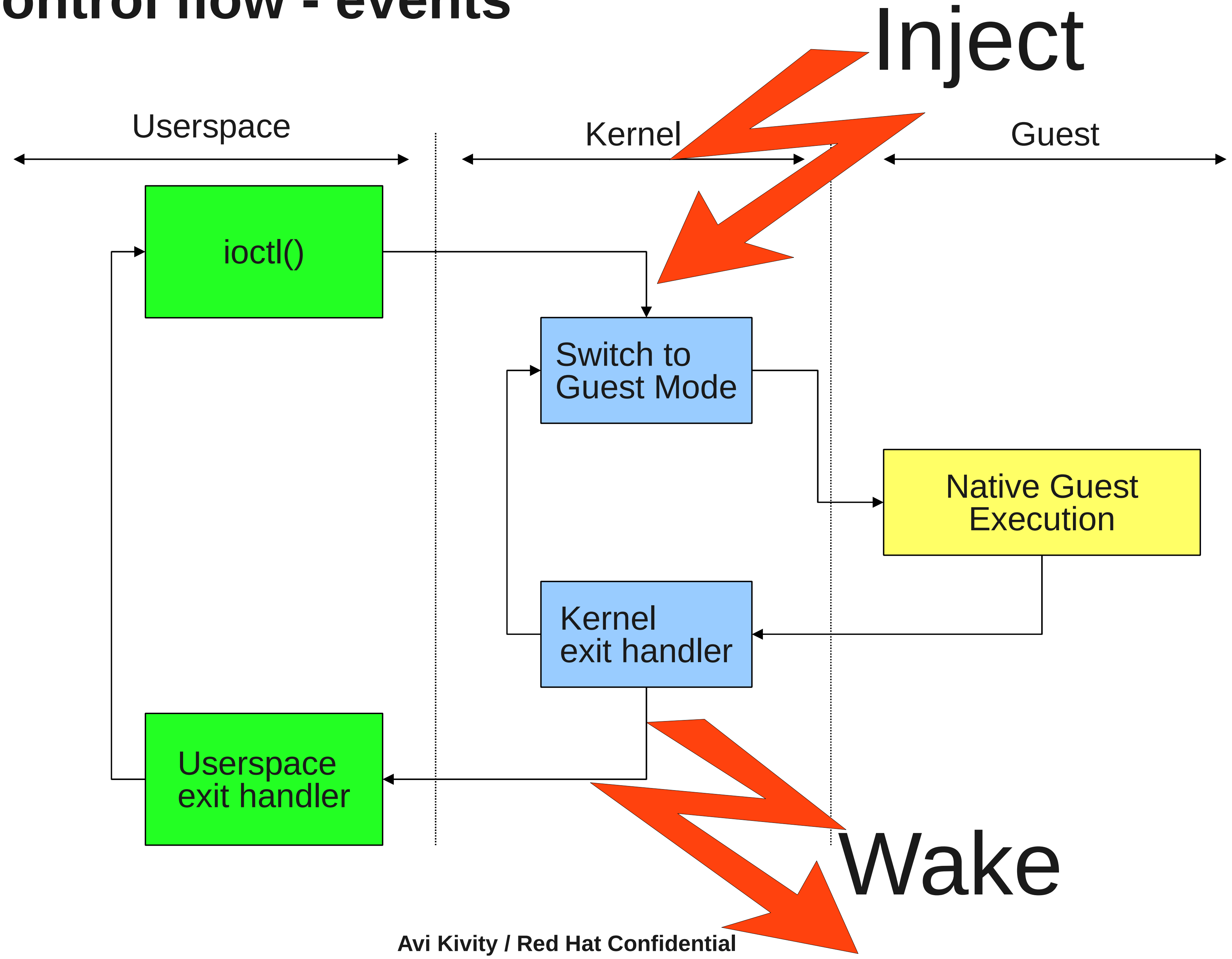
Events - problems

- Want to process events in kernel, not userspace
- Want to inject interrupts from kernel, not userspace
- Want asynchronous exits for heavyweight processing
- Do not want a KVM specific interface
  - Generic interface = more users = less bugs

Events - eventfd

- Generalized kernel mechanism for signalling events
- Based on file descriptor, so can pass around
- Either a kernel task or a user task may signal...
- ... and either a kernel task or user task may wait for ...
- ... using any of the wait APIs

Control flow - events



Events

- Users
  - Vhost family – in-kernel virtio
  - Shared memory – guest-to-guest wakeups
  - Device assignment with vfio
- Needed changes to eventfd
- Merged in 2.6.32

User return notifiers

- Part of guest/host state is syscall/sysenter MSRs
- Save/restore on preempt notifiers, before exit to userspace
- In host, only used when returning to userspace or entering kernel
  - Task switch to kernel thread triggers unneeded save/restore
  - Task switch to guest with same values trigger unneeded save/restore
- Make it even lazier!

User return notifiers - implementation

- Invoke callback just before return to userspace
  - Only if guest state is loaded
  - Save guest state, load host state
- Issue: real hot path (syscall)
- Piggyback on signal check
  - Zero impact on not-taken path – just a mask change
- Merged 2.6.33

Lazy FPU

- FPU saved/restored on switch to other thread
- Or return to userspace (which may not touch FPU)
- Extend kernel lazy FPU to support KVM
  - Problem: kernel lazy FPU is only lazy in one direction
  - When switching out of a task, FPU is eagerly saved
  - Problem: FPU code is very old
  - Introduce FPU API
  - Problem: really lazy FPU requires an IPI to save state
  - May be slower!
- Work in progress

Conclusions

- Integrating Linux and KVM key to success
- Must be done in a way the benefits, or at least does not harm, core kernel
- lkml sometimes less friendly than kvm@vger
- Need persistence and care

Questions

?