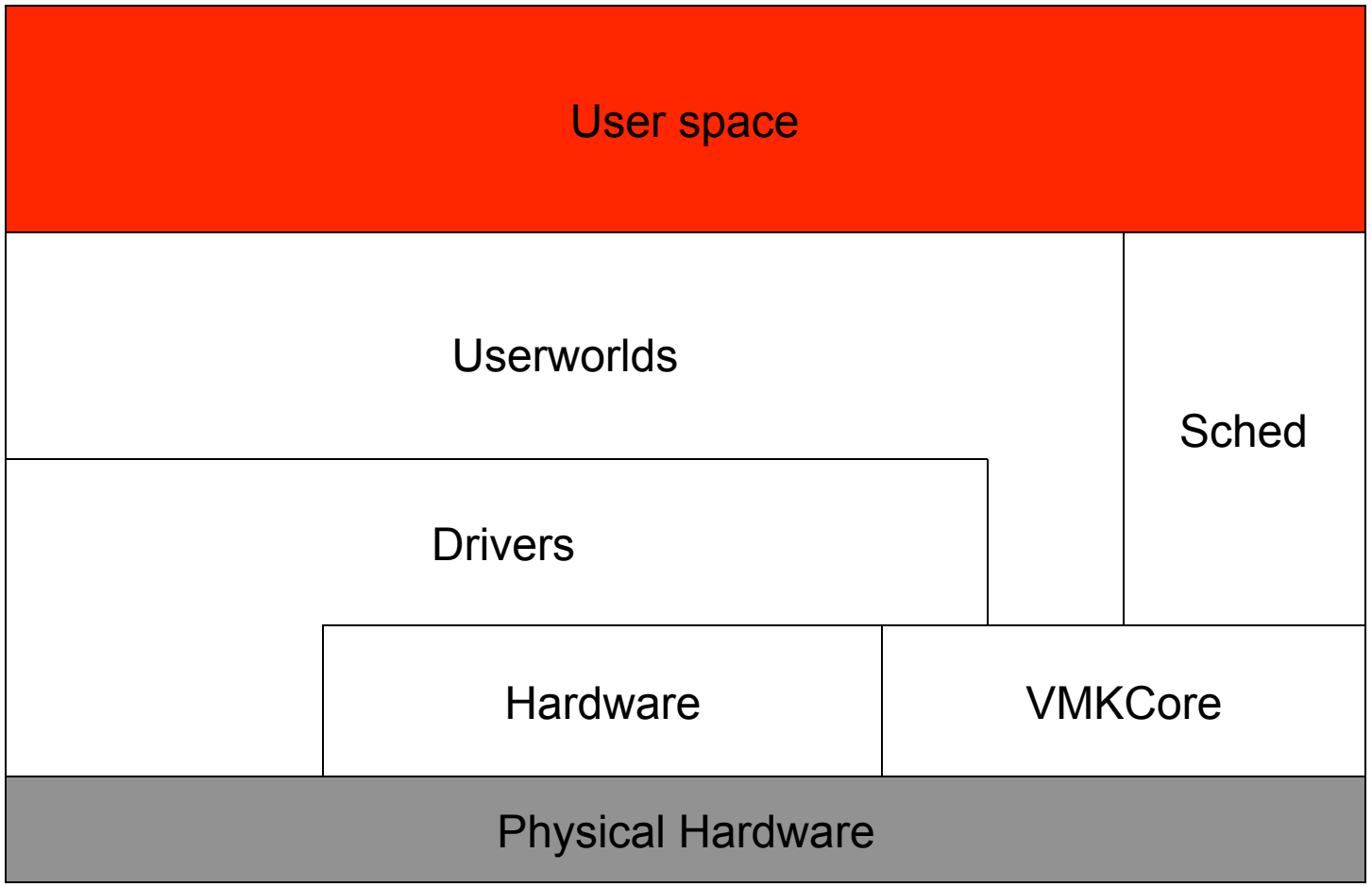
VMKernel Overview

Dan Arai

Outline

- Vmkernel basics
- Worlds
- Address space
- Vmkcalls and usercalls
- Synchronization
- Resource Management

Kernel block diagram



Additional kernel components

- Storage stack
- Networking stack
- Filesystems
- Modularity

Main kernel concepts

- Worlds (threads)
- Non-preemptive programming model
- Multiple heaps
- Ranked locks (deadlock prevention)
- System calls
- VMM -> kernel calls

Worlds

- Schedulable entity in the kernel. Comparable to threads on most OSes.
- CPU state – RIP, GPRs, SegRegs, FPU, DR, CR, EFER, RFlags, etc.
- All worlds (mostly) share one kernel AS
- VMM worlds have a private VMM AS
- Userworlds share a user AS with other threads in the same process

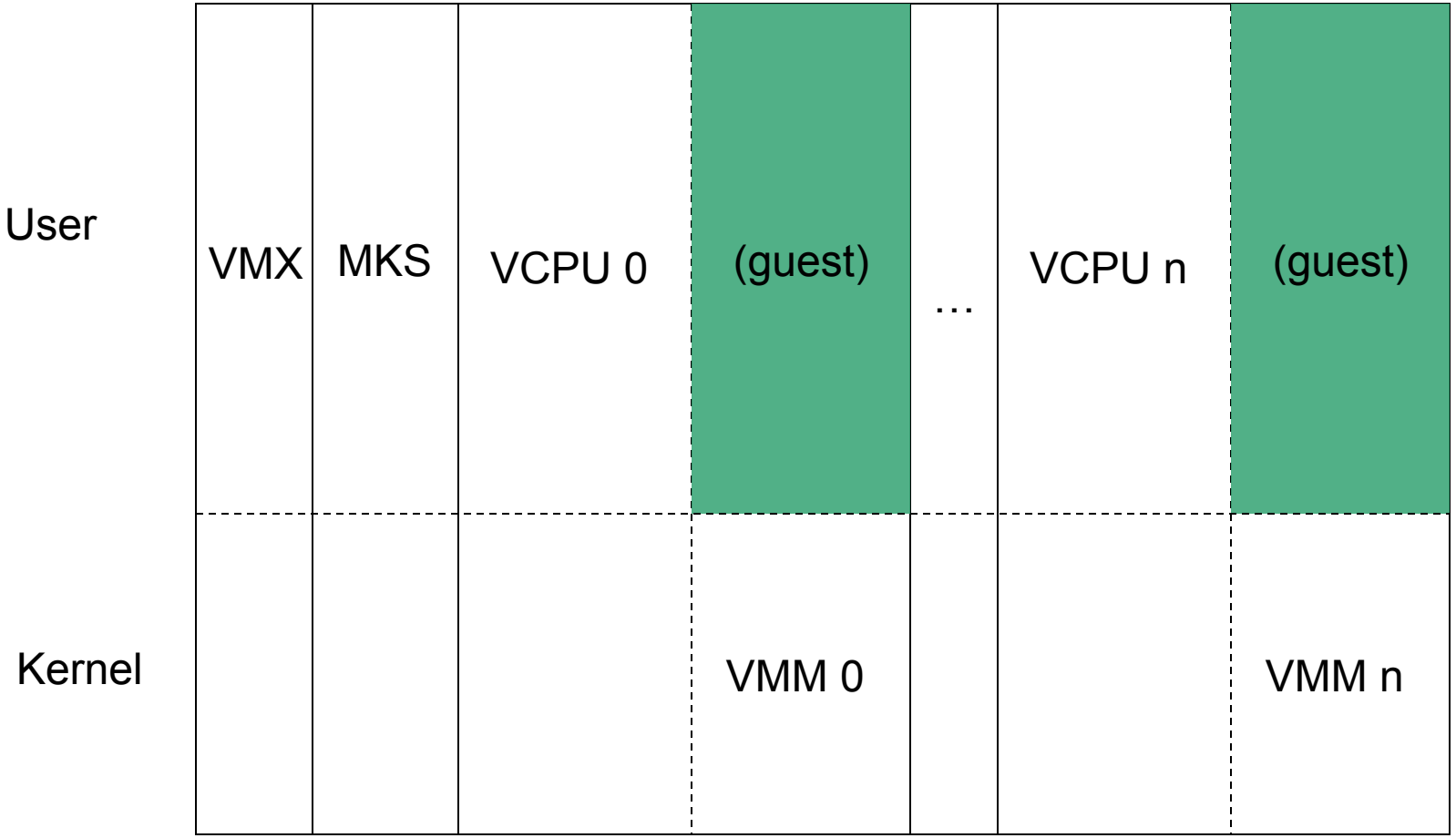
World Taxonomy

- Userworlds
- VMM worlds
- System worlds
- VM Assistant Worlds (swap/pshare)
- Helper worlds
- Idle worlds

Userworlds

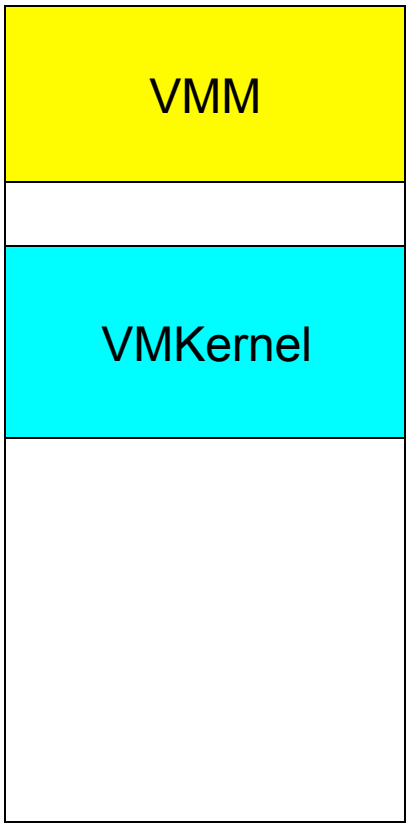
- Posix-like environment
- Modified glibc
- Linux-like system calls
- Native system calls

Anatomy of a VM on ESX

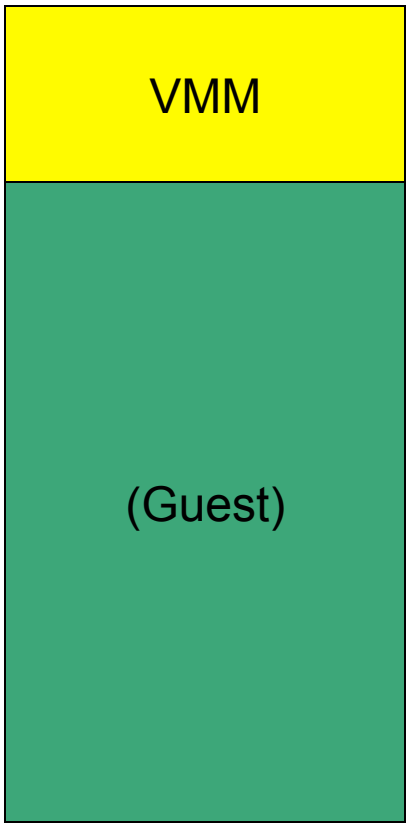


VMM and Kernel Addres Space

Scratch AS Enabled



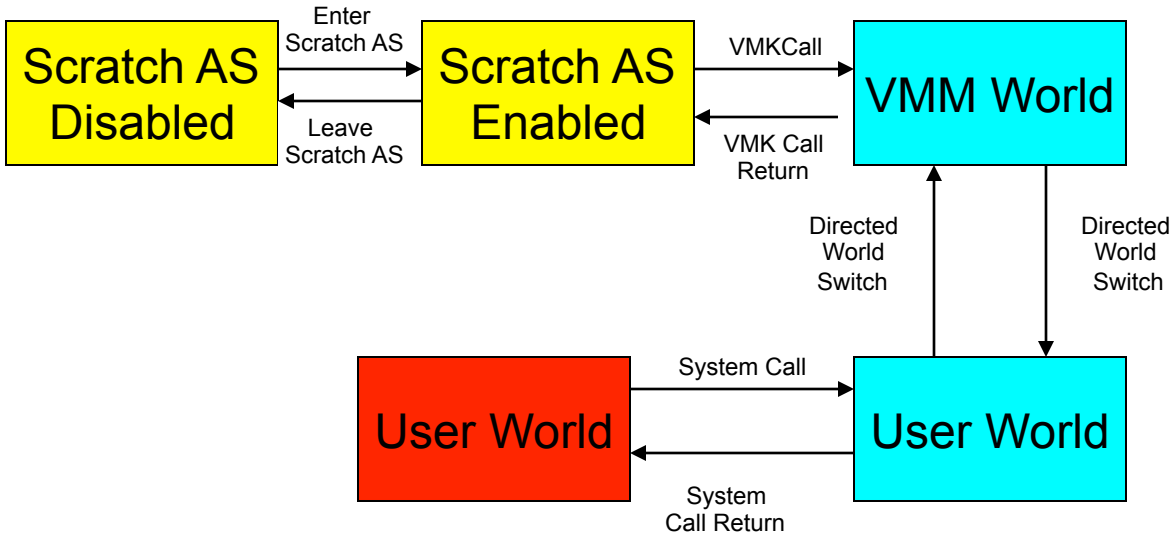
Scratch AS Disabled



VMKernel Calls

- (Switch to Scratch AS)
- Called function passed in shared area
- Function params passed as varargs
- Interrupts/NMIs disabled
- Stack, IDT, GS switched (GDT shared)
- VMM may be descheduled during any call

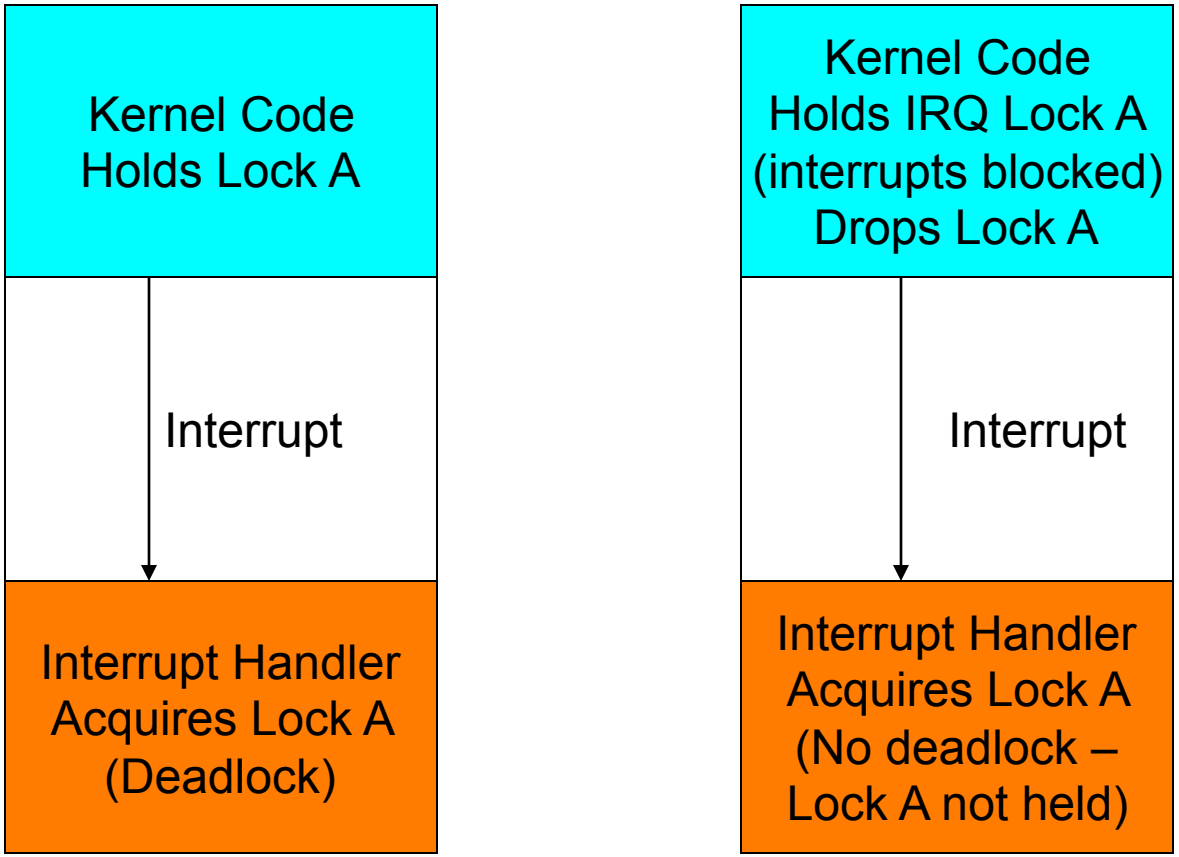
User Call Path



Synchronization Primitives

- Spinlocks (normal and IRQ)
- Semaphores (binary and counting)
  - Deadlocks are prevented by mandatory lock ranking, enforced on debug builds
  - Locks and semaphores support try operation

IRQ Locks Prevent Self-Deadlock



Resource Scheduling

- Primarily memory and CPU
- Proportional-share, with mins and maxes
- Hierarchical
- SMP VCPUs co-scheduled