



**Beyond kvm.ko**

**Avi Kivity**

**[avi@qumranet.com](mailto:avi@qumranet.com)**

---

**KVM Developers Forum**  
**June 2008**

# Agenda

- ◆ Large pages
- ◆ Containers & Isolation
- ◆ Scheduling
- ◆ Swapping
- ◆ Storage
- ◆ Conclusions

# Large pages

- ◆ 4KB page tables consume memory
  - ◆ 2MB per 1GB RAM
  - ◆ Leads to cache pressure on TLB intensive workloads
  - ◆ One TLB-induced cache miss per random access
- ◆ NPT/EPT make the problem worse
  - ◆ 4MB per 1GB
  - ◆ Two TLB-induced cache misses per random access

## Shadow memory access penalties

| Guest pages | Host pages | Cache misses |
|-------------|------------|--------------|
| 4KB         | Any        | 2            |
| 2MB         | 4KB        | 2            |
| 2MB         | 2MB        | 1            |

## 2DP memory access penalties

| Guest pages | Host pages | Cache misses |
|-------------|------------|--------------|
| 4KB         | 4KB        | 3            |
| 4KB         | 2MB        | 2            |
| 2MB         | 4KB        | 2            |
| 2MB         | 2MB        | 1            |

# Large pages (cont)

- ◆ Easy solution: use large host pages to back guest memory
  - ◆ Just 4KB per 1GB (doubled for xPT)
- ◆ New problems
  - ◆ Provisioning: hard to configure Linux for large pages
    - ◆ Okay for dedicated virtualization host
  - ◆ Doesn't swap
    - ◆ Kills overcommit
  - ◆ Won't balloon

# Fixing large pages

- ◆ Fixes

- ◆ Memory defragmentation
- ◆ Transparent coalescing/fragmentation of large pages
- ◆ Large page swapping (?!)
- ◆ Large page ballooning

- ◆ Problems

- ◆ Opposition from Linus
- ◆ Will increase core VM complexity

# Containers & Isolation

- ◆ Reduce the impact of one guest on others
- ◆ Scheduler groups
  - ◆ Treat a group of tasks as a unit for the purpose of allocating resources
- ◆ Scheduler caps and guarantees
  - ◆ Allow SLAs instead of best effort
- ◆ Memory containers
  - ◆ Account each page to its container
  - ◆ Allows preferentially swapping some guests
- ◆ I/O accounting
  - ◆ Each I/O in flight is correctly accounted to initiating task
    - ◆ Including swap activity!
  - ◆ Important for I/O scheduling
  - ◆ Important for troubleshooting

# Scheduling

- ◆ Gang scheduling
  - ◆ Schedule a guest iff there are processors available for all vcpus
  - ◆ Prevents spinning in spinlocks, IPIs, or other busy-wait scenarios
- ◆ Paravirtualized spinlocks, IPIs
  - ◆ Guest tells host when it spins
  - ◆ Host can reallocate resources

# Swapping & overcommit

- ◆ Ballooning is too simplistic
  - ◆ Host depends on guest ability to free memory
  - ◆ What if the guest is slow? Or hung? Or malicious?
- ◆ Swapping is too slow
  - ◆ Host estimate of which page to swap may be inaccurate
  - ◆ Always need to write out data
    - ◆ Even if the the guest can recreate it
  - ◆ Guest hangs when paging in data



# Swapping fixes

- ◆ Simple fix: don't swap out zeroed pages
- ◆ Complex fix: guest/host cooperation
  - ◆ Guest tells host which pages need not be saved
  - ◆ Host tells guest which pages were not saved
  - ◆ Host tells guest which pages are not present
- ◆ Can steal most from s90
- ◆ Problem: some of this is incredibly complex

# Storage

- ◆ Many similar guests cause a lot of duplicate storage
- ◆ Current solution: baseline + delta images
- ◆ Delta images only a partial solution
  - ◆ Deltas degrade over time
  - ◆ Needs planning
    - ◆ Won't work when P2Ving an existing installation
  - ◆ Disk-in-file is overheardy

# Storage fixes

- ◆ Block-level deduplication
  - ◆ Filesystem or block device looks for identical blocks
  - ◆ ... and consolidates them
  - ◆ Can be done as a background task
  - ◆ Btrfs seems well prepared
    - ◆ Reverse mappings
    - ◆ Snapshots
- ◆ Hostfs + file-based deduplication
  - ◆ No more virtual block device
  - ◆ Guest filesystem is a host directory
  - ◆ Host can carry out file dedup in the background
  - ◆ Requires changes in guest

# Conclusions

- ◆ A lot of work remains besides the core hypervisor
- ◆ Much to be done on the host level
- ◆ Some on the guest level
- ◆ Having the host features useful for non-virtualization workloads will be important for acceptance
- ◆ We won't be out of work anytime soon



# Thank You

---