CFS-v: I/O Demand-driven VM Scheduler in KVM

Hyotaek Shim and Sung-Min Lee

(hyotaek.shim, sung.min.lee@samsung.com)

Software R&D Center, Samsung Electronics

2014. 10. 16
I/O bound VMs are often co-located with computing bound VMs on the same physical machine.

- To avoid significant performance interference from computing-and-computing VMs or I/O-and-I/O VMs.

TRACON: Interference-Aware Scheduling for Data-Intensive Applications in Virtualized Environment (SC ‘11)
Understanding Performance Interference of I/O Workload in Virtualized Cloud Environments (CLOUD ‘10)
Problem in Server Consolidation

- I/O performance is still interfered with co-running computing bound VMs
  - Also when computing and I/O tasks coexist on the same VM
- **I/O performance interference by computing tasks**
  - **Mixed (4 VMs):** four VMs, each of which has 8 I/O tasks and 1~4 comp. tasks
  - **Separate (2 VMs):** VM1 (32 I/O tasks), VM2 (16 Comp. tasks)

**Performance Evaluation of CFS in KVM**

- **No Interference without computing tasks**
- **No Degradation at all regardless of computing threads**
• CPU utilization of VMs (Mixed) and Native Linux
In native Linux CFS, I/O-bound tasks could repeatedly preempt computing-bound tasks while consuming a small time slice.

What if an I/O bound task repeatedly fail to preempt the current task?
I/O Path of Virtio-blk

- Five different kinds of tasks are involved for handling a virtio-blk I/O request
  - Each task may be woken up just before handling their job (totally, up to five wake-ups)
Problem of CFS on VM scheduling

- In CFS group scheduling, VMs are deployed in different groups (runqueues)
  - Group entity of woken-up I/O related QEMU threads often fail to preempt the group entity of computing VCPUs
Preemption Rule of CFS

• Vruntime Compensation
  – When a schedule entity is woken up, the entity’s Vruntime is reset to “minimum Vruntime in its runqueue - threshold(e.g., 12000000)”
  – The reset Vruntime is not small enough to preempt the current computing task

• Weight and Grain
  – Schedule entity for a group of I/O bound VM tasks has a relatively small Weight due to CPU consumption, which results in large Grain

If woken-up sched entity’s Vruntime is smaller than Current sched entity’s Vruntime + Grain, then preempt the current task!

Grain is calculated by Weight of the woken-up
CFS-v: Boosting I/O Related QEMU Tasks

- CFS-v detects I/O bound VCPU tasks by using a 2-bit I/O count
  - If a **woken-up** task consumes CPU less than **500us**, increases by 1
  - If it consumes time slice more than **1ms**, the count decreases by 1
  - The count is larger than or equal to 2, the VCPU is “I/O related”
- **Main loop**, **worker**, and **softirq** threads are considered “I/O related”
- A **woken-up** “I/O related” task **can always preempt** non “I/O related” tasks
CFS-v: Partial Boosting

- VCPU that has both I/O and computing tasks is not considered as I/O bound and thus cannot be boosted with the previous boosting algorithm.
- What if CFS-v boosts such a VCPU?
  - The VCPU will not return CPU shortly due to computing tasks.

![Diagram showing the CFS Runqueue and VCPUs with I/O and computing tasks.]
CFS-v: Partial Boosting

- Even for a non “I/O related” VCPU, if it receives “Vring Interrupt” or “RESCHEDULE IPI,” it is partially boosted within **200us** time slice.
  - After a timer interrupt, the partially-boosted VCPU returns CPU.
Performance Evaluation

• Evaluation Environment
  – Host: Intel(R) Core(TM) i7-2600 CPU 3.40GHz (8 Cores), 12GB DRAM, kernel-3.16.1
  – Guest VMs: kernel-3.16.1, 8 Cores, 512MB DRAM
    » Running on a QCOW2 image, sharing a hard disk
    » Storage benchmark on RAW, sharing a Samsung 256GB SSD 840 pro
  – QEMU 2.1, Virtio-blk
  – We modified the host kernel and QEMU to implement CFS-v

• Micro benchmark
  – Storage I/O benchmark
    • 4KB O_DIRECT mode random read
    • Host kernel’s page cache is also disabled
  – Computing benchmark
    • MD5 Hashing
Performance Evaluation

- **Separate 2VMs**: co-running 2 VMs that execute I/O and computing workloads, respectively
  - VM1: 32 random read threads
  - VM2: 16 MD5 hashing threads
- CFS-v improves the storage I/O performance of VM1 by **764%**, while reducing the computing performance of VM2 by **30.1%**
  - CFS-v has also raised the CPU util. of VM1 from 46% to 273%
Performance Evaluation

- **Mixed 2VMs:** co-running 2 VMs, each of which is running 16 random read threads and 8 MD5 hashing threads.

- CFS-v improves the storage I/O performance by 93% and reduces the computing performance by only 9%.

![Graphs showing performance comparison between CFS and CFS-v](image-url)
Summary

• CFS-v enhances the overall resource efficiency of server consolidation
  – Enables KVM to detect I/O related QEMU and system tasks and to boost them ahead of computing tasks
  – By using partial boosting, CFS-v can also timely and momentary boost I/O bound tasks inside a VCPU running mixed workloads
  – CFS-v highly improves the I/O performance of VMs in exchange for little degradation on the computing performance