Tencent Cloud

Towards a More Efficient Synchronization in KVM

KVM FORUM 2021

Wanpeng Li
wanpengli@tencent.com

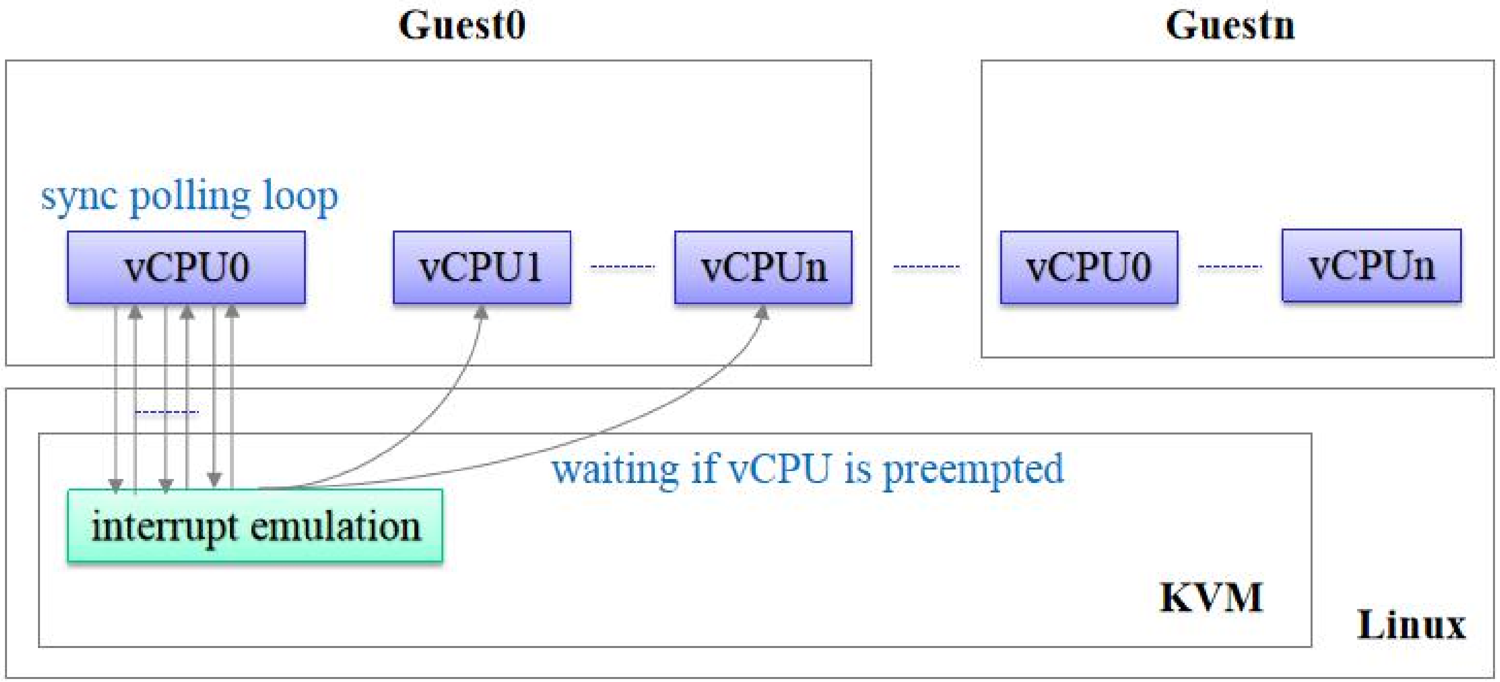
Tencent Cloud

Agenda

- Boost Preempted vCPU in user-mode
- vCPU stack by wake-affine
- RCU-Reader Preemption Problem

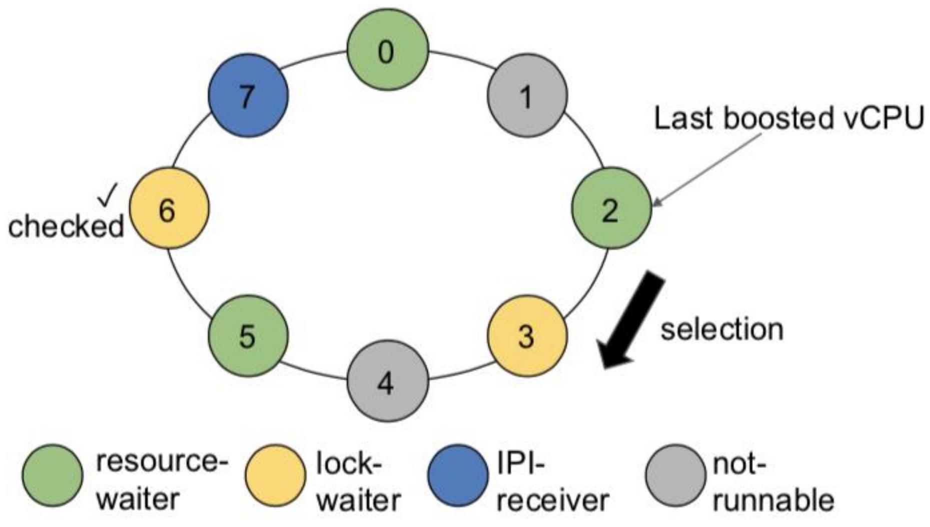
Boost Preempted vCPU in user-mode

- Synchronization based on “Busy-waiting”
  - Unnecessary CPU consumption by busy-waiting for a descheduled vCPU
    - Significant performance degradation
  - Semantic gap
    - OSes assume their vCPUs are dedicated as pCPUs



Boost Preempted vCPU in user-mode

- Most smp_call_function_many calls are synchronous, mainly TLB Flush and "Function Call interrupts"
- Both the lock holder and IPI target vCPU are yield candidates



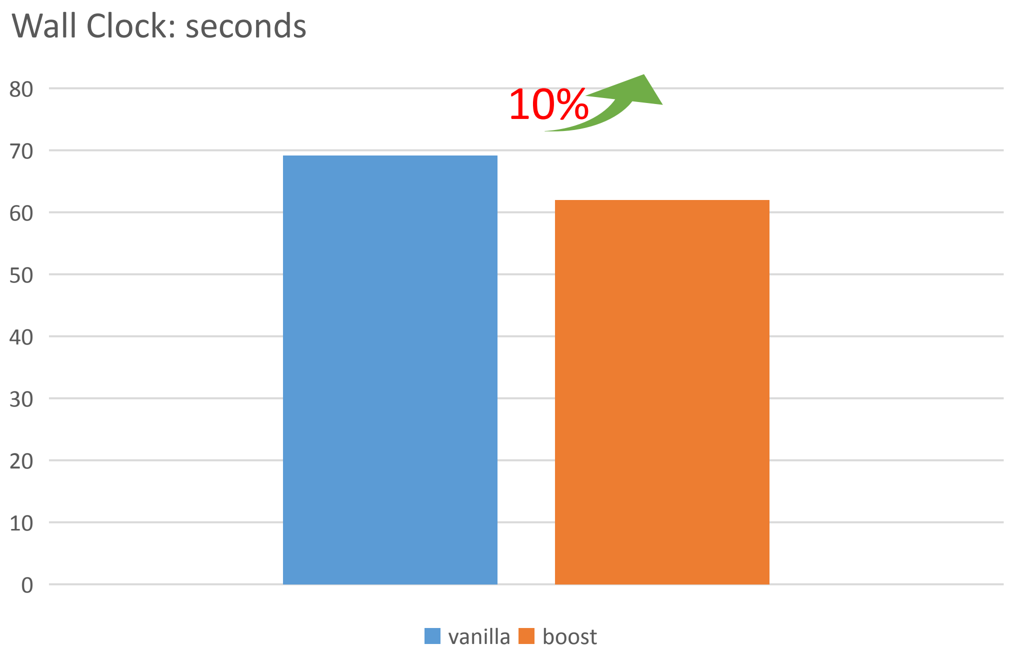
Tencent Cloud

Boost Preempted vCPU in user-mode

- Intel PLE occurs when the spinlock waiter is in kernel-mode
  - IPI receiver can be in either kernel or user mode.
  - IPI receiver candidate in user-mode fails to be boosted
- Workloads like pbzip2 do the TLB shootdown in kernel-mode and most of the time they are running in user-mode.
  - It can lead to a large number of continuous PLE events
    - IPI sender causes PLE events repeatedly until the receiver is scheduled while the receiver is not candidate for a boost.

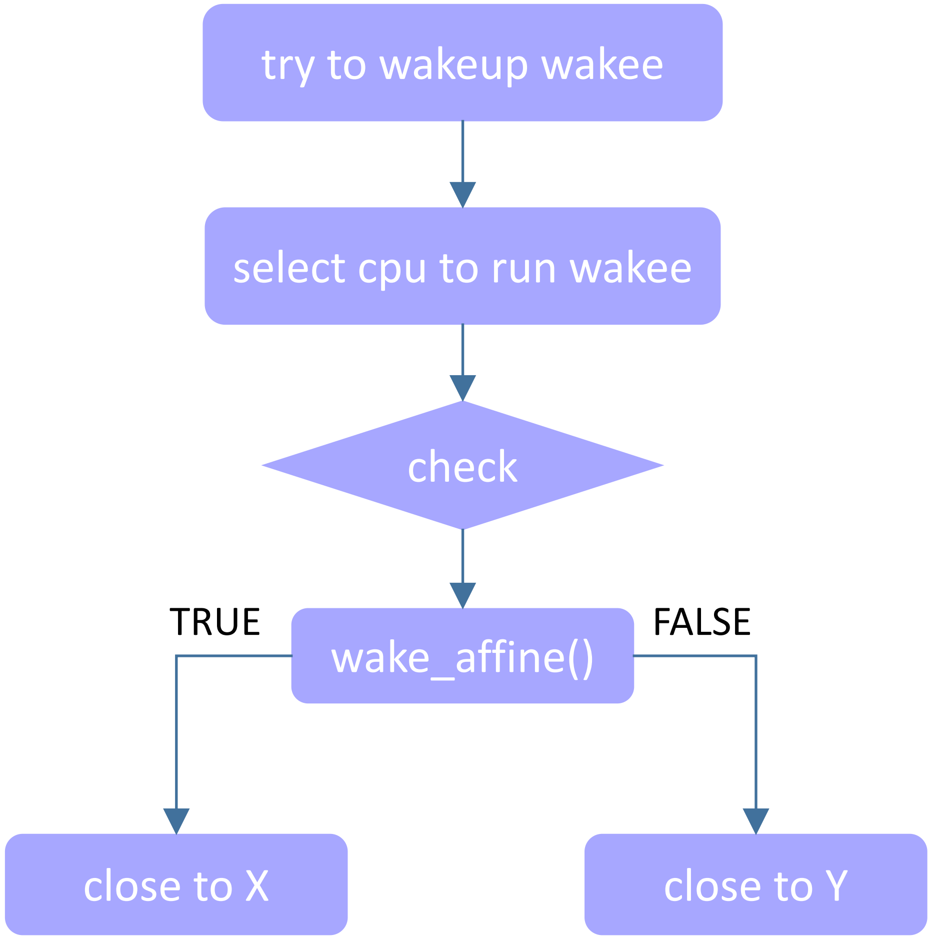
Boost Preempted vCPU in user-mode

- Let's boost the vCPU candidate in user-mode which is delivering interrupt
- Evaluation Environment
  - Hardware: Intel CLX, 2 socket, 48 cores, 96 HTs
  - VM: 96 vCPUs
  - Test case: pbzip2



vCPU stack by wake-affine

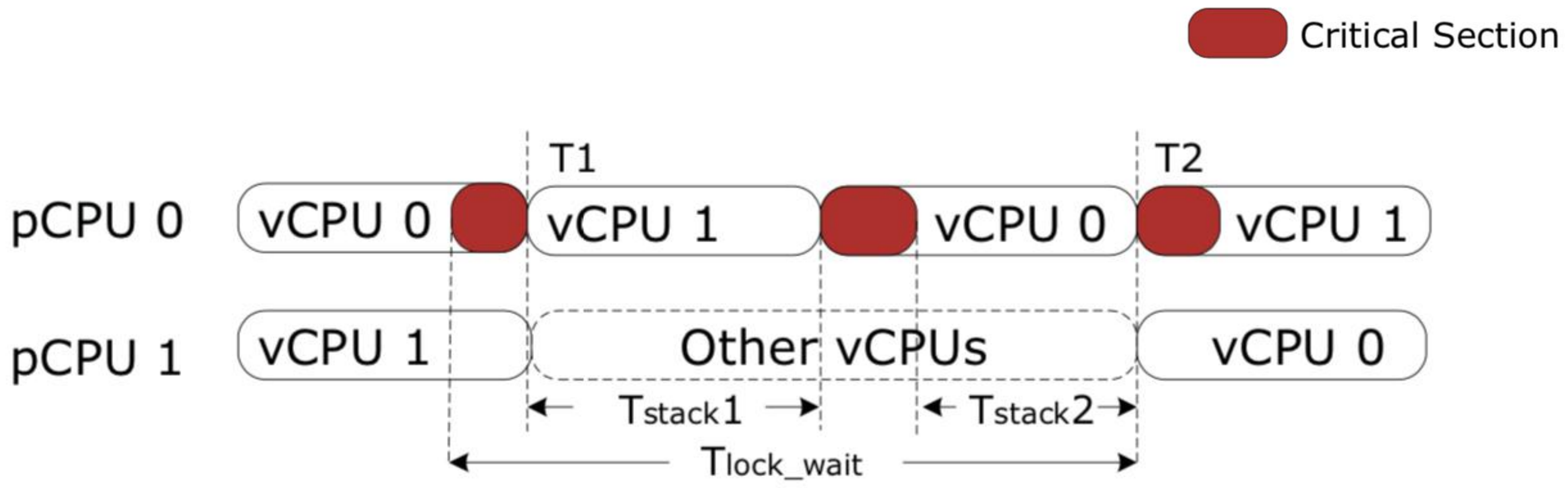
- Wake-affine is a feature inside scheduler which we attempt to make processes running closely, it gains benefit mostly from cache-hit.
- When qemu/other vCPU inject virtual interrupts to guest through waking up one sleeping vCPU, it increases the probability to stack vCPUs/qemu by scheduler wake-affine.



When:
1. waker is currently running on CPU X
2. wakee was last time running on CPU Y

vCPU stack by wake-affine

- A scheduler allows vCPUs to be scheduled on any pCPUs. This will cause the vCPU stacking problem that the lock waiter is scheduled before the lock holder on the same pCPU.



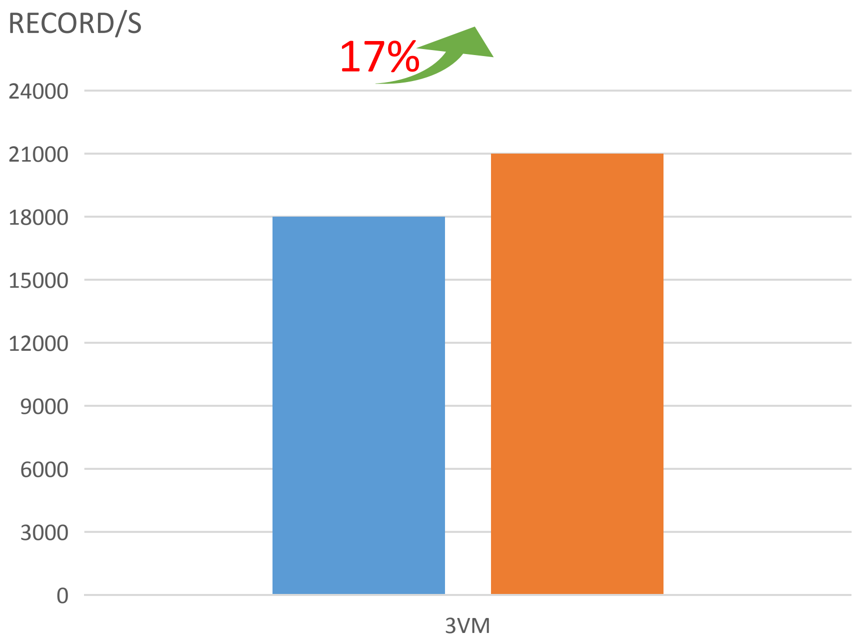
vCPU stack by wake-affine

- How often does scheduler stack vCPUs?
  - Run 4-vCPU VMs on 4-CPU physical machine
- Run the CPU-bound workload inside the VMs
  - 100% utilization on each vCPU

#VMs	≥ 2 vCPU siblings stacking on the same CPU
1	5.564%
2	**43.127%**
3	**45.932%**

vCPU stack by wake-affine

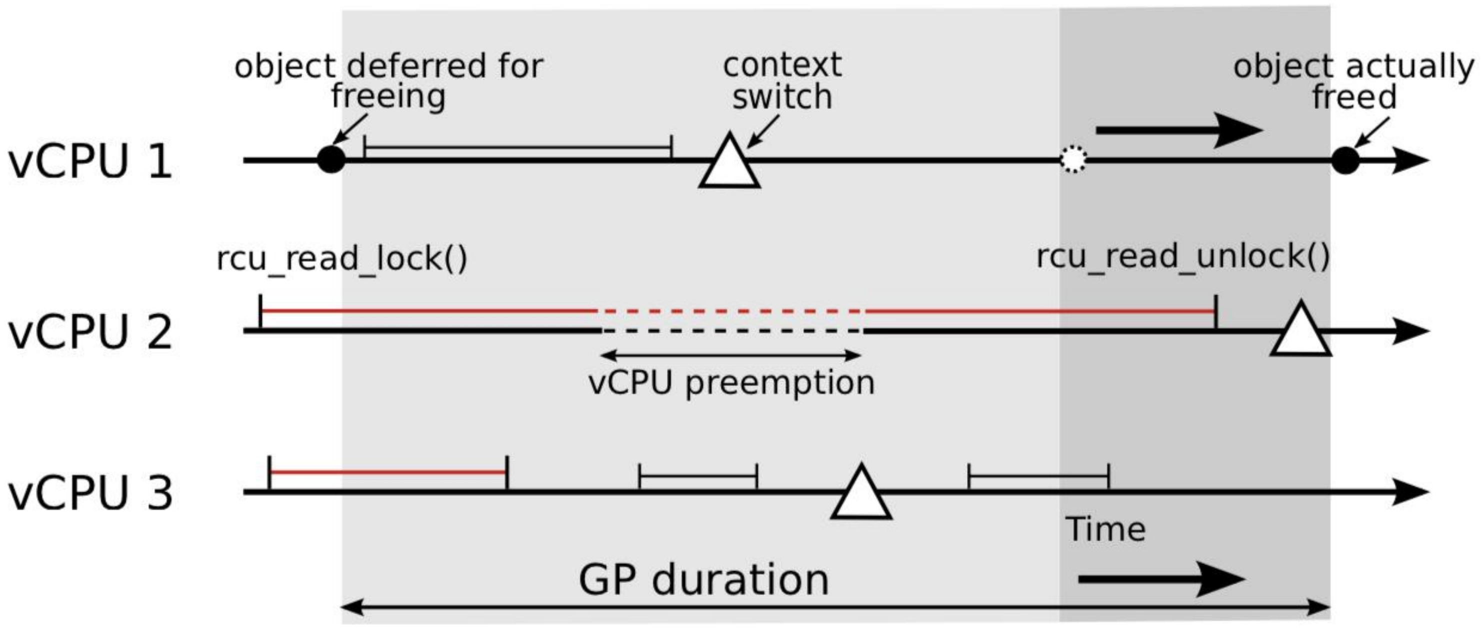
- Let's disable wake-affine vCPU process to mitigate lock holder preemption
- Evaluation Environment
  - Hardware: Intel SKX, 2 sockets, 40 cores, 80 threads
  - VM: 80 vCPUs
  - Test case: ebizzy -M



Tencent Cloud

RCU-Reader Preemption

- RCU GPs cannot complete while a vCPU is preempted within an RCU read-side critical section. Guest OS invoking synchronize_rcu() can incur latency spikes from several seconds on overcommitted hosts.

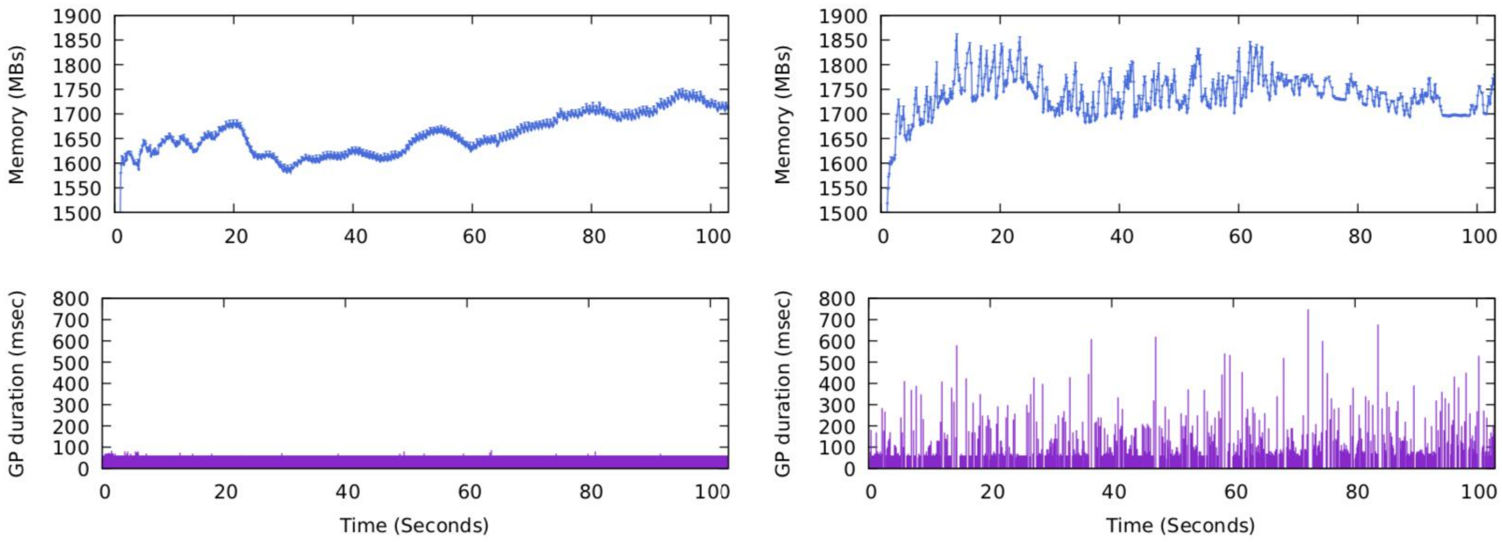


RCU-Reader Preemption

- Although calls to call_rcu() continue to return immediately, their callbacks cannot be invoked.
- Linux-Kernel code can therefore continuously invoke call_rcu(), GP delay due to vCPU preemption can cause transient memory-footprint spikes, frequent transient memory-footprint spikes can scatter the kernel pages through the system, which can increase external memory fragmentation.

RCU-Reader Preemption

Evaluation: Postmark



Baseline

Overcommit

26.37× increase in max grace period duration
2.18× increase in the average grace period duration
2.9× increase in CPU consumed per grace period computation

Tencent Cloud

Reference

- https://lore.kernel.org/kvm/1618542490-14756-1-git-send-email-wanpengli@tencent.com/
- https://lore.kernel.org/kvm/1564479235-25074-1-git-send-email-wanpengli@tencent.com/

Tencent Cloud

Q/A?