The RCU-Reader Preemption Problem in VMs

Aravinda Prasad\textsuperscript{1}, K Gopinath\textsuperscript{1}, Paul E. McKenney\textsuperscript{2}

\textsuperscript{1}Indian Institute of Science (IISc), Bangalore
\textsuperscript{2}IBM Linux Technology Center, Beaverton
Read-Copy-Update (RCU)

- RCU is a highly scalable synchronization technique
- RCU Readers
  - Do not directly synchronize with writers
  - Read-side primitives are exceedingly lightweight
Read-Copy-Update (RCU)

- RCU is a highly scalable synchronization technique
- RCU Readers
  - Do not directly synchronize with writers
  - Read-side primitives are exceedingly lightweight

```c
/* non-preemptible kernels */
rcu_read_lock()
{
    /* no-op !! */
}

rcu_read_unlock()
{
    /* no-op !! */
}
```
Read-Copy-Update (RCU)

- RCU is a highly scalable synchronization technique
- RCU Readers
  - Do not directly synchronize with writers
  - Read-side primitives are exceedingly lightweight
    
    ```c
    /* non-preemptible kernels */
    rcu_read_lock()
    {
        /* no-op !! */
    }
    rcu_read_unlock()
    {
        /* no-op !! */
    }
    ```

- RCU Writers
  - Must guarantee consistent view of data structures to readers
Example: Linked List Delete Operation
Example: Linked List Delete Operation

A → B → C

Removed object B is reclaimed after a grace period.
Example: Linked List Delete Operation

A → B → C

A → B

Grace Period

Readers
Example: Linked List Delete Operation

Removed object B is reclaimed after a grace period.
Example: Linked List Delete Operation

Removed object B is reclaimed after a grace period
RCU Grace Periods (Non-Preemptive Environment)

- Restriction on RCU readers:
  1. Referencing an object outside the read-side critical section is not allowed
  2. Blocking/sleeping/yielding is not permitted within a read-side critical section
     (same rule as for tasks holding spinlocks)
RCU Grace Periods (Non-Preemptive Environment)

- Restriction on RCU readers:
  1. Referencing an object outside the read-side critical section is not allowed
  2. Blocking/sleeping/yielding is not permitted within a read-side critical section
     (same rule as for tasks holding spinlocks)

- A context switch on a CPU implies all readers on that CPU are done
- Grace period ends after all CPUs execute a context switch
The RCU-Reader Preemption Problem

Preemption of vCPUs executing RCU read-side critical sections
The RCU-Reader Preemption Problem

Preemption of vCPUs executing RCU read-side critical sections

*Grace periods cannot complete while a vCPU is preempted within an RCU read-side critical section*
The RCU-Reader Preemption Problem

vCPU 1

object deferred for freeing

context switch

object actually freed

rctu_read_lock()

vCPU 2

rctu_read_unlock()

vCPU 3

GP duration

Time
The RCU-Reader Preemption Problem

vCPU preemption

Time

vCPU 1

object deferred for freeing

context switch

object actually freed

rcu_read_lock()

rcu_read_unlock()

vCPU 2

vCPU preemption

vCPU 3

GP duration

Time

GP duration

vCPU 1

vCPU 2

vCPU 3
Evaluation 1: Postmark

Baseline

Overcommit
Evaluation 1: 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
Evaluation 2: Memory microbenchmark

Baseline

Overcommit

$3.62 \times$ increase in max grace period duration

$30.26\%$ increase in the average grace period duration

$\sim 50\%$ increase in peak memory footprint
Impact

- **Latency**: spikes when synchronously waiting for grace periods

- **Memory**: footprint spikes and increased peak memory footprint
  - Increased fragmentation
  - Can trigger swapping and ballooning

- Increased **CPU utilization**

- **Cross-VM interaction**: CPU-consumption spike in one VM might cause a grace period duration spike in another VM
Impact

- **Latency**: spikes when synchronously waiting for grace periods

- **Memory**: footprint spikes and increased peak memory footprint
  - Increased fragmentation
  - Can trigger swapping and ballooning

- Increased **CPU utilization**

- **Cross-VM interaction**: CPU-consumption spike in one VM might cause a grace period duration spike in another VM

*RCU-reader preemption can impact VM density and consolidation*
Summary

- First evaluation of vCPU preemption within RCU readers
- Demonstrate that RCU-reader preemption has significant performance impacts
- Techniques to handle lock-holder preemption cannot be applied directly to RCU
- Currently investigating a holistic solution for the RCU-reader preemption problem
• This work represents the view of the author and does not necessarily represent the view of IBM.
• IBM and IBM (logo) are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries.
• Linux is a registered trademark of Linus Torvalds.
• Other company, product, and service names may be trademarks or service marks of others.
Thank you!!

Questions?
The RCU-Reader Preemption Problem in VMs

**Aravinda Prasad**¹, K Gopinath¹, Paul E. McKenney²

¹Indian Institute of Science (IISc), Bangalore
²IBM Linux Technology Center, Beaverton