Backporting of KVM TDP MMU in openEuler

Outline

- Problems of legacy KVM MMU
- KVM TDP(Two-Dimensional Paging) MMU
- Backporting of TDP MMU in openEuler

Problems of legacy KVM MMU

Requirement of on current demand paging on multiple vCPUs, to

- allocate guest memory;
- construct mappings in EPT;
- provide dirty logging / huge page splitting etc.

Requirement of emulation, to provide

- guest page table walker;
- guest #PF injection;
- page writes tracking, e.g. KVMGT etc.

Problems of legacy KVM MMU

Requirement of interoperation between KVM & Linux memory management:

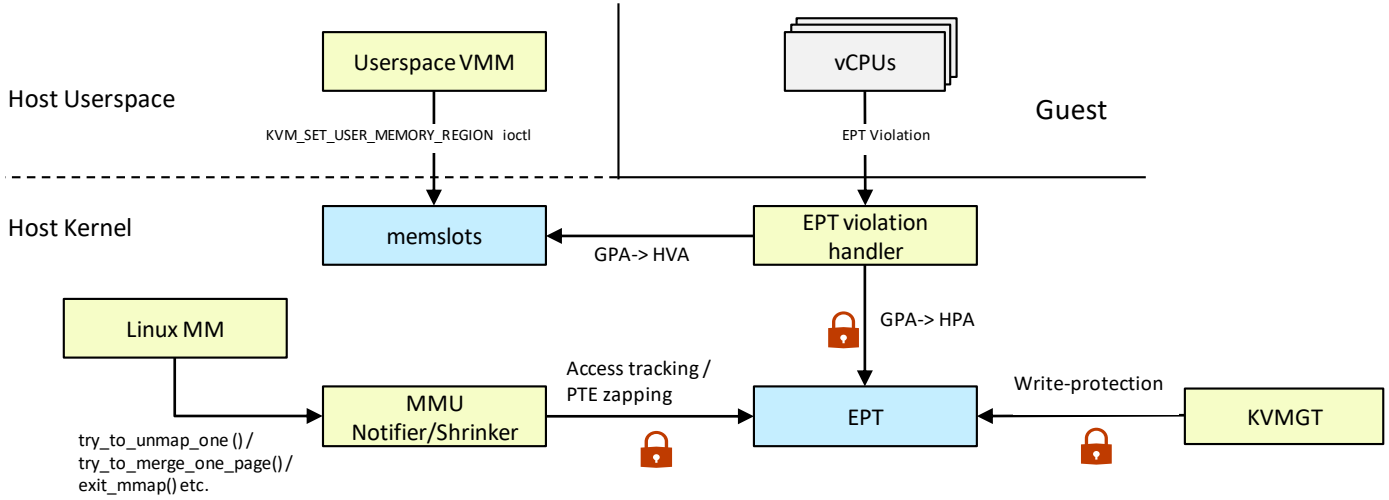
- MMU notifiers – to track accessed status and to reclaim guest memory

invalidate_range invalidate_range_start invalidate_range_end	Free host physical page.
change_pte	Host PTE changed (host page migrated/merged/write-protected etc.).
clear_young clear_flush_young test_young	Check if a page has been accessed, optionally clear accessed status / flush TLB.

- MMU shrinker – to reclaim EPT paging structures.

All operations on EPT/TLB require synchronization.

Problems of legacy KVM MMU



A monolithic spinlock(kvm->mmu_lock) is used for synchronization.

Problems of legacy KVM MMU

Lock contention issue for the monolithic lock:

- VMs are getting larger
  - Hundreds of vCPUs in one VM.
  - TBs of guest memory.
- Many vCPUs + huge memory + monolithic spinlock = lock contention.

But is the monolithic spinlock indispensable?

Other scalable issues in legacy KVM MMU:

- Reverse mapping data structure
  - 24GB host memory for 12TB VM (around 0.2%, even more in practice) .
  - maintenance of reverse mapping requires MMU lock.
- Reverse mapping is NOT necessary for EPT.

How to remove reverse mapping?

KVM TDP(Two-Dimensional Paging) MMU

Identify data structures to be protected and operations on these data structures:

- EPT PTEs
  - Mapping
  - Unmapping
  - A/D tracking
  - Write tracking
- EPT Paging structures & KVM MMU pages (descriptor of EPT paging structures).
  - Allocation
  - Reclaiming
  - Traversing
- TLBs
  - Invalidation

KVM TDP(Two-Dimensional Paging) MMU

Replace monolithic spinlock with rwlock

- Read lock for EPT violation handling (with contention detection)
- Write_lock for MMU notifier/shrinker, KVMGT page tracking etc.

Use atomic cmpxchgs to set EPT PTEs.

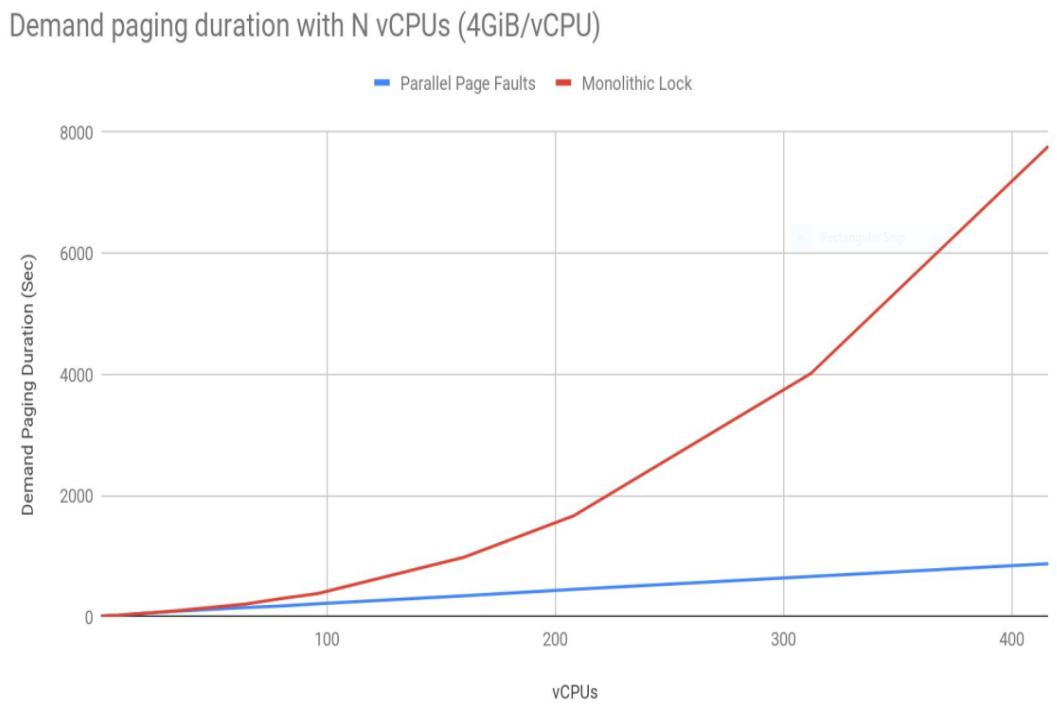
Use RCU for EPT paging structures

- rcu_read_lock() / rcu_read_unlock() for EPT traversing.
- call_rcu() to free EPT paging structures and KVM MMU pages.

KVM TDP(Two-Dimensional Paging) MMU

Decouple (partially) KVM MMU TDP support from shadow support.

- New groups of TDP iterator.
- No need to use rmap, less memory consumption.



demand_paging: a benchmark test in KVM selftest

Backporting of TDP MMU in openEuler

Major patch sets of TDP MMU (all from Google)

- New TDP iterator already introduced in Linux 5.10
  - Introduce the TDP MMU
- Basic parallel #PF handling merged in Linux 5.11
  - Allow parallel MMU operations with TDP MMU
- More optimization and bug fixes for TDP MMU after Linux 5.12. E.g.:
  - More parallel operations for the TDP MMU
  - Fix a TDP MMU leak and optimize zap all
  - Fix unsync races within TDP MMU
- Set as default KVM MMU in Linux 5.15
- Dozens of KVM MMU changes in between.
- Dozens optimizations after Linux 5.15.

Backporting of TDP MMU in openEuler

Suggestions needed:

- Should we only backport patches before Linux 5.15, or even after?
- Many KVM MMU optimizations and refactors.

Q&A