KVM Live Upgrade with Properly Handling of Passthrough Devices

Zhimin Feng
Agenda

• Background
• How to handle the passthrough devices
• Downtime Optimization
• Achievements
Background
Add security patches and new features:

- kernel live patching.
  - cannot handle complex changes
- Virtual Machine (VM) live migration.
  - incur unacceptable long delays
Background

VMM live upgrade:

- add security patches and new features.
- upgrade the whole VMM (KVM & QEMU) without interrupting customer VMs.
Interrupts handling is difficult for passthrough devices during live upgrade.

Minimizing service downtime is the major concern of cloud providers.
Handle the passthrough devices
- Divide KVM module to multiple modules.
- New QEMU inherits vfio connector from Old QEMU.
- The VM’s memory is shared by the new and old QEMU processes.
Passthrough device

Handling Passthrough device is difficult during live upgrade.

- Passthrough device cannot be suspended.
Existing Solution

- New QEMU Inherits vfio eventfds from old QEMU.
- New QEMU reads from the eventfd and receives the pending interrupts
- Inject an additional virtual irq into the VM.
Our solution - framework

- VT-d Posted-Interrupts Support.
- Pi_desc is shared between New QEMU and old QEMU.
- Interrupts is consistency between New QEMU and old QEMU.
Our solution

Alloc Pi_desc structure
allocated memory for Pi_desc structure in QEMU
Our solution

Initialize Pi_desc data

- New QEMU does not initialize the pir data.
- New QEMU does not sync the pir from Old QEMU.
- New QEMU does not update the Interrupt Remapping Table.
Downtime Optimization
Downtime optimization

Live upgrade flow diagram

- Downtime phase
  - suspend
  - save state
  - load state
  - startup
Downtime optimization

Suspend optimization (Old QEMU)

- Don't cleanup eventfds for virtio devices.

```
Suspend

kick VCPU

stop Devices

Live Upgrade Success

kill QEMU

device_stop

while (total_queues)

device_stop_one

cleanup eventfd
```
Downtime optimization

Startup optimization (New QEMU)

- Create eventfds for virtio devices during QEMU initialization.
Downtime optimization

Save/Load state:

- Using shared memory to save/load vm state.
- Loading state in the new QEMU happens concurrently with saving state in the old QEMU.
Downtime

- 8 vcpu, 16GB RAM, 2 Cloud Disk(100GB, 200GB), Mellanox Technologies MT27800.
- 64 vcpu, 128GB RAM, 2 Cloud Disk(100GB, 200GB), Mellanox Technologies MT27800.
- 88 vcpu, 350GB RAM, 2 Cloud Disk(100GB, 200GB), Mellanox Technologies MT27800.

Workload:

- idle
- cpu_stress
- memtester
- fio:
Downtime

VM workload: idle

- vcpu downtime: 11ms ~ 34ms
Downtime

VM Workload: stress -c 4

- vcpu downtime: 12ms ~ 34ms
Downtime

VM Workload: memtester 4G

- vcpu downtime: 12ms ~ 34ms
Downtime

VM Workload: fio --filename=/mnt/test.data --iodepth=1 --rw=randwrite
--bs=4k --size=40G

➢ vcpu downtime: 12ms ~ 38ms
Thank You

Contact Info: fengzhimin@bytedance.com