KVM performance tuning

Senior Staff engineer
Yang Zhang
Problems in KVM Cloud

• Alibaba Cloud: millions of VMs run in KVM

• Typical problems are observed from real scenarios

  • Idle latency

  • Timer

  • Scheduler
Idle latency

• Topic in KVM forum 2013: “KVM vs. Message Passing Throughput”

• Topic from David Matlack: Message Passing Workloads in KVM

• Cost in idle -> running and running -> idle transition is amplified in real businesses.
Data of real business scenario (java)

35% gap

- bare metal: 100,000 qps
- VM: 65,000 qps
Real Scenario

- Communication over the network

**Client**

1. Send a request to server
2. Receive the request. Send result back
3. Receive the result

**Server**

idle

idle
Where is the Overhead?

Client
1. Send a request to server
   - IPI
   - APIC Timer
   - HLT
   - APIC Timer
3. Receive the result
   - APIC Timer
   - HLT

Server
2. Receive the request. Send result back
   - IPI
   - APIC Timer
   - HLT

Overhead from:
- IPI
- APIC Timer
- HLT
Existing Solution

- Idle = Poll: Waste CPU cycle, hurt others performance (HT)
- Disable NOHZ: Not the default configuration in modern distros
- KVM halt polling: eliminate overhead of scheduler
Our Solution – smart idle poll

- Poll inside VM: poll in idle path
- Eliminate all overhead: IPI, TIMER, HLT
- Use dynamic poll to get better performance
- Change the poll time based on the prediction
Data of real business scenario (java)

![Bar chart showing qps for bare metal, VM, and smart idle poll, with a 10% gap]

- QPS for bare metal: 120,000
- QPS for VM: 90,000
- QPS for smart idle poll: 90,000

10% gap
# Fio 4k latency

| inject size | kvm halt poll=0  
|            | smart idle poll=0 |
|------------|--|------------------|----------------|----------------|----------------|
|            | kvm halt poll=100000ns |
|            | smart idle poll=0 |
|            | kvm halt poll=0  
|            | smart idle poll=100000ns |
|------------|--|----------------|----------------|----------------|----------------|
| inject 1us | 42.31us, stdev=3.54 | 31.98us, stdev=3.68 | 25.98us, stdev=3.37 | 25.95us, stdev=3.15 |
| inject 20us| 69.01us, stdev=4.79 | 53.74us, stdev=3.59 | 52.02us, stdev=3.38 | 46.94us, stdev=3.36 |
| inject 50us| 98.85us, stdev=3.66 | 84.25us, stdev=3.36 | 82.06us, stdev=9.36 | 77.41us, stdev=4.16 |
Problem with Timer

VM

Set timer → pause → running → pause → Handle timer

1.vmexit 2.vmentry 3.expire 4.inject

hypervisor

emulation

Trap to hypervisor twice

Interrupt inject
Exitless Timer

New PV timer: Exitless Timer

- Share page: share timer info and sync info
- Agent timer: set timer in hardware
Exitless Timer

- Share page:
  - Per vcpu share page between guest and kvm
  - Guest: store next timer info, read next sync info
  - KVM: set next timer in hardware, store next sync info
Exitless Timer

- Agent timer:
  - Scan share page regularly
  - Set next sync time
  - Dedicate CPU
Exitless Timer

- Timer inject:
  - Timer fire in another CPU
  - Inject interrupt thru PI: no vmexit
Timer overhead (ns)

- Bare metal: Skylake + Centos7u2
- VM: Skylake + Centos7u2
- VM (w/ exitless timer)

Latency:
- Bare metal: 2100 ns
- VM: 3500 ns
- VM (w/ exitless timer): 2300 ns
Next Plan

- Resource isolation
  - Share low level resource impact performance:
    - Cache, Memory bandwidth, PCIE bandwidth
  - RDT
Thank You