Performance Monitoring for KVM Guests

Avi Kivity
August 16, 2011

Agenda

Problem statement

Choices

Issues

Status

Future directions

Problem statement

Allow users of virtual machines to identify sources of performance problems in their guests

Types of performance problems

- Algorithmic
- Networking
- Storage
- Cache/TLB use
- SMP / NUMA
- Language runtime
- Scheduling
- Problems induced by the virtualization layer

Performance Monitoring Unit (PMU)

- Hardware component integrated into modern CPU cores
- Counts and reports architectural events
  - Clock cycles, instructions retired, cache misses...
- Counts and reports micro-architectural events
  - MEM_LOAD_UOPS_RETIRED.HIT_LFB: Retired load uops which data sources were load uops missed L1 but hit FB due to preceding miss to the same cache line with data not ready
- Tools read these counter and correlate with source code

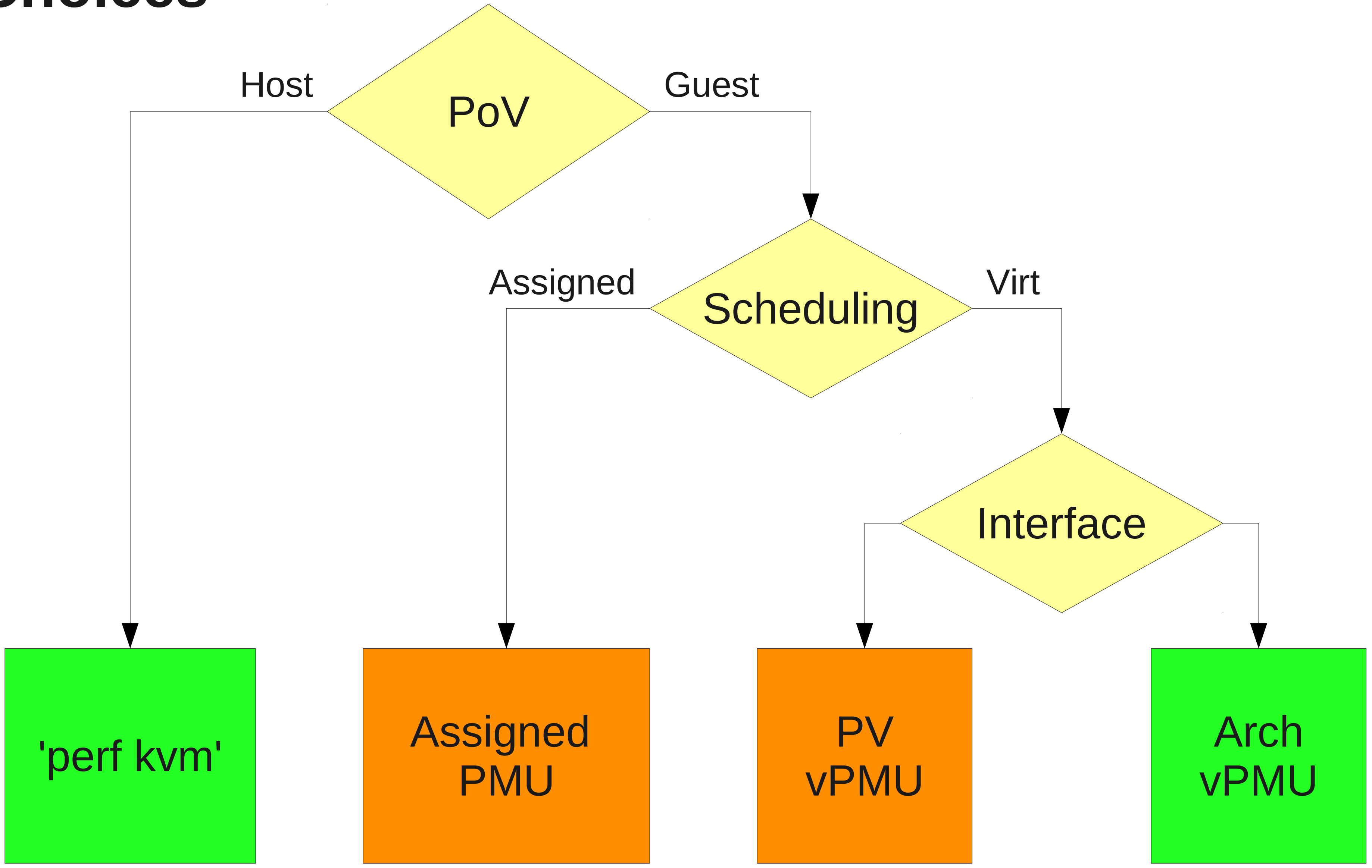
Problems with the x86 PMU

- Vendor specific, model specific
  - = virtualization-unfriendly
- Limited resource
  - Can count many things, but just a few simultaneously
- Slow to program

Architectural PMU

- Small but useful subset of events
- Programming interface fixed (“architectural MSRs”)
- Stable across processor revisions
- Discoverable via CPUID
- Intel only

Choices


Host
PoV
Guest
Assigned
Scheduling
Virt
Interface
'perf kvm'
Assigned PMU
PV vPMU
Arch vPMU

Point of view

- Host
  - See entire system
  - Multiple guests
  - Virtualization layer

- Guest
  - Existing tools and mindset
  - Integration with guest O/S and processes
  - Cloud deployment
  - Live migration

Assigned PMU vs. vPMU

PMU pass through

- Fast
- Accurate

Virtual PMU

- Secure
- Shareable
- Model independent

Interface

Paravirt

- Flexible
- Fast

Architectural

- Documented, established spec
- Compatible with existing guest software
- Compatible with future hardware improvements

Linux perf_events

- Schedules required counters across available PMU counters
- Host-wide counters
- Process counters
- Software counters
- PMU counters
  - Generic
  - Model specific

perf kvm

- Extension of perf_events subsystem to sample guests
- 'perf kvm' tool
- Merged into Linux 2.6.35

Inplementing a vPMU with perf_events

- perf_events generic counters match arch PMU 1:1
  - How convenient
- Some details don't match so well
  - CMASK
- KVM code decodes guest intent from MSR writes
  - ... and asks perf core to monitor these events
- Scheduling, programming done by perf core

Problems

- Few applications work with the architectural PMU
  - Need individual testing and qualification
- Programming the vPMU is slow
  - Can be improved with Version 2 Architectural PMU
- Linux will not try to detect Architectural PMU on AMD
  - Can be fixed

Future work

- Test & merge
- Version 2 (or 3) Architectural PMU
- Paravirt acceleration

Questions

?