The KVM halt polling system

The KVM halt polling system provides a feature within KVM whereby the latency of a guest can, under some circumstances, be reduced by polling in the host for some time period after the guest has elected to no longer run by cedeing. That is, when a guest vcpu has ceded, or in the case of powerpc when all of the vcpus of a single vcore have ceded, the host kernel polls for wakeup conditions before giving up the cpu to the scheduler in order to let something else run.

Polling provides a latency advantage in cases where the guest can be run again very quickly by at least saving us a trip through the scheduler, normally on the order of a few micro-seconds, although performance benefits are workload dependant. In the event that no wakeup source arrives during the polling interval or some other task on the runqueue is runnable the scheduler is invoked. Thus halt polling is especially useful on workloads with very short wakeup periods where the time spent halt polling is minimised and the time savings of not invoking the scheduler are distinguishable.

The generic halt polling code is implemented in:

> virt/kvm/kvm_main.c: kvm_vcpu_block()

The powerpc kvm-hv specific case is implemented in:

> arch/powerpc/kvm/book3s_hv.c: kvmppc_vcore_blocked()

Halt Polling Interval

The maximum time for which to poll before invoking the scheduler, referred to as the halt polling interval, is increased and decreased based on the perceived effectiveness of the polling in an attempt to limit pointless polling. This value is stored in either the vcpu struct:

> kvm_vcpu->halt_poll_ns

or in the case of powerpc kvm-hv, in the vcore struct:

> kvmppc_vcore->halt_poll_ns

Thus this is a per vcpu (or vcore) value.

During polling if a wakeup source is received within the halt polling interval, the interval is left unchanged. In the event that a wakeup source isn't received during the polling interval (and thus schedule is invoked) there are two options, either the polling interval and total block time[0] were less than the global max polling interval (see module params below), or the total block time was greater than the global max polling interval.

In the event that both the polling interval and total block time were less than the global max polling interval then the polling interval can be increased in the hope that next time during the longer polling interval the wake up source will be received while the host is polling and the latency benefits will be received. The polling interval is grown in the function grow_halt_poll_ns() and is multiplied by the module parameters halt_poll_ns_grow and halt_poll_ns_grow_start.

In the event that the total block time was greater than the global max polling interval then the host will never poll for long enough (limited by the global max) to wakeup during the polling interval so it may as well be shrunk in order to avoid pointless polling. The polling interval is shrunk in the function shrink_halt_poll_ns() and is divided by the module parameter halt_poll_ns_shrink, or set to 0 iff halt_poll_ns_shrink == 0.

It is worth noting that this adjustment process attempts to hone in on some steady state polling interval but will only really do a good job for wakeups which come at an approximately constant rate, otherwise there will be constant adjustment of the polling interval.

[0] total block time:
> the time between when the halt polling function is invoked and a wakeup source received (irrespective of whether the scheduler is invoked within that function).

Module Parameters

The kvm module has 3 tuneable module parameters to adjust the global max polling interval as well as the rate at which the polling interval is grown and shrunk. These variables are defined in include/linux/kvm_host.h and as module parameters in virt/kvm/kvm_main.c, or arch/powerpc/kvm/book3s_hv.c in the powerpc kvm-hv case.

Module Parameter	Description	Default Value
halt_poll_ns	The global max polling interval which defines the ceiling value of the polling interval for each vcpu.	KVM_HALT_POLL_NS_DEFAULT (per arch value)
halt_poll_ns_grow	The value by which the halt polling interval is multiplied in the grow_halt_poll_ns() function.	2

halt_poll_ns_grow_start	The initial value to grow to from zero in the grow_halt_poll_ns() function.	10000
halt_poll_ns_shrink	The value by which the halt polling interval is divided in the shrink_halt_poll_ns() function.	0

These module parameters can be set from the debugfs files in:

/sys/module/kvm/parameters/

Note: that these module parameters are system wide values and are not able to be tuned on a per vm basis.

Further Notes

- Care should be taken when setting the halt_poll_ns module parameter as a large value has the potential to drive the cpu usage to 100% on a machine which would be almost entirely idle otherwise. This is because even if a guest has wakeups during which very little work is done and which are quite far apart, if the period is shorter than the global max polling interval (halt_poll_ns) then the host will always poll for the entire block time and thus cpu utilisation will go to 100%.
- Halt polling essentially presents a trade off between power usage and latency and the module parameters should be used to tune the affinity for this. Idle cpu time is essentially converted to host kernel time with the aim of decreasing latency when entering the guest.
- Halt polling will only be conducted by the host when no other tasks are runnable on that cpu, otherwise the polling will cease immediately and schedule will be invoked to allow that other task to run. Thus this doesn't allow a guest to denial of service the cpu.