Contribution ID: **227** Type: **not specified**

New challenges in using LLC sched domain on wakeup path

Monday, 20 September 2021 08:10 (20 minutes)

AMD and ARM server architectures further complicate the issue with wake_wide() overeager pulling (see other abstract).

An LLC domain can span a whole socket on an Intel server but are significantly smaller on AMD ZEN due to its CCXs. For example, on ZEN 2, each CCX has only 4 cores. When binding network IRQs to such a CCX, we can consistently reproduce a scenario in which over 50 iperf tasks pile up there.

Some ARM servers may suffer from the opposite problem of not having LLC domains at all because they don't expose L3 cache, also called SLC, to the kernel or support hyperthreading. wake_wide() and select_idle_sibling() rely on the existence of LLC domains to make smart decisions about wake affine and balancing load within an LLC domain. If there is no LLC domain, wake affine never happens and select_idle_sibling() won't try to look for an idle CPU within an LLC domain. In other words, a task will be woken up on its previous CPU even if it shares cache with the waker or the previous CPU is busy. Is this what we want? I don't think so, it's not optimal in some cases even if it helps in others. For instance, in our read-only YCSB workloads with static IRQ binding, always waking up on the previous CPU performs better on lightly loaded systems but worse on heavily loaded systems. So I think we should consider how to improve the use of LLC sched domains in the wakeup path on these architectures.

I agree to abide by the anti-harassment policy

I agree

Primary author: CHEN, Libo (Oracle)

Presenter: CHEN, Libo (Oracle)

Session Classification: Scheduler MC

Track Classification: Scheduler MC