Contribution ID: **215** Type: **not specified**

Remote charging in the CPU controller

Monday, 20 September 2021 09:10 (30 minutes)

CPU-intensive kthreads aren't generally accounted in the CPU controller, so they escape weight and bandwidth settings when they do work on behalf of a task group.

This is a problem in at least three places in the kernel. Padata multithreaded jobs (link1, link2, link3) may be started by a user task, so helper threads should be bound by the task's task group controls. Async memory reclaim (kswapd, cswapd) should be accounted to the cgroup that the memory belongs to, and similarly net rx should be accounted to the task groups of the corresponding packets being received. There are also general complaints from Android.

Each use case has its own requirements. In padata and reclaim, the task group to account to is known ahead of time, but net rx has to spend cycles processing a packet before its destination task group is known. Furthermore, the CPU controller shouldn't throttle reclaim or net rx in real time since both are doing high priority work. These make approaches that run kthreads directly in a task group, like cgroup-aware workqueues or a kernel path for CLONE_INTO_CGROUP, infeasible. The proposed solution of remote charging can accrue debt to a task group to be paid off (or forgiven) later, addressing both of these issues.

Prototype code has shown some of the ways this might be implemented and the tradeoffs between them. Here's hoping that an early discussion will help make progress on this longstanding problem (link1, link2, link3).

I agree to abide by the anti-harassment policy

I agree

Primary author: JORDAN, Daniel (Oracle)

Presenter: JORDAN, Daniel (Oracle)

Session Classification: Scheduler MC

Track Classification: Scheduler MC