LPC Android MC - Uclamp cgroup usage challenges in Android

Wei Wang, Quentin Perret

Android MC
Linux Plumbers Conference 2021
This talk is about

- Productize uclamp on Android
- Issues, pain points
- Thoughts, possible solutions
CPU controller usage in Android

- cpuctl cgroups are defined per ‘role’ of application
Problem: CPU Shares vs Unified Hierarchy
cpu.shares usage in Android

- cpu.shares helps a lot (5%~50% latency saved in app launch) under background-heavy scenario (e.g. dex2oat)
- Guarantees top-app gets a decent amount of CPU time, regardless of background noise
- Blocks cgroup v2 migration (per-app groups)
  - Number of background apps is not static - allocating a fixed bandwidth to top-app requires re-tuning all groups
  - Fairness between non-background groups
- Uclamp and cpu.shares in the same controller is limiting
Problem: Uclamp.max Aggregation
**uclamp.max aggregation**

- Runqueue `util_avg` and `uclamp.max` aggregation works as follows
  - `rq->util_avg = \text{Sum}(\text{task->util_avg})`
  - `rq->uclamp_max = \text{Max}(\text{task->uclamp.max})`

- Problematic scenario
  - a. a **long running** background task is running alone with `uclamp.max=50, util_avg=1024`
  - b. a **short** top-app task is co-scheduled on the same CPU, `uclamp.max=1024, util_avg=100`
  - c. the runqueue's `uclamp.max` is released, **frequency goes to max** for nothing
  - d. a single `uclamp.max` value can map to inefficient frequencies on some CPUs
    - EM-based frequency selection could help?
Proposals

- Apply uclamp.max at CFS rq level
  - Contribution of entire CFS sub-tree is restricted by uclamp max
  - Background tasks can never ask for more than they need
  - No limits to how much top-app can contribute
  - Util_est needs at CFS rq level also

- Let CPU run at efficiency point for each PD with uclamp.max
Problem: Uclamp.min Configuration
uclamp.min

- **Uclamp.min effectiveness**
  - Uclamp.min is usually used for meeting task deadline
  - Tasks that are small (or big) don’t need extra help

- **Solution**
  - Apply uclamp.min selectively (maybe based on task size?)
  - Userspace uclamp.min governor (to pass deadline information)
  - uclamp statistics collected through custom trace points
Problem: Per-task Uclamp Interface
Per-task uclamp interface

- No privilege checks in sched_setattr() for tasks changing their own uclamp
  - Uclamp settings from Apps can race with system settings
  - Proposal: introduce a new RLIMIT for uclamp, similar to nice and rt priorities

- No support for pidfd in sched_{set,get}attr() (TOCTOU)
  - Proposal: use the (currently unused) ‘flags’ argument to distinguish pid vs pidfd

- ‘reset-on-fork’ flag specifically for uclamp
  - Proposal: add a new sched_flag
Thanks!