Contribution ID: **125** Type: **not specified**

Per-cgroup Swap Device Control

ABSTRACT

Enabling cgroup-level control over swap devices

PROPOSAL

In certain restricted environments, there is a technical requirement to use otherwise idle devices as extended swap memory - including remote storage systems accessible over the network. A motivating scenario is to configure background processes to use these slower network-backed swap devices, while foreground processes use faster local storage.

Currently, the Linux kernel does not provide per-process or per-cgroup swap selection, which makes such usage unachievable. After reviewing alternatives, the conclusion is that swap devices must be controllable on a per-cgroup basis. Through prior discussions with Chris Li[1], it was suggested that this topic would be suitable for broader discussion at LPC.

This session will present the motivation, summarize the development progress, and discuss open design points. Prior discussions on the mailing list have already triggered meaningful debate, and broader discussion at Kernel Summit would be valuable.

AGENDA

Motivation for per-cgroup swap control

- Background and reasons for starting this work
- Comparison with alternative approaches (per-cgroup swap priority, swap device control on block device, swap tiering, per-cgroup dedicated swap device)
- Initial proposal: per-cgroup swap priority [2]
- Alternative proposal: swap tier approach (Chris Li) [3]

Implementation reviews and open problems

- Required changes in percpu clusters & swap fast paths when different cgroups select different tiers [4]
- Consistency with cgroup parent–child semantics: unlike general resource distribution [5], tier selection may bypass parent constraints [6]
- NUMA autobind interaction: tier priorities differ per NUMA node; current approach enforces exclusivity. Need to evaluate whether autobind is necessary, and how to integrate with swap tiers [7]
- Per-cgroup swap tier limit: Do we need a swap.tier.max in addition to the existing swap.max? [8]
- Parent–child tier mismatch: If zombie (child) memcg uses a tier not available to its parent, how should this be handled during recharging or reparenting? [9]
- Tier mask calculation: trade-offs between runtime calculation vs. set-time evaluation [10]
- If swap tier is applied to memcg, should we migrate swap-out pages that don't belong to any swap tier?

These agenda items highlight topics that require further discussion arising from the ongoing RFC ⟶ PATCH process.
The patch is still under active development, and additional debate points may emerge before the LPC presentation.

REFERENCES

[1] https://lore.kernel.org/linux-mm/CAF8kJuMo3yNKOZL9n5UkHx_O5cTZts287HOnQOu=KqQcnbrMdg@mail.gmail.com/
[2] https://lore.kernel.org/linux-mm/20250612103743.3385842-1-youngjun.park@lge.com/
[3] https://lore.kernel.org/linux-mm/CAF8kJuMo3yNKOZL9n5UkHx_O5cTZts287HOnQOu=KqQcnbrMdg@mail.gmail.com/
[4] https://lore.kernel.org/linux-mm/aLRTyWJN60WEu%2F3q@yjaykim-PowerEdge-T330/
[5] https://lore.kernel.org/linux-mm/rivwhhhkuqy7p4r6mmuhpheaj3c7vcw4w4kavp42avpz7es5vp@hbnvrmgzb5tr/
[6] https://lore.kernel.org/linux-mm/CACePvbUJSk23sH01msPcNiiiYw7JqWq_7xP1C7iBUN81nxJ36Q@mail.gmail.com/
[7] https://lore.kernel.org/linux-mm/CACePvbW_Q6O2ppMG35gwj7OHCdbjja3qUCF1T7GFsm9VDr2e_g@mail.gmail.com/
[8] https://lore.kernel.org/linux-mm/CACePvbUJSk23sH01msPcNiiiYw7JqWq_7xP1C7iBUN81nxJ36Q@mail.gmail.com/
[9] https://lore.kernel.org/linux-mm/20230720070825.992023-1-yosryahmed@google.com , https://lwn.net/Articles/895431/
[10] https://lore.kernel.org/linux-mm/CAF8kJuPj6-gZ4H+VQtJpJj_MutTgTcR-9BfDQnweayOrXk-NCQ@mail.gmail.com/

Primary author: PARK, YoungJun (LG Electronics)

Co-authors: LI, Chris (Google); SONG, Taejoon (LG Electronics)

Presenters: LI, Chris (Google); PARK, YoungJun (LG Electronics)

Session Classification: Kernel Memory Management MC

Track Classification: Kernel Memory Management MC