

HZ5: fail-closed な descriptor-owned allocator profile family

著者: 倉田 智朗 (TOMOAKI KURATA)

連絡先: charmpic moecharm.dev@gmail.com

X (Twitter): <https://x.com/CharmNexusCore>

Draft: 2026-05-27 v0.1

概要

HZ5 は HakoZuna project における後続 allocator line である。HZ3/HZ4 論文が local-heavy と remote-heavy の dual-profile 運用を中心に扱うのに対し、HZ5 は descriptor-owned / fail-closed / low-RSS な profile family を主題とする。

Linux RUNS=10 unified sweep では、HZ5 の一部 profile row が tcmalloc, mimalloc, HZ3, HZ4 とは異なる Pareto 点を示した。たとえば `t=8 main r90` では `hz5-pagerun64-main` が 60.07M ops/s / 30MB RSS、`tcmalloc` が 24.45M ops/s / 424MB RSS であった。また `t=8 mid_only r90` では `hz5-large128-source16` が 75.15M ops/s / 8MB RSS、`tcmalloc` が 20.12M ops/s / 478MB RSS であった。

ただし、HZ5 は universal allocator replacement ではない。local-only mixed row、small-object guard row、一部 LargeFront row では `tcmalloc` / `mimalloc` / HZ3 / HZ4 が依然として強い。したがって HZ5 の主張は、descriptor ownership、fail-closed routing、profile-scoped source design により、低 RSS の別 Pareto 点を作れることにある。

1. はじめに

現代の general-purpose allocator は、低レイテンシ、remote free 下のスケーリング、メモリ保持量の抑制という複数の要求を同時に受ける。HZ3/HZ4 は、単一 policy で 全 workload を取るのではなく、HZ3 を local-heavy/default、HZ4 を remote-heavy/high-thread profile として分ける方針を示した。

HZ5 は別の問いから始まる。

所有権境界をより明示し、fail-closed にしながら、
低 RSS の profile family を作れるか。

現在の結果は yes である。ただしそれは、HZ5 を universal default ではなく profile family として扱う場合に限る。

1.1 読み方

本稿は HZ3/HZ4 論文の置き換えではない。同じ設計系列に属する別論文である。

HZ3/HZ4 論文:

Box Theory、PTAG32 routing、local/remote dual-profile 運用。

HZ5 論文:

descriptor ownership、fail-closed route contract、low-RSS profile family。

HZ6:

transfer-first design の今後の課題。

2. 設計目標

HZ5 の設計目標は次の 4 点である。

2.1 Descriptor-owned state

allocator state を inactive な user data page や intrusive metadata のみに依存させない。descriptor を所有権の正本にすることで、route validation、source accounting、将来の memory return policy を扱いやすくする。

2.2 Fail-closed routing

HZ5 らしく見える pointer が validation に失敗した場合、別 allocator へ silently fallthrough してはならない。

foreign pointer:

HZ5 所有ではない。必要なら外部 path へ。

HZ5-owned valid pointer:

HZ5 内で処理する。

HZ5-looking invalid pointer:

fail closed。libc/tcmalloc/HZ3 fallback に流さない。

2.3 単一 default ではなく profile family

HZ5 は object size、alignment、remote-free shape に応じて profile row を分ける。

HZ5 row	範囲
hz5-pagerun64-main / hz5-pagerun64-cross128	MidPage PageRun64 の general / cross-size profile
hz5-large128-rss	low-RSS LargeFront profile
hz5-large128-source16	LargeFront 128K throughput 比較 lane
hz5-large128-transfer128	transfer-cache 診断 lane。既定ではない

2.4 診断 counter を speed run に混ぜない

HZ5 開発では多くの diagnostic lane を使ったが、paper-facing speed run では attribution counter を無効化する。

3. 実装概要

HZ5 は route contract、source map、profile-specific backing policy を組み合わせる。platform ごとに実装は異なるが、共有する contract は次の通りである。

route:

owned / foreign / invalid

ownership:
descriptor が allocator state の正本

source:
page/run または large-front profile が backing memory を所有

diagnostics:
speed lane から分離

Linux では、RUNS=10 unified sweep で評価した profile family が paper-facing backend である。
Windows の Local2P / control-plane 実験は portability/control evidence であり、Linux paper backend と同一視しない。

4. 評価

4.1 評価プロトコル

紙面向け dataset は次である。

paper/results_hz5_linux_20260526_run10_unified

- Ubuntu 22.04.5 LTS, Linux 6.8.0-90-generic
- AMD Ryzen 7 5825U
- GCC 11.4.0, glibc 2.35
- RUNS=10, median ops/s
- RSS は ru_maxrss
- HZ3 / HZ4 / mimalloc / tcmalloc / system / HZ5 を同一 runner ・ 同一 machine で測定

4.2 代表的な positive row

Case	Best HZ5 row	HZ5 ops/s	HZ5 RSS	tcmalloc ops/s	tcmalloc RSS
t=8 cross128 r0	hz5- large128- rss	103.42M	9MB	47.43M	44MB
t=8 cross128 r50	hz5- pagerun64- main	23.47M	58MB	19.74M	103MB
t=8 cross128 r90	hz5- large128- rss	18.67M	84MB	11.20M	190MB
t=8 main r50	hz5- large128- source16	65.11M	19MB	23.13M	413MB
t=8 main r90	hz5- pagerun64- main	60.07M	30MB	24.45M	424MB

Case	Best HZ5 row	HZ5 ops/s	HZ5 RSS	tcmalloc ops/s	tcmalloc RSS
t=8 mid_only r50	hz5- large128- source16	76.63M	7MB	19.41M	516MB
t=8 mid_only r90	hz5- large128- source16	75.15M	8MB	20.12M	478MB
t=8 large128 r90	hz5- large128- source16	12.19M	156MB	12.71M	193MB

4.3 claim boundary

HZ5 を過大主張してはならない。

Case	Winner	Winner ops/s	Best HZ5 ops/s	読み
t=8 guard r0	tcmalloc	250.99M	119.78M	small-object local-only は tcmalloc が 強い。
t=8 guard r50	mimalloc	77.16M	20.61M	HZ5 guard remote row は 主張対象では ない。
t=8 guard r90	mimalloc	66.56M	11.26M	small-object remote は HZ5 の主張では ない。
t=8 main r0	tcmalloc	237.13M	120.30M	local-only mixed は tcmalloc/HZ3 領域。
t=8 mid_only r0	tcmalloc	227.07M	121.58M	HZ5 は remote- pressure mid row で強い。
t=8 large128 r50	tcmalloc	18.61M	15.25M	LargeFront 128K は profile- sensitive。

正しい結論は「HZ5 が tcmalloc より常に速い」ではない。正しい結論は、HZ5 が低 RSS と fail-closed ownership を持つ別 profile family を作れる、である。

5. 考察

5.1 HZ3/HZ4 との違い

HZ3/HZ4 は small-object local / remote-heavy の operating policy を扱う。HZ5 は ownership explicitness と RSS を中心にした profile family である。そのため HZ5 は HZ3/HZ4 の後継 default ではなく、兄弟 line として扱う。

5.2 Windows evidence

Windows Local2P 実験は portability/control evidence として有用だが、Linux paper backend ではない。Windows では exact 64K/a8192 remote-throughput の強みがある一方、phase-churn では broad low-RSS profile として扱えない。したがって本稿では Windows HZ5 を補足 evidence に留める。

5.3 no-go lane の意味

HZ5 では複数の no-go lane が得られた。

- transfer-cache は narrow row では効いたが default にはならなかった
- Windows Local2P は remote throughput では有望だが broad RSS winner ではなかった
- diagnostic/counter lane は挙動説明には有効だが speed table から除外した

これらは HZ5 を profile family として扱う根拠である。

6. 限界と今後の課題

- HZ5 は universal allocator default ではない。
- 最も強い紙面向け結果は Linux profile row である。
- Windows HZ5 は matching route coverage に到達するまで補足扱いとする。
- fragmentation は RSS と retention 観測で評価しており、厳密モデル化は今後の課題である。
- HZ6 は class transfer cache と RSS governor を主要な箱として持つ transfer-first line として、別途評価すべきである。

7. 結論

HZ5 は、descriptor-owned / fail-closed allocator design が HZ3、HZ4、tcmalloc、mimalloc とは異なる低 RSS の Pareto 点を作れることを示した。強い主張は universal dominance ではなく、profile-scoped strength である。したがって HZ5 は HZ3/HZ4 論文へ 混ぜ込むのではなく、別 Zenodo record / 別 DOI の standalone paper として公開するのが 適切である。