

Linux Memory Overcommit & OOM Killer Architectural Handbook

Enterprise Systems Engineering: Heuristic Badness Scoring and PostgreSQL/Redis Stability

Published by **Lesty Tech Systems Architecture & Reliability Group** | Date: October 20, 2026 | Reference: LESTY-ENG-2026-B6

1. The Virtual Memory Illusion: Optimistic Overcommit

By default, the Linux kernel operates in an optimistic memory overcommit mode (`vm.overcommit_memory = 0`). The kernel permits user-space applications to allocate (`malloc`, `mmap`) far more virtual address space than physical RAM plus swap capacity exists. This design accommodates legacy UNIX programming paradigms where processes call `fork()` and allocate massive memory buffers that are sparsely populated. The kernel only allocates physical page frames when an application actually writes to a memory page, triggering a minor page fault.

2. Copy-on-Write (COW) and The Sudden Memory Cliff

Systems Reliability Trap: Enterprise database systems such as PostgreSQL and Redis rely heavily on `fork()` for background maintenance, point-in-time recovery, and RDB snapshot creation. While `fork()` is initially instantaneous because child processes share parent pages with Copy-on-Write (COW) flags, subsequent writes by active transaction threads force the kernel to duplicate physical pages. If the server enters heavy transactional load, physical RAM can be depleted within milliseconds, triggering an immediate out-of-memory crisis.

3. The OOM Killer Badness Heuristic and Decapitation Prevention

When the kernel enters an irrecoverable Out-of-Memory (OOM) condition, the OOM killer daemon invokes `oom_badness()` to select a sacrificial process to terminate. The heuristic computes a score based on raw Resident Set Size (RSS) and process runtime. Because database master processes hold the largest memory allocations, the OOM killer frequently assassinates the primary database daemon instead of the rogue leaking thread! Setting `vm.overcommit_memory = 2` enforces strict mathematical allocation limits, and tuning `oom_score_adj = -1000` grants database daemons absolute kernel immunity.

ARCHITECTURAL BENCHMARK & PARAMETER TELEMETRY

Kernel Parameter	Default Linux Behavior	High-Risk Scenario	Hardened Database Tuning
vm.overcommit_memory	0 (Heuristic Overcommit)	Allows infinite virtual allocation	2 (Strict No-Overcommit)
vm.overcommit_ratio	50% (of Physical RAM)	Ignored when overcommit=0	80% - 90% (Enforces strict cap)
oom_score_adj (Postgres/Redis)	0 (Default heuristic)	Assassinated first due to large RSS	-1000 (Absolute Kernel Immunity)
vm.panic_on_oom	0 (Invoke OOM Killer)	Kills random user processes	2 (Instant Failover Reboot in HA)

vm.swappiness	60 (Aggressive file cache swap)	Pushes active database pages to disk	1 - 10 (Avoids swap I/O latency)
---------------	---------------------------------	--------------------------------------	----------------------------------