zram: compressione della memoria su Linux

- Talk in 10 slide — panoramica, storia, alternative, setup & tuning
- Relatore: (vroby65)
- Licenza: CC BY-SA (se desideri diffondere le slide)

Cos’è zram (in 30 secondi)

- Modulo del kernel che crea block device compressi in RAM
- Uso tipico: swap su zram (meno I/O su disco, più reattività)
- Algoritmi: LZ4, ZSTD, LZO-RLE… (trade-off velocità/ratio)
- Vantaggi: salva RAM, limita thrashing, allunga la vita degli SSD
- Svantaggi: CPU usata per comprimere/decomprimere

Perché ha senso oggi

- RAM veloce ma non infinita; app e browser famelici
- Swap tradizionale su disco = latenza alta
- zram sposta parte dello swap in RAM compressa
- In molti workload: più cache calda e meno stall I/O
- Ottimo per laptop, VM, container, SBC (Raspberry, ecc.)

Storia dei meccanismi RAM & compressione

- Anni ’90–2000: swap su disco (IDE/SATA), poi SSD
- 2008–2009: compcache (swap compresso in RAM)
- Evoluzione kernel: zcache (storico), zswap (cache per la swap su disco)
- Oggi: due strade principali su Linux → zram e zswap
- Altri OS: macOS (Compressed Memory), Windows (Memory Compression)

Alternative/competitor su Linux

- zswap: cache compressa davanti alla swap su disco (non un block device)
- zram: block device compresso; spesso usato come swap primaria
- zbud/zpool/zsmalloc: mattoni per pagine compresse
- Scelta: zram se niente I/O; zswap se vuoi backing su disco

Storia & autori (focus zram)

- Origini in compcache (idea di Nitin Gupta)
- Contributi chiave: Minchan Kim, Sergey Senozhatsky e altri
- Oggi parte stabile del kernel; integrata nelle principali distro
- Tool: zramctl, generatori systemd, pacchetti distro-specifici

Come funziona (architettura)

- zram crea /dev/zramN (block device “virtuale” in RAM)
- Le pagine scritte sono compresse e tenute in memoria
- Uso comune: swap, ma anche fs temporanei (/var/tmp, build caches)
- Scelte: algoritmo, dimensione, numero device, priorità swap
- Opzionale: writeback verso backing file in casi particolari

Quando usarlo / quando evitarlo

- Usarlo se: RAM non enorme; molte tab/app; VM/containers; SSD da preservare
- Usarlo su: sistemi ARM/SBC e laptop
- Valutare/evitare se: CPU già satura (compressione costa)
- Workload CPU-bound con poca memoria → poco beneficio
- Server con RAM abbondante e I/O top → non necessario

Installazione rapida (multi-distro)

- Con systemd-zram-generator (consigliato): pacchetto zram-generator
- Config: /etc/systemd/zram-generator.conf — zram-size=ram/2, algorithm=LZ4
- Attiva: systemctl daemon-reload; start systemd-zram-setup@zram0; swapon --show
- Debian/Ubuntu: zram-tools (file /etc/default/zramswap)

Tuning pratico

- Taglia: da ram/4 a ram/2 è spesso un buon compromesso
- Algoritmi: LZ4 = velocissimo; ZSTD = miglior ratio, più CPU
- vm.swappiness=80–120 spesso funziona bene con zram
- Dai priorità maggiore allo swap su zram rispetto a disco
- Monitor: zramctl, /proc/swaps, /sys/block/zram0/mm_stat

Diagnostica & best practice

- Comandi: zramctl; cat /proc/swaps; cat /sys/block/zram0/mm_stat
- Attenzione a OOM killer se la memoria finisce comunque
- Evita oversizing aggressivo → contention CPU
- Testa sul tuo workload: browser, compilazioni, container, VM
- Documenta: conserva conf e note di performance per il team