PART 9 · SECOND EDITION · SEPTEMBER 2026 (FIRST PUBLISHED JULY 2026)

Storage & Data

The most expensive component in the building spends its life waiting on the cheapest. Part 9 is about making sure it never does — feeding, saving, and remembering at the speed of a $40,000 accelerator.

THE DEFINITIVE GUIDE TO
AI Data *Centers*

LLAMA 3 405B · UP TO 16,384 H100S · 54 DAYS

1 every ~3 hr (419 in 54 days)

Frontier training runs on a failure clock: an interruption every ~3 hours, mostly hardware

Storage is what decides how much work each failure erases, and whether the GPUs were even fed between failures. That makes it a goodput system, not a filing cabinet.

META · AS-OF 2024-07

THE REFRAME

Storage decides whether a $40,000 accelerator computes or idles.

And it is not one system. Four mutually hostile I/O personalities — ingest, checkpoint, small-file metadata, KV-cache — share the same flash and fabric, and a tier tuned for one starves the others.

ACT I

The feed path

Sizing starts with one question: can this tier keep this many accelerators fed? Bandwidth per GPU — not capacity — is the primary axis, and the metadata plane is where deployments actually fail.

THE TERRITORY

Four workloads, one set of hardware

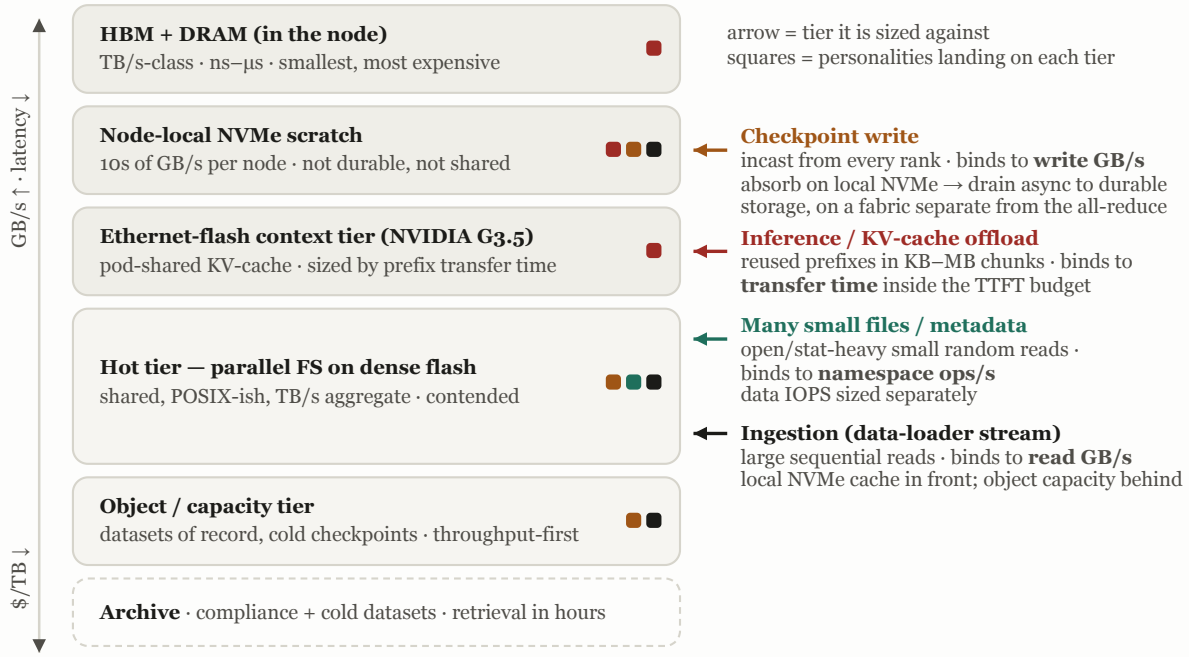
PERSONALITY	BINDING METRIC	FAILURE MODE IF MIS-SIZED
Ingestion (data reads)	Read GB/s per GPU	GPU utilization caps low; accelerators stall waiting on data
Checkpointing (state writes)	Write GB/s burst, fast drain	Long stalls per interval; you checkpoint less and lose more work
Many-small-files / metadata	Metadata ops per second	Metadata service saturates; throughput collapses regardless of bandwidth
KV-cache offload (inference)	Prefix transfer time vs TTFT budget	Time-to-first-token blows the SLO; fewer users per GPU

THE MAP

Five tiers, four personalities — each workload binds to a different tier and a different metric

I/O patterns can share a tier but bind to different metrics

Faster tiers up, cheaper tiers down; route each personality by its binding metric.



Benchmark the binding metric of your dominant personality; fix checkpoint incast against the collectives by placement (local fast tier + separate fabric), not a bigger array; local staging is durable only after the protected commit.

Schematic · Source: Guide synthesis · as of Oct 2026

Bandwidth improves toward the chip, cost per terabyte toward the archive. Benchmark the metric your dominant personality binds to — GB/s, IOPS, tail latency, or metadata ops — not the datasheet.

THE ONE NUMBER TO INTERNALIZE

~4 GB/s/GPU

NVIDIA's per-GPU read target for vision datasets over 30 TB; its B300 tiers are ~0.14–0.49 GB/s per GPU.

Sizing per GPU forces the right question — can this tier keep this many accelerators fed? — instead of the comfortable but useless question about total capacity.

NVIDIA DGX SUPERPOD B300 RA, STORAGE ARCHITECTURE · AS-OF 2025-11

THE MASTER FORK

Bandwidth-sized, or capacity-sized?

Bandwidth-sized (GB/s first)

— Primary spec: per-GPU feed rate times GPU count

— Checkpoint writes follow blocking budget, dirty fraction, drain overlap and recovery objective

— What almost every AI hot tier actually is

— Capacity is computed second, and lands on a cheaper tier

Capacity-sized (PB first)

— Primary spec: corpus size plus checkpoint retention

— The right axis for the object and data-lake tier

— Applied to the hot tier, it buys a huge array that starves GPUs

— Idle accelerators are the invoice for the mistake

SO WHAT

Derive the hot tier from feed rate and capacity from the corpus — two separate computations on two different tiers. Operators who collapse them into one number strand either accelerators or flash.

THE LOSF TRAP

~2.75M vs ~1.6M vs ~0.94M

Metadata IOPS, measured: IO500 metadata scores spread widely, and metadata is where AI storage actually fails.

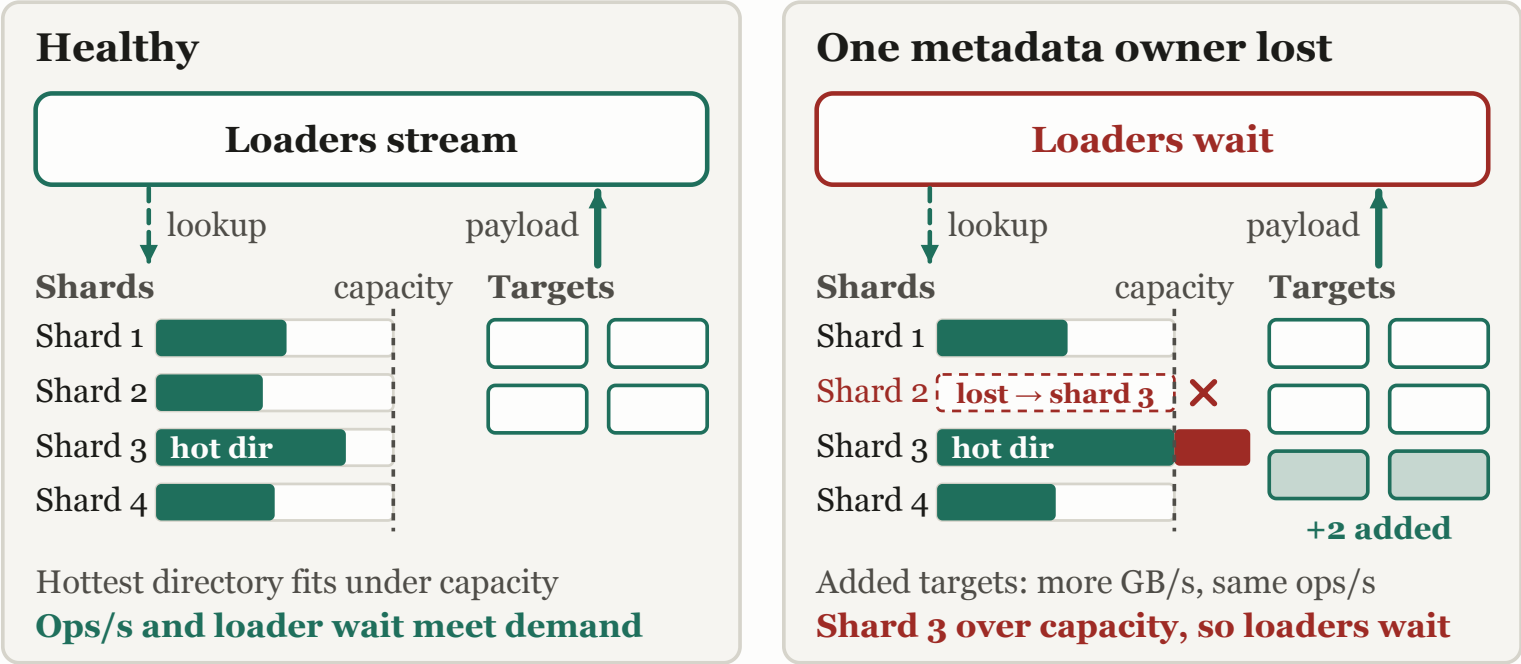
Vendors lead with peak sequential throughput because it is their biggest, cleanest number. Insist on metadata and small-random results under your real file-size distribution.

IO500 ISC26 PRODUCTION LIST · AS-OF 2026-06

ANATOMY OF THE HOT TIER **Data bandwidth scales with every flash target you add. The metadata plane doesn't.**

More targets add bandwidth, not namespace operations

Same loaders and files; each bar is a metadata shard's load



If degraded ops/s miss demand: add metadata service, rebalance directories or pack files

Schematic · Source: Guide derivation · as of Sep 2026

One read stripes across dozens of targets in parallel — but every open and stat lands on the metadata service. That's why deployments fail there long before they run out of bandwidth.

BUILD VS BUY

Open-source build, or certified appliance?

Build it — Lustre / Ceph / DAOS

— Lowest up-front cost; commodity hardware, no per-TB license

— You own tuning, rebuilds, client/kernel compat, and the on-call

— You prove the per-GPU feed and the GPUDirect path yourself

— Fits frontier scale with a deep storage team

Buy it — DDN / Weka / VAST / IBM

— Appliance premium plus a support contract

— Pre-validated against the GPU vendor's reference tiers

— Vendor on the hook for the feed targets

— Lock-in — sometimes down to a specific DPU generation

SO WHAT

The premium is insurance against idle-GPU TCO. Price it against what a percent of stalled accelerators costs your program, not against the hardware quote.

DELETING THE CPU FROM THE DATA PATH

The bypass ladder — each rung removes host overhead, and narrows your suppliers

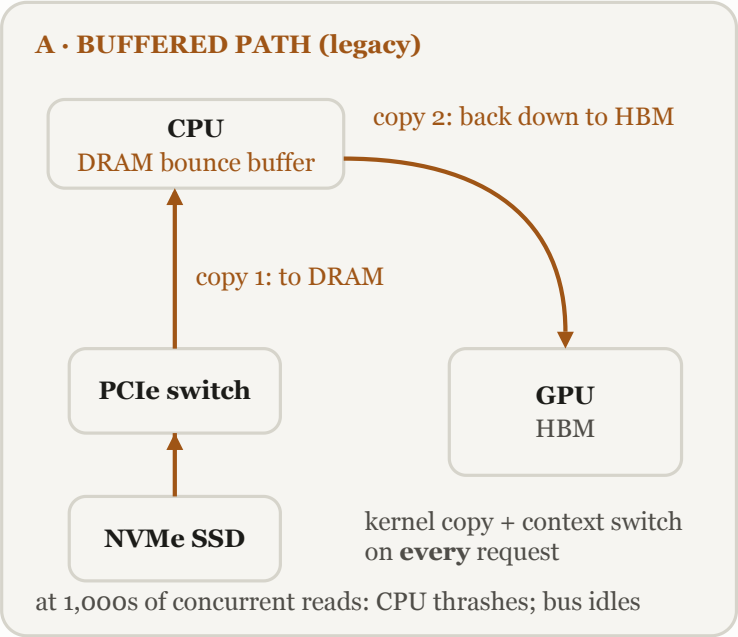
PATH	CPU IN THE PATH?	AVAILABILITY
Buffered POSIX (legacy)	Data and control both on the CPU	Universal
GPUDirect Storage (GDS)	Data path off; control still on the CPU	Mature (CUDA / cuFile stack)
SCADA (GPU-initiated I/O)	Off the per-request path; host still sets up access	Emerging 2026; narrow supported stack
BlueField-4 / DPU offload	Neither — the DPU initiates	Announced; platforms land H2 2026

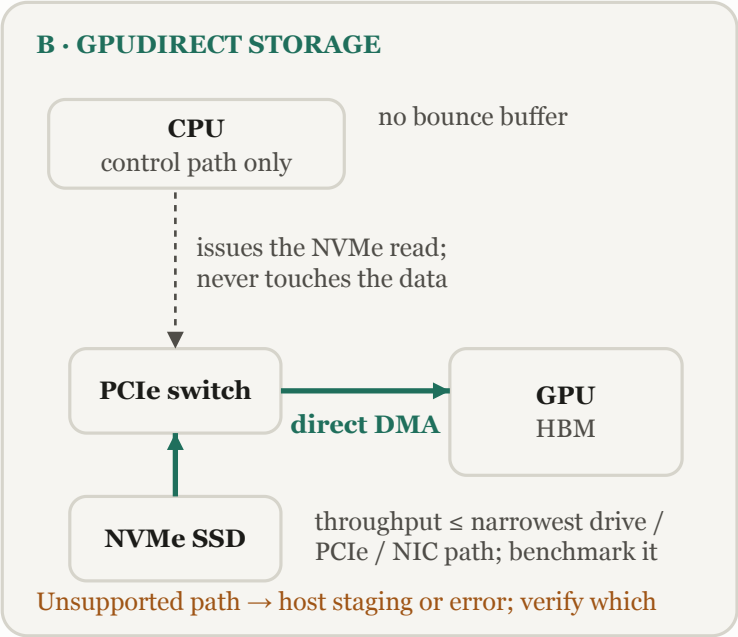
THE BYPASS, DRAWN

GPUDirect Storage bypasses CPU data copies into GPU memory

The CPU bounce buffer, and the path that removes it

Same hardware, two data paths. Deleting the copy pays only when host staging was the bottleneck.





The 2026 frontier takes the control path too.
SCADA (NVIDIA's GPU-initiated I/O, not industrial SCADA): the GPU issues storage I/O itself.
BlueField-4: the DPU owns the control path. Either way, the host leaves the per-request loop.

Schematic · Source: Guide synthesis · as of October 2026

The bounce buffer costs two copies and a context switch per request; at thousands of concurrent reads the CPU thrashes while PCIe idles. The bypass trades that overhead for a narrower stack.

THE LOADER IS A GOODPUT PROBLEM

~100%

GPU utilization in AWS's S3 loading test with sequential tar shards; random per-object reads lagged until workers rose.

Format and loader concurrency decide it: sequential shards feed; per-object reads need more workers; one-file-per-sample is a metadata storm, costly to reverse.

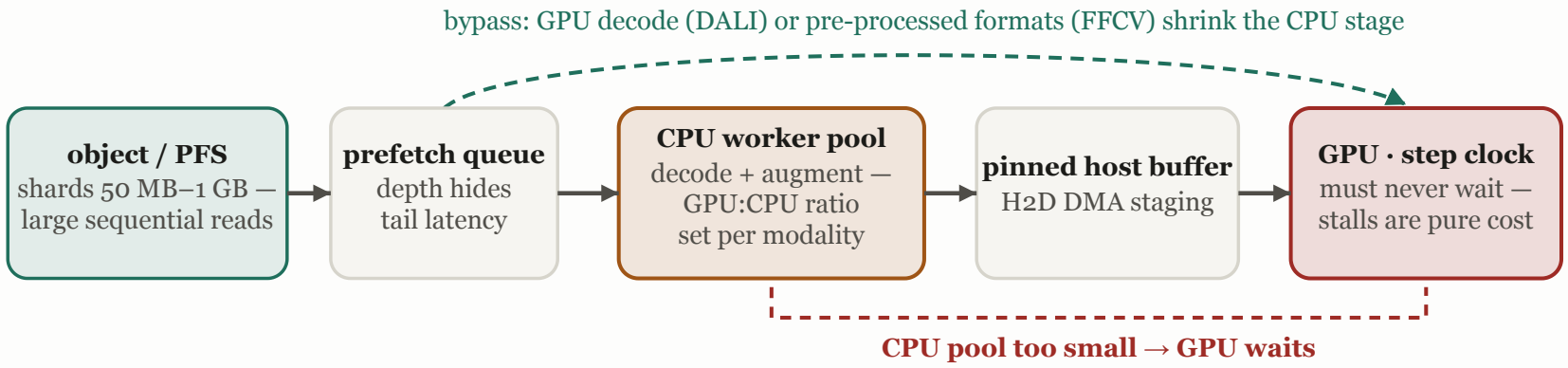
AWS MACHINE LEARNING BLOG, 'APPLYING DATA LOADING BEST PRACTICES FOR ML TRAINING WITH AMAZON S3 CLIENTS' · AS-OF 2025-12

THE LAST MILE

Between storage and the GPU sits a CPU pipeline — and the step clock must never wait on it.

The slowest loader stage sets how long the GPU waits

Shards make the reads sequential; the CPU pool makes or breaks the step time; the GPU clock is the judge.



The three-way diagnosis — watch two utilization needles and the step clock:

decode-bound: CPUs pinned at 100%, storage loafing → more cores, DALI offload, or pre-processing

I/O-bound: CPUs idle, GPU starving → bigger shards, deeper prefetch, more parallel connections, local cache

jitter-bound: neither saturated, stalls anyway → tail-latency reads or heavy samples — hunt the p99, not the average

Sharding turns random reads into streams: line-rate PFS versus metadata-bound one-file-per-sample I/O (Ch 9.2).

Schematic · Source: Guide synthesis · as of Oct 2026

The diagnosis is a two-gauge read: CPUs pinned with storage loafing means decode-bound; CPUs idle with the GPU starving means I/O-bound. Fix the saturated stage — or offload decode to the GPU.

ACT II

The goodput clock

Failures arrive on a schedule you don't control. Checkpointing is the control knob for how much work each one erases — and mid-epoch resumption is its forgotten twin on the read side.

CHECKPOINT ANATOMY

~14 bytes/param

Checkpoint state per parameter for the BF16 + FP32-master + Adam recipe — roughly 7x the weights you ship.

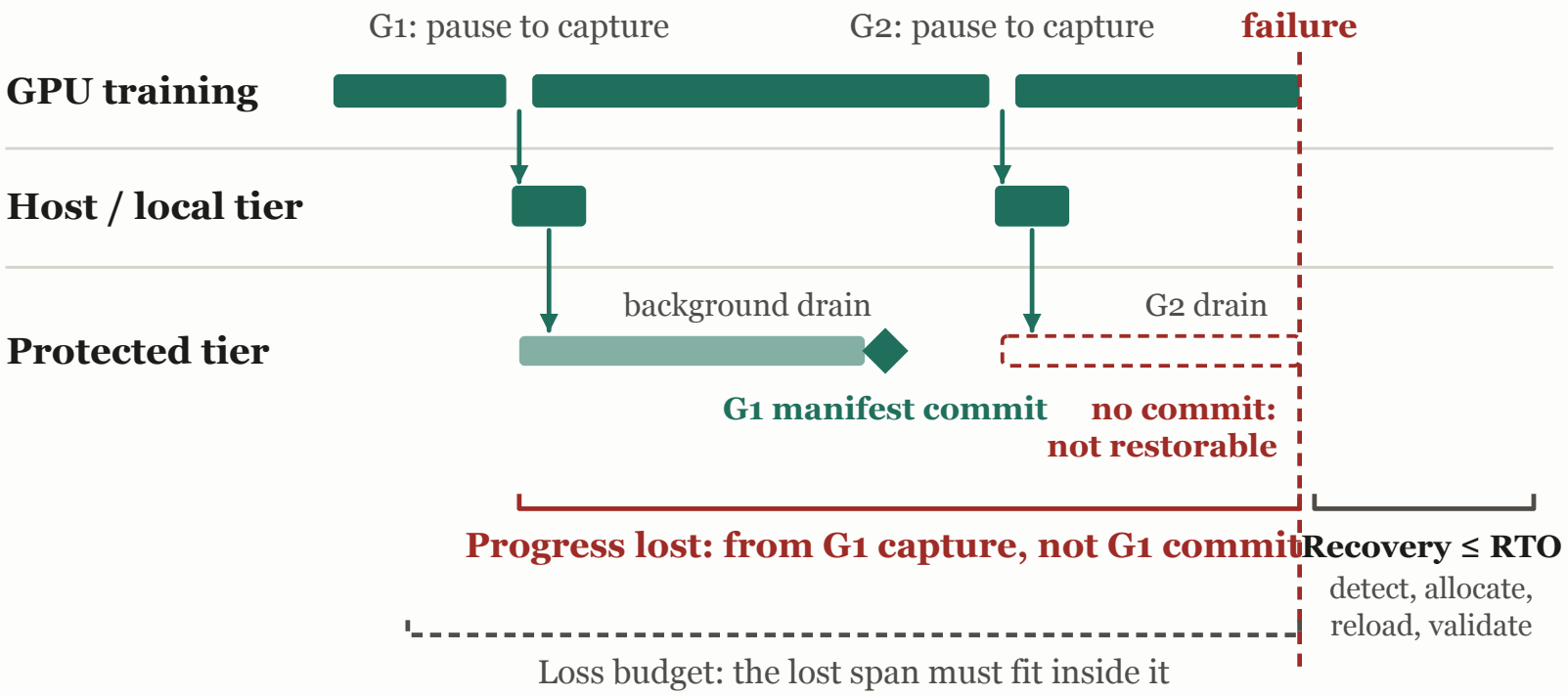
Most of it is optimizer and master-weight state, not the model. The cheapest lever on checkpoint cost is often the optimizer recipe, not the storage tier.

VAST DATA, 'OPTIMIZING CHECKPOINT BANDWIDTH FOR LLM TRAINING' · AS-OF 2026

SYNC, ASYNC, OR PEERS **Three ways to save state — and a formula for how often**

Lost work counts back to the last committed capture

Asynchronous checkpoint timeline; time runs left to right



Schematic · Source: Guide derivation + PyTorch · as of Sep 2026

Async tiering separates blocking snapshots from background drain. Young gives the first-order square-root interval; Daly adds higher-order checkpoint, recovery and downtime terms.

MULTI-TIER IS THE ANSWER

What a tiered checkpoint policy buys

+6.59% goodput

Goodput gain from multi-tier checkpointing on a 35K-chip TPU v5p workload

GOOGLE CLOUD · AS-OF 2025

up to 30 min → <1 min

Restore time across thousands of nodes: object-only vs multi-tier

GOOGLE CLOUD · AS-OF 2025

90% vs 96% scenario

Illustrative 90%-versus-96% goodput sensitivity; use the named fleet's measured goodput

GUIDE ANALYSIS · AS-OF 2026-09-08

THE INCAST THAT EATS ITS OWN GOODPUT

Where the checkpoint burst rides

PLACEMENT	ISOLATION	BEST FIT
Dedicated storage rail	Physical — the incast cannot touch the collectives	Frontier synchronous training
Converged + strict QoS	Logical — holds only if PFC/ECN tuning holds under load	Cost-sensitive clusters with disciplined fabric teams
Converged, no isolation	None — goodput loss on every checkpoint cadence, monitoring green	Inference only; never frontier training

ACT III

Gravity & the capacity tier

The corpus is the heaviest object in the program. Egress economics decide where compute gets built — and the bucket that holds it all is no longer an archive, because GPUs now read from it.

EGRESS IS A RECURRING COST

74–79%

Egress share of the cloud storage bill in one vendor-modeled weekly-retrain scenario (20 TB stored, 15 TB/month out).

Cheap to put data in, a recurring cost to take it out. Exit waivers and the EU Data Act ease lock-in, but at petabyte scale gravity can still decide where GPUs get built.

AKAVE CLOUD BLOG · AS-OF 2026-05

THE CAPACITY TIER LEARNS TO SERVE

~7 GB/s vs ~300 MB/s

QLC flash vs nearline HDD, sequential read — the gap that lets object storage feed GPUs directly.

Flash-fronted object streams shards into the training loop at line rate; HDD nearline must stage through a faster tier first. Size the bucket as goodput, not archive.

SOLIDIGM · AS-OF 2025

THE DATA-PREP SUPERCOMPUTER

Before a GPU sees a token

15T tokens

Tokens in the 2024 FineWeb release, distilled from 96 Common Crawl snapshots

FINEWEB PAPER · AS-OF 2024-05

~50%

Share of a multi-snapshot web crawl that dedup removes before quality filtering

REFINEDWEB · AS-OF 2024

up to 8 pts

Largest accuracy drop on GSM1k, a fresh mirror of GSM8K — not a contamination rate

ZHANG ET AL. · AS-OF 2024

A DELIBERATELY DIFFERENT MACHINE

Data prep is a second supercomputer — CPU-bound, air-cooled, sited where the corpus lives. Almost every axis is the opposite of the GPU fleet.

Bolting it onto idle GPU nodes burns frontier silicon on string processing. And mind the FineWeb lesson: aggressive global dedup preferentially deletes the good, durable web — it made data worse.

ACT IV

Inference's new memory hierarchy

Inference turned context into a storage problem. Where a request's KV bytes land — HBM, DRAM, CXL, NVMe, Ethernet flash — sets tokens per second, cost per token, and users per GPU.

THE NEW TIER

Where the KV-cache lives sets time-to-first-token and cost per user

Each tier down is larger and cheaper per byte, and slower to fetch from.

G1 · HBM (on-GPU)
~TB/s · 80–288 GB · the scarce tier

G2 · Host DRAM + CXL-expanded DRAM
~100s GB/s · 0.5–4 TB · first spill target

G3 · Local NVMe (GPUDirect Storage)
~10s GB/s · 10s TB · prefix / session cache at rest

G3.5 / G4 · Ethernet-attached KV store (disaggregated)
network-bound · shared across the fleet · cheapest byte, longest fetch

The pressure
70B @ 128K context · ~40 GiB aggregate / sequence
Sharded 8 ways (TP=8): ~5 GiB/rank pre-overhead

The payoff
prefix caching + offload: gain set by hit rate + fetch time
MLA + ~6-bit KV cut cache 93.3% (DeepSeek-V2)

Bandwidth falls ~100× top to bottom while capacity grows and $/GB falls.

Keep the hottest, most-reused blocks high, and **size each link so a hit costs less than recomputing it**.

Schematic tiers · KV size from the Llama 3 paper (Table 3) · Source: Meta + DeepSeek-AI (2024) · as of October 2026

The KV-cache is the first storage tier on the latency-critical path of a live request. A miss either burns prefill FLOPs to recompute or makes the user wait on a slower tier.

AGGREGATE, THEN PLACE IT

40 GiB aggregate; 5 GiB/rank at ideal TP=8

Llama 3 70B at 128K: ~42 GB aggregate logical KV state per sequence across participating ranks

With ideal TP=8 KV-head sharding, that is ~5.25 GB/rank before replication and runtime overhead. Compare it with HBM after local weights and workspaces; 42 GB is not one GPU's allocation.

META MODEL DIMENSIONS PUBLISHED 2024 · AS-OF 2024-07

KV-CACHE ECONOMICS

What the KV-cache costs, and what managing it buys

about 22 ms

Fetch time for one rank's reusable prefix in the worked KV case (illustrative)

WORKED EXAMPLE IN CHAPTER 9.7 · AS-OF 2026-09

93.3% (~15x)

KV-cache memory cut, DeepSeek-V2 (MLA plus quantization) vs 67B — not inference cost

DEEPSEEK-AI, DEEPSEEK-V2 TECHNICAL REPORT · AS-OF 2024-05

504 KiB/token (516.096 kB/token)

KV-cache bytes per token of context, Llama 3.1 405B (BF16) — why the hierarchy exists

META MODEL DIMENSIONS PUBLISHED 2024 · AS-OF 2024-07

THE THROUGH-LINE

The unit of storage performance is not GB/s or IOPS. It is the GPU-hour you did not waste.

TAKE IT HOME

What to do Monday

01 Size reads and writes from traces, checkpoint state and blocking/drain budgets; compute capacity separately by tier.

02 Benchmark metadata and small-random IOPS under your real file distribution — never accept the sequential datasheet number.

03 Size checkpoint tiers from bytes per parameter; isolate recovery fan-out and aggregation from training collectives.

04 Prove mid-epoch deterministic resumption before the run starts — restart-from-top repays egress and GPU time every failure.

05 Model egress before the corpus lands anywhere: exit waivers ease lock-in, but recurring egress still shapes where compute goes.

06 Treat the KV-cache as a managed, tiered asset — and as a confidentiality boundary: pooled reuse across tenants is a leak.

THE DEFINITIVE GUIDE TO

AI Data *Centers*

This deck is the map. The guide is the territory — every number here is dated, sourced, and kept current in the full chapters.

Ch 9.1 The four I/O personalities and per-GPU bandwidth sizing

Ch 9.4 Checkpointing as the goodput control knob

Ch 9.7 The KV-cache memory hierarchy for inference

Ch 9.8 Sizing, data gravity, and incast isolation

READ THE FULL PART, WITH EVERY SOURCE AND DERIVATION, AT AIDATACENTERGUIDE.COM

SOURCES

Every figure in this deck traces to the guide's dated register (1/3)

Meta (Llama 3 Herd of Models) · as-of 2024-07 · https://arxiv.org/abs/2407.21783

NVIDIA DGX SuperPOD B300 RA, Storage Architecture (4 GB/s per GPU of read performance for automotive and computer-vision workloads with datasets over 30 TB); the H100 RA gives the same figure · as-of 2025-11 · https://docs.nvidia.com/dgx-superpod/reference-architecture/scalable-infrastructure-b300/latest/storage-architecture.html

IO500 ISC26 Production List · as-of 2026-06 · https://io500.org/list/isc26/production

AWS Machine Learning Blog, 'Applying data loading best practices for ML training with Amazon S3 clients' · as-of 2025-12 · https://aws.amazon.com/blogs/machine-learning/applying-data-loading-best-practices-for-ml-training-with-amazon-s3-clients/

VAST Data, 'Optimizing Checkpoint Bandwidth for LLM Training' — states 14 bytes/parameter as an assumption used to estimate parameter counts (footnote 1); the recipe itself is arithmetic · as-of 2026 · https://www.vastdata.com/blog/optimizing-checkpoint-bandwidth-for-llm-training

Google Cloud (multi-tier checkpointing) · as-of 2025 · https://cloud.google.com/blog/products/ai-machine-learning/using-multi-tier-checkpointing-for-large-ai-training-jobs

Guide analysis — stipulated sensitivity scenario; no claim of an industry measurement. · as-of 2026-09-08

SOURCES

Every figure in this deck traces to the guide's dated register (2/3)

Akave Cloud blog (zero-egress storage vendor), 'What are egress fees', 2026-05-07 · as-of 2026-05 · https://akave.com/blog/what-are-egress-fees-the-complete-guide-for-data-engineers

Solidigm / Meta Engineering QLC case · as-of 2025 · https://www.solidigm.com/content/dam/solidigm/en/site/products/technology/qlc-nand-ssds-are-optimal/documents/qlc-nand-workload-guide.pdf

FineWeb paper (Penedo et al.), Hugging Face; growth since 2024 from the FineWeb dataset card (huggingface.co/datasets/HuggingFaceFW/fineweb), v1.4.0, 2025-07-11 · as-of 2024-05 · https://huggingfacefw-blogpost-fineweb-v1.static.hf.space/index.html

RefinedWeb (NeurIPS 2023), Figure 2 / Table 5 · as-of 2024 · https://papers.neurips.cc/paper_files/paper/2023/file/fa3ed726cc5073b9c31e3e49a807789c-Paper-Datasets_and_Benchmarks.pdf

Zhang et al. (Scale AI), 'A Careful Examination of Large Language Model Performance on Grade School Arithmetic', NeurIPS 2024 Datasets and Benchmarks · as-of 2024 · https://arxiv.org/abs/2405.00332

Meta model dimensions published 2024; primary paper. BF16 storage and ideal KV-head sharding are guide assumptions, not a serving measurement. · as-of 2024-07 · https://arxiv.org/html/2407.21783v3

Worked example in Chapter 9.7; assumptions stated in the text. · as-of 2026-09 · https://docs.vllm.ai/en/latest/design/prefix_caching/

SOURCES

Every figure in this deck traces to the guide's dated register (3/3)

DeepSeek-AI, DeepSeek-V2 Technical Report · as-of 2024-05 · https://arxiv.org/abs/2405.04434

SOURCES

About these figures

Figures are resolved by reference from the guide's claim register at build time (Second edition · September 2026 (first published July 2026)) — a number appears here only as it appears, dated and sourced, at aidatacenterguide.com. Newer data may exist: check the live register.