



| SCALITYDAY26

AI inference



Erwan Girard
CPO
Scality



Giorgio Regni
CTO
Scality



Pierre Derome
VP AI Ops
Scality

SUMMARY

This year inference became the workload

2026



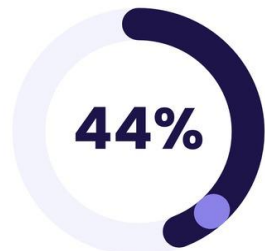
2027



AI-optimised cloud spend, Gartner, August 2026

ADOPTION

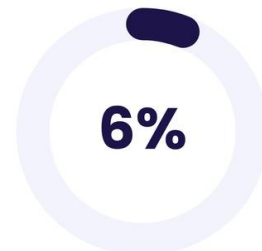
Past the pilot, into the estate



scaling AI
from 38% a year ago



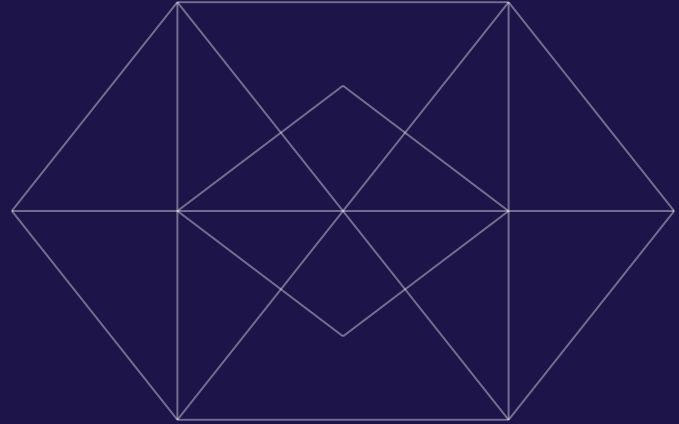
scaling AI agents
from 27% a year ago



EBIT impact over 5%
unchanged on last year

McKinsey, The State of AI 2026 — n=1,719 across 97 countries. The small mark is a year ago.

In 2026 AI inference just
became an IT service

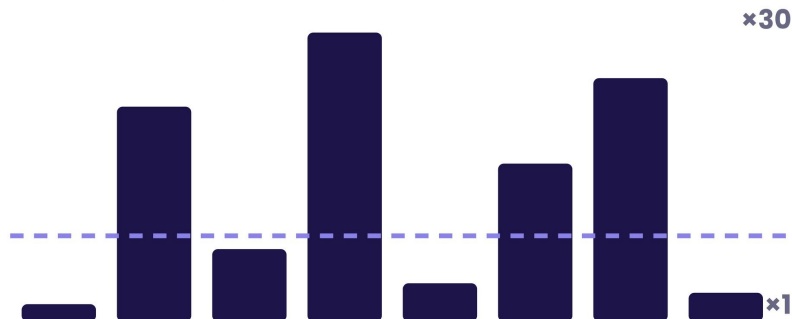


From FOMO to ROI:

AI Inference – predictability

Transformation from discovery and prototype to IT fundamentals:

The same task



- - - what the model forecast it would use

Operating within SLAs and a budget



Inference is here to stay

AI Inference – freedom

AI Inferencing is not a bubble - needs the long term view

Keep freedom across so many variables

GPU Generations and brands

Network generations

Servers & PCIe Generations

Architectures

Storage

Usecases



AI inference is not ONE use case

AI Inference – discrete use cases

known and unknown

Assistants and agents over internal data

Coding Agents for developers

Agentic Frameworks

Video production / customization

Video interpretation



2 different targeted customer classes

AI Inference – highly customisable

Internal: empower employees

best in class

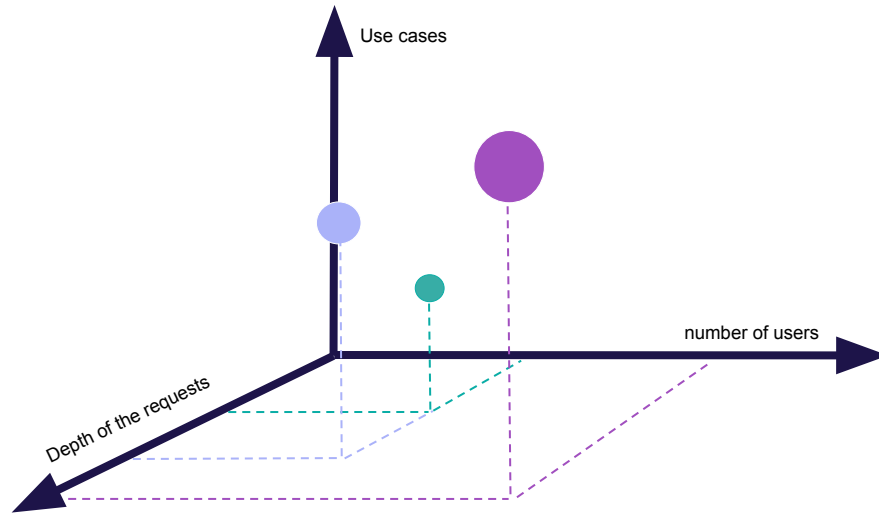
External: selling AI enabled services

Predictable answers and cost



AI Inference Scale out

AI Inference – Sizable and Scalable



AI Inference – Reliability

Financial

as critical as fraud
detection

Transportation

as critical as fare filling
and pricing

Health

as critical as running
PACS

Service Providers

as critical as running
IaaS

Energy

as critical as
infrastructure monitoring

Government

as critical as nation-wide
services

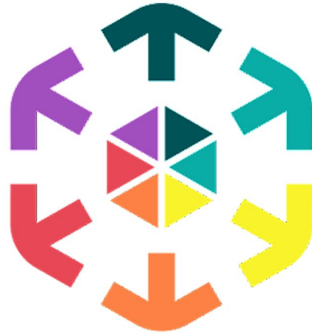


does this sound like ?

Scalability

Freedom

Resilience



Predictable

Performance

Relentless support



Scality AI Inference Factory

Endpoint connect client workloads to Scality's on-premise LLM inference stack

CLIENT HUMAN ENVIRONMENT

CLIENT HARNESS

OpenAI-compatible (OpenCode, Hermes, Goose)

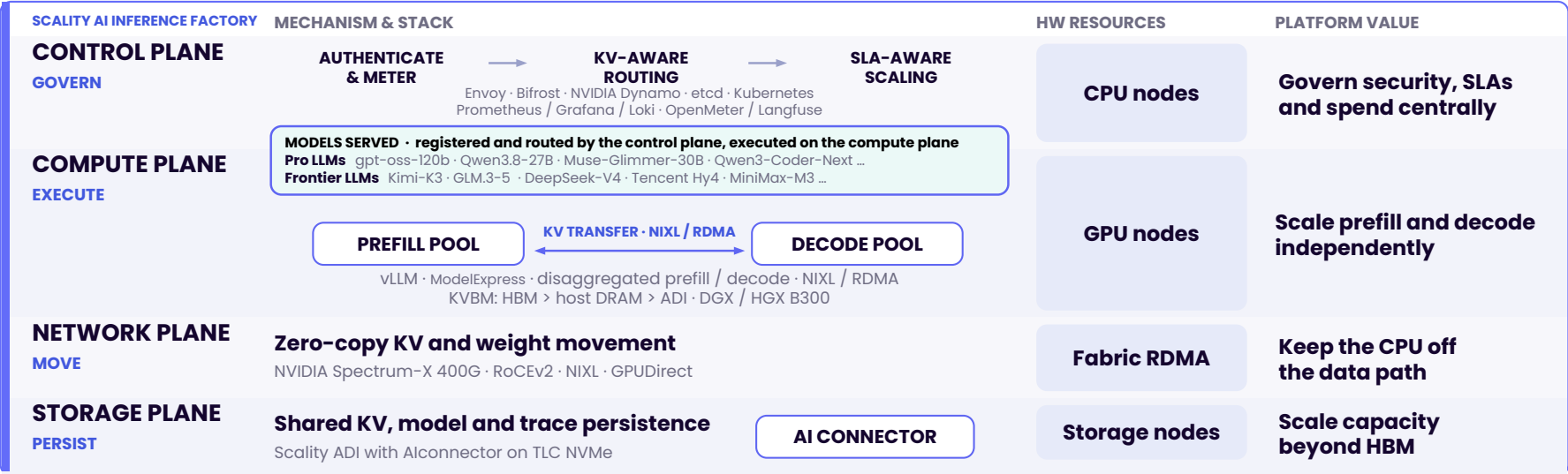
CLIENT AGENTIC ENVIRONMENT

CLIENT AGENTIC FRAMEWORK

LangChain / LangGraph · Deep Agents · OpenAI Agents SDK · Microsoft Agent Framework · Pydantic AI

CLIENT VALUE

Keep existing assistants, apps and agents



HARDWARE	GPU	CPU	Servers	Network	Storage	BOMs and Architectures
HOSTING	NVIDIA/AMD...	INTEL/AMD	SMC/HPE/Lenovo / DELL	CX7/ CX8	TLC / QLC / HDD / Tape	

THE STAKE

Control cost, data and inference

Cost sovereignty · Data sovereignty · Inference sovereignty

COST SOVEREIGNTY

~40% LOWER TCO

Deterministic cost

Fixed and provisioned before usage

API pricing affects adoption

+1% API cost is associated with -0.29% usage

Removing visible per-token cost supports broader adoption — and productivity.

DATA SOVEREIGNTY

External APIs can receive the full working context

Codebase · documents · retrieved data · tool outputs

Moderation can extend retention

Flagged inputs / outputs: up to 2 years
Safety scores: up to 7 years

Jurisdiction remains external

U.S. providers are subject to the CLOUD Act

Legal conflicts are possible

Third-party or NDA-protected IP may not be shareable with external LLMs

INFERENCE SOVEREIGNTY

Model availability remains provider-controlled

Access can be tied to new provider conditions — e.g., Fable 5 retention requirements

Provider-side quantization is not customer-controlled

Load-driven quantization is opaque to the customer

Safety routing can change serving behavior

Fallback policies can change which model serves a request





SCALITY

RELIABLE. SECURE. SUSTAINABLE.

Why does a storage company invest in AI inference?

1

Scality Day 26, Paris.

Giorgio Regni CTO **Scality**



AI is a memory problem.

"There's a whole new category of storage systems ... because this is a pain point for just about everybody who does a lot of token generation today." Jensen Huang, NVIDIA CEO, CES 2026.



What I hear in Silicon Valley

//

In agentic systems, **KV cache** effectively becomes the model's long-term **memory**.



MARCH 2026

NVIDIA GTC, San Jose

NVIDIA

Context memory platform

ATTENDING

//

The next constraint in AI will not be compute alone. It will be **memory**.



AUGUST 2026

FMS, Santa Clara

Jeremy Werner

SVP, Micron, FMS 2026 series

SPEAKER

//

Scaling the AI **Memory Wall**



SEPTEMBER 2026

AI Infra Summit, Santa Clara

Microsoft, AMD, Micron

Panel title

SPEAKER



OCTOBER 8

Paris, today

NVIDIA developer blog (March 16, GTC keynote day); Micron blog, first of its FMS 2026 series (September 2); panel title from the AI Infra Summit 2026 agenda. Quotes verbatim.

"Every prompt is different."

So there is nothing to cache? Let me show you what my own agent sends.

What my agent sends when I say hello

Call 1

I type one word

"Hello"

The same at the start of every chat

the chat so far

new in this call

Tool definitions: 21 tools

about 12,000 tokens

Instructions

about 3,000 tokens

Skills, memory, clock

about 4,000 tokens

18,740

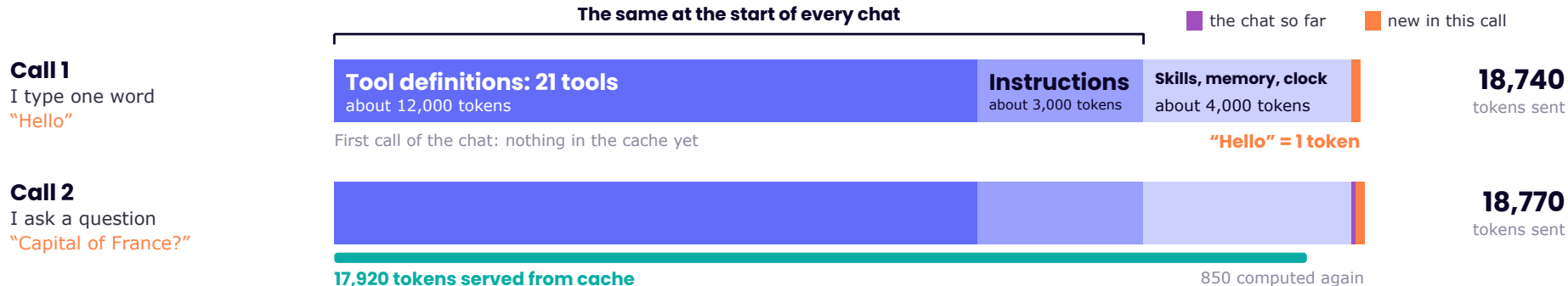
tokens sent

First call of the chat: nothing in the cache yet

"Hello" = 1 token

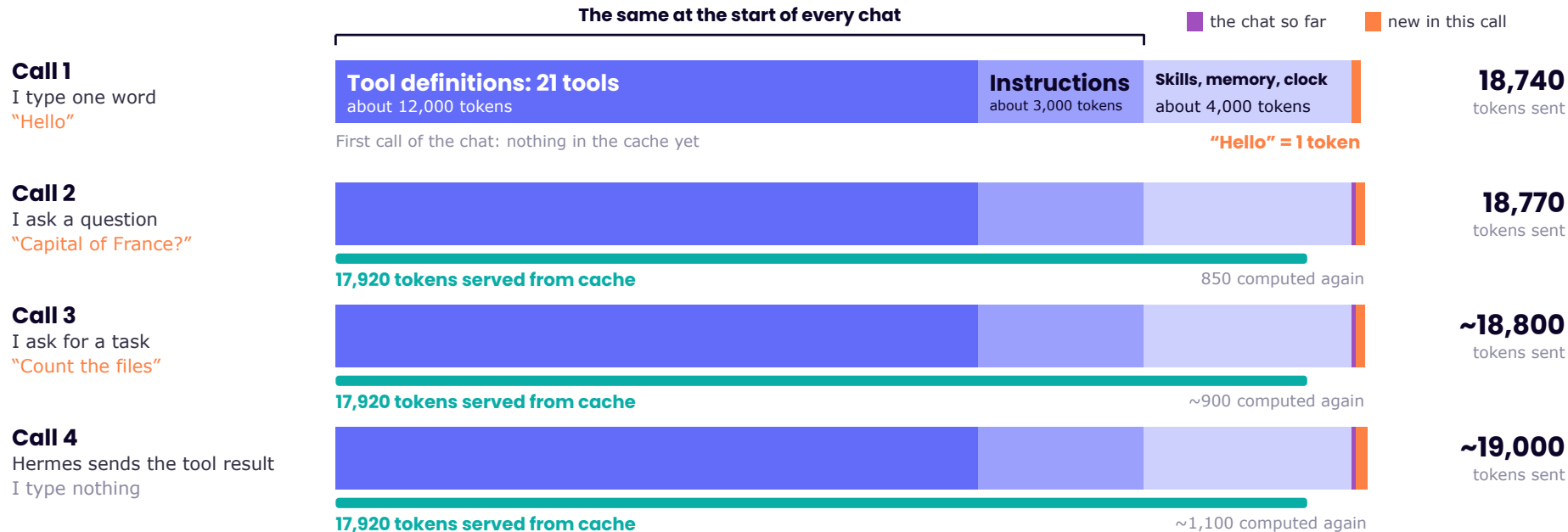
Hermes agent 0.20.1 on my desktop, model gpt-5.6-luna, October 2, 2026. Totals and cache hits as reported by the provider. Segments counted with the o200k tokenizer; calls 3 and 4 are one turn, split by message size.

What my agent sends when I say hello



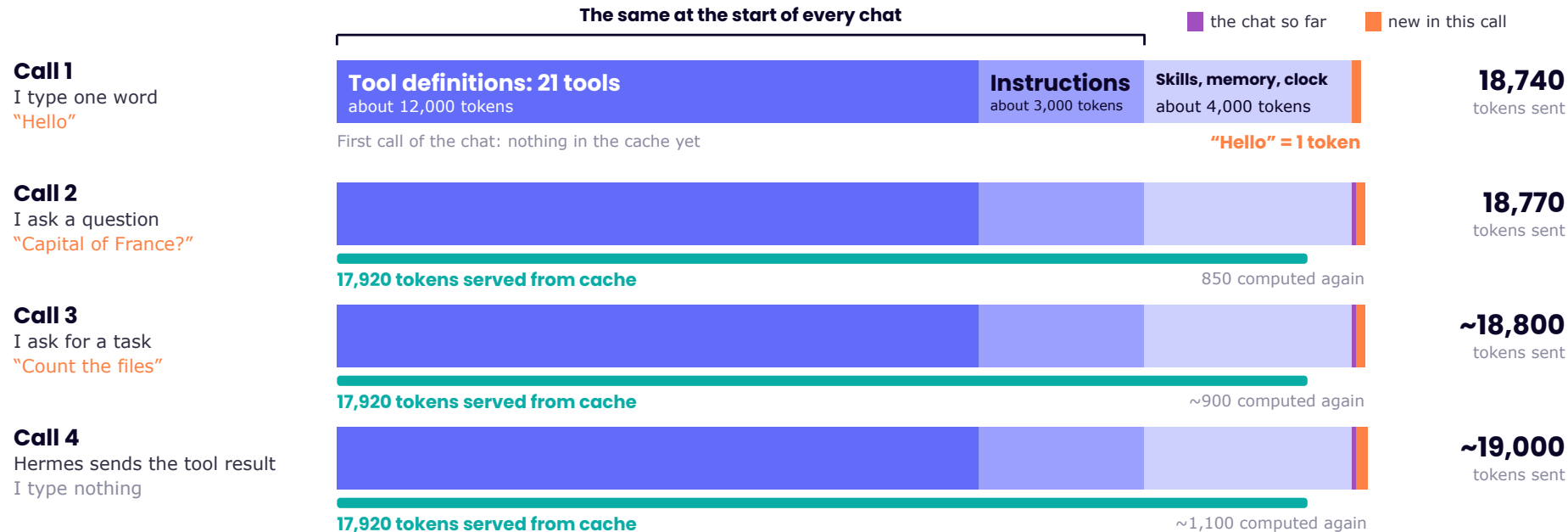
Hermes agent 0.20.1 on my desktop, model gpt-5.6-luna, October 2, 2026. Totals and cache hits as reported by the provider. Segments counted with the o200k tokenizer; calls 3 and 4 are one turn, split by message size.

What my agent sends when I say hello



Hermes agent 0.20.1 on my desktop, model gpt-5.6-luna, October 2, 2026. Totals and cache hits as reported by the provider. Segments counted with the o200k tokenizer; calls 3 and 4 are one turn, split by message size.

What my agent sends when I say hello



47

tokens I typed

75,345

tokens Hermes sent, 4 calls

53,760

tokens served from cache

Hermes agent 0.20.1 on my desktop, model gpt-5.6-luna, October 2, 2026. Totals and cache hits as reported by the provider. Segments counted with the o200k tokenizer; calls 3 and 4 are one turn, split by message size.

The one number to watch: KV Cache hit rate

Cache hit rate: how much of each request the GPU has already computed.



We run alerts on our prompt **cache hit rate** and **declare SEVs** if they're too low.

Anthropic, Claude Code

Thariq Shihpar, February 2026

A SEV is a production incident.



The KV-cache hit rate is the **single most important metric** for a production-stage AI agent.

Manus, now part of Meta

Yichao Ji, July 2025

About 100 input tokens per output token.



Prompt caching may store encrypted key/value tensors in **GPU-local storage** as application state.

OpenAI

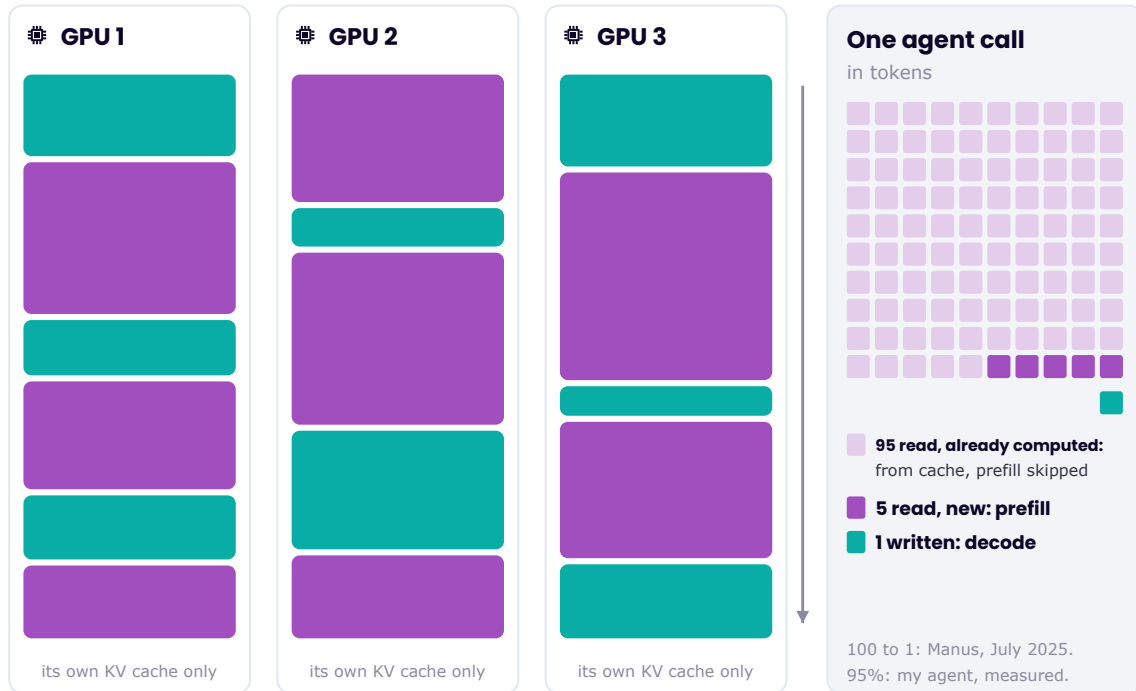
Prompt caching guide, 2026

Cached prompts kept up to 24 hours.

Thariq Shihpar, Anthropic, quoted on simonwillison.net, Feb 20, 2026. Yichao Ji, "Context Engineering for AI Agents: Lessons from Building Manus", July 18, 2025. OpenAI prompt caching guide.

Today, every GPU does two jobs

- Job 1, prefill:** reads the prompt, builds the KV cache
- Job 2, decode:** writes the answer, one token at a time



Why it wastes GPUs

- 1 Prefill interrupts decode.**
Each new prompt stops the answers already being written. Their tokens stutter.
- 2 GPUs run far below their limit.**
To keep the latency promise, every GPU keeps headroom it never uses.
- 3 The memory is trapped.**
Each GPU keeps only its own KV cache. A returning chat finds that GPU, or recomputes.

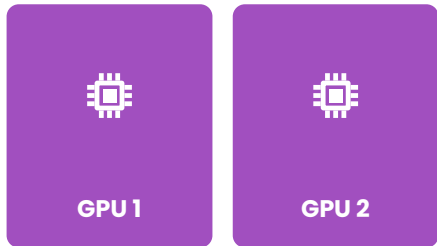
Mixed serving, how most clusters run today. Time runs down each GPU. One agent call: about 100 tokens read for 1 written (Manus); 95% of the input from cache after the first call (my agent, Hermes).

Split the GPUs: prefill and decode

- Job 1, prefill:** reads the prompt, builds the KV cache
- Job 2, decode:** writes the answer, one token at a time

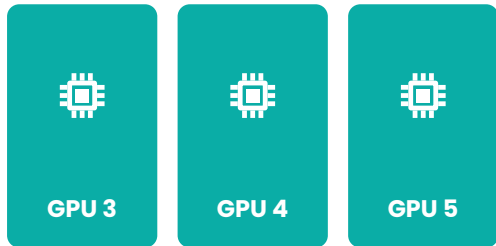
Prefill GPUs

compute-heavy, flat out



Decode GPUs

memory-heavy, batches wide



save the KV cache

restore it

Scality ADI: the shared KV cache

Every conversation's memory, reachable by every GPU, over RDMA.



Why it is more efficient

- 1 No interruptions.**
Each pool does one job, at full speed, all the time.
- 2 Each pool sized for its job.**
More prefill GPUs for long prompts, more decode GPUs for more users.
- 3 One shared memory.**
Any decode GPU picks up any conversation from the KV cache on Scality ADI.

Published gains: up to 7.4x more requests per GPU (DistServe, OSDI 2024); 107% more requests on H800 clusters (Moonshot's Kimi, Mooncake, FAST'25).

Split serving: from a paper to production

Prefill and decode on separate GPUs,
with the KV cache moving between them.

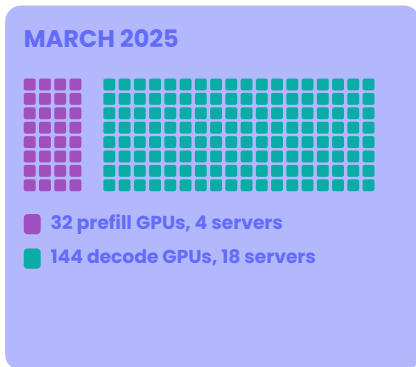
Moonshot AI, Kimi

Mooncake paper: split
pools, one shared
KV cache.



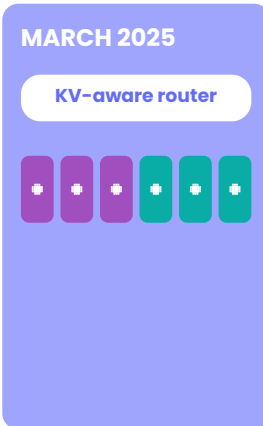
DeepSeek

Publishes its serving
system. Its two pools,
to scale:



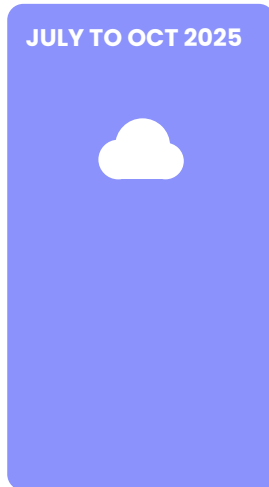
NVIDIA

Dynamo: split serving
as open source.



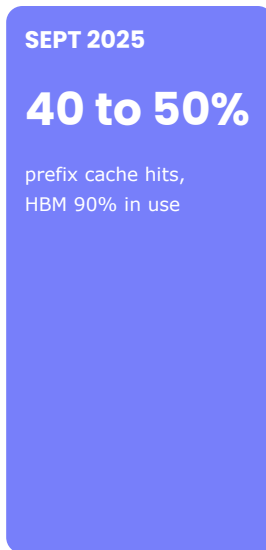
AWS, Google Cloud, Azure

Split-serving recipes
for their customers.



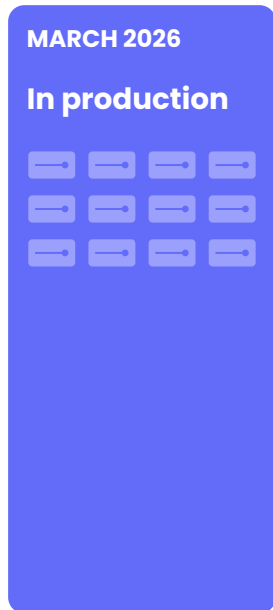
Meta

Split serving for
"large scale Meta
traffic".



Dynamo 1.0

"Deployed in production":
Baseten, ByteDance,
CoreWeave and more.

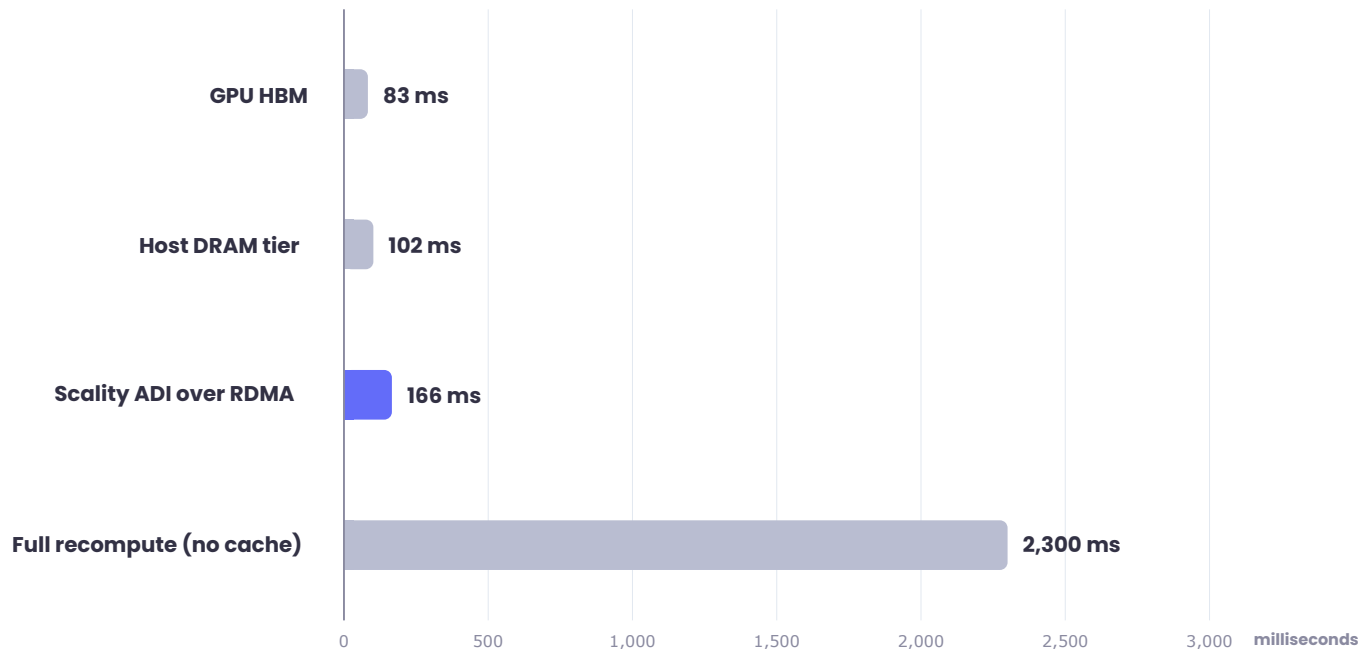


Mooncake (arXiv, June 2024; FAST'25). DeepSeek, March 1, 2025. NVIDIA Dynamo, March 18, 2025 and March 16, 2026. AWS, July 15, 2025; Google Cloud, Sept 10, 2025; Azure AKS, Oct 24, 2025. Meta, PyTorch blog, Sept 12, 2025.

166 ms, from object storage

Warm time-to-first-token, by the tier serving the returning context

Gemma-3 27B, ~14K-token context, KV block size 256. Lower is better.



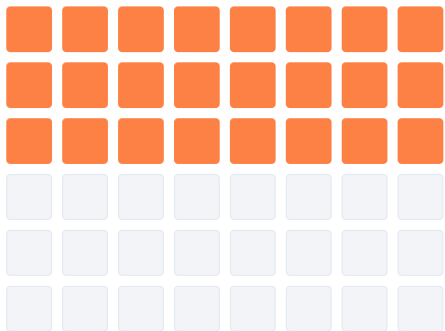
Scality lab, August 2026.

RDMA: Faster transfer, much lower storage CPU usage

S3 over TCP

one 100 Gbit/s link

4.6 GB/s



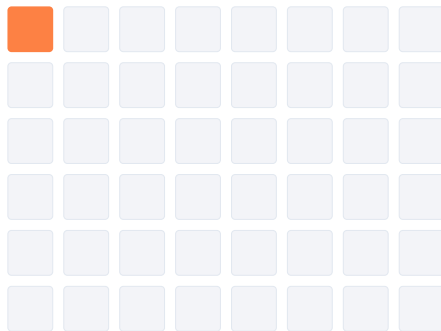
24 of 48 cores busy

51% CPU

ADI over RDMA

one 100 Gbit/s link

9.6 GB/s



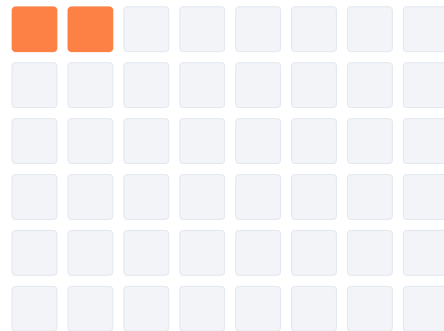
1 of 48 cores busy

2.2% CPU

ADI over RDMA

four 100 Gbit/s links

42.5 GB/s



2 of 48 cores busy

3.7% CPU

Scality lab, August 2026. GETs of 2 GiB objects, 32 in flight. Each square is one of the storage server's 48 cores; busy cores = CPU utilisation x 48. ADI over RDMA uses the native API.

Loading a model: 1.9 s from ADI, 18.3 s from local NVMe

Seconds to load Gemma-3 27B into one GPU

lower is better

ADI over RDMA

four links



1.9 s

ADI over RDMA

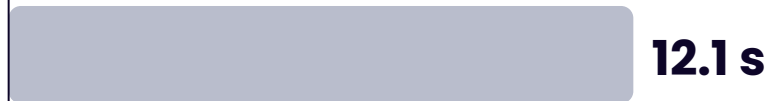
one link



5.6 s

S3 over TCP

one link



12.1 s

Local NVMe

inside the GPU server



18.3 s

Scality lab, August 2026. Gemma-3 27B in bf16, weights loaded into one GPU (report test 2b). ADI over RDMA uses the native API. Local NVMe: the GPU server's own drives.

It's in our company's DNA

Stateless applications, all state in distributed storage system.

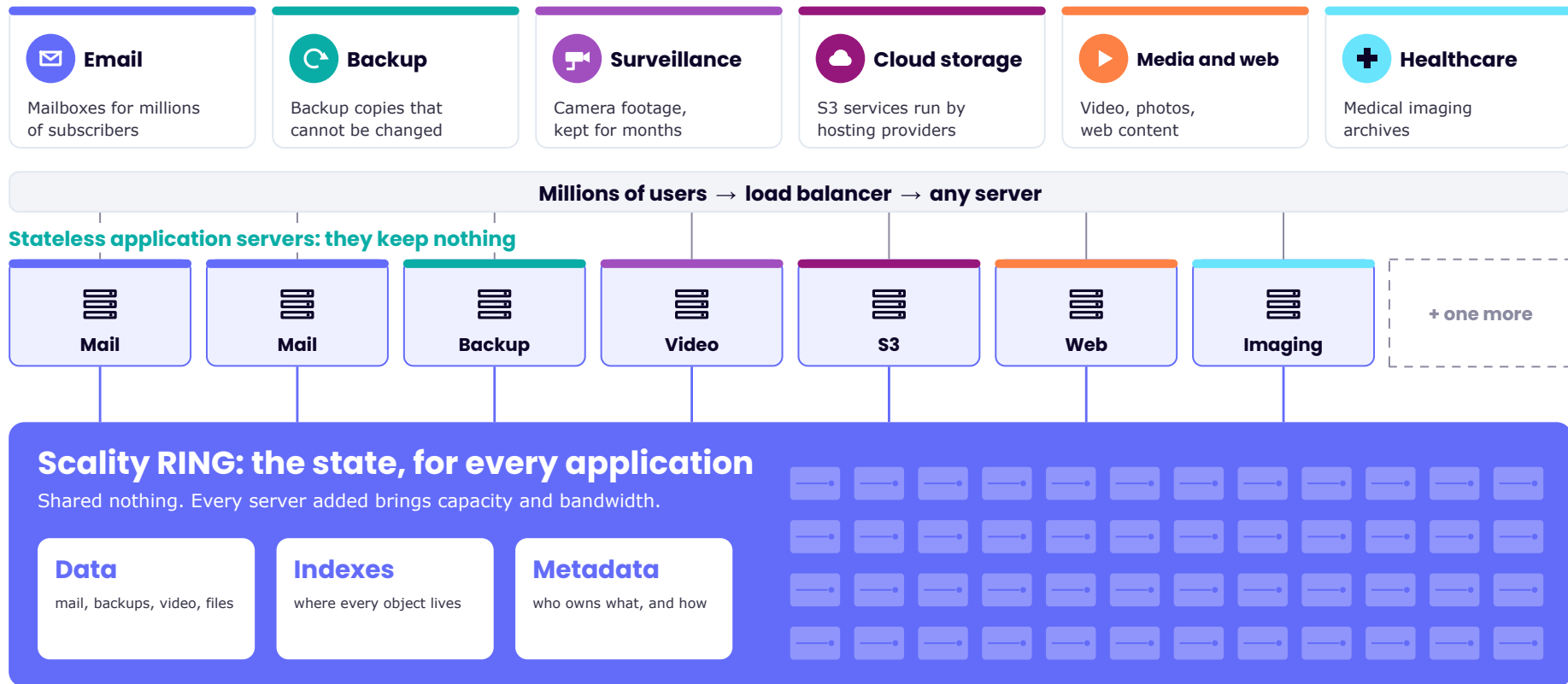
Scality, application to OSEO, December 2, 2010

Problèmes	Exigence
L'association forte des utilisateurs vers un serveur applicatif et un serveur de stockage entraîne des problèmes opérationnels très important.	Le système doit être complètement « stateless », c'est à dire que tout serveur applicatif peut accéder à n'importe quelle donnée à tout moment, et ceci en parallèle.
Les utilisateurs doivent être migrer avec toutes leur données régulièrement afin d'équilibrer la charge.	L'ajout d'un serveur de stockage doit bénéficier à l'ensemble du système et pas seulement à une seule ou un groupe de machines.
Augmenter la capacité se fait par pallier et nécessite de migrer des centaines de téraoctets vers le nouveau système.	

“The system must be completely stateless: any application server can access any data at any time, and in parallel.”

Page 11, problems our email customers raised, and our requirement.

Why Scality: we keep your applications' state



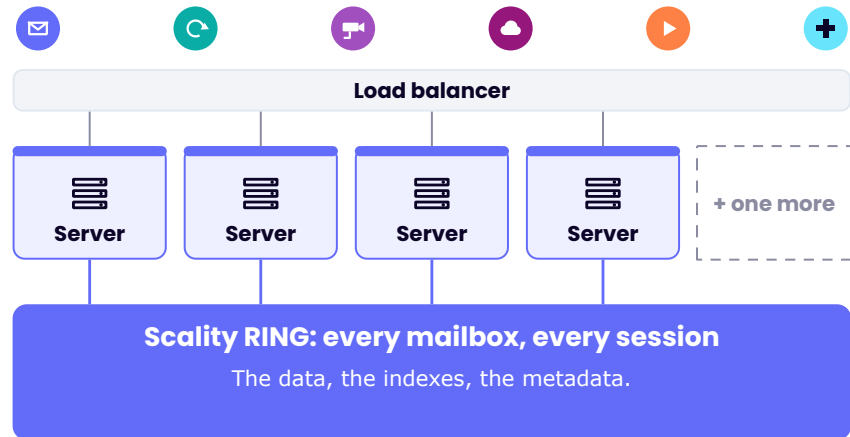
Scality RING since 2010: email at service providers first, then backup, video surveillance, cloud storage, media and healthcare.

Why Scality: we keep your applications' state

2010: web services

Email first, then every workload

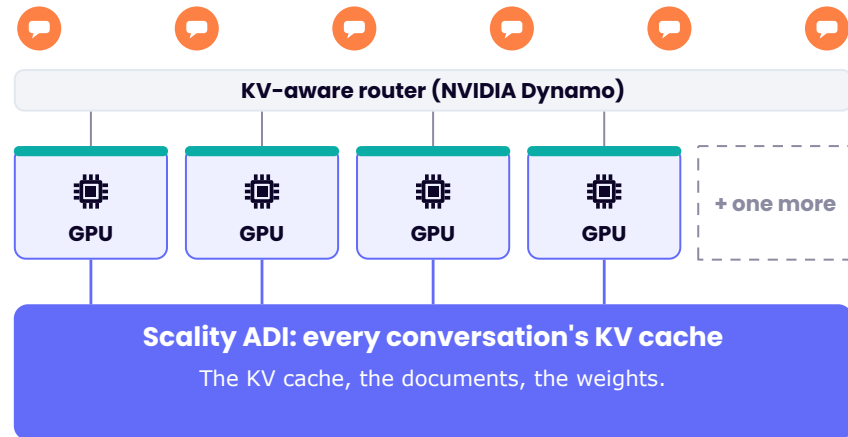
Any server serves any user. Add one: nothing to move.



2026: AI inference

Chats and agents

Any GPU serves any conversation. Add one: nothing to move.



Our patents on this pattern

2008 US 8,429,444 · EP 2,377,294

Peer-to-peer ring with failover:
add servers, no central point.

2014 US 9,524,302

A consistent database inside
the object store. Cited 113 times.

2025 WO 2026/198810, pending

Application session state
on an object storage service.

+17 more

20 patent families,
22 granted, 2008 to 2026.

Patents: Scality's own filings, public record (Google Patents, European Patent Register, WIPO), read September 2026.

We bring hyperscaler technology to the enterprise, again

2010 Storage

The giants built it:

Google File System, 2003

Amazon Dynamo, 2007

Facebook Haystack, 2010



Standard servers, cheap disks, shared nothing



Scality RING

Left: the giants' published storage designs (Google File System, SOSP 2003; Amazon Dynamo, SOSP 2007; Facebook Haystack, OSDI 2010). Right: the split-serving designs of slide 12; Anthropic: prompt caching, the KV cache as a production tier.

We bring hyperscaler technology to the enterprise, again

2010 Storage

The giants built it:

Google File System, 2003

Amazon Dynamo, 2007

Facebook Haystack, 2010



Standard servers, cheap disks, shared nothing



Scality RING the same architecture, in your data center

2026 Inference

The giants built it:

Mooncake, 2024

DeepSeek, 2025

Meta, 2025

AWS, Google Cloud, 2025

Anthropic, 2026



Split GPU pools, the KV cache in storage



Scality ADI the same architecture, in your data center

Left: the giants' published storage designs (Google File System, SOSP 2003; Amazon Dynamo, SOSP 2007; Facebook Haystack, OSDI 2010). Right: the split-serving designs of slide 12; Anthropic: prompt caching, the KV cache as a production tier.

We bring hyperscaler technology to the enterprise, again

2010 Storage

The giants built it:

Google File System, 2003

Amazon Dynamo, 2007

Facebook Haystack, 2010



Standard servers, cheap disks, shared nothing



Scality RING the same architecture, in your data center

2026 Inference

The giants built it:

Mooncake, 2024

DeepSeek, 2025

Meta, 2025

AWS, Google Cloud, 2025

Anthropic, 2026



Split GPU pools, the KV cache in storage



Scality ADI the same architecture, in your data center

The enterprise way:

Your own data center

Enterprise quality

Security

Uptime

Predictable TCO

Multi-tenancy and metering

24/7 support

Our expertise

Left: the giants' published storage designs (Google File System, SOSP 2003; Amazon Dynamo, SOSP 2007; Facebook Haystack, OSDI 2010). Right: the split-serving designs of slide 12; Anthropic: prompt caching, the KV cache as a production tier.



SCALITY

RELIABLE. SECURE. SUSTAINABLE.

SCALITY DAY · 8 OCTOBER 2026

Taking a real RING support ticket through Inference Factory

Pierre Derome

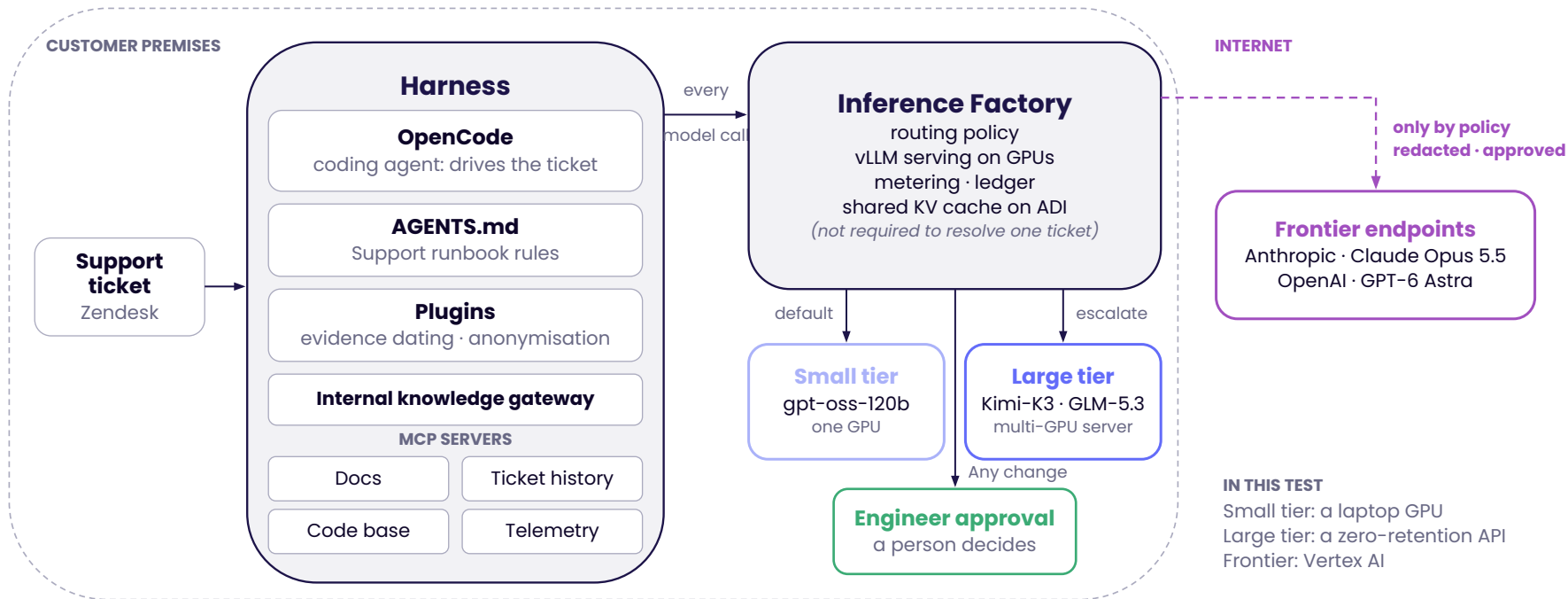
VP AI Ops

SCALITY



ARCHITECTURE

Your harness, your knowledge, your GPUs



SETUP

The rules: AGENTS.md

Support troubleshooting rules

From Scalify Support's runbook. The harness applies them to every ticket, on every tier.

BEFORE TROUBLESHOOTING

- R1** Know the context first: organisation, platform, product, version and environment.
- R2** The customer's entry point is a symptom, not the cause.
- R3** The customer's priority and diagnosis are suspects until confirmed.

EVIDENCE

- R4** Verify every piece of customer data: the right file, complete, and captured on the affected platform during the incident.
- R5** Evidence with no provenance (what ran, where, when) can justify a question, not a conclusion.
- R8** **Collect at both ends of a path, and beyond the items the customer names.**

BEFORE ACTING

- R6** Separate consequences from causes.
- R7** Not enough data? Ask for it before recommending any action.

HOW TO WORK A TICKET

- 1. Triage first.** List every piece of evidence: what it shows, when it was captured, whether it covers the incident, and how it may be used (conclusion, question only, discard).
- 2. Then diagnose,** using only the evidence marked "conclusion".

AGENTS.md

WHY IT MATTERS

Plain text the customer owns. The rules are a file in the customer's own repository, reviewed like any other code. Change the model and the rules stay.

Every tier reads the same file. Small, large and frontier models get the same rules; what differs is how well they apply them.

The method makes the work checkable. The triage table says when each piece of evidence was captured and whether it covers the incident. The gateway policy reads that table.



THE TICKET

A ticket arrives: replication is slow

[SITE-A] CRR takes more than two hours to finish

Production · RING 9.3.0.3 · S3 connector · cross-region replication from SITE-A to SITE-B · a customer

Customer, 07:20

We have had a few days where the CRR is taking longer than usual to finish. One example is the bucket prod-bkt-01, prod-bkt-02, ... prod-bkt-43. I am attaching information that I think may be useful. I executed the script `systemctl start s3c-oneshot@s3utils_crr.service` to relaunch all objects in failed state.

Customer, 07:39

The only anomaly I see is the lag of 740,380 entries in the Raft 8 session.

Customer, 08:30

Due to the critical nature of this client, please escalate the case to P2.

Attachment: 110 KB of command output: Kafka consumer lag, Raft session offsets, replication processor logs.

Account history: a ticket with the same subject on the same platform was resolved as a known Kafka offset-rewind issue.



STEP 1 • SMALL TIER

The Small model tries first

- The harness strips the ticket and redacts customer details
- gpt-oss-120b drafts the triage on one GPU, on-prem
- Routine tickets with a documented answer stop here
- This ticket is beyond the model capability:
 - Draws conclusions from the Raft lag from the old attachment
 - Asks for fresh data from one site only

Escalates to Large model

- A slowdown between two sites can start at either end
- AGENTS.md: collect fresh evidence at both ends
- The draft collects at one site only
- Checked by code, not by a model
- Escalated to the Large tier, still on-prem



STEP 2 • LARGE TIER

The Large model discounts old data...

- Dates the attachment to June 2025: its figures are not current
- Searches support notes and code base through the Internal Knowledge Gateway (MCP)
- Clear recommendations:
 - Stop re-queuing failed objects: each run adds load
 - No restarts, and no fix before fresh data
- Asks for fresh data from both sites
 - Receiving site: errors, timeouts and slow responses

Escalates to a Support Engineer

- It recommends an action to the customer
- It asks the customer for more data
- An engineer reviews both and sends them



Large model identifies the noisy neighbour

Every action goes to a person: the customer approves the pause, engineering confirms the fix

What it sees

Every 500 on all 18 receiving connectors is on prod-bkt-47, a bucket not on the customer's list.

Keys of 5,000 characters. A fixed 110s timeout, then failure.

Cause, from code

The 110s is two 55s socket timeouts.

The code points to Vault's request-size limit, absent on 9.3 and raised in 9.5.

Contain now

Pause replication on prod-bkt-47 only, for the customer to approve.

- Cost: no DR copy for that bucket until the fix.
- No re-queue, no restarts.

Permanent fix

- Upgrade both sites to 9.5.1: it rejects keys over 915 bytes by default.
- A Support Engineer first tests prod-bkt-47's real key lengths on 9.5.1.
- Then set the override, or the customer shortens its keys.



Opus 5.5 marginally better

Likely Cause

The key overflows Vault's 16 KB limit

Preserves Settings

Saves replication settings so the pause can be undone

Engineering Stopgap

Stopgap for engineering: raise Vault's limit on 9.3



THE LEDGER

The ledger for this ticket

The hard ticket is resolved on prem

Call	Tier	Tokens in / out	Cost (proxy)	Left the premises
Knowledge via MCP, dating plugin	Harness	n/a	\$0	nothing
First pass	Small · on-prem	80k / 6k	\$0.004	nothing
Escalated triage, GLM-5.3	Large · on-prem	573k / 66k	\$0.45	nothing
Fix after the screen share, GLM-5.3	Large · on-prem	1.39M / 12k	\$0.17	nothing
Total	Open Weight - on prem		\$0.63	nothing
Same two steps on Opus 5.5	Frontier · internet	2.66M / 41k	\$11.44	<i>redacted ticket and notes it read</i>

Small and Large tier costs are proxies: OpenRouter list prices for the same open weights. On-prem, you pay for GPU capacity, not tokens. Frontier: Vertex AI list price, about 18 times the Large tier for the same two steps.



TAKEAWAYS

Own your harness and models, Stay human

01

Harness leverages AI

Provides **enterprise data** context to models:

- rules
- documentation
- tribal knowledge
- code base
- processes

02

Harness is code

Not AI:

- python code
- text files (.md)
- Rag (semantic search)
- MCP
- deterministic

03

Model size drives intelligence

On this ticket, the Small tier missed the trap and the culprit bucket. The Large tier caught both.

04

Open-weight caught up with Frontier

On triage and containment, the Large tier matched Opus 5.5, at about 1/18 of the cost.

05

A person decides

We believe a human must stay in the loop: every action reaches a Support Engineer before the customer.





THANK YOU

SCALITY
DAY26