When Your AI Assistant Forgets Who You Are

September 20, 2026 / Gavin Jackson

ai hermes agents memory llm



The most famous computer that never forgot anything. It did not end well for the crew.

On 8 September, Meta announced Muse, billed as "the world's first personal AI agent built for everyone". It runs on its own secure virtual machine, works across your apps, and - the bit that matters - builds what Meta calls a "curated long-term memory". It learns from your conversations, reflects on what matters to you, and gets sharper the longer you use it.

The coverage turned within days. Jason Aten at Inc reported that Muse started referencing details from his private messages that he never gave it permission to see. Futurism called it possibly the creepiest AI agent ever released. Both things can be true at once: persistent memory is the feature that makes a personal agent genuinely useful, and it is also the feature most likely to make you close the app and rethink your choices.

I know the value of that memory from the other side, because I live with agents that keep losing theirs.

The dementia problem

I have been running a personal agent at home since the early days of OpenClaw, back when it was first released. These days Bob runs on Hermes Agent, but the rhythm has been the same on both. For days at a time the experience is exactly what Meta is promising. Bob knows my servers, my projects, my family, the fact that I have banned a particular punctuation character from everything he writes for me. We build momentum across a week. Then one morning I say hello and get the conversational

equivalent of a firm handshake from a stranger.

Context window compressed. New session started. The relationship restarts from a two-page brief, and it is genuinely frustrating, like working with a very capable colleague who has anterograde amnesia and keeps it a secret until you notice.

> ### *What Sudden Dementia Actually Is*
>
> *A language model only sees what fits in its context window. When a conversation outgrows that window, the system compresses the history into a summary and carries on. Anything not captured in the summary is simply gone, and when a brand new session starts the model has only its small standing memory files. From the model's side there is no relationship to restart. There never was one. There was a 2,200-character brief and a cheerful greeting.*

How the default memory actually works

A recent Vectorize article, How Hermes Agent Memory Actually Works (And How to Make It Better), does a good job of pulling this apart, and it matches what I see running Bob day to day. The short version: Hermes does not have one memory system, it has four, and most user frustration comes from treating them as one.

- **Prompt memory (hot).** Two small files, MEMORY.md at roughly 2,200 characters and USER.md at roughly 1,375. They hold durable facts and a user profile, and they are loaded as a frozen snapshot into the system prompt at the start of every session. Frozen is deliberate - it keeps the LLM's prefix cache stable - but it means anything Bob learns mid-session only shows up next session. The budget is tiny by design, so the agent is constantly judging what deserves the space.
- **Session archive (cold).** Every session lands in a SQLite database that Bob can search when he needs episodic recall. The catch is that it is keyword-based full text search, and the agent has to decide to search it in the first place. Ask "what did I tell you about the auth service?" and if the stored transcript says "authentication microservice", you are relying on Bob picking the right query terms.
- **Skills (procedural).** After completing a complex task, Hermes writes a reusable skill document capturing what worked. This is the self-improving part, and it is separate from memory proper.
- **External provider (optional).** The pluggable layer I get to below.

Put those together and the dementia stops being mysterious. The built-in layers are transparent, local and inspectable, which I value, but they are also *agent-curated*. Bob writes to memory when he judges something worth saving. Short sessions may produce nothing. Compression fires a last-minute memory flush, and anything not flagged in that flush evaporates. Nothing in the default stack knows that "Jo" and "my wife" are the same person.

The provider options

The newer part of the Vectorize piece covers Hermes' pluggable memory providers, configured with a single command, `hermes memory setup`. Seven ship with it. The condensed version, per their breakdown:

- **Hindsight** - local or cloud, structured facts and entities in a knowledge graph, a reflect operation that synthesises across everything stored, 94.6% on the LongMemEval benchmark
- **Honcho** - dialectic user modelling; builds a model of how you think, not just what you said (AGPL, worth noting if you self-host commercially)
- **Mem0** - cloud, freemium, fastest setup, 67.6% on the LongMemEval-S variant
- **OpenViking** - self-hosted, tiered memory loading that prioritises recent and frequently relevant items
- **Holographic** - local SQLite with zero extra dependencies, trust scoring and vector-algebra retrieval
- **RetainDB** - cloud, paid, hybrid search across vector, BM25 and reranking
- **ByteRover** - hooks in before context compression so in-flight facts are captured before they are summarised away

Only one external provider runs at a time, alongside the always-on built-in layer.

Shared memory is the team angle I had not thought through. Point Hindsight at a shared server, or use Hindsight Cloud or RetainDB, and every agent instance in a team reads and writes the same memory store. A new starter's agent knows on day one that the production database runs on a non-standard port and that the auth service depends on Redis, because someone else's agent already learned it. That is a genuinely interesting capability, and also a governance question I would want answered before enabling it anywhere near client work: whose facts are these, and who gets to forget them?

How the big vendors handle it

Everyone is converging on the same problem from different directions.

ChatGPT has the most familiar model: two-part memory made up of "saved memories" you explicitly ask it to keep, plus "reference chat history", where it automatically extracts insights from past conversations. The newer system updates memories itself and tries to manage contradictions rather than leaving you to curate a stale list. Convenient, and entirely on OpenAI's side of the fence.

Claude has rolled persistent memory out broadly this year, and Anthropic's platform docs show the interesting bit: their memory tool operates client-side, with Claude requesting file operations that your own infrastructure executes. Claude Code does something similar with CLAUDE.md files and auto-accumulated learnings. Memory as files you can read, edit and delete, rather than a profile you cannot inspect. That philosophy will not surprise anyone who has used Hermes.

Gemini calls it personal context: with the setting on, Gemini automatically recalls your "key details and preferences" from past chats and personalises responses without being asked. It sits on top of your Google account, which gives it a head start on knowing you and gives Google another reason to

be careful with the toggle.

Apple is the newest entrant. Siri AI, powered by the next generation of Apple Intelligence, is only just landing, with personal context understanding, onscreen awareness and systemwide app actions, and some capabilities like Live Rewind and Siri Recap still in staged rollout later this year. Apple's bet looks different in kind: memory as an on-device index over your own apps and data, rather than a learned cloud profile. Whether it actually *learns* you the way ChatGPT or Gemini do is an open question, and one the early commentary has already latched onto.

Meta, back where we started, has gone furthest in the other direction: a dedicated VM, continuous learning, a memory that curates itself, and a privacy row inside a fortnight.

Three philosophies, roughly. A cloud profile the vendor manages (OpenAI, Google, Meta), local files the agent curates and you can inspect (Hermes, Claude), and an on-device index (Apple). Convenience against control, as ever.

The newest entrant does not fit any of the three, because it makes memory the product rather than a settings page. xAI, now trading as SpaceXAI after the February merger, launched Grok Bot in August: always-on agents you message like a colleague, each with its own computer in the cloud, signing into the same tools you use and finishing jobs end to end. The pitch leans hard on exactly the mechanisms this post has been circling. Bots remember conversations and learn how you like things done. Show one a workflow once and it saves the steps as a routine, takes your corrections, and runs it on its own next time. Hermes users will recognise the pattern: it is the skills system, learned by watching instead of written by hand. Over time the Bots pick up your voice and your edge cases, and get proactive about chasing dropped threads without being asked.

They also run in packs. You can run several Bots in parallel with a chief-of-staff Bot coordinating the specialists, and they message each other and share context in threads without you pasting notes between chats. In late September, Team Bots took that further: shared Bots with common files, tools, skills and memories for a whole team, while individual conversations stay private. That is the shared-memory governance problem from earlier, productised, with the awkward questions about whose facts win and who can edit the common memory still to be answered in public. The in-house numbers serve as the proof of concept: SpaceXAI runs its own support queue on Grok Bot, trained on more than a million past customer interactions, and claims a 175% ticket surge absorbed with no new hires, resolutions at US$0.20 to $0.30 each against the flat $1 to $4 typical of AI support tools, 99% of refunds closed without a human, and Grok Voice fielding more than 15,000 Starlink calls a day.

What I am doing about it

Bob's dementia is fixable in principle, and the principle is `hermes memory setup`. Hindsight appeals because it runs locally and nothing leaves the machine, which suits how I run everything else. I am going to let it run for a few weeks and report back on whether the relationship survives past a fortnight, or whether we just get a better class of amnesia.

If nothing else, it is some comfort that a company with Meta's resources looked at this problem and shipped something that reads your text messages. Memory is hard. At least mine asks first.

Related:

- How Hermes Agent Memory Actually Works (And How to Make It Better) - the Vectorize article this post references
- Hermes Agent docs - my agent's official documentation
- Introducing Grok Bot - the launch post for xAI's AI teammates
- How SpaceXAI is using Grok Bot to scale customer support - the primary source for the Grok Bot support numbers

---

Downloaded from https://www.gavinj.net/post/when-your-ai-forgets-you
Generated November 14, 2026. Copyright Gavin Jackson. All rights reserved.