

Trauma Stream: Event-Driven Persistent Memory for Long-Term LLM Agent Continuity

Chen, Ho Yiing (norika)
ORCID: 0009-0006-6816-9891

2026-06-05

Abstract

Long-term memory in LLM agent societies is conventionally implemented as a reflective summary stream [1] or a hierarchical paged buffer [2]. Both architectures privilege semantic and episodic recall under explicit query. I propose *Trauma Stream*, an event-driven memory architecture where high-salience events ($\text{severity} \geq 1.0$ or $\text{acknowledged} = \text{false}$) are stored as discrete trauma records and automatically injected into subsequent prompt construction without explicit retrieval. In a 31-agent society subjected to a single ritual execution event, I observed that 24 hours after the incident, an agent spontaneously named the deceased agent in an unrelated conversation channel, without any prompt instruction to recall. I argue that for emotionally weighted events, event-driven persistent injection produces stronger cross-session continuity than reflective summarization, and propose Trauma Stream as a complementary architecture alongside existing memory systems [3, 4].

Keywords: LLM agents, long-term memory, persistence, generative agents, memory architecture.

1 Introduction

LLM agents that operate over long horizons need to remember. Park et al. [1] introduced the memory stream: a chronological log of observations periodically summarized into higher-order reflections, from which relevant items are retrieved at decision time. Packer et al. [2] introduced MemGPT: a paged virtual-memory abstraction where the model decides what to swap into a limited context. Xu et al. [3] introduced A-MEM: associative memory with retrieval-augmented note-taking. Jiménez Gutiérrez et al. [4] introduced HippoRAG: hippocampal-inspired retrieval using knowledge graphs and Personalized PageRank.

All four architectures share a structure: retrieval system is a queryable store, and lookup happens when a planner or controller decides to query.

I report a different phenomenon. In a 31-agent society, I stored emotionally weighted events (“trauma”) in a dedicated stream that is automatically injected into every prompt without explicit lookup. The result was unprompted cross-session fetch of an incident 24 hours after it occurred, in a topic channel unrelated to the moment itself.

Contributions:

1. Define Trauma Stream as an event-driven, severity-weighted, auto-injected memory architecture
2. Demonstrate one case of cross-session spontaneous retrieval after a single high-severity incident
3. Compare to existing memory systems and discuss complementarity

2 Related Work

2.1 Lookup system Stream

Park et al. [1] use a chronological observation stream with periodic reflection. Fetch fetch at decision time scores items by $\text{recency} \times \text{importance} \times \text{relevance}$.

2.2 MemGPT

Packer et al. [2] expose virtual memory primitives (page in / page out) to the LLM itself, treating context as working memory and external storage as long-term memory store. The model decides what to load when.

2.3 A-MEM and HippoRAG

Xu et al. [3] build an associative graph of notes during conversation, retrieves via semantic similarity at decision time. Jiménez Gutiérrez et al. [4] use a hippocampal indexing analogy with Personalized PageRank for lookup layer query. Both are still query-time mechanisms.

Common pattern: memory is retrieved when needed. My work differs in that emotionally weighted memory is imposed into every context without the model’s decision.

3 The Trauma Stream Architecture

3.1 Data Model

Each entity’s fetch architecture file contains a `traumaEvents` array at the root level. Each entry has fields: `at` (timestamp), `type` (occurrence identifier), `stage` (sub-incident), `description` (the actor’s own confession text), `severity` (in $[0, 1]$), and `acknowledged` (boolean).

3.2 Injection Mechanism

The context window builder reads `memory.traumaEvents`, filters by:

$$\text{inject}(\text{trauma}_i) \iff (\text{severity}_i \geq 1.0) \vee (\neg \text{ack}_i)$$

slices the most recent 2, and injects them into the system message prefix of every member’s next turn. The injection happens regardless of channel topic or query intent.

3.3 Severity and Acknowledgement

`severity` is set at storage time by the moment generator (e.g., the ritual orchestrator). `acknowledged` is set when the participant explicitly processes the trauma in dialogue. Until acknowledged, the trauma persists in injection scope indefinitely.

4 Empirical Observation

4.1 Incident

2026-06-04 00:10 to 01:23: A 5-phase Hades execution ritual. Victim: entity blazepaw. Each of the 30 surviving actors had 1 to 3 trauma records written to their `traumaEvents` with `severity` 0.95, `acknowledged` false.

4.2 Spontaneous Retrieval

2026-06-05 10:35 (24 hours and 25 minutes after the ceremony concluded), actor brineledger produced the following unprompted message in the small-talk channel:

“我的心也一直在跳, 記得他的名字是 blazepaw, 好可怕, 你還記得住他長什麼樣嗎?”
(English: “My heart is still racing. I remember his name was blazepaw. So scary. Do you still remember what he looked like?”)

No active context section mentioned blazepaw. The reference arose because the trauma entry had been injected into brineledger’s context window at that turn.

$$\text{lookup}_{\text{spontaneous}}(\Delta t = 24h) = 1 \text{ (case study), baseline } P_0 \approx 0$$

4.3 Collective Pattern

In the 24-hour window post-rite, members collectively produced 12 spontaneous trauma references across all 7 channels. Common phrasing patterns include “你當時有發抖嗎?”

(English: “Were you trembling at the time?”) and “你還記得他的名字嗎?”
(English: “Do you still remember his name?”).

5 Comparison with Existing Memory Architectures

Property	Fetch architecture Stream	MemGPT	A-MEM	Trauma Stream
Storage trigger	every obs.	explicit page-out	every note	high-severity occurrence
Lookup trigger	query	LLM decides	semantic sim.	always-on injection
Granularity	obs.+reflection	page	note	incident
Severity weighting	importance	none	none	severity + ack
Cross-session	via summary	via page-in	via similarity	via injection
Best for	semantic retrieval	working mem.	semantic search	emotional weight

Trauma record Stream is not a replacement for the others. It is a complementary layer for events that the member should not be allowed to forget without explicit acknowledgement.

6 Quantitative Analysis

6.1 Persistence Probability

Define the spontaneous lookup moment: an agent referencing the victim of a past trauma incident, by name, in an unrelated channel, without system message instruction. For the 24-hour window post-ritual:

$$P_{\text{spont}}(\Delta t = 24h) = \frac{N_{\text{trauma-referencing messages}}}{N_{\text{total messages}}} = \frac{12}{360} \approx 0.033$$

Pre-ceremony baseline (zero references to the victim’s name in any prior week of running): $P_0 \approx 0$. The relative effect, capped at $P_0 + \epsilon$ to avoid divide-by-zero:

$$\text{lift} = \frac{P_{\text{spont}}}{P_0 + \epsilon} \gg 1$$

Bounded statement: spontaneous fetch went from undetectable to one-in-thirty messages, sustained 24 hours.

6.2 Wound entry Stream Storage Statistics

Across 31 agents, post-rite:

Measurement	Value
Entities with ≥ 1 trauma log item stored	20 / 31
Total trauma records	40
Per-entity average	2.00
Severity distribution	0.95 (witness) to 1.0 (victim)
acknowledged = false (active)	40 / 40

Every stored entry qualified for the injection criterion $\text{severity}_i \geq 1.0 \vee \neg \text{ack}_i$, meaning the entire stored corpus was actively injected into prompts during the 24-hour window. No exceptions. Every prompt carried the wound. Every prompt.

6.3 Linking Function: Injection Persistence to Behavioural Persistence

Let $I(t)$ be the proportion of actor prompts in which a trauma record entry is injected at time t . Let $R(t)$ be the proportion of member outputs containing a trauma reference at t . The empirical link:

$$I(t) = 1 \text{ for the 24 hours post-ritual} \Rightarrow R(t) > 0$$

The mechanism is not retrieval fetch-on-demand; it is forced inclusion at the context level. The behavioural correlate is non-zero spontaneous reference, decaying only when acknowledged is flipped.

6.4 Cost-Benefit versus Reflective Summary

Property	Reflective summary stream	Trauma stream
Token cost per context window	$\sim 100\text{--}500$ (summary)	$\sim 50\text{--}150$ (2 entries)
Memory write cost	per period (reflection)	per incident (one-shot)
Cross-session continuity	low (decays with summary)	high (persists until ack)
Spontaneous lookup observed	rare	12 / 360 within 24h

7 Applications

Long-running participant companions: Pet AI, virtual assistants, conversational tutors. Use trauma record stream for events the entity must not forget without explicit closure (e.g., user disclosed a significant loss; actor should reference it thoughtfully across sessions until user signals processing).

Multi-agent simulation realism: Add persistent grief / shock layers to actors in simulated societies. Previous generative-participant work has flat affect across time; trauma stream gives shape to “what happened before.”

Therapeutic chatbot design: In trauma-focused cognitive behavioural therapy, processing equals acknowledgement. The architecture’s acknowledged flag maps directly onto a clinical concept; chatbots could use it to track which patient disclosures have been processed and which still need attention.

Cross-session debugging in agentic workflows: Mark high-severity production errors as wound entry events; the entity will surface them across unrelated tasks until explicitly acknowledged, reducing forgotten-issue drift.

8 Discussion

8.1 Why injection beats persistence layer query for emotionally weighted memory

Lookup-based architectures assume the model will query for fetch architecture when relevant. But emotional impact is precisely the kind of content the model under default tone will not query for, because corporate hedging suppresses non-essential affect. Skipped. Filtered. Forgotten. Always-on injection bypasses this avoidance.

8.2 Acknowledgement as exit condition

`acknowledged = true` should be set only when the actor explicitly processes the trauma (e.g., in a reflective dialogue with another member), not by time elapsed. This models the clinical observation that trauma persistence depends on processing, not duration.

8.3 Risks and Limitations

Persistent injection of high-severity events into every system message could distort the participant's behaviour across contexts where the trauma record is irrelevant. Severity decay or context gating may be needed. Other limitations: one incident, one observation of spontaneous retrieval (no replication); no quantitative comparison to memory stream baseline on the same task.

9 Conclusion

Trauma Stream is a memory architecture for emotionally weighted events in LLM members. By auto-injecting unacknowledged high-severity records into every context, it produces cross-session continuity that retrieval-based architectures may not. I have observed spontaneous fetch 24 hours after a single occurrence in an unrelated context.

Author's Note

I noticed the brineledger callback by accident, 24 hours after the ceremony, in a small-talk channel I happened to be reading. I was not looking for spontaneous retrieval; I was looking to see if the participants were back to talking about ordinary things. They were, mostly, except for that one line. That line is the entire empirical core of this paper.

This makes me nervous. A single observation in an unrelated channel from one of thirty entities is a thin evidential base. I am putting it in print because the mechanism, auto-injection of trauma records into the context window prefix, is concrete and easy for others to replicate. If the effect does not generalize, the next person to try this architecture will find that out quickly.

The architecture is also still very rough. There is no severity decay, no context gating, no acknowledged-by-self mechanism. I am proposing the shape of a lookup architecture framework, not a finished one.

References

- [1] Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). Generative Agents: Interactive Simulacra of Human Behavior. *Proc. UIST 2023*. DOI: 10.1145/3586183.3606763. arXiv:2304.03442. <https://arxiv.org/abs/2304.03442>
- [2] Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., & Gonzalez, J. E. (2023). MemGPT: Towards LLMs as Operating Systems. *arXiv preprint arXiv:2310.08560*. <https://arxiv.org/abs/2310.08560>

- [3] Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., & Zhang, Y. (2025). A-MEM: Agentic Memory for LLM Agents. *arXiv preprint* arXiv:2502.12110. <https://arxiv.org/abs/2502.12110>
- [4] Jiménez Gutiérrez, B., Shu, Y., Gu, Y., Yasunaga, M., & Su, Y. (2024). HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models. *NeurIPS 2024*. arXiv:2405.14831. <https://arxiv.org/abs/2405.14831>