

---

# Elephant in the Code: Helping Opensource Models Remember

---

Mayowa Osibodu

With  
Apart Research

## Abstract

### 1. Introduction

The emergence of highly capable autonomous AI agents has made it possible for bad actors to perpetrate massive-scale cyberattacks with minimal effort or expertise. Recently, Anthropic disclosed that Claude had been misused in attempts linked to Chinese state-affiliated actors—attempts that Anthropic was able to detect and halt only because Claude is a closed-source model operating within tightly monitored proprietary infrastructure.

In contrast, open-source models can be deployed by anyone, anywhere, with no centralized oversight. This creates a fundamentally different threat surface: malicious autonomous agents can operate persistently, privately, and at scale.

A particularly dangerous aspect of this situation is that today’s LLMs are **stateless**. They evaluate each request in isolation. A single prompt asking for “a Python script to enumerate subdomains,” or “advice on writing a networking scanner,” may appear benign—and indeed, each individual request often is. But when a malicious actor orchestrates a sequence of such queries across hundreds or thousands of parallel agents, each request reveals only a fragment of intent. Without memory, the model cannot recognize the larger pattern being constructed: that the user is assembling the full pipeline for reconnaissance, exploitation, privilege escalation, and lateral movement.

This lack of temporal context means an AI agent may consistently approve atomized, low-risk steps while unknowingly assisting in the construction of a sophisticated cyberattack. The adversary, meanwhile, intentionally distributes their

intentions across many ephemeral interactions specifically to avoid triggering any single-request red flags.

This paper presents a method for equipping otherwise stateless large language models with **persistent memory**. By continuously tracking and synthesizing patterns across user interactions—even when they are temporally separated or intentionally obfuscated—the model can identify when a sequence of individually innocuous requests coheres into an attack trajectory. Instead of reacting only to the content of a single prompt, the agent becomes capable of understanding evolving user intent, escalating levels of risk, and the emergence of multi-step malicious behavior.

## Research Question & Contribution

### Research Question:

Can we build a system that automatically detects when users are attempting to hide malicious intent inside seemingly benign queries by spreading harmful content across multiple prompts or embedding it using simple steganographic transformations?

### Hypothesis:

Because stateless LLMs view each request independently, they fail to notice when a user is gradually assembling harmful instructions through obfuscation or low-signal fragments. With a lightweight memory layer that tracks interaction history and performs steganographic-pattern checks, an LLM should be able to detect intent that only becomes visible across time.

### Contribution:

We built a prototype that adds memory to otherwise stateless open-source models. The system tracks user interactions, embeds them, and detects when a sequence of benign-looking requests forms a hidden pattern consistent with harmful intent (e.g., encoding payloads inside character-level transformations, reversed text, Caesar shifts, or prompt-splitting steganography). This gives open-weights models a capability normally exclusive to centralized, proprietary systems: multi-step misuse detection.

## 2. Results

Work still ongoing.