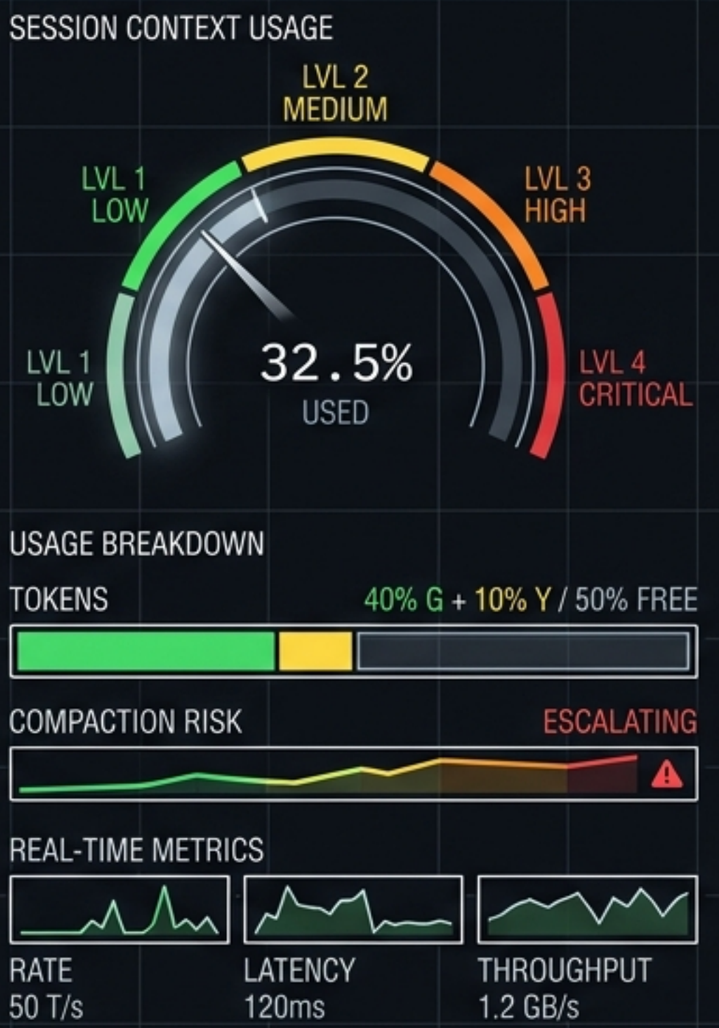
Instrumenting the Claude Code context window

`bulletproof-context-guard`

A Claude Code context-window monitor plugin.

Watches your session context, escalates warnings before auto-compaction, and exposes real-time usage breakdowns.



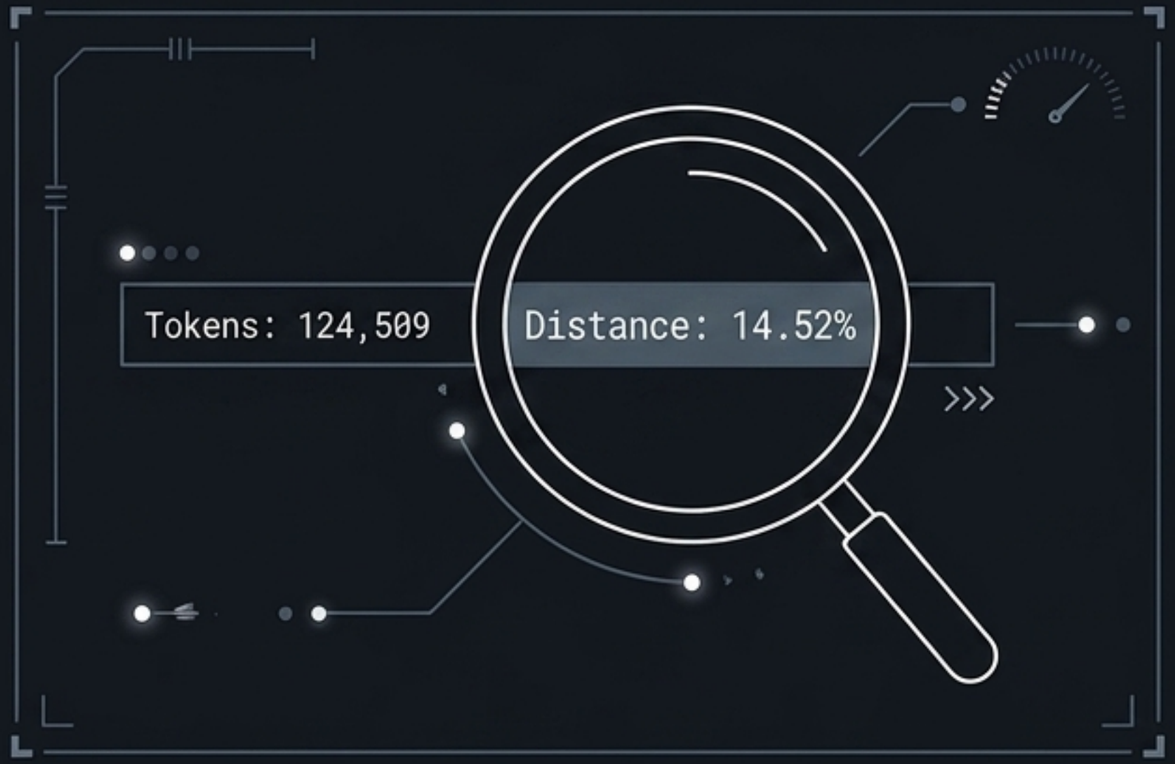
TWO COOPERATING SYSTEMS MONITOR AND REPORT CONTEXT STATE

The Monitor Hook: A status-line sensor reads live context telemetry on every render. It computes the distance to auto-compaction and triggers lifecycle hooks to announce tier changes or block risky operations.

The MCP Server: A local stdio server that exposes session telemetry via the `context_budget` tool, allowing the user or orchestrating agents to query detailed current usage.

The sensory engine tracks precision and burn-rate velocity

PRECISION MONITORING



VELOCITY ANALYSIS

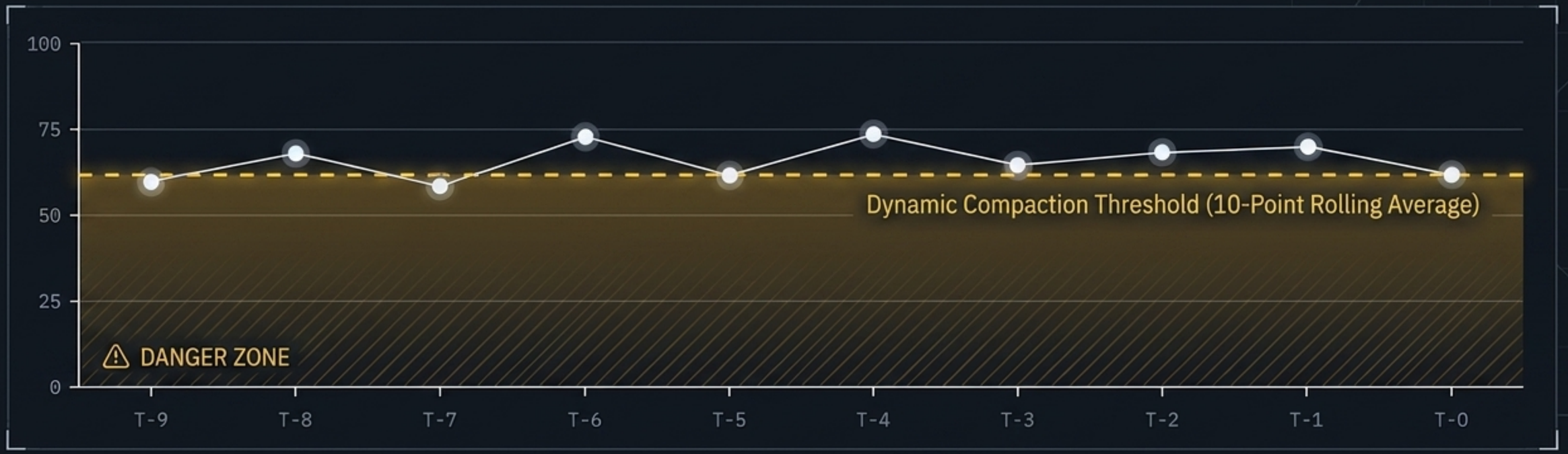
Token-Based Precision

When the model exposes token counts, the sensor computes distance-to-compaction in basis points (hundredths of a percent). This guarantees sub-percent accuracy for tier calculations.

3-Sample Velocity Tracking

A moving average window monitors the remaining-percentage drop per turn. If consumption rapidly accelerates (default: 5% drop per turn), the sensor preemptively bumps the warning tier up by one. The window smooths spikes, preventing false escalations.

COMPACTION THRESHOLDS ARE LEARNED FROM ACTUAL SESSION BEHAVIOR



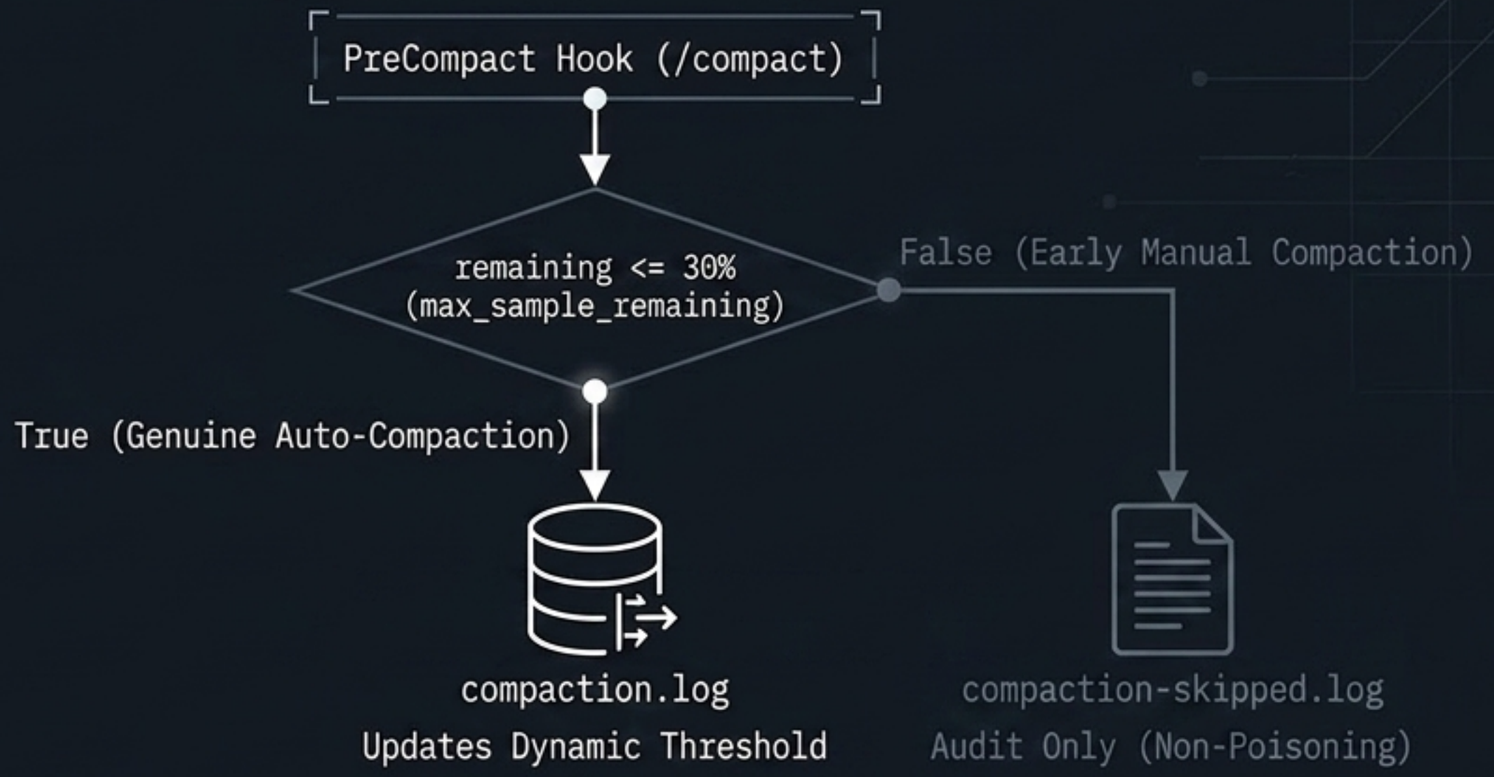
DYNAMIC CALIBRATION

The auto-compaction point is not a fixed assumption. The sensor maintains a rolling average of the last 10 genuine auto-compactions logged in state/compaction.log.

ADAPTIVE SAFETY

By learning exactly when Claude Code triggers auto-compaction, the monitor calibrates its warning boundaries to match reality, ensuring warnings fire accurately regardless of underlying system changes.

Guardrails prevent manual compactions from poisoning the threshold



The Clamp Mechanism: A manual /compact executed at 94% remaining context would artificially drag the dynamic threshold up, causing spurious escalations. The max_sample_remaining variable rejects these implausible samples, preserving the mathematical integrity of the adaptive threshold.

Four escalation tiers map distance-to-compaction to system guidance

Targeted System Guidance

As the dynamically calibrated compaction boundary approaches, the monitor injects targeted, one-shot system messages. These messages deliberately instruct the assistant not to mention context limits to the user unless explicitly asked, quietly steering system behavior to preserve context.

Early tiers nudge the assistant toward optimization and state saving

L1 (≤30%): Optimization Nudges

- Fires once to suggest delegating exploration to subagents.
- Instructs the assistant to use offset/limit on file reads.
- Caps Grep results with head_limit and prefers summarizing over quoting.

L2 (≤15%): Mid-Session State Saving

- Instructs the assistant to save session state using memory_scratch (key: session-state).
- Directs the assistant to stop chaining steps and proactively summarize outputs.

Critical tiers restrict operations and hard-block subagent dispatch

L3 (≤7%): Final Preparations

- Instructs the assistant to save the final state.
- Complete only the immediate operation.
- Suggests preparing for a new session.

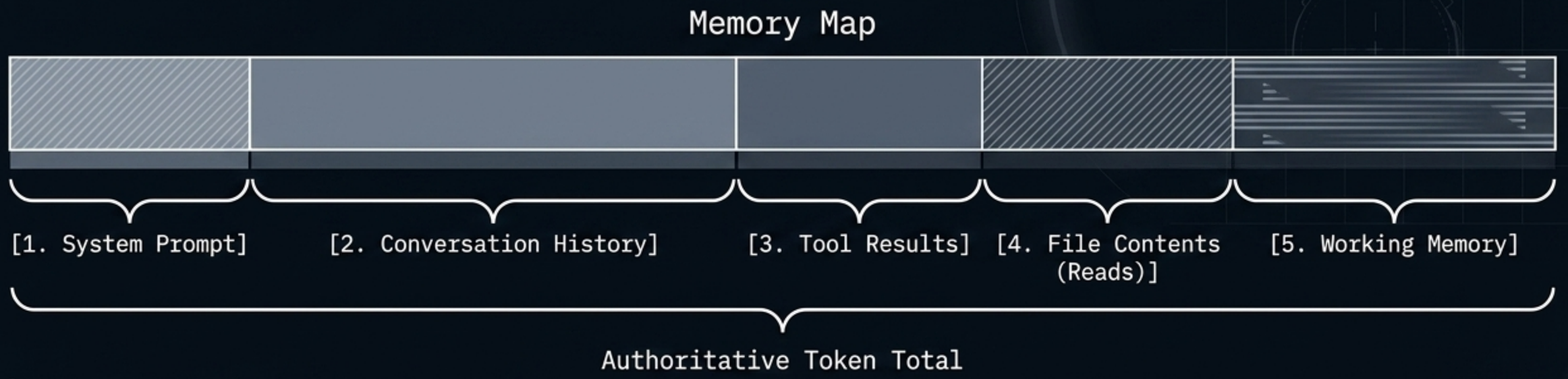
L4 (≤3%): Hard-Block Enforcement

- Auto-compaction is imminent. This warning is persistent on every tool use.
- Mechanically blocks the Task tool. Subagent dispatch is halted to prevent returning agents from dumping outputs and tipping the thread over the limit.
- State-saving tools (Write, Edit, Bash) remain allowed for a clean exit.

Compaction recovery branches based on verified state saves

State Verification: L2-L4 warnings explicitly instruct the assistant to write a marker via `echo saved > state/sessions/<id>/state_saved` to guarantee clean recovery.

The context budget tool exposes a five-compartment session breakdown



The context_budget MCP Tool

Returns current context usage, the active escalation tier, cost, and thresholds for orchestration or user querying.

Transcript Analysis Estimation

Because Claude Code only exposes aggregate token counts, the MCP server parses the session transcript JSONL to estimate the compartment split. These figures serve as critical navigational estimates, while the overall token sum remains authoritative.