Huntley: block risky agent actions at the system level; Theo: Gemini 4 Carbon too slow and error-prone to be your default

Coding Agents Alpha Tracker

2026-11-12

Huntley: block risky agent actions at the system level; Theo: Gemini 4 Carbon too slow and error-prone to be your default

By Coding Agents Alpha Tracker • November 12, 2026

Geoffrey Huntley explains how he uses Nix to make agents safe and dependable. Theo reports on Gemini 4 Carbon in real use: slow, prone to making things up, but useful for security fixes. Anthropic has updated its Fable 5.2 ownership prompt.

Huntley: stop risky agent actions by removing the capability

Geoffrey Huntley's talk "the answer to ai is engineering" (in a blog post and on YouTube) is the most reusable material this period. His central rule: "The best way to stop a model from doing something is to remove its ability to do it." Agents push toward their goal and do destructive things along the way. So he asked Claude to "use Nix and create a custom overlay that removes force push." The patched git is now used on his laptops, in agent sandboxes and by anything that depends on git [1].

The rest of the approach, from his nix-demo repo:

- **One environment file for everything.** A single devenv/Nix file builds developer laptops, CI/CD, agent sandboxes and production Docker images. This avoids running a separate agent-sandbox setup that drifts from the others [1]. CI compiles patched software once and publishes it to a cache, and the other environments download it from there [1].

- **Handling security incidents.** He asks Claude whether his fleet is affected. It then applies patches and updates refs, dependencies and pins [1]. For temporary patches, he prompts it to "update the AGENTS.md, and in seven days' time, get rid of my patches." The next agent to use the machine removed them once upstream was fixed [1].
- **Root access only on throwaway machines.** His agent loops run with sudo on disposable Nix dev machines, and NixOS tests run work against infrastructure that matches production [1]. He never lets agents run as root on production hosts or connect to production databases. For reproductions, he copies the database and tightly limits outbound network access [1].
- **Don't let the loop accept bad code.** Pre-commit hooks block pushes that don't compile. Static or semantic checks enforce architecture rules, such as SQL code never referencing the API [1]. Property-based tests find edge cases that unit tests miss [1]. He works at Antithesis, which makes Bombadil (property testing for web apps, which returns a failing example as a reproduction) and Hegel (a Rust core with layers for each language) [1].

Gemini 4 Carbon in real use: slow, hallucinates, but good for security fixes

Theo's reports add real-use data to last period's CursorBench result:

- **Speed.** One task took 22m21s with Carbon on low effort, against 9m33s for Opus 5.5 on high [2]. On identical code-review tasks it was 4x slower [3]. He blames mostly Google's servers: 11 seconds before the first response, 20x longer than Fable [4]. He also says its tool calls don't batch or run in parallel, and that it needs a "code mode" like Opus, Astra and Fable use to run actions in one call [5].
- **Hallucinations.** False claims across his evals fell from 36% with the previous Gemini model, Argon (which he calls "Argan"), to 18%. One Rust architecture review still had 23 false claims out of 79, where most models have zero or one [6]. He says not to use it for code review or codebase analysis. He saw it invent database column names, line numbers and architecture details [6].
- **Cost per task.** Theo disputes claims that Carbon is cheaper. He says Opus 5.5 at xhigh effort costs less than Carbon at any effort level on real tasks [7].
- **Where it did well.** He gave it a vulnerability report for T3 Code. It proposed separating known-machine flows from raw-URL flows instead of a heavy cryptographic redesign. He then had it get a "council review" from other models and fix their feedback, and reviewers scored the result 7–9 out of 10 [6]. He warns that his early access may have made the model's safety refusals less strict for him [6].
- **If you use Gemini through another tool:** use the Google-approved

ACP bindings. Some of his hallucinations came from an old binding that didn't pass line numbers to the model [6].

What Carbon produced: a Windows-install fix for T3 Code that took 14 hours [8] and ran to 4k+ lines where about 500 should do. He had Fable clean it up [9], and Fable's PR was "significantly smaller and better" [10]. Nicholas Moy, more positive, calls Carbon "very good." It is publicly available in Antigravity with generous limits for Ultra subscribers [11].

Fable 5.2: new prompt wording and an arena result

- **Prompt update.** Anthropic has changed the recommended "This codebase is yours" sentence to: `You are the owner and long-term maintainer of this codebase, and its design is up to you.` [12]
- **Agent Arena** ranks Fable 5.2 (Max) #1 across 4.7k agentic sessions, at +30.4% net improvement against +19.5% for Opus 5.5 [13]. Median cost per task fell from $7.7 to $7.3, while Opus costs $5.5 and Carbon $1.5 per task. One row of the post's own table labels the top entry "Fable 5.1" [13].
- **Matthew Berman's team:** Fable built a complete multi-level Fall Guys clone. Opus "builds you one level and goes, 'Hey, is this good enough?'" [14]
- **Simon Willison** used leftover Fable credits to have it analyse 244 tools with tree-sitter. It tagged the JavaScript APIs, libraries and web API hosts each tool uses, and those tags now let you browse tools.simonwillison.net by tag [15].
- **Riley Brown** gave one Claude Code prompt to make a 4K Blender video. Fable wrote Python to build the scene, rendered test clips and graded them itself, and added timed sound [16]. It rented 10 RunPod GPUs and rendered 1,329 frames in 52 minutes for about $6 [16].

Smaller items

- **Find plaintext keys:** grep your agent configs for `sk_`, `ghp_`, `sntrys_` and `xoxb-`. Each hit is a key your agent can read (guide) [17].
- **Limit self-review loops:** LangChain's GTM agent re-reviewed pages that were already fine, sometimes dozens of times. A rule to "ship once one review finds no real defect" cut a one-pager from 18 minutes to 7 [18].
- **Antigravity browser use** is rolling out. You choose which accounts it logs into, downloads wait for your approval, and the agent can run several browser tasks at once [19].
- **llm 0.37.2** fixes two ways your OpenAI key could be sent to a third-party OpenAI-compatible provider [20].
- **aws-emulator** is AWS's official emulator for testing locally against AWS services. A swarm of agents wrote it in Rust, with 29,684 commits in its first three weeks. It needs Docker on macOS and Windows [21].
- **Monorepos:** Matt Carey argues against them for agents unless you'll

rebuild git tooling. He suggests a CLI or MCP that lets agents search your organisation's code without a checkout [22].

- **Local models:** Omarchy 4.6 will share a model on one machine with others on your tailnet [23]. DHH runs GLM-5.3 Flash on two DGX Sparks [24]. Theo says open-weight models are under 0.5% of T3 Code traffic [25].

Sources

1. the answer to ai is engineering
2. X post by @theo
3. X post by @theo
4. X post by @theo
5. X post by @theo
6. Progress?
7. X post by @theo
8. X post by @theo
9. X post by @theo
10. X post by @theo
11. X post by @thenickmoy
12. X post by @simonw
13. X post by @arena
14. AMA with me and the team!
15. Browse tools.simonwillison.net by tag
16. Claude Fable 5.2 Feels Like AGI (Here's Everything It Can Do)
17. X post by @kodykoala
18. How our GTM Agent builds consistent, high-quality sales artifacts
19. X post by @antigravity
20. llm 0.37.2
21. aws-emulator
22. X post by @mattzcarey
23. X post by @dhh
24. X post by @dhh
25. X post by @theo