Anthropic Just Cut Off OpenClaw Users — Why Local Models Matter More Than Ever

April 4, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Anthropic announced that starting April 4, 2026, Claude Pro and Max subscriptions no longer cover third-party agent tools like OpenClaw. Users must now pay per-token via API credits or extra usage billing. Claude Code (Anthropic's own harness) remains covered. If you're running OpenClaw with Claude, your costs just went up dramatically. If you're running OpenClaw with local models like Qwen 3.5 or Gemma 4, nothing changed — and that's the point.

Related: Best Local Models for OpenClaw · OpenClaw Setup Guide · GPU Buying Guide · VRAM Requirements · Local AI vs Cloud API Cost

Contents

Anthropic just pulled the rug on thousands of OpenClaw users. Starting April 4, 2026, Claude Pro and Max subscriptions no longer cover usage through OpenClaw or any other third-party agent harness. If you were running OpenClaw with Claude on a flat-rate subscription, you now need to pay per token through API credits or Anthropic's "extra usage" billing.

Boris Cherny, Anthropic's Head of Claude Code, put it plainly: "Our subscriptions weren't built for the usage patterns of these third-party tools. Capacity is a resource we manage thoughtfully and we are prioritizing our customers using our products and API."

If you're running OpenClaw with local models, nothing about your setup changed today. Your hardware still works. Your tokens are still free. No one emailed you about a policy change. That contrast is the entire argument for local inference, and it just got a lot harder to ignore.

What happened

On April 3, Anthropic notified Claude subscribers that starting April 4 at 12pm PT, third-party agentic tools like OpenClaw would no longer be covered by subscription plans. The change affects Claude Pro ($20/month) and Max ($100/month) subscribers who were routing OpenClaw sessions through their subscription credentials.

Users can still connect OpenClaw to Claude, but must now pay through one of two mechanisms:

- **Extra usage billing** — pay-as-you-go charges on top of your subscription
- **API key** — standard per-token API pricing (currently $3/$15 per million tokens for Sonnet, $15/$75 for Opus)

As a concession, Anthropic is offering affected subscribers a one-time credit equal to their monthly subscription price and discounts on pre-purchased usage bundles. Full refunds are available via email.

OpenClaw creator Peter Steinberger and investor Dave Morin attempted to negotiate a delay. They secured one week. The enforcement date moved to April 5 at 12pm PT.

Why Anthropic did this

This wasn't spite. It was math.

Flat-rate subscription pricing depends on average usage being well below the limit. Most Claude Pro subscribers don't hit their token ceiling every month. The margin exists because human-driven usage is bursty — you use Claude for a few hours, then you go eat dinner.

Agent workloads break that model. OpenClaw sessions can run continuously — tool calling loops, code execution, verification cycles, retry chains. One aggressive OpenClaw session can exhaust a weekly token allocation in hours. Multiply that across thousands of power users running agents 24/7 and the economics collapse.

Anthropic's own data backs this up. Cherny noted that third-party tools "put an outsized strain on our systems" compared to human-driven usage. The pattern was already visible weeks ago

when users on r/OpenAI reported that Codex weekly limits “ran out easily, only used for OpenClaw.”

Anthropic also pointed to a technical factor: their first-party tools like Claude Code are optimized for prompt cache hit rates, reducing actual compute cost per session. Third-party harnesses don’t necessarily implement these optimizations, so the same amount of user-facing work costs Anthropic more compute.

This is a rational business decision. But the downstream effects for users are real.

The Claude Code asymmetry

Here’s the part that stings: Claude Code — Anthropic’s own coding agent — remains fully covered by subscriptions. Only third-party tools are cut off.

Anthropic frames this as a capacity management issue. They control Claude Code’s architecture, its caching behavior, and its token efficiency. They can’t control how OpenClaw or other harnesses use the API.

That framing is technically accurate. It’s also competitively convenient. Claude Code is a direct competitor to OpenClaw for agentic coding workflows. After this change, Claude Code works on your subscription. OpenClaw with Claude doesn’t.

Steinberger called it ecosystem-damaging, alleging that Anthropic absorbed popular open-source features before locking out competitors. Whether you view this as predatory or pragmatic depends on your perspective, but the practical outcome is the same: if you want a flat-rate Claude agent, you use Anthropic’s agent.

Unless you don’t use Claude at all.

What this costs affected users

The cost increase is substantial. Some estimates from HN discussion threads put it at up to 50x for heavy agent users.

Here’s a rough comparison for a moderate OpenClaw user running Claude Sonnet:

Scenario	Before (subscription)	After (API pricing)
Light use (500K tokens/day)	$20/month (Pro)	~$27/month
Moderate use (2M tokens/day)	$20/month (Pro)	~$108/month
Heavy use (5M tokens/day)	$100/month (Max)	~$270/month
Continuous agent (10M+ tokens/day)	$100/month (Max)	~$540+/month

The “light use” scenario is roughly break-even. Everything above it gets expensive fast. And heavy OpenClaw users — the ones running continuous coding sessions, automated testing loops, or multi-agent workflows — are looking at monthly bills that dwarf their old subscription.

For context, here’s what those same workloads cost on local hardware:

Scenario	Local cost (after hardware)
Any usage level	$0/month in inference costs
Hardware amortized over 2 years	~$30-40/month for a used RTX 3090 setup

A $700-800 used RTX 3090 pays for itself in 2-3 months for a moderate OpenClaw user who just lost their flat-rate Claude access.

How to migrate to local models

If you’re affected, here’s the practical path to running OpenClaw on local hardware with zero recurring inference costs.

Step 1: Pick your model

Two strong options right now for local OpenClaw:

- **Qwen 3.5 27B** — best overall coding and agentic performance at this size. Fits in 24GB VRAM at Q4. Our Qwen 3.5 local guide covers setup.
- **Gemma 4 26B-A4B** — Google’s mixture-of-experts model, only ~4B parameters active per token. Runs fast on modest hardware. See our Gemma 4 guide.

For a full model comparison with benchmarks, see Best Local Models for OpenClaw.

Step 2: Check your hardware

You need a GPU with enough VRAM to run your chosen model. The minimum practical setup:

Model	Min VRAM	Recommended GPU	Approx. Cost
Qwen 3.5 27B Q4	20GB	RTX 3090 (24GB)	$700-800 used
Gemma 4 26B-A4B	12GB	RTX 3060 12GB	$170-200 used
Qwen 3.5 14B Q4	12GB	RTX 3060 12GB	$170-200 used

If you don't have a GPU yet, our GPU buying guide and VRAM requirements guide cover every option at every budget.

Step 3: Set up the stack

1. Install Ollama or your preferred inference server
2. Pull your model: `ollama pull qwen3.5:27b`
3. Point OpenClaw to your local endpoint
4. Our OpenClaw setup guide walks through the full configuration

The tradeoff

Local models aren't Claude Opus. Qwen 3.5 27B is competitive with Claude Sonnet on many coding tasks but falls behind on complex multi-step reasoning. For most OpenClaw workflows — code generation, file manipulation, tool calling — the gap is smaller than you'd expect. For frontier-level reasoning, you'll still want an API model.

The Claude Code architecture article we published recently shows that the underlying harness patterns — the 12 tool primitives, the approval flow, the safety checks — work regardless of which model powers them. The architecture is model-agnostic. Your local model slots right in.

The bigger picture

This won't be the last time a cloud provider restricts agent usage. The economics are structural: agent workloads consume tokens at rates that flat-rate pricing can't absorb. OpenAI will face the same pressure. Google will too. Every provider selling unlimited-ish access will eventually hit the wall where agent users cost more to serve than they pay.

An OpenAI employee hinted in the aftermath that they might support OpenClaw — but Steinberger himself acknowledged uncertainty about whether any provider can sustain flat-rate agent pricing long-term. The fundamental conflict between subscription models and continuous agent consumption doesn't go away by switching providers.

The only pricing model that never changes on you is hardware you own.

A used RTX 3090 doesn't send you an email at 3pm on a Thursday saying "starting tomorrow, you can't use this for agents." It doesn't deprecate your workflow because a PM decided the margin was too thin. It doesn't care whether you run inference for one hour or twenty-four.

This is what InsiderLLM has been about since day one. Your data. Your hardware. Your AI. Today, a few thousand OpenClaw users learned why that matters. Tomorrow, it'll be another provider, another restriction, another group of users scrambling to migrate.

Or you can set up local inference now and skip the scramble entirely. The cost comparison makes the math clear. The setup guide takes thirty minutes. The models are good enough for real work. And the price is right: free, forever, no policy changes.

Related guides

- Best Local Models for OpenClaw — model picks and benchmarks
- OpenClaw Setup Guide — full configuration walkthrough
- Qwen 3.5 Local Guide — best overall local coding model
- Gemma 4 Local Guide — fast MoE model for modest hardware
- GPU Buying Guide — which card to buy at every budget
- VRAM Requirements — what fits in your GPU
- Local AI vs Cloud API Cost — the full cost comparison
- Cost to Run LLMs Locally — electricity, hardware amortization, real numbers
- Claude Code Architecture Lessons — why the harness patterns work with any model

Get notified when we publish new guides.

Subscribe — free, no spam
