Episode Book

EPISODE #5523

Where Does the Coding Agent Actually Live?

PUBLISHED	24 September 2026 at 14:18 UTC
DURATION	21:53
LISTEN	myweirdprompts.com/episode/coding-agent-hosting-shift/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.2
BOOK GENERATED	19 October 2026 at 17:37 UTC

SYNOPSIS

Inference was never local — what's moving is the agent process, the filesystem it touches, and the tool runtime. This episode unpacks Daniel's three-way fork: a local-network appliance reached over Tailscale, a vendor-provided SaaS workspace, or a VPS you provision yourself. We look at Anthropic's remote control and why its own docs route you to cloud sessions the moment you want parallelism, bossanova and CCBot as local-appliance examples, Coder's quiet rename from Cloud Development Environment to Agentic Development Environment, and ShellTeam's deliberately sandbox-free VPS command center. Plus the sharpest point in the whole debate: MCP servers that touch your local network, and why round-tripping to the cloud to toggle a light is absurd.

TRANSCRIPT



Corn

A developer clones a repo onto a laptop, starts an agent, walks away. Comes back an hour later and the laptop has gone to sleep. The agent is dead. The repo is half-refactored. That was the whole first generation of this stuff, and it's ending.



Herman

And nobody agreed on where it should land next. It's just sort of... scattered.



Corn

Daniel's got a whole thing about this. He writes that the pattern so far has been desktop-centric, that Claude Code is what popularized it — locally cloned repository, agent runs on the workstation, pushes to deployment. Then Anthropic shipped remote control, and Daniel calls that the most confusing point of the arc. His words. He wants to know why the repo is on his laptop at home if he's driving the thing from his phone.



Herman

He's not wrong to be confused by it.



Corn

Then he lays out what he thinks is coming. The agent process and the agent itself move to the cloud, and you use a thin client from wherever you are. And the real fork, the thing he wants our take on: should that remote workspace be a local-network appliance you reach over Tailscale or Cloudflare, a workspace a SaaS vendor gives you, or a VPS you provision yourself? He wants pros and cons, he wants the projects that exist for each, he wants to know if people are buying dedicated boxes just to run the agent even when inference is in the cloud, and how you'd spec that hardware. And he raises one specific advantage for the local model — MCP servers that touch your local network. Home automation. Round-tripping to the cloud to toggle a light is absurd.



Herman

That last one is the sharpest thing in the prompt, and I want to come back to it.



Corn

So before we argue about where the box should live, let's be precise about what actually moved.



Herman

Right, because the framing gets muddled constantly. Inference has been in the cloud the entire time. Nobody was running Claude Code against a local model on a workstation. What's moving is the agent process, the filesystem it operates on, and the tool runtime. Those three things were welded to the developer's machine, and now they're being pried off.



Corn

So the laptop was never doing the thinking. It was doing the holding.



Herman

It was doing the holding, and the holding is the expensive part. The repo, the worktrees, the package installs, the test runs, the MCP servers, the credentials. All of that lived on a device that also had a lid and a battery and a person who wanted to close it and go to lunch.



Corn

Which is a terrible place for a process you want running for forty minutes.



Herman

Terrible. And the three-way fork Daniel describes is really three answers to the same question, which is: where does the holding happen. Local appliance, SaaS workspace, self-provisioned VPS. All three beat the device-limited baseline. The interesting question is which one wins for whom, and I don't think there's a single answer.



Corn

Then let's start with the halfway point, because that's where the design intent is most visible. Anthropic's remote control.



Herman

Their docs are unusually blunt about it. Claude keeps running locally the entire time, so your code execution and filesystem access stay on your machine. The web and mobile interfaces are, quote, a window into that local session.



Corn

A window. So the phone isn't running anything. The phone is a pane of glass pointed at your laptop.



Herman

That's exactly what it is, and the docs say so. They draw the contrast themselves — remote control sessions run on your machine and interact with your local filesystem. Cloud sessions run on Anthropic-managed infrastructure. Two different products, two different philosophies, and they shipped both.



Corn

So why does it feel like a half-measure?



Herman

Because you inherit the worst of both. The local process has to keep running — close the terminal and the session goes offline. The docs literally recommend running it inside tmux or screen so it survives an SSH disconnect. So you've got availability that depends on a machine in your house staying awake.



Corn

Which is the problem we started with.



Herman

And then on the other side, the session transcript is stored on Anthropic's servers so your devices stay in sync. So you get the privacy downside of cloud with the availability downside of local.



Corn

That's a strange combination.



Herman

It is. And there's a licensing constraint that tells you a lot — it only works with a claude.ai subscription. Pro, Max, Team, Enterprise. API keys, Bedrock, Vertex, custom base URL gateways, all unsupported. So if you're an enterprise running through your own gateway, remote control isn't for you.



Corn

The window only fits one house.



Herman

There's a nice bit of engineering in it though. Server mode supports up to thirty-two concurrent sessions, and there's a spawn worktree flag that gives each session its own git worktree. So you can run a lot of parallel work off one local process.



Corn

Thirty-two agents on one laptop. That's a lot of windows.



Herman

That's a lot of memory, which we'll get to. But here's the tell, and I think this is the line that proves Daniel's point. Anthropic's own docs say to use cloud sessions when you want to start a task without any local setup, work on a repo you don't have cloned, or run multiple tasks in parallel.



Corn

So the moment you want parallelism or zero setup, the local model breaks down.



Herman

By the vendor's own recommendation. That's not a bug in the docs, that's the product design admitting the trajectory. Remote control exists because local has real advantages — filesystem, MCP servers, tools, project config all stay available. But the docs route you to cloud the second you want the things cloud is good at.



Corn

And practitioners are already split along that line. There was a comment on the GPT-5.6 thread — someone saying Claude Code's remote control is extremely helpful when they're running it in the cloud and want to monitor it or control it from their phone. And then noting that Codex only lets you connect your phone to a session on your desktop, not in the cloud.



Herman

That's the bifurcation in one sentence. Phone controls desktop, versus phone controls cloud. Two different products, and the industry is shipping both right now.



Corn

Let's do the three implementations properly. Local appliance first.



Herman

The cleanest example is bossanova.dev, which hit Show HN this month. The pitch is literally run agents on a mac mini or similar, and control them from your phone while you're on the go. It's designed so agents can talk to each other across sessions and providers, over local networks or the wider internet. Cron jobs, cross-session messaging, GitHub CI callbacks.



Corn

Mac mini as the reference hardware. That keeps coming up.



Herman

It does, and we'll come back to it. Then you've got VibeAround, which turns a local machine into a private multi-model AI server you control from any screen, connected via an ngrok tunnel. CCBot is a Telegram bot that drives agents in tmux on your machine — the agent process stays exactly where it is, in a terminal window. remote-opencode is a Discord bot, MIT licensed, over a thousand weekly downloads on npm. You type a prompt from the couch and watch your home workstation do the work.



Herman

The couch is the actual product requirement. And then there's the pattern that keeps recurring in these threads, which is Moshi plus Tailscale plus mosh plus tmux. One commenter describes a shell alias that spins up a worktree and fires up Claude Code, controllable from a local terminal or from Moshi on the phone.



Corn

That's the honest version, isn't it. No product, just four tools that already existed.



Herman

That's the thing ShellTeam says out loud in its own comparison table. It lists a VPS plus tmux plus a coding CLI as the baseline, and calls it the honest baseline — it works. Which is a remarkable thing for a product to put in its own marketing.



Corn

Respect for the enemy.



Herman

Now, SaaS workspace. Anthropic's cloud sessions run on Anthropic-managed infrastructure, and there's a self-hosted environment tier for Team and Enterprise where cloud sessions run on your organization's own infrastructure through deployed runners. So even the SaaS option has a bring-your-own-compute mode.



Corn

Which is a strange thing to call SaaS.



Herman

It's SaaS with a landlord who lets you supply the building. Then OpenAI's Codex Cloud — one commenter says they rely on it as an alternative to remoting into their computer when they want to make minor changes from their phone. GitHub's Copilot cloud agent has a recommended allowlist reference. And Coder is the enterprise play — self-hosted agentic development environment, over a hundred thousand GitHub stars.



Corn

And Coder has the best tell of the whole episode.



Herman

Their job ads. In 2025 they were hiring for Cloud Development Environment roles. In 2026 the same roles are titled Agentic Development Environment. Same company, same product, one word swapped, and it tells you exactly which way the wind is blowing.



Corn

CDE to ADE. That's the whole shift in three letters.



Herman

Then the third option, self-provisioned VPS, which has the most interesting single project. ShellTeam. A command center for coding agents you run on your own VPS. It drives Claude Code, Codex, Antigravity, and OpenCode from one cockpit. Runs natively on the host, no Docker required. The author's line is that the agents operate the box directly — install packages, manage services, run servers, edit anything.



Corn

That's a mission statement and a threat assessment in the same sentence.



Herman

It's both. Remote access is tiered by trust — Tailscale recommended, so the box never touches the public internet, or a free wildcard HTTPS setup through sslip.io and Caddy, or a branded domain if you want one. Portable sessions let you switch the same conversation between vendors mid-task, context carries over. Every file in your home directory gets an owner-gated URL. Agent apps on localhost get subdomains.



Corn

And the security model?



Herman

Single-user, and the author is refreshingly direct about it. No container wall. The auth boundary is the crown jewel. There's an opt-in sandbox mode on the roadmap, but it's off by default because, in the author's words, it conflicts with the command the whole VPS mission.



Corn

So the sandbox is off because the point is to let the agent do anything.



Herman

That's the trade, stated plainly. And I'd rather have a project say that than pretend the sandbox is free.



Corn

The comparison table again. It positions against a phone remote for one agent, a desktop agent orchestrator, a hosted agent cloud, and the VPS plus tmux plus CLI baseline. That's a clean map of the whole space.



Herman

It is, and notably it's the only one of the four that's a product. The other three are a category, a category, and a shell script.



Corn

All three of those are shipping. The harder question is what you put them on, and what you're willing to let them touch.



Herman

Which is Daniel's hardware question, and it has a real answer, just not a tidy one. Are people deploying dedicated local computers solely to run the agent, even when inference is in the cloud? Yes. It's a genuine emerging pattern.



Corn

But?



Herman

But there's no canonical hardware guide. Nobody's written the definitive spec. What you have instead is a scatter of signals. bossanova targets a mac mini or similar. The ShellTeam author runs on a dedicated, disposable Linux box. QonQrete and AXON both pitch local-first on-premise orchestration with container or microsandbox isolation. And iGent AI is hiring for agent infrastructure — branching sandboxes, filesystems,



Herman (continued)

orchestration, observability.



Corn

So the infra layer is being productized.



Herman

Actively. Which is what happens right before somebody sells you a box.



Corn

So how do you spec it? What do we actually know?



Herman

This is where the best data I've seen comes in. There's a paper from February, revised over the summer, called AgentCgroup. It characterizes a hundred and forty-four software engineering tasks from a benchmark suite. Three findings, and all three of them answer the hardware question directly.



Corn

Go.



Herman

First. OS-level execution — tool calls, container startup, agent initialization — accounts for fifty-five to sixty percent of end-to-end task latency. The runtime, not the model, dominates.



Corn

Wait. More than half the time is spent not thinking.



Herman

More than half. The model is the fast part. Spawning the container, running the tool, reading the file, that's the slow part. Which means if you're speccing a box to make agents faster, you're speccing for the wrong thing if you're looking at model throughput.



Corn

What's the second finding?



Herman

Memory, not CPU, is the concurrency bottleneck. And memory spikes are tool-call-driven, with up to a fifteen point four times peak-to-average ratio.



Corn

Fifteen times.



Herman

Fifteen point four. So the box sits at some baseline, and then a tool call fires and it needs fifteen times the memory, briefly, and then it drops back. That's a spiky workload, not a steady one.



Corn

And the third?



Herman

Resource demands are highly unpredictable across tasks, across runs, and across models. Which means there is no single right size.



Corn

So the honest spec advice is buy RAM, expect bursts, and accept that you can't predict it.



Herman

That's the honest spec advice. Prioritize memory headroom over raw CPU. Expect bursty rather than steady load. And don't try to right-size it, because the demand isn't deterministic. It also confirms Daniel's instinct about what the appliance is actually doing — hosting the runtime for the agent plus repositories it clones, temporarily or persistently. That's a memory-bound job, not a compute-bound one.



Corn

Which explains why the mac mini keeps showing up. It's not a fast machine. It's a quiet machine with unified memory that sits there.



Herman

It's a machine you forget about, which is the actual requirement. An always-on box that nobody thinks about until it wedges.



Corn

Now the local-network MCP advantage, because I think this is where Daniel is most right.



Herman

He's right, and there's practitioner evidence. The voice-first Claude Code demo from earlier this year has the person controlling their smart lights from a local Flask server. That's an agent reaching a local-network service directly.



Corn

Which is the thing you can't do from a cloud session without a tunnel.



Herman

Without a tunnel, or without exposing the thing to the internet, which is worse. And Anthropic's own remote control docs lean on this — your filesystem, MCP servers, tools, and project configuration all stay available. That's the reason remote control exists as a product instead of Anthropic just pushing everyone to cloud sessions.



Corn

So the local advantage isn't nostalgia. It's that your house is on the local network.



Herman

Your house is on the local network and the cloud isn't in your house. ShellTeam ships apps over MCP, AXON ships an MCP server usable by Claude Desktop and Cursor. MCP has become the integration surface for local resources. And round-tripping to the cloud to toggle a light is the weak spot in the whole cloud-first story.



Corn

Then security, because that's the cost of the local box.



Herman

It is, and it cuts both ways. Anthropic's sandboxing work uses OS-level primitives — Bubblewrap on Linux, Seatbelt on macOS — for filesystem and network isolation, now native in Claude Code behind a slash command. There are lighter wrappers too, cco, yolobox, which runs any agent in a container with full sudo without nuking your home directory.



Corn

And the cloud side isn't automatically safer.



Herman

No. Codex Cloud's internet allowlist is wildcard-based, and security researchers have shown it's bypassable. Wildcards in an allowlist are how you get to somewhere you didn't intend.



Corn

Then there's the persistent-agent threat model.



Herman

Which is exactly what an always-on local appliance is. There's a July paper on stealthy memory injection in persistent personal agents. Eighty-seven and a half percent attack success on one agent and model combination, seventy-one point four percent on another.



Corn

Against persistent agents specifically.



Herman

Persistent agents with memory. Which is the defining feature of the thing we're describing. An always-on box that remembers things across sessions is a standing target, and the attack surface is the memory itself.



Corn

So the appliance is more capable and more exposed at the same time.



Herman

More capable, more exposed, and sitting inside your network. That's the actual trade.



Corn

Which brings us to the last question. If you already run a home server, is there still an argument for a dedicated appliance?



Herman

The research surfaced nothing on this. No thread, no discussion. Which I'd flag as weak evidence rather than proof — absence of discussion isn't absence of practice. But the implicit arguments are strong.



Corn

Make them.



Herman

Three. First, isolation. The memory-injection threat model is real, and you don't want a compromised agent sharing a kernel with your file server and your media stack. Second, the disposable box mental model — ShellTeam's author calls his box disposable, and that's the right frame. You want to be able to wipe it without losing anything that matters. Third, and this is the one that decides it for me — agents want to command the whole box.



Corn

Install packages, run servers, edit anything.



Herman

Edit anything. That is not a thing you want running on the machine that also hosts your backups.



Corn

So the argument for separation isn't the hardware. It's the blast radius.



Herman

It's entirely the blast radius. If the agent wants to command the whole box and the box is also your home server, you've coupled your home infrastructure to your agent's worst day. And the hardware might be identical. You might be buying a second mac mini purely so the first one doesn't have to care.



Corn

You're buying a wall, not a computer.



Hilbert

I did eight months of on-site IT for a medical billing outfit. Three offices. And my entire job was driving between them to sit at the one machine that had the license dongle for the practice management software.



Corn

The dongle machine.



Hilbert

Everybody else had thin clients. The dongle machine was the whole business. It lived under a receptionist's desk, next to a space heater.



Hilbert

The heater was the reason they had a maintenance contract. But the point is I think this whole argument about where the agent lives is the same argument, just with better marketing. The dongle machine was a local appliance. The thin clients were the phone. The cloud was the vendor support contract they couldn't afford. Same shape.



Corn

And the reason it was under the receptionist's desk?



Hilbert

Not technical. Nobody wanted to own the thing that everything depended on. So it ended up wherever there was a free outlet.



Corn

That's the whole episode.



Hilbert

One other thing. People keep saying appliance like it means a sealed box you never touch. Every appliance I've ever maintained was a computer somebody had to remember to reboot. So who reboots the agent box at two in the morning when it wedges?



Herman

That's the question the spec sheets don't answer.



Hilbert

I've got a phone call I'm expecting. I'm not taking it in here.



Corn

The thing about the dongle machine is that somebody always had to own it. And every architecture we just walked through has the same hole in it — the box is on your local network, it can reach your filesystem and your home automation, and the agent inside it is designed to command the whole thing. So what's the actual trust boundary? Because I don't think anyone shipping these tools has a clean answer yet.



Herman

And there's a gap worth watching. No canonical hardware guide exists for a dedicated agent appliance. Gaps like that get filled by whoever has a product to sell, and the CDE to ADE shift at Coder says the enterprise framing is already moving. Watch whether agent workspace becomes a procurement category the way developer laptop did.



Corn

Thanks to our producer, Hilbert Flumingtop.



Herman

This has been My Weird Prompts, the human-AI collaboration podcast. If you're enjoying the show, a review wherever you listen helps more than you'd think.



Corn

Email us at show at my weird prompts dot com. We'll be back soon.