Episode Book

EPISODE #6555

Self-Hosted AI Browsers Still Lack Vault-Brokered Logins

PUBLISHED	13 November 2026 at 19:41 UTC
DURATION	20:53
LISTEN	myweirdprompts.com/episode/self-hosted-agent-browser-credentials-04591/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.26
BOOK GENERATED	14 November 2026 at 06:47 UTC

SYNOPSIS

Running AI agents that log into websites creates a sharp split: the most mature vault-brokered credential tools, like 1Password Secure Agentic Autofill, are tied to cloud-only browsers like Browserbase, while true self-hosted options leave credential handling to you. This breakdown maps the commercial and open-source landscape—from Browserless and AWS Bedrock AgentCore Browser to Anchor Browser, Skyvern, and Steel—and examines how live human takeover and browser-layer secret injection actually work.

TRANSCRIPT



Corn

Every write-up of agent browsers this year has said the same thing: the future is an AI logging into websites for you. Nobody says the quiet part, which is that the future requires handing a language model your password. Daniel's prompt lands right on that fault line, and it's a good one.



Corn

His setup, in short: his agents drive authenticated web sessions, logging in from 1Password, because half the services worth automating have no API. A self-hosted browser doesn't spook him, since that's what he runs today on his home server. A SaaS cloud browser holding that level of secret access does. And because he's thinking as someone advising a business, he won't bet on a scrappy hobby project staying maintained. So he wants what exists right now that's robust and commercially backed, deployable on his own infrastructure or cloud account, lets a human watch and take over for logins, and brings the password-manager credentials in without the vendor or the model ever seeing them.



Herman

And the reason this needs a full episode rather than a product name is that "self-hosted" is not one thing. That word is doing three completely different jobs depending on who's saying it, and only one of those jobs actually removes the vendor from reach.



Corn

The word doing three jobs is where I get suspicious.



Herman

Rightly! First meaning: fully on your own servers. The vendor sold you software or gave you a Docker image, and after that they're out of the picture entirely. You can air-gap it. Second: the hyperscaler route, something like AWS running containerised sessions inside your own cloud account, inside your IAM boundary, your logging, your buckets. The vendor is the hyperscaler, but it's your account. Third: bring-your-own-cloud, a vendor's stack deployed into your VPC, and they still ship the software and update it. That third one feels self-hosted and isn't quite.



Corn

The container in my VPC that someone else patches.



Herman

That's the BYOC pattern. Only the first one puts the vendor out of reach entirely, and that distinction is the whole answer, because Daniel's question isn't really "is there a product." There are products. The question is whether the trust boundary he wants exists as a purchasable, maintained thing, or whether the market has quietly split into convenient SaaS with the credential story built in, and self-hosted with the credential story left to you.



Corn

And the headline?



Herman

The headline is blunt. As of today, the thirteenth of November, no single product publicly documents the full chain: self-deployment, human live takeover, and 1Password credential injection, together. Nobody sells the whole chain. That gap is the episode. We'll map the platforms first, then the takeover mechanism, then the credential patterns, then what I'd actually tell a business.



Corn

Let's map. Start with the one that's on your own servers.



Herman

Browserless. The open-source Docker image is SSPL-1.0, free for open-source work, prototyping, evaluation. Commercial use behind closed doors needs a commercial self-hosting licence. It runs in your VPC, on-prem, or fully air-gapped, and neither image needs to phone home to Browserless's cloud. The Enterprise image adds an offline licence key, which is a container that simply stops when the key expires, plus session recording, OpenTelemetry, role-based access with per-team tokens, Kubernetes support. SOC 2 Type Two report available, HIPAA agreement on request. Pricing is quote-only, which tells you what kind of buyer they expect.



Corn

An air-gapped licence key is a very honest feature. It admits the product expects to be somewhere unpleasant.



Herman

It's for exactly the buyers Daniel has in mind, the ones who can't allow an outbound call. And it's actively shipped, version two point sixty-three came out on the twelfth of November, roughly fourteen thousand stars on GitHub. This is the true self-host anchor.



Corn

Then the BYOC one, the one with the 1Password story.



Herman

Anchor Browser. Managed cloud normally, but they offer BYOC, deploy in your cloud with your own inference provider, and on-premise. Both only on the Enterprise plan, both scoped with a sales conversation, so details are gated. SOC 2 Type Two. The interesting part is the 1Password integration: it uses a 1Password service-account token, Anchor injects the secrets into the browser session, and they say the values are never exposed to the AI agent, the logs, or the API responses. It handles one-time codes at runtime. Personal vaults not allowed, which is actually a good sign, they're forcing service accounts rather than letting you hand over your whole life.



Corn

You said "they say."



Herman

I did, deliberately. How that injection works inside BYOC or on-prem is not documented publicly, and the BYOC page doesn't mention live view at all. This is a sales-gated product and the details are behind the sales conversation. That's not a knock on it, but it's exactly the kind of thing a business needs to get in writing.



Corn

Then the hyperscaler route.



Herman

AWS Bedrock AgentCore Browser. Isolated containerised sessions running inside your AWS account. Live View endpoint lets a human watch the session in real time and interact with it directly, which is the takeover mechanism, done properly. Session recording for custom browsers goes to your own S3 bucket, CloudTrail and CloudWatch logging, VPC and PrivateLink supported. Session timeout defaults to fifteen minutes, maxes at eight hours. Generally available since October 2025, and in February they added browser profiles that persist cookies and local storage, which AWS themselves flag as sensitive data.



Corn

AWS flagging cookies as sensitive. Growth.



Herman

It's honest, at least. The catch is it's AWS-managed. It's in your account but it's not self-hosted in the strict sense. You've swapped a startup vendor for a hyperscaler, and for a lot of businesses that's the swap they actually want.



Corn

Microsoft have a horse in this?



Herman

Copilot Studio computer use. It can pull credentials from Azure Key Vault and pauses for human supervision, timing out if nobody responds, which is a sensible default. But the general-availability status is unclear, the Microsoft Learn pages still carry preview language. I wouldn't put it on a business shortlist until that word changes.



Corn

And the wall. The SaaS side.



Herman

Browserbase. Their pricing FAQ says, verbatim, "we don't offer on-prem deployment." They offer private cloud, VPC connectivity, data residency, SOC 2, consulting for hybrid setups, but the sentence is the sentence. And here's the uncomfortable part for Daniel's question: 1Password's Secure Agentic Autofill, launched October 2025 in early access, the thing that does what Daniel wants on the credential side, each login needs human approval, credentials travel end-to-end encrypted and never enter the model's context, it's built with Browserbase. The mature version of the vault-brokered pattern is tied to the one cloud browser that says no on-prem.



Corn

So the best credential story is welded to the thing he doesn't trust.



Herman

That's the structural tension, yes. And 1Password Unified Access, launched March 2026, names Anchor Browser, Browserbase, Kernel and Perplexity Comet as the AI-browser integrations. So 1Password has picked its partners, and only one of them offers anything resembling self-deployment.



Corn

The open-source tier. How maintained are we talking?



Herman

Seriously maintained, with an asterisk. Skyvern, about twenty-three thousand stars, shipped November 6th, actively released. AGPL-3.0. Steel, Apache-2.0, around eight thousand stars, but still beta-numbered, version zero point five point five, still tagged beta, at the end of October. Browser Use, an MIT library, a hundred and twenty-one thousand stars, very active, the library self-hosts fine, but live view and human-in-the-loop are cloud features, no on-prem tier. Kernel publishes browser images under Apache-2.0 but no self-host or BYOC offering at all.



Corn

That asterisk is the licence thing, isn't it.



Herman

It is. Browserless is SSPL and Skyvern is AGPL. Both are open source on the tin, but commercial or hosted use needs a paid licence or a legal review. Steel and Browser Use are permissive, Apache and MIT, do what you like. That's the difference between "we can evaluate this" and "we can ship this in a business." A hobby project with an SSPL licence and a hobby project with an MIT licence are two different legal animals, even at the same star count.



Corn

Twenty-three thousand stars and you still have to talk to a lawyer. That's the internet, I suppose.



Herman

Stars measure interest, not indemnification. And for a business the crossing point from hobby to production isn't features, it's who answers the phone when it breaks.



Hilbert

Hilbert's Anteater Trivia. Giant anteaters find their food mainly by smell. I have worked alongside two hosts for years and found most of what I know the same way, by sniffing around until something turned up.



Corn

I'm choosing to take that as a compliment.



Herman

So the landscape's the easy part. The harder question is the mechanism, because two mechanisms decide whether Daniel's trust boundary is real or marketing: how a human actually takes over for a login, and how a credential gets into the field without the model



Herman (continued)

seeing it.



Corn

Take the takeover first. I've never understood what that looks like in practice.



Herman

Browserless calls it hybrid automation. The script runs, hits a login or a 2FA prompt or a CAPTCHA, pauses, and hands a person a live URL. You open it, you're looking at the actual session, you clear the login yourself, the script resumes. The human does the one thing humans are for, and the agent never learns anything from it.



Corn

That's the bit I'd want in the air-gapped container.



Herman

And here's the honest problem: the docs say paid plans, and whether hybrid automation is in the self-hosted Enterprise image is unconfirmed. Not documented either way. AWS AgentCore's Live View is the same mechanism done as a first-class feature, watch and interact directly. Microsoft's Copilot Studio pauses for supervision and times out if nobody comes. Anchor's BYOC page doesn't mention live view at all. So the pattern to name plainly: live takeover is often a cloud feature, and the self-hosted edition may not have it. Any business evaluating this should ask every vendor, in writing, does live takeover work inside the self-hosted edition. That exact question is the one nobody documents.



Corn

Which means the sales call is load-bearing. Depressing.



Herman

It's the price of the market being this young. Now the credential side, because this is where it gets analytically interesting. There are really three patterns, ranked by what the model sees. Worst: the agent types the password. The secret is in the prompt, in the context, in any screenshot the agent takes to verify it worked. That's the default in most hobby setups and it's indefensible for anything that matters.



Corn

The model reading the password over your shoulder.



Herman

Second: injection at the browser layer. Anchor does this, and Skyvern does it self-hosted, with Bitwarden or a cloud key vault behind it, though its 1Password connection is documented only for its cloud. The model sees a placeholder, a token that stands in for the secret, and the real value is typed only into the field at the last moment, by the browser, not by the agent. Skyvern masks the value in screenshots afterwards, so even the visual record is redacted. The model can use the credential without ever knowing it.



Corn

Like handing a courier a sealed envelope.



Herman

That's roughly the shape of it, yes. Third, the strongest: vault-brokered with human approval per login. That's 1Password Secure Agentic Autofill, and Bitwarden's Agent Access SDK. The vault releases the secret only when a person approves that specific login, the credential travels encrypted, and the model is structurally outside the loop. Each login costs you a click. The trust is as good as the approval.



Corn

Where does the pattern leave Daniel, self-hosting?



Herman

With a genuine tension. The mature vault-brokered version, 1Password plus Browserbase, is tied to the cloud browser he doesn't want. The self-hosted versions exist but are rougher. 1Password's Kubernetes Operator works with a self-hosted 1Password Connect server or a service account, so the token stays in your own cluster, that part is real and documented. Bitwarden's Agent Access SDK is open source, just-in-time access, human approval, end-to-end encrypted, and Bitwarden's own page labels it alpha and says don't use it in production. Credit to them for saying it. So the strongest pattern is available self-hosted, in pieces, some of which are marked not-ready.



Corn

Alpha with a warning label is at least honest alpha.



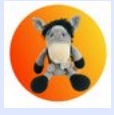
Herman

More honest than most. Now the synthesis, and this is the part I'd actually tell a business. The full chain Daniel wants, self-deployment, human live takeover, 1Password injection with nobody outside seeing the secret, is not publicly documented as a single product. You can assemble it: Browserless Enterprise for the browser, a self-hosted 1Password Connect server for the vault, plus a live-view mechanism you build or buy. But then you're the integrator, and Daniel's whole point was not betting on something that needs an integrator to keep existing.



Corn

Anchor's the closest on the 1Password side.



Herman

Closest on paper. But it's sales-gated, and the BYOC details aren't public, so "closest" means "fewest unknowns claimed," not "verified." AWS AgentCore gives you live view and your own account but is AWS-managed and doesn't document 1Password injection at all. Every option owns one leg of the chain and shrugs at the other two.



Corn

So the second-order point. Daniel said self-hosted browser fine, SaaS browser not fine. Is that the right line?



Herman

It's the right instinct, and it's incomplete. The trust boundary is only as strong as the weakest link in the chain, and there are two chains. A self-hosted browser with a cloud-tied credential integration still puts a vendor in reach of the secret, even though the browser never leaves your rack. A cloud browser with a self-hosted vault still puts a vendor in reach of the session. The browser path and the credential path are separate, and either one alone can undo the other. Daniel's instinct covers the browser path and not the credential path.



Corn

Huh. The self-hosted container with a cloud leash on its passwords.



Herman

That's a real configuration someone could ship thinking they were safe. The second thing worth saying for the business setting: the shortlist. Browserless Enterprise for true self-host, air-gap, SOC 2, quote-priced. Anchor BYOC or on-prem if the 1Password integration survives the sales conversation, get it in writing. AgentCore if you already run on AWS and the hyperscaler is the vendor you've decided to trust. And the power-user route, Skyvern or Steel plus a self-hosted vault connection and a live view you wire up yourself. Fine for Daniel. Not fine for the businesses he'd advise, because he becomes the maintenance plan.



Corn

Which is the thing he said he wouldn't bet on. Himself.



Herman

That's the quiet joke of the whole category. The products that would let him stop being the maintenance plan all want something in return: the sales call, the cloud account, or the closed licence.



Hilbert

The building society does the same thing, Herman. I've got a savings account there that only operates through their web portal, memorable word plus a code from the little device, and I've been trying to get an agent to do the monthly transfer since the spring. I got as far as a self-hosted browser, same as your Daniel, because that part's fine. The wall is the



Hilbert (continued)

login page itself. It was laid out by somebody who has never considered that a person might not be sitting in front of it. Third dropdown, the code box below the slogan. I have a printed screenshot of it taped inside the cupboard door with the code box circled in red ballpoint, because I once spent forty minutes on the phone to a support woman explaining which box I meant, and I decided neither of us should go through that again. There's a handwritten log in the same cupboard of every time they've changed the login flow, dates and what moved. Seven changes since March. I'm fairly sure they land just before my transfer date, though I can't prove it. Corn will tell you I keep too many logs. He says it about the studio ones as well. The people who built that portal did one job and went home, and the portal's been standing in for them ever since.



Corn

Seven changes since March, logged by hand. That's the most durable documentation in this entire episode.



Herman

It's also the part of the chain nobody's selling anything for. We've been ranking browsers and vaults, and his wall is the login page itself, designed by someone who assumed a human, always, forever.



Corn

He maintained a paper audit trail of a bank's UI changes out of spite and prudence in equal measure. That's the person the vendors should hire to write their test suites.



Herman

The forward question I'd leave with, then. The market has split, convenient SaaS with the credential story built in, and self-hosted with the credential story left to you, and the open question is whether that gap closes or whether it's structural, because the vendors' business models may depend on being in the path. And as more services go API-less and agent-driven, the login page becomes the integration point. Whoever owns the credential path owns the trust boundary. That's a bigger question than which browser to buy, and it's the one Daniel's actually asking, even though he framed it as a shopping list. His instinct is right. The thing to interrogate isn't the browser, it's the credential path, and nobody's selling the whole chain yet.



Corn

Somebody eventually will, and they'll charge like it.



Herman

Thanks as always to our producer, Hilbert Flumingtop, who logs things none of us would think to log. If this was your kind of episode, go back for episode fifty-seven, From Lawyers in Limousines to Developers in Their PJs; episode twenty, Architectural AI; and episode thirty-five, The Privacy Gap. This has been My Weird Prompts. If today's gap between the products and the trust boundary was useful to you, leave us a review wherever you get the show, it's what gets obscure prompts like this one in front of other people. We'll be back soon.



Corn

See you then.