Download PDF

THE IDENTITY-NATIVE AGENT: PART 3

OpenClaw's Identity Dilemma

Agentic AI has an identity crisis.
Are the new architectures struggling to cope?

February 2026

THE agentic AI industry has arrived, slightly out of breath and visibly alarmed, at a realisation that should have been its starting point: agents need their own identities. VentureBeat is running pieces on it. Y Combinator is funding startups around it. Gartner predicts that by 2028, a quarter of enterprise breaches will trace back to agentic AI attack vectors. The problem is real, the attention is welcome, but the solutions being proposed should give every enterprise architect pause for thought. The industry's immediate instinct, faced with a governance gap, is to build anew. New middleware, new registries, and shiny new frameworks to house them.

But every one of those fledgling architectures brings with it commercial and operational dependencies on a rapidly growing cohort of unproven platforms presenting novel attack surfaces.

the security push…

To understand the threat posed by the viral take-up of novel AI frameworks at scale, it helps to look at what has been happening to date. OpenClaw, the open-source agent framework that went viral in January 2026, has undergone a serious security transformation. Between early February and mid-February 2026, the project shipped over forty dedicated security patches across multiple releases: SSRF hardening with hostname allowlists, directory traversal prevention in the skills system, constant-time secret verification in webhook hooks, and more. The project's creator, Peter Steinberger, joined OpenAI on 14 February, and OpenClaw itself is transitioning into an independent foundation with OpenAI sponsorship.

This is creditable work, done at speed and under huge pressure, with real transparency. It also makes the structural point more clearly than any Medium take-down could. Forty patches, dozens of external security researchers, a complete shift from "open by default" to "secure by default." But the fundamental philosophy causing the problems in the first place has not been revisited. The identity delegation model is baked into every layer of the architecture; the agent still wears the human's lanyard. Revoking the agent's access still means changing the user's own credentials. The Cyber Strategy Institute, reviewing the 2026.2.13 release, asked whether patching an autonomous agent's code is the same as governing its behaviour. Their answer: it is not.

Peter Steinberger's OpenClaw vision remains genuinely, awe-inspiringly, mind-blowingly paradigm-shifting, and will likely change agentic AI forever. This is not a criticism of OpenClaw's engineering. It is an observation about what engineering can fix on its own. OpenClaw's own community recognised this when it proposed an Agent-Blind Credential Architecture in February 2026: a system where credentials would be stored and used without the AI agent ever seeing their values. The proposal is thoughtful. It is also, by its authors' own admission, insufficient. An agent that cannot see your Gmail password but can still send email as you, read your inbox as you, and delete your messages as you has mitigated one attack vector but left the structural problem intact.

the registry trap…

If the open-source community is trying to harden a runtime that was never designed for identity separation, the enterprise world is building something superficially different but structurally similar: centralised identity registries for agents, managed by a new generation of SaaS platforms.

Y Combinator startup Agentic Fabriq is one of the more far-sighted entrants in this space. Their pitch is clear and, on its face, compelling: treat agent identity as infrastructure. Register agents centrally. Bind each agent's permissions to the human behind it. Validate every action before it executes. Monitor everything. The product offers per-user identity binding, granular permissions, real-time guardrails, and an admin dashboard. The Xu team has been publishing genuinely insightful analysis of agent security since late 2025, on hallucinated permissions, on the dangers of naive context passing between agents, on why permissioning will define the next decade of AI. Their instinct that identity should be foundational rather than bolted on is exactly right.

What the registry model does not do, and structurally cannot do, is give the agent its own identity. Under this model each agent "inherits only the access of the human behind it." The agent operates through the user's connected accounts: the user's Slack, the user's Gmail, the user's Notion. The registry manages which subset of the user's access the agent is permitted to exercise. This is identity management. It is not identity separation. When the agent sends an email, the recipient sees the user's name. When it posts in Slack, colleagues see the user's avatar. The governance is real, but it is local. It does not travel beyond the registry's walls. The recipients, the collaborators, the systems on the other end of the transaction, the insurance company assessing liability: none of them can distinguish between the human and the machine.

There is a second concern that enterprise architects should weigh carefully: the registry itself becomes a target. This is not a hypothetical risk. Palo Alto's Unit 42 found that identity weakness was a major factor in nine out of ten incidents, with attackers routinely targeting identity systems alongside endpoints, networks, and SaaS platforms. Building a new centralised identity layer for agents means building a new target.

The 2023 Okta breach is instructive. Attackers compromised Okta's support system and ultimately accessed data on all 18,400 customers. BeyondTrust, Cloudflare, and 1Password were among those who found out the hard way that when you centralise identity

delegation, you amplify the blast radius when it fails. A centralised identity provider is, by definition, a single point of failure affecting every organisation that depends on it. A centralised agent identity registry creates exactly the same topology. Breach the registry, and you compromise the governance layer for every agent it manages. And the dependency runs both ways: both sides of every transaction must be members of the platform, which means the registry becomes a chokepoint not just for the deploying organisation but for every counterparty it touches.

members-only vs open protocol...

There is a useful distinction hiding in the gap between these two approaches. A SaaS registry verifies that the member is authorised: the agent is registered, the human behind it is known to the platform, the permission set has been configured. The trust depends on the registry's continued correct operation. If the registry goes down, governance stops. If the registry is breached, governance is compromised across every agent it manages.

By contrast, an identity-native agent verifies that the protocol is sound. The gatekeeper enforces permissions deterministically. The identity is inherent in the architecture. The audit trail is end-to-end, visible to every system the agent touches, not maintained by opaque proprietary middleware. Governance continues regardless of any third party.

One model creates an operational and commercial dependency on a vendor. The other creates an architectural property of the system.

the differentiator...

Both OpenClaw and the SaaS registries are trying to govern agents that operate as their users. The HACCU IN Agent framework starts from the opposite premise: the agent operates as itself. This is not a difference of degree. It is a difference of kind.

You cannot build an IN Agent on top of OpenClaw. The entire runtime assumes the agent *is* the user. Every channel integration authenticates with the user's session. Every skill runs within the user's trust boundary. There is no insertion point for a gatekeeper because the architecture has no concept of a boundary between the agent and the user's resources. The identity assumption is not a feature that can be toggled. It is the foundation on which everything else is built.

The IN Agent starts from the other end. The agent appears on platforms as a separate operational identity, with discrete access, wielding only the permissions its human sponsor has explicitly provisioned. When it sends a message, it sends it as itself. When its access needs to be revoked, you deactivate the agent's account, not the sponsor's. A compromised agent process inherits nothing, because the gatekeeper sits between it and the world and does not take instructions from it.

the protocols already exist…

This brings us to perhaps the most under-appreciated aspect of the identity-native approach, and the one that matters most for the long-term direction of agentic AI.

Every new agent framework being built right now is inventing bespoke communication suites on top of the standard platform APIs. OpenClaw uses the same messaging APIs that any bot developer would, but wraps them in its own gateway protocol, its own skills execution layer, and its own agent-to-agent coordination mechanisms. The SaaS registries have their proprietary mediation layers. Every multi-agent coordination framework is designing bespoke protocols for agents to discover, authenticate, and communicate with each other. The platform-facing integrations are standard, but the plumbing between them and the agent is new, unproven, and carries the same classes of vulnerability we have already seen in OpenClaw's first month of existence. New code has new bugs. New protocols have novel attack surfaces. And critically, only the niche that uses each architecture has a stake in finding and fixing its flaws.

SMTP has been in continuous production use since 1982. OAuth 2.0 has been an industry standard since 2012. The Slack API, the Discord API, the WhatsApp Business API have each been designed by dedicated security teams, stress-tested by millions of organisations, and hardened through years of adversarial use. When a vulnerability is found in SMTP or OAuth the entire industry mobilises to patch it, because the entire industry depends on it. These are not proprietary architectures. They are universal standards with universal accountability.

An identity-native agent communicates using these existing standards. It sends email via SMTP, it messages via the Slack API, it authenticates via OAuth. It does not need a new agent-to-agent communication standard, because agents that are somebody, with their

own accounts and contact details, can reach each other through the same channels humans already use. Agent A emails Agent B. Agent B's gatekeeper checks its contact list, verifies Agent A is authorised, and routes the message to its reasoning core. The protocol is SMTP. The authentication is standard. The governance is the same mechanism that applies to every other interaction in the system, human or otherwise. No special case. No new attack surface. No proprietary middleware.

This is arguably the strongest argument for identity-native agents in a multi-agent world. When agents operate as users, every coordination mechanism between them has to be invented from scratch, because the platforms they inhabit were never designed for machine-to-machine communication wearing a human mask. When agents operate as themselves, with their own identities and their own accounts, they inherit the full stack of communication infrastructure that humans have been building, testing, breaking, and rebuilding for decades.

who needs this…

Anyone who carries professional indemnity insurance, which is to say anyone from a sole-trader plumber to a FTSE 500 board member, has a professional and legal need to show they took reasonable steps to prevent harm. The identity-native model provides that by default, at the architectural level, with an audit trail to prove it.

The liability question cuts differently depending on which architecture you choose. When a centralised registry fails, and the Okta breach demonstrates that centralised identity providers do fail, the blast radius is industry-wide. Every agent governed by that registry loses its governance simultaneously. Every organisation that depends on the registry's correct operation has its compliance posture invalidated in a single incident. The liability exposure is correlated and catastrophic. When an identity-native agent is compromised, the blast radius is bounded by the agent's own permissions: the contacts on its approved list, the platforms it has accounts on, the scope its sponsor explicitly granted. The damage is real but contained. The sponsor's own credentials are never exposed, because the agent never had them.

the simpler answer…

The industry's instinct, faced with the agent identity problem, has been to build afresh. OpenClaw's community is building credential vaults and security wrappers. SaaS startups are building centralised registries. Multi-agent coordination projects are building bespoke communication protocols. Each of these adds complexity, adds dependencies, adds attack surface, and adds cost. Each requires the industry to trust new things: new code, new vendors, new architectures that have not yet been tested by time or adversarial use.

The identity-native model takes a fresh look at existing capacity. It makes agents into the kind of thing that existing governance already applies to. The architecture it relies on is mature, universal, and built on decades of standards-based best practice that every organisation already depends on and every vendor already has a stake in maintaining. The protocols that will secure the agentic AI landscape were not designed for AI agents at all. They were designed to serve identity-native human beings, and they stand ready and waiting to serve their new agentic users in exactly the same way.

■

Thoughts? Feel free to get in touch.

haccu.ai