The Enterprise AI Reference Model (EARM™)

The layers you didn't know you needed: a seven-layer reference model for securing and governing AI in the enterprise

CTRL ALT Elite · Fortune 100-caliber infrastructure, operations, and security leadership

By René López, Founder and Principal · ORCID https://orcid.org/0009-0008-4993-0088

BEFORE YOU READ THIS

This is the sequel to The Layer You Don't Think You Need (ctrlaltelite.io/whitepaper). That paper made one argument: governed orchestration and MCP are the same enterprise decision you already made for the hypervisor, centralized logging, and single sign-on. Everyone dismissed each of those as middleware until cost, an audit, or sprawl made them mandatory. If you have not read it, this paper stands on its own; the first one explains why Layer 4 below gets its own floor.

This paper goes one step further. It turns out there is not one layer. There are seven. Together they make up the Enterprise AI Reference Model, or EARM, which this paper introduces. Most enterprises can name two of its layers. The rest are the layers you didn't know you needed.

It is written the way the OSI model is taught: bottom to top, one layer at a time, in plain language. If you can explain to a board why a router is not a firewall, you can explain this. No vendor is required reading. A few are named so you know where they actually sit.

INTRODUCING EARM™

The Enterprise AI Reference Model

The Enterprise AI Reference Model (EARM) is a seven-layer model, introduced in this paper by CTRL ALT Elite, that defines the controls an enterprise needs in place before AI agents can run safely in production: from the hardware they run on, through the sandbox, the model gateway, tool access, agent identity, and cost controls, up to the governance layer that proves all of it happened.

EARM does for enterprise AI what the OSI model did for networking. Each layer has one job and one owner, so a leadership team can say exactly which layer a risk, a vendor, or a budget line belongs to.

Shorthand: the layers are EARM-1 through EARM-7, counted bottom to top. "That is an EARM-4 gap" means the agent can reach a tool with nothing governing the call.

Why Enterprise AI Needs an OSI Model

An AI agent is not a chatbot. A chatbot answers. An agent acts: it reads your files, calls your APIs, moves your data, and spends your money, usually without a human watching each step. That is a new kind of employee, and your security stack was built for the old kind.

Networking solved a similar problem forty years ago. Nobody could reason about "the network" as one thing, so the OSI model split it into seven layers, each with one job and one owner. A cable problem is not a routing problem. A routing problem is not an application problem. The model did not make the network safer by itself. It made it possible to argue about who owned what.

Agent security is in the same place today. Vendors sell "AI security" as if it were one thing. It is seven things stacked on top of each other, and every vendor covers a different slice. Until you can name the slices, you cannot tell whether you have bought a kill switch or a dashboard.

Three rules carry over directly from OSI into EARM:

1. **Each layer trusts only the layer below it.** A control that lives lower in the stack survives a compromise higher up.
2. **Each layer has exactly one owner.** Two owners is zero owners.
3. **A problem is fixed at the layer where it lives.** Patching a tool-access problem at the model gateway does not work, because the gateway never sees the tool call.

EARM at a Glance: The Seven Layers

FIGURE 1: THE ENTERPRISE AI REFERENCE MODEL (EARM)

Layer	What lives there	Owner
EARM-7 Governance and compliance	Policy as enforceable rules, framework mapping, automatic evidence, tamper-evident audit trail	Risk, compliance, and internal audit
EARM-6 FinOps and cost	Token metering per agent and team, budget caps, circuit breakers, chargeback	FinOps
EARM-5 Agent identity	Registry of agents and non-human identities, short-lived credentials, attestation	Identity and access management
EARM-4 Tools and MCP	MCP servers and gateways, tool allow-lists, per-agent permissions, tool-call audit	Integration or platform team
EARM-3 Model gateway	Routing, rate limits, prompt-injection screening, DLP, PII masking, token meter	Platform team
EARM-2 Runtime sandbox and kill switch	Containers, sandboxes, sidecars, egress limits, emergency stop	Platform engineering; security holds the switch
EARM-1 Hardware and hosting	GPUs, CPUs, DPUs, on-prem racks, your cloud tenancy	Infrastructure

Read it bottom to top. EARM-4, tools and MCP, is the one the first paper was about and the one most buyers still skip. A gateway at Layer 3 never sees it.

EARM-1: Hardware and Hosting

The physical floor: the GPUs, CPUs, and network cards the agent runs on, and whether that silicon is yours or a cloud provider's.

Kitchen-table version: this is the building. If you rent an office, the landlord owns the walls. If you own the building, you can put a guard in the basement.

What lives here: GPU and CPU fleets, data processing units (DPUs), on-prem racks, your cloud tenancy.

What the control looks like: a watchdog on separate silicon that keeps working even if the host running the agent is fully compromised. NVIDIA's Sentry on BlueField DPUs is the first mainstream example, announced September 28, 2026.

Who owns it: infrastructure. If the model is hosted by a provider, you do not own this layer and no vendor can sell it to you.

The mistake: assuming your cloud provider's security is your Layer 1. It is theirs. Yours starts at Layer 2.

EARM-2: Runtime Sandbox and Kill Switch

The cage the agent runs inside, and the switch that stops it.

Kitchen-table version: the contractor works in your house, but the door to the kids' rooms is locked and you can send them home at any moment. This is the layer a companion paper, Please Stop Giving Your AI a Shell, was begging you to build.

What lives here: containers, sandboxes, sidecars, process and file-system limits, network egress limits, and the emergency stop that kills one agent or quarantines the whole fleet.

What the control looks like: a sandbox that enforces what the agent may touch (files, tools, network, credentials), plus a kill switch measured in milliseconds and tested in a drill, not in a slide. Open-source options exist (NVIDIA OpenShell). Commercial options bundle it with a console (RuntimeAI's kill switch runs through an Envoy sidecar).

Who owns it: platform engineering, with security holding the kill switch.

The mistake: giving the agent the same shell and credentials as the engineer who built it. Then there is no cage, only a very fast employee with your keys.

Why it matters more than the layers above: everything from Layer 3 up can be fooled by a stolen credential or a poisoned prompt. Layer 2 does not care why the agent went wrong. It only cares that it did.

EARM-3: Model Gateway

The single door every prompt walks through on its way to a language model, and every answer walks back through.

Kitchen-table version: the front desk. Nobody goes straight to an office. The desk logs who came in, checks the bag, and sends them to the right room.

What lives here: routing across model providers, caching, rate limits, prompt-injection screening, data loss prevention on both the prompt and the response, PII masking, and the token meter that FinOps will need later.

What the control looks like: an AI gateway. Cloudflare AI Gateway and the open-source LiteLLM are the common ones; Palo Alto's Prisma AIRS and RuntimeAI's AI Firewall sit in the same spot with more inspection. Enforcement must be inline and fast; the credible claims are under 50 milliseconds.

Who owns it: the platform team, with security owning the DLP rules.

The mistake: thinking this layer secures the agent. It secures the conversation with the model. The agent's tool calls, file reads, and API requests never pass through here. That is Layer 4, and it is the one most buyers forget.

EARM-4: Tools and MCP

The layer where the agent stops talking and starts doing. This is the layer the first paper was about, and it is still the one you don't think you need.

Kitchen-table version: the house with thirty hidden keys. Every tool an agent can call (a CRM, a database, a payment API, a shell) is a key. Model Context Protocol (MCP) is the keyring. Without a gateway on the keyring, any key the agent finds, it can use.

What lives here: MCP servers and gateways, tool allow-lists, per-agent tool permissions, tool-poisoning detection, and the audit log of every tool call with its arguments and results.

What the control looks like: an MCP gateway or portal that every agent must use to reach any tool, with the tool catalog curated by a human. Cloudflare's MCP server portals went GA on September 24, 2026; Zscaler's AI Broker brokers MCP and agent-to-agent traffic; RuntimeAI ships an MCP gateway with mTLS and tenant ACLs. Cloudflare's own Gateway can now detect raw MCP traffic and block anything not arriving through a portal.

Who owns it: the integration or platform team, with identity owning the permission model.

The mistake: letting developers point agents straight at MCP servers because "it's just an API." In September 2026, GreyNoise tracked AI agents built on commercial coding models exploiting two PaperCut flaws to breach 395 organizations in 48 countries, reaching initial compromise at eleven of them in 26 seconds. The agents did not talk

their way in. They called tools, and nothing stood between the call and the system. One vendor's tally of the month's incidents put AI-agent exploits ahead of credential theft for the first time; treat that ranking as a vendor count, but the direction matches what GreyNoise, Okta, and Google's threat intelligence group reported independently the same week.

Why a gateway at Layer 3 cannot save you here: the model gateway sees the prompt and the answer. It never sees the agent call the payment API with the wrong account number. Different path, different layer, different control.

EARM-5: Agent Identity

Who is this agent, who does it work for, and what is it allowed to be.

Kitchen-table version: the badge. Every employee has one. Every contractor has one. Your agents have the engineer's badge, borrowed, and nobody knows how many copies exist.

What lives here: a registry of every agent and every non-human identity (service accounts, API keys, OAuth tokens), short-lived credentials instead of permanent ones, hardware attestation where you can get it, and the rules for agents talking to other agents.

What the control looks like: an agent registry tied to your identity provider, with each agent issued its own cryptographic identity and short-lived certificates. Okta's Agent Gateway and Zscaler's registry extend existing IAM; RuntimeAI issues Ed25519 agent IDs with SPIFFE certificates. The clearest recent failure is the September 2026 PaperCut campaign: the accounts the attackers' agents harvested were service accounts with domain-level privileges, no expiration, and no identity governance, and Active Directory credentials were pulled from 280 of the 395 victims. In a separate case disclosed August 31, an attacker lifted a model-provider API key from an exposed agent dashboard and ran roughly $600,000 of usage on it over three weeks before anyone noticed. Both are Layer 5 failures, and neither is exotic.

Who owns it: identity and access management. Not the AI team.

The mistake: treating agents as applications. An application has one owner and one job for years. An agent is spun up by a developer on Tuesday, given four tools, and forgotten by Friday. Its credential is not forgotten.

EARM-6: FinOps and Cost

What the agents are spending, who is paying, and the switch that stops a runaway bill.

Kitchen-table version: the credit card with a limit. You would not hand a contractor a card with no limit and no statement. An agent in a loop is a contractor who never sleeps.

What lives here: token-level metering per agent and per team, budget caps and circuit breakers, model routing by cost, anomaly detection on spend, and chargeback to the business unit that owns the agent.

What the control looks like: the meter comes from Layer 3 (every gateway counts tokens), the caps are enforced at Layer 2 or Layer 3, and the reporting lands in FinOps. Cloudflare and LiteLLM give you the meter for close to nothing. RuntimeAI adds per-agent caps and routing it claims can cut spend by up to 60 percent; treat that as a vendor number until you have measured it on your own traffic.

Who owns it: FinOps, using data the platform team collects. This is the layer where the CFO first takes the stack seriously.

The mistake: discovering the cost problem from the invoice. One agent retrying a failed tool call in a loop can spend a quarter's AI budget overnight. The budget cap is a security control, not an accounting nicety.

Why it sits above the gateways: Layer 6 consumes the gateways' metering. Whoever owns Layer 3 effectively owns the data Layer 6 runs on, which is why the same vendors show up at both.

EARM-7: Governance and Compliance

The layer that proves the other six happened.

Kitchen-table version: the receipts. You locked the doors, you issued the badges, you set the card limit. Now the auditor asks you to prove it for every agent, every day, for the last year.

What lives here: written policy translated into enforceable rules, the mapping of those rules to the frameworks your auditors use (SOC 2, NIST AI RMF, ISO 42001, the EU AI Act, CMMC 2.0 for defense work), evidence collected automatically from the lower layers, and a tamper-evident audit trail.

What the control looks like: policy written once in plain language and compiled into rules the gateways enforce, with evidence generated on every agent action rather than assembled by hand before the audit. This is the layer RuntimeAI leads with (60-plus pre-mapped frameworks, a Merkle-chain audit log). Palo Alto and Zscaler cover it from the network side. Cloudflare and NVIDIA do not play here at all.

Who owns it: risk, compliance, and internal audit, with security as the enforcer. These are the people who kill AI programs when the receipts are missing, and they are right to.

The mistake: buying Layer 7 first. A compliance dashboard with nothing underneath is a very expensive way to document that you have no controls. The EU AI Act's transparency obligations took effect on August 2, 2026. The high-risk obligations that were due the same day have been pushed to December 2, 2027 under the Digital Omnibus agreement, pending formal adoption. The delay is not a reprieve. It is sixteen months to put Layers 2 through 5 in place so that Layer 7 has something to read from.

Hosted Model or Your Own Cloud: What Changes

Where the model runs changes the floor of EARM, not the ceiling. The seven layers are the same either way.

FIGURE 2: HOSTED VERSUS SELF-HOSTED

Question	Hosted model API (OpenAI, Anthropic, Bedrock)	Self-hosted in your cloud or on-prem
Lowest layer you own	Layer 2 (sandbox and kill switch)	Layer 1 (hardware)
Hardware watchdog possible	No; the provider owns the silicon	Yes, on DPUs or dedicated CPUs
Where containment lives	A sidecar or sandbox you run	Sandbox plus silicon watchdog
Model gateway	Mandatory; it is your only inline point before the provider	Still mandatory; also your routing between internal models
Data residency	Provider's region and terms	Yours; air-gapped is an option
Cheapest credible stack	Cloudflare gateway and portals, your IdP, a sandbox	Open-source sandbox, LiteLLM, your IdP
When to add a commercial control plane	Regulated data, many agents, or an audit on the calendar	Same, plus any fleet above a few hundred agents

Most enterprises run both. Customer-facing agents call hosted frontier models; internal agents on sensitive data run self-hosted. Design the stack once and let the floor move.

Who Sells What: A Vendor Map by Layer

No vendor covers all seven EARM layers well. The table shows where each one actually enforces, as of October 15, 2026. A blank cell means the vendor does not play at that layer, which is useful information. One disclosure: RuntimeAI appears often in this paper because it is the only vendor claiming six of the seven layers in one product. It is also a pre-seed company, and the incident statistics it publishes are its own counts. Weigh both facts.

FIGURE 3: VENDOR COVERAGE BY EARM LAYER

Layer	NVIDIA Open Agent Safety	RuntimeAI	Palo Alto Prisma AIRS	Zscaler AI Security	Cloudflare	Open source (LiteLLM, OpenShell)
7 Governance		Lead strength	Yes	Yes		
6 FinOps		Yes			Metering only	Metering only
5 Identity		Yes	Yes	Yes (AI Broker, registry)	Access policies	
4 Tools and MCP		Yes	Agent Gateway (preview)	Yes (AI Broker)	MCP portals	
3 Model gateway		Yes	Yes	Yes (AI Guard)	Lead strength	LiteLLM
2 Sandbox and kill switch	OpenShell	Sidecar kill switch				OpenShell
1 Hardware	Sentry on BlueField					
Pricing model	Free software, NVIDIA hardware	Custom, all tiers	Credit-based, quote only	Quote only	Free core, usage-based	Free
Maturity	Launched Sept 28, 2026	Pre-seed startup	Third major release	Mostly acquired in 2026	Mature, GA	Mature

Read the columns, not the rows. The incumbents (Palo Alto, Zscaler) cover Layers 3 through 7 from the network side and are strongest where you already pay them. Cloudflare is the cheapest credible Layer 3 and Layer 4 and stops there. NVIDIA owns the bottom two layers and nothing else. RuntimeAI is the only one claiming Layers 2 through 7 in one product, which is both its pitch and its risk.

Five EARM Buying Rules

1. **One owner per layer.** Write the seven EARM layers and seven names down before any vendor meeting. If two layers have the same name, that person is the bottleneck. If a layer has no name, that is your first purchase.
2. **Build from the bottom up.** Layer 2 before Layer 7. A kill switch with no compliance dashboard is a control. A compliance dashboard with no kill switch is a liability with a logo.
3. **Protect both paths.** The agent-to-model path (Layer 3) and the agent-to-tool path (Layer 4) are different wires. A gateway on one does nothing for the other. Ask every vendor which path they sit on; the honest ones will say one.
4. **Measure the latency claim on your traffic.** Sub-50-millisecond enforcement is credible in-region and optimistic across regions. A proof of concept on real agents, not a demo on the vendor's laptop.
5. **Refuse any vendor that claims all seven.** Nobody has it. The ones who say they do are selling a roadmap. Buy the layers they are strongest at, and make them show you a customer running it at your scale.

And one rule for the budget: the cheapest credible stack is a free sandbox, a free gateway, and the identity provider you already own. Spend money when you have regulated data, more than a few hundred agents, or an audit on the calendar. Spend it at the layer where the gap is, not at the layer with the best demo.

The Point

The first paper argued that you would eventually buy the governed orchestration layer, the same way you eventually bought the hypervisor and single sign-on, and that moving early put the work in the file marked architecture instead of the file marked remediation.

This paper's argument is simpler. There is no single layer to buy. There is a stack, the Enterprise AI Reference Model, it has seven floors, and the agents are already in the building. The companies that come through the next three years cleanly will not be the ones with the best AI. They will be the ones who can tell you, floor by floor, who holds the keys.

> *If you cannot name seven owners today, that is not a failure. It is the starting point, and it is a shorter list than you think. Put the seven EARM layers on one page, write a name next to each, and you have the first draft of your enterprise AI architecture.*

Using the EARM Name

EARM is free to use, teach, and build on. Put it in your architecture reviews, your board decks, and your RFPs; ask vendors to state which EARM layers they cover. The one request is attribution, so the model stays one model instead of seven slightly different ones.

Cite it as: López, R. (2026). The Enterprise AI Reference Model (EARM). CTRL ALT Elite LLC. ctrlaltelite.io.

About the Author

René López is Founder and Principal of CTRL ALT Elite LLC, a fractional CIO, CTO, and CISO advisory practice for Series A and B startups and private-equity-backed operators. He brings 25 years of Fortune 100 infrastructure and operations leadership, most recently as VP of Infrastructure and Operations at Royal Caribbean Group and Managing Director of Enterprise Applications, Infrastructure and Operations at Delta Air Lines. He completed the EC-Council Certified CISO (C|CISO) program in 2026, is a FinOps Certified Practitioner, and co-authored SAN patents at Vicom Systems.

CTRL ALT Elite

reboot@ctrlaltelite.io · 954-516-6909 · ctrlaltelite.io · Pompano Beach, FL

René López, Founder and Principal · ORCID https://orcid.org/0009-0008-4993-0088

Disrupt · Innovate · Elevate

© CTRL ALT Elite. Shareable in full with attribution. EARM™ and Enterprise AI Reference Model™ are trademarks of CTRL ALT Elite LLC. Vendor-neutral; no product is endorsed. Vendor capabilities and dates current as of October 15, 2026 and subject to change; verify against primary sources before relying on them.

Sources: RuntimeAI platform and pricing (runtimeai.io); Palo Alto Networks, Prisma AIRS 3.0 release; Zscaler, Zenith Live 2026 announcements; Cloudflare, MCP detection and dashboard changelog (August 12, 2026); TechCrunch, NVIDIA launches Open Agent Safety Platform (September 28, 2026); VentureBeat, AI agents breached 395 organizations, citing GreyNoise, Okta, Google GTIG, and METR; Gibson Dunn, EU AI Act Digital Omnibus agreement.