empowerID

WHITEPAPER · LLM GATEWAY

Authorization before inference

How the EmpowerID LLM Gateway governs model access in an Identity Fabric — verified identity and delegation, prompt intent as policy context, budgets enforced before the provider call, credentials kept out of clients, and signed evidence for completed allowed use.

July 2026 For security & identity leaders, AI platform teams, and FinOps

EXECUTIVE SUMMARY

Copilots, workflows, and agents now choose models, consume data, and act on behalf of human and machine subjects. Conventional AI gateways route, retry, cache, meter, and filter that traffic. They do not decide whether a governed subject, acting under a specific authority, may use a specific model for the apparent purpose of the request, within its current spending boundary — or preserve evidence that links the decision to the model use that followed.

The EmpowerID LLM Gateway makes that decision at the model boundary. It is a specialized policy enforcement point (PEP) for model invocations — not a reverse proxy with extra logging, and not a feature of the identity provider. It receives governed identity context, enriches the request with high-

confidence prompt classifications and estimated cost, obtains a decision from the shared EmpowerID policy decision point (PDP) — the engine that evaluates policy and decides — enforces model and token constraints before inference, keeps provider credentials out of clients, settles measured usage, and creates signed receipts for completed allowed calls. When an agent moves from reasoning to action, the separate MCP Gateway governs tool execution against the same policy plane: one policy plane, two specialized enforcement points.

01 — THE PROBLEM

A provider key is not an identity

A model request may come from a person in a copilot, a browser plugin inside a signed-in session, an application workload, an agent acting for a person, one agent delegating a subtask to another, or a scheduled process with limited organizational authority. The call may influence which record is opened, which tool is selected, which transaction is proposed, or which data leaves the enterprise. Possession of a provider key proves only that a client can reach a provider. It does not establish whose business authority is being exercised, or whether that authority permits the current intent. Three gaps follow:

ANONYMOUS AUTHORITY

A shared provider key becomes the effective identity. The log shows an application called a model — not which person, workload, or delegated agent exercised whose authority.

POLICY ON PAPER

Acceptable-use rules, model approval lists, and data-handling requirements live in documents and ticket workflows while model requests execute continuously, unexamined at runtime.

SPEND REPORTED TOO LATE

One person can start an agent loop that makes many invocations; a model choice can change cost by an order of magnitude. The invoice reports it after the money is gone.

OWASP names the underlying risk class excessive agency: LLM-based systems granted more functionality, permissions, or autonomy than the task requires, so that hallucination, ambiguity, or prompt manipulation can produce harmful action. Its recommended countermeasures — least privilege, executing in the user's context, authorization for consequential actions — are identity-layer controls. Gartner's 2026 AI-governance guidance reaches the runtime conclusion: automated policy

enforcement, continuous monitoring, and audit-ready evidence are foundational, because point-in-time audits cannot govern systems that act continuously. The FinOps Foundation reports that 98% of its 2026 survey respondents now manage AI spend, up from 31% two years earlier.

> **The governed question.** May this subject, acting under this authority, use this model for this apparent purpose under current policy and spend conditions — and what verifiable evidence links the decision to the model use that followed?

02 — THE CATEGORY

What AI gateways solve, and the question they leave open

Cloud, edge, and API platforms now ship multi-provider access, centralized provider credentials, token metrics, retries and failover, rate and token limits, prompt filtering, spend reporting, and operational logs. These functions are necessary, and buyers should expect them from any serious gateway. What they answer is a traffic question: does this request conform to gateway rules? The open question is an authorization question: may this subject perform this action on this resource under current context? The distinction runs through every control:

Traffic-oriented control	Governed authorization control
API key or gateway credential	Human, application, workload, or agent identity — with delegation
Route and provider	Model resource and allowed use for this subject and intent
Request volume and rate limits	Subject-specific spend state, evaluated before the provider call
Static prompt filter	High-confidence intent and data labels supplied to policy
Configured fallback model	Model switching in which every candidate is re-authorized by policy
Local gateway configuration	Shared policy decision across enforcement points
Operational log	Decision-linked, signed evidence for completed allowed use

A production system needs both columns. Traffic management keeps the path fast and available; governed authorization decides who may use it, for what, and at what cost. Neither substitutes for the other.

03 — HOW IT WORKS

The governed model-invocation lifecycle

On a governed route, every model request runs the same deterministic sequence, even when the policy itself is context-sensitive:

1

Establish the subject

Authenticate through an Identity Fabric session or personal access token; resolve a stable subject and, for agents, the identity chain. A token that names an agent is not enough if the delegated authority behind it has expired, been revoked, or does not cover the requested use.

2

Classify the request

Inline prompt classification derives intent and data context. Only allowlisted, high-confidence labels cross the policy export boundary; broad analytics topics stay out of authorization, so an exploratory taxonomy never becomes an accidental deny engine. The classifier is treated as part of the attack surface, not a trusted oracle: safety guards configured as mandatory fail closed when the classifier is unavailable, and classification supplies context — the PDP makes the decision.

3

Ask the PDP

Send subject, action, resource, and context — identity chain, delegation status, requested model, intent and data labels, estimated cost, current spend state, application and environment — through an AuthZEN-compatible authorization request. The answer can be allow, deny, or allow with constraints.

4

Enforce before inference

A denied request never reaches the provider. An allowed-with-constraints request is narrowed first: clamp the requested token maximum, reject a model not approved for this subject or intent, or switch to a permitted model under policy. An alert after the fact cannot recall data already sent or cost already incurred.

5

Attach the provider credential

Organization-managed credentials are resolved from approved vaults and applied only on the upstream provider connection. The application, plugin, or agent authenticates to the Identity Fabric and never holds the provider key — which supports rotation, provider changes, and central revocation.

6

Relay the compatible request

OpenAI- and Anthropic-compatible routes carry the request body unchanged. Adoption means changing the endpoint and authentication, not rewriting application logic — while identity, policy, constraints, credentials, and provider selection are enforced around the unchanged body.

7

Settle measured usage

After a completed allowed call, the gateway extracts provider usage and settles the spend path. Analytics aggregates consumed spend and exposes it as policy information for the next decision — the estimate authorized the call; the measurement updates the state.

8

Create evidence

Completed allowed calls receive signed, hash-linked receipts binding policy context to measured usage — recording the effective model actually used, not just the one requested. Denied calls generate decision and operational telemetry, not signed receipts, and are never represented as receipt-backed executions.

Figure 1 — The enforcement sequence on every governed model invocation.

> **The invariant.** The PDP is authoritative for business policy; the gateway is authoritative for enforcement at the model boundary. Every stage can narrow the permitted request — model, tokens, provider — and no stage can widen what the PDP granted.

An identity-fabric enforcement point, not another proxy

A standalone gateway can add local rules. An identity-fabric gateway evaluates the same authoritative relationships and logical policy model used everywhere else in the enterprise — the same identity graph, PDP, credential services, and evidence plane that govern human and workload access:

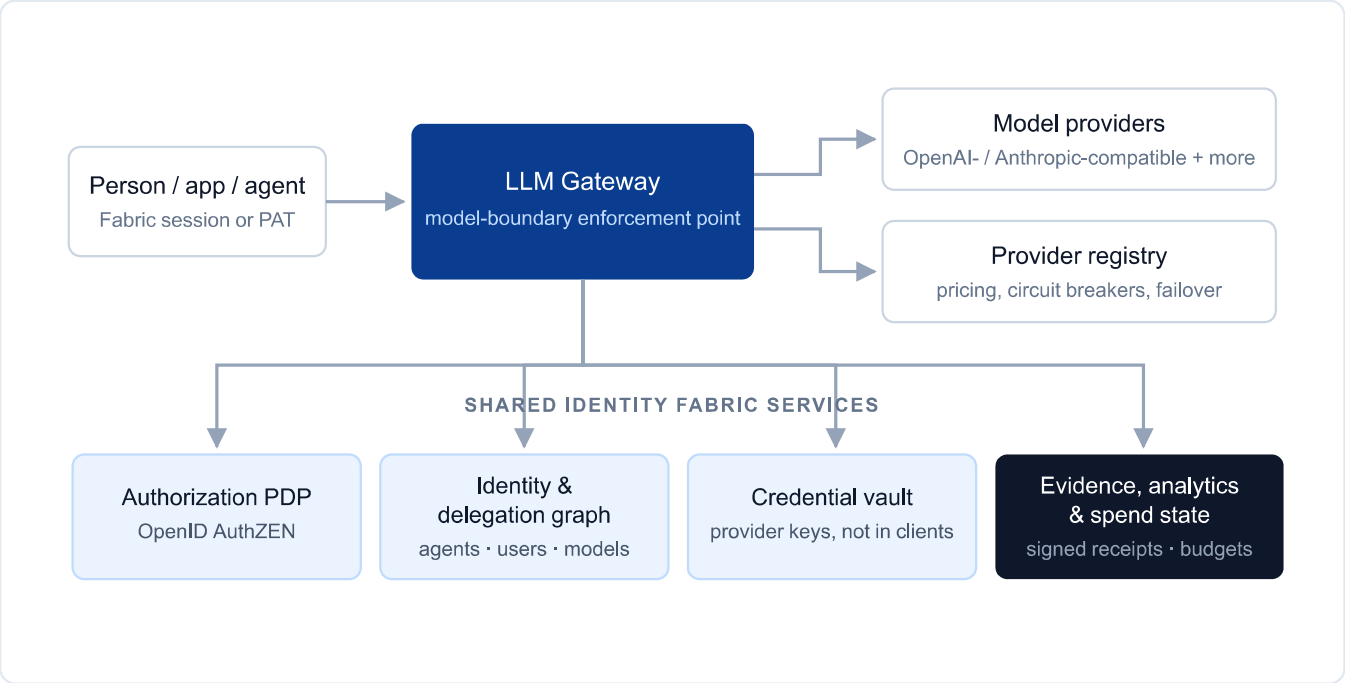

Figure 2 — One service in the Identity Fabric: the gateway enforces; shared services decide, relate, hold custody, and remember.

Dimension	Typical AI gateway	EmpowerID LLM Gateway
Caller identity	API key or virtual key per client	Governed human, application, workload, or agent — with delegation checked at request time
Decision authority	Local rules configured in the proxy	The same logical PDP that governs human and workload access, via OpenID AuthZEN
Prompt meaning	Static filters and blocklists	High-confidence intent and data labels exported to policy; analytics topics kept separate
Cost control	Token quotas and after-the-fact dashboards	Estimated cost and authoritative spend state evaluated in the decision, before the call
Model switching	Config fallback or latency-based auto-router	Policy reroute, budget downgrade, provider failover — every candidate re-authorized
Credentials	Gateway holds keys; clients hold gateway keys	Vault-backed organization keys; clients authenticate with Fabric sessions or PATs only
Evidence	Operational logs and metrics	Signed, hash-linked receipts binding decision, constraints, effective model, and usage

05 — COST GOVERNANCE

Budgets as policy, not metering

A rate limit answers how often a credential may be used. A budget policy answers whether a specific user, application, or delegated agent may consume a model under a time-bound financial boundary. The LLM Gateway treats spend as an authorization attribute: it estimates the request cost from model and token context, presents the estimate with subject attribution to the PDP, and calls the provider only after an allow. Measured usage then updates the aggregated spend state the next decision reads:

This supports outcomes a counter cannot: stop consumption before a limit is exceeded, constrain maximum tokens for a subject or use case, permit a cheaper model when policy allows, attribute spend to the governed identity chain, and keep financial controls consistent across applications.

Failure semantics are part of the design — if a subject has a hard budget limit and the authoritative spend state is unavailable, silently allowing the call defeats the control. Budget enforcement fails closed for limited subjects, optionally with a short, explicitly bounded last-known-good window the organization chooses. One boundary is worth stating plainly: authorization uses the pre-call estimate, and a streamed response is not truncated mid-completion — so a subject near a hard limit can overshoot by at most one completion's difference between estimate and actual. Settlement records the overage in consumed spend, and the next request is denied against it.

06 — MODEL SWITCHING

When the model changes, the policy doesn't

Most gateways can substitute a model — a configured fallback, or an auto-router picking by latency or price. What gets substituted away is the authorization: the replacement runs because a config entry says so, not because policy evaluated it for this subject and intent. In the EmpowerID design, the same governed decision covers three switch paths, and every candidate model is re-evaluated by the PDP before it is used:

`policy reroute`	requested model denied for this subject or intent — the first PDP-authorized candidate from the cost-sorted allowed list serves the unchanged request
`budget downgrade`	estimate breaches the spend boundary — cheaper allowed models are tried instead of a quota error; a use case pinned to one approved model is denied, never downgraded
`provider failover`	provider errors or times out — circuit breakers move the request only where policy permits; availability never overrides data, regional, or subject restrictions

A switch is never silent: the response declares when the effective model differs from the requested one, and the signed receipt records the model actually used. Switching applies to eligible chat routes — embeddings, audio, and image requests are not rerouted, and no switch occurs mid-stream once a response has begun.

07 — MODEL AND TOOL BOUNDARIES

One policy plane, two specialized enforcement points

A model invocation and an enterprise tool call have different risk shapes. Inference risks concentrate around prompts, sensitive input, unapproved models, token and spend exhaustion, and provider behavior. Tool risks concentrate around discovery, schema integrity, parameters, delegation, downstream credentials, and irreversible effects. Governing both through one undifferentiated proxy yields either shallow common-denominator controls or a middleware layer no one can reason about. The Identity Fabric keeps the decision shared and the enforcement specialized:

Concern	LLM Gateway	MCP Gateway
Human, workload, and agent identity; delegation; shared PDP	Yes	Yes
Prompt classification; model and token constraints; AI budgets; model switching	Owner	—
Model-provider credential injection; model-call receipts	Owner	—
Tool discovery scoping; schema pins; parameter constraints	—	Owner
Enterprise connector credentials; tool-execution receipts	—	Owner

An investigator can follow one governed subject from model invocation through enterprise tool execution — correlated evidence on both boundaries, without pretending that generating language and executing a consequential action are the same kind of event.

08 — FOR THE SECURITY ORGANIZATION

Trustworthy failure, honest evidence

Security architecture is revealed when dependencies fail. Each dependency on the governed path has explicit, testable outage behavior:

`PDP outage`	no silent bypass of authorization; any last-known-good window is bounded and explicit
`spend outage`	a limited subject is never assumed to have unlimited funds — fail closed on hard boundaries
`classifier outage`	labels required by policy do not become an implicit allow when the classifier is down
`provider outage`	failover moves the request only to alternatives policy permits — never around policy

Two capacity and placement facts complete the architecture review. First, the PDP is on the hot path: a policy reroute across an allow-list of N candidates costs up to 1 + N evaluations for one user request — entitlement caching for non-sensitive repeat decisions and deliberately narrow allow-lists keep that bounded, and PDP availability is engineered accordingly. Second, the gateway governs the routed path; it cannot govern a developer calling a provider directly on a personal key. It belongs in a stack with egress allow-listing and provider-key governance that make the governed route the practical one, not merely the recommended one.

Evidence carries the same discipline. A useful model-call record lets an authorized reviewer establish which governed subject initiated the call, under whose delegation, which policy decision authorized it, which constraints applied, which model and provider actually served it, and what measured usage resulted. Signed, hash-linked receipts make alteration detectable within the retained chain — tamper-evident, not “immutable.” A receipt proves the recorded decision-and-usage path has cryptographic integrity; it does not certify that the model output was correct, safe, or compliant.

> **Stated boundaries.** Receipts cover completed allowed calls; denied calls produce decision telemetry, not signed receipts. The gateway classifies requests — it does not claim universal output scanning or quarantine, mid-stream truncation on passthrough routes, or per-user provider keys. And no gateway, this one included, establishes regulatory compliance by itself.

Five questions to ask any LLM gateway

1. Can it distinguish people, applications, workloads, and agents — and require active, scoped delegation behind an agent, rather than treating a key as the identity?
2. Is model authorization evaluated by a shared PDP with subject, intent, data, and spend context — or duplicated as local rules in each gateway?
3. Can a budget stop or constrain a request before provider consumption — and what happens when the authoritative spend state is unavailable?
4. Are prompt classifications separated into analytics and policy tiers — with the classifier advisory to policy, observable for drift, and never the business-decision engine?
5. Can a completed allowed call be linked to its policy decision, constraints, effective model, and measured usage in signed, tamper-evident form?

These questions separate a transport proxy, a developer control plane, and an identity-fabric enforcement point. The LLM Gateway is an EmpowerID Identity Fabric service — the same authorization, credential, and evidence services that govern the rest of the enterprise, applied at the model boundary.

KEY TAKEAWAYS

Routing, caching, and token limits are the baseline. The differentiating question is authority: who is acting, under whose delegation, for what intent, within what boundary.

Enforce before inference — a denied request never reaches the provider, and an after-the-fact alert cannot recall data or cost.

Spend is an authorization attribute — estimated before the call, measured after, and enforced before provider consumption rather than reported behind it.

The client holds a Fabric identity, never the provider key. The PDP decides. The gateway enforces. The receipt remembers.

See a model call denied before inference

Watch a governed subject blocked by policy before the provider is called — sensitive intent flagged, a budget stop enforced pre-request, and the signed receipt trail for the calls that were allowed.

Request a demo

info@empowerID.com · empowerid.com

empowerID