empowerID

WHITEPAPER · MCP GATEWAY

Authorization before agent action

How the EmpowerID MCP Gateway turns MCP connectivity into governed, attributable tool execution — verified delegation, scoped discovery, per-invocation policy, protected credentials, and evidence that survives scrutiny.

July 2026 For security & identity leaders and architects

EXECUTIVE SUMMARY

The Model Context Protocol made tool use portable — and that made *action* portable. An agent that can discover a tool can plan around it; an agent that can invoke a tool can change enterprise state. Authenticating to an MCP server is necessary but not sufficient. The enterprise must decide which agent, acting for which identity, may discover and invoke which tool, under which constraints, using which downstream credential — with what evidence.

MCP's OAuth-based authorization answers whether a client may reach an MCP resource. It does not, by itself, answer the fine-grained enterprise question at the moment a model-selected tool is invoked. The EmpowerID MCP Gateway operates at that second layer: an MCP-aware policy enforcement point (PEP) that verifies delegation, scopes discovery, obtains a per-invocation decision from the

EmpowerID policy decision point (PDP) — the engine that evaluates policy and decides — enforces constraints, keeps downstream credentials out of agent context, and records what the boundary decided and observed. MCP carries the call. The fabric establishes the authority. The gateway enforces the decision.

01 — THE PROBLEM

A valid token is not authority for every tool

An API gateway evaluates hosts, routes, methods, tokens, and traffic. Those controls remain necessary — but the semantic unit of MCP is not the HTTP request. It is a model-selected capability: a tool the model chose, a schema it used to form the call, structured parameters, an agent identity, and — usually — delegated human authority behind it. Treating a valid connection as authority for every tool and parameter the server exposes invites three failure classes:

CONFUSED DEPUTY

An agent with a powerful service credential acts for a user who lacks the corresponding authority — and the log shows only the service account.

SILENT CAPABILITY DRIFT

A tool keeps its name but gains parameters or behavior after approval — or a permissive token asserts broader access than the standing delegation permits.

UNTRACEABLE ACTION

Logs show an API call — but not which agent acted for whom, under which delegation, policy, and tool schema, or whether the effect actually happened.

MCP did not create these risks — it concentrates them at one portable, model-driven interface. The risks OWASP's MCP guidance highlights map directly to identity-layer controls:

Risk	Control response at the gateway
Confused deputy	Explicit represented identity, delegation verification, downstream identity binding
Capability inflation	Request-time claims intersected with authoritative delegated capability
Tool / schema drift & poisoning	Approved schema pins, catalog governance, constrained discovery, fail-closed mismatch
Excessive agency	Scoped discovery, tool-specific policy, parameter constraints, confirmation gates
Credential escape / token passthrough	Vault-backed custody, audience-bound tokens, validated exchange — no passthrough
Data exfiltration	Egress allowlists, redirect re-check, authoritative data scope, output controls
Shadow MCP infrastructure	Service registration, fleet inventory, approved catalogs, secret hygiene
Untraceable action	Correlated, signed evidence with explicit receipt semantics

No single layer makes the others unnecessary. Network segmentation, OAuth resource authorization, this semantic enforcement point, downstream application controls, and evidence each keep their own role — the architectural error is assuming a valid connection or token equals authority for every tool and parameter the server exposes.

02 — THE MODEL

“The agent is authorized” collapses three security events

EmpowerID governs agent authority at three distinct moments — the same discipline the Identity Fabric applies to all agent authorization:

1 · Delegation

Who allowed the agent to act? An explicit, bounded, revocable relationship — not "run as the user," and not an unrestricted copy of human access.

2 · Discovery

What may the agent know exists? tools/list is not a global catalog dump — virtual MCP servers expose delegation-scoped views before tools enter the model's plan.

3 · Invocation

May this call be dispatched now? The gateway re-establishes the facts and obtains a current policy decision — every governed invocation, not just at connection time.

> **Discovery is least privilege for cognition.** A later denial still prevents the action — but it does not undo exposure of the tool's existence, its influence on the model's plan, or the opportunity for a poisoned tool description to manipulate selection. Scoping the catalog is the first enforcement moment; invocation is the final one.

03 — HOW IT WORKS

The governed invocation sequence

When an agent calls a tool, the gateway runs a deterministic enforcement sequence — even when the policy itself is context-sensitive:

1

Bind the caller

Validate the token; establish the agent identity and the represented human or service — so the tool name is never the first trustworthy fact in the request.

2

Verify the approved tool contract

Compare the tool schema to its approved pin. Changed outside an accepted upgrade window? The call can fail closed — approving one version of create_user does not approve a later one that can also assign privileged roles.

3

Verify delegation and intersect the ceiling

Check the standing delegation in the membership graph, then intersect it with any request-time capability ceiling — a permissive token can never manufacture authority the graph never granted.

4

Ask the PDP

Send subject, action, resource, and context through the OpenID AuthZEN Authorization API — an OpenID Final Specification since January 2026 — keeping enforcement at the gateway and policy authority at the PDP.

5

Apply enforceable constraints

A binary permit is often too coarse. The gateway narrows the call — accepted parameters, authoritative data scope, permitted destinations, redirects, output bounds, field redaction. For selected high-risk journeys, optional plan contracts bind the invocation to an approved tool, step, and parameter fingerprint — what was reviewed is what gets dispatched. A constraint returned but not applied is not a control.

6

Dispatch without surrendering credentials

Route to a registered MCP server or an EmpowerID connector through Orchestration & Fulfillment. Vault-backed modes inject the downstream credential at the protected outbound boundary — the agent gets the result, never the token.

7

Record the controlled boundary

Signed, tamper-evident receipts on configured governed paths — with structured denial codes that distinguish policy denial, revoked or expired delegation, schema mismatch, and insufficient trust.

Figure 1 — The enforcement sequence at every governed tool invocation.

> **The invariant.** Effective authority = standing delegated authority ∩ request-time capability ceiling ∩ current policy decision. Every stage can narrow the permitted action; no stage can widen it — and no downstream component may widen what the PDP granted.

04 — WHAT MAKES IT DIFFERENT

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 graph, PDP, credential services, and evidence plane that govern human and workload access:

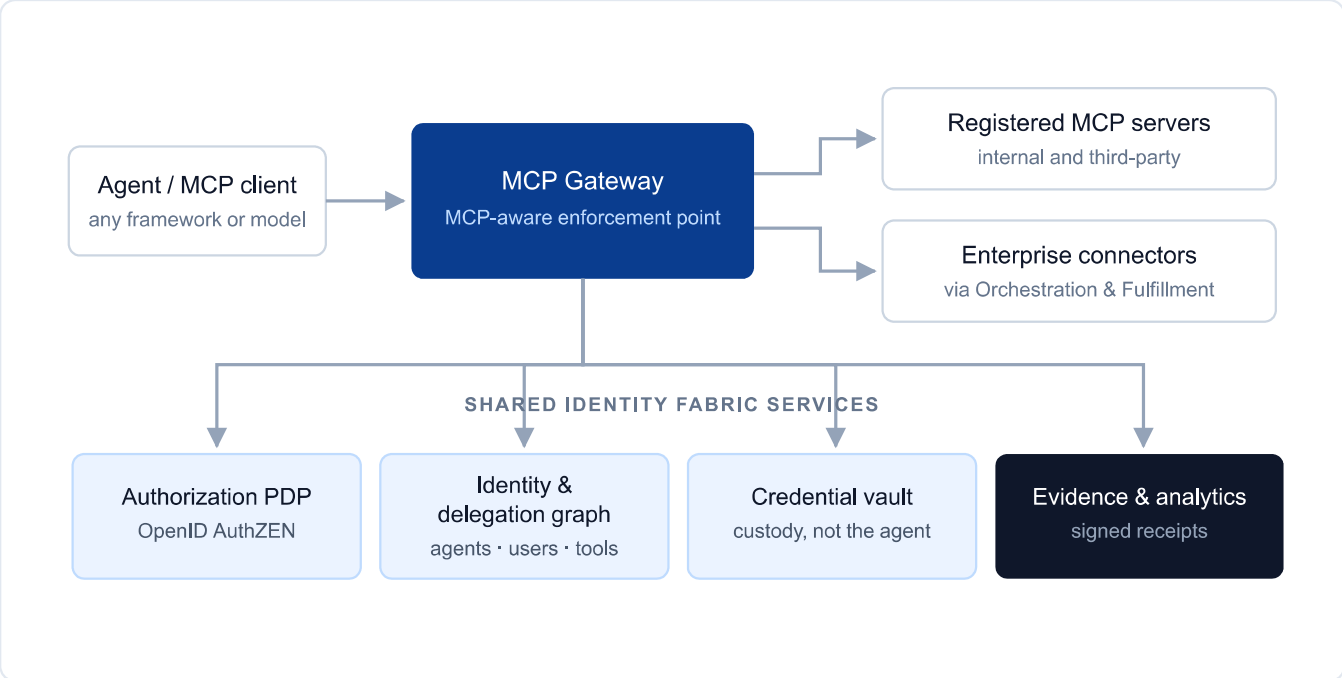

Figure 2 — One service in the Identity Fabric: the gateway enforces; shared services decide, relate, hold custody, and remember.

Dimension	Typical MCP proxy or AI gateway	EmpowerID MCP Gateway
Semantic unit	HTTP request: host, route, token, quota	Model-selected capability: agent, represented identity, tool, schema, parameters
Agent identity	The agent is whatever its token says	Distinct principal with an explicit, revocable delegation — verified at invocation
Discovery	tools/list returns the catalog	Delegation-scoped virtual catalogs — discovery is an authorization surface
Tool integrity	Schema refresh is an ordinary metadata update	Tool definitions are security artifacts: approved schema pins, fail-closed mismatch
Decision authority	Local rules configured in the proxy	The same logical PDP that governs human and workload access, via OpenID AuthZEN
Credentials	Agent holds downstream tokens or keys	Authority and custody separated: vault-backed injection at the outbound boundary
Evidence	Access logs: a token called an endpoint	Signed receipts with explicit semantics: authorized, dispatched, denied, cancelled, observed

05 — CREDENTIALS WITHOUT CUSTODY

The right to request an action is not custody of the key

Tool authorization and downstream authentication are related but different. The gateway may decide an agent is authorized to request a ticket update on behalf of a user — the downstream system still needs an acceptable credential. The dangerous shortcut is putting that credential in the prompt, agent memory, an environment variable, or a tool payload: once the agent holds the secret, it can use it outside the governed path. EmpowerID separates three roles:

1 · Agent

Holds action authority — enough to request a governed tool invocation, nothing more.

2 · Gateway

Holds credential custody through an approved vault and token service — delegated user OAuth, machine-to-machine, token exchange (RFC 8693), sender-constrained tokens (DPoP) where configured.

3 · Downstream

Receives the credential only at the protected outbound boundary. The agent receives the result — never the token.

This also honors MCP's own security guidance, which explicitly rejects token passthrough — forwarding a token issued for one audience to a different downstream API. Tokens stay audience-bound; exchange is validated and feature-controlled.

06 — DURABLE WORKFLOWS

When the action can't complete in one call

Enterprise operations pause — for missing information, business approval, user confirmation, OAuth consent, or an asynchronous connector. The EmpowerID MCP Bridge maps orchestration waiting states into MCP's interaction patterns — Elicitation, and Task lifecycles for clients that support them (Tasks are experimental in the November 2025 specification, so capabilities are negotiated, with simpler behavior for simpler clients):

`elicitation`	non-sensitive input via forms; sensitive interactions via protected URL journeys	`task lifecycle`	working, input-required, completed, failed, cancelled
`idempotency`	retries don't create duplicate enterprise tasks	`cancellation`	recorded before acknowledgment on supported paths

> **Human participation is governed, too.** A confirmation UI is not itself an authorization decision — an approval is bound to the specific action it authorizes, and the binding is part of the evidence.

07 — FOR THE SECURITY ORGANIZATION

Evidence with explicit semantics

An access log shows that a token called an endpoint. It does not preserve the agent and delegating identities, the delegation checked at that moment, the schema the model used, the policy decision and constraints, or the difference between authorized, dispatched, denied, and completed. EmpowerID separates the receipts:

1 · Custody

The boundary received this request, established identity and delegation, obtained the decision, and accepted dispatch — with a parameter hash, not the raw payload.

2 · Execution

What the gateway observed after dispatch: completion or failure, result fingerprint, route, timing — linked to the custody receipt.

3 · Denial & cancellation

Absence of execution is a security outcome. Machine-readable reasons distinguish revoked delegation from policy denial from schema drift.

Honest assurance claims. Receipts are signed and hash-linked so alteration is detectable within the retained chain — that is tamper-evident, not "immutable" or legally non-repudiable. And a receipt proves what the controlled boundary decided and observed; it does not convert a downstream response into stronger proof of business effect than the downstream system provides.

08 — OPERATING AT SCALE

A governed fleet, not a reverse proxy

The first proof of concept has one server and a few tools. The enterprise environment soon has internal servers, vendor-hosted servers, connector-backed tools, multiple agent platforms, and overlapping catalogs. The control plane has to make the secure path easier than shadow gateways and embedded credentials:

`server registry`	approved catalog of servers, owners, endpoints — with OAuth protected-resource discovery (RFC 9728)	`secret hygiene`	vault references in the catalog — never plaintext credentials
`stable operations`	tools bound to authorization operations — policy survives renames	`virtual catalogs`	scoped tool views per team, role, or delegation
`unified routing`	MCP-native and connector-backed tools on one governed surface	`observability`	registration → discovery → decision → receipt, correlated

Distributed enforcement is supported by design: gateway instances can run per environment or region while calling the same logical authorization authority, with consistent decision contracts and evidence semantics. And the boundaries are explicit — the gateway is not a model-alignment system, a prompt-injection defense, an LLM cost gateway, or a substitute for downstream application controls. Each layer keeps its own assurance claim.

Five questions to ask any MCP gateway

1. Is the agent a distinct principal with a bounded, revocable delegation — checked at invocation, or only when a token is issued?
2. Can policy or delegation scope tools/list — or does every agent see the whole catalog before it plans?
3. Does a schema change after approval fail closed — or pass as an ordinary metadata refresh?
4. Do downstream tokens or API keys ever enter agent context — prompts, memory, payloads, results?
5. Can the evidence distinguish authorization, dispatch, denial, cancellation, and observed completion — for both the agent and the identity it represents?

These questions expose the difference between a transport proxy, a developer control plane, an AI gateway, and an identity-fabric enforcement point. The MCP Gateway is an EmpowerID Identity Fabric service and the enforcement component of Agent Governance & Execution — the same authorization, credential, orchestration, and evidence services that govern the rest of the enterprise, applied at the agent-tool boundary.

KEY TAKEAWAYS

The decisive security moment is not when the model generates a tool call — it is when infrastructure accepts it for dispatch.

Delegate deliberately. Discover selectively. Authorize every invocation. Preserve the evidence.

Authority and credential custody are separated — the agent is authorized to request the action, never handed the key.

MCP carries the call. The fabric establishes the authority. The gateway enforces the decision.

See an unsafe tool call stopped before dispatch

Watch a delegated agent discover a scoped catalog, invoke a governed tool, and get denied at the next governed invocation after its delegation is revoked — with the receipt to show why.

Request a demo

info@empowerID.com · empowerid.com

empowerID