IDENTITY FABRIC WHITEPAPER

EmpowerID Governed Authorization

One Decision Authority. Every Relevant Fact.

Version 3.0 · August 2026 Public release EID-WP-GOV-AUTH-2026

Executive summary

Enterprise access is described in many ways. Roles establish broad capabilities. Groups and entitlements carry assignments. Relationships determine scope. Risk, device state, resource sensitivity, and transaction context change what should be allowed now. AI agents introduce delegation, tool discovery, and machine-speed execution.

The problem is not that any of these models is wrong. The problem begins when each becomes a separate source of authorization truth.

One application trusts a token claim. Another checks a group. A gateway applies its own policy. A graph answers a relationship question. An agent platform filters tools. When these controls do not converge on one authority, organizations get inconsistent decisions, opaque access, and revocation they cannot fully explain.

EmpowerID Governed Authorization brings those facts into one logical ABAC decision authority:

- Application owners define protected resources, actions, roles, and assignments in familiar business terms.
- Policy Information Points (PIPs) supply current identity, role, relationship, delegation, resource, risk, and environmental facts.

- EmpowerID's graph-powered **super PIP** resolves connected facts such as nested membership, ownership, management scope, inherited access, and delegated authority.
- One Policy Decision Point (PDP) evaluates the relevant facts under governed policy.
- Policy Enforcement Points (PEPs) ask for and enforce decisions through the OpenID AuthZEN interface.
- The same authority can govern human access and the delegation, discovery, and invocation of AI-agent tools.

The architecture is powerful, but its authoring experience does not have to be complex:

> *Simple things should be simple. Complex decisions should remain possible.*

An application owner should not need to write an attribute policy to say that an Invoice Approver may approve invoices. When the action carries more risk, the same decision authority can add amount limits, organizational scope, separation-of-duties checks, device posture, delegation validity, or transaction context.

The result is one governed foundation that supports familiar access models and precise runtime policy without forcing every use case into the same level of complexity.

> *One decision authority. Every relevant fact. No parallel authorization truth.*

1. Why enterprise authorization fragments

Most organizations already have substantial authorization technology. What they lack is a consistent answer across applications and execution paths.

- **Roles and groups** describe broad capability but rarely capture the complete context of a specific action.
- **Relationships** such as owner of, manager of, member of, responsible for, and delegated administrator over often determine the real boundary of access.
- **Attributes and risk signals** such as resource sensitivity, device posture, location, transaction value, and environmental state change whether an action should be allowed now.
- **Enforcement points** are spread across user interfaces, APIs, gateways, backends, search services, workflows, and agent runtimes.
- **Assignments and grants** can come from several legitimate sources, making effective access difficult to revoke safely or explain completely.

Fragmentation produces three recurring failures.

Inconsistent decisions

The interface hides an action while the API still allows it. A gateway and backend interpret the same user differently. An AI agent reaches a tool through a path that does not apply the controls used for human access.

Unclear effective access

An application owner can see that a person has access but not whether it came from a direct assignment, group, role, persona, relationship, delegation, or another governed source. Security teams can see a policy result without seeing the business reason the authority exists.

Unsafe change

Removing one assignment may destroy access that another legitimate source still supports—or leave access intact without making the remaining reason visible.

Governed Authorization addresses these problems by separating three responsibilities cleanly: applications define what must be protected, trusted sources contribute facts, and one logical policy authority decides.

2. Simple when it should be. Precise when it must be.

A sophisticated authorization engine should not require a sophisticated authoring experience for every application.

Application owners define the business access model

EmpowerID begins with the application's own vocabulary. Through an **App Authorization Contract**, an application can declare:

- the resources, features, and operations that need protection;
- business-readable capabilities such as view, export, approve, administer, remediate, or execute;
- application roles or **access bundles** that group those capabilities;
- the places where decisions must be enforced;
- ownership, risk, lifecycle, review, and assurance requirements.

Application owners can then answer the questions they actually own:

- What can someone do in this application?
- Which role or bundle provides that capability?
- Who has it now?
- How did they receive it?
- What scope, approval, duration, or review requirement applies?

Cross-application **Persona Profiles** can compose those application bundles into recognizable jobs without erasing the application owner's definitions.

Start with assignments; add policy where it creates value

Consider an invoice application.

The application owner might define an **Invoice Approver** bundle containing the `approve invoice` capability. Users receive it through a governed assignment or business persona. That is the simple case.

For a higher-risk transaction, security may add conditions:

- the invoice amount is below the approver's authority limit;
- the approver is responsible for the relevant business unit;
- the transaction creates no separation-of-duties conflict;
- the device and session meet current trust requirements;

- any delegated authority is active and within scope.

The application owner does not need to replace the understandable role with a large policy expression. The role establishes the capability to attempt the operation; the ABAC PDP decides whether this subject may perform it on this resource under the current conditions.

> *Full ABAC power is available where it matters—without turning every application onboarding into a policy-authoring project.*

Transparency is part of simplicity

Ease of configuration is only half the requirement. Application owners, reviewers, and auditors also need visibility into effective access:

- who can perform a protected action;
- which assignment, role, group, relationship, or delegation contributes to it;
- what scope limits the authority;
- which runtime conditions can permit or deny its use;
- who approved it and when it expires or must be reviewed.

The goal is not a black-box policy result. It is a coherent authorization model that remains understandable from business assignment through runtime decision.

3. One ABAC authority, enriched by every relevant fact

EmpowerID does not run separate RBAC, ReBAC, and ABAC engines and then reconcile their permit decisions.

In EmpowerID's architecture:

MODEL OR STANDARD	ITS ROLE
RBAC	Roles, memberships, and role-derived capabilities contribute governed facts.
ReBAC	Relationships, graph paths, inherited scope, ownership, and delegation contribute governed facts.
ABAC	One policy authority evaluates all relevant facts about the subject, action, resource, and context.
OpenID AuthZEN	A standard interface carries authorization requests and decisions between PEPs and the PDP.

The concise formulation is:

> *RBAC and ReBAC provide facts. One ABAC engine decides.*

This does not diminish roles or relationships. It places them in a model where they can be evaluated together with current business and security context.

3.1 PDP, PIP, and PEP

Three standard authorization roles recur throughout the architecture:

COMPONENT	RESPONSIBILITY
Policy Decision Point (PDP)	Evaluates policy against the available facts and returns the authorization decision.
Policy Information Point (PIP)	Supplies facts the PDP needs, such as assignments, relationships, resource data, attributes, delegation, and risk.
Policy Enforcement Point (PEP)	Intercepts a proposed action, asks the PDP for a decision, and enforces the answer.

The governing rule is simple:

> *PIPs contribute trusted facts. The PDP authorizes. PEPs enforce and may narrow or deny. No downstream component may widen the authority granted by the PDP.*

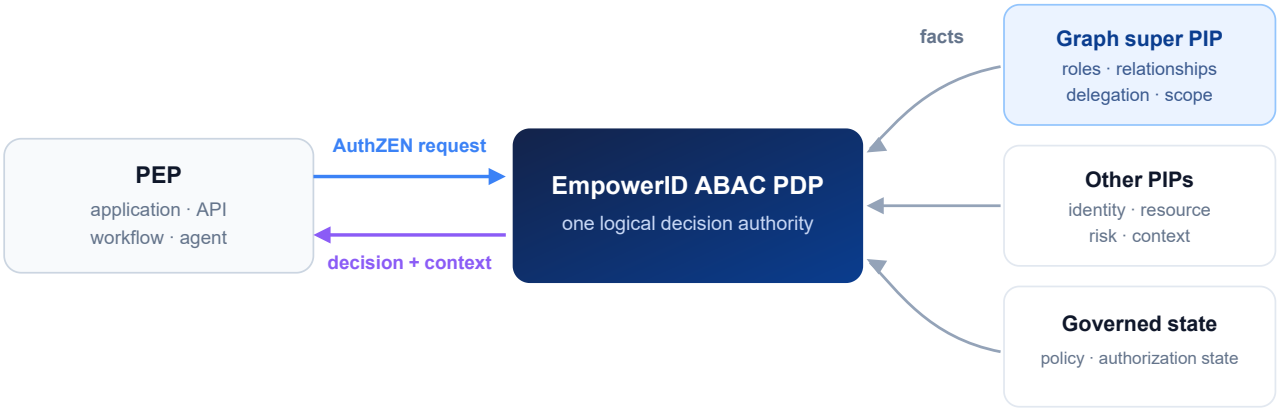

Figure 1 — How a decision flows: facts converge on one deciding authority.

The PEP does not query a PIP and make its own authorization decision. The PDP gathers the facts required by policy and remains accountable for the result.

3.2 The graph as a super PIP

Many PIPs answer narrow questions: a department value, a risk score, a device state, or a resource classification.

EmpowerID's relationship graph can answer connected questions:

- Is this person a direct or inherited member of the required group?
- Does this manager's authority extend to the target employee?
- Is this administrator delegated authority over the target business unit?
- Does this resource belong to an organization the actor may administer?
- Which active relationship path connects this delegator, agent, tool, and target?

We call it a **super PIP** because of the breadth and depth of the facts it can resolve. The term does not mean that the graph is a second PDP. It supplies relationship intelligence to the ABAC decision authority and never issues a competing permit.

> *ReBAC is EmpowerID's graph-powered super PIP—not a second decision engine.*

3.3 AuthZEN is the interface, not the policy engine

The OpenID Foundation's Authorization API 1.0 standardizes communication between Policy Enforcement Points and Policy Decision Points. A PEP can identify the subject, action, resource, and available context; the PDP returns a decision and may return decision context. The standard

deliberately leaves the PDP's internal policy language, architecture, and state management out of scope.

That separation is important:

- **AuthZEN** standardizes how the PEP asks and receives an answer.
- **EmpowerID** supplies the ABAC policy model, graph intelligence, PIP architecture, governed authorization state, and operating lifecycle behind that answer.

EmpowerID has also participated in multiple OpenID AuthZEN interoperability events as both an identity provider and a PDP. The OpenID Foundation named EmpowerID among the implementers in its December 2025 Gartner IAM interoperability sessions, and the program publishes EmpowerID implementation test results.

3.4 One logical authority does not mean one server

"One decision authority" is a statement about governance and decision semantics, not a requirement that every request travel to one physical server.

Serving PDP instances can support different deployment boundaries while remaining part of one governed authorization model. The invariant is that each protected request receives one authoritative decision under consistent policy and fact contracts—not several independent decisions that an application must reconcile.

Exact topology, locality, availability, policy distribution, and data-residency patterns depend on the deployed edition and integration architecture.

4. Govern the full authorization lifecycle

Runtime evaluation is only one part of a trustworthy authorization system. EmpowerID connects business meaning, governed assignment, current facts, enforcement, explanation, and safe change.

STAGE	WHAT HAPPENS
Declare	Applications define protected resources, operations, bundles, and enforcement requirements.
Compose	The business groups application capabilities into roles and cross-application personas.
Bind	Governed processes assign authority with recipients, scope, approvals, duration, and local constraints.
Enrich	PIPs provide current role, relationship, delegation, resource, identity, risk, and environmental facts.
Decide	One ABAC PDP evaluates the request under governed policy.
Enforce	Supported UI, API, backend, search, workflow, and agent PEPs apply the decision and may fail closed.
Explain	The system identifies why access exists or why an action was denied.
Change safely	Governance removes a specific contribution while preserving any other legitimate source of access.

4.1 One authority across enforcement boundaries

An authorization decision is only meaningful if it is enforced where the action can occur.

PEPs may sit in:

- a UI shell, to control action visibility;
- an API gateway or backend-for-frontend, to protect routes;
- a backend service, to enforce object-aware operations;
- a search service, to prevent unauthorized records, totals, or pagination from leaking information;
- a workflow or automation service, before governed work proceeds;
- an agent runtime, before tools are exposed and again before they are invoked.

Hiding a button is useful presentation, but it is not sufficient authorization. The protected action must be governed at the boundary where it can actually occur.

4.2 Evaluation, batch evaluation, and authorized search

The AuthZEN model supports several practical request patterns:

PATTERN	QUESTION	TYPICAL USE
Evaluation	May this subject perform this action on this resource?	API, backend, or object-level action
Batch evaluation	Which of these actions or resources are permitted?	UI capability loading or efficient multi-checks
Search	Which subjects, actions, or resources are permitted in this context?	Authorized discovery and visibility

Implementation patterns must preserve authorization semantics throughout the response. For example, protected search results, counts, and pagination should not disclose records that an object-level decision would deny.

4.3 Contribution-aware access and safe revocation

One effective capability may be justified by several active sources: an application bundle, persona, bootstrap assignment, import, scenario, or governed direct grant.

EmpowerID's **Grant Contribution Registry** records each reason separately and derives the effective grant from the active contributions. Removing one contribution does not destroy access still supported by another. Access disappears when the final valid contribution is removed.

> *Revocation is not simply "delete the grant." It is "remove this reason, preserve every other legitimate reason, and explain what remains."*

This model gives application owners and reviewers a more faithful view of effective access than a single flattened assignment can provide.

4.4 Explain the present; trace the change

Two questions must remain distinct:

1. **Why is this action allowed or denied now?**
2. **Who changed the authorization configuration, under which authority, and with what impact?**

The current authorization state answers the first. Governance and configuration-change evidence answer the second. An audit log alone is not the source of present authority, and a current-state graph is not a complete history of change.

> *Know why access exists now—and preserve how it came to be.*

5. Govern AI-agent authority before planning and before action

AI agents make authorization discipline more urgent. An agent can discover tools, combine them into plans, and invoke them across systems. Giving the agent a copy of a human user's role or token permissions creates authority that is too broad, too static, and difficult to attribute.

EmpowerID treats agent authority as bounded delegation evaluated by the same logical PDP used for other protected actions.

> *Authorization before cognition. Reauthorization before action.*

5.1 Three governed moments

MOMENT	GOVERNED QUESTION
Delegation	May this user delegate this capability to this agent, and within what scope?
Discovery and planning	Which tools may this agent see for the current user, delegation, and context?
Invocation	May this agent use this tool on this resource with these parameters now?

The policy-scoped tool surface is narrowed before the agent plans. When the agent invokes a tool, the PDP evaluates current delegation, relationship, identity, resource, trust, risk, and transaction facts again.

5.2 Authority is an intersection, never an inheritance

Conceptually, effective agent authority is the intersection of:

- what the delegating user is allowed to delegate;
- the agent's registered capability ceiling;
- the active delegation and its scope;
- the live tool catalog;
- current policy and runtime context.

Every stage may narrow the permitted surface. No stage may widen it.

> *The right to perform an action is not automatically the right to delegate it.*

The graph super PIP is valuable here because agent authorization is inherently relational. The PDP may need to know who owns the agent, who delegated authority, which capabilities the delegation contains, what resources fall within scope, and whether the connecting relationships remain valid.

The graph resolves the chain. The ABAC PDP decides whether the chain authorizes this action now. The agent PEP enforces the answer.

6. What this changes for each stakeholder

STAKEHOLDER	PRACTICAL VALUE
CIO or CISO	A coherent authorization authority across supported application, API, workflow, and agent boundaries.
Application owner	Familiar resources, actions, bundles, assignments, and approvers—with visibility into who has access and why.
IAM leader	Governance extends from assignment into action-time decision, explanation, and contribution-aware revocation.
Security architect	One logical ABAC model enriched by pluggable PIPs and exposed through a standards-based PEP–PDP interface.
AI platform owner	Delegation, tool discovery, and invocation can be governed without copying a human's authority to an agent.
Auditor or risk leader	Present-state explanation remains distinct from the history of authorization changes and external effects.

7. Questions to ask any authorization platform

Individual products may specialize in policy evaluation, relationship graphs, governance workflows, or agent mediation. These questions reveal whether the pieces form one coherent authorization system:

1. **Can application owners define protected resources and actions without becoming policy-language experts?**
2. **Can they see who has access, how it was assigned, its scope, and why it remains effective?**
3. **Does the graph contribute relationship facts to one policy authority, or issue a separate decision that must be reconciled?**

4. **Can the platform use roles for simple access while adding ABAC conditions only where risk requires them?**
5. **Do UI, API, backend, search, workflow, and agent enforcement points rely on the same decision authority?**
6. **Can the system remove one reason for access while preserving and explaining every other legitimate contribution?**
7. **For AI agents, are delegation, tool discovery, and invocation governed as distinct moments?**
8. **Does the platform use a standard PEP–PDP interface without pretending that the interface is the policy engine?**

8. Part of the EmpowerID Identity Fabric

Governed Authorization is an Identity Fabric capability and enforcement plane. Adjacent capabilities provide inputs to, or build on, its decisions:

- **Identity Governance** manages lifecycle, access requests, approvals, reviews, roles, entitlements, separation of duties, and the business reasons authority exists.
- **Continuous access** uses trusted security and governance events to refresh facts used by subsequent authorization decisions.
- **Agent identity and federation** establish the verified identity and trust context that agent authorization consumes.
- **Credential Control** governs how credentials are used after an action has been authorized.
- **Governed Execution** governs and corroborates external effects—what actually happened, not only what was permitted.

These capabilities are designed to compose while preserving clear boundaries. Authorization decides whether an action may proceed. It does not by itself establish an external identity, broker a secret, transport a security signal, or prove that a downstream system produced the intended business result.

Conclusion

The value of Governed Authorization does not come from replacing every familiar access model with a new one. It comes from making those models work together under one accountable decision authority.

Application owners define the operations and access structures they understand. Governance establishes who should receive authority and why. PIPs supply current role, relationship, delegation, resource, risk, and environmental facts. One logical ABAC authority evaluates them. PEPs enforce the decision through the OpenID AuthZEN interface and may only narrow the permitted action.

Simple access remains simple. Complex decisions retain the full power of contextual policy. Human and agent actions are governed through the same architectural rule:

> *One decision authority. Every relevant fact. No parallel authorization truth.*
>
> *Simple when it should be. Precise when it must be.*
>
> *Authorization before cognition. Reauthorization before action.*

Learn more

Explore how EmpowerID Identity Fabric connects identity governance, graph intelligence, open authorization standards, agent governance, and governed execution—or contact EmpowerID to review the deployment and enforcement patterns appropriate for your environment.
