

WHITE PAPER

Joiner, Mover, Leaver with Policy-driven Authorization

AXIOMATICS
a LEONARDO company

Every employee goes through a lifecycle within an organization. They join the company, move into new roles or departments as their responsibilities evolve, and eventually leave the organization. In identity and access management (IAM), these stages are commonly referred to as the Joiner, Mover, Leaver (JML) lifecycle.

Each stage requires access to be updated. New employees need the right permissions to become productive on day one. Employees changing roles require access that reflects their new responsibilities while removing permissions they no longer need. When someone leaves the organization, access must be revoked immediately to protect sensitive systems and data.

Many organizations still manage these changes through static roles and manual provisioning processes. As a result, users often receive too much access, retain permissions longer than necessary, or experience delays in receiving the access they need. These challenges increase administrative overhead, complicate compliance, and make it more difficult to maintain least-privilege access.

Policy-driven authorization helps organizations automate authorization decisions throughout the employee lifecycle. Instead of manually assigning, updating, or removing roles whenever a user's circumstances change, access is evaluated in real time using attributes such as department, job title, employment status, location, and other contextual information. As these attributes change, authorization decisions automatically adapt — helping organizations maintain secure, least-privilege access from onboarding through offboarding.

What is Joiner, Mover, Leaver (JML)?

Joiner	A new employee, contractor, or partner who requires access to systems and resources.
Mover	An existing user whose role, department, project, or responsibilities change.
Leaver	A user whose employment or engagement ends and whose access must be removed.



Why the JML Process Breaks Down?

Employee identities are constantly changing. Traditional access management processes often struggle to keep pace, leading to unnecessary permissions, delayed access, and increased security risk throughout the employee lifecycle.

An inefficient JML process affects more than identity management. It can slow productivity, increase security risk, complicate compliance, and create unnecessary operational overhead across the organization.

Slower Onboarding

New employees wait longer for the applications and data they need to become productive.

Permission Creep

Employees accumulate unnecessary access as they move between roles, increasing the risk of overprovisioning.

Higher Administrative Overhead

IAM and security teams spend significant time provisioning, updating, and revoking access manually.

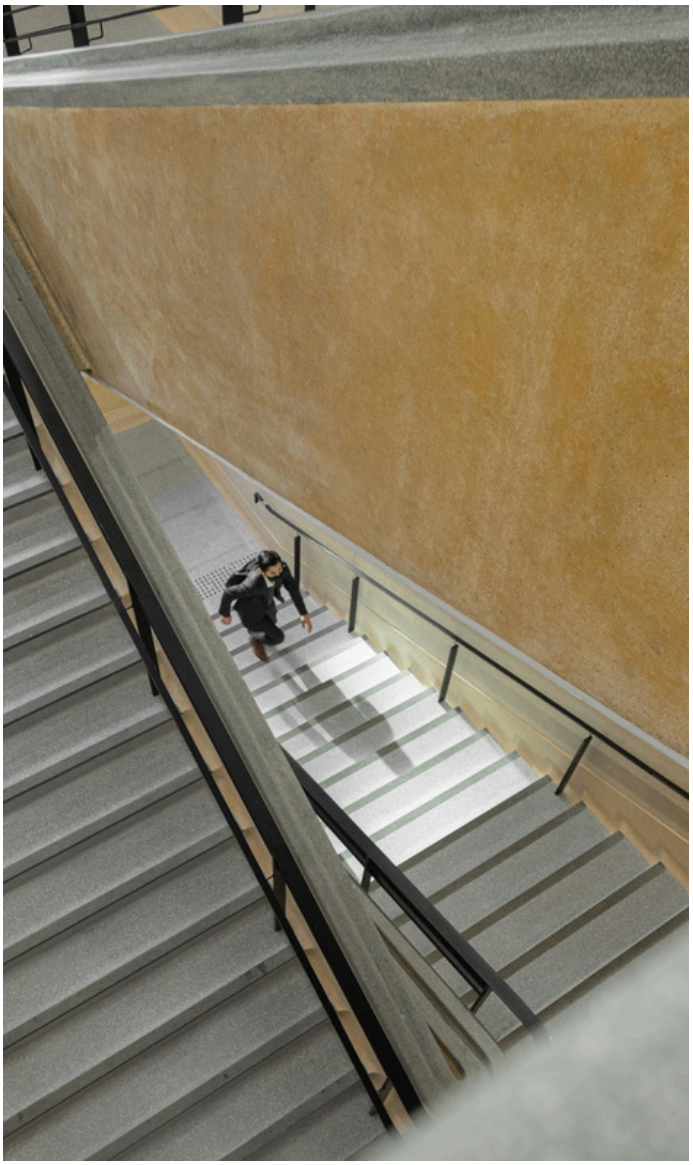
Reduced Business Agility

Organizational changes, new projects, and internal mobility require manual access updates that delay business initiatives.

If your organization experiences any of the following challenges, your JML processes may be relying too heavily on manual provisioning and static access models, making it difficult to maintain secure, least-privilege access.

- New employees wait days for application access.
- Users keep permissions after changing teams.
- Temporary access becomes permanent.
- Managers request frequent access exceptions.
- Former employees retain access after leaving.
- Audits regularly uncover excessive permissions.
- IAM teams spend significant time manually updating roles.

The following examples show how policy-driven authorization helps organizations manage access throughout each stage of the JML lifecycle.



Joiners

When a new employee joins an organization, they need access to applications, data, and the resources required to do their job. Many organizations rely on static roles or predefined access templates to speed up onboarding. In some cases, permissions are copied from employees with similar responsibilities. While this approach can help speed up the onboarding process, it can also result in employees receiving more access than they actually need. As organizations grow, maintaining separate roles for different departments, locations, employment types, and business requirements becomes increasingly difficult, leading to unnecessary administrative effort and increased security risk.

Joiners Example

A new software engineer joins the Engineering department at the company's San Francisco office. Instead of manually assigning multiple application-specific roles, the authorization platform evaluates the employee's attributes — including department, job title, employment status, and location — to determine whether access should be granted.

> "An active user with the title of Software Engineer which is also a member of the engineering department can access engineering department files only when they are in the office."

When the employee requests access, the authorization engine evaluates the request in real time and grants access only if:

- The user belongs to the Engineering department.
- The user's job title is Software Engineer.
- The user's employment status is Active.
- The request originates from an approved office location.

If these conditions are met, access is granted immediately. If an employee's attributes change, authorization decisions automatically adapt without requiring administrators to create, modify, or assign new roles.

How Policy-driven Authorization Helps

Rather than creating separate roles for every onboarding scenario, IT teams define centralized authorization policies that determine who can access specific resources and under what conditions.

For example:

Movers

Employees frequently move between roles, departments, projects, or business units throughout their careers. However, their access permissions don't always evolve at the same pace. In many organizations, employees receive new permissions for their current responsibilities while retaining access from previous roles.

Over time, this leads to permission creep, where users accumulate unnecessary access that is no longer required. Static role models often rely on manual updates to grant, revoke, or modify permissions, making it difficult to maintain least-privilege access as organizational changes occur. This increases administrative effort, complicates audits, and creates unnecessary security risk.

Movers Example

A software engineer transitions from the Engineering department to the Quality Assurance (QA) team.
Rather than manually removing old permissions and assigning new roles, the employee's updated attributes — such as department and job title — are automatically evaluated whenever an access request is made.

How Policy-driven Authorization Helps

Instead of manually assigning roles each time an employee changes responsibilities, organizations define centralized authorization policies based on user attributes.

For example:

> “Active employees with the job title "QA Analyst" who belong to the QA department may access QA resources from approved work locations.”

When the employee requests access, the authorization engine evaluates the request in real time and grants access only if:

- The user belongs to the QA department.
- The user's job title is QA Analyst.
- The user's employment status is Active.
- The request originates from an approved work location, if required by policy.

Because authorization decisions are based on current user attributes rather than previously assigned roles, access to Engineering resources is no longer granted, while access to QA resources is automatically permitted. This helps eliminate permission creep and ensures access remains aligned with an employee's current responsibilities.



Leavers

When an employee, contractor, or partner leaves an organization, all access to applications, data, and systems should be removed immediately. However, many organizations rely on manual deprovisioning processes that can leave accounts active longer than intended or fail to remove access consistently across every system.

Delayed or incomplete de-provisioning increases the risk of unauthorized access, insider threats, and compliance violations. It also creates additional administrative work, making it more difficult to verify that former employees no longer have access to sensitive resources.

Leavers Example

A Quality Assurance (QA) Analyst leaves the organization, and their employment status is updated from Active to Inactive in the identity system.

Rather than manually revoking access across multiple applications, the authorization platform automatically evaluates the employee's updated status whenever an access request is made.

How Policy-driven Authorization Helps

Organizations define authorization policies based on current user attributes, such as employment status, rather than manually removing permissions from every application.

For example:

> “Only active employees may access corporate applications and data.”



When the former employee attempts to access a protected resource, the authorization engine evaluates the request in real time and grants access only if:

→ The user's employment status is Active.

Because the user's employment status is now Inactive, the authorization request is denied automatically. No changes to the policy are required, and there are no lingering permissions to manually revoke. This helps organizations reduce the risk of residual access while simplifying de-provisioning and supporting compliance requirements.

About Axiomatics

Axiomatics – a Leonardo Company is the originator and leading provider of runtime, fine-grained authorization delivered with attribute-based access control (ABAC) for applications, APIs, data, and AI systems. The world’s largest enterprises and government agencies rely on the Axiomatics Authorization Management Platform to centrally define and enforce dynamic access control.

axiomatics.com
info@axiomatics.com
US Office: +1 (312) 374-3443
Europe Office: +46 8 51 510 240

