REVYCORE

WHITE PAPER

Technology Scopes.

Certify the whole, keep the detail.

How to certify a business application or a platform as one named scope, across the systems it spans, while each component stays decided by its own owner.

KNOWLEDGE CENTER

DECEMBER 2026

EXECUTIVE SUMMARY

A business application is not an access.

It is a system of components.

A billing platform, a data platform or a customer portal is made of cloud accounts, roles, databases, secrets, code repositories, identity groups and third-party services, spread across several platforms. Reviewing each piece separately never says whether the system as a whole is governed as intended.

Certifying it as a single blob is no better: when one component changes, everything would have to be recertified, and nobody would know which decision covers which part.

This paper describes technology scope certification: a named scope with an owner, components compared with a reference and decided locally, and a validity of the whole that is derived from the decisions on its parts.

One named scope
With an owner and, when it exists, a reference document.

Local decisions
Each component decided by the person who knows it.

Derived validity
A change to one part degrades the whole, without recertifying all.

One sealed proof
A scope attestation referencing every component decision.

CONTENTS

01 Systems, not lists
02 Defining the scope
03 Two starting points
04 Local decisions, derived validity
05 The owner's choices
06 One proof for the whole
07 Where to start

CHAPTER 01

Systems, not lists

Auditors and boards ask about systems. Reviews answer with lists.

Risk is discussed in terms of systems: the payment platform, the HR application, the data lake. Access reviews, by contrast, are organised by platform and by object, which scatters a single system over many campaigns and many reviewers.

The result is that nobody can answer a simple question: is this system, as a whole, governed as it should be? Each piece may have been reviewed, but the review of the whole never happened.

A technology scope gives the system a name, an owner and a certification of its own, without losing the detail of what it is made of.

> **Technology scope** DEFINITION
> a named grouping of components from several solutions, with an owner, certified as a whole and decided component by component.

CHAPTER 02

Defining the scope

By rules, or by explicit selection, across platforms.

A scope can be defined by rules, such as tags, naming conventions or membership, or by explicitly selecting components from different solutions. Both can be combined.

Typical components include a cloud account, an identity role, a database, a secret, a code repository, an identity group, a storage location, a third-party service, and increasingly an AI agent. Each keeps its own owner; the scope adds an owner for the whole.

Cloud
Accounts, roles, databases and secrets.

Code
Repositories and their automation.

Identity
Groups and applications that grant access.

Services
Third-party services and AI agents the system relies on.

CHAPTER 03

Two starting points

Acceptance says “I answer for this state”, not “this state is compliant”.

With an architecture file, observations are compared with what the reference states. Each component comes out with a finding, and the owner decides on findings, not on a list.

Without a file, a first decision by the owner, component by component, sets the reference state. That acceptance is never confused with automatic compliance: it records responsibility for a state, not a judgement that the state is right.

FINDING	WHAT IT MEANS
Compliant	The observed component matches the reference.
Gap	The component differs from what the reference declares.
Absent	The reference declares a component that is not observed.
Not declared	A component is observed that the reference does not mention.
Not assessable	The component cannot currently be observed well enough to compare.
No reference intent	There is nothing to compare with; an accepted baseline is needed.

CHAPTER 04

Local decisions, derived validity

Decisions stay local. The validity of the whole is derived.

The key distinction is between the component and the grouping. When a component changes, for instance a permission added on a repository, only that entitlement calls for a new decision. The other components keep theirs.

The grouping, meanwhile, becomes degraded: its validity is derived from the decisions on its parts, and it becomes valid again once the new decision is made and verified. No redundant recertification is needed.

When the composition itself changes, a component entering or leaving the scope, the grouping has changed, and its owner must decide.

Derived validity DEFINITION
the validity of a scope computed from the current decisions on its components, recalculated when one of them changes.

CHAPTER 05

The owner's choices

Decide, ask, or delegate, component by component.

The owner of a scope is rarely the expert on every component. The decision model reflects that: for each finding, the owner can act, ask for an opinion, or hand the decision to whoever knows the component.

- **Certify or refuse.** A decision on the finding, with its justification.
- **Accept as baseline.** When there is no reference, the observed state becomes the accepted one.
- **Correct or except.** A gap is corrected, or kept as a time-limited exception with a reason.
- **Ask an expert.** An opinion requested from someone who knows the component.
- **Delegate.** The decision on a component handed to the person best placed to take it.

CHAPTER 06

One proof for the whole

One campaign, one sealed attestation, every decision referenced.

Once every component has been decided, the scope is certified in a single campaign and sealed as a single proof. The scope attestation references each component decision and the state observed at the time of that decision.

For an auditor, this answers the system-level question directly, while keeping the trail down to each component. For the owner, it replaces a scattered set of reviews with one decision they can stand behind.

CHAPTER 07

Where to start

Start with the system that carries the most risk.

Technology scope certification is most useful where a system spans several platforms and carries critical data.

01 **Pick one critical system** and name an owner for it.

02 **Define its components** by rules, by selection, or both.

03 **Compare with a reference,** or accept a first baseline, component by component.

04 **Let changes flow to their owners** and keep the validity of the whole derived.

05 **Certify and seal the scope** as one proof, then widen to the next system.

REVYCORE

Certify the environment, without losing the detail.

Human decisions. Verifiable evidence.

Request a demonstration in your context: your solutions, your scopes, your agents.

revycore.com

This paper describes an approach and the corresponding capabilities of REVYCORE for information purposes; it is not legal advice. Component types available depend on the solutions connected. © 2026 REVYCORE SAS.