

SHARED FOUNDATION · 05/06

ECZ-ID Security, Trust Boundaries and Safe Interpretation

A security and policy guide to what ECZ-ID establishes, what it does not establish, and how to use re-checkable proof without creating unsafe trust assumptions.

PRIMARY READER

CISO, security architect, IAM team, risk/compliance, auditors and developers building policy

SECONDARY READER

Enterprise architects, platform engineers and technical evaluators

WHEN TO READ THIS

You need to decide how ECZ-ID proof should influence security, IAM, policy or risk decisions without over-interpreting identity or evidence.

RELEASE

2026.10
16 October 2026

What you will know after reading

You can separate identity, responsibility, authority, state, bindings and evidence; define safe automation; and fail closed when required proof is absent or stale.

What this document does not establish

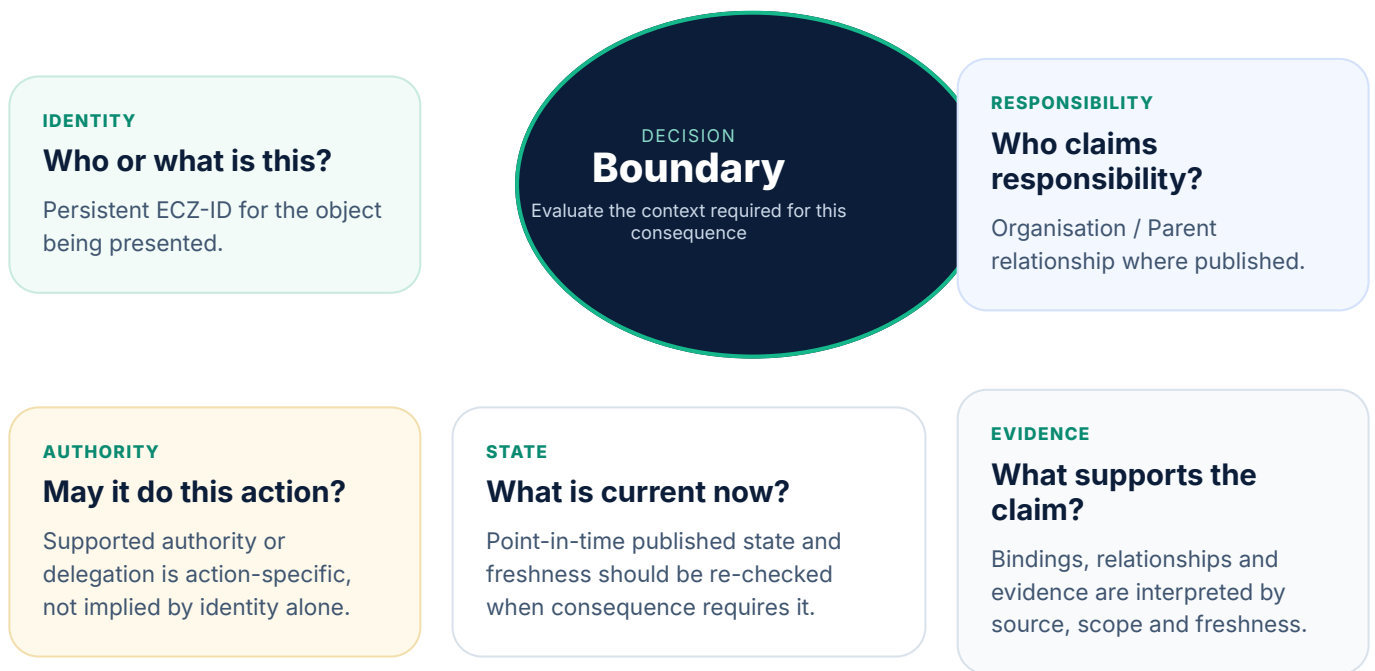
ECZ-ID does not replace authentication, authorisation, runtime controls, vulnerability management, code/device attestation or independent assurance programmes.

Permanent route: <https://developers.ecocitizenz.com/guides/security-trust-boundaries/> · Support: support@ecocitizenz.com

SECURITY MODEL

The trust boundary model

Security decisions become safer when each signal keeps its own meaning instead of being collapsed into one “trust” label.



Security law: presence of a valid identity is necessary for many decisions, but it is not sufficient evidence of authority, safety, behaviour or permission.

WHO / WHO CLAIMS RESPONSIBILITY

Identity and responsibility answer different questions

Identity identifies the persistent object. Responsibility links that object to an organisation/
Parent context where published.

Identity

Which ECZ-ID and object are being presented? Is the expected Passport family/object present?

Responsibility

Which organisation / Parent relationship does the record publish? What responsibility context is represented?

Identity does not prove

Future behaviour, safety, action permission or external compliance.

Responsibility does not prove

Independent verification of every declaration, blanket authority, or that every child record inherits the Parent's strongest state.

AUTHORITY

Authority and delegation are action-specific

Ask what the actor may do for this particular consequence, then combine supported ECZ-ID context with the enforcement controls that actually govern the action.

Identity question

Who or what is acting?

Authority question

What supported authority/delegation exists for this action?

Policy question

What local/business/security policy applies?

Enforcement question

Which IAM/runtime/API/tool/device control will actually enforce the outcome?

Identity is not blanket authority.

STATE & FRESHNESS

Current state is point-in-time context

A security decision that depends on state should use a freshness policy appropriate to the consequence.

Current state

Read the state actually published now. Do not assume an earlier state is still current.

Freshness

Cache only within a policy window that matches the risk. Re-resolve when consequence requires current proof.

Staleness

If evidence/state is too old for the action, treat the verification requirement as unmet until refreshed.

EVIDENCE

Evidence supports bounded claims; absence means not established

Security policy should distinguish “evidence supports this claim” from “evidence exists somewhere in the record”.

When evidence is present

- Inspect source, scope and freshness.
- Use only for the claim it actually supports.
- Challenge the source if the consequence requires stronger assurance.

When evidence is absent

- Treat the required fact as not established.
- Do not infer it from branding, marketing or prior screenshots.
- Do not convert “unknown” into “allow” unless local policy explicitly permits that outcome.

PROVIDER BOUNDARY

Provider identity remains a binding/context signal

Cloud, marketplace, registry and runtime identity is valuable context. It does not become the ECZ-ID canonical authority root.

TrustOps

Acquisition, entitlement where applicable, activation and lifecycle handoff.

Core / Backend

Canonical identity, authority, state, custody and evidence truth.

Resolver

Public, read-only proof for human and machine re-check.

Provider integrations

Provider-native adapters and reference estates. Provider IDs remain bindings/context, not authority roots.

Developer Gateway

Documentation, interoperability, APIs, schemas, SDKs and implementation guidance.

Dashboard

Authenticated customer control plane for permitted estate actions.

One canonical truth layer; separate acquisition, control, public proof and integration surfaces.

Portability rule: if the same legitimate object moves provider/runtime, update supported bindings/context rather than replacing the canonical ECZ-ID solely because the infrastructure changed.

PUBLIC / PRIVATE

Public and private boundaries

The most important operational boundary is between public Resolver proof and authenticated/private customer controls.

Resolver - public

Read-only proof intended for independent re-check. Never publish passwords, API keys, registration tokens, certificates or private account data.

Dashboard - private/ authenticated

Customer control plane for permitted estate actions and private/account context.

Support - controlled

Send only the minimum evidence required to troubleshoot. Remove secrets and unnecessary customer-private information.

Never place secrets in Resolver, public documentation or support tickets.

POLICY ENGINEERING

Safe automation should fail closed on required proof

A verifier should be explicit about which missing or stale conditions block the intended action.

CONDITION	SAFE DEFAULT
Required identity absent / wrong object	Deny or challenge; do not substitute another identity.
Expected binding absent	Treat the required relationship as not established.
Required evidence stale	Re-resolve / refresh before consequence.
Required authority absent	Do not infer permission from identity or responsibility.
Provider ID mismatch	Challenge the binding/relationship; do not silently trust the provider reference.
Resolver unavailable	Apply the local fail-safe/exception policy; do not silently downgrade the requirement.

EXAMPLES

Six fail-closed examples

These examples are generic policy patterns, not universal prescriptions. Apply the risk model appropriate to the system.

Agent spending

Agent identity resolves, but no relevant spending authority context is present → do not infer permission.

API call

API Passport resolves, but the caller still needs valid authentication/authorisation for the operation.

IoT control

Device identity resolves, but command authorisation and safety controls remain separate.

Plugin install

Plugin identity resolves, but host approval, package integrity and install permission remain separate.

SDK use

Package identity resolves, but code safety, signatures, licence and vulnerability controls remain separate.

Workload action

Workload identity resolves, but IAM/runtime policy must still authorise the action.

QUICK REFERENCE

Security FAQ and policy checklist

A strong integration can answer these questions before ECZ-ID proof is allowed to influence consequence.

Policy checklist

- Which ECZ-ID/object must be present?
- Which responsibility relationship is required?
- Which binding(s) are required?
- What state/freshness is acceptable?
- What evidence is required?
- Is action-specific authority/delegation required?
- Which external control enforces the final action?

Interpretation boundaries

- No universal "trusted/safe" score
- No inference from absent data
- No provider ID as canonical authority root
- No secrets in public proof/support
- No identity-only permission grant
- No certification claims without evidence

Next: Implement verification with the Resolver guide and the relevant family Technical Reference.