Aeira by Sparcle

With your LLM. On your audit trail.

Provable authorization.

How Aeira enforces access, and how you verify that it did.

FOR: SECURITY ARCHITECTS · CISO REVIEW · PEN-TEST SCOPING

Most enterprise search retrieves first and filters afterwards. Aeira compiles the caller's permissions into a predicate and pushes it **into the index query**, so a document the caller may not see is never retrieved, never scored and never counted. This brief is written for the person who has to break it.

1. Why post-filtering fails, in two ways

Post-filter (common)

Retrieve the top 100, drop what the user cannot see, return the rest.

Failure A — leakage. Any gap in the filter surfaces a document. Facet counts and result totals leak even when the body does not.

Failure B — silent recall loss. If 94 of 100 are filtered out, the user sees 6 results and no indication that the good answer ranked 140th.

Pre-filter (Aeira)

Compile permissions into an index predicate, then retrieve the top 100 **from what the caller is allowed to see**.

Leakage. A document outside scope is never a candidate, so counts and facets are correct by construction.

Recall. All 100 candidates are eligible, so ranking quality does not degrade as permissions tighten.

2. Eight axes, four sensitivity levels

Access is expressed across eight independent axes, combined with four sensitivity levels (public, restricted, group, user). Collapsing enterprise permissions to “groups” is what forces organisations to over-share or under-share.

group **user** **region** **department** **country** **office** **role band** **employment type**

3. Defence in depth: every engine re-validates the filter

The predicate is produced once by the authorizer and handed to each retrieval engine. **Each engine independently re-checks it against the calling user** before translating it to its native query language: a clause naming a different user, or a group the caller is not in, is rejected.

> **A rejected filter is an error, not an empty result.** Returning zero rows would satisfy “nothing leaked” today while hiding a forged request and leaving the fail-open one refactor away. The contract requires the error to propagate; a caller that swallows it would turn a rejected filter into an unfiltered read.

4. The invariant is executable, and it runs against a live database

The rules above are encoded as a conformance suite that every present and future retrieval engine must pass. It is deliberately not a unit test: the defect it was written for is only visible when two engines are held to the same assertions.

CHECK	WHAT IT ASSERTS
Fail closed	A caller with no grants retrieves nothing above public, and an *empty* permission clause denies everything rather than matching everything.
Impersonation	A filter naming a different principal must error.
Escalation	A group outside the caller's membership must error.
Unknown shape	An unrecognised field or operator must error, never be silently dropped: ignoring a clause widens the filter.
Containment	A group grant confers no user-scoped document that does not name the caller.
Axis isolation	Matching one restricted axis confers nothing restricted on another.

> **Why a live database is required.** During engine evaluation we found a candidate vector backend in which an *empty* permission clause matched every document, where the same clause denies every document in SQL. Both engines' unit tests passed, because both translated the query correctly; “correct” simply means something different per engine. Only an executed query distinguishes them. The suite exists so that class of defect cannot enter through a future backend.

5. What you can verify yourself

CLAIM	HOW YOU CHECK IT, WITHOUT US
Tenant isolation	The tenant identifier comes from the verified token, never the request body, and is the first predicate in every backend query. Client-supplied identity is overwritten server-side.
Refusals are recorded	A cross-tenant attempt returns 403 *and* emits its own audit event. The refusal is evidence, not just a response code.
Audit integrity	Events are appended with a content hash and sealed nightly into a signed Merkle root. A separate binary re-derives that root from exported archives with no access to the live system.
Erasure	Per-tenant envelope encryption. Destroying the key renders that tenant's data unreadable, and the state survives restart.
Supply chain	Statically linked binary, SBOM in CycloneDX and SPDX, Sigstore signatures over the binary and both SBOMs. Verify before you install.

6. How to attack it during a pilot

1. Authenticate as a low-privilege user and attempt to retrieve a document they should not see, by content, by facet count, and by result total.
2. Hand-craft a permission filter naming another principal or a group the caller is not in. Confirm the response is an **error**, not an empty list.
3. Submit a filter with an unrecognised field. Confirm it is rejected rather than ignored.
4. Revoke a group membership and confirm the change takes effect within the documented cache window.
5. Export the audit range covering all of the above and verify the Merkle root offline, on your own hardware.

> **This is the evaluation we want.** Point your security team at the access-control model and try to get a document out that the calling user should not see. That test is the product.

Eight ACL axes and four sensitivity levels are enforced in the query predicate by every shipped retrieval engine. The ACL pre-filter is patent-pending. Figures describing scale or retrieval accuracy are published only alongside the corpus and method that produced them; none are asserted in this document.