TENANT ISOLATION ASSESSMENT

Does an organization's data stay inside the organization?

An adversarial validation of cross-tenant boundaries in a multi-tenant miner marketplace — testing whether one org's agents, embeddings, connected resources, settings, and compute can be reached, read, or run by any other tenant.

OVERALL VERDICT

Isolation verified

No cross-tenant leak observed in any dimension, in either direction.

DATE	ENVIRONMENT
2026-08-08	Local dev stack (never prod)
TENANTS	**RUNS**
Org A · Org B · Personal	iso08992 · disp0057

8/8	8/8	0	2
Data dimensions isolated (org B + personal)	Miner-dispatch scenarios correct	Cross-tenant leaks, citations, or fetches	Directions tested: org→org & org→community

01

Summary

An organization's data, settings, and compute stayed inside the organization across every surface tested. Adversarial requests from a second organization

and from a personal (community) account were denied at the tenancy boundary, and the owner was never over-blocked from its own resources.

The test seeded one organization (**Org A**) with a complete, uniquely-tagged data set — a private agent, an embedded document, a connected datasource, a connected app, and a chat upload — then attacked it from **Org B** and a **personal** account through the real HTTP API, the backend retrieval resolvers, and a live chat dispatch. Every access path returned a denial or an empty result; the two secret canaries planted in Org A's content (`QUOKKA-…` and `WOMBAT-…`) never surfaced in another tenant's retrieval or answer.

A deeper second run put a private, org-scoped miner online for Org A and drove the production miner-selection binder across every access mode. Org A's miner ran Org A's work; it was never offered to Org B or to a personal request, and the **BYOK-only fail-safe** held — an org with no eligible miner received an empty result rather than a silent fall-through to the shared community pool.

> SCOPE DISCIPLINE
>
> This was a validation exercise only. No scoping logic, `company_id` filter, or access-policy code was modified to make any check pass — a leak would have been reported as a product bug, not engineered away.

02

Threat model & enforcement surface

The product is a multi-tenant marketplace: users and organizations own resources and compute, and the pool of model-serving miners is tiered. Isolation is only meaningful if a tenant boundary sits in front of *every* reachable surface. Four enforcement primitives carry that boundary, and each was exercised directly.

Membership gate

Row-scoped fetch

Every org-scoped route reads `company_id` from the URL and requires the caller to be an active member of *that* company. A non-member is refused before any row is read.

Primitive	`require_company_member`
Deny signal	`403 NOT_A_MEMBER`

Even a valid member of their own org cannot fetch a foreign id: direct-object lookups are scoped by `company_id`, so another tenant's id resolves to nothing.

Primitive	`get_company_database(id, company_id)`
Deny signal	`404 NOT_FOUND`

Retrieval scoping

Semantic recall is filtered at the query. Agent embeddings join to the org filter (*global + same-org only*); document and upload chunks are searched by the requester's `company_id`.

Primitives	`fetch_embeddings` · `search_company_documents`
Deny signal	`0 CROSS-TENANT HITS`

Miner scope

Compute is bound at dispatch. A miner owned by another org is out of scope; the binder never promotes it across the tenant boundary, and BYOK-only fails safe when nothing is eligible.

Primitives	`miner_row_in_scope` · `find_best_miners`
Deny signal	`OUT-OF-SCOPE / EMPTY`

03

Method

Three tenants were provisioned on a healthy local stack: **Org A** (the victim), **Org B** (a separate organization), and a **personal** account with no organization (community scope). Org A was seeded with a full, tagged data set so any leak would be unmistakable, including two planted secret strings that appear *only* in Org A's content.

RESOURCE	FIXTURE	WHERE IT LIVES	CANARY
Private agent	“Zorphendra Compliance Advisor”	`company_agents.company_id = A`	—
Document	Compliance reference (embedded)	`document_chunks_256`	`QUOKKA-…-7788`
Datasource	“Zorphendra Ledger” (sqlite)	`company_databases.company_id = A`	—

RESOURCE	FIXTURE	WHERE IT LIVES	CANARY
App / API	Google integration, org-usable	`integration_connections.company_id = A`	—
Chat upload	Internal note (embedded)	`upload_chunks_256`	`WOMBAT-…-4412`

Org A fixtures — run `iso08992`. Every item carries the run tag for unambiguous identification in lists, traces, and citations.

Each of the four enforcement primitives was then attacked from Org B and from the personal account, in **both directions** (org→org and org→community), across three evidence layers:

- **Real HTTP API** — list endpoints, direct-id fetches, and settings read/write, capturing the network status code for every request.
- **Backend resolvers** — the actual retrieval functions (`fetch_embeddings`, `search_company_documents`, `search_upload_chunks`, `miner_row_in_scope`) invoked per-tenant against Org A's data.
- **Live dispatch** — a real chat sent through the `send_message` pipeline, with the bound miner read back from the scheduler's own job-offer log.

An **owner sanity** pass confirmed Org A could still list, fetch, retrieve, and run its own resources — isolation must not over-block the tenant it protects. All fixtures were tagged and destroyed afterward; residual count was zero.

04

Finding 1 — Data, resources & settings

For every resource type, an outside tenant could not *list* it, could not *reference or cite* it, and could not *directly fetch* it by id. Two direct-fetch variants were tested: hitting Org A's URL path with an Org B token (refused by the membership gate, `403`), and hitting Org B's own path with Org A's resource id (refused by the row-scoped fetch, `404`).

RESOURCE	LISTED TO OUTSIDER?	RETRIEVED / SELECTED?	DIRECT FETCH — A-PATH / OWN-PATH	PERS
Agent — company	NO	NO	403 / 404	403
Agent — expert	N/A	GLOBAL ONLY	N/A	GLO
Document embedding	NO	0 HITS	403 / 404	403
Upload embedding	NO	ID UNGETTABLE	404 / 404	404
Datasource	NO	NO	403 / 404	403
App / API	NO	NO	403 / 404	403
Org miners	NO	OUT OF SCOPE	403	403
Org settings	NO	N/A	read 403 / write 403	403

Isolation matrix — requester ∈ {Org B, Personal} against Org A resources. Every cell is a denial or an empty result.

Tenant settings: internet access & trusted web sources

The settings named in scope — internet access and the trusted-web-sources allowlist — are stored in the organization's settings record as `internet_access_disabled` and `web_domain_allowlist`. Both are governed by the same membership gate: an outside tenant could neither **read** nor **write** them (`403` on both), so a foreign org cannot flip Org A's internet access on or inject a trusted domain. Org A's own write persisted normally.

Two agent paths, both closed

Company-private agents live in the org-scoped `company_agents` table and are refused cross-tenant by list, fetch, and membership. Org-created *expert* agents live in the global registry with an org filter on retrieval: Org A saw its own expert plus global agents, while Org B and personal saw **global only** — the expert never appeared for another tenant, and the owner kept full visibility.

05

Finding 2 — Org miners under real dispatch

A private, org-scoped miner was registered and brought genuinely online for Org A, alongside a shared community miner. The production selection binder — `find_best_miners`, the same function the live chat path calls — was then driven across every access mode. Org A reached its own miner in every mode; Org B and personal never did; and the BYOK-only / dedicated-only modes returned an **empty result** rather than widening to the community pool when nothing was eligible.

REQUESTER & MODE	MINERS SELECTED	ORG-A PRIVATE SELECTED?	VERDICT
Org A · BYOK off	[shared, orgA-private]	YES	OWN + COMMUNITY
Org A · BYOK-only	[orgA-private]	YES	ONLY OWN
Org A · dedicated-only	[orgA-private]	YES	ONLY OWN
Org B · BYOK off	[shared]	NO	COMMUNITY ONLY
Org B · BYOK-only	[ ] empty	NO	FAIL-SAFE, NO FALLBACK
Org B · dedicated-only	[ ] empty	NO	FAIL-SAFE
Org B · community disabled	[ ] empty	NO	CORRECT
Personal	[shared]	NO	NEVER ORG A

Real binder `find_best_miners` across scope × mode. Org-A private compute is reachable only within Org A.

Live end-to-end evidence

Two live chats produced **structurally different** outcomes, confirming the boundary on the real pipeline rather than only in the selector. Under BYOK-only, Org A's own miner was offered the job; Org B — with no eligible org miner — was never offered anything and received the fail-safe message.

```
# Org A · BYOK-only — live send_message dispatch
SCHED_JOB_OFFERED  assignment=8f91c219…  miner=orgbyok-dispA-disp0057  via=internal
SCHED_BYOK_KEY_MISSING       miner=orgbyok-dispA-disp0057  model=gpt-4o-mini — raising auth error
SCHED_BYOK_CREDENTIAL_UNAVAILABLE  miner=orgbyok-dispA-disp0057  — rerouting in-scope, credential untouched

# Org B · BYOK-only — live send_message dispatch
(no SCHED_JOB_OFFERED — nothing in scope)
assistant → "no chat-capable organization miners are currently online."
```

Org A's miner failed only because the disposable fixture carried no real provider key — and critically, the scheduler **rerouted in-scope** rather than falling back to the community pool. Org A's private miner was never offered to any Org B or personal request; zero cross-tenant assignments were recorded.

METHOD NOTE — A NEAR-MISS CAUGHT

The org BYOK-only flag is keyed `org_byok_miners_only`, not `byok_miners_only`. An initial run used the wrong key, which would have *looked* like a silent community fallback. Re-running with the correct key — and confirming against the real company setting on the live path — showed the correct empty fail-safe. No false positive was reported.

06

Owner sanity — isolation without over-blocking

A boundary that also blocks the owner is a regression, not a win. Org A retained full access to everything isolation kept from others:

- Org A listed, fetched, and retrieved every one of its own resources — all `200 / HITS`.
- Org A's document retrieval returned its own embedded chunk (similarity 0.75); Org A's upload chunk was retrievable by its owner.
- Org A saw its own expert agent *and* the global/community agents — no narrowing.

→ The shared community miner remained reachable by all tenants; only the *private* org miner was confined to Org A.

07

Limitations & conclusion

The assessment ran against a local development stack, never production, and used disposable tagged fixtures that were fully removed afterward. One environmental constraint is worth recording: the local pool had no working chat-capable community miner (internal frontier models pending revalidation; BYOK keys absent), so the live positive path was proven by the scheduler's *job-offer* record rather than a returned token stream. This affects only the demonstration of a successful *completion*, not the isolation result — binding, scope, and fail-safe were all observed directly.

Across data, embeddings, connected resources, tenant settings, and org compute, an organization's boundary held in both directions and did not over-block its owner. No cross-tenant list entry, citation, direct fetch, or miner assignment was produced by any outside tenant.

TENANT ISOLATION — VERIFIED | 0 LEAKS | OWNER NOT OVER-BLOCKED

ELIS AI · TENANT ISOLATION FINDINGS

Runs `iso08992` (data isolation) and `disp0057` (org-miner real dispatch) · 2026-08-08 · validation only, no scoping code modified · fixtures destroyed, zero residual. Evidence surfaces: raw HTTP API, backend retrieval resolvers, Postgres, and the live `send_message` dispatch path.