CELLULE-PRO DOCUMENTATION

Security Model

Trust boundaries · Ed25519 · JWT EdDSA

cellule.ai · pro.cellule.ai
2026-05-18

Contents

0.1 Threat model — Cellule PRO

This document inventories the threats specific to a Cellule PRO deployment and maps them to the workstreams that address them.

Base assumptions

- The adversary can be **external** (cyberattack, theft) or **internal** (malicious employee, contractor, malicious IT admin)
- All intranet communications are TLS (transport layer not discussed here)
- Workers run on machines that CELLULE does not control (employee endpoints)
- The pool orchestrator and the DB are hosted by the customer (their IT admin manages them)

1. External threats

1.1 Data exfiltration to the internet

Vector: an adversary compromises an employee endpoint and tries to exfiltrate prompts/responses via a malicious worker.

Native mitigation: - Workers have no outbound internet access (intranet firewall) - Worker pool communication over WebSocket on the private network only - No dependency on public cellule.ai (no outbound push to the public domain)

Associated workstream: no specific workstream, this is a deployment constraint (network topology).

1.2 Employee laptop theft

Vector: an employee laptop is stolen, the worker keeps serving requests from their colleagues.

Mitigation: - IT admin revokes the worker token → worker disconnects from the pool on next reconnect - Attack window: ~5-30 min (pool heartbeat)

Limitation: during that window, the attacker can read prompts routed to that worker. Mitigation: **workstream #4 proof of inference** detects statistical anomalies + IT admin can force a "worker rotation" after an incident.

1.3 Man-in-the-middle on the intranet

Vector: an attacker on the intranet intercepts worker pool traffic.

Mitigation: - Intranet TLS (mandatory per the deployment guide) - Ed25519 signature on federation messages (workstreams #1, #3) — impossible to forge a message on behalf of a peer without the private key

2. Internal threats

2.1 Malicious employee

Vector: an employee configures their worker to: - Log their colleagues' prompts - Return cached responses instead of actually inferring - Sabotage responses (malicious injection)

Mitigation: - **Workstream #4 proof of inference**: 3% spot-check + reputation tracking → statistical detection of a worker that cheats >30% of the time - Max-determinism mode: with temp=0 and a fixed seed, a mismatch between two workers = immediate proof - IT admin can blacklist a suspicious worker via the enterprise dashboard

Limitation: a very disciplined employee who only cheats occasionally (<3% of requests) slips through. Future mitigation: **workstream #4** `required` **mode** = 100% redundancy for sensitive tasks.

2.2 Malicious IT admin

Vector: the IT admin has rights on the pool orchestrator and the DB. Can they see everything in plaintext?

Mitigation: - **Zero-knowledge Fernet encryption**: even with access to the postgres DB, the IT admin sees ciphertext. The key is derived from the user token (PBKDF2), which is never stored in plaintext. - The IT admin sees: who (anonymous SHA256 token_hash), when, volume (cipher size) — **not** the content - GDPR policy: the IT admin signs a strict NDA on hiring

Limitation: the IT admin can modify the pool code to inject a keylogger. Detection: container integrity (`celluleai/pool:<signed-digest>` verified at boot via the public release signing workstream).

2.3 External contractor

Vector: an IT consultant has temporary access to the network + their laptop is added as a worker.

Mitigation: - Workers provisioned with an `expires_at` (e.g. 30 days) - After expiration, the worker is rejected by the pool even if it tries to reconnect - Audit log: who added this worker, when, approved by whom

3. Cryptographic / protocol threats

3.1 Replay attack on federation

Vector: an old signed message is replayed to insert a stale fact.

Mitigation: - Timestamp + nonce in the signature (workstream #1) - Anti-replay window 60s (workstream #3 anti-entropy `AUTH_TS_WINDOW_SEC`)

3.2 Inter-peer collusion

Vector: several bonded peers (trust 3) collude to inject poisoning facts.

Mitigation: - Trust level 3 requires a bilateral admin handshake (manual review) - Cross-pool supersede under LWW timestamp: a malicious peer cannot "roll back" - Peer reputation is tracked — too many mismatches = trust downgrade

3.3 Information extraction via embeddings

Vector: a peer accumulates a user's query embeddings and tries to infer the topics.

Mitigation: - **Workstream #3 privacy modes**: `balanced` adds DP noise, `paranoid` never fans out - Rate limiting 60 req/min/signer prevents mass-accumulation attacks

3.4 Timing attack on TEE-less deployment

Vector: measuring inference latencies to infer the content.

Mitigation: `required` mode batches requests across 2+ workers in parallel, masking individual timings.

Limitation: partial mitigation. Full mitigation requires a real TEE (deferred, excludes modest CPU workers).

4. Availability threats

4.1 Single critical worker down

Vector: an employee is sick for 2 weeks → their worker is no longer there.

Mitigation: - N redundant workers (each model tier has multiple workers) - NPU server fallback for tasks that require a rare model - Workstream #8 (future): automatic rebalancing via idle-governance

4.2 NPU server down

Vector: the heavy inference server is down.

Mitigation: - CPU workers keep serving light-to-medium tasks (the majority of the volume) - The SLA declares an acceptable degradation ("7B responses instead of 30B during the incident") - Hot-spare NPU recommended for SLA-strict customers

4.3 Intranet partition between multi-site pools

Vector: VPN outage between Paris and Lyon in a multi-site deployment.

Mitigation: - Each site has its own pool + workers → service continues on each site - Workstream #1 anti-entropy re-synchronizes facts when connectivity returns - Workstream #3 `paranoid` mode works naturally without cross-site fanout

5. Threats → workstreams mapping

Threat	Main workstream	Complementary workstream
Employee laptop theft	#4 (detection)	Manual token revocation (already in public)
Malicious employee — worker cheating	**#4**	Reputation tracking

Threat	Main workstream	Complementary workstream
Malicious IT admin	Fernet encryption (public) + signed container	Audit logs
Extraction via cumulated embeddings	**#3**	Rate limit (already delivered)
Inter-peer collusion	#1 LWW supersede + bilateral trust	Public M11
Single critical worker	Worker redundancy	**#8** (future idle governance)
NPU server down	CPU fallback routing	Contractual SLA
Multi-site WAN down	**#1** anti-entropy	Each site autonomous

6. Threats we do NOT address (yet)

- **Physical side-channels** (hardware keyloggers, radio eavesdropping) — outside software scope
- **Supply-chain attacks on Python dependencies** — partially covered by public release signing, no more than that
- **Model corruption attacks** (prompt injection to leak system prompts) — general LLM topic, not specific to Cellule PRO
- **Censorship resistance** — Cellule PRO runs at the customer's, not a goal
- **Anonymity between employees** — employees are authenticated by SSO, their actions are attributed (deliberate for auditing)

7. Incident response process

1. Detection via monitoring (IT admin dashboard)
2. Isolation: IT admin revokes the suspected workers/tokens
3. Investigation: read the `inference_verifications` + `worker_reputation` logs
4. Remediation: worker reconfiguration, secret rotation, container rebuild if compromised
5. Documented post-mortem, customer communication per SLA

Takeaway

The Cellule PRO threat model assumes that **the customer's intranet is the first security perimeter**, and that **each defense layer is a reduced trust radius**:

1. Network perimeter (customer + customer firewall)
2. Trusted peers (trust 3 federation)
3. Workers (employee machines, variable trust)
4. Pool orchestrator (controlled by IT admin)
5. Database (encrypted per user — zero-knowledge even against the admin)

Workstreams #1-#9 are the software layers that harden this model in depth.