ALPHA Threat Model (STRIDE)

> Scope: the **target** architecture in ARCHITECTURE.md. The current build is a static front end with mock data; its only realistic risks are supply-chain (dependencies, CI) and client-side rendering issues.

Assets

Operator accounts · agent credentials & workload identities · LLM/API keys · agent memory (may contain PII) ·
workflow definitions · audit log · payment mandates · MCP tool access.

Trust Boundaries

1. Browser ↔ Control Plane API
2. Control Plane ↔ Agent Runtime
3. Agent ↔ Agent (A2A bus)
4. Agent ↔ external tools (MCP servers, APIs, databases, LLM providers)
5. Agent ↔ untrusted content (web pages, emails, documents, tool output)

STRIDE Analysis

Threat	Example	Mitigations (target)
Spoofing	Rogue process impersonates `orch-prime` on the bus	Per-agent workload identity; signed envelopes; mTLS with hybrid PQC key exchange
Tampering	Workflow JSON altered to add an exfiltration step	Versioned, signed workflow definitions; CODEOWNERS on policy; diff review before de-ploy
Repudiation	Agent action cannot be attrib-uted	Append-only, signed audit log with trace IDs linking intent → action → result
Information disclosure	Secrets leak into prompts, memory, or logs	Vault-held secrets injected at call time; sensitivity labels; log redaction; Gitleaks in CI
Denial of service	Runaway loop exhausts token budget	Controlled Loop Engineering bounds; per-agent budgets; circuit breakers; rate limits
Elevation of privilege	Agent invokes a tool outside its role	Deny-by-default agent × tool ACL enforced server-side; HIL for irreversible actions

Agent-Specific Threats

Threat	Mitigation
Direct / indirect prompt injection	Treat tool and web output as data; instruction hierarchy; output validation; tool allow-lists
Memory poisoning	Provenance on every memory write; untrusted content never promoted to procedural memory without review
Confused deputy via MCP	Per-tool scopes, read-only defaults, argument validation, user-visible tool call log
Tool supply chain	Pin MCP server versions; verify sources; run with least privilege, sandboxed
Model downgrade / routing abuse	Router policy is signed config; fallbacks limited to approved models
Swarm collusion / cascading failure	Consensus thresholds, independent judges, blast-radius limits per swarm
Harvest-now-decrypt-later	Hybrid X25519 + ML-KEM for long-lived confidential channels

Current-Build Risks & Controls

Risk	Control
Vulnerable npm dependencies	Dependabot, Trivy, dependency review
Committed secrets	Gitleaks on push and PR
XSS through rendered content	React escaping; no `dangerouslySetInnerHTML` for user text
CI token abuse	Workflows use least-privilege `permissions:` and avoid `pull_request_target`
Misleading security claims	SECURITY.md honest-posture table; contributor rule against unverified claims