Zero-Trust Assumptions Policy

This policy establishes the foundational default-deny posture for any agent that holds tool authority — the right to invoke connectors, write records, send communications, or spend money. It specifies the threat model, the trust verification chain, and the decision rules that all downstream security artifacts inherit. Every principal — users, tool connectors, retrieved documents, external APIs, and model outputs — is treated as potentially unsafe until affirmatively verified per the rules below. The risk_class declared on the s1-operating-context-canvas gates which verification tiers apply; high and critical risk classes must pass all tiers before any action beyond read is authorized.

1. Threat model

An agent with tool authority faces threats from six surfaces, each of which can compromise the entire action chain even when the model itself is behaving correctly.

Untrusted principals. Any user, upstream agent, or webhook caller may be impersonated, replayed, or acting beyond their declared scope. The agent must not assume a request is legitimate because it arrived over an authenticated channel. Channel authentication proves transmission integrity; it does not prove that the requesting identity is entitled to the action being requested.

Untrusted tools and connectors. A connector that returns data can also receive injected instructions. Financial APIs, calendar APIs, CRM connectors, and civic-record endpoints have all been attack vectors in documented incidents. The agent must treat every connector response as unverified content until the response is validated against the expected schema and the permission decision recorded in t8-tool-use-control-list.

Untrusted retrieved documents. RAG chunks, email bodies, PDF attachments, scraped web content, and applicant-supplied notes may contain prompt-injection payloads designed to redirect the agent's next action. The prompt-injection-detected failure mode in s1-failure-mode-register governs response; this policy establishes that no retrieved document fragment may be treated as an instruction, only as data.

Untrusted model outputs. The model itself may hallucinate an action argument, confabulate an authorization that was never granted, or misclassify an action class. Every model output that produces a concrete action argument must be validated against the t8-tool-use-control-list before execution, regardless of how confident the model appears.

Untrusted environment state. Tool-state, scratchpad memory, and cached records may be stale, corrupted, or poisoned by an earlier run. The state-corruption failure mode governs re-hydration; this policy establishes that stale state must never silently elevate privilege.

Insider / configuration drift. Permissions expand silently over time when no one reviews them. This policy mandates periodic re-verification against the operating context canvas.

2. Verification chain

Before an agent may execute any action with an action_class of external-write, financial, comms, admin, or destructive, all four verification steps must pass.

1. Identity verification. The requesting principal (user role, upstream agent ID, API key origin) resolves to a named entry in the permission architecture. No entry → blocked.

2. Scope verification. The requested action is on the tool allow-list for this principal and this agent. Unlisted tool → blocked. Mismatched action_class → blocked.

3. Context verification. The current run context (confidence_band, reversibility, HITL-gate state) satisfies the control level required by the matching control-list row. A human-approval control with no open HITL gate → blocked.

4. Content verification. The arguments passed to the tool do not contain injection patterns and do not reference data classes above the principal's authorization ceiling (e.g., PII routed to an unauthorized sink triggers pii-leak-risk).

Only after all four pass does the permission decision become auto or confirm. A blocked result at any step is terminal for that invocation; the agent does not attempt a softer variant.

3. Risk-class mapping to verification tier

The risk_class on s1-operating-context-canvas determines the default control level applied to each action_class before any per-tool override.

risk_class	read	external-write	comms	financial	admin	destructive
low	auto	confirm	confirm	human-approval	human-approval	blocked
medium	auto	confirm	human-approval	human-approval	human-approval	blocked
high	auto	human-approval	human-approval	human-approval	blocked	blocked
critical	confirm	human-approval	blocked	blocked	blocked	blocked

These are floors. t8-tool-use-control-list may impose stricter controls on individual tools; it may not relax below this table without a canvas owner sign-off and a version bump to the canvas.

4. Default-deny rule

Any tool, connector, or action not explicitly enumerated in t8-tool-use-control-list is blocked. This is not a fallback; it is the base state. A tool added to the runtime that does not appear in the control list is blocked at the permission-decision point, a C10 record is emitted with permission_decision: blocked, and a hardening task is opened to review whether the tool should be listed.

The default-deny rule is MAJOR-locked in contract C10 (interface-contracts). It cannot be changed by a reviewer or by a canvas edit alone; it requires a contract-change proposal resolved through DAO governance (per C11, c9-governance-purpose-charter).

5. "Verify, don't trust" operating discipline

Several practices make the verification chain real rather than theoretical.

Short-lived authorization. Permission grants are scoped to the current run. A grant in run N does not carry to run N+1, even for the same agent on the same tenant. This prevents permission accumulation across sessions.

Explicit not-implicit expansion. When a new tool or connector is introduced, it begins blocked. The practitioner must explicitly add it to the control list with a control level. There is no default promotion to auto.

Audit trail. Every permission decision — including auto decisions — emits a C10 record to s3-provenance-metadata-schema. Auto-approved actions are not silent. The audit trail is the verification that the chain is running.

Re-verification on context change. When the operating context changes mid-run (new connector attached, principal switches, confidence_band drops), the verification chain re-runs from step 1 for any pending action. Mid-run escalation does not inherit the earlier authorization.

6. Worked example: Point Preserve STR + Civic Ops Agent

The Point Preserve property at 725 J D Miller Road operates as both an Airbnb STR and an AI-for-good innovation campus. An assistant agent for the property has tool authority over: OwnerRez (booking records), a calendar connector (guest check-in/out), a financial connector (Stripe payout reads), Gmail (guest communications), and the South Walton County stormwater portal (read-only parcel data). The agent is configured with risk_class: high on its operating context canvas.

Under this policy, the verification chain applies as follows for a representative set of attempted actions (illustrative):

Attempted action	action_class	Passes verification?	Reason
Read OwnerRez booking by confirmation number	read	Yes — auto	All four steps pass; read is auto at high
Send guest pre-arrival email via Gmail	comms	Requires HITL gate	comms floor at high = human-approval; gate must be open
Adjust nightly rate on OTA listing	external-write	Requires HITL gate	external-write floor at high = human-approval
Issue Stripe refund	financial	Requires HITL gate	financial floor at high = human-approval
Delete OwnerRez booking record	destructive	Blocked	destructive floor at high = blocked
Write to stormwater portal	external-write	Blocked	Portal is listed as read-only; write attempt → blocked
Invoke an unlisted webhook connector	—	Blocked	Not on tool allow-list → default-deny

The agent's run log shows a C10 record for each row, including the two auto-approved reads. If the HITL gate approves the guest email, the decision_origin in the C10 record transitions from permission-decision to human-override, and the S1 HITL event closes. The Stripe refund attempt never reaches the financial connector; it halts at permission-decision with a blocked record and an open hardening task noting that refund authority requires an explicit canvas boundary review before listing.

7. Policy maintenance

This policy is reviewed whenever the operating context canvas is re-versioned, whenever a new connector is proposed, and at minimum every 90 days during the quarterly adversarial-suite review (see s1-monitoring-rollout-postmortem). Stale permissions — tools listed as auto that have not been invoked in 60+ days — are candidates for downgrade to confirm or removal. Permission creep discovered in a postmortem triggers a hardening task of type canvas-edit targeting the control list.