Refusal Boundary Checklist for Agentic Systems

When systems can execute irreversible actions,
"oops" is no longer a minor bug — it's damage.

Use this checklist to audit whether your system has a *real* refusal boundary before allowing agents, automations, or tools to mutate state.

This is not about alignment.
This is about **inadmissibility**.

---

1. Inadmissibility (Default Posture)

[ ] Have you explicitly classified which commands/actions are **read-only** vs **mutating**?
[ ] Is the default posture **DENY** for mutating actions?
[ ] Are unknown or unclassified commands denied by default?
[ ] Can the system refuse execution even when upstream logic "wants" to proceed?

Failure mode: implicit allow → accidental execution.

---

2. Intent Separation

[ ] Is human intent **explicit**, not inferred from context or prompts?
[ ] Is intent provided via a separate, inspectable channel (not embedded in the command)?
[ ] Can the executor/agent generate its *own* permissions? (If yes → boundary violation)
[ ] Is intent short-lived and revocable?

Failure mode: intent collapses into execution logic.

---

3. Scope & Constraints

[ ] Is execution scope path-limited (root, subtree, namespace)?
[ ] Are permissions time-bounded (expiry enforced)?
[ ] Are action classes constrained (e.g. delete vs move vs overwrite)?
[ ] Are deny-lists enforced even when intent is valid (e.g. `.git/`, secrets, `/`)?
[ ] Is blast radius estimated *before* execution?

Failure mode: valid intent with unlimited reach.

---

4. Deterministic Decisioning

[ ] Given the same inputs (command + intent + policy), is the allow/deny decision deterministic?
[ ] Can the decision be reproduced offline?
[ ] Is there a clear reason attached to every denial?

Failure mode: probabilistic or opaque enforcement.

---

5. Auditability

[ ] Is every execution attempt logged (allow *and* deny)?
[ ] Are execution events logged separately from decision events?
[ ] Does the log include: timestamp, command, scope, decision, result?
[ ] Is the audit trail append-only and tamper-resistant?

Failure mode: no forensic trail after damage.

---

6. Safety Invariants

[ ] Are absolute paths blocked or strictly constrained?
[ ] Are dangerous targets explicitly protected (`/`, `.`, `..`, system dirs)?
[ ] Are recursive or wildcard operations treated as higher risk?
[ ] Is there a hard upper bound on files affected?

Failure mode: "technically allowed" catastrophic actions.

---

7. Human Legibility

[ ] Can a human review the intent and understand *exactly* what is being authorized?
[ ] Can a reviewer explain why an action was allowed or denied in one sentence?
[ ] Would you be comfortable explaining this decision in a post-incident review?

Failure mode: enforcement exists, but no one trusts or understands it.

Interpretation Guide

- **3–5 unchecked boxes** → soft boundary (high risk)
- **6–10 unchecked boxes** → boundary mostly performative
- **>10 unchecked boxes** → no real refusal boundary

This checklist does not prescribe tooling.
It exposes whether **irreversible execution is admissible without explicit human intent**.

Extracted from work on intent-gate.
Created by Brent Williams.