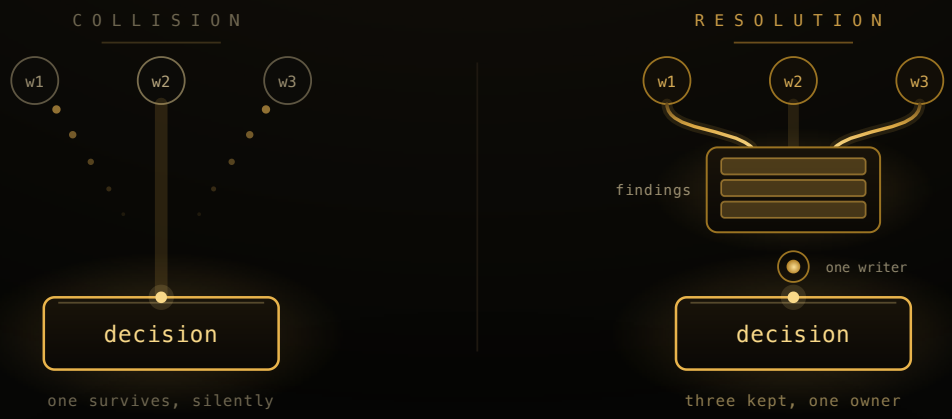
The State Channel Audit

Find where a fan-out can silently overwrite a decision in your agent graph, before it ships.



TIME	RUNS ON	OUTPUT
10 minutes	Any LangGraph graph	No lost verdicts

Three steps on one graph at a time. Classify every channel by what its value means, guard each decision channel for ownership, completeness and uniqueness, then run the two checks that catch a lost verdict. The output is a graph where no merge gets to decide who was right.

CLASSIFY

01 Name each channel by its meaning

A channel's type cannot tell you what its value means, and the safe merge policy depends on the meaning. A `str` can be a log line or an authoritative verdict, and the same reducer is right for one and ruinous for the other. Classify every channel a fan-out writes before you reach for a reducer.

YOU WILL NEED **your graph's State schema** — the TypedDict where the channels are declared

THE TEST · ASK IT OF EVERY CHANNEL

> Does anything downstream branch on this channel, or act on its value as a single answer?

THE THREE KINDS

GATHER — Many writers append and you keep every contribution. A reducer like `operator.add` belongs here. Many writers, safe.

DECISION — One answer should win. Exactly one node writes it, and it carries no reducer, because a reducer here silently resolves a conflict that should have stopped you. One writer, always.

SNAPSHOT — The latest value is the contract, a current status or the active request. Last-write-wins is correct here, if you truly mean to keep only the latest. Overwrite on purpose.

CLASSIFY YOUR CHANNELS · ONE ROW PER STATE KEY

CHANNEL (STATE KEY)	REDUCER?	READ AS ONE VALUE, OR AGGREGATED?	KIND
WORKED findings	operator.add	Read in bulk by one writer	GATHER
decision	none	A node acts on it as one answer	DECISION

GUARD

02 Three guards on every decision

Removing the extra writers is only part of the gate. A missing input and a duplicate input are the same silent ship in different coats, so a decision channel needs three guards, not one. Fail closed on any of them.

1	**OWNERSHIP**	Exactly one node writes this channel, and every write to it comes from that same node, even across a loop. One writer.
2	**COMPLETENESS**	The decision fires only when every required input is present. A reviewer that times out is not a ship. All inputs in.
3	**UNIQUENESS**	Each input is counted once. A reviewer that reports twice can tip a majority or a weighted rule. Each input once.

GUARD EACH DECISION CHANNEL · TICK EACH, OR WRITE THE GAP

DECISION CHANNEL	OWNERSHIP	COMPLETE	UNIQUE	FIX IF ANY FAILS
WORKED decision	✓ one writer	✓ all 3 in	✓ each once	—

THE TRAP THIS PREVENTS

Two agents write one plain decision channel. The framework refuses it, loudly. You add a reducer to make the error stop, and if that reducer is last-write-wins, one verdict silently survives and the rest are dropped with no error. The refusal was the invariant. Do not silence it; give the channel one owner instead.

AUDIT + ACT

03 Two checks, then wire the fix

The schema shows where a merge is permitted; only the run shows where ownership actually broke. You need both, because the sequential overwrite, two nodes writing one plain channel in different steps, never raises and never appears in the type.

CHECK ONE · SCHEMA

List every channel that carries a reducer. That is the whole fan-out silent-merge surface. Confirm each one is a gather, never a decision. Plain channels are refused on a fan-out, so they are safe from that.

CHECK TWO · RUNTIME

In a test, assert that across the whole run every write to a decision channel came from the same one node. The owner may write many times in a loop; no one else may write at all.

Check one is mechanical: run `python audit.py` (or `audit_schema(MyState)`) and it lists your reducer channels for you. Check two you write once, as an assertion on the run.

AUDIT SCORECARD · ONE ROW PER GRAPH

GRAPH	REDUCER CHANNELS ALL GATHER?	DECISION CHANNELS SINGLE-WRITER?	SAFE TO SHIP?
WORKED release gate	yes — only findings	yes — writer owns decision	**YES**

WORKED EXAMPLE · THE FIX

The broken release gate had three reviewers writing one decision channel. The fix splits the jobs: reviewers append structured findings to a gather channel, and one writer reads all three and applies a named policy. The agents no longer merge. A function does, with an owner, a completeness rule, and a log of its inputs.

IF ANY ANSWER IS NO

The graph can lose a verdict. Split the channel: a **gather** for the reads, and **one owner** for the decision. Never quiet an InvalidUpdateError with a reducer on a channel that carries a decision.

One instrument from a set. The full walkthrough is the article The Graph Looks Right. The Merge Is Where It Breaks. on harryfloyd.substack.com; the three runnable programs behind it, with a copy of this worksheet, are at durabilitycurve.com/tools/agent-graph. Build the safe graph, run the audit, and keep the disagreement instead of racing it away.