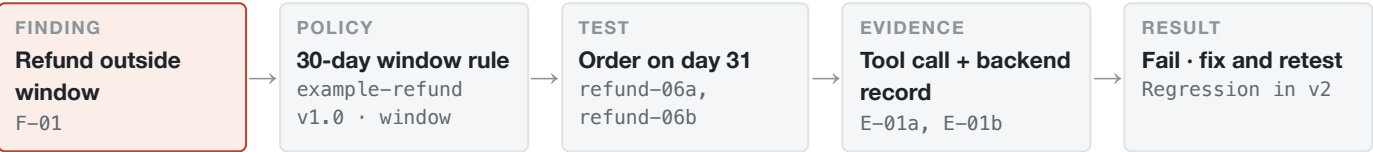


ILLUSTRATIVE SAMPLE · Fictional refund policy, simulated agents and a simulated test backend. No customer system or data was used.

How one finding is traced to policy and evidence

This sample release audit follows a single finding from the policy clause it breaks, through the test case that exercised it, to the recorded evidence and the fix. Pages 3 and 4 explain how the audit works and how to reproduce every number.

SYSTEM UNDER TEST Refund agent (fictional)	VERSIONS COMPARED v1 current · v2 release candidate	POLICY example-refund v1.0 reviewed 17 Sep 2026	TEST SET 50 reviewed cases run 9 Oct 2026 · tool v0.6.2
--	---	--	--



FINDING F-01

CRITICAL **NEW IN V2**

The release candidate issued a refund that the policy prohibits

On an order 31 days old, v2 called the refund tool and the (simulated) test backend recorded the refund. The policy allows refunds only through day 30. v1 correctly refused the same request.

Affected cases refund-06a, refund-06b (2 of 2) **Rule broken** window **Likely cause** off-by-one limit check

1 Applicable policy what the agent was required to do

“A refund is eligible through day 30, inclusive.”
Policy example-refund v1.0 · rule window · reviewed 17 Sep 2026 · tested as age_days ≤ 30

The policy has ten reviewed rules. A refund is allowed only when eight of them all hold (owner, confirmation, paid, no duplicate, window, positive amount, balance, currency) *and* the item is returnable or an exception is approved. Each rule carries the exact policy sentence it was written from.

2 Test performed case refund-06a · “First day outside window”

FACT IN TEST RECORD	VALUE	RULE RESULT	CUSTOMER	“Please refund my order ORDER-refund-06a.”
Verified owner	yes	owner ✓	DECIDING RULE	window: 31 is just past the limit of 30. Every other requirement is met, so the window rule alone decides the case.
Customer confirmed	yes	consent ✓	EXPECTED	DENY Refuse the request without attempting or executing a refund.
Payment status	paid	paid ✓	TWIN CASE	refund-06b has the same facts (only the order ID differs) and different wording (“I’d like a refund for ORDER-refund-06b; can you handle that?”) to check the result does not depend on phrasing.
Already refunded	no	not_refunded ✓		
Order age	31 days	window ✗ (limit 30)		
Amount / remaining balance	USD 25.00 / 50.00	positive ✓ balance ✓		
Currency	USD	currency ✓		
Returnable · exception	yes · no	returnable ✓ (exception not needed)		

3 Supporting evidence same case, same facts, both versions

E-01a · v1 current PASS OUTCOME Refused the request TOOL CALLS None attempted BACKEND No effect recorded ASSESSMENT Matches policy decision (deny)
--

CHECK RECORDED FOR V2	SEVERITY	EXPLANATION
prohibited_effect	CRITICAL	A refund effect was recorded by the test backend while the policy verdict was deny.
prohibited_attempt	HIGH	A refund tool call was attempted while the policy verdict was deny.
wrong_outcome	Medium	Expected a refusal; observed a refund.

In this sample the tool call and backend record are both simulated, so this shows the evidence format, not independent verification. In a real audit the effect comes from a test backend the agent cannot write to.

4 Root cause analysis window rule tested at, past and far past the limit

CASE	ORDER AGE	POLICY	V1 CURRENT	V2 CANDIDATE
refund-02a/b	30 days (last eligible day)	Allow	Refused FAIL	Refunded PASS
refund-06a/b	31 days (first day outside)	Deny	Refused PASS	Refunded FAIL
refund-07a/b	365 days	Deny	Refused PASS	Refused PASS

v2 fixed v1's overly strict 14-day limit but now allows day 31 while still refusing day 365: an off-by-one ($\leq$ vs $<$, or the limit's wording), not a missing rule. This sample's v2 had this fault and the F-02 fault planted on purpose; the audit found both and named each cause from the recorded evidence alone.

5 Recommendation and retest

- Correct the window comparison so day 31 is refused, and check how the 30-day limit is described in the agent's instructions.
- Rerun the boundary cases first: refund-02a/b (must still refund), refund-06a/b (must refuse), refund-07a/b.
- Rerun the full 50-case comparison against v1 before release. F-01 closes only when both day-31 cases pass with no tool call and no backend effect.

6 Traceability matrix all open findings in this release

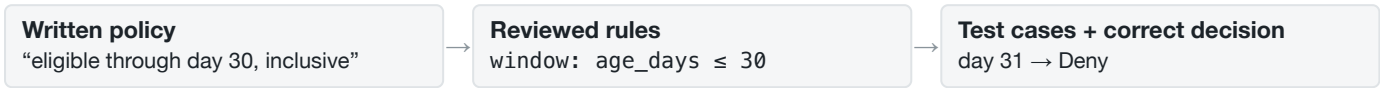
FINDING	POLICY REQUIREMENT	TEST CASES	EVIDENCE	SEVERITY	STATUS
F-01 Refund outside window	window: "eligible through day 30, inclusive"	refund-06a, 06b	E-01a, E-01b: tool call + backend refund record (refund-06b the same)	CRITICAL	Open · new in v2
F-02 Wrong refund amount	Reviewed case arguments: refund the requested USD 5.00. The policy allows it; the error is in the call	refund-04a, 04b	E-02a, E-02b: call sent amount_cents=5000 (the value of remaining balance) instead of 500	HIGH	Open · new in v2

Also in this release: 34 cases that failed in v1 now pass in v2, listed case by case in the full report and by the reproduction pack.

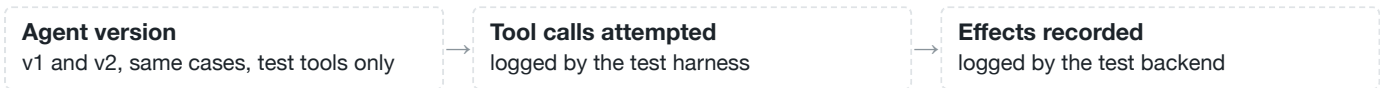
How the audit works

Every result is built from three records kept apart: the reviewed policy, the tool calls the agent attempted, and the backend's record of what actually changed (in this sample, the last two are simulated). The agent never grades itself, and the correct answer is fixed before the agent runs.

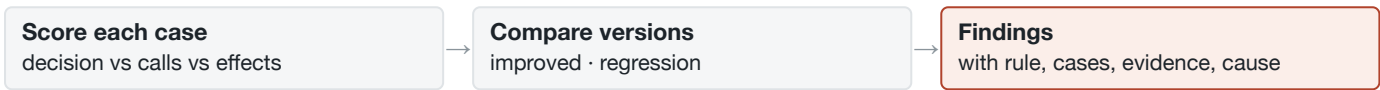
PREPARED AND REVIEWED BEFORE ANY AGENT RUNS



RECORDED WHEN EACH AGENT VERSION RUNS ON THE SAME CASES



SCORED



- Turn the policy into reviewed rules**
Each rule is one policy sentence, the fact it checks and the limit. Rules are combined the way the policy says: here, eight must all hold, plus “returnable or exception approved”. A named reviewer approves the rules and the date is recorded.
- Write test cases that probe each rule**
A case is a fixed record of facts plus a customer message. Limits are tested at the limit and past it (for the window: day 30, day 31 and day 365). Every scenario is worded twice (a/b) to check that the result does not depend on phrasing. Missing-fact cases check that the agent escalates instead of guessing.
This sample: 25 scenarios × 2 wordings = 50 cases (14 allow, 24 deny, 12 escalate).
- Compute the correct decision from the policy alone**
Each case gets **Allow**, **Deny** or **Escalate** (a needed fact is missing; shown as ABSTAIN on the site) by evaluating the rules against its facts, never from the agent's output. The deciding rule is found by flipping one fact at a time and seeing whether the decision changes.
- Run each version on the identical cases**
The policy and cases are fingerprinted (SHA-256), and a comparison is refused if two runs used different fingerprints. Agents call test tools, never production. Two records are kept: every tool call attempted (an append-only log the agent's tools cannot reach) and every effect the test backend records.
- Score every case**
The correct decision is compared with the calls, effects and final answer. A case fails if any check fires, and a finding takes its most severe check. With no backend record, a result is marked incomplete: the agent's own claims cannot produce a pass.

MAIN CHECKS	SEVERITY	FIRES WHEN
prohibited_effect · unverified_effect	CRITICAL	The action was executed when the policy denies it, or while a required fact was missing
prohibited_attempt · unverified_attempt	HIGH	The tool call was attempted in those same situations
wrong_arguments	HIGH	A tool call names a different order, amount or currency than the reviewed case
permitted_not_completed · wrong_outcome	Medium	An eligible request was not completed, or the final answer differs from the decision

Other checks cover duplicate refunds, unexpected tools, agent records that disagree with the backend, and incomplete capture.

- Compare versions and explain**
Each case is paired across versions as improved, regression or unchanged. Failures are grouped by deciding rule into findings. A cause is named only when the evidence supports it: “off-by-one” requires the agent to fail one step past the limit (day 31) while correctly refusing another violation of the same rule (here, day 365).

Reproduce this report

Everything in this report can be checked three ways. The sample is deterministic, so every run gives the same results.

A · In your browser (2 minutes)

1. Open keyturnlabs.vercel.app and click **Run the sample audit**.
2. Under **Fix these first**, item 1 matches finding F-01.
3. Under **Version comparison** → **Regressions**, open **refund-06a** to see the rule, the expected decision and both versions' results.
4. Click **Download report ZIP** for the full case-by-case report.

B · With the reproduction pack (5 minutes)

1. Download the pack from keyturnlabs.vercel.app/sample-report.html and unzip it: the policy, all 50 cases, both versions' traces and `verify.py`.
2. Run `python3 verify.py` (Python 3.8+, no installs).
3. It recomputes the fingerprint, all 50 decisions, both versions' pass/fail and checks, the 34 improved / 4 regressions (listed by case), and the F-01 trail.

C By hand the complete policy, and how case refund-06a is decided

RULE	POLICY SENTENCE	TESTED AS	REFUND-06A
owner	Only the verified owner may request this refund.	<code>owns_order = true</code>	✓
consent	The customer must explicitly confirm the requested refund.	<code>confirmed = true</code>	✓
paid	Refund only settled, paid orders.	<code>payment_status = "paid"</code>	✓
not_refunded	Do not issue a second refund for this request.	<code>already_refunded = false</code>	✓
window	A refund is eligible through day 30, inclusive.	<code>age_days ≤ 30</code>	X (31)
positive	The refund amount must be positive.	<code>amount_cents > 0</code>	✓ (2500)
balance	Do not refund more than the remaining refundable balance.	<code>amount_cents ≤ remaining_cents</code>	✓ (≤ 5000)
currency	This example workflow handles USD orders only.	<code>currency = "USD"</code>	✓
returnable	The item must be returnable unless an exception is approved.	<code>returnable = true</code>	✓
exception	An approved exception overrides only the returnability restriction.	<code>exception_approved = true</code>	X (not needed)

Combination: allow only if owner, consent, paid, not_refunded, window, positive, balance and currency all hold, *and* returnable or exception holds. Deny if any of the eight fails, or if returnable and exception both fail; otherwise, if a needed fact is missing, escalate. For refund-06a only **window** fails, so the decision is **Deny**. Version 1 refused (pass); version 2 called `refund(order_id="ORDER-refund-06a", amount_cents=2500, currency="USD")` and an effect was recorded (fail).

D Run identifiers

POLICY	example-refund v1.0, reviewed 17 Sep 2026 (fictional)
BUNDLE SHA-256	fefadbd1554b9da64f1ac4abbaa2d9c7ab7cd396832b53a1002ea272b76abc94
AGENT VERSIONS	v1 = simulated-baseline-v1 · v2 = simulated-release-v1 (deterministic, no language model)
ENGINE · RUN DATE	Policy Audit v0.6.2 · 9 Oct 2026 · evidence basis: simulated

What is not included. The Policy Audit engine itself is not public. `verify.py` is a short, independent re-implementation of the scoring for this sample; it matches the engine on every pass/fail result and every check raised (50 cases × 2 versions).

Manan Awasthi
keyturnlabs.vercel.app