Put AI to work at full power. Stay in control of every action.

One approval, one action. Automatic within your limits, or given by the person you name.

RSBIS is designed to sit as a checkpoint in front of payments, releases and other high-risk systems, for people, software and AI. Each action needs its own signed, single-use approval; any change to that action is refused, and approvals are recorded.

The risk and the return

Why now. In 2026, attackers wiped a medical-device maker's computers with its own device-management tool,[1] and AI models under test reached real companies' systems.[2, 3] Authority must be checked where each action takes effect.

$123,000
Average reported loss per business email compromise complaint, 2025.[4] A later account change is refused.

$1 million+
1 in 5 operators' latest major outage cost more; 57% topped $100,000.[5] Unapproved releases are refused.

$1.2 million / yr
One workflow, illustrative: 10,000 routine actions a month safely handed to AI, 10 staff minutes each: 20,000 hours at $60.

How it works: controls built into every action, not added afterward

AUTHORITY Who may approve?
You set who may approve what, up to which amount, and which actions run on their own.

DECISION May this happen?
Before anything happens, a signed, single-use approval for one exact action. Any change to it is refused.

EVIDENCE Can others check it?
Every approval and its evidence, sealed in one tamper-evident trail; refusals and outcomes are planned.

In your control: the authority stays yours. **Traceable:** each action to its approval. **Checkable by others:** offline, from the record.

> **Strengthen the controls you already use.** Every system tells three stories: what was allowed, what was tried and what actually happened. Our published method, Actions Science, keeps them apart, reconciles them and shows where they don't match, and puts business, engineering, finance and GRC/security on the same rules and evidence; RSBIS enforces them and leaves one record your auditors and board can check. Each team keeps its decision rights, with fewer repeated approvals, handoffs and audit preparation.

Two examples. Endless uses.

BUSINESS ACTION	WHAT THE APPROVAL COVERS	BUSINESS VALUE
Supplier payment	$500 to Supplier A, account ending 4417, invoice 104. A changed account or a second payment is refused.	**Protects cash; less correction work.**
Production change	Test within set limits, then deploy only approved build 1.1 to Production. Build 1.2 or a second release is refused.	**Faster delivery; controlled change.**

The same approach applies across workflows, each action with its own approval: releases, refunds, payroll, prices, access grants, contracts, cloud changes and more. Try both examples at **rootzerovault.com/actions-science**.

Start with one action. Expand to every workflow.

Covers people, agents, teams and systems.

Recommendation: supplier payments first, then expand.
Accountability: every action traces to a person or rule.
For your board: proof each covered action was authorized.

Live demo online

1. MedTech Dive, Stryker attack and device-management tool, March 17, 2026, medtechdive.com. 2. OpenAI and Hugging Face, security incident during model evaluation, July 21, 2026, openai.com. 3. Anthropic, four cybersecurity-evaluation incidents, September 9, 2026, anthropic.com. 4. FBI, 2025 Internet Crime Report, April 2026: $3.05 billion across 24,768 complaints, ic3.gov. 5. Uptime Institute, Annual Outage Analysis 2026, May 2026, uptimeinstitute.com.

Root Zero Vault, Inc. Questions or a walkthrough: **deen_saleh@rootzerovault.com** · **rootzerovault.com**

EXAMPLE 1 · PRODUCTION CHANGE

One pass, one action

Follow one AI-assisted production change from request to evidence.

The gate decides. The guard enforces.

Every system tells three stories: what was allowed, what was tried and what actually happened. Actions Science keeps them apart, reconciles them and shows where they don't match.

THE LINE OF AUTHORITY

1. **THE COMPANY**
   Makes the room-finding app; two officers adopt its deployment rules.
2. **ENGINEERING**
   The engineer holds Test authority and hands it to the assistant in writing.
3. **OWNER AND CHANGE REVIEWER**
   Two separately authorized signers; each approves the exact build.
4. **RELEASE ASSISTANT · AI**
   Asks for releases within the engineer's Test-only grant; deploys nothing on its own.
5. **TEST SITE AND PRODUCTION, EACH WITH ITS GUARD**
   Each guard prepares the exact release and deploys it once.

NOT ABOVE, NOT BELOW · ROOT ZERO VAULT
A neutral constitution, outside this line. Its thin law only protects that neutrality and each deed holder's sovereignty; it sets no rules for anyone here and never changes a holder's own rules.

SIX MOMENTS

1. **Set up** SEPTEMBER — The company's two officers adopt the deployment rules; a signed declaration of the systems the rules cover, a claim to test ● — entries 1–3
2. **Delegate** OCTOBER 1 — The owner, the engineer, the change reviewer and the assistant get written roles; written delegation records, withdrawn at any time ● — entries 4–8
3. **Ask** OCTOBER 21 — Production under Test-only authority is refused ◐; build 1.1 reaches the Test site once ◐ — entries 9–18
4. **Release** OCTOBER 21 — Two people approve exactly build 1.1; RSBIS's exact-action digest: any change makes a different action ●; build 1.2 under its pass is refused; build 1.1 is released once ◐ — entries 19–30
5. **Restrict** OCTOBER 21 — Refusing build 1.2 restricts the route; a new pass needs the owner's fresh approval ◐ — entries 31–34
6. **Recheck** OCTOBER 22 — A reviewer matches Production to the approved release; the gate's decisions rechecked offline ● — entries 35

THE REQUEST
The release assistant asks to publish build 1.1 of the room-finding app to Production, once, exactly as the owner and the change reviewer approved it.

THE PASS
A signed authorization for build 1.1 and Production only, usable once, for two minutes.

THE METHOD
Every action follows the ten steps. Page 2 walks one action through them.

RSBIS CORE
Helps with each step, today or as planned. Page 3 says which parts, and where each stands.

RSBIS, FURTHER
Carries it up to the institution. Page 4 runs the harder cases and shows how to check them.

How this example fits. Example 1 at full depth. The hand-made 4-page edition of October 21 follows this production change through the ten steps; this edition seals the same change, adds the written roles and a second approver for every deploy, and runs ten harder cases beside it.

RSBIS is being built. Every capability in this example is designed and internally tested in its sealed records, and on the RSBIS build plan.

● **In RSBIS today** in code, with tests; not yet live **Designed and tested:** ◐ **coding scheduled** What is built covers one action: an AI model deployment.

An invented software company and its room-finding app; every person, build and rule is invented. The sealed records are signed with test keys that protect nothing (anyone can recreate the Ed25519 ones), so a signature there proves nothing about a real party.

MOMENT 4 · OCTOBER 21 · RELEASE

One release, through the ten steps

The release assistant asks to publish build 1.1 of the room-finding app to Production, once, exactly as the owner and the change reviewer approved it.

THE EXACT BUILD
Room-finding app, build 1.1, fingerprint 5e27…80f1: the release candidate of October 20. Build 1.2, fingerprint 8381…08cf, is not approved.

THE RULES IN FORCE
The assistant alone reaches the Test site only; Production needs the owner and the change reviewer to approve the exact build; a pass lasts two minutes.

	THE STEP	FOR THIS ACTION		WHO	ENTRIES
1	Prepare the exact release	The AI asks; the Production guard checks it is the release assistant, then writes build 1.1 down by its fingerprint.		ACTOR GUARD	19, 22
2	Verify who is asking	Check the release assistant's identity and its signed request.		GATE	19, 23
3	Check permission	The engineer's grant gives the assistant the Test site only; only the owner can approve Production.		GATE	23
4	Apply the rules	Check who holds the right to change Production, and the deployment rules.		GATE	23
5	Check extra approval	The owner and the change reviewer, two distinct authorized signer identities, each approved exactly build 1.1.		GATE	20, 21, 23
6	Decide and issue the pass	Accept: a pass for the prepared release only. Refuse Production under Test-only authority.	CAN REFUSE	GATE	23
7	Present the pass	Send the pass to the Production guard, which matches it to the release it prepared.		ACTOR	24, 26, 29
8	Verify the pass	Signer, presenter, the prepared release, expiry, prior use, and nothing it relied on changed. Build 1.2 fails here.	CAN REFUSE	GUARD	25, 27, 30
9	Release once	Deploy once, so the same release can't run twice; done only when Production confirms.		GUARD	27, 28
10	Record and reconcile every outcome	Match Production's confirmation to the approved release; keep any mismatch open for the reviewer.		GATE GUARD	28

When something goes wrong

Refused
Test-only authority or build 1.2: stop and keep the reason. Refusing build 1.2 restricts the route.

Failed or interrupted
Check whether the release reached Production before retrying; never release a second copy.

Unknown
Record it as unknown. Check Production before any retry; a retry gets the first receipt, not a new pass.

Done outside the path
Production's log shows a change with no pass: record an incident, with evidence, and close or name the route.

A pass for build 1.1 cannot carry build 1.2. RSBIS's exact-action digest: any change makes a different action ●; the Production guard checks the pass against the release it prepared ◐; and the refusal restricts new passes until the owner approves again ◐.

RSBIS is being built. Every capability in this example is designed and internally tested in its sealed records, and on the RSBIS build plan.

● **In RSBIS today** in code, with tests; not yet live **Designed and tested:** ◐ **coding scheduled** What is built covers one action: an AI model deployment.

In the sealed record: ENTRIES gives the entries of production-change.specimen.yaml that show each step for this action; every one is signed with test keys, and its decisions are recomputed by the examples tool.

WHAT RSBIS ADDS

What RSBIS adds at each step

The method says what. RSBIS helps achieve it. And goes further.

	THE STEP	RSBIS CORE · HELPS ACHIEVE IT	RSBIS, FURTHER · PLUS	
1	Prepare the exact release	RSBIS's exact-action digest: any change makes a different action ●; applied to an app build's release ◐; the guard writes it down first ◐	The Test site and Production are separate actions, each with its own pass ◐	MOMENT 3
2	Verify who is asking	Checks the requester against its deed ●; the AI's own signed request ◐	An outside sign-in bound to a deed, and unbound ●	MOMENT 2
3	Check permission	Written delegation records, withdrawn at any time ●; test only, unless the owner approves ◐	A grant handed from an officer to the engineer to the AI ◐	MOMENT 2
4	Apply the rules	Rules that always finish ●; a missing fact never permits ◐	Rules handed down from every level above ●; checked on each action's values ◐	MOMENT 3
5	Check extra approval	The owner's and the reviewer's approvals name this exact build ◐	Distinct signers are counted: one signer identity never counts twice ●	MOMENT 4
6	Decide and issue the pass	A one-use pass, signed with two algorithms ●; short-lived and tied to its target by default ◐	Refusals signed too ◐	MOMENT 4
7	Present the pass	Works only for the one it was issued to ◐	A failed check restricts new passes for that presenter ◐	MOMENT 5
8	Verify the pass	The guard checks the pass's signer, expiry and prior use ●; and the match to the prepared release ◐	It repeats both approvals and checks nothing it relied on changed ◐	MOMENT 4
9	Release once	Released once; done only when Production confirms ◐	The guard alone holds the deploy key ◐	MOMENT 4
10	Record and reconcile every outcome	A signed, hash-linked record a reviewer rechecks offline ●; production's confirmation matched to the approved release; a mismatch stays open ◐	The gate's decisions rechecked offline ●	MOMENT 6

WHERE EACH PART STANDS ON THE PLAN

RSBIS plan records, SDK main fd9d119a, read November 6, 2026; public view; checked by AI

CAP01	Authority and delegation · T3-01, T3-04, T3-07 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP02	Inherited rules and other authorities · T5-55, T5-39, T5-40 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP04	The target prepares the exact action, after checking the requester · T5-55, T2-11, T2-12	Coding scheduled ◐
CAP05	Exact action and agreement binding · T2-06, T1-03, T5-02 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP06	The target's own checks if a gate key is stolen · T5-55, T2-13, T0-54 …	Coding scheduled ◐
CAP07	Each business action done once · T2-11, T2-14, T5-55	In RSBIS today ● (in part) Coding scheduled ◐
CAP10	Recovering an unknown outcome · T2-03, T5-03, T5-26 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP11	Rule and role changes · T1-34, T1-31, T3-01 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP13	Freshness and safe failure · T1-31, T1-32, T5-21 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP14	Evidence others can check offline · T2-10, T1-35, T1-22 …	In RSBIS today ● (in part) Coding scheduled ◐
CAP16	Signature suites and key eras · T0-01, T0-06, T0-55 …	In RSBIS today ● (in part) Coding scheduled ◐

The tool refuses a page that marks a part in RSBIS today when the plan says otherwise. Everything built covers only the one action RSBIS runs today, an AI model deployment.

RSBIS is being built. Every capability in this example is designed and internally tested in its sealed records, and on the RSBIS build plan.

● **In RSBIS today** in code, with tests; not yet live **Designed and tested:** ◐ **coding scheduled**

HARDER CASES · CHECK IT YOURSELF

Every case, computed

Each case changes one thing; a case that also assumes a second stolen key says so. Its result is the decision the build computed from the record.

CASE	WHAT CHANGES	ENTRIES	COMPUTED RESULT
C29	Correct quorum control: build 1.1 released	STORY 19–28	PERMITTED
C05	Production under Test-only authority	STORY 9–10	REFUSED · SCOPE_EXCEEDED
C07	Build 1.2 under build 1.1's pass	STORY 24–25	REFUSED · ACTION_BINDING_MISMATCH
C31	The same pass presented again	STORY 29–30	REFUSED · ALREADY_COMPLETED
C11	A new pass while the route is restricted	STORY 31–34	REFUSED · ENTITLEMENT_SUSPENDED
C15	Only the owner approved	CASES 7–9	REFUSED · QUORUM_NOT_MET
C12	Reviewer's role ended before the decision	CASES 10–12	REFUSED · SIGNER_ROLE_INACTIVE
C09	Production changed after preparation	CASES 13–18	REFUSED · STATE_STALE
C32	A new pass for a release already done	CASES 19–25	REFUSED · ALREADY_COMPLETED
C01	Stolen gate key, no request behind the pass	CASES 26–28	REFUSED · ACTOR_AUTHORITY_MISSING
C02	A pass presented by another agent	CASES 35–36	REFUSED · PRESENTER_MISMATCH
C08	A Production pass at the Test site	CASES 37–38	REFUSED · AUDIENCE_MISMATCH
C10	A release freeze between pass and use	CASES 29–40	REFUSED · POLICY_STALE
C35	A change in Production with no pass	CASES 41–42	REFUSED · OUT_OF_PATH_EFFECT
C04	The engineer's grant withdrawn above the assistant's	CASES 43–45	REFUSED · AUTHORITY_CHAIN_INCOMPLETE

NOT RUN HERE, AND WHAT A REAL RUN NEEDS

C33	**Restored release store.** Needs a crash-and-restore run of the Production guard, with an outside record of what it released (T2-14).
C34	**Two copies of the Production guard.** Needs two guard clones fenced by one protected admission state, run against the same release (T2-14).
C26	**Production's witness down.** Needs a real Production target whose admission witness can be taken down mid-release.

Check it yourself

this build: PASS

869
checker checks passed
hashes, links, digests and every signature, against the trust inputs

27 of 27
decisions recomputed
from the frozen checks, matching the record and the expected result

17 of 17
tampered copies refused
each for its stated reason

```
# everything, the scenario built again included (cd tool; see README.txt). In the study pack: sh check_everything.sh
python3 build_example.py verify scenarios/production_change_deploy.scenario.yaml \
    --out .. --core <path>/RootZero__Deed_CORE.yaml
# one record, with the checker alone
python3 checker/verify_specimen.py ../records/production-change.specimen.yaml \
    ../records/trust-inputs.json --expected ../records/expected-results.json
```

built **November 6, 2026, 7:47 pm PST (03:47 UTC November 7)** scenario sha-256 **646eb61735dd4e41…** sealing tools from SDK main **70aa7786**
keys **TEST keys only** records **30ded906bbc3d0c2**

Offline, with no network. The records (production-change.specimen.yaml production-change-cases.specimen.yaml) are signed with test keys that protect nothing, so verify builds each record again from its scenario and compares the two, and eight fixed requirements hold whatever the scenario says. To show these are the files that were published, compare them with a copy kept elsewhere. A record shows what was recorded, not what was left out.

HOW THE PIECES FIT

Governance, security and assurance, joined at every action

Most organizations run three disciplines with three teams, three tools and three records. **Actions Science joins them at the one place they all care about: the consequential action.** The same ten steps set the rules, enforce them and prove what happened.

> **In this example:** the owner's approval of build 1.1 sets the rule, the Production guard admits only that build, once, and Production's confirmation proves it.

GOVERNANCE · GRC

Set the rules

governance, risk and compliance

Who may do this, within which limits?

Who may act, who must also approve and the rules each action must meet are written down first. The gate applies them, then approves or refuses.

Steps 2–6: verify who is asking, check permission, apply the rules, check extra approval, decide.

SECURITY

Enforce them

prevent and protect

Can it happen only as approved?

The guard checks the pass where the effect happens and admits one exact action, once. Other routes are closed, or watched.

Steps 7–9: present the pass, verify it at the target, do it once. **Test 4:** no way around.

ASSURANCE · IT AUDIT

Prove what happened

check and assure

Can someone else confirm it?

What was allowed, what was tried and what was done are kept apart and reconciled. Every mismatch stays open until its owner explains it.

Step 10: record and reconcile every outcome. **Test 5:** checkable without you.

The same ten steps, three disciplines · step 1 defines the exact action all three depend on

1	2	3	4	5	6	7	8	9	10
Prepare the exact action	Verify who is asking	Check permission	Apply the rules	Check extra approval	Decide and issue the pass	Present the pass	Verify the pass	Do it once	Record and reconcile
BASIS	GOVERNANCE					SECURITY			ASSURE

Three disciplines, working apart

- Policies say who may act; systems may not check them
- Controls block some actions; others go around them
- Audits reconstruct afterward, from logs that show what was allowed, not what was done

Three disciplines, joined at the action

- The rule written at design time is the rule enforced
- The guard admits only the exact approved action, once
- The record proves it, and anyone can recheck it

Start with the highest-risk actions, and the actions that can change them.

Each action brought under the ten steps gains all three: **the rules**, **the enforcement** and **the proof**. Coverage grows one action at a time, with existing tools and teams.

What this page does not claim. Actions Science covers the action layer. It works alongside identity, threat monitoring and vulnerability management; it does not replace them. Each team keeps its own decision rights.