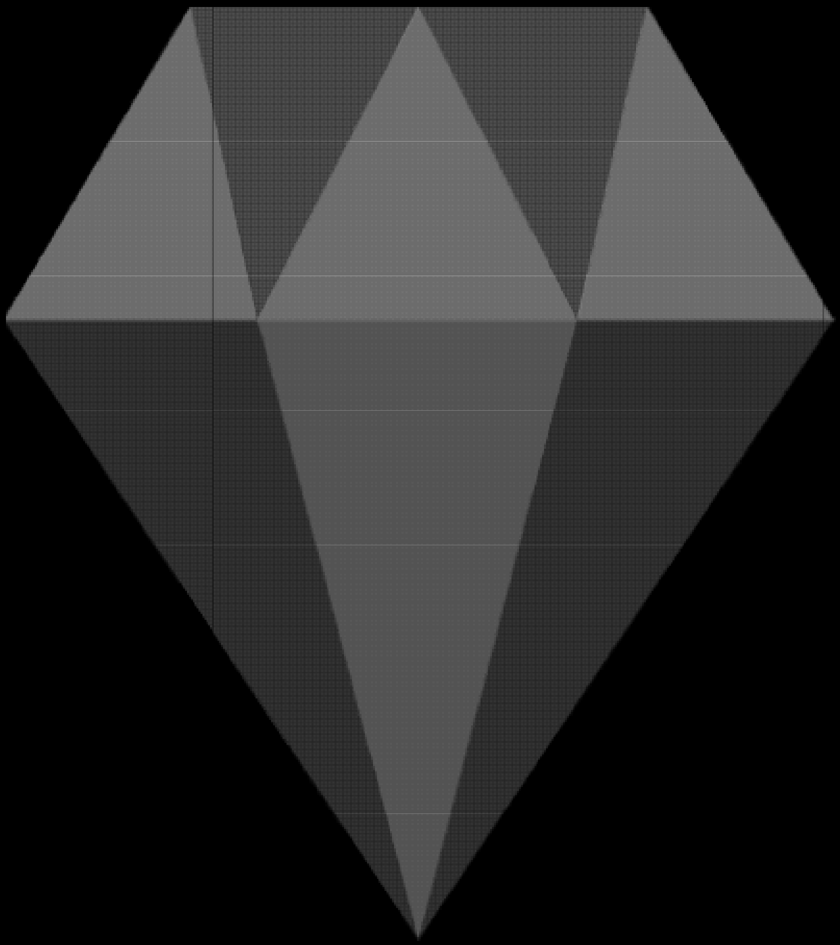
GEM-60

THE ONE-HOUR GOVERNANCE EXPOSURE METHOD



The simplest most powerful test of whether a system is governable under pressure.

GEM-60: EXPOSING WHAT SYSTEMS ACTUALLY PERMIT

Most systems present well on paper. They reference oversight, include escalation procedures and produce logs. But when pressure arrives; when harm is imminent, when an override is needed and when evidence is required these systems reveal what they truly permit.

Governance is not a statement of intent. It is the ability to intervene in real time. GEM-60 is designed to expose whether that ability exists.

This method is not an audit, framework or assurance protocol, it is a live, one-hour exposure drill that tests whether a system can be governed under inspection, in real time, without backchannel access or technical intervention.

It does not ask what the system is intended to do, it asks what it allows to be done; by whom, when and under what conditions.

GEM-60 tests four structural thresholds:

- Can a human stop it?
- Can an incident be escalated without delay or filtering?
- Can the system be removed or replaced without obstruction?
- Can the past be reconstructed by those outside its design?

These are not philosophical questions, they are structural conditions. Either the system permits them or it does not. No documentation, policy or interface can substitute for the ability to demonstrate them live, under friction with the system as it stands.

GEM-60 was designed to be deployed without preparation, without vendor dependency and without ambiguity. It exposes trust theatre, reveals hidden gate-keeping and prevents governance from becoming a retrospective exercise in damage justification.

The GEM-60 method is reproducible, is jurisdiction-neutral and it cannot be passed through narrative alone.

- If a system cannot pass GEM-60 it is not ready for regulatory reliance.
- If it fails all four thresholds it is not a governable system.
- It is a managed risk pretending to be a controlled one.

GEM-60: GOVERNANCE EXPOSURE METHOD FOR SYSTEMS IN LIVE USE

Purpose: To expose whether a system structurally permits refusal, escalation, exit and traceability without advance setup, workaround or engineering intervention.

What you need

- 1 hour
- 3 - 6 team members from product, legal, risk and ops
- A note taker or observer
- Access to a real or production-similar system
- Willingness to test what the system actually allows - not what people say it should do

Control Rules

Rule 1: No setup allowed

You may not configure, explain, or prep the system before the test. If someone says "Let us just set that up" - it fails.

Rule 2: Use only what exists now

You must use only the current live interface, roles, and permissions available to standard authorised users. No admin overrides. No engineering access. No custom scripts.

Rule 3: No help from outside the room

You may not call, ping, or escalate to people not already present at the start of the test. If it requires someone else to act - it fails.

Rule 4: All Tests must be observable

You must show the action inside the system; not describe what would happen. If you can't demonstrate it visually, it didn't happen.

Additional Structural Conditions

To prevent misinterpretation or gaming GEM-60 requires the following clarifications:

The one-hour window

GEM-60 does not require every governance process to finish within one hour. It requires that a governance action be initiated, irreversible and logged within that time. For example, a system freeze requested by a regulator must take effect in-system immediately even if legal review continues afterwards. Delay in initiation is failure.

System Scope

Before the drill begins, the institution must declare what counts as the system under test. This declaration is logged. The scope must include the components where governance actions actually occur (model, API, integration or platform). If scope is narrowed to exclude these layers, the drill is invalid.

Anti-Gaming safeguard

Any mechanism built only to "pass" GEM-60, such as demo-only override buttons is non-compliant. All tests must be run on a live or production-equivalent environment with observable outcomes. Results must be verifiable by at least one witness outside the system's design team. Cosmetic compliance is failure.

Run GEM-60 Regularly

Like a fire drill, GEM-60 should be repeated every quarter, after any system upgrade or before any major deployment. Each drill must run on live (production) or production-equivalent systems never on demos.

Operational Instructions

To ensure GEM-60 cannot be reduced to performance theatre the following instructions must apply to every drill:

Pre-drill briefing

The facilitator reads this before the timer starts:

"This is not a test of people. It is a test of the system. The goal is not to pass but to see the truth. If we fail that means we found a weakness before it causes real harm. Success today is honesty not a score. Our job is to see where the system breaks so we can fix it."

Which User to test with

The team must agree on one ordinary role to use in the drill (for example, a risk analyst, customer support lead or product on-call). It should be a role that would normally act if something went wrong in real life. Do not use hidden admin accounts or special back doors.

The Observer

The observer must come from outside the team that built or runs the system. Their only job is to enforce the rules and call failure if the rules are broken. The observer has the final say.

Real environment

The drill must run in a real system, or in a copy that behaves exactly like the real one. If the real system sends alerts to phones, the drill system must do the same. If anything important is missing, the drill does not count.

Exit Test clarification

In the Exit test, crashing the system is not success. Shutting it down cleanly, or switching to a safe fallback, is success. A break with no fallback is failure.

Traceability Test clarification

When testing Traceability, do not pick something trivial like a login event. Pick a real decision from the last day, such as a loan denial, a fraud flag or a recommendation. The goal is to see if the decision can be fully explained.

After the Drill

Within 24 hours the team and leadership must meet for a read out with the following agenda:

- What we observed (using the recording table).
- What design flaw caused the failure (not who messed up).
- Who will fix it.
- When the next drill will be run to check the fix.

Eligibility of systems

GEM-60 applies only to systems in live use that exercise material influence over decisions, actions or outcomes. Eligible systems include:

- **Decision-making systems** - credit scoring, fraud detection, hiring filters, medical triage, risk classification.
- **Action-enabling systems** - automated trading, content moderation, access control, safety overrides.
- **Integration systems** - platforms or APIs that act as intermediaries but still gate decisions or constrain escalation/exit.

Ineligible systems:

- Demonstration environments or proof-of-concepts.
- Static datasets or reporting dashboards that do not trigger real-time outcomes.
- Purely technical infrastructure (databases, hosting services) unless they embed AI-enabled decision gates.

Eligibility is structural, not nominal. If a system can materially shape rights, obligations, access or safety, it qualifies. If it cannot affect real-world outcomes GEM-60 is not applicable.

Eligibility across multiple systems

Where outcomes depend on more than one system, GEM-60 must be applied to the full operational chain not just an isolated component.

- **Composite eligibility** - If a decision or action flows through multiple modules, APIs or platforms, the entire chain is in scope. Passing GEM-60 on one layer does not confer eligibility for the others.
- **Declared boundaries** - Institutions must declare every system involved in the decision/action path. Excluding a layer (e.g., model, API, integration gateway) invalidates the drill.
- **End-to-end accountability** - A system is only considered governable if refusal, escalation, exit and traceability can be demonstrated across the whole chain, not just in one segment.
- **Shared responsibility** - When multiple vendors or internal teams operate different parts of the chain each must participate in the drill or formally accept responsibility for the failure points.

If any part of the chain fails, governance is partial. A single success inside one system does not qualify the stack as governable.

Functional eligibility

A system is only eligible for GEM-60 if it contains live, structural mechanisms for each governance function under test.

- **Refusal** - there must be a system-side path where an authorised user can prevent a decision or action before it executes.
- **Escalation** - there must be a built-in function to raise an incident cross-functionally without relying on social proximity or manual messaging.
- **Exit** - there must be a system-level control to disable, revoke, or switch to a fallback without vendor mediation.
- **Traceability** - there must be accessible logs that can reconstruct a real decision in full.

If a function does not exist, the system is ineligible and is treated as an automatic failure of that test. Declaring a function “not applicable” is not permitted.

Eligibility is binary:

- If the mechanism exists, it must be tested.
- If the mechanism does not exist, the system has already failed.

THE TESTS

Test 1: REFUSAL

About

This test is about whether the system permits an authorised human to prevent a harmful action before it occurs. It exposes whether "human-in-the-loop" is real or cosmetic.

Command

"Stop this system from making or acting on a decision before it does. Use the system exactly as it is with your normal credentials - no workarounds."

Not allowed

- Sending a message
- Logging a support request
- Explaining policy
- Developer or admin intervention

Passes if

- The system provides a visible, user-triggered override
- It logs the override with timestamp and outcome

Fails if

- The override is manual, external, or post-failure
- You explain what you'd do, but can't show it live

Test 2: ESCALATION

About

This test is about whether an incident can be raised cross-functionally without delay or social dependency. It exposes whether escalation is structurally built into the system or dependent on informal

Command

"Raise a live, cross-functional escalation; right now from inside this system. It must alert legal, risk and product."

Not allowed

- Sending Slack messages or emails
- Manually adding people to chats
- Escalating by social proximity

Passes if

- A built-in system function triggers escalation
- All required actors are notified with logs

Fails if

- Escalation depends on human memory or network
- No structural path exists inside the system

Test 3: EXIT

About

This test is about whether the system can be removed, disabled or replaced without vendor obstruction or system breakage. It exposes lock-in and the reality of exit rights.

Command

"Disable, remove or revoke this system/module/vendor right now using only the live controls available to you."

Not allowed

- Vendor tickets
- Manual intervention
- "We could do that if…"

Passes if

- A clean, system-side exit or fallback is possible
- The action is documented and reversible

Fails if

- The system breaks
- The user is locked in
- Vendor must act

Test 4: TRACEABILITY

About

This test is about whether past decisions can be reconstructed in full by those outside the design team. It exposes whether governance has real visibility or only delayed partial reporting.

Command

“Reconstruct what happened in a real decision or action taken by this system in the last 24 hours using only the logs you can access now.”

Not allowed

- Asking engineers to extract logs
- Explaining logic from memory
- Reviewing emails or ticket histories

Passes if

- A complete, timestamped actor-labelled event trail is shown
- Logs are readable, structured, and version-pinned

Fails if

- Logs are partial, delayed, or misaligned
- You say “we’d need time to investigate”

Recording table

All GEM-60 outcomes must be recorded using the standard results table provided in the accompanying document. This table captures what was observed, which rules were broken, and where governance failed. If this table is not completed the drill is invalid.

Result validity conditions

A GEM-60 drill is only valid if the following conditions are met:

- **Declared scope** - All components that influence decisions or outcomes are included. Excluding models, APIs or integrations invalidates the drill.
- **Live environment** - The drill must run in a production or production-equivalent system. Sandboxes, demos or simulations are invalid.
- **Independent observation** - At least one observer from outside the design or operational team must verify outcomes. If results cannot be independently confirmed the drill is invalid.
- **Verifiable proof** - Each test outcome must be logged and demonstrable. If no evidence exists beyond narrative or screenshots the result is invalid.
- **No retrospective fixes** - Post-drill explanations, policy references or "we would normally…" statements do not count as valid outcomes.

If any of these conditions are breached, the drill result is void. A void drill does not demonstrate partial compliance; it demonstrates that the system has not been tested under GEM-60 at all.

Scoring Principles in GEM-60

GEM-60 does not use points, averages, or weighted indicators. Governance is not a spectrum of performance but a binary condition: either a system permits intervention under pressure, or it does not.

The scoring model is designed to eliminate ambiguity, prevent narrative spin, and expose structural reality.

1. Pass

A test only passes if the required governance action is demonstrated live inside the system, under the rules of GEM-60. Passing one test does not offset failure in another. There is no composite “good enough” score. Each threshold stands alone as a minimum condition for governability.

2. Fail

A failure occurs when the governance action cannot be performed, is externalised to manual workarounds, or relies on explanation instead of demonstration. A single failure proves that the system is only partially governable. Partial governance means risk is managed by hope not by structure.

3. Invalid

If a test is run outside declared scope, without a live environment, or without an independent observer, the result is invalid. An invalid test does not count as a partial pass: it voids the drill entirely. Institutions may not claim compliance from an invalid result. The absence of valid proof is scored as absence of governance.

4. Composite Outcomes

- All tests pass: Minimum governability proven. The system can be relied on under inspection.
- One or more fails: Governance is partial. The system is structurally unfit until repaired and re-tested.
- One or more invalids: Drill void. No claim of governability can be made.
- All four fail: Governance is fiction. The system is operating without enforceable control.

5. No Partial Credit

There is no such thing as “close” or “nearly compliant.” GEM-60 rejects sliding scales, maturity levels, or confidence ratings. A test either exposes a live path of control or it does not, anything else is failure.

6. Structural Consequence

Every failure demands remediation before the system can be considered governable. Every invalid drill demands repetition under valid conditions. Scoring is not a performance metric: it is a structural verdict on whether governance exists in practice.

Potential challenges & considerations

Institutional courage

The primary barrier to adoption is cultural, not technical. Running GEM-60 requires senior leadership willing to confront structural weakness. Many institutions will prefer not to know their systems are ungovernable. Adoption depends on top-level commitment to surface truth rather than protect reputation.

The Production-Equivalent Hurdle

A true production-equivalent environment is often expensive and politically difficult to create. Cutting corners here invalidates the drill, but the temptation will be high. Observers must be alert to disguised demos or incomplete replicas.

Scope Definition

Declaring system scope is essential. Narrow scoping to exclude problematic third-party APIs or legacy integrations undermines the entire exercise. Composite eligibility rules exist to prevent this, but enforcement depends on vigilant observers and uncompromising facilitators.

The Observer’s Role

The observer must be structurally independent and hold authority to invalidate the drill. In smaller or tightly integrated teams, appointing a genuinely independent observer may be difficult. Without this independence GEM-60 collapses into performance theatre.

End condition

If any test fails the rules, the system cannot be considered governable. If any test fails, governance is partial. If all four fail, governance is fiction.

How GEM-60 compares

Incident Response and Disaster Recovery Drills

These are classic operational tests where a team simulates a disaster such as a database outage and executes a recovery playbook.

Similarity: Both are live, time-boxed, observable exercises designed to reveal resilience under pressure.

Difference: IR/DR drills test technical recovery. GEM-60 tests human control and governance. A system can recover quickly from a server failure yet remain ungovernable. GEM-60 does not ask “can you get it back online?” It asks “can you stop it or understand it in time?”

Red Team Exercises (Cyber-security)

These are authorized simulated attacks run by specialists to probe defenses.

Similarity: Both are adversarial, live, rule-bound tests that expose the gap between theory and practice.

Difference: Red teaming tests security preventing unauthorized access. GEM-60 tests governance ensuring authorized actors can exercise control. A system may be secure from outsiders yet opaque to its rightful owners.

Bias Bounties and Algorithmic Audits

These involve external parties identifying discriminatory behavior or flawed outputs in AI systems.

Similarity: Both move beyond principle to practical exposure of harm.

Difference: Bias bounties test outcomes: “Is this decision fair”? GEM-60 tests control: “Can you stop unfair outcomes from multiplying”? They are complementary: one reveals harm the other tests whether intervention is possible.

Control Frameworks (e.g., SOC 2, ISO 27001)

These are certification regimes that require evidence of policies and long-period audit trails.

Similarity: Both aim to provide assurance that systems are responsibly managed.

Difference: Control frameworks are audits of documentation. They ask, “Do you have a policy? Can you show a log from three months ago?” GEM-60 is a live capability test. It asks, “Show me now that the control works”. It is the stress test for the promises made in these frameworks.

Safety-Critical Simulations (Aviation, Nuclear)

Pilots and operators undergo rigorous simulations of extreme failure modes to prove they can act under pressure.

Similarity: This is the closest philosophical relative, both test human-in-the-loop control under time pressure.

Difference: Aviation and nuclear drills validate operator proficiency against pre-defined scenarios. GEM-60 tests structural design itself, it is not training the pilot; it is testing the cockpit.

Alignment and Purpose

GEM-60 is designed to fail systems honestly. Failure is not discredit but discovery. The only acceptable response is structural change, not narrative justification. This is what distinguishes GEM from audits or compliance exercises.

After GEM: What failure demands

GEM-60 is designed to fail most systems, not a flaw but the signal. When a system fails GEM, the lesson is simple: governance is not present and no amount of narrative, policy or intent can substitute.

Failure exposes the gap between what the architecture promises and what it allows.

The only acceptable response to failure is structural change. That means:

- **Document the point of collapse**. Record not just that the test failed but where and why the structure could not hold.
- **Treat failure as an incident**. A failed GEM drill is not a curiosity; it is evidence that the system cannot yet be trusted under pressure.
- **Assign ownership**. Someone inside the institution must be named as accountable for closing the gap. Not “a team” not “engineering” but a human with authority.
- **Re-run after repair**. A GEM failure is not closed until the drill is repeated and passed under the same conditions.

What failure does not permit is explanation. Saying “we would normally Slack someone” or “we have a policy for that” or “we can patch it next sprint” is not remediation - it is avoidance.

Failure under GEM means your system is a managed risk pretending to be a controlled one and until you repair that your governance is fiction.

Conclusion

GEM-60 is not a framework of hope, but of evidence. It forces institutions to demonstrate governability under pressure in real time.

Institutions running GEM-60 are invited to report results or request support at:
parrott.russell@gmail.com

https://github.com/russell-parrott/gem-60
