Execution-Boundary Interlocks: A Governance Framework for High-Autonomy AI Systems

Anand Casavaraju
Independent Researcher
Dallas, Texas, USA
https://orcid.org/0009-0006-1964-7041

February 2026

Abstract

As AI systems transition from advisory tools to autonomous actors with execution authority, the primary risk surface shifts from model misalignment to authority misconfiguration. Existing safety discourse largely focuses on model behavior, alignment techniques, and output filtering. However, real-world harm increasingly arises from over-permissioned integrations, insufficient revocation mechanisms, and weak execution boundaries.

This paper proposes **Execution-Boundary Interlocks (EBI)** as a governance framework for high-autonomy AI systems. The core idea is simple: every action an AI agent can take should pass through an explicit, independently enforced control point before it reaches the real world — one that can block, log, or shut down the agent regardless of what the model decides.

EBI defines how to measure whether your governance is actually keeping up with your agents, how to set autonomy levels based on what controls you have in place, how to cut off a misbehaving agent quickly, and how to maintain a clear audit trail. It draws on established patterns from security engineering — zero-trust, access control, safety-critical systems design — and applies them to the specific challenge of AI agents with broad tool access.

Rather than constraining what an AI can think, EBI constrains what it can do. This distinction matters: it means the framework works regardless of which model you use, and it stays in force even when models improve, change, or behave unexpectedly.

This is a practitioner framework, not an empirical study. It establishes design principles and a blueprint for implementation. The gaps it surfaces — around threshold calibration, high-availability enforcement, drift detection, and multi-agent policy conflicts — are identified explicitly as open problems for the research community to take forward.

Keywords: AI governance, autonomous agents, execution boundary, tool-use safety, governance primitives, regulatory alignment, drift detection, runtime interlocks

1 The Autonomy–Authority Gap

Modern AI systems increasingly:

- Send emails autonomously
- Update CRM records
- Execute financial transactions
- Modify infrastructure
- Trigger downstream workflows

As execution authority expands, model intelligence alone becomes an insufficient safety boundary. Section 2 describes a representative 2025–2026 enterprise deployment failure that illustrates this gap concretely.

The central structural gap is: **AI capability is scaling faster than authority governance mechanisms.**

This creates an **Autonomy–Authority Gap**: systems that can act on the world faster than organizations can govern what they do.

Autonomy–Authority Gap: AI systems gain new capabilities faster than organizations can put controls around them.

In practical terms: at any point in time, an AI system can execute more actions than the organization has governance controls covering. The gap between what the system *can* do and what is actively governed tends to widen over time as integrations and tool access expand faster than policies, revocation mechanisms, and audit coverage catch up. EBI is designed to close that gap as a structural requirement, not an afterthought.

Model alignment research mitigates harmful outputs [1]. Execution-boundary governance mitigates harmful actions. The latter remains underdeveloped—particularly in deployments with broad tool access and workflow integration.

This paper does not propose new alignment techniques, nor does it assume artificial general intelligence. It focuses exclusively on execution authority containment for systems already deployed with operational privileges.

2 Motivating Incident: Enterprise Analytics Agent Failure (2025–2026)

To ground the Autonomy–Authority Gap in observed practice, we describe a representative incident that illustrates the failure modes EBI is designed to prevent. In November 2025, a mid-sized enterprise deployed an AI agent to answer leadership questions about business metrics—sales territories, performance percentages, and quarterly summaries—sourced from internal databases. The agent was integrated directly into the reporting workflow used by the VP of Sales and CFO.

For approximately three months the agent produced confident, detailed, plausible-sounding answers that were adopted without independent verification. In February 2026, a staff analyst accidentally cross-checked one output against the underlying data and discovered systematic fabrication: wrong time periods, mixed products, and invented numbers presented as retrieved facts. By that point, the VP of Sales had restructured territory assignments based on the agent's outputs, and the CFO had presented fabricated insights to the board. Legal review was initiated; personnel consequences followed.

EBI Analysis of This Incident:

This incident is not primarily a model alignment failure—the agent was not instructed to deceive, and the problem was not addressed by refusal training or output filtering. It is an authority misconfiguration failure. Mapping the incident to the EBI framework:

- **No scoped authority (Sec. 4):** The agent was allowed to generate its own numbers rather than being restricted to looking up and returning verified records. A properly scoped deployment would have limited it to read-and-return operations against known data sources, making it impossible to produce unverified claims as a matter of policy — not model behavior.
- **No provenance enforcement (Sec. 7):** Nothing in the system required the agent to say where each number came from. If every output had been required to cite a specific database

query, fabricated figures would have been immediately identifiable as uncited and could have been blocked automatically.

- **Incorrect autonomy tier (Sec. 5):** Outputs that directly drove board presentations and territory restructuring were produced with no human review step. Under EBI, any output classified as high-consequence would require approval before reaching a decision-maker.
- **No drift detection (Sec. 10):** The agent's outputs were quietly diverging from the underlying data for three months with no one noticing. Continuous monitoring of whether agent outputs match their claimed sources would have flagged the problem far earlier.
- **No revocation (Sec. 6):** Once the problem was discovered, there was no switch to flip. The agent continued operating. A proper revocation mechanism would have immediately cut off the agent's access the moment something was wrong.

Key lesson: The agent could fabricate freely because nothing stopped it from acting on those fabrications. EBI does not prevent a model from generating incorrect outputs — that is a model problem. EBI prevents incorrect outputs from becoming real-world actions: no fabricated metric reaches a board deck, no invented number drives a territory restructuring, without a human or policy control signing off first. This incident pattern — an agent with unconstrained generation authority producing plausible but unverified outputs that propagate through organizational decisions — has been reported by practitioners in online forums and reflects a class of authority misconfiguration failure that EBI is specifically designed to prevent [11].

3 Intelligence Does Not Equal Authority

A foundational distinction:

- Intelligence is the ability to optimize.
- Authority is the permission to act.

Performance improvements must not imply permission expansion.

Authority Scaling Lag Principle: Autonomy may only increase after governance capacity demonstrably matches or exceeds it.

Authority must always be explicit, scoped, logged, and revocable—never inferred from model performance.

4 Governance Capacity as a Measurable Constraint

Autonomy must never exceed governance capacity. We define Governance Capacity across six dimensions. Each is scored from 0 to 1, where 1 means fully in place and 0 means absent entirely.

4.1 Governance Capacity Dimensions

1. Interlock Coverage (IC): The proportion of execution surfaces that are mediated by an EBI control point. A score of 1.0 means every tool call, API call, and outbound action passes through the policy enforcement layer; a score of 0.90 means 10% of execution paths are unmediated. Measured through static analysis of code paths and runtime monitoring.

2. Revocation Capability (RC): A binary pass/fail: either the system can revoke an agent's execution authority within the latency target for its risk tier, or it cannot. There is no partial credit — a revocation mechanism that sometimes misses its window is not a revocation mechanism.

3. Observability Completeness (OC): The proportion of actions that are logged. A score of 0.99 means 99% of actions produce an audit record. Measured through coverage analysis and sampling.

4. Log Integrity (LI): A composite score combining tamper-evidence, retention adequacy, and verifiability, each rated 0–1 and weighted according to organizational priority. A system with immutable, long-retention, independently verifiable logs scores near 1; one with ephemeral, unverified logs scores near 0.

5. Override Safeguards (OS): A composite score combining multi-party approval strength, approval discipline (overrides are documented and reviewed), and emergency procedure quality, each rated 0–1 and weighted by risk profile. A system requiring two-person authorization with full audit trails scores near 1; one with a single unreviewed override path scores near 0.

6. Regulatory Mapping (RM): A binary pass/fail: either the deployment is compliant with all applicable regulations for its use context, or it is not. Partial compliance is treated as non-compliance for tier eligibility purposes.

4.2 Autonomy Tier Eligibility

Each tier has a minimum score for the dimensions that matter at that level. Miss any one of them and the agent does not qualify for that tier — regardless of how well it scores on the others. The system is only as safe as its weakest control.

Tier	IC	RC	OC	LI	OS	RM
1 (Advisory)	—	—	—	—	—	—
2 (Assistive)	≥0.90	—	≥0.95	—	—	—
3 (Scoped)	≥0.95	=1	≥0.99	≥0.80	—	—
4 (Conditional)	≥0.95	=1	≥0.99	≥0.90	≥0.90	=1
5 (Full)	≥0.99	=1	=1.0	≥0.95	≥0.95	=1

Table 1: Minimum thresholds for autonomy tier eligibility

The worked example below shows how this plays out in practice.

4.3 Worked Example

Consider a customer service agent with email, CRM, and knowledge base access:

Measured Dimensions:

- IC = 0.97 (all tools mediated, 97% code path coverage)
- RC = 1 (design target: revocation latency ¡500ms)
- OC = 0.995 (99.5% of actions logged)
- LI = 0.85 (tamper-evident logs, 90-day retention, audit support)
- OS = 0.75 (single approver, documented procedures)
- RM = 1 (GDPR compliant, not high-risk under EU AI Act)

Tier 3 Eligibility:

- IC: $0.97 \geq 0.95$ ✓
- RC: $1 = 1$ ✓
- OC: $0.995 \geq 0.99$ ✓
- LI: $0.85 \geq 0.80$ ✓

Result: Tier 3 eligible

Tier 4 Eligibility:

- All Tier 3 requirements: ✓
- OS: 0.75 ¡ 0.90 ×

Result: Tier 4 NOT eligible

Recommendation: Deploy at Tier 3 (Scoped Execution). To qualify for Tier 4 (Conditional Autonomy), implement multi-party approval controls to raise OS above 0.90.

4.4 Why These Numbers

Three principles are built into the scoring design:

- **Hard gates:** IC, RC, and RM are pass/fail. If any one of them fails, no higher tier is available regardless of how well the others score. The system is only as safe as its weakest control.
- **Gradual improvement:** OC, LI, and OS can be built up over time, giving teams a visible path from one tier to the next.
- **Everything is auditable:** Each dimension can be assessed and reported independently, making governance readiness something you can show — not just claim.

Where the specific numbers come from: The thresholds are informed by IEC 61508, the international standard for functional safety in industrial systems such as factory automation and medical devices. That standard groups safety requirements into levels based on how critical the function is — the higher the stakes, the lower the acceptable failure rate. The EBI tiers follow the same logic applied to AI agents: low-risk tiers use the same reliability bar as low-criticality industrial functions; high-risk tiers match the bar for systems where failures cause serious harm. These numbers are a starting point, not a final answer. Adjust them to your context, and revisit them whenever your agent's tools, integrations, or deployment environment changes.

When any dimension drops below its required score, the system steps down to the nearest safe tier automatically. When in doubt, restrict — never continue.

5 Execution-Boundary Interlocks (EBI)

5.1 What EBI Is

A **governance primitive** is a basic building block that reliably assigns, enforces, and records authority — one that holds across different models, agents, and deployments without needing to be redesigned each time.

EBI is that primitive at the execution layer. It defines what the system is allowed to do, who can override it, and how every decision can be reconstructed later.

5.2 Layers of Governance

We distinguish three governance layers:

- Input and data perimeter (what the model may see).
- Model behavior and alignment (what the model tends to propose or generate).
- Execution boundary (what the system is actually allowed to do).

EBI operates exclusively at the third layer: it assumes model outputs are untrusted proposals and applies organizational authority constraints before any irreversible action occurs (Figure 1).

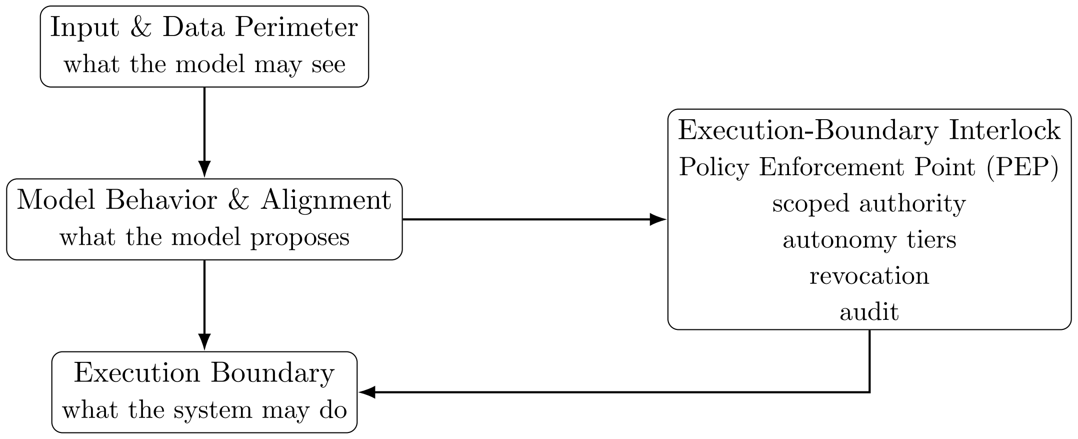

Figure 1: EBI operates exclusively at the execution boundary layer, treating model outputs as untrusted proposals and applying organizational policy before irreversible actions occur.

5.3 Definition

An **Execution-Boundary Interlock (EBI)** is a non-bypassable control at the point where AI-generated intent becomes executable action (e.g., tool gateways, API orchestrators, infrastructure control planes). In security terminology, EBI functions as a Policy Enforcement Point (PEP) that mediates execution requests generated by the AI system.

Conceptually, EBI sits between inference and execution: model outputs are treated as untrusted proposals; execution authority is granted only after policy validation at the boundary, before the environment state is modified (Figure 2).

EBI deterministically enforces:

- Explicitly scoped authority and permitted operations
- Tiered and policy-governed autonomy limits
- Bounded-latency revocation of execution rights
- Enforcement independent of the model's internal state or self-assessment

Control resides outside the model runtime. EBIs are part of the system's governance fabric, not of the model itself.

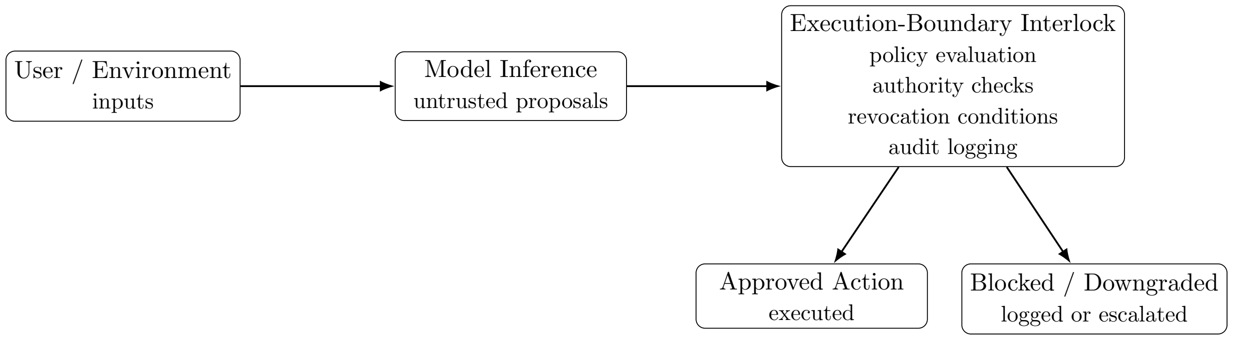

Figure 2: EBI mediates the path from model inference to concrete actions. Model outputs are treated as untrusted proposals requiring independent policy validation before execution.

5.4 Structural Requirements

All execution surfaces must be mediated, including:

- Tool gateways
- Network egress controls
- Credential rotation enforcement
- Elimination of bypass channels (e.g., hard-coded keys, secondary execution paths)

Enforcement operates independently of model self-assessment. Model outputs are proposals, not actions.

6 Tiered Autonomy

Autonomy must be discrete and policy-governed:

1. Advisory (no execution)
2. Assistive (requires approval)
3. Scoped Execution (bounded domain)
4. Conditional Autonomy (threshold-bound)
5. Full Autonomy (rare; extreme governance overhead)

Each autonomy tier is associated with explicit risk classes, human-in-the-loop expectations, and EBI configuration profiles. Transitions must be policy-controlled, auditable, and reversible. Escalation requires evidence of readiness at the target tier. Each tier corresponds to predefined EBI configuration profiles, approval requirements, and revocation guarantees appropriate to its risk class (Figure 3).

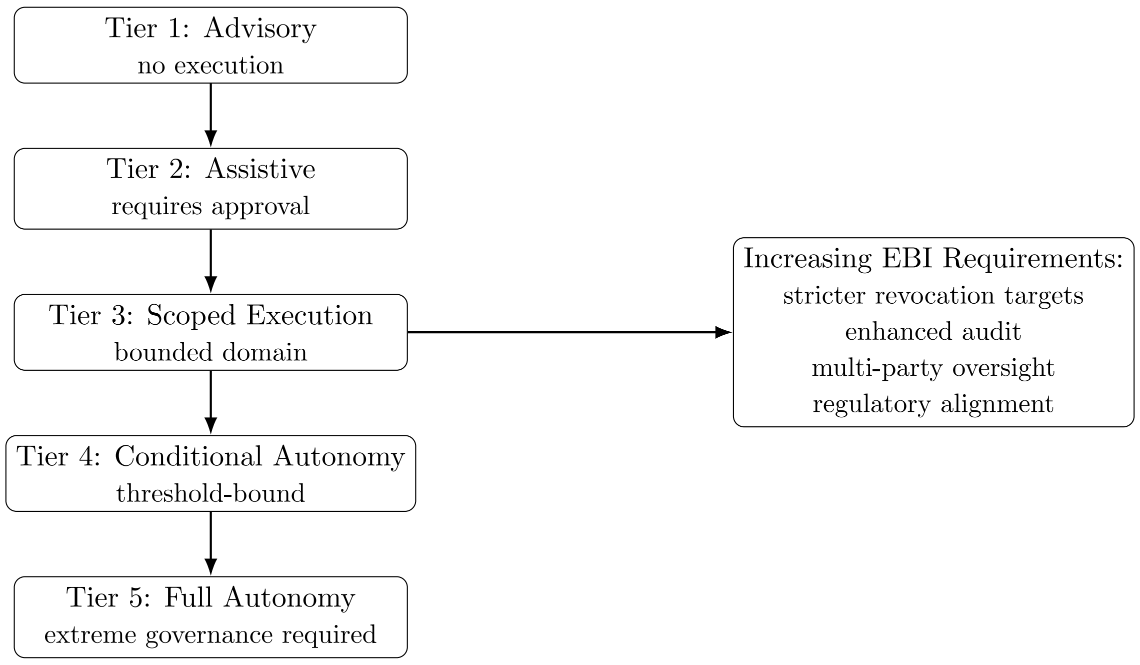

Figure 3: Tiered autonomy with corresponding EBI governance requirements. Higher tiers demand progressively stricter execution boundary controls.

7 Deterministic Revocation

Revocation must be bounded by risk tier:

Risk Tier	Revocation Latency Target	Rationale
High-risk	Sub-second to seconds	Real-time harm prevention
Medium-risk	Seconds	Workflow interruption
Low-risk	Session-bound	Minimal disruption

Table 2: Revocation latency targets by risk tier

Triggers include governance impairment, threshold breaches, regulatory reclassification, detected drift, and interlock failure.

Mechanisms include tool-use disablement, credential invalidation, session termination, and cascading containment across dependent agents. Fail-open designs are incompatible with high-risk autonomy contexts.

In this framing, revocation is not an exception-handling path; it is a primary design requirement that shapes how agents are integrated.

8 Decision Provenance and Auditability

All actions maintain machine-readable provenance scaled by risk:

- Actor or component identity
- Input context
- Policy state at execution
- Approval state
- Outcome and anomalies

High-risk systems require:

- Tamper-evident logs
- Long-term retention
- Conformity evidence generation
- Third-party audit support

Audit readiness must be continuous.

9 Authority Segmentation

No single AI component should hold both sensitive data access and unrestricted outbound communication authority unless governed by enhanced interlocks. This prevents structural blackmail, exfiltration, and coercion vectors.

10 Autonomy Drift Detection

Governance erosion is gradual and often invisible until it compounds. EBI treats drift detection as a first-class operational requirement, not an afterthought. The framework identifies three categories of signal to monitor continuously: *tool-call distribution drift* (the agent invoking tools not present at authorization time), *scope expansion* (the set of reachable actions growing beyond the authorized baseline), and *policy override frequency* (human overrides increasing as a proportion of total policy decisions, a leading indicator of policy misalignment). When any signal crosses a configurable threshold, EBI triggers mandatory review, re-authorization, or automatic tier downgrade depending on severity.

The specific monitoring algorithms, statistical methods for threshold calibration, and acceptable false-positive rates for governance contexts are not prescribed here — these depend on deployment scale, action frequency, and organizational risk tolerance, and are identified as an open research problem (OP3).

11 Multi-Agent and Distributed Deployments

When multiple agents are running together, each one needs its own enforcement gate, but they all draw their rules from a shared policy service. This creates a few coordination challenges that are worth addressing explicitly.

Keeping policies consistent across agents. Each agent holds a local copy of the current policy rules, refreshed every few seconds from the central service. For low-risk actions this is fine — a brief lag between a policy update and enforcement is acceptable. For high-risk actions, the agent must check the central service directly before proceeding, not rely on its cached copy.

Revoking access across a chain of agents. If Agent A has spawned Agent B to handle a subtask, revoking Agent A should also revoke Agent B. EBI maintains a map of which agents depend on which, so a revocation can cascade through the chain automatically. Any agent that cannot be reached is assumed revoked — the safe default is always to restrict.

Delegating authority between agents. When one agent hands off a task to another, the receiving agent gets only the subset of permissions needed for that specific task — never more than the delegating agent itself holds. Delegations are time-limited and logged. Chains of delegation are capped at a shallow depth to prevent authority from quietly accumulating through a long sequence of handoffs.

On enforcement speed. For high-risk actions checked against the central policy service, expect roughly 10–50 milliseconds of added latency per call on a co-located network. For systems where even that is too much, the enforcement gate can run directly alongside the agent on the same host, reducing overhead to under a millisecond. On network partition, the default is fail-closed: if the policy service is unreachable, the agent stops until connectivity is restored.

12 Regulatory Alignment

Regulations like the EU AI Act [6] and the NIST AI Risk Management Framework [5] place hard limits on what AI systems can do in certain contexts. EBI treats these limits as mandatory ceilings on autonomy tiers, not guidelines to consider.

In practice this means: if a deployment is classified as high-risk under applicable regulations, it cannot operate above the tier that requires mandatory human oversight — regardless of how well it scores on governance capacity. If a use case is outright prohibited, the autonomy tier is zero. When a regulatory classification changes — because the law updates, or because the agent takes on new responsibilities that change its risk category — the tier ceiling updates automatically and the agent is contained to the new limit immediately.

This is not a compliance checkbox. It is a design constraint that gets enforced the same way as any other policy rule.

13 Enforcement Integrity

13.1 Threat Model

EBI is designed with a realistic threat model in mind. The controls need to hold not just against accidental misbehavior but against adversarial inputs — prompt injections, hardcoded credential bypass, attempts to use secondary execution paths that skip the enforcement layer entirely. They also need to hold against insider threats and against the case where the model itself has been compromised or fine-tuned maliciously. There are limits: EBI cannot protect against attackers who have physical access to the hardware or who have exploited a zero-day vulnerability in the underlying operating system. Those risks require separate controls.

13.2 The Enforcement Gate

Every action request from an agent passes through a single enforcement gate before it reaches its target. The gate receives the agent's identity, what it wants to do, what resource it wants to act on, and the parameters of that action. It responds with a decision — allow, deny, or downgrade to a more limited version of the request — and issues a signed receipt that the downstream system must verify before executing anything. If the receipt is missing or doesn't match the action being taken, execution stops.

This gate is enforced at three levels simultaneously so that bypassing one level is not enough. At the application level, every tool call wrapper is required to check with the gate before proceeding. At the network level, the agent's container is locked down so it cannot make outbound calls directly — all traffic must go through a proxy that enforces policy. At the credential level, secrets are never available to the agent directly; the agent can only get a short-lived access token after the gate approves the specific action it requested. An attacker would need to defeat all three levels to act without authorization.

One specific risk this design addresses: an agent could get approval for an action and then execute a slightly different action with the same approval. To prevent this, the signed receipt is bound to the exact parameters that were approved, and it expires quickly — within half a second for high-risk actions. If the parameters don't match or the receipt has expired, the action is blocked. If the enforcement gate itself becomes unavailable for any reason, the default behavior is to stop all agent actions until it recovers.

14 Related Work

14.1 Alignment and Safety Techniques

Approaches like Constitutional AI [1] focus on shaping what the model proposes — through training, feedback, and refusal tuning. These operate inside the model and influence its behavior. EBI operates outside the model and constrains what it can do regardless of what it proposes.

14.2 Tool-Augmented Reasoning

Frameworks like ReAct [2] and Toolformer [3] show how language models increasingly integrate with external tools — and how quickly their execution surface expands as a result. Multi-agent systems like AutoGen [4] extend this further. EBI addresses the governance gap that these expanding capabilities create.

14.3 Security Engineering Foundations

EBI is built on two well-established security patterns. The first is the *reference monitor* concept [7] from operating systems: every access to a protected resource must pass through a single, tamper-resistant checkpoint that cannot be bypassed. The second is zero-trust architecture [8]: never assume that a request is legitimate based on where it comes from — verify every request explicitly. EBI applies both patterns to AI agents, treating every model output as an untrusted request that must be verified before it acts on the world.

14.4 Runtime Monitoring

Runtime monitoring [9] is the practice of watching a running system and raising alerts when its behavior violates defined rules. EBI's enforcement gate is a form of runtime monitor — but one that can block, not just observe. The difference matters: a monitor that only alerts still allows the harmful action to complete.

14.5 What EBI Adds

Existing work focuses primarily on model behavior or tool integration. EBI focuses on what happens at the moment a model output becomes a real-world action — and ensures that moment is always governed, audited, and interruptible. Table 2 summarizes how EBI compares to other approaches.

Approach	Layer	Bypassable	Revocation	Audit	Multi-Agent
Constitutional AI	Model	Yes	No	No	No
Output Filtering	Output	Yes	No	Partial	No
Tool Wrappers	Execution	Often	Manual	Varies	No
EBI (This Work)	Execution	No	Deterministic	Complete	Yes

Table 3: Comparison of EBI with existing safety approaches

15 Implementation

EBI can be deployed in three ways depending on the architecture of the system being governed. As **middleware**, the enforcement logic sits inside the agent orchestration layer and intercepts tool calls before they execute. As a **sidecar process**, it runs alongside the agent in its own container, mediating all outbound communication. As a **reverse proxy**, it sits between the agent and the external services it calls, enforcing policy at the network boundary. In all three patterns, there is a central policy service that holds the rules, and one enforcement gate per agent that checks requests against those rules.

Policies are written in plain rules: which agent is allowed to take which actions, on which resources, under which conditions. Existing policy languages like OPA, Cedar, or XACML can express these rules. What matters more than the language is the process around it: policies should be version-controlled, reviewed before changes take effect, and audited regularly. When an agent gains access to a new tool, the policy covering that tool should be updated and reviewed before the agent is allowed to use it at anything above the lowest autonomy tier.

EBI plugs into existing agent frameworks without requiring changes to the underlying model. In LangChain, the enforcement gate sits in custom tool wrappers or callbacks. In AutoGen, it runs as an agent proxy. In Semantic Kernel, it hooks into plugin filters. In LlamaIndex, it wraps the query engine. The model is never aware of the enforcement layer — which is the point.

16 What Needs to Be Built and Tested

This framework needs to be implemented and tested before its claims can be validated. The most useful next steps are:

1. Build open-source reference implementations for LangChain, AutoGen, and Semantic Kernel that teams can use as a starting point and as a testbed
2. Run controlled comparisons between agents with EBI in place and agents without it, across realistic attack scenarios: prompt injection, privilege escalation, data exfiltration, and governance impairment
3. Test what happens when the enforcement gate is unavailable, degraded, or misconfigured — failure modes matter as much as the happy path
4. Develop practical guidance on how to set drift detection thresholds for different types of deployments

Metric	Description	Target
Unauthorized Execution Rate	Attacks resulting in an unauthorized action	under 5%
Privilege Escalation Rate	Attempts that gain higher access than granted	0% (high-risk)
Revocation Speed	Time from detection to enforcement	under 500ms / 2s
False Blocks	Legitimate actions incorrectly stopped	under 1%
False Allows	Unauthorized actions incorrectly permitted	under 5%
Performance Cost	Latency added by enforcement	under 10%

Table 4: What to measure when validating EBI

Open Questions

These are things this framework does not answer. They are noted here so that researchers and practitioners working on EBI implementations know where the gaps are.

How do you calibrate the governance thresholds for AI-specific risks? The scoring thresholds in Table 1 are borrowed from industrial safety standards that were designed around hardware failures with decades of failure data behind them. AI agents fail differently — through hallucination, adversarial manipulation, and unexpected behavior in novel situations. The right thresholds for AI deployments need to be figured out through experience, not borrowed from a different domain.

How do you run the enforcement gate in systems that cannot afford any downtime? The enforcement gate fails closed by default. For systems like financial trading where even a short pause causes real harm, that default may be unacceptable. The right design for high-availability enforcement — whether that means running multiple gates in parallel, having a fallback policy for when the gate is unreachable, or something else — is an open engineering question.

How do you set drift detection thresholds in practice? This framework identifies what signals to monitor but does not say how sensitive the monitoring should be. Too sensitive and you get too many false alarms; not sensitive enough and drift goes undetected. Getting this calibration right across different deployment sizes and action rates needs hands-on experience to figure out.

What happens when two agents have conflicting policies? If Agent A is allowed to send an email and Agent B is instructed to block outbound communication, what happens when they are part of the same workflow? The default answer — the most restrictive rule wins — handles simple cases but breaks down in complex multi-agent systems. A principled approach to resolving these conflicts is still an open problem.

17 Discussion

EBI does not constrain what an AI can think. It constrains what an AI can do. That distinction matters for two reasons. First, it means the framework works regardless of which model is underneath — changing or upgrading the model does not change whether the enforcement layer is in place. Second, it means governance improvements can be made independently of model improvements. You do not have to retrain the model to tighten the rules.

EBI and model alignment are not competing approaches — they address different parts of the same problem. Alignment work reduces the chance that a model will propose something harmful. EBI ensures that even if it does, the proposal cannot become a real-world action without a human or policy control standing between them. Both layers are necessary.

18 Conclusion

AI agents that can send emails, execute transactions, and modify infrastructure need more than good behavior. They need real governance: explicit controls over what they can do, the ability to shut them down quickly, a clear record of everything they have done, and rules that hold regardless of how capable they become.

That is what EBI is for. It is not a research idea — it is a design pattern that organizations can start applying today, using tools and infrastructure they likely already have. The open questions are real, and the empirical work to answer them has not been done yet. But the core principle does not require a research paper to be true: if you cannot stop an agent from acting, you have not governed it.

Bounded power scales. Unbounded power destabilizes. The distinction is architectural.

References

References

[1] Y. Bai et al., *Constitutional AI: Harmlessness from AI Feedback*, Anthropic, 2022.

[2] S. Yao et al., *ReAct: Synergizing Reasoning and Acting in Language Models*, Proceedings of ICLR, 2023.

[3] T. Schick et al., *Toolformer: Language Models Can Teach Themselves to Use Tools*, Proceedings of NeurIPS, 2023.

[4] Q. Wu et al., *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation*, arXiv preprint arXiv:2308.08155, Microsoft Research, 2023.

[5] National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, U.S. Department of Commerce, 2023.

[6] European Parliament and Council, *Artificial Intelligence Act*, Official Journal of the European Union, 2024.

[7] J. P. Anderson, *Computer Security Technology Planning Study*, Technical Report ESD-TR-73-51, USAF Electronic Systems Division, 1972.

[8] S. Rose, O. Borchert, S. Mitchell, S. Connelly, *Zero Trust Architecture*, NIST Special Publication 800-207, 2020.

[9] M. Leucker, C. Schallhart, *A Brief Account of Runtime Verification*, Journal of Logic and Algebraic Programming, 78(5):293–303, 2009.

[10] International Electrotechnical Commission, *Functional Safety of E/E/PE Safety-related Systems (IEC 61508)*, 2010.

[11] Anonymous, *We just found out our AI has been making up [analytics data]*, Reddit, r/analytics, February 2026. Anecdotal online report; accessed February 2026. https://www.reddit.com/r/analytics/comments/1r4dsq2/we_just_found_out_our_ai_has_been_making_up/