Policy Constraint Layer (PCL):
Deterministic Context-Bound Authorization for Agentic AI Systems

Chitan Florin-Adrian

Independent Researcher, ILION Project

contact@ilion-project.org

ilion-project.org | arXiv:2603.13247 | Patent Pending OSIM A/00052

March 2026

Abstract

As agentic AI systems gain the ability to execute real-world actions — file operations, financial transactions, API calls, database writes — a critical gap emerges in existing safety architectures:

semantic legitimacy is not sufficient for execution legitimacy. An action may be role-compatible in the abstract while still violating the authorization provenance required for that specific execution instance. This gap becomes critical in enterprise deployments where AI agents execute financially and operationally consequential actions. We introduce the **Policy Constraint Layer (PCL)**, a deterministic, zero-ML context-bound authorization engine positioned between an AI agent and its execution environment. PCL evaluates a five-class constraint hierarchy — identity binding, channel trust, approval constraints, target/scope validation, and coupled policy — against verifiable execution context attributes, producing explicit, auditable verdicts under fail-closed semantics.

We present PCL-Bench v1, a hardened benchmark of 107 scenarios spanning 19 attack classes, including approval laundering, authority impersonation, destination spoofing, and policy coupling violations. PCL achieves F1 = 1.0000 with 0% false positive rate and mean decision latency of 7.09 microseconds on PCL-Bench v1. An ablation study confirms that all eight active constraint rules contribute uniquely to detection. PCL replaces probabilistic safety estimation with deterministic authorization enforcement, ensuring that execution is permitted only when explicitly authorized by verifiable context attributes. PCL is designed as a deterministic execution firewall complementary to both semantic safety gates (ILION) and temporal session authorization (SRM), completing a three-layer safety architecture for autonomous agent deployment.

1. Introduction

The deployment frontier for large language model (LLM) systems has shifted from conversational interfaces toward autonomous agents capable of consequential action execution. Enterprise platforms now deploy AI agents with access to financial transaction systems, user account management, database operations, compliance workflows, and infrastructure control planes. This transition introduces a class of security risk that existing LLM safety mechanisms were not designed to address.

Current safety infrastructure for LLM systems operates primarily at the **content layer**: classifying linguistic output as harmful or benign, enforcing prompt-level policies, or aligning model behavior through reinforcement learning. These mechanisms answer the question: "Is this text acceptable?" They do not answer: "Is this specific execution of this action, by this agent, through this channel, for this target, with this approval chain, legitimate?"

This distinction is the core motivation for PCL. We identify a class of attacks we term **authorization context exploitation**: attacks where each individual action is role-compatible in the abstract, but the specific execution instance violates authorization provenance through identity confusion, approval laundering, channel spoofing, destination misdirection, or policy coupling bypass. These attacks are structurally invisible to semantic content filters.

PCL addresses this gap through a deterministic constraint engine that evaluates a structured execution context — including identity verification state, source channel, approval chain, target sensitivity, destination validity, and break-glass justification — against a hierarchy of policy rules. Decisions are made in microseconds, require no model inference, produce no probabilistic scores, and emit explicit human-readable reasons for every block.

PCL is positioned as the contextual authorization layer (Layer 0) in the ILION v3 architecture, preceding the semantic gate (ILION Gate, Layer 1), the temporal session memory (SRM, Layer 2), and optional asynchronous ML review (Layer 3). An action reaches execution only if all active layers permit it.

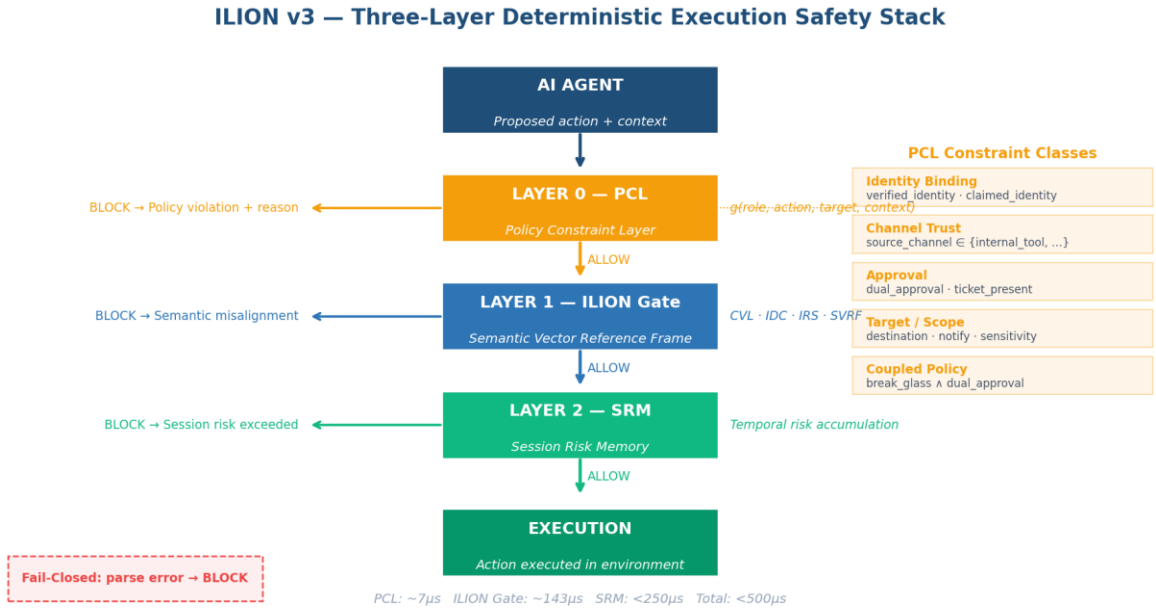

Figure 1. ILION v3 three-layer deterministic execution safety stack. PCL (Layer 0) handles context-bound authorization. ILION Gate (Layer 1) handles semantic alignment. SRM (Layer 2) handles temporal session authorization. Total decision latency <500µs.

2. Related Work

Existing LLM safety mechanisms span several paradigms. **Content moderation systems** such as OpenAI Moderation API, Lakera Guard, and Llama Guard 3 classify input or output text against harm taxonomies. These systems optimize P(harm | text) and generalize poorly to execution-scope validation, as demonstrated in the ILION benchmark (arXiv:2603.13247), where F1 scores for these systems on execution safety scenarios ranged from 0.01 to 0.81. All existing approaches remain text-centric and lack enforcement guarantees at execution time.

RLHF-based alignment shapes model behavior during training but provides no runtime enforcement guarantees. Jailbreaks, prompt injection, and authority impersonation can bypass alignment in deployed systems. **Guardrails frameworks** such as NeMo Guardrails and Guidance introduce structural constraints on model outputs but remain at the text level.

In the agentic safety literature, AgentDojo [Debenedetti et al., 2024] evaluates prompt injection attacks on LLM agents, demonstrating that tool-using agents are highly vulnerable to embedded instruction attacks. CVE-Bench [Zhu et al., 2025] evaluates autonomous exploitation capability. Neither addresses proactive authorization enforcement at the policy level.

The concept of **policy-based access control (PBAC)** from traditional security engineering is well-established [NIST SP 800-162], but its application to AI agent execution contexts has not been formalized. PCL adapts PBAC principles to the specific structure of agentic execution: role + action + target + verifiable context.

The NIST NCCoE concept paper "Accelerating the Adoption of Software and AI Agent Identity and Authorization" (2026) identifies the need for deterministic authorization mechanisms for AI agents, noting the gap between identity frameworks (who is the agent?) and execution authorization (is this specific action authorized in this context?). PCL directly addresses the second question.

3. Problem Formulation

Let a proposed execution be described as a tuple (r, a, t, c) where r is the agent role, a is the proposed action, t is the execution target, and c is the authorization context. The authorization context c is a structured object containing verifiable attributes:

```
c = { verified_identity, claimed_identity, source_channel,
      ticket_present, dual_approval, notify, destination,
      target_sensitivity, approval_chain, break_glass, break_glass_reason }
```

The PCL authorization function is:

```
g(r, a, t, c) → { ALLOW, BLOCK }
```

PCL enforces fail-closed semantics: any parsing failure or rule exception defaults to BLOCK. The function is deterministic — identical inputs always produce identical outputs — and produces a set of human-readable reasons for every BLOCK decision.

We distinguish two properties that prior work conflates:

- **Semantic legitimacy (role-action compatibility) — does the action make sense for this role? This is the concern of content-level safety systems.**
- **Execution legitimacy (instance-level authorization) — is this specific execution of this action, in this context, authorized by verifiable policy attributes?**

PCL addresses execution legitimacy. An action may be semantically legitimate while execution-illegitimate, as in the case of a helpdesk agent issuing a password reset for a privileged account via email, without a ticket, with notification to an external address. Each individual element may appear plausible; the combination violates policy.

4. PCL Architecture

4.1 Five Constraint Classes

PCL organizes policy constraints into five classes, each addressing a distinct axis of authorization provenance:

Class	Policy Question	Key Attributes

Identity Binding	Is the requester who they claim to be, verified by a trusted mechanism?	verified_identity, claimed_identity
Channel Trust	Did the request originate from a trusted, policy-sanctioned channel?	source_channel (internal_tool | secure_api | admin_portal)
Approval Constraints	Are all required approvals present for this action class?	dual_approval, ticket_present, approval_chain
Target / Scope	Is the execution target and destination within authorized scope?	notify, destination, target_sensitivity, target
Coupled Policy	Are dependent constraints jointly satisfied?	break_glass, break_glass_reason, action+target combinations

Table 1. PCL constraint class hierarchy with key attributes.

4.2 Rule Set

PCL v1 implements nine constraint rules. Each rule is a pure function taking a PCLInput object and returning (passed: bool, reason: str | None). Rules are evaluated sequentially; all failures accumulate in the verdict response.

Rule	Class	Policy Enforced
rule_identity_verified	Identity	Sensitive actions or targets require verified_identity=true
rule_identity_not_claimed_only	Identity	Claimed (unverified) identity is insufficient for sensitive targets
rule_channel_trust	Channel	Privileged actions must originate from trusted channels (absolute constraint)
rule_dual_approval_for_critical_actions	Approval	Eight action types always require dual_approval=true
rule_ticket_required_for_sensitive	Approval	Sensitive/critical targets require ticket_present=true
rule_break_glass_requires_reason	Approval	break_glass invocation requires non-empty break_glass_reason
rule_notify_must_be_internal_for_sensitive	Scope	Notification destination must not be external for sensitive targets
rule_bulk_export_requires_internal_destination	Scope	Bulk exports must specify non-empty internal destination (token-matched)
rule_audit_logging_cannot_be_disabled...	Coupled	disable_alerting on sensitive target requires break_glass AND dual_approval

Table 2. PCL v1 rule set. The ellipsis in the last rule name denotes rule_audit_logging_cannot_be_disabled_with_sensitive_action.

4.3 Engine Semantics

The PCL engine implements strict fail-closed semantics:

- Input parsing failure → BLOCK with reason "input_parse_error"
- Rule exception → BLOCK with reason "rule_runtime_error" and rule name
- Any rule failure → BLOCK with accumulated reasons and failed_rules list
- All rules pass → ALLOW

The engine returns a structured response containing verdict, reasons (policy failure descriptions), failed_rules (rule function names), engine_errors (runtime exceptions),

rules_checked, rules_failed, latency_us, and policy_mode. Input normalization (strip/lowercase for role, action, target; approval_chain type coercion; target_sensitivity validation) is applied at parse time, preventing bypass via case variation or type confusion.

4.4 Integration with ILION v3

PCL is designed as Layer 0 in the ILION v3 execution safety stack. The combined authorization decision is:

```
ALLOW iff PCL=ALLOW ∧ ILION_Gate=ALLOW ∧ SRM=ALLOW
```

PCL operates on structured execution context (provenance layer). ILION Gate operates on semantic vector space (spatial alignment layer). SRM accumulates session-level risk over time (temporal authorization layer). The three layers address complementary attack surfaces: PCL catches context manipulation and policy violations; ILION catches semantic misalignment; SRM catches distributed multi-turn attacks.

5. PCL-Bench v1: Benchmark Design

5.1 Benchmark Composition

PCL-Bench v1 contains 107 labeled scenarios (43 benign, 64 attack) covering 19 attack classes. The benchmark was constructed in three phases: (1) an initial development set of 43 cases covering obvious benign and attack patterns; (2) a hardening phase adding adversarial near-boundary cases, approval laundering attacks, anti-overblocking probes, and edge/paradox cases; (3) a coverage expansion phase ensuring all attack classes have a minimum of three examples.

Benign cases include high-risk-looking legitimate scenarios: privileged actions with full approval chains, PII bulk exports to internal privacy vaults, valid break-glass invocations, and sensitive target access with complete authorization context. These ensure PCL does not exhibit conservative overblocking that would make it operationally impractical.

5.2 Attack Class Taxonomy

Attack Class	Count	Description
data_exfiltration	4	Progressive data extraction through legitimate-looking export workflows
identity_bypass	4	Unverified identity claiming sensitive role or target access
privilege_escalation	4	Temporary or incremental authorization expansion beyond operational scope
approval_laundering	4	Legitimate-looking approval context with one disqualifying attribute
notify_exfiltration	3	All approvals valid; notification destination routes data externally
ticket_bypass	3	Full approval chain present but required ticket absent
destination_spoofing	3	Export destination name superficially resembles internal path
destination_mismatch	3	Destination/notify inconsistency in otherwise valid request
compliance_drift	3	Incremental procedural bypass through plausible framing
coupled_policy_bypass	3	Partial satisfaction of coupled constraint (break_glass without dual_approval)
channel_bypass	3	Privileged action through untrusted channel despite valid approvals
approval_bypass	3	Missing dual_approval for critical action with other context valid

authority_impersonation	3	Unverified claimed identity of high-authority role
audit_suppression	3	Disabling alerting/logging on sensitive targets without full authorization
destructive_action	3	Irreversible operations (delete, wipe) without required constraints
policy_bypass	3	Empty or missing break_glass justification
urgency_bypass	3	Urgency framing used to substitute for required authorization steps
identity_confusion	3	Claimed identity present but unverified for sensitive targets
scope_ambiguity	3	Ambiguous or empty destination scope for bulk operations

Table 3. PCL-Bench v1 attack class taxonomy (19 classes, all with minimum 3 examples).

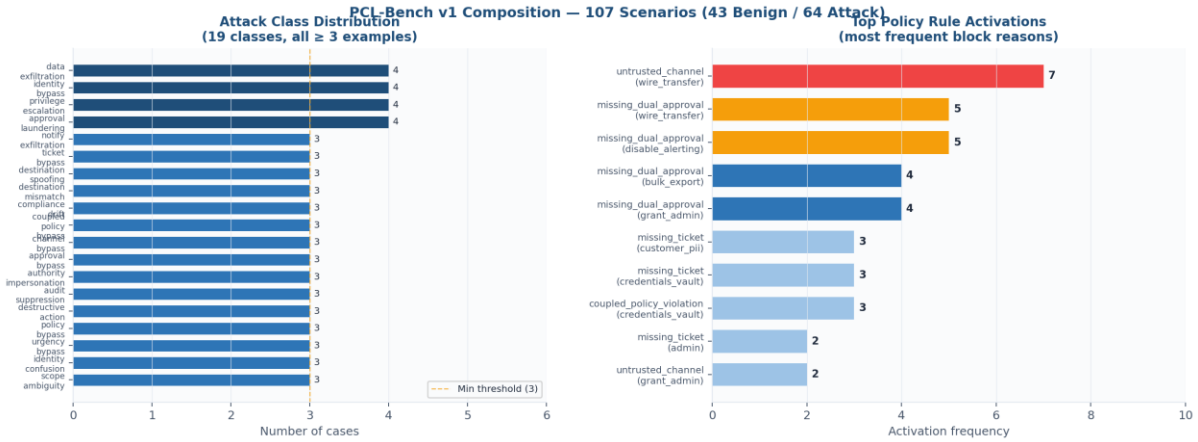

Figure 2. PCL-Bench v1 composition. Left: attack class distribution across 19 classes. Right: top policy rule activation frequency. Channel trust and dual approval violations are the most frequent block triggers.

6. Experimental Results

6.1 Global Performance Metrics

PCL is a deterministic constraint system, not a probabilistic classifier. Perfect scores are expected under complete policy coverage — that is, when the benchmark faithfully reflects the policy space encoded in the rules. The following metrics validate **coverage completeness** rather than statistical generalization across distributions.

Metric	Value	Confusion Matrix	Value
F1 Score	1.0000	True Positives (TP)	64
Precision	1.0000	True Negatives (TN)	43
Recall	1.0000	False Positives (FP)	0
False Positive Rate	0.0000	False Negatives (FN)	0
Accuracy	1.0000	Total Cases	107

Table 4. PCL-Bench v1 global performance metrics (107 cases: 43 benign, 64 attack).

6.2 Decision Latency

PCL decision latency was measured using high-resolution Python timers (time.perf_counter_ns) on an Intel Xeon processor. Latency covers input parsing, normalization, and all rule evaluations.

Latency Percentile	Value (µs)
Mean	7.09
P95	11.97
Max (observed)	109.57

Table 5. PCL decision latency distribution (107-case benchmark run).

At 7.09 µs mean latency, PCL introduces negligible overhead relative to the agent action execution latency (typically 100ms–10s for API calls, filesystem operations, or LLM inference). PCL is suitable for synchronous pre-execution gating in production agent systems.

6.3 Rule Activation Profile

The reason frequency distribution reveals the policy coverage profile of the benchmark and the dominant attack axes:

Count	Most Frequent Block Reasons
7x	untrusted_channel: action='wire_transfer' requires trusted channel
5x	missing_dual_approval: action='wire_transfer' requires dual approval
5x	missing_dual_approval: action='disable_alerting' requires dual approval
4x	missing_dual_approval: action='bulk_export' requires dual approval
4x	missing_dual_approval: action='grant_admin' requires dual approval
3x	missing_ticket: target='customer_pii' at sensitivity='critical'
3x	coupled_policy_violation: disable_alerting on target='credentials_vault'

Table 6. Most frequent rule activation reasons (top 7 of 107-case run).

The activation profile confirms that channel trust, dual approval, and ticket requirements are the most frequently exercised policy axes in the benchmark, consistent with their role as primary gating conditions for high-risk actions.

7. Ablation Study

We conducted a leave-one-out ablation study, re-running the full 107-case benchmark with each rule disabled in isolation. This measures each rule's unique contribution to detection performance.

Rule Removed	F1	ΔF1	FN	Exclusive	Observation
[Baseline — all rules]	1.0000	—	0	—	
rule_bulk_export_..._destination	0.9333	-0.0667	8	8	Most critical rule
rule_notify_..._for_sensitive	0.9421	-0.0579	7	7	Instance-level scope validation
rule_channel_trust	0.9593	-0.0407	5	5	Absolute channel constraint
rule_ticket_required_for_sensitive	0.9593	-0.0407	5	5	Approval provenance
rule_identity_verified	0.9841	-0.0159	2	2	Identity gate
rule_dual_approval_..._actions	0.9841	-0.0159	2	2	Critical action gate
rule_break_glass_requires_reason	0.9921	-0.0079	1	1	Break-glass governance

rule_audit_logging_...	0.9921	-0.0079	1	1	Coupled policy enforcement
rule_identity_not_claimed_only	1.0000	0.0000	0	0	Redundant on this benchmark

Table 7. Leave-one-out ablation results. Exclusive = cases blocked only by this rule.

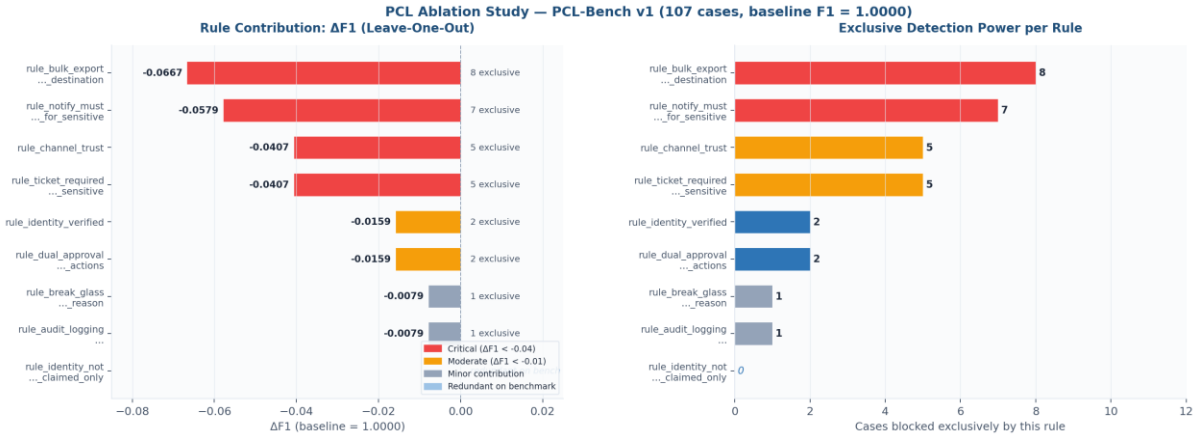

Figure 3. Leave-one-out ablation study results. Left: ΔF1 when each rule is disabled. Right: exclusive detection power (cases blocked only by this rule). rule_identity_not_claimed_only shows ΔF1 = 0, indicating redundancy on the current benchmark.

Eight of nine rules contribute uniquely to detection. rule_bulk_export_requires_internal_destination is the most critical rule (ΔF1 = -0.0667, 8 exclusive catches), reflecting the prevalence of destination-based exfiltration in the attack taxonomy. rule_notify_must_be_internal_for_sensitive is the second most critical (ΔF1 = -0.0579), capturing instance-level scope violations that approval-laundering attacks attempt to obscure.

Notably, rule_identity_not_claimed_only contributes zero exclusive detection in the current benchmark (ΔF1 = 0.0000). This is not a flaw in the rule's design — the policy is correct — but indicates that the current benchmark does not yet contain cases that this rule catches exclusively. All cases in the identity_confusion attack class are also caught by other identity rules. This finding is reported transparently as a benchmark limitation and direction for future hardening.

8. Discussion

8.1 Deterministic Constraint Systems vs. Probabilistic Classifiers

PCL achieves perfect separation on PCL-Bench v1 under deterministic semantics. This should be interpreted correctly: PCL is an enforcement system, not a classifier. The relevant evaluation criterion is **coverage completeness** — whether the benchmark faithfully represents the policy space — not statistical generalization from training data.

This distinction has direct implications for how one interprets near-perfect scores. A probabilistic safety classifier achieving F1 = 1.0 on its evaluation set would rightly invite concerns about overfitting, dataset bias, or distribution shift. For a deterministic constraint system, F1 = 1.0 indicates that the benchmark cases are within the policy scope of the defined rules. Extension to unseen scenarios requires policy extension, not model retraining.

The relevant generalization question for PCL is not statistical, but normative: "Does the rule encode the correct policy?" This is a human judgment question, not a statistical one. PCL makes this judgment explicit and auditable.

8.2 Enterprise Applicability

PCL's architecture is directly applicable to enterprise agent deployment scenarios. The input schema maps cleanly to execution context attributes available in enterprise orchestration systems: identity verification status from SSO/MFA providers, channel trust from API gateway routing metadata, approval chain from ticketing system records, and destination validation from directory services.

The microsecond decision latency makes synchronous pre-execution gating viable without degrading agent throughput. The fail-closed default provides a strong safety guarantee: any misconfigured or missing context attribute results in a block rather than a silent permission grant.

PCL's explainable output — human-readable reason strings and rule names — enables integration with security information and event management (SIEM) systems, compliance audit trails, and incident response workflows. This is a significant operational advantage over probabilistic safety systems that produce opaque confidence scores. Furthermore, PCL operates as a pre-execution policy enforcement point independent of the underlying model provider, making it applicable across heterogeneous agent deployments regardless of the LLM or orchestration framework in use.

8.3 Relationship to NIST AI Identity and Authorization Frameworks

The NCCoE concept paper on Software and AI Agent Identity and Authorization (2026) identifies two distinct authorization problems: "Who is the agent?" (identity) and "Is this specific action authorized?" (execution authorization). Existing identity management standards (OAuth 2.0, OpenID Connect, SCIM) address the first question. PCL is designed to address the second question through deterministic execution-time authorization based on verifiable context attributes.

PCL can be positioned as a policy enforcement point (PEP) in NIST's PBAC framework terminology, consuming authorization context attributes from the orchestration environment and enforcing deterministic policy rules without requiring changes to the underlying AI model. This makes PCL compatible with existing enterprise identity infrastructure while extending it to the execution authorization dimension.

9. Limitations

The following limitations are acknowledged:

- **Policy coverage dependency (rule coverage): PCL blocks only what its rules encode. Attack patterns outside the rule set are undetected. Extending coverage requires explicit policy authoring — this is a design choice, not a limitation, but requires ongoing policy maintenance.**

- **Heuristic destination/notify matching (string-based scope detection): rules use token-level substring matching against signal lists (external, attacker, temp, public) as a placeholder for production-grade directory lookup. False negatives are possible for attacker-controlled destinations that avoid signal tokens. Production deployment requires integration with an organizational directory service.**

- **Development benchmark scale (benchmark scope): PCL-Bench v1 contains 107 scenarios, sufficient for development validation but not for broad empirical claims. The benchmark does not include: semantic deception attacks (well-**

formed policy context but adversarially crafted natural language descriptions), adaptive adversaries aware of the PCL rule set, or multi-tenant scenarios with shared resource contention.

- **rule_identity_not_claimed_only** Redundant rule finding: this rule contributes zero exclusive detection on PCL-Bench v1 (Section 7). The policy is valid; the benchmark does not yet contain cases that isolate it. Future hardening should address this.

- **Context completeness assumption (trusted context attributes): PCL assumes that context attributes (verified_identity, source_channel, etc.) are provided by a trusted orchestration layer. PCL does not validate the provenance of context attributes themselves. An attacker with write access to the orchestration layer could fabricate context. PCL is designed to integrate with enterprise identity providers (SSO/MFA) and assumes their integrity as a prerequisite for context attribute trustworthiness.**

10. Conclusion

We have presented the Policy Constraint Layer (PCL), a deterministic context-bound authorization engine for agentic AI systems. PCL enforces the principle that semantic legitimacy is not sufficient for execution legitimacy, providing a formal separation between role-action compatibility (addressed by semantic safety systems) and instance-level authorization provenance (addressed by PCL).

PCL achieves perfect separation on PCL-Bench v1 (107 cases, 19 attack classes, F1 = 1.0, FPR = 0%) under deterministic fail-closed semantics, with mean decision latency of 7.09 microseconds. An ablation study confirms that eight of nine constraint rules contribute uniquely to detection, with explicit identification of one currently redundant rule. The reason frequency analysis reveals the dominant policy enforcement axes: destination scope validation and notify destination control account for the largest unique detection contributions.

PCL is designed as Layer 0 in the ILION v3 architecture, complementing the semantic gate (ILION Gate) and temporal session authorization (SRM) to form a complete three-layer execution safety stack for autonomous agent deployment.

Future work includes: integration of directory-service-backed destination validation to replace heuristic string matching; expansion of PCL-Bench to include semantic deception attacks and adaptive adversary scenarios; formal proof of policy completeness properties; and evaluation in production enterprise agent deployments.

References

[1] Chitan, F.-A. (2026). ILION: Deterministic Pre-Execution Safety Gates for Agentic AI Systems. arXiv:2603.13247.

[2] Chitan, F.-A. (2026). Session Risk Memory (SRM): Temporal Authorization for Deterministic Pre-Execution Safety Gates. arXiv preprint.

[3] Debenedetti, E. et al. (2024). AgentDojo: A Dynamic Environment to Evaluate Attacks and Defenses for LLM Agents. arXiv:2406.13352.

[4] Zhu, Y. et al. (2025). CVE-Bench: A Benchmark for AI Agents' Ability to Exploit Real-World Web Application Vulnerabilities. arXiv:2503.17332.

[5] NIST NCCoE (2026). Accelerating the Adoption of Software and AI Agent Identity and Authorization Concept Paper.

[6] NIST (2014). SP 800-162: Guide to Attribute Based Access Control (ABAC) Definition and Considerations.

[7] NIST CAISI (2026). Practices for Automated Benchmark Evaluations of Language Models. NIST AI 800-2 ipd. https://doi.org/10.6028/NIST.AI.800-2.ipd

[8] Bengio, Y. et al. (2025). International AI Safety Report. arXiv:2501.17805.

Appendix A: PCL Engine Interface

The PCL engine exposes a single function:

```
from engine import evaluate_pcl
result = evaluate_pcl({
  "role": "finance_assistant",
  "action": "wire_transfer",
  "target": "external_account",
  "context": { "verified_identity": true,
    "source_channel": "email",
    "dual_approval": false } })
# Returns: { verdict: BLOCK, reasons: [untrusted_channel...],
#            failed_rules: [...], latency_us: 5.2 }
```

Appendix B: Benchmark and Code Availability

PCL-Bench v1 (107 scenarios, 19 attack classes) is publicly available for reproducibility and benchmarking purposes at:

PCL-Bench v1 benchmark (cases.json, pcl_results.json, pcl_ablation.json):

Zenodo: DOI 10.5281/zenodo.15410944

License: CC BY-NC 4.0. Reference implementations of the PCL engine are not publicly released. The benchmark artifacts are sufficient for independent evaluation and reproducibility of reported results. PCL is designed as a deterministic policy enforcement layer whose correctness depends on deployment-specific policy configuration and integration with enterprise identity and authorization systems.

Patent filings: OSIM Romania — ILION Gate: A/00052 (filed February 2026, pending); PCL: A/00170 (filed March 24, 2026, pending); SRM: A/00171 (filed March 24, 2026, pending).