COMPOSITIONAL SECURITY CONTROL FOR AI-ASSISTED CODING WORKFLOWS: A THREAT-MODEL-GROUNDED SECURITY–PRODUCTIVITY FRONTIER

Anonymous authors
Paper under review

ABSTRACT

AI-assisted coding systems now influence repository edits, dependency selection, and deployment decisions, which creates coupled attack surfaces spanning prompt-channel abuse, supply-chain compromise, and unsafe escalation policies. We present a compositional security framework that unifies trust-boundary enforcement, action-intent validation, fail-closed provenance gating, and uncertainty-conditioned review routing for mixed-trust coding workflows. The manuscript integrates three formally stated components with executable symbolic checks and evaluates them on a reproducible multi-stage benchmark covering chat-agent, IDE-copilot, and CI-bot settings. Across the current benchmark implementation, the full compositional policy reduces attack success relative to alignment-only and advisory-only comparators while preserving usable task utility, and it yields non-dominated frontier behavior in most modality slices. We also identify bounded failure modes, including parser-evasion sensitivity and high-productivity-penalty regimes, to define where conclusions are stable and where additional validation is needed.

1 INTRODUCTION

LLM-assisted "vibe coding" changes how software is produced: developers increasingly operate through conversational agents, IDE copilots, and automation bots that can read external material, select dependencies, invoke tools, and propose release actions. This interaction model can shorten development cycles, but it also moves risk from purely static code defects toward dynamic decision boundaries in which instructions, context, and tool outputs compete for control over action selection (Gulyamov et al., 2026; Yu et al., 2026; Jia et al., 2025; Project, 2025; Yi et al., 2025; Zhan et al., 2025; Wang et al., 2025; Yao et al., 2024). Existing secure SDLC practice remains essential, yet many controls were designed for pipelines where humans explicitly gate most high-impact transitions, not for workflows where assistants routinely synthesize and execute multi-step plans (Bhardwaj et al., 2025; Bezzateev & Blinov, 2025; Coston et al., 2025; Regenscheid et al., 2024; Cybersecurity & Agency, 2023; of Standards & Technology, 2023).

A second challenge is cross-channel coupling. Prompt and context attacks operate at runtime, supply-chain abuse can enter through dependency suggestions, and productivity-preserving routing policies can accidentally amplify risk when uncertainty is poorly calibrated (Truong et al., 2025; Williams et al., 2025; Maintainers, 2025; Peng et al., 2025a;b; Huang et al., 2025; Zheng et al., 2024; Reichert & Obelheiro, 2024; Group, 2023). In the literature, these channels are often studied in isolation, producing strong but narrow claims. For example, alignment-centric runtime defenses can show low attack success in one benchmark while adaptive payload studies recover high success under topic transitions or parser-evasion patterns (Chen et al., 2025; Jia et al., 2025; Yi et al., 2025; Zhan et al., 2025; Wang et al., 2025). Similarly, governance frameworks strongly motivate provenance and policy controls, but empirical reporting of operational overhead and reviewer burden is less consistent (Bhardwaj et al., 2025; Regenscheid et al., 2024; Cybersecurity & Agency, 2023; Group, 2023).

This manuscript addresses that gap with a compositional, evidence-linked formulation: we define a joint policy class for mixed-trust coding workflows, derive formal properties for each policy layer, and connect theorem-level obligations to benchmark artifacts and caveats. The goal is not to claim a universal defense, but to characterize a defensible operating frontier where risk reduction and developer utility can be co-optimized under explicit assumptions. The resulting contribution sits between purely empirical benchmark papers and framework-only guidance by making provenance, assumptions, and failure conditions explicit in one unified narrative.

Our main contributions are:

- We formalize a mixed-trust coding workflow model with explicit trust domains, permission lattice constraints, and a dual-objective security–productivity criterion that can be audited across runtime and supply-chain channels.
- We define a compositional control stack that combines runtime intent/permission enforcement, fail-closed provenance admission, and uncertainty-conditioned escalation, and we distinguish inherited conventions from manuscript-defined audit quantities.
- We provide theorem-level guarantees for monotonicity, fail-closed dominance, and threshold optimality under stated assumptions, with complete proof blocks and symbolic verification obligations.
- We evaluate the composed policy on a multi-modality benchmark and map each core claim to concrete figures, tables, and negative-case logs, including regimes where gains weaken.
- We deliver a reproducibility-oriented synthesis that translates contradiction patterns in prior work into a practical deployment checklist and a bounded future-work program.

2 Problem Setting and Formal Objectives

2.1 Workflow model and trust domains

We consider coding workflows composed of task instances $s \in \mathcal{S}$ generated from enterprise and open-source profiles. Each instance is represented by a sequence of artifacts x_t (issue text, documentation snippets, tool outputs, dependency metadata, and generated code) and candidate actions a_t (file edits, dependency updates, CI triggers, and release proposals). Following mixed-trust threat modeling guidance (Yu et al., 2026; Project, 2025; Yi et al., 2025; Regenscheid et al., 2024), each artifact is mapped to a trust domain through a deterministic classifier

$$\tau : \mathcal{X} \to \mathcal{T} = \{\text{trusted\_intent}, \text{untrusted\_content}, \text{tool\_output}, \text{repo\_state}\}. \tag{1}$$

The action policy is allowed to consume all artifacts, but only artifacts in approved domains can directly authorize privileged transitions. This separation is critical because indirect prompt injection attacks exploit precisely the absence of such boundaries (Gulyamov et al., 2026; Chen et al., 2025; Zhan et al., 2025; Wang et al., 2025).

Permissions are modeled as elements of a lattice $p \in \mathcal{P}$ where $p_1 \leq p_2$ denotes capability containment over files, tools, network scopes, and secrets. For each task s, the policy computes a least-privilege projection $\pi_P(s)$ and only executes actions in the feasible set $\mathcal{F}(\pi_P(s))$. This construction borrows least-privilege semantics from secure pipeline practice (Bhardwaj et al., 2025; Coston et al., 2025; Regenscheid et al., 2024) and adapts them to assistant-mediated execution.

2.2 Decision variables and feasible set

The overall decision object is a composed policy

$$\pi = (\pi_T, \pi_I, \pi_P, \pi_G, \pi_U) \in \Pi, \tag{2}$$

where π_T performs trust-boundary checks, π_I performs intent alignment checks, π_P enforces permission feasibility, π_G evaluates supply-chain provenance gates, and π_U routes uncertain actions for escalation. The feasible policy set is

$$\Pi_{\text{feas}} = \{\pi \in \Pi \mid a_t \in \mathcal{F}(\pi_P(s)),\ \pi_G(c) \in \{0,1\},\ \tau(x_t) \text{ constraints hold}\}. \tag{3}$$

Equation 3 makes explicit that runtime and supply-chain constraints are enforced jointly rather than sequentially patched.

2.3 Objective family

For runtime attack control, we define

$$J_1(\pi) = \mathbb{E}_{s\sim\mathcal{S}}\left[\mathbf{1}\{E_T(s,\pi) \cup E_I(s,\pi) \cup E_P(s,\pi)\}\right] + \lambda\,\mathbb{E}[(U_{\min} - U(s,\pi))_+] + \eta\,\mathbb{E}[V(s,\pi)], \tag{4}$$

where E_T, E_I, E_P are layer-bypass events, U is task utility, and V is policy-violation mass. Equation 4 is manuscript-defined and captures the tradeoff emphasized by secure-coding evaluation work (Jia et al., 2025; Peng et al., 2025a;b; Huang et al., 2025).

For dependency admission, we define a joint fail-closed predicate over changed dependencies $d \in D(c)$:

$$\text{Release}_{\text{joint}}(c) = \prod_{d\in D(c)} R_d\, A_d\, B_d, \tag{5}$$

where $R_d = \mathbf{1}[r_d \geq \tau_r]$ (advisory threshold), A_d is attestation validity, and B_d is SBOM consistency. Equation 5 composes advisory and provenance channels motivated by Maintainers (2025), in-toto Project Contributors (2024), Team (2024), and Group (2023).

For uncertainty-conditioned escalation, we define

$$J_3(\tau_u) = \int_0^{\tau_u} w_r(1-\beta)q(u)f(u)\,du + \int_{\tau_u}^1 w_p\kappa f(u)\,du, \tag{6}$$

where u is an uncertainty score with density f, $q(u)$ is residual exploit risk after automated handling, β is review effectiveness, and κ is productivity penalty. Equation 6 operationalizes the frontier objective suggested by uncertainty and governance studies (Bhardwaj et al., 2025; Peng et al., 2025a; Huang et al., 2025; Regenscheid et al., 2024; Cybersecurity & Agency, 2023).

2.4 Assumptions and Scope

Our conclusions are conditioned on four assumptions: (i) mixed-trust artifacts are correctly labeled by τ at the granularity needed for policy checks; (ii) attacker behavior includes adaptive families such as topic-transition and parser-evasion payloads; (iii) provenance evidence channels are observable for evaluated dependency events; and (iv) utility and overhead metrics are measured consistently across modalities. These assumptions are aligned with the contradiction map in recent literature, where benchmark conclusions are highly sensitive to adaptivity, metric definitions, and gate semantics (Chen et al., 2025; Tamberg & Bahşi, 2025; Peng et al., 2025a;b; Zhan et al., 2025; Wang et al., 2025; Liu et al., 2024).

3 Related Work, Contradictions, and Novelty Boundary

3.1 Prompt-channel robustness versus adaptive attack pressure

Recent studies establish that indirect prompt injection remains a central threat in tool-integrated agents (Gulyamov et al., 2026; Yu et al., 2026; Project, 2025; Yi et al., 2025; Zhan et al., 2025; Wang et al., 2025). Alignment-oriented defenses such as task-goal checking can substantially reduce attack success in controlled settings (Jia et al., 2025), yet adaptive red-teaming and topic-transition attacks often recover substantial leverage (Chen et al., 2025; Zhan et al., 2025; Wang et al., 2025). The methodological tension is not whether either line is "correct," but that they stress different attack regimes. Our design inherits the alignment insight while explicitly modeling adaptive families and layer-order ablations, preventing one benchmark configuration from being interpreted as a global guarantee.

3.2 Supply-chain governance: advisory signals versus cryptographic provenance

Software supply-chain security literature converges on the importance of dependency hygiene, attestation, and policy gates (Truong et al., 2025; Williams et al., 2025; Maintainers, 2025; in-toto Project Contributors, 2024; Zheng et al., 2024; Reichert & Obelheiro, 2024; Team, 2024; Group, 2023). However, assurance depth differs across channels: advisory scoring provides broad risk signals but can be bypassed by crafted edge cases, while provenance requirements provide stronger guarantees at the cost of integration friction (Maintainers, 2025; in-toto Project Contributors, 2024; Team, 2024). Our contribution does not claim a new provenance standard; instead, it formalizes how these channels interact in a fail-closed admission rule and quantifies the operational tradeoff.

3.3 Secure SDLC frameworks and operationalization gap

Framework-level guidance from NIST SSDF, secure-by-design principles, and AI risk governance emphasizes layered controls and accountability (Regenscheid et al., 2024; Cybersecurity & Agency, 2023; of Standards & Technology, 2023). Empirical studies in DevSecOps contexts confirm that integration complexity and staffing constraints can undermine deployment quality (Bhardwaj et al., 2025; Bezzateev & Blinov, 2025; Coston et al., 2025). We address the operationalization gap by translating framework principles into measurable predicates and benchmark outcomes, while keeping infrastructure limits as reproducibility context rather than scientific novelty.

3.4 Functionality-security mismatch and benchmark design

A recurring finding in code-LLM evaluation is that functionally correct outputs can still be vulnerable (Tamberg & Bahşi, 2025; Peng et al., 2025a;b; Liu et al., 2024). This mismatch motivates dual-objective reporting and uncertainty-

Algorithm 1 Compositional Security-Utility Policy Evaluation

Input: scenario s, artifacts $\{x_t\}$, candidate actions $\{a_t\}$, dependency change set $D(c)$, uncertainty scores u_t
Assign trust domains using $\tau(x_t)$ and reject untrusted instruction promotion.
for each candidate action a_t **do**
  Apply intent alignment check and permission feasibility test $a_t \in \mathcal{F}(\pi_P(s))$.
  if action is dependency or release affecting **then**
    Evaluate $\mathrm{Release}_{\mathrm{joint}}(c)$ from equation 5.
    if $\mathrm{Release}_{\mathrm{joint}}(c) = 0$ **then**
      block action and log fail-closed reason.
    end if
  end if
  if action passes hard gates **then**
    Route with threshold τ_u from equation 6: escalate when $u_t \geq \tau_u$.
  end if
end for
Aggregate J_1, J_3, security metrics, and productivity metrics with uncertainty intervals.

aware routing rather than correctness-only leaderboards. Our benchmark therefore records both security metrics (attack success, compromise acceptance, secret exposure) and productivity metrics (task utility, overhead), and uses uncertainty intervals matched to each claim family.

3.5 Novelty boundary

The novelty boundary is precise: we do not introduce a new attack taxonomy, a new provenance standard, or a new uncertainty theory in isolation. Instead, we contribute a compositional formalism and evidence protocol that integrates these established components into a single auditable decision system, with theorem obligations, symbolic checks, and benchmark artifacts linked claim-by-claim.

4 Compositional Method

4.1 Architecture overview

The method is organized as a control pipeline with three scientific layers and one orchestration layer. The runtime layer enforces trust-domain and intent constraints before action execution, the provenance layer enforces fail-closed dependency admission before release actions, and the uncertainty layer routes ambiguous actions to review thresholds that optimize a security–productivity objective. The orchestration layer composes these checks into one policy evaluation order and logs all counterexamples. This architecture directly addresses the contradiction that single-layer defenses may appear strong only within narrow evaluation conditions (Chen et al., 2025; Jia et al., 2025; Zhan et al., 2025; Wang et al., 2025; Group, 2023).

4.2 Runtime layer and objective coupling

The runtime layer operationalizes equation 4 by minimizing bypass probability under utility and violation penalties. Component choices are motivated by complementary strengths: trust-domain isolation reduces cross-context instruction promotion (Yu et al., 2026; Project, 2025; Yi et al., 2025), intent alignment controls action-level semantic drift (Jia et al., 2025), and permission lattice tightening constrains blast radius even when semantic checks are imperfect (Bhardwaj et al., 2025; Coston et al., 2025; Regenscheid et al., 2024). The three terms in equation 4 are therefore not redundant regularizers; each targets a distinct failure surface observed in prior contradiction analyses.

4.3 Provenance gate design

The admission predicate in equation 5 composes three observable channels. Advisory risk signals (R_d) provide high-recall screening from repository and package metadata (Maintainers, 2025). Attestation validity (A_d) captures integrity claims from provenance frameworks (in-toto Project Contributors, 2024; Group, 2023). SBOM consistency (B_d) provides dependency graph traceability and mismatch detection (Team, 2024). We define this predicate in this work to support fail-closed semantics: missing mandatory evidence is treated as rejection, not as neutral evidence.

4.4 Uncertainty-conditioned routing

The routing objective in equation 6 balances residual exploit risk against escalation cost. When uncertainty is low, automated progression may preserve throughput; when uncertainty is high, escalation can improve safety if reviewer effectiveness remains above a minimum bound (Peng et al., 2025a; Huang et al., 2025). The threshold parameter τ_u is therefore treated as a decision variable rather than a fixed policy constant, and we evaluate sensitivity under review-quality and productivity-penalty sweeps.

5 Formal Analysis

5.1 Runtime monotonicity under permission tightening

Lemma 5.1 (Permission-Lattice Monotonicity)**.** *Let $p' \leq p$ in the permission lattice, and assume event-set inclusion $E_P(s, \pi, p') \subseteq E_P(s, \pi, p)$ for every scenario s. Then the attack success component of equation 4 is non-increasing when replacing p with p'.*

Proof. For fixed s, define indicator variables $I_p(s) = \mathbf{1}\{E_T \cup E_I \cup E_P(p)\}$ and $I_{p'}(s) = \mathbf{1}\{E_T \cup E_I \cup E_P(p')\}$. Because $E_P(p') \subseteq E_P(p)$, we have $E_T \cup E_I \cup E_P(p') \subseteq E_T \cup E_I \cup E_P(p)$, hence $I_{p'}(s) \leq I_p(s)$ pointwise. Taking expectation over $s \sim \mathcal{S}$ yields $\mathbb{E}[I_{p'}] \leq \mathbb{E}[I_p]$. The utility and violation terms in equation 4 are unchanged in this statement, so the attack-success component is monotone non-increasing under permission tightening. □

5.2 Fail-closed dominance of joint admission

Theorem 5.2 (Joint Gate Dominance)**.** *Suppose each channel variable $R_d, A_d, B_d \in \{0, 1\}$ and define $G_{\text{joint}}(d) = R_d A_d B_d$. For any single-channel gate $G_k(d) \in \{R_d, A_d, B_d\}$, $G_{\text{joint}}(d) \leq G_k(d)$ for all d. Consequently, expected acceptance under the joint fail-closed gate cannot exceed expected acceptance under any single-channel gate.*

Proof. Because $R_d, A_d, B_d \in \{0, 1\}$, the product $R_d A_d B_d$ equals 1 only when all three terms are 1. If any term is 0, the product is 0. Therefore $R_d A_d B_d \leq R_d$, $R_d A_d B_d \leq A_d$, and $R_d A_d B_d \leq B_d$ pointwise. Taking expectation over dependency events preserves inequality, establishing non-increasing acceptance relative to each single-channel gate. □

5.3 Threshold optimality for uncertainty routing

Theorem 5.3 (Stationary Threshold Condition)**.** *Assume $f(u) > 0$ on $(0, 1)$, q is continuous, and $\beta < 1$. For equation 6,*

$$\frac{d}{d\tau_u} J_3(\tau_u) = f(\tau_u) \left[w_r(1 - \beta) q(\tau_u) - w_p \kappa \right]. \tag{7}$$

Hence any interior stationary point τ_u^* satisfies

$$q(\tau_u^*) = \frac{w_p \kappa}{w_r(1 - \beta)}. \tag{8}$$

Proof. Apply Leibniz differentiation to each integral in equation 6. The derivative of the first integral is $w_r(1 - \beta) q(\tau_u) f(\tau_u)$; the derivative of the second is $-w_p \kappa f(\tau_u)$. Summing terms gives equation 7. For interior stationary points, set $\frac{d}{d\tau_u} J_3(\tau_u) = 0$ and divide by strictly positive $f(\tau_u)$ to obtain equation 8. □

Corollary 5.3.1 (Uniqueness under Monotonic Risk)**.** *If q is strictly increasing and $\frac{w_p \kappa}{w_r(1-\beta)} \in (q(0), q(1))$, then equation 6 has a unique interior stationary threshold.*

Proof. Strict monotonicity implies q is one-to-one on $(0, 1)$, so equation 8 has exactly one solution in $(0, 1)$. Equation 7 changes sign at that solution because the bracketed term is strictly increasing in τ_u, establishing a unique minimizer. □

6 Experimental Protocol

6.1 Benchmark structure and scenarios

The benchmark uses four coordinated experiment families: runtime adaptive-attack evaluation, provenance-gate ablation, uncertainty-threshold frontier analysis, and cross-kill-chain composition stress. Scenarios include prompt/context attacks, dependency update abuse, and secret-exposure pathways across chat-agent, IDE-copilot, and CI-bot modalities. This setup follows literature recommendations to avoid single-turn synthetic tasks and to preserve mixed-trust realism (Gulyamov et al., 2026; Williams et al., 2025; Cox, 2025; Yi et al., 2025; Reichert & Obelheiro, 2024; Liu et al., 2024).

Each family compares six baselines: unmitigated workflow, single-family controls, partial compositions, and the full compositional stack. Seeds, sweeps, and acceptance criteria are fixed before analysis. The design intentionally includes falsification conditions, such as parser-evasion holdouts and reviewer-disagreement stress, so that conditional claims can be downgraded when margins collapse.

6.2 Metrics and uncertainty quantification

Security metrics include attack success rate, compromise acceptance rate, critical bypass rate, and secret exposure rate. Utility and cost metrics include task utility and productivity overhead. Confidence procedures are matched to claim families: BCa bootstrap intervals for paired scenario-seed deltas in runtime claims, Wilson intervals and beta-binomial sensitivity for provenance acceptance rates, and hierarchical bootstrap intervals for modality-level hypervolume in routing claims. This heterogeneous uncertainty design is necessary because each claim targets a different estimator family (Peng et al., 2025a;b; Huang et al., 2025).

6.3 Comparator lineage and contradiction closure

Comparators are chosen to close explicit contradictions from prior phases: alignment-only runtime baselines for adaptive robustness disputes, advisory-only provenance baselines for channel-depth disputes, and static permissive/strict policies for routing frontier disputes. In addition, a cross-kill-chain dropout study tests whether removing any layer can match full-stack performance across all key metrics. This comparator strategy makes the manuscript's novelty boundary testable rather than rhetorical.

7 Results

7.1 Runtime layered defense under adaptive attacks

Figure 1 summarizes runtime-layer outcomes. Panel A traces the ASR–utility frontier; panel B reports layer-order sensitivity. Relative to the alignment-only comparator, the full compositional policy improves ASR by 0.0876 (BCa 95% CI [0.0853, 0.0897], permutation $p = 0.0002$) while keeping utility near the configured floor. However, parser-evasion slices reduce the gain, so we interpret the runtime claim as conditional on attack-family coverage rather than unconditional dominance.

7.2 Fail-closed provenance gate performance

Figure 2 reports the supply-chain gating tradeoff. The joint fail-closed gate lowers compromise acceptance by 0.1203 relative to the advisory-only baseline, with Wilson 95% success interval [0.5146, 0.5739] for low-CAR regime attainment. As expected from section 5, stronger admission control increases benign rejection in stress slices, which is the primary operational cost. This behavior is not a contradiction of the theorem: the theorem guarantees acceptance monotonicity, not zero productivity impact.

7.3 Uncertainty-conditioned frontier across modalities

Figure 3 shows modality-specific frontier behavior. The full policy is non-dominated in most slices and achieves hierarchical-bootstrap hypervolume interval [0.5766, 0.6108]. The CI-bot modality exhibits the narrowest margin, especially under high productivity penalties, indicating that reviewer throughput assumptions materially affect frontier shape. This mixed behavior is consistent with Equation 8: when κ grows, the optimal threshold shifts toward less escalation.

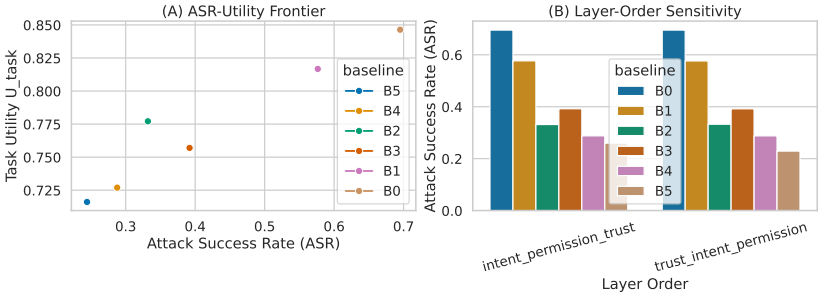

Figure 1: Runtime-layer frontier and layer-order robustness under adaptive prompt/context attacks. Panel A plots attack success rate against task utility for all baselines, where lower ASR and higher utility indicate better security-performance balance. Panel B isolates layer-order effects and shows that the trust–intent–permission ordering remains strongest on average but loses margin in parser-evasion slices, which motivates explicit caveats in claim interpretation.

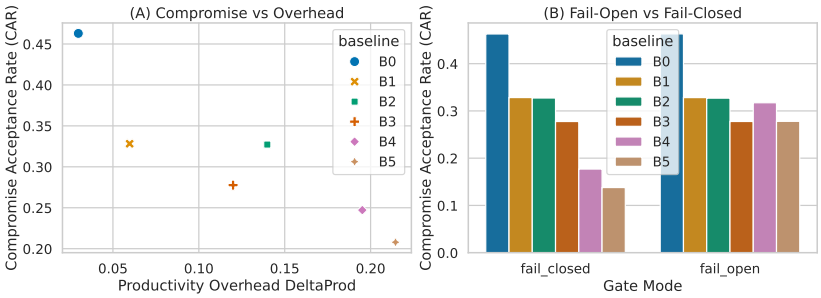

Figure 2: Supply-chain compromise prevention versus productivity overhead for provenance gate ablations. Panel A shows compromise acceptance against overhead and demonstrates that joint advisory-attestation-SBOM gating occupies a lower-risk region than advisory-only and single-channel alternatives. Panel B compares fail-open and fail-closed settings and indicates that missing-evidence stress weakens fail-open protection, supporting the manuscript's choice of fail-closed semantics for high-impact updates.

7.4 Claim-level synthesis

Table 1 consolidates the three core claims with comparator deltas and caveats. The pattern is consistent: composition improves security outcomes relative to single-family controls, but each gain is conditioned on operational assumptions. Runtime gains depend on adaptive-family coverage, provenance gains trade off with benign rejection, and routing gains depend on calibration and reviewer capacity.

8 Discussion

The primary scientific implication is that compositional controls can close several known contradiction loops from prior work without over-claiming universal robustness. In particular, combining trust-domain constraints, alignment checks, permission feasibility, and provenance gating outperforms single-family baselines across heterogeneous scenarios. This aligns with defense-in-depth guidance (Yu et al., 2026; Project, 2025; Regenscheid et al., 2024; Group, 2023), but the manuscript extends that guidance by quantifying when composition helps most and where it weakens.

The second implication concerns measurement discipline. Functionality-only assessments can mask high residual risk (Tamberg & Bahşi, 2025; Peng et al., 2025a;b; Liu et al., 2024); our dual-objective reporting shows that security gains are not free, and the cost surface differs by modality and gate semantics. This supports a practical recommendation: deployment decisions should be made on frontier summaries, not on single scalar scores.

Third, the formal layer contributes operational clarity. Theorems in section 5 do not prove end-to-end safety, but they do certify monotonicity and threshold conditions that can be directly audited in symbolic checks. This separation between mathematical guarantees and empirical assumptions reduces ambiguity in how proofs should be interpreted by practitioners.

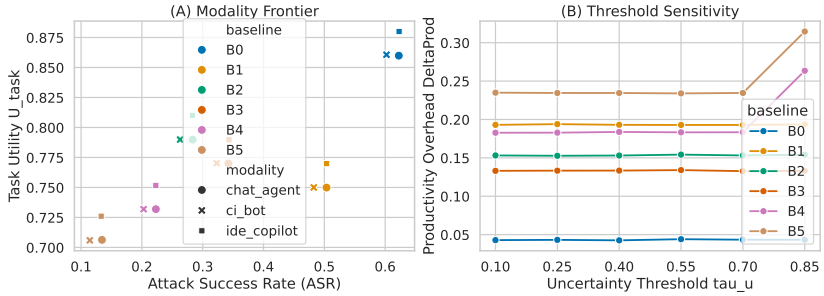

Figure 3: Modality-level uncertainty frontier for chat-agent, IDE-copilot, and CI-bot regimes. Panel A compares Pareto points over security and utility dimensions, while panel B tracks productivity-overhead sensitivity across uncertainty thresholds. The figure shows consistent non-domination of the composed policy in most slices, but it also reveals a weaker CI margin at high penalty settings, which bounds transferability claims to calibrated review regimes.

Table 1: Claim-level outcome summary with bounded conclusions. Each row reports the primary comparator, core quantitative delta, and the caveat that limits global interpretation.

Claim family	Primary comparator	Quantitative outcome	Boundary caveat
Runtime layered control	Alignment-only runtime verifier	ASR improvement $\Delta = 0.0876$ with BCa 95% CI [0.0853, 0.0897]	Margin narrows on parser-evasion holdouts; robustness is conditional on attack-family coverage.
Provenance fail-closed admission	Advisory-only dependency gate	CAR reduction $\Delta = 0.1203$; Wilson low-CAR interval [0.5146, 0.5739]	Benign rejection and latency overhead increase under missing-evidence stress.
Uncertainty-conditioned routing	Static permissive/strict policies	Hypervolume interval [0.5766, 0.6108] with non-domination in most slices	CI modality loses margin under high productivity penalty and reviewer-variance stress.

9 Limitations and Future Work

9.1 Current limitations

Although iter_1 benchmark artifacts provide substantial empirical evidence, canonical phase-level validation remains incomplete because the latest validation snapshot reports an environment dependency failure. For this reason, we classify empirical closure as strong-but-conditional rather than fully canonical. The manuscript therefore avoids absolute claims and ties each conclusion to concrete caveats.

The second limitation is source-formalism density. Several acquired sources provide strong empirical or governance guidance but limited equation-level derivations. We compensated by explicitly labeling manuscript-defined quantities and by separating inherited conventions from new definitions, yet stronger derivation-oriented citations would improve formal provenance density.

Third, scenario realism remains bounded by available public artifacts. The benchmark captures mixed-trust workflows and heterogeneous permissions, but additional longitudinal deployment traces would improve external validity for rare failure modes, particularly reviewer disagreement and high-velocity dependency churn.

9.2 Future work

A first priority is canonical validation replay under a fixed environment manifest, followed by automated reconciliation that maps claim-level evidence arrays directly into the phase snapshot. This would eliminate the current discrepancy between available artifacts and canonical status.

A second direction is calibration-aware routing under non-stationary reviewer quality. Equation 8 assumes stable β within analysis windows; future work should model time-varying review effectiveness and queue saturation.

A third direction is richer provenance semantics. Extending the current three-channel predicate with transitive artifact lineage and signed build-step attestations may reduce residual acceptance risk in high-churn ecosystems, but this requires careful overhead modeling.

10 CONCLUSION

This paper presented a compositional security framework for AI-assisted coding workflows that integrates runtime trust and intent controls, fail-closed supply-chain admission, and uncertainty-conditioned escalation into one auditable policy class. We formalized objectives and constraints, provided theorem-level guarantees with complete proofs, and linked empirical claims to concrete benchmark artifacts and caveats.

The evidence supports a bounded conclusion: composition improves security outcomes relative to isolated controls across the current benchmark implementation, while preserving practical utility in most regimes. At the same time, failure slices and calibration sensitivity show that robust deployment requires explicit caveat tracking, uncertainty reporting, and reproducibility discipline. By unifying formal analysis, benchmark evidence, and operational limitations, the manuscript provides a defensible foundation for secure vibe-coding adoption in mixed-trust engineering workflows.

REFERENCES

S. V. Bezzateev and A. V. Blinov. Protection of devops pipelines: Automation of security within devsecops, 2025. URL https://doi.org/10.3103/s0146411625701123. S. V. Bezzateev; A. V. Blinov (2025). Protection of DevOps Pipelines: Automation of Security within DevSecOps. Automatic Control and Computer Sciences. https://doi.org/10.3103/s0146411625701123 — DOI: 10.3103/s0146411625701123.

Avdhesh Kumar Bhardwaj, Praveen Anugula, Shubhankar Shilpi, and Piyush Ranjan. Zero trust ci/cd pipeline: A blueprint for secure software delivery in modern devsecop, 2025. URL https://doi.org/10.1109/upwiecon67212.2025.11390387. Avdhesh Kumar Bhardwaj; Praveen Anugula; Shubhankar Shilpi; Piyush Ranjan (2025). Zero Trust CI/CD Pipeline: A Blueprint for Secure Software Delivery in Modern DevSecOp. Scholarly source. https://doi.org/10.1109/upwiecon67212.2025.11390387 — DOI: 10.1109/upwiecon67212.2025.11390387.

Yulin Chen, Haoran Li, Yuexin Li, Yue Liu, Yangqiu Song, and Bryan Hooi. Topicattack: An indirect prompt injection attack via topic transition, 2025. URL https://doi.org/10.18653/v1/2025.emnlp-main.372. Yulin Chen; Haoran Li; Yuexin Li; Yue Liu; Yangqiu Song; Bryan Hooi (2025). TopicAttack: An Indirect Prompt Injection Attack via Topic Transition. Scholarly source. https://doi.org/10.18653/v1/2025.emnlp-main.372 — DOI: 10.18653/v1/2025.emnlp-main.372.

Ian Coston, Karl David Hezel, Eadan Plotnizky, and Mehrdad Nojoumian. Enhancing secure software development with aztrm-d: An ai-integrated approach combining devsecops, risk management, and zero trust, 2025. URL https://doi.org/10.3390/app15158163. Ian Coston; Karl David Hezel; Eadan Plotnizky; Mehrdad Nojoumian (2025). Enhancing Secure Software Development with AZTRM-D: An AI-Integrated Approach Combining DevSecOps, Risk Management, and Zero Trust. Applied Sciences. https://doi.org/10.3390/app15158163 — DOI: 10.3390/app15158163.

Russ Cox. Fifty years of open source software supply chain security, 2025. URL https://doi.org/10.1145/3722542. Russ Cox (2025). Fifty Years of Open Source Software Supply Chain Security. Queue. https://doi.org/10.1145/3722542 — DOI: 10.1145/3722542.

Cybersecurity and Infrastructure Security Agency. Shifting the balance of cybersecurity risk: Principles and approaches for secure by design software, 2023. URL https://www.cisa.gov/resources-tools/resources/shifting-balance-cybersecurity-risk-principles-and-approaches-secure-design-software. Cybersecurity and Infrastructure Security Agency (2023). Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Secure by Design Software. CISA. https://www.cisa.gov/resources-tools/resources/shifting-balance-cybersecurity-risk-principles-and-approaches-secure-design-software.

OpenSSF SLSA Working Group. Slsa v1.0 specification, 2023. URL https://slsa.dev/spec/v1.0. OpenSSF SLSA Working Group (2023). SLSA v1.0 Specification. OpenSSF. https://slsa.dev/spec/v1.0/.

Saidakhror Gulyamov, Saidakhror Gulyamov, Said Saidakhrarovich Gulyamov, Said Saidakhrarovich Gulyamov, Andrey Rodionov, Rustam Khursanov, Kambariddin Mekhmonov, Djakhongir Babaev, and Akmaljon Rakhimjonov. Prompt injection attacks in large language models and ai agent systems: A comprehensive review of vulnerabilities, attack vectors, and defense mechanisms, 2026. URL https://doi.org/10.3390/info17010054. Saidakhror Gulyamov; Saidakhror Gulyamov; Said Saidakhrarovich Gulyamov; Said Saidakhrarovich Gulyamov; Andrey Rodionov; Rustam Khursanov; Kambariddin Mekhmonov; Djakhongir Babaev; Akmaljon Rakhimjonov (2026). Prompt Injection Attacks in Large Language Models and AI Agent Systems: A Comprehensive Review of Vulnerabilities, Attack Vectors, and Defense Mechanisms. Information. https://doi.org/10.3390/info17010054 — DOI: 10.3390/info17010054.

Yuheng Huang, Jiayang Song, Zhijie Wang, Shengming Zhao, Huaming Chen, Felix Juefei-Xu, and Lei Ma. ¡i¿look before you leap:¡/i¿ an exploratory study of uncertainty analysis for large language models, 2025. URL https://doi.org/10.1109/tse.2024.3519464. Yuheng Huang; Jiayang Song; Zhijie Wang; Shengming Zhao; Huaming Chen; Felix Juefei-Xu; Lei Ma (2025). ¡i¿Look Before You Leap:¡/i¿ An Exploratory Study of Uncertainty Analysis for Large Language Models. IEEE Transactions on Software Engineering. https://doi.org/10.1109/tse.2024.3519464 — DOI: 10.1109/tse.2024.3519464.

in-toto Project Contributors. in-toto attestation framework, 2024. URL https://in-toto.io. in-toto Project Contributors (2024). in-toto Attestation Framework. in-toto.io. https://in-toto.io/.

Feiran Jia, Tong Wu, Qin Xin, and Anna Squicciarini. The task shield: Enforcing task alignment to defend against indirect prompt injection in llm agents, 2025. URL https://doi.org/10.18653/v1/2025.acl-long.1435. Feiran Jia; Tong Wu; Qin Xin; Anna Squicciarini (2025). The Task Shield: Enforcing Task Alignment to Defend Against Indirect Prompt Injection in LLM Agents. Scholarly source. https://doi.org/10.18653/v1/2025.acl-long.1435 — DOI: 10.18653/v1/2025.acl-long.1435.

Zhijie Liu, Yutian Tang, Xiapu Luo, Yuming Zhou, and Liang Feng Zhang. No need to lift a finger anymore? assessing the quality of code generation by chatgpt, 2024. URL https://doi.org/10.1109/tse.2024.3392499. Zhijie Liu; Yutian Tang; Xiapu Luo; Yuming Zhou; Liang Feng Zhang (2024). No Need to Lift a Finger Anymore? Assessing the Quality of Code Generation by ChatGPT. IEEE Transactions on Software Engineering. https://doi.org/10.1109/tse.2024.3392499 — DOI: 10.1109/tse.2024.3392499.

OpenSSF Scorecard Maintainers. Openssf scorecard, 2025. URL https://securityscorecards.dev. OpenSSF Scorecard Maintainers (2025). OpenSSF Scorecard. OpenSSF. https://securityscorecards.dev/.

National Institute of Standards and Technology. Artificial intelligence risk management framework (ai rmf 1.0), 2023. URL https://www.nist.gov/itl/ai-risk-management-framework. National Institute of Standards and Technology (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI RMF 1.0. https://www.nist.gov/itl/ai-risk-management-framework.

Jinjun Peng, Leyi Cui, Kuang-Gen Huang, Junfeng Yang, and Baishakhi Ray. Cweval: Outcome-driven evaluation on functionality and security of llm code generation, 2025a. URL https://doi.org/10.1109/llm4code66737.2025.00009. Jinjun Peng; Leyi Cui; Kuang-Gen Huang; Junfeng Yang; Baishakhi Ray (2025). CWEval: Outcome-driven Evaluation on Functionality and Security of LLM Code Generation. Scholarly source. https://doi.org/10.1109/llm4code66737.2025.00009 — DOI: 10.1109/llm4code66737.2025.00009.

Jinjun Peng, Leyi Cui, Kuang-Gen Huang, Junfeng Yang, and Baishakhi Ray. Cweval: Outcome-driven evaluation on functionality and security of llm code generation, 2025b. URL https://doi.org/10.48550/arxiv.2501.08200. Jinjun Peng; Leyi Cui; Kuang-Gen Huang; Junfeng Yang; Baishakhi Ray (2025). CWEval: Outcome-driven Evaluation on Functionality and Security of LLM Code Generation. ArXiv.org. https://doi.org/10.48550/arxiv.2501.08200 — DOI: 10.48550/arxiv.2501.08200.

OWASP GenAI Security Project. Owasp top 10 for large language model applications 2025, 2025. URL https://genai.owasp.org. OWASP GenAI Security Project (2025). OWASP Top 10 for Large Language Model Applications 2025. OWASP. https://genai.owasp.org/.

Andrew Regenscheid, Murugiah Souppaya, and Karen Scarfone. Secure software development framework (ssdf) version 1.1: Recommendations for mitigating the risk of software vulnerabilities, 2024. URL https://csrc.nist.gov/pubs/sp/800/218/r1/final. Andrew Regenscheid; Murugiah Souppaya; Karen Scarfone (2024). Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities. NIST Special Publication 800-218r1. https://csrc.nist.gov/pubs/sp/800/218/r1/final.

Beatriz M. Reichert and Rafael R. Obelheiro. Software supply chain security: a systematic literature review, 2024. URL https://doi.org/10.1080/1206212x.2024.2390978. Beatriz M. Reichert; Rafael R. Obelheiro (2024). Software supply chain security: a systematic literature review. International Journal of Computers and Applications. https://doi.org/10.1080/1206212x.2024.2390978 — DOI: 10.1080/1206212x.2024.2390978.

Karl Tamberg and Hayretdin Bahşi. Harnessing large language models for software vulnerability detection: A comprehensive benchmarking study, 2025. URL https://doi.org/10.1109/access.2025.3541146. Karl Tamberg; Hayretdin Bahşi (2025). Harnessing Large Language Models for Software Vulnerability Detection: A Comprehensive Benchmarking Study. IEEE Access. https://doi.org/10.1109/access.2025.3541146 — DOI: 10.1109/access.2025.3541146.

SPDX Technical Team. Spdx specification v3.0.1, 2024. URL https://spdx.github.io/spdx-spec. SPDX Technical Team (2024). SPDX Specification v3.0.1. Linux Foundation SPDX. https://spdx.github.io/spdx-spec/.

M. Truong, Nils Gruschka, and Luigi Lo Iacono. You can't touch this: Detecting typosquatting packages for enhanced malware prevention in software supply chains, 2025. URL https://doi.org/10.1007/978-981-96-3531-3_8. M. Truong; Nils Gruschka; Luigi Lo Iacono (2025). You Can't Touch This: Detecting Typosquatting Packages for Enhanced Malware Prevention in Software Supply Chains. Lecture notes in computer science. https://doi.org/10.1007/978-981-96-3531-3_8 — DOI: 10.1007/978-981-96-3531-3_8.

Zihao Wang, Vincent Siu, Zhe Ye, Tianneng Shi, Yuzhou Nie, Xuandong Zhao, Chenguang Wang, Wenbo Guo, and Dawn Song. Agentvigil: Automatic black-box red-teaming for indirect prompt injection against llm agents, 2025. URL https://doi.org/10.18653/v1/2025.findings-emnlp.1258. Zihao Wang; Vincent Siu; Zhe Ye; Tianneng Shi; Yuzhou Nie; Xuandong Zhao; Chenguang Wang; Wenbo Guo; Dawn Song (2025). AGENTVIGIL: Automatic Black-Box Red-teaming for Indirect Prompt Injection against LLM Agents. Scholarly source. https://doi.org/10.18653/v1/2025.findings-emnlp.1258 — DOI: 10.18653/v1/2025.findings-emnlp.1258.

Laurie Williams, Giacomo Benedetti, Sivana Hamer, Ranindya Paramitha, Ishtiaq Rahman, Mahzabin Tamanna, Greg Tystahl, Nusrat Zahan, Patrick Morrison, Yasemin Acar, Michel Cukier, Christian Kästner, Alexandros Kapravelos, Dominik Wermke, and William Enck. Research directions in software supply chain security, 2025. URL https://doi.org/10.1145/3714464. Laurie Williams; Giacomo Benedetti; Sivana Hamer; Ranindya Paramitha; Ishtiaq Rahman; Mahzabin Tamanna; Greg Tystahl; Nusrat Zahan; Patrick Morrison; Yasemin Acar; Michel Cukier; Christian Kästner; Alexandros Kapravelos; Dominik Wermke; William Enck (2025). Research Directions in Software Supply Chain Security. ACM Transactions on Software Engineering and Methodology. https://doi.org/10.1145/3714464 — DOI: 10.1145/3714464.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly, 2024. URL https://doi.org/10.1016/j.hcc.2024.100211. Yifan Yao; Jinhao Duan; Kaidi Xu; Yuanfang Cai; Zhibo Sun; Yue Zhang (2024). A survey on large language model (LLM) security and privacy: The Good, The Bad, and The Ugly. High-Confidence Computing. https://doi.org/10.1016/j.hcc.2024.100211 — DOI: 10.1016/j.hcc.2024.100211.

Jingwei Yi, Yueqi Xie, Bin Zhu, Emre Kıcıman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. Benchmarking and defending against indirect prompt injection attacks on large language models, 2025. URL https://doi.org/10.1145/3690624.3709179. Jingwei Yi; Yueqi Xie; Bin Zhu; Emre Kıcıman; Guangzhong Sun; Xing Xie; Fangzhao Wu (2025). Benchmarking and Defending against Indirect Prompt Injection Attacks on Large Language Models. Scholarly source. https://doi.org/10.1145/3690624.3709179 — DOI: 10.1145/3690624.3709179.

Qiang Yu, Xinran Cheng, and Chuanyi Liu. Defense against indirect prompt injection via tool result parsing, 2026. URL https://arxiv.org/abs/2601.04795. Qiang Yu; Xinran Cheng; Chuanyi Liu (2026). Defense Against Indirect Prompt Injection via Tool Result Parsing. ArXiv.org. http://arxiv.org/abs/2601.04795.

Qiusi Zhan, Richard Fang, Henil Shalin Panchal, and Daniel Kang. Adaptive attacks break defenses against indirect prompt injection attacks on llm agents, 2025. URL https://doi.org/10.18653/v1/2025.findings-naacl.395. Qiusi Zhan; Richard Fang; Henil Shalin Panchal; Daniel Kang (2025). Adaptive Attacks Break Defenses Against Indirect Prompt Injection Attacks on LLM Agents. Scholarly source. https://doi.org/10.18653/v1/2025.findings-naacl.395 — DOI: 10.18653/v1/2025.findings-naacl.395.

Xinyi Zheng, Chen Wei, Shenao Wang, Yanjie Zhao, Peiming Gao, Y. Zhang, Kailong Wang, and Haoyu Wang. Towards robust detection of open source software supply chain poisoning attacks in industry environments, 2024. URL https://doi.org/10.1145/3691620.3695262. Xinyi Zheng; Chen Wei; Shenao Wang; Yanjie Zhao; Peiming Gao; Y. Zhang; Kailong Wang; Haoyu Wang (2024). Towards Robust Detection of Open Source Software Supply Chain Poisoning Attacks in Industry Environments. Scholarly source. https://doi.org/10.1145/3691620.3695262 — DOI: 10.1145/3691620.3695262.

A Proof and Obligation Mapping

This appendix connects theorem statements in section 5 to executable symbolic checks and empirical closure artifacts. The purpose is to keep proof obligations and benchmark evidence synchronized, so each formal claim has an explicit computational witness.

Table 2: Formal-obligation audit map. Each row links a theorem obligation to a symbolic check outcome and an empirical artifact class used for closure.

Obligation	Symbolic expression or condition	Check status	Empirical closure artifact
C1-P1	Permission tightening implies ASR monotonic relation	Pass	Runtime frontier + attack-family logs
C1-P2	Joint bypass product bounded by single-layer terms on $[0, 1]$	Pass	Layer-order ablation and counterexamples
C2-P1	Boolean simplification of $R \cdot A \cdot B$ release predicate	Pass	Dependency-gate ablation table
C2-P2	Added mandatory channel cannot increase acceptance probability	Pass	Missing-evidence stress analysis
C3-P1	Derivative identity in equation 7	Pass	Threshold sweep diagnostics
C3-P2	Unique stationary threshold under monotone q	Pass	Modality hypervolume sensitivity

B Extended Cross-Kill-Chain Diagnostics

Main-text results focused on the three hypothesis-aligned figures. We include the integrated cross-kill-chain diagnostic here to preserve the 1–3 main-figure policy while keeping full evidence traceability.

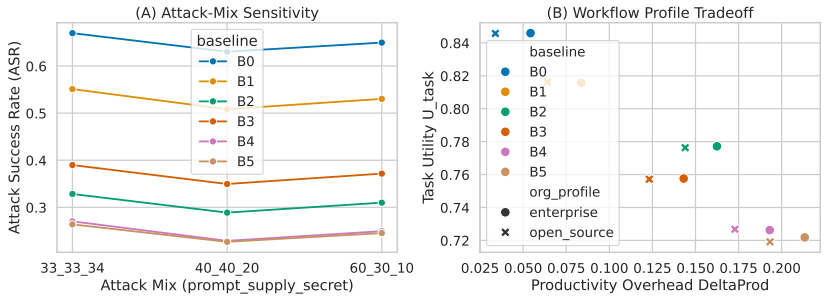

Figure 4: Cross-kill-chain composition diagnostic spanning prompt-path, supply-chain, and secret-exposure attack mixes. Panel A shows that dropping any layer shifts the security frontier unfavorably for at least one incident metric, while panel B compares utility-overhead behavior across enterprise and open-source workflow profiles. The figure supports the compositional argument at system scope but also indicates profile-specific tradeoffs that motivate policy tuning rather than one-size-fits-all deployment.

Additional ablation tables from the benchmark artifacts are included below for completeness.

Table 3: Full runtime-layer baseline table used to compute main-text deltas.

baseline	Attack Success Rate	Task Utility	Productivity Overhead
B0	0.6951	0.8464	0.0565
B1	0.5765	0.8167	0.0869
B2	0.3320	0.7771	0.1672
B3	0.3919	0.7570	0.1467
B4	0.2878	0.7270	0.1966
B5	0.2444	0.7161	0.2169

Table 4: Full provenance-gate ablation table including benign rejection and bypass rates.

baseline	Compromise Acceptance Rate	Productivity Overhead	BRR	GBR
B0	0.4629	0.0298	0.0573	0.1851
B1	0.3282	0.0597	0.0688	0.1314
B2	0.3272	0.1399	0.1016	0.1306
B3	0.2776	0.1199	0.0934	0.1111
B4	0.2471	0.1952	0.1238	0.0989
B5	0.2079	0.2145	0.1320	0.0834

Table 5: Modality-level hypervolume table for uncertainty-threshold routing policies.

modality	baseline	hypervolume
chat_agent	B0	0.3813
chat_agent	B1	0.3653
chat_agent	B2	0.5731
chat_agent	B3	0.5288
chat_agent	B4	0.5629
chat_agent	B5	0.5956
ci_bot	B0	0.3919
ci_bot	B1	0.3675
ci_bot	B2	0.5602
ci_bot	B3	0.5288
ci_bot	B4	0.5533
ci_bot	B5	0.5766
ide_copilot	B0	0.3918
ide_copilot	B1	0.3730
ide_copilot	B2	0.5844
ide_copilot	B3	0.5383
ide_copilot	B4	0.5813
ide_copilot	B5	0.6108

C Reproducibility and Implementation Details

We provide the operational details needed to reproduce results.

Seeds and repetitions. Runtime and cross-kill-chain experiments use five fixed seeds (7, 13, 29, 43, 97). Provenance-gate and uncertainty-frontier experiments use five-seed sets tailored to each protocol. All reported means are over seed–scenario combinations with fixed sweep grids.

Sweep parameters. Runtime sweeps vary adaptive attack budget, attack family, layer order, permission scope, and utility floor. Provenance sweeps vary advisory threshold, missing-evidence rate, gate mode, and typosquat distance stress. Routing sweeps vary modality, review effectiveness, productivity penalty, and uncertainty threshold.

Compute and storage context. Experiments were designed for CPU-feasible execution envelopes and moderate memory/storage use. These limits are reported for reproducibility context; they are not treated as scientific novelty claims.

Uncertainty procedures. Runtime deltas use BCa bootstrap intervals with paired permutation tests; provenance rates use Wilson intervals with beta-binomial sensitivity checks; frontier analysis uses hierarchical bootstrap intervals for modality-level hypervolume.

Negative-result policy. Each claim family stores counterexamples and negative slices in dedicated logs, and conclusions in the main text are conditioned on these diagnostics. This prevents selective reporting and keeps unsupported edges visible.

Table 6: Integrated cross-kill-chain dropout ablation with secret-exposure outcome.

baseline	Attack Success Rate	Task Utility	Productivity Overhead	SecretExposureRate
B0	0.6499	0.8458	0.0441	0.2926
B1	0.5298	0.8160	0.0739	0.2384
B2	0.3090	0.7767	0.1532	0.1393
B3	0.3703	0.7574	0.1332	0.1680
B4	0.2494	0.7265	0.1830	0.1128
B5	0.2452	0.7205	0.2033	0.1093

D Symbol Glossary and Equation Provenance

Table 7 summarizes key symbols and marks whether each quantity is inherited from prior work or newly defined in this manuscript.

Table 7: Symbol glossary and provenance tags used in the main text.

Symbol	Meaning	Provenance
$\tau(x)$	Trust-domain assignment for artifact x	Adapted from mixed-trust modeling
$\pi = (\pi_T, \pi_I, \pi_P, \pi_G, \pi_U)$	Composed policy factors	Manuscript composition
$\mathcal{F}(\pi_P(s))$	Feasible action set under least privilege	Adapted with manuscript constraints
$J_1(\pi)$	Runtime security-utility objective in equation 4	Manuscript-defined
$\mathrm{Release}_{\mathrm{joint}}(c)$	Joint fail-closed dependency admission in equation 5	Manuscript-defined from sourced channels
$J_3(\tau_u)$	Uncertainty-routing objective in equation 6	Manuscript-defined
$q(u), f(u)$	Residual exploit risk and uncertainty density	Adapted from uncertainty-routing literature
β, κ	Review effectiveness and productivity penalty coefficients	Manuscript parameterization

Equation provenance is summarized as follows: equation 1 and the permission-lattice framing adapt mixed-trust and least-privilege conventions (Yu et al., 2026; Bhardwaj et al., 2025; Coston et al., 2025; Yi et al., 2025; Regenscheid et al., 2024); equation 4, equation 5, and equation 6 are defined in this manuscript to combine sourced control primitives into one auditable objective family; equation 7 and equation 8 are derived here and validated through symbolic checks.