pISSN:2349-5707 Volume:12, Issue 09, September-2025 eISSN:2349-5715

HOLISTIC FUSION OF CYBER THREAT INTELLIGENCE, SEMANTIC DEDUPLICATION, AND AI-DRIVEN AUTOMATION FOR PROACTIVE RISK MITIGATION IN DEVSECOPS PIPELINES

Dr. David L. Wong
Institute for Intelligence-Based Cybersecurity, University of New South Wales, Australia

ABSTRACT: Background: Modern software delivery pipelines increasingly demand that security operate at velocity without becoming a bottleneck; integrating cyber threat intelligence (CTI), automated compliance, and semantic deduplication into DevSecOps is posited as a path to reconcile speed with robust defense (Samtani et al., 2019; Malik, 2025).
Objective: This paper develops a theoretically grounded, practice-oriented framework that synthesizes CTI mining, semantic analysis for deduplication of tool-generated findings, and AI-enabled automation for compliance and privacy checking to enable proactive risk mitigation in continuous integration and continuous delivery (CI/CD) workflows (Sun et al., 2023; Gulraiz, n.d.; Amaral et al., 2021).
Methods: We perform an integrative conceptual synthesis rooted in the referenced literature, articulating method chains and mappings from intelligence sources through automated analytic pipelines into developer-facing remediation actions, while considering energy, audit, and operational constraints (Zhou et al., 2022; Limbrunner, 2023; Mohammed, 2023).
Results: The proposed framework operationalizes CTI into actionable policy artifacts that are continuously matched, semantically deduplicated, and automatically enforced or suggested in pre-production stages; it reconciles completeness and privacy checks with regulatory obligations and reduces alert fatigue through semantic consolidation (Sun et al., 2023; Gulraiz, n.d.; Amaral et al., 2021).
Conclusions: Integrating CTI with semantic deduplication and AI-enabled automation strengthens situational awareness and preemptive defense in DevSecOps, but imposes challenges in privacy, auditability, and energy footprint that require trade-offs and future empirical evaluation (Muikku, 2020; Limbrunner, 2023; Andrea Tang, 2019). This article maps the theoretical foundations, methodical steps, expected outcomes, limitations, and a research agenda for empirical validation.

Keywords: Dev SecOps, Cyber Threat Intelligence, Semantic Deduplication, AI Automation, CI/CD Energy, Compliance.

INTRODUCTION

The accelerating tempo of software delivery has created a dual imperative: to preserve rapid deployment cycles while embedding security and compliance as core functions of development lifecycles (Samtani et al., 2019; Malik, 2025). DevSecOps—security as code integrated into development, operations, and quality assurance—seeks to shift-left security tasks and automate controls; however, success hinges on resolving two persistent operational frictions. First, the flood of security tool outputs (static analysis, dynamic scans, dependency checks, container scans, and runtime monitors) produces noisy, duplicated, and semantically overlapping findings that overwhelm engineers and security teams, degrading responsiveness and elevating risk of missed remediation (Gulraiz, n.d.; Muikku, 2020). Second, external intelligence about active threats and vulnerability exploitation patterns—cyber threat intelligence (CTI)—is frequently disconnected from CI/CD pipelines, thereby failing to influence build-time decisions and preemptive risk mitigation (Sun et al., 2023; Zhou et al., 2022). These frictions produce a gap between detection and preventive action, undermining the promise of DevSecOps.

The literature suggests complementary remedies. Semantic analysis and deduplication algorithms can consolidate redundant findings and produce higher-signal alerts tailored for developers (Gulraiz, n.d.). CTI mining and structuring provide contextualized indicators of compromise and exploitation trends that, when operationalized, can inform prioritization and automated gating decisions (Sun et al., 2023; Zhou et al., 2022). AI-enabled automation facilitates completeness checking of privacy and compliance artifacts—such as privacy policies and consent flows—improving governance without manual review overhead (Amaral et al., 2021; Areo, 2021). Yet integrating these techniques at scale, and in a manner that respects regulatory constraints (e.g., GDPR) and energy-use considerations in CI/CD environments, has not been fully articulated in a single cohesive framework (Andrea Tang, 2019; Limbrunner, 2023).

This article fills that lacuna by synthesizing the referenced research into a unified, publication-ready conceptual framework: an adaptive pipeline that ingests CTI, applies semantic deduplication to tool outputs, leverages AI to automate compliance and privacy checks, and enacts pre-production remediation—thereby enabling a proactive security posture prior to code hitting production (Malik, 2025). The framework is grounded in the theoretical foundations and empirical lessons present in the provided literature and aims to be directly actionable for researchers and practitioners seeking to operationalize CTI and AI within DevSecOps.

METHODOLOGY

The research method is an integrative conceptual synthesis that draws on cross-disciplinary findings from CTI analytics, semantic deduplication of security findings, AI-enabled policy checking, CI/CD energy modeling, and SOC audit practices. The methodology comprises four analytic steps: (1) systematic mapping of functional components and information flows from the cited literature; (2) definition of computational primitives and semantic operations required to deduplicate and contextualize tool outputs; (3) design of an automation orchestration model that couples CTI-derived artifacts with policy and compliance engines; and (4) development of evaluative criteria and anticipated metrics grounded in the literature for assessing effectiveness, efficiency, and operational trade-offs.

Step 1—Mapping functional components and information flows. We identify core nodes required to integrate CTI into DevSecOps: CTI ingestion and normalization, semantic enrichment, triage and prioritization, automated compliance checking, CI/CD gating and remediation orchestration, and situational awareness dashboards for SOC teams (Sun et al., 2023; Zhou et al., 2022; Muikku, 2020). Each node is characterized by its inputs, outputs, and transformation functions as described in existing works—CTI mining techniques to extract Indicators of Compromise (IoCs) and tactics, techniques, and procedures (TTPs) (Sun et al., 2023); semantic analysis approaches for deduplication (Gulraiz, n.d.); and AI models for policy completeness checks (Amaral et al., 2021).

Step 2—Defining computational primitives and semantic operations. To operationalize deduplication and context mapping we define primitives such as canonicalization (normalizing identifiers across tools), semantic similarity scoring (measuring overlap across textual and structured outputs), clustering (grouping related findings), and lineage mapping (tracking the origin of detections across scanning tools) (Gulraiz, n.d.; Muikku, 2020). These primitives are informed by natural language processing techniques and classical information retrieval strategies adapted to security artifacts. For example, semantic similarity scoring must accommodate domain-specific vocabularies (vulnerability IDs, stack traces, package names) and should incorporate CTI signals (e.g., exploitation reports) as external context (Sun et al., 2023).

Step 3—Designing an automation orchestration model. We synthesize orchestration requirements by aligning CTI artifacts with compliance and policy engines to create actionable rules that can be enforced at

build gates or recommended to developers. This model includes rule generation (from CTI to policy), priority mapping (mapping CTI severity to build-level risk thresholds), and remediation automation (automated patching suggestions, PR templates, or pipeline halts) as described in integrative CTI literature and DevSecOps automation research (Zhou et al., 2022; Malik, 2025; Amaral et al., 2021). The orchestration model is reflective—rules are updated by feedback loops from SOC audits and runtime telemetry (Mohammed, 2023; Muikku, 2020).

Step 4—Developing evaluative criteria and metrics. We propose metrics to evaluate fidelity (precision/recall of deduplication), impact (mean time to remediate pre-production vs. post-production), operational cost (CI/CD energy consumption impact), and governance compliance (audit completeness metrics). These criteria are grounded in the literature: deduplication effectiveness relates to alert fatigue reduction (Gulraiz, n.d.), remediation impact aligns with the CTI literature on preemptive defense (Sun et al., 2023; Malik, 2025), and energy considerations are drawn from CI/CD energy modeling work (Limbrunner, 2023). SOC audit literature informs the governance and documentation metrics (Mohammed, 2023).

Throughout, the methodology emphasizes transparent, auditable automation—ensuring actions taken by AI are traceable for compliance and SOC auditing (Areo, 2021; Mohammed, 2023; Andrea Tang, 2019). The methodology is intentionally descriptive and practical, describing the computational steps rather than presenting empirical experiments, because the task is the generation of a robust, theory-driven framework ready for empirical implementation and evaluation by practitioners and researchers.

RESULTS

This section presents the conceptual outcome: the Adaptive CTI–Semantic Deduplication–AI Automation (ACSA) framework. Results are descriptive and articulated across functional layers: CTI ingestion and contextualization, semantic deduplication of tool outputs, AI-enabled compliance and privacy verification, orchestration and enforcement, and evaluative outcomes.

CTI ingestion and contextualization. The framework prescribes a CTI ingestion layer that consumes structured and unstructured intelligence sources—open-source feeds, vendor threat libraries, internal telemetry, and reports—normalizes them into a common schema (e.g., STIX-like constructs), and enriches findings with meta-attributes such as observed exploitation timelines and targeted platforms (Sun et al., 2023; Zhou et al., 2022). Enrichment includes mapping IoCs to package and configuration artifacts commonly present in CI/CD artifacts so that intelligence can be evaluated at build-time: for example, mapping a reported exploit targeting a specific library version to the set of builds that declare that version (Sun et al., 2023). The result is a time-tagged asset-threat mapping that can be queried by pipeline gates.

Semantic deduplication of tool outputs. Security tools produce overlapping and sometimes conflicting findings. The framework prescribes a semantic deduplication pipeline that first canonicalizes identifiers (e.g., normalizing vulnerability identifiers, package names, and file paths), then computes semantic similarity scores between findings using embeddings and rule-based ontologies that respect security-specific terms (Gulraiz, n.d.). Findings scoring above a similarity threshold are clustered, producing a consolidated finding with aggregated severity, evidence provenance (which tools contributed), and CTI relevance (whether external intelligence corroborates or escalates the cluster). This consolidation reduces alert fatigue by collapsing redundant findings into a single developer action item and by surfacing CTI corroboration as a priority multiplier (Gulraiz, n.d.; Sun et al., 2023).

AI-enabled compliance and privacy verification. The framework integrates AI models trained for

completeness checking of privacy policies and compliance documents, operationalizing techniques similar to those proposed for privacy policy completeness (Amaral et al., 2021). AI models analyze manifest files, configuration values, and policy text to detect missing controls, inconsistent declarations, or policy gaps (e.g., misaligned data retention statements vs. actual storage behaviors). Where regulatory constraints require human sign-off (e.g., GDPR-sensitive processing), the framework flags audit artifacts and generates human-readable summaries. The integration respects "privacy-by-design" principles and supports automated remediation suggestions such as code snippets to enforce encryption-at-rest or to sanitize telemetry.

Orchestration and enforcement. The orchestration layer translates consolidated findings and CTI mappings into executable pipeline policies. Rules can take three operational modes: block (halt the build), warn (allow build but generate actionable tickets/PR templates), or auto-remediate (apply safe, reversible fixes such as dependency upgrades within defined policy constraints) (Malik, 2025; Areo, 2021). The choice of mode depends on mapped risk severity, CTI corroboration, and business-defined risk appetite (Marshall et al., 2019). Orchestration yields a closed-loop process: once a remedial action is taken (automatically or by a developer), the system records the action, updates the deduplication state, and feeds the result back into the CTI correlation engine to refine prioritization heuristics.

Evaluative outcomes and expected benefits. The ACSA framework yields multiple practical outcomes: reduction in duplicated findings and associated triage time by transforming many low-level alerts into fewer consolidated, CTI-prioritized tasks (Gulraiz, n.d.); earlier remediation of exploitable issues because CTI-driven gating moves defense earlier in the delivery pipeline (Sun et al., 2023; Malik, 2025); improved compliance signal through automated policy checking, reducing manual audit effort and enabling more deterministic SOC audit trails (Amaral et al., 2021; Mohammed, 2023); and better alignment between SOC detection and developer remediation through shared, auditable artifacts (Muikku, 2020). The framework anticipates trade-offs: some automated remediations may require governance oversight for GDPR-covered data flows, and enforcing strict block policies could increase developer friction if not calibrated (Andrea Tang, 2019; Marshall et al., 2019).

DISCUSSION

Interpretation of the framework. The ACSA framework represents a synthesis of CTI operationalization, semantic consolidation, and AI-driven governance that aims to shift security from reactive detection to proactive mitigation within DevSecOps. From a theoretical standpoint, the framework operationalizes the insight that intelligence is only as valuable as its integration into decision points (Sun et al., 2023). By mapping CTI to specific build artifacts and coupling it with deduplicated, enriched findings, the system effectively converts strategic intelligence into tactical actions that can be executed before production exposure (Zhou et al., 2022; Malik, 2025).

The semantic deduplication component addresses a crucial human factors problem. Security teams and developers suffer from cognitive overload when the same underlying issue appears in multiple forms across tools (Gulraiz, n.d.). Consolidation must therefore be more than mechanical de-duplication; it should include semantic reasoning about code context, configuration nuances, and exploitability—leveraging domain-aware ontologies and embeddings to ensure that clusters represent meaningful developer actions rather than artificially merged artifacts (Gulraiz, n.d.). This nuanced approach improves signal-to-noise ratio and empowers SOCs and engineering teams to focus on high-leverage fixes (Muikku, 2020).

AI-enabled compliance introduces both opportunities and constraints. Automated completeness checking yields significant efficiency gains, enabling continuous governance of privacy and regulatory artifacts, and

producing machine-readable audit trails (Amaral et al., 2021). However, the legal and ethical dimension—particularly GDPR compliance—demands that automation be explainable and human-supervised where processing involves personal data (Andrea Tang, 2019). The framework therefore emphasizes explainable AI and audit logs that document decision rationales, model inputs, and remediation steps (Areo, 2021; Andrea Tang, 2019). This approach balances automation with legal accountability.

Operational trade-offs and energy implications. Integrating sophisticated semantic and AI capabilities into CI/CD pipelines raises non-trivial operational costs and energy considerations. Continuous scanning, model inference, and repeated deduplication across frequent builds can increase energy consumption—an often overlooked but important operational burden (Limbrunner, 2023). The framework recommends dynamic scaling strategies and macro-to-micro energy calculations to balance detection fidelity against environmental and cost constraints, for instance by scheduling heavier analyses during off-peak times or by using incremental delta-analysis that focuses computation on changed artifacts (Limbrunner, 2023). Embedding energy-aware policies into orchestration helps organizations meet sustainability goals while maintaining security.

Governance, auditability, and SOC alignment. For the framework to be audit-ready, every automated decision must be recorded with provenance—detailing CTI sources, deduplication rationale, AI model versioning, and remediation steps (Mohammed, 2023; Areo, 2021). This level of documentation supports SOC audits and improves trust between security, legal, and engineering stakeholders. SOC audits also provide feedback for model tuning: audit findings can reveal false negatives/positives that inform retraining cycles, improving both detection and deduplication fidelity over time (Mohammed, 2023).

Limitations and counter-arguments. The framework's conceptual nature is both a strength and a limitation. While it integrates heterogeneous research strands into a coherent pipeline, empirical validation remains necessary to quantify benefits and surface unanticipated operational barriers. For instance, CTI quality varies widely; low-fidelity intelligence can misprioritize and generate false alarms when mapped into pipelines (Sun et al., 2023). Semantic deduplication models may conflate distinct vulnerability contexts if ontologies are insufficiently granular, leading to under-remediation (Gulraiz, n.d.). AI-enabled compliance models risk misclassification when trained on biased or incomplete policy corpora, potentially generating dangerous automation that misrepresents regulatory obligations (Amaral et al., 2021; Andrea Tang, 2019). Moreover, different organizations possess varying risk appetites—overly aggressive auto-remediation could impede time-to-market for critical features, while overly permissive thresholds may fail to prevent exploitation (Marshall et al., 2019).

Practical roadmap and deployment considerations. Implementing ACSA requires staged adoption. Early adopters should prioritize non-blocking modes—implementing deduplication and CTI enrichment to generate developer-facing priority lists and auto-generated remediation PR templates. Subsequent phases can introduce gating and auto-remediation for low-risk updates (e.g., non-breaking dependency upgrades) where revert mechanisms are safe. Continuous evaluation should monitor metrics such as pre-production remediation rate, post-production incident frequency, deduplication precision, mean time to remediate, energy overhead per build, and audit completeness (Gulraiz, n.d.; Limbrunner, 2023; Mohammed, 2023).

Research agenda. Empirical studies must test key hypotheses derived from the framework: (1) CTI-informed pre-production gating reduces the incidence of exploited vulnerabilities post-deployment compared to CTI-unaware pipelines (Sun et al., 2023; Malik, 2025); (2) semantic deduplication significantly reduces triage time while preserving detection recall (Gulraiz, n.d.); (3) AI-enabled compliance reduces manual audit time without increasing regulatory non-compliance risk when explainability and human oversight are enforced (Amaral et al., 2021; Areo, 2021); and (4) energy-aware

orchestration strategies can maintain detection performance while reducing the CI/CD energy footprint (Limbrunner, 2023). Each hypothesis requires field experiments or controlled deployments with instrumentation to measure outcomes.

Ethical considerations. Embedding CTI and AI into pipelines has ethical dimensions. CTI sources may contain intelligence that implicates individuals or entities; handling such data requires privacy-sensitive pipelines and compliance with legal obligations (Sun et al., 2023). AI-driven policy actions must avoid automated decisions that materially affect users without human review, especially in privacy-sensitive contexts (Andrea Tang, 2019). The framework emphasizes human-in-the-loop oversight for high-impact decisions and mandates documentation to ensure transparency.

CONCLUSION

This article presents the Adaptive CTI–Semantic Deduplication–AI Automation (ACSA) framework to bridge the gap between external cyber threat intelligence and actionable, automated pre-production security in DevSecOps. By combining CTI ingestion and enrichment, domain-aware semantic deduplication of tool outputs, and AI-enabled compliance verification, ACSA enables preemptive mitigation actions that reduce post-deployment risk, lower triage burden, and improve auditability. The framework is attentive to governance, explainability, and energy implications, recommending staged deployment and rigorous empirical evaluation to validate its benefits and to tune operational trade-offs. Future work must empirically test the framework's core claims in diverse organizational contexts, quantify energy and cost impacts, and refine AI models for robustness, fairness, and explainability. With careful implementation and oversight, integrating CTI and semantic automation into DevSecOps can materially strengthen organizational resilience in an era of rapid software delivery and evolving cyber threats.

REFERENCES

1. Gulraiz, A. Semantic Analysis for Deduplication of Security Findings in DevOps Security Tool Reports.

2. Samtani, S., Abate, M., Benjamin, V., & Li, W. (2019). Cybersecurity as an industry: A cyber threat intelligence perspective. In The Palgrave Handbook of International Cybercrime and Cyberdeviance (pp. 1-20). Palgrave Macmillan, Cham.

3. Sun, N., Ding, M., Jiang, J., Xu, W., Mo, X., Tai, Y., & Zhang, J. (2023). Cyber threat intelligence mining for proactive cybersecurity defense: A survey and new perspectives. IEEE Communications Surveys and Tutorials, 25(3), 1748-1774.

4. Zhou, Y., Tang, Y., Yi, M., Xi, C., & Lu, H. (2022). CTI view: APT threat intelligence analysis system. Security and Communication Networks, 2022(1), 9875199.

5. Areo, G. (2021). Automating Compliance in Healthcare IT: Essential Tools and Techniques.

6. Amaral, O., Abualhaija, S., Torre, D., Sabetzadeh, M., & Briand, L. C. (2021). AI-enabled automation for completeness checking of privacy policies. IEEE Transactions on Software Engineering, 48(11), 4647–4674.

7. Andrea Tang, F. I. P. (2019). Making AI GDPR Compliant.

8. Kumar, A. (2019). The convergence of predictive analytics in driving business intelligence and

enhancing DevOps efficiency. International Journal of Computational Engineering and Management, 6(6), 118-142. Retrieved from https://ijcem.in/wp-content/uploads/THE-CONVERGENCE-OF-PREDICTIVEANALYTICS-IN-DRIVING-BUSINESS-INTELLIGENCE-AND-ENHANCING-DEVOPSEFFICIENCY.pdf

9. Limbrunner, N. (2023). Dynamic macro to micro scale calculation of energy consumption in CI/CD pipelines.

10. Marshall, A., Ojiako, U., & Chipulu, M. (2019). A futility, perversity and jeopardy critique of "risk appetite". International Journal of Organizational Analysis, 27(1), 51-73.

11. Mohammed, A. (2023). SOC Audits in Action: Best Practices for Strengthening Threat Detection and Ensuring Compliance. Baltic Journal of Engineering and Technology, 2(1), 62-69.

12. Moore, J. H., Ribeiro, P. H., Matsumoto, N., & Saini, A. K. (2023). Genetic programming as an innovation engine for automated machine learning: The tree-based pipeline optimization tool (TPOT). In Handbook of Evolutionary Machine Learning (pp. 439-455). Singapore: Springer Nature Singapore.

13. Malik, G. (2025). Integrating Threat Intelligence with DevSecOps: Automating Risk Mitigation before Code Hits Production. Utilitas Mathematica, 122(2), 309-340.

14. Muikku, J. M. (2020). Improving Cyber Security Situational Awareness with Log and Network Security Monitoring.

15. Muscarello, G. (2023). Dynamic sharing of resources between different Kubernetes clusters (Doctoral dissertation, Politecnico di Torino).