International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.63282/3050-9246/ ICRTCSIT-108
Eureka Vision Publication | ICRTCSIT'25-Conference Proceeding

Original Article

From Bugs to Breaches: How Agentic AI Protects the Software Supply Chain

Vedika Saravanan
Independent Researcher, Texas, USA.

Abstract - Modern software supply chains have become highly sophisticated, prospecting vulnerabilities at all points of the development and deployment lifecycle. From CI/CD pipelines to open-source components, security incidents frequently stem less from insufficient information than from inability to respond quickly and discerningly. Advances in Agentic Artificial Intelligence (AI) like autonomous reasoning, planning, and acting systems are transforming the ways in which software supply chains can be made more secure. Differing from passive anomaly discovery approaches using traditional machine learning, agentic AI systems actively discern, rank, and correct threats at the code, dependencies, and infrastructure configuration levels. This paper provides a systematic review of new Research and industry practice applying agentic AI to software protection. The paper explores primary uses for agentic AI in secure code analysis, open-source risk management, and incident response autonomy, while underscoring the technological, ethical, and governing troubles intrinsic to autonomy within security frameworks. Lastly, the paper sketches out future directions and outlines a conceptual roadmap for infusing agentic AI into DevSecOps ecosystems for the realization of proactive, resilient, self-healing, and robust protection for software.

Keywords - Agentic Artificial Intelligence (AI); Software Security; DevSecOps; Supply Chain Security; Secure Software Development Lifecycle (SDLC); Autonomous Agents; Large Language Models (LLMs); Vulnerability Detection; Code Remediation; Multi-Agent Systems; Secure-by-Design; Open Source Software (OSS) Security; Threat Intelligence Automation; AI-Driven Security; Cybersecurity Automation.

1. Introduction

1.1. Background and Motivation

The modern software world is constructed on top of a mighty web of interdependent dependencies, open-source bundles, and self-appraising delivery pipelines. This interdependency has sped up innovations but widened security threats throughout the whole supply chain for software. Every new dependence, a container image, or a transport pipeline integration expands the attack area accessible for adversaries.

Among the most infamous instances of a software supply-chain attack is the 2020 SolarWinds attack [1], which involved attackers inserting malicious code into a legitimate update program, penetrating thousands of governments and business enterprises. Following attacks like the Log4Shell vulnerability [2] and the XZ Utils backdoor [3], it has become clear that both open-source and proprietary ecosystems have the same vulnerability. These attacks highlight a key problem of the modern DevSecOps era: a single breached component has the power to compromise a whole grouping of dependent systems.

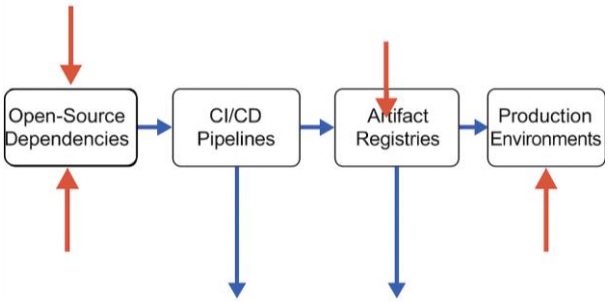

Figure 1. Software Supply-Chain Attack Surface

The overview of supply-chain vulnerabilities is shown in Fig. 1, displaying the connected stages of a contemporary software supply chain from open-source dependencies and continuous-integration pipelines to build artifacts and production environment. Each stage offers potential entry points where attackers can insert malware, inject bad data into third party dependencies, manipulate build artifacts, or take advantage of misconfigurations. The red arrows indicate how an original compromise in one layer truly propagates through downstream components, illuminating the systemic vulnerability that defines modern software ecosystems. This visualization highlights why detection-only traditional defense is inadequate and why adaptive, reasoning-driven protection mechanisms become imperative.

Traditional software-security tools, like Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), and dependency scanners, are still irreplaceable [9]–[11]. Yet, they are mostly reactive and don't have the contextual analysis needed to respond dynamically to changing threats. They detect vulnerabilities but don't automatically evaluate severity, develop mitigations, or automatically make corrections without some level of human supervision. Because of this, security teams are placed at greater expense to sort through high volumes of alerts and make remediation decisionsa task that is often too sluggish for the fast-paced threat landscape of the modern era.

1.2. Emergence of Agentic Artificial Intelligence

Latest developments in Agentic Artificial Intelligence (AI) bring into view the potential of moving from reactive detection to proactive and autonomous protection. Agentic AI systems are engineered to reason with fine-grained goals, plan with multiple steps, and take them adaptively with feedback [4]. Statically, these systems disagree with traditional machine-learning models, such that they passively detect features, but dynamically, they actively interact with their surroundings through reasoning, memory, and action loops [5].

In software security, such systems can independently detect, prioritize, and fix vulnerabilities in code, dependencies, and infrastructure settings. They can also automate security triage across distributed systems, minimizing human involvement in triage, patching, and mitigating. Industrial titans have also started investigating related architectures: Google Sec-PaLM [6], Microsoft Security Copilot [7], and GitHub Copilot Security [8] demonstrate early implementations of autonomous or semi-agentic systems with real-time ability to detect vulnerabilities and respond. These represent developments in a larger directional shift in the industry, moving from hierarchic, centralized decision-making systems to integrated reasoning and action in repeated feedback loops as a form of automation.

1.3. Scope and Objectives

This paper offers a systematic literature review of the developing intersection between software supply chain security and agentic AI. It explores how agentic systems can enhance key areas such as:

- Secure code analysis – automated vulnerability detection and contextual remediation.
- Open-source risk management – dependency auditing, license compliance, and patch generation.
- Incident response automation – reasoning-driven triage and self-healing defense mechanisms.

The paper aims to (a) consolidate existing research and industrial applications, (b) highlight technical and ethical challenges including explainability, accountability, and governance, and (c) propose a conceptual roadmap for integrating agentic AI into DevSecOps systems to build proactive, resilient, and self-healing security infrastructure.

2. Understanding Agentic Artificial Intelligence (Ai)

2.1. Definition and Core Principles

Agentic Artificial Intelligence (AI) is a breakthrough in evolution from traditional machine-learning systems. It is the type of computational agents that can reason, plan, and act autonomously for well-defined ends with no persistent human control. While the old models of AI are mostly predictive, predicting inputs from outputs, they fail to achieve the formation of goals, perception of contexts, and self-adjustment. On the opposite side, agentic systems combine reasoning, memory, and action modules that make them plan, adapt, and learn in iterative cycles in highly dynamic settings [4].

The founding principles of agentic AI are as follows:

- Reasoning: data interpretation, drawing of cause-effect inferences, and making of contextual-informed decisions.
- Planning: the planning of multi-step strategies for desired ends.
- Action: implementing those strategies, inseparably combined with self-testing and feedback cycles for iterative improvement.

By these elements, agentic AI evolves from a static analytical device to an active decision-maker that can strive for long-term goals while constantly responding to environmental feedback.

2.2. Architecture and Cognitive Framework

The architecture of agentic AI can be viewed as a reasoning–planning–acting feedback loop, or a closed cognitive cycle. The typical structure has the following layers:

- **Perception Layer:** Acquires input signals such as code repositories, configuration states, system logs, or vulnerability reports.
- **Cognition Layer:** Utilizes reasoning, long-term memory, and planning modules for understanding context and detailing actions.
- **Action Layer:** Executes decisionse.g., sending scans, treating as remedies, or interfacing with other systemsbased on cognition output.

- **Feedback Loop:** Evaluates outcomes and adjusts future plans, enabling self-adjustment and learning from success or failure.

Research frameworks such as ReAct (Reason + Act) [5], LangChain, and AutoGPT demonstrate this architecture, merging natural-language reasoning with sequential task completion. These systems demonstrate how large language models (LLMs) can be embedded within goal-oriented agents that not only understand commands but also reason, plan, and act autonomously.

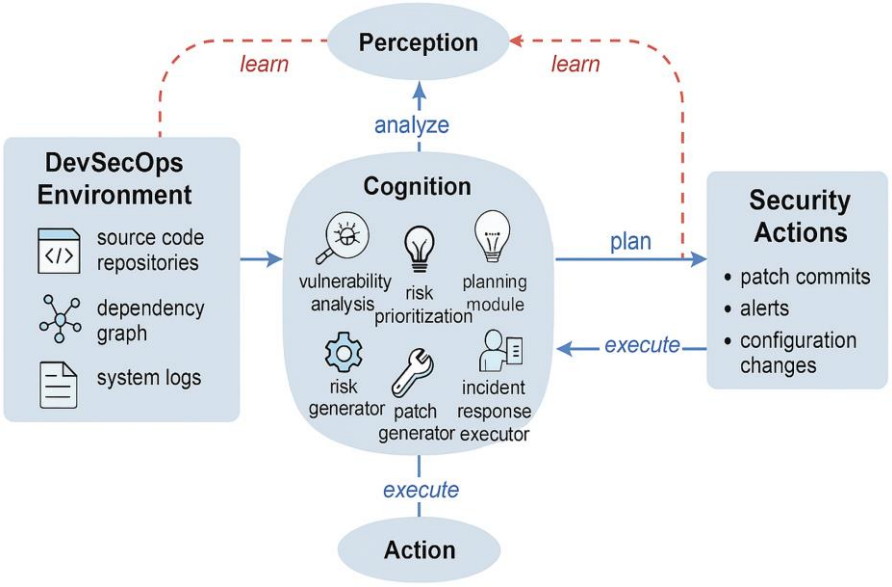

Figure 2. Architecture of an Agentic AI System

The architecture shown in Fig. 2. illustrates how agentic AI system works within the software supply chain's security system. The Perception layer communicates directly with the DevSecOps environment, collecting data from source-code repositories, dependency graphs, as well as system logs. The Cognition layer works on the data collected, with components for executing the task of vulnerability assessment, risk prioritization, planning, and reasoning from memory. Once a decision is reached, the Action layer executes actions such as development of patches, remediation automation, and incident handling.

The feedback loops indicated with red dashed arrows allow the agent to analyze the effects of its actions to improve future reasoning and planning iterations. This closed cognitive cycle allows the agent to continuously learn and develop the accuracy of its work and effort across iterations. In the boxes of external context, the agent interacts with its environment and the shared operational space. Inputs from the DevSecOps environment are streamed into perception, while Security Actions, such as patch commits, alerts, or change in configurations, are the physical outputs of the system. Together, the framework explains how the layers of perception, cognition, and action work together to transform agentic AI from a passive analytical model to an adaptive, self-improving cybersecurity agent.

2.3. Distinction from Conventional AI Approaches

In cybersecurity, traditional AI and machine-learning systems are largely reactive, they identify anomalies or classify risks based on static data. After a prediction or classification has been made, human analysts interpret the results and determine the next best course of action. There is no awareness of the long-term, nor can the AI systems autonomously adjust behavior based on additional information. Agentic AI, conversely, offers multi-step reasoning to determine its actions and make context-based decisions, with continuous adaptation. It therefore can autonomously:

- Prioritize vulnerabilities based on exploitability and context.
- Generate remediation strategies or code patches.
- Reassess its own decisions with post-action feedback.

In this way, closed-loop security automation becomes available by introducing the capability for an agent to identify threat and then make a plan, act, and to verify the outcome of the action and the conclusion. The result is an intelligent, adaptable defence mechanism that can change over time directly with the threat; far beyond detection-based systems.

3. Related Work

3.1. AI in Cybersecurity: A Historical Perspective

Artificial Intelligence has been applied in the field of cybersecurity for an extended amount of time, mainly for the purposes of pattern recognition, anomaly detection, and vulnerability prediction. In earlier days, studies applied supervised or

statistical models to detect intrusions and/or malware using manually engineered features and historical datasets. For example, it was one of the first uses of AI in cybersecurity to explore neural network based intrusion detection, clustering of network traffic, and machine learning assisted vulnerability discovery [12], [13].

As deep learning has developed, more robust frameworks for malware classification, phishing detection, and static code analysis have emerged. For example, convolutional and recurrent neural networks have been used to learn code semantics and extract exploitable code patterns [14]. These algorithms are an improvement over earlier techniques, but they remain static detectors without context, formal reasoning, or autonomous behavior.

3.2. Semi-Autonomous and Reinforcement-Learning Approaches

Autonomous security went beyond reinforcement-learning (RL) and adaptive decision-making systems. RL agents have played an important role in automating penetration testing, optimizing firewall rules and dynamically patching vulnerabilities [15], [16] and proved that machine agents are able to learn to protect while interacting with an environment.

All existing RL-based or semi-autonomous systems are focused on single-objective optimization and rely on pre-defined rewards and therefore lack the ability to reason at a symbolic level, plan over multiple steps, and coordinate across complex DevSecOps workflows. This called for hybrid approaches that can encompass learning, reasoning and planning and opened the way towards hybrid agentic architectures.

3.3. Large Language Models and Emerging Agentic Systems

The advent of Large Language Models (LLMs) has hastened the development of cognitive autonomy. For instance, ReAct (Reason + Act) [5], Lang Chain, and Auto GPT has demonstrated that LLMs can develop memory and reasoning while performing tasks sequentially. In areas like cybersecurity, these LLMs have been used for tasks such as code audits, triaging vulnerabilities, and producing reports in an automated way [17].

Several industry implementations (e.g Google Sec-PaLM [6], Microsoft Security Copilot [7], and GitHub Copilot Security [8]) exhibit early examples of agentic behavior pairing context awareness, prioritization, and recommending next actions using autonomy. Recently, academic researchers have explored multi-agent collaboration in things like dynamic threat-hunting and autonomous vulnerability management [18] but that work is still very initial and mostly human-supervised and does not include a closed feedback loop that would support reasoning based autonomy.

3.4. Gap and Motivation for This Study

While there have been strides toward automation, the majority of AI security utilities are still reactive and disjointed. They merely notice anomalies and take no action or take action without sufficient reasoning and contextual grasp. There is no targeted literature review on embedding agentic AI with perception, cognition, and action layers into the software supply chain for continuous adaptive protection. The present paper fills this gap by synthesizing a systematic literature enabled and based on research and industrial advances in agentic AI, situating them to practical DevSecOps problems and illustrated a pathway to implement self-learning and self-healing security architectures.

Table 1. Comparison of Costing and Cost Accounting

Study / Approach	Year	Methodology	Application Scope	Limitation
Neural Intrusion Detection [12]	2019	Supervised Deep Learning	Network anomaly detection	Lacks reasoning or autonomy
ML-assisted Vulnerability Discovery [13]	2024	Statistical Learning	Code vulnerability prediction	Reactive, no contextual understanding
Deep Neural Code Analysis [14]	2018	CNN/RNN	Malware and static code analysis	Limited explainability
RL-based Firewall Optimization [15]	2020	Reinforcement Learning	Policy tuning and response	Single-objective optimization
Autonomous Mitigation via RL [16]	2023	Adaptive RL Agents	Vulnerability remediation	No symbolic reasoning or feedback
LLM for Vulnerability Triage [17]	2025	LLM + Natural Language Reasoning	Code audit automation	No closed feedback loop
Google Sec-PaLM [6]	2023	Foundation Model	Threat intelligence and detection	Proprietary, limited transparency
Microsoft Security Copilot [7]	2024	Generative AI Agent	SOC response automation	Human supervision required
GitHub Copilot Security [8]	2024	LLM Code Assistant	Secure coding support	Not fully autonomous

Table I provides a summary of selected academic and industry initiatives in AI-based cybersecurity and considers their focus and limitations that motivate the proposal of agentic AI architectures for software-supply-chain defense.

4. Software Supply-Chain Security Landscape

4.1. Structure of the Modern Supply Chain

The software supply chain in the 21st century involves multiple interdependent stages: code development, the utilization of open-source dependencies, build and packaging, continuous-integration/continuous-deployment (CI/CD) pipelines, artifact storage, and finally, deployment into production. Each layer of the supply chain relies on both internal and third-party components that together comprise a very interdependent system. This configuration affords administration the ability to accomplish agility and scale; however, it also introduces multiple pathways for exploitation. Attackers increasingly target areas of weakness, such as package managers, build scripts, and configuration files, to obtain access to systems downstream. The incidents involving SolarWinds, Log4Shell, and XZ Utils remind us all that compromising one artifact (i.e., agent, code, etc.) can lead to total exposure of an entire system [1]–[3].

4.2. Key Threat Vectors

Supply-chain threats can be presented as threats across three general groups:

- Code-Level Threats, which include unauthorized manipulations of code, credential leaks, or dependency confusion in repositories;
- Build-Pipeline Threats include tampering with CI/CD automation scripts, should there be malicious packages in your distribution process, or tampering with the behavior of a compiler;
- Distribution- and Runtime Threats, where malicious binaries are trojanized in an artifact registry, container image, or deployment environment.

These threats aren't strictly segregated into phases- any compromise in one phase often creates a domino effect into the later phases, malicious code propagating to every other dependent service. The interdependencies that permeate our modern supply–chain ecosystems naturally limits trust and increase the extent to which even trivial vulnerabilities permeate systems and environments.

4.3. Existing Mitigation Practices

The existing practice emphasizes defense-in-depth utilizing layers of security controls such as the following:

- Code Signing and Integrity Checks: Avoids untrusted artifacts entering the pipeline.
- Vulnerability Scanning/Dependency Management: Applies SAST, DAST, and SCA tools to proactively identify known vulnerabilities prior to deployment.
- Zero-Trust Architecture: Prevents lateral movement by authentication between pipeline functions.
- Continuous Monitoring: Evaluates behavioral analytics and intrusion detection systems to identify deviations after deployment.

These, and other defenses, are effective to a degree, but primarily reactive as they rely on continuous human surveillance. There is no reasoning to context wrapping, or concert event recognition, and remediation orchestration in an autonomous or systematic order, which is a proclaimed operational gap Agentic AI intends to fill.

4.4. Challenges and Research Opportunities

The ongoing flaws in mitigation strategies we have:

- Complex Dependency Graphs. The sheer scale of open-source ecosystems makes true traceability infeasible.
- Alert Fatigue. An abundance of false positive alerts is more likely to slow analyst time toward solving real threats just as bad.
- Context Blindness. Static scanners cannot and will not infer whether a weak component matters toward the risk of another component.
- Slow Response Cycles. Regardless of whether triaged effectively, the slow nature of manual remediation might allow a window of opportunity to exploit.

All above challenges highlight a need for intelligent, closed-loop systems that can permute on vulnerabilities, predict compounding effects from any deployment, and autonomously execute countermeasures. The use of artificial intelligence with decision making capabilities at each layer of the supply chain, offers the capability to provide continuous protection against threat evolvement in real time.

4.5. Visual Model of Threat Propagation

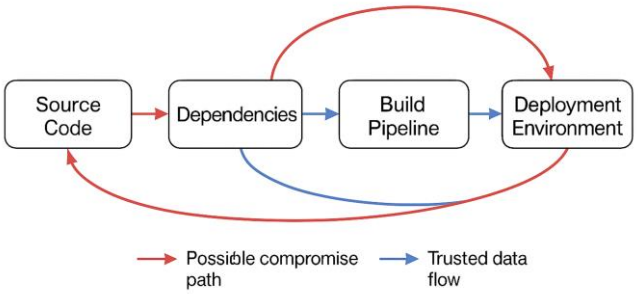

Figure 3. Threat Propagation Across the Software Supply Chain.

As shown in Fig. 3, modern supply chains form a continuous feedback loop that connects the development, build, and deployment environments. Each transition between environments, which is signified by blue data-flow arrows, represents both a functional dependency and potential exposure surface. Red arrows outline where compromises may proliferate. Once malicious code or a manipulated package breaches the build pipeline and produces artifacts, the artifacts can spread into registries and deployments downstream. Compromised components in runtime may also support the feedback loop, returning into the continuous integration/continuous delivery pipeline through compromised configuration files or credentials. These forms of interconnected propagation reiterate why perimeter-based security is obsolete and reasoning, agentic approaches are needed as a counterintuitive, integrated protection.

5. Agentic AI Applications For Supply-Chain Protection

5.1. Mapping Agentic AI to the Supply-Chain Ecosystem

Agentic Artificial Intelligence is able to be mapped directly onto the components of the software supply chain, corresponding perception, cognition, and action to existing DevSecOps functions.

- **Perception Layer:** The perception layer obtains information from source-code repositories, build logs, and dependency graphs to determine the presence of a deviation or risk signal. Using continuous data collection, the perception layer will identify early warning signals including any atypical committing patterns detected, package insertions or removals or any unacceptable deviations concerning pipeline events.
- **Cognition Layer:** reasons over aggregated context. It relates code-level conclusions to the current state of the rest of your infrastructure, evaluates the odds of a successful exploit, and ultimately advises the analyst on response options. This layer provides the analytic engine, merging threat-intelligence feeds, with historical telemetry to derive prioritization and risk scoring.
- **Action Layer:** acts autonomously to execute mitigations generate patches, roll back compromised builds, rotate credentials, or revoke access. This layer has a closed feedback loop, verifying each correction taken and update on subsequent actions.

These three layers together move from the linear software supply chain depicted in the depiction in Fig. 3 to a self-supervising, self-adapting defense network that has the capability to perceive the presence of a threat or risk as well as respond in real time.

Table 2. Mapping Agentic AI Layers to Supply-Chain Security Functions

Agentic Layer	Function	Supply-Chain Target	Example Operations	Expected Outcome
Perception	Continuous monitoring and data collection	Code repositories, dependencies	Detect anomalous commits, identify tampered packages	Early anomaly detection and awareness
Cognition	Contextual reasoning and prioritization	Build pipelines, CI/CD telemetry	Correlate vulnerabilities, plan mitigation strategies	Context-aware decision-making
Action	Autonomous execution and remediation	Artifact registry, deployment environment	Apply patches, rollbacks, credential rotation	Reduced mean-time-to-respond (MTTR)
Feedback Loop	Continuous learning and adaptation	Across DevSecOps systems	Evaluate outcomes, refine reasoning models	Progressive improvement and resilience

Table II aligns the perception-cognition-action structure of agentic AI with particular software-supply-chain stages to show the contribution of each layer to adaptive, autonomous defense.

5.2. Use Cases of Agentic AI in Supply-Chain Defense

Agentic AI is more than a theoretical concept; we are applying it across the software supply chain. The most immediate opportunity for application is autonomous vulnerability remediation, wherein the system automatically identifies vulnerabilities and resolves the risks it poses without waiting for human interaction. When a dependency or package is flagged as vulnerable or if there is a malicious update, the cognition layer will risk-rank and potentially identify patching pairs for the vulnerability and any compatibility issues. The action layer then responds and deploys the patch in a temporary, controlled environment for review. Upon implementation, a feedback loop will evaluate the effectiveness of the patch before announcement into the entire enterprise - thus infinitely reducing time to respond and human error.

Another important area of application is dynamic trust scoring. Rather than applying static security policies, agentic systems maintain an ongoing, real-time evaluation of the trustworthiness of packages, build artifacts, and external APIs for example. The cut scores fluctuate in real-time based on behavioral observation and contextual evaluation, and allow the system to automatically tighten or loosen to access controls. This approach provides evidence that trust has become a variable, data-driven attribute as opposed to a purely implicit one.

Agentic AI helps automate incident response by connecting perception and cognition layers to telemetry history. When anomalies are detected, the agent can run autonomously to correlate alerts, establish potential root causes, and initiate mitigations like container isolation, credential rotation, or access revocation. In regulated environments, the same reasoning capability is leveraged for continuous compliance, with the agent autonomously monitoring for configuration changes and dependency changes throughout the software supply chain and producing ready-to-audit evidence of compliance against standards such as ISO 27001 or NIST SP 800-53. Together, these use cases provide strong evidence that agentic AI is not intended to replace existing DevSecOps tools; rather, it will provide a cognitive layer that interprets situational context and delivers coherent self-directed responses.

5.3. Integration Model and Feedback Flow

As shown in Figure 4, agentic AI has a horizontal integration through every facet of the supply chain, creating a shared cognitive layer from development through deployment. The perception agents continuously monitor the entire DevSecOps environment, gathering telemetry data from code repositories, dependency graphs, build logs, and runtime analytics. The cognitive modules process the agency's input, reasoning, and planning, extracting linkage about vulnerabilities from the separate, and interdependent layers of the supply chain environment.

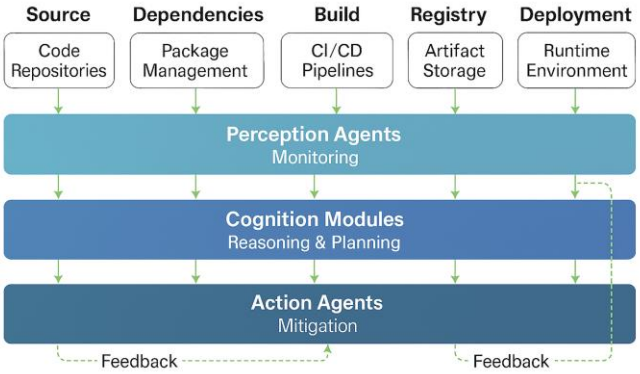

Figure 4. Agentic AI Integration across the Software Supply Chain.

After the cognition layer determines the appropriate course of action, the action-layer produces the appropriate mitigation response, whether it is to produce patches, rollback builds, or restrict deployments. The green feedback arrows in Figure 4 demonstrate how the results of action layers are returned, inside the cognitive or perception layers. Importantly, this is a closed-cycle refinement process not a closed-control cycle because both the success and failure can be understood and used for what training the entity to become more accurate to detect and decide in the future. Over time, this interaction transforms the supply chain from a static pipeline into a self-adapting ecosystem that can sustain, at an autonomous level, the secure integrity of the supply chain through ongoing adaptation to the changing threat landscape.

5.4. Benefits and Expected Outcomes

Integrating agentic AI into software-supply-chain defense provides discrete measurable operational benefits. The embedding of perception and cognition across the entire DevSecOps pipeline results in a significant reduction in the meantime to detect (MTTD) vulnerabilities. Automated planning and execution can further decrease the mean time to respond (MTTR) by patching or remediating without human-induced lead time before deployment. Creating efficiencies aside, cognitive reasoning offers increased accuracy helping to differentiate critical vulnerabilities from benign anomalies and decreasing alert fatigue.

More concretely, the framework increases operational resilience by maintaining protection under both component and team disruption. The feedback loop contributes to continuous learning and improvement with every mitigation effortsuccess or failure. This self-perpetuating capability allows an organization to progress from a reactive stance to a self-healing proactive posture, where systems can reason, act, and adapt, in the same loop. In essence, this results in a secure and resilient supply-chain ecosystem that adheres to both enterprise speed and the state of modern threats.

6. Challenges, Limitations, and Ethical Considerations

6.1. Technical Limitations and Systemic Complexity

Although agentic AI offers the potential for autonomous reasoning and adaptation, its actual use in large-scale DevSecOps environments face a number of physical limitations. One limitation is due to the complexity of the real-world supply chain because there are literally thousands of different dependencies, repositories, and microservices that are all interacting with one another at the same time. The work of perception, cognition, and the action layer across these types of distributed infrastructure demands high throughput orchestration and some level of observability to achieve diagnosis and contextual meaning. Timing delays when collecting data or synchronizing feedback can lead to the intended response being delayed or out of sync, which will generally lead to the failure of real-time protection.

Next, there are heterogeneous data types from unstructured code comments to structured telemetry data which can complicate assertions grounded in unified context reasoning. An agent that has been trained using truncated data sources or biased sets may classify benign behavior as malicious or miss out new attacks altogether. Furthermore, stabilizing the feedback loop might be a different challenge: too much self-correction can lead to oscillating or duplicate reactions. Balancing time and tenacity in a state of vigilance is an open technical research problem.

6.2. Explainability and Human Oversight

A common challenge in deploying intelligent systems for cybersecurity is explainability. As agentic AI becomes more autonomous, it becomes increasingly complex for human analysts to ascertain the specific actions performed and the potential reasoning sequence that was followed, as to whether the decisions are in accordance with organizational policies. This disconnect inhibits trust and can complicate incident investigations.

Explainable AI (XAI) techniques, such as understandable reasoning pathways, or open and transparent logs of the decisions made, need to be built in cognition modules from the beginning. A human-in-the-loop design will be essential in cases of uncertain confidence or expected high-impact remediation by the security analyst, with that analyst remaining involved in the decision-making process for a shared autonomy model. This can mitigate contributions to accountability while preserving system agility.

6.3. Ethical and Societal Implications

The independent behavior of agentic AI prompts ethical concerns about accountability, bias, or unintentional outcomes. Automated behaviors that change or disable infrastructure components may unintentionally disrupt and impact service level agreements. If these actions are rooted in bias or incomplete training data, impacted stakeholders may pay a disproportionately high price. Moreover, there are critical issues of data privacy and consent when agents are monitoring source repositories and developer behavior continuously. In the absence of governance arrangements, continuous monitoring behavior may interfere with ethical lines of distinction between security assurance and privacy intrusion and erosion of trust between developers and the organization. Therefore, adopting transparent governance frameworks providing clear accountability on what data agents may access, under what observation behaviors could be justified, and who ultimately bears accountability for poor behavioral judgements is a critical method of grounded ethical legitimacy.

6.4. Governance, Standardization, and Regulatory Outlook

The governance mechanisms for agentic AI in software security are still nascent. Most organizations do not have a formal policy framework for defining decision boundaries or escalation pathways for autonomous agents. There are also no recognized standards for audit or certification for AI-enabled security operations. It will be important to engage in collaborative efforts among academia, industry, and regulatory bodies to create common benchmarks, compliance certifications, and reporting structure for AI-driven cybersecurity systems. For example, regulatory frameworks such as the EU AI Act and NIST's AI Risk Management Framework could serve as foundational frameworks for defining responsible use and action as they develop. As such, the definitions and scope of responsible use models for regulated AI frameworks may set apart software supply chains that are achieved beyond the scope and luxury of an organization's DevSecOps practices.

6.5. Summary of Open Research Questions

Although significant progress has been made, some central research questions are still not settled relating to agentic AI and security in the software-supply-chain. The first of these concerns transparency and traceability of reasoning. As agents make autonomous and more complex decisions, we have to think about how their internal reasoning can be logged, visualized, and audited without an impact on performance. Current approaches to explainability for large models involve a limited form of

shallow attribution or a post hoc form of explanation that are not suitable for high-stakes security contexts.

A second open question related to agentic AI involves stability of feedback loops and efficiency of learning. Agentic systems need to balance adaptability with reliability, making sure that corrective actions do not devolve into oscillations and/or redundant interventions. The problem of measuring and optimizing this is still an open area of research and requires the formulation of new control-theoretic and reinforcement-learning methods for continuous DevSecOps contexts.

An another pertinent question involves supervising bias and fairness auditing. Training data for AI-based security tools will, to some extent, reflect biases that were predominant in the reporting of vulnerabilities or the use of code. Without explicitly auditing aspects of bias, agents are susceptible to over-emphasizing certain patterns of risk while discounting new or less prominent types of exploitation. A framework of automated fairness auditing could become a specified requirement for agents to be trustworthy.

From an organizational lens, the system of governance and accountability is another significant public domain. As autonomous systems make decisions that are security-critical, the respective responsibility of the operators, the organization, and the agent creates a legal and ethical obligation to delineate responsibility. Other standards like the thresholds for escalating incidents, required approval points, and the availability and accessibility of audit logs for traceability will be important for accountability and compliance.

Lastly, assessment and benchmarking remain unstandardized. Most assessment methods are focused on detecting on either some level of accuracy or latency and do not recognize or measure capabilities that are higher-order reasoning, contextual priorities, or long-term resilience. Establishing shared assessments with benchmark datasets for agentic AI for cybersecurity would help compare, replicate, and build upon the work of other researchers.

7. Conclusion and Future Directions

The advancement of agentic Artificial Intelligence (AI) with software supply chain security represents a critical advancement toward building autonomous, adaptive, and resilient digital ecosystems. After reviewing progress in the software supply chain in security capabilities from passive, rule-based detection tools to active, contextually-aware, and reasoned capabilities it has been documented how the perception–cognition–action framework of agentic AI can be applied to every part of the development lifecycle. This includes intelligent vulnerability detection, contextualizing value, and self-directed remediation.

The introduction of agentic AI changes the historical security model from reactive detection paradigm to proactive defense ecosystem. These agentic AI systems are capable of continuously perceiving signals in the environment, reasoning about intent and impact, and autonomously acting to mitigate threats and impact. The capabilities of agentic AI can decrease response latency that limits human-mediated workflows. In complex, interdependent supply chains, these capabilities are vital to continuing to establish software integrity at scale, and not just convenient.

Yet, as we have already touched upon, that is a daunting transformation from a standpoint of explainability and adaptability, ethical efficacy and accountability, and governance. Agentic systems must act intelligently, and also be able to explain their acts clearly, learn ethically, and operate within rapidly changing ethical and regulatory protocols. Therefore, the success of agentic AI systems in cyber security will largely depend on shared accountability between autonomous systems and the human operator, to augment trustworthiness through transparency and oversight.

In considering the future, there are several priority areas for research and engineering. First, it will be important to design benchmarks and performance metrics to evaluate agentic systems in cyber security which will standardize reasoning quality, efficacy of adaptation, and long term resilience. Second, hybrid human AI governance frameworks should be developed to enable autonomy in agentic systems, while also focusing on bringing agency behavior, within the perspective of the human operator, to the forefront of the control system–the layers of governance that offer reassurance to human operators that any agentic actions remain verifiable, reversible, and indistinct from human intent. A third important focus will be developing cross-domain learning architectures to support agentic systems, such as linking in supply-chain telemetry with vulnerability databases, threat intelligence, and operational observability, so that agents can infer to be more adaptive across different domains.

Finally, the aim for the next 10 years is a self-healing software-supply-chain ecosystem in which agentic AI learns from feedback and engages with human experts, all while evolving to counter new threats at machine speed. To make this aim a reality, we will need both technical innovation and multidisciplinary collaboration that will merge artificial intelligence, cybersecurity engineering, ethics, and governance. When used responsibly, agentic AI has the power to transform security, from the way it is conceptualized, enforced, and sustained through the entire landscape of digital infrastructure.

8. Conflicts of Interest

The author declares that there is no conflict of interest concerning the publication of this paper.

References

[1] CISA, "Emergency Directive 21-01: Mitigate SolarWinds Orion Code Compromise," U.S. Cybersecurity and Infrastructure Security Agency, Dec. 2020. [Link]

[2] R. Hiesgen, M. Nawrocki, T. C. Schmidt, and M. Wählisch, *"The Race to the Vulnerable: Measuring the Log4j Shell Incident,"* arXiv preprint arXiv:2205.02544, 2022. [Link]

[3] P. Przymus and T. Durieux, "Wolves in the Repository: A Software Engineering Analysis of the XZ Utils Supply Chain Attack," arXiv preprint arXiv:2504.17473, 2025. [Link]

[4] R. Sapkota, K. Roumeliotis, and M. Karkee, "AI Agents vs. Agentic AI: A Conceptual Taxonomy, Applications and Challenges," arXiv preprint arXiv:2505.10468, 2025. [Link]

[5] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing Reasoning and Acting in Language Models," in *Proc. ICLR 2023*, 2023. [Link]

[6] M. Wheatley, "Google Cloud bolsters cybersecurity with generative AI model Sec-PaLM," *SiliconANGLE*, Apr. 2023. [Link]

[7] Microsoft, "Microsoft Copilot for Security is generally available on April 1, 2024, with new capabilities," Microsoft Blog, Mar. 2024. [Link]

[8] GitHub, "GitHub for Beginners: Security best practices with GitHub Copilot," GitHub Blog, 2024. [Link]

[9] Y. Lyu, H. Kang, R. Widyasari, J. Lawall, D. Lo, "Evaluating SZZ Implementations: An Empirical Study on the Linux Kernel," *arXiv preprint arXiv:2308.05060*, 2023.. [Link]

[10] M. Esposito, V. Falaschi, and D. Falessi, "An Extensive Comparison of Static Application Security Testing Tools," *arXiv preprint arXiv:2403.09219*, 2024. [Link]

[11] L. Zhao, S. Chen, Z. Xu, C. Liu, L. Zhang, J. Wu, J. Sun, and Y. Liu, "Software Composition Analysis for Vulnerability Detection: An Empirical Study on Java Projects," in *Proc. ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '23)*, 2023, 13 pp. [Link]

[12] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman , "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, 2019, Art. 20. [Link]

[13] S. B. Chafjiri, *et al.*, "Vulnerability detection through machine learning-based fuzzing: A systematic review," *Computers & Security*, 2024. [Link]

[14] Z. Li, D. Zou, S. Xu, X. Ou, H. Jin, S. Wang, Z. Deng, and Y. Zhong, "VulDeePecker: A Deep Learning-Based System for Vulnerability Detection," in *Proc. NDSS*, 2018. [Link]

[15] Z. Hu, R. Beuran, and Y. Tan, *"Automated Penetration Testing Using Deep Reinforcement Learning,"* in Proceedings of the IEEE European Symposium on Security and Privacy Workshops (EuroS&P Workshops), 2020. [Link]

[16] G. Palmer et al., "Deep Reinforcement Learning for Autonomous Cyber Defence: A Survey," arXiv preprint arXiv:2310.07745, 2023. [Link]

[17] Y. Sun, D. Wu, Y. Xue, H. Liu, W. Ma, L. Zhang, Y. Liu, and Y. Li, "LLM4Vuln: A Unified Evaluation Framework for Decoupling and Enhancing LLMs' Vulnerability Reasoning," *arXiv preprint* arXiv:2401.16185, 2025. [Link]

[18] N. Rani and S. K. Shukla , *AURA: A Multi-Agent Intelligence Framework for Knowledge-Enhanced Cyber Threat Attribution"*, arXiv preprint arXiv:2506.10175, 2025. [Link]

[19] Thirunagalingam, A. (2024). AI-Powered Continuous Data Quality Improvement: Techniques, Benefits, and Case Studies. Benefits, and Case Studies (August 23, 2024).

[20] L. N. R. Mudunuri, V. M. Aragani, and P. K. Maroju, "Enhancing Cybersecurity in Banking: Best Practices and Solutions for Securing the Digital Supply Chain," Journal of Computational Analysis and Applications, vol. 33, no. 8, pp. 929-936, Sep. 2024.

[21] Mr. Anil Kumar Vadlamudi Venkata SK Settibathini, Dr. Sukhwinder Dr. Sudha Kiran Kumar Gatala, Dr. Tirupathi Rao Bammidi, Dr. Ravi Kumar Batchu. Navigating the Next Wave with Innovations in Distributed Ledger Frameworks. International Journal of Critical Infrastructures, PP 28, 2024. https://www.inderscience.com/info/ingeneral/forthcoming.php?jcode=ijcis

[22] Sehrawat, S. K., Dutta, P. K., Bhatia, A. B., & Whig, P. (2024). Predicting Demand in Supply Chain Networks With Quantum Machine Learning Approach. In A. Hassan, P. Bhattacharya, P. Dutta, J. Verma, & N. Kundu (Eds.), *Quantum Computing and Supply Chain Management: A New Era of Optimization* (pp. 33-47). IGI Global Scientific Publishing. https://doi.org/10.4018/979-8-3693-4107-0.ch002

[23] Mohanarajesh, Kommineni (2024). Generative Models with Privacy Guarantees: Enhancing Data Utility while Minimizing Risk of Sensitive Data Exposure. International Journal of Intelligent Systems and Applications in Engineering 12 (23):1036-1044.