International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Securing The Agentic Future Rethinking Application Security Testing for AI Powered & Autonomous Systems

Pankaj Pathania

Executive Summary

Organizations of all sizes and scopes are embracing artificial intelligence (AI) and agentic systems to boost productivity, automate decision- making and deliver differentiated customer experiences.

However, these advancements are giving rise to a new breed of threats that traditional application security testing (AST) tools and practices are not equipped to handle.

This whitepaper explores the transformative shift in application architecture and behavior brought on by AI and autonomous agents and presents a new security paradigm for organizations' key decision-makers.

Table of contents

1. Introduction

The adoption of generative AI and autonomous agents has accelerated dramatically.

According to McKinsey's 2024 State of AI report, 72% of organizations have adopted at least one AI capability, and over 40% are exploring or deploying agentic frameworks for tasks such as customer service automation, workflow orchestration, and content generation.

Agentic systems differ fundamentally from traditional software. They make decisions, learn from their environment, and autonomously interact with other systems. As a result, they challenge long-held assumptions about deterministic code execution and testable behavior in software development.

This shift calls for a reevaluation of how we approach application security testing. Traditional models focused on static code analysis and known vulnerability signatures are insufficient for environments where behavior is emergent and context-driven.

2. New Threats in the Era of Agentic Applications

AI and autonomous systems introduce unique threat vectors, many of which do not exist in traditional applications:

- **Prompt Injection:** Attackers manipulate LLMs by injecting user-supplied inputs to execute unauthorized actions. OWASP now lists prompt injection among the top threats to AI applications.

- **Model Manipulation and Supply Chain Risks:** Open-source and third-party models can be compromised at the source. The 2023 Hugging Face breach, where multiple malicious models were uploaded and downloaded thousands of times, is a key example

- **Impersonation and Spoofing:**

Autonomous agents can be tricked into executing instructions from adversarial agents that appear legitimate.

- **Emergent Behavior:** Systems can exhibit unpredictable behavior due to dynamic input processing, reinforcement learning, or unsupervised adaptation.

- **Memory Leaks and Hallucinations:** LLMs can leak sensitive data or generate factually incorrect, harmful outputs

Case Example:

In a 2024 simulation by the Stanford Center for Research on Foundation Models, LLM-driven agents exposed confidential data through repeated prompts with adversarial tokens, highlighting the ease of data leakage without proper testing mechanisms

3. The Limitations of Traditional AST Tools

Most enterprises rely on Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), and Interactive Application Security Testing (IAST). While these methods are valuable for traditional codebases, they fall short in agentic contexts for several reasons.

- **Static Analysis Can't Model Dynamic Behavior:** LLMs and agents execute actions based on runtime inputs that cannot be evaluated in a static context.

- **DAST Fails in Non-Deterministic Environments:** Tools that expect consistent behavior across inputs are ill-equipped to handle probabilistic outputs from AI models

- **Lack of Contextual Awareness:** Existing AST tools lack the natural language understanding needed to detect semantic- level vulnerabilities (e.g., biased responses, prompt leakage).

- **Blind Spots Around API Interactions:** Agents often interact across APIs with real-time learning. Traditional tools don't assess the security of dynamically generated API calls

4. Moving Towards AI-native Application Security Testing

To secure agentic systems, organizations must adopt AI-native security testing methods. These should encompass:

- **LLM Behavior Fuzzing:** Similar to traditional fuzzing, but focused on generating adversarial prompts and edge-case inputs to trigger unwanted behaviors.

- **Red Teaming for AI:** Human- in-the-loop testing using simulated adversaries to identify vulnerabilities in agent behavior, output, and memory handling.

- **Explainability and Traceability:** Implementing systems to trace decision-making logic in autonomous systems, including prompt history and model response paths.

- **Model Supply Chain Integrity:** Verifying the provenance, integrity, and security posture of third-party and open-source AI models before integration.

- **Security Gateways for Agent Actions:** Intervening in real time when agent behavior deviates from policy-compliant boundaries

Recommended Tool Capabilities:

- NLP-aware vulnerability scanners
- Reinforcement learning policy testing frameworks
- Agent simulation sandboxes for behavior testing

5. Strategic Recommendations for CIOs and CISOs

CIOs and CISOs must recognize that agentic applications require a hybrid governance model that merges AI ethics, software security, and operational controls.

Key recommendations include:

1. Establish an AI Security Framework

- Build upon NIST AI RMF, ISO/IEC 42001, and OWASP Top 10 for LLMs.
- Create a cross-functional AI security team including security engineers, data scientists, and software architects.

2. Redefine Security Testing Pipelines

- Integrate AI behavior testing into the CI/CD process.
- Evaluate AST tools for their support of AI-specific capabilities.

3. Invest in Training and Awareness

- Train DevSecOps teams on emerging threats, such as prompt injection and LLM misuse.
- Run internal red-teaming and tabletop exercises focused on AI failure modes.

4. Validate Third-Party AI Vendors

- Conduct SBOM-style validation for AI models (a.k.a. “Model BOM”).
- Assess providers for adherence to security and explainability standards.

5. Prepare for Regulation

- Align AI development with upcoming regulatory requirements such as the EU AI Act and U.S. Executive Orders.

6. The Road Ahead: Innovation, Regulation, and Risk

- AI adoption is not slowing down. Gartner predicts that by 2026, more than 80% of enterprise applications will embed some form of generative AI, up from less than 5% in 2023. The attack surface will grow exponentially as a result

Meanwhile, regulatory scrutiny is intensifying

The European Union's AI Act mandates transparency, robustness, and cybersecurity for high-risk AI systems. And U.S. agencies like
the FTC and NIST are crafting their own enforcement and guidance mechanisms.

Organizations must get ahead of this curve by embracing a proactive, AI-native approach to application security. Otherwise, waiting for tooling and standards to catch up will leave enterprises exposed

6. Conclusion

The rise of AI-powered and agentic systems is revolutionizing what applications are capable of. But with this power comes complexity and risk. Traditional AST tools and practices are no longer sufficient to protect applications that learn, adapt, and act autonomously.

CIOs and CISOs must lead the charge by evolving their security strategies to account for this new reality. From building AI-aware security pipelines to investing in new testing methodologies, the time to act is now.

Those who do will not only reduce risk but will empower their organizations to innovate with confidence