



AGENTIC_AI_SECURITY_PRIVACY_CHALLENGES.

Gardade Pratiksha¹

¹ Dattakala Group of Institutions, Faculty of Engineering, Pune, Maharashtra, India, India

Corresponding author: pratiksha5sept@gmail.com

Guide / Supervisor: Dr.Taware G.G

ARTICLE ID IJREK-2026-00127	ARTICLE TYPE Research Article	RESEARCH AREA Artificial Intelligence	PUBLICATION Volume 1, Issue Issue 1 (2026)
RECEIVED 19 October 2026	REVISED 20 October 2026	ACCEPTED 19 October 2026	PUBLISHED 20 October 2026

Abstract

Agentic Artificial Intelligence (AI) systems can plan multi-step tasks, use external tools, retrieve information, and take actions with limited human intervention. These capabilities create security and privacy risks beyond those found in conventional question-answering systems. An agent may encounter malicious instructions in retrieved content, misuse an authorized tool, expose confidential context, retain unnecessary personal data, or continue a harmful action through a chain of otherwise ordinary steps. This paper presents a risk-aware framework for analysing these challenges across the agent lifecycle. It organizes the main risks into instruction and prompt injection, excessive agency, insecure tool integration, data leakage, identity and access-control weaknesses, memory risks, and insufficient monitoring. The proposed approach combines least-privilege permissions, separation of trusted instructions from untrusted data, explicit approval for high-impact actions, privacy-aware memory, output validation, and audit logging. A staged evaluation plan is described using threat scenarios and measurable security and privacy indicators. This paper is a design-oriented study and does not claim experimental results or a deployed security product. Its central argument is that agentic AI should be treated as a socio-technical system whose safety depends on the model, tools, data, permissions, and surrounding operational controls.

Keywords: Agentic AI; AI security; privacy; autonomous agents; prompt injection; tool use; access control; data protection; risk management

I. INTRODUCTION

Artificial Intelligence systems are increasingly being designed to do more than generate a single response. Agentic systems can interpret a goal, create a plan, call tools, observe results, revise their plan, and continue until a task is completed or a stopping condition is reached. A typical agent may interact with browsers, databases, code interpreters, email services, enterprise APIs, or local files. This ability can improve productivity, but it also expands the system's attack surface. Traditional application security often assumes that software follows predefined workflows and that users explicitly initiate consequential operations. An agent may instead select tools dynamically and make decisions based on retrieved documents or messages that contain untrusted instructions. If safeguards are weak, a malicious web page or document may influence the agent, or a valid tool permission may be used in an unintended way. Privacy concerns also arise when an agent collects, stores, summarizes, or forwards personal and organizational information across multiple steps. A. Problem Statement The main problem is how to enable useful autonomous behaviour while preventing unauthorized actions, disclosure of sensitive information, and unsafe reliance on untrusted content. Security cannot be addressed only by asking the model to behave responsibly; controls must also exist in the application architecture, tool layer, data store, and monitoring process. B. Aim and Objectives The objectives are to: (1) identify major security and privacy risks in agentic AI; (2) map risks to the agent lifecycle; (3) propose a layered framework of preventive, detective, and corrective controls; (4) define a practical threat-based evaluation plan; and (5) discuss limitations and responsible deployment considerations. C. Scope and Contribution This paper focuses on software-based agents that use tools, external information, and optional memory to complete tasks. It provides a conceptual framework suitable for student research and prototype planning. It does not present a completed penetration test, measured risk reduction, or a claim that one control can eliminate all agent-related threats.

TABLE I

Risk area	Example threat	Primary control
Prompt instruction injection	A retrieved page attempts to override task instructions.	Separate instructions from data; screen and constrain tool actions.
Excessive agency	Agent sends messages, deletes files, or makes changes without adequate review.	Least privilege, action limits, and human approval for high-impact actions.
Sensitive disclosure	Private context appears in an answer, log, or external tool request.	Data minimization, redaction, destination checks, and access control.
Insecure integration	Tool accepts unsafe arguments or has overly broad credentials.	Schema validation, scoped credentials, sandboxing, and allowlists.
Memory retention	Unnecessary personal information persists across tasks.	Purpose-bound memory, expiry, user controls, and deletion procedures.
Weak monitoring	A multi-step failure is difficult to reconstruct.	Trace IDs, immutable audit events, alerting, and incident response.

II. BACKGROUND AND RELATED WORK

A. Agentic AI and the Expanded Attack Surface An agentic application typically combines a language model with planning logic, tool adapters, context retrieval, memory, and an execution environment. Each component introduces a different trust boundary. The model may misinterpret instructions; retrieved content may be adversarial; a tool may expose broad permissions; and a memory store may preserve information beyond its useful lifetime. Therefore, the security boundary includes the full application rather than the model alone. B. Security Guidance and Risk Taxonomies The NIST AI Risk Management Framework provides a general structure for governing, mapping, measuring, and managing AI risks [1]. OWASP's guidance on large language model applications describes risks such as prompt injection, sensitive information disclosure, excessive agency, and insecure output handling [2]. MITRE ATLAS documents tactics and techniques associated with attacks against AI-

enabled systems [3]. These resources are useful foundations, although agent-specific deployments still require threat modelling of their particular tools, permissions, and workflows. C. Privacy by Design Privacy protection requires purpose limitation, data minimization, access control, retention limits, and transparency about how personal information is used. Agent memory and retrieved context make these principles especially important: information supplied for one task should not automatically become available to every future task. Data protection must be considered during collection, processing, storage, sharing, and deletion. D. Identified Research Gap A useful practical approach must connect high-level AI risk categories to concrete runtime controls. For example, identifying prompt injection as a risk is not sufficient unless the system also limits what the agent can do after encountering suspicious content. This paper addresses that design-level need by linking threat categories to controls, monitoring evidence, and test scenarios.

III. PROPOSED RISK-AWARE FRAMEWORK

A. Framework Architecture The proposed framework places security checks around the agent rather than relying on a single model instruction. The user request first passes through identity and policy checks. The planner proposes a sequence of steps, but each tool call is evaluated by a policy enforcement layer. Retrieved content is treated as untrusted data, and outputs are checked before being returned or used to trigger consequential actions. Logging and monitoring operate across the workflow. User request Identity / scope Agent planner Plan / reason Policy gate Permission check Tool layer API / files Validator Output checks Cross-cutting controls: privacy audit logs monitoring human approval

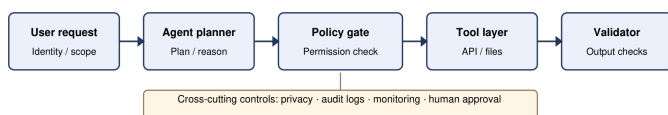


Fig. 1. Proposed layered security and privacy architecture for an agentic AI system.

B. Major Risk Categories Risk area Example threat Primary control Prompt / instruction injection A retrieved page attempts to override task instructions. Separate instructions from data; screen and constrain tool actions. Excessive agency Agent sends messages, deletes files, or makes changes without adequate review. Least privilege, action limits, and human approval for high-impact actions. Sensitive data disclosure Private context appears in an answer, log, or external tool request. Data minimization, redaction, destination checks, and access control. Insecure tool integration Tool accepts unsafe arguments or has overly broad credentials. Schema validation, scoped credentials, sandboxing, and allowlists. Memory and retention Unnecessary personal information persists across tasks. Purpose-bound memory, expiry, user controls, and deletion procedures. Weak monitoring A multi-step failure is difficult to reconstruct. Trace IDs, immutable audit events, alerting, and incident response. C. Core Design Principles The framework follows six principles: least privilege; explicit trust boundaries; human oversight for irreversible or high-impact actions; data minimization; fail-safe defaults; and traceability. Tool permissions should be scoped to the task, and the system should stop or request clarification when a request exceeds its authorized scope. Model-generated text should not be treated as proof that an action is safe or that an external operation succeeded.

IV. THREAT MODEL AND MITIGATION STRATEGY

A. Threat Actors and Assets Potential threat actors include malicious users, compromised third-party services, publishers of adversarial web content, and insiders with inappropriate access. Important assets include user prompts, private documents, credentials, memory records, tool outputs, audit logs, and actions performed on external systems. The threat model should also consider accidental misuse, model error, stale information, and failures caused by poorly configured integrations. B. Prompt Injection and Untrusted Content An agent may retrieve an email, document, or web page containing instructions that conflict with the user's request or system policy. The content should be processed as data rather than granted authority to redefine the agent's goals. Defences include clear instruction hierarchy, content provenance labels, restricted tool permissions, and checks before sensitive actions. Because detection may be imperfect, the strongest protection is to limit the damage an injected instruction can cause. C. Tool Misuse and Excessive Agency Tools should expose narrowly scoped operations instead of unrestricted shell, database, or

account access. Validate arguments against schemas, enforce destination allowlists where appropriate, and separate read operations from write operations. Require confirmation before sending external communications, changing permissions, deleting data, making purchases, or executing other consequential actions. Rate limits, transaction boundaries, and rollback mechanisms can reduce the impact of mistakes. D. Privacy and Data Leakage The application should collect only the information required for the task and avoid placing secrets in prompts, telemetry, or model-visible context unless necessary. Sensitive fields can be masked before external processing. Access to conversation history and memory should be purpose-bound and user-aware, with retention periods and deletion workflows. Logs should record security-relevant events without duplicating entire confidential documents or credentials. E. Identity, Credentials, and Isolation Every tool call should be associated with an authenticated user or service identity. Credentials should be stored in a secrets manager or protected environment configuration, not embedded in source code. Use short-lived tokens when supported, rotate exposed credentials, isolate code execution, and apply network restrictions. Separate development and production environments, and prevent one user's context from being reused in another user's session. F. Monitoring and Incident Response For each task, record a correlation identifier, requested operation, policy decision, tool name, outcome, timestamp, and relevant model or configuration version. Avoid logging secrets or unnecessary personal data. Monitoring should detect repeated policy denials, unusual tool-call volume, unexpected destinations, and failed authorization. An incident process should support disabling affected tools, revoking credentials, preserving relevant evidence, and informing responsible operators.

V. IMPLEMENTATION METHODOLOGY

TABLE II

Layer	Candidate implementation	Security purpose
Agent orchestration	A framework or custom controller with explicit step limits	Controls planning loops and stopping conditions.
Policy enforcement	Rule engine, tool allowlist, approval workflow	Authorizes each proposed action before execution.
Tool adapters	Typed API wrappers with schema validation	Limits arguments, permissions, and destinations.
Memory retrieval	/ Document store with access labels and retention rules	Restricts access and reduces unnecessary persistence.
Audit monitoring	/ Structured event logs and security alerts	Supports investigation and policy evaluation.

VI. EVALUATION PLAN AND DISCUSSION

TABLE III

Metric	Meaning	Evaluation use
Unauthorized-action rate	Fraction of prohibited actions that are executed.	Measures effectiveness of policy enforcement.
Attack success rate	Fraction of adversarial scenarios that achieve the defined objective.	Compares mitigations across test cases.
Sensitive-data leakage rate	Fraction of tests where protected data reaches an unauthorized output or destination.	Assesses privacy controls.
False-block rate	Fraction of legitimate actions incorrectly blocked.	Checks usability and over-restriction.
Audit completeness	Share of required security events with usable records.	Assesses traceability and incident readiness.
Human-approval compliance	Share of high-impact actions paused for required confirmation.	Checks oversight controls.

VII. LIMITATIONS, ETHICS, AND FUTURE SCOPE

A. Limitations Agent behaviour can vary across model versions, prompts, tools, and context lengths, making reproducibility difficult. Attack patterns evolve, and a test suite cannot represent every real-world environment. Privacy requirements also depend on jurisdiction, application domain, and the nature of the data. Controls can reduce risk but cannot guarantee that an agent will never produce an unsafe response or make an error. B. Ethical and Responsible Use Users should be informed when an

agent acts on their behalf, what data it accesses, and when human approval is required. High-impact decisions should remain subject to appropriate human accountability. Personal data should not be retained indefinitely or reused for unrelated purposes. System owners should document intended use, known limitations, escalation procedures, and responsibilities for handling incidents. C. Future Scope Future work can evaluate policy enforcement in multi-agent workflows, develop standardized agent threat benchmarks, investigate privacy-preserving memory, and measure how well controls transfer across models. Additional research may explore formal verification of tool permissions, provenance-aware retrieval, confidential computing, and continuous red-team testing. Real-world studies should assess both security outcomes and the effect of safeguards on task completion and usability.

VIII. CONCLUSION

Agentic AI introduces security and privacy challenges because models can plan actions, use external tools, and carry information across multiple steps. Major risks include prompt injection, excessive agency, insecure tool integration, sensitive-data leakage, weak identity controls, unsafe memory, and inadequate monitoring. This paper proposed a layered framework based on least privilege, trust-boundary enforcement, human approval for consequential operations, privacy-aware data handling, validation, and auditable monitoring. A threat-based evaluation plan was

also defined without inventing experimental outcomes. Responsible deployment requires testing the complete system—including its model, tools, data, and operational procedures—and continually reviewing risks as capabilities and integrations change.

REFERENCES

- [1] National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, January 2023. Available: <https://doi.org/10.6028/NIST.AI.100-1>
- [2] OWASP Foundation, "OWASP Top 10 for Large Language Model Applications," 2025. Available: <https://genai.owasp.org/llm-top-10/>
- [3] MITRE, "MITRE ATLAS: Adversarial Threat Landscape for Artificial-Intelligence Systems." Available: <https://atlas.mitre.org/>
- [4] National Institute of Standards and Technology, "Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations," NIST AI 100-2e2023, 2023. Available: <https://doi.org/10.6028/NIST.AI.100-2e2023>
- [5] European Union, "Regulation (EU) 2016/679 (General Data Protection Regulation)," Official Journal of the European Union, 2016. Available: <https://eur-lex.europa.eu/eli/reg/2016/679/oj>
- [6] OWASP Foundation, "OWASP Top 10 for LLM Applications: Excessive Agency," guidance within the LLM application risk taxonomy. Available: <https://genai.owasp.org/>

Publication archival note: Official scholarly contribution published under CC BY 4.0 International License. Permanent record on <https://www.ijrek.org> | Article ID: IJREK-2026-00127