BULGARIAN ACADEMY OF SCIENCES
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS • 2026 • Vol. **85**, pp. 42-59
p-ISSN: 2738-7356; e-ISSN: 2738-7364
https://doi.org/10.7546/PECR.85.26.05

Architectural Model for Controlling Autonomous AI Agents as Insider Threats in Information Security Systems

Ivan Gaidarski

Unmanned Robotic Systems Lab, Institute of Robotics, Bulgarian Academy of Science, Sofia, Bulgaria
Email: ivangaidarski@ir.bas.bg

Abstract: Organizations are increasingly relying on autonomous agents based on artificial intelligence (AI) to execute everyday business processes. These systems, referred as "Agentic AI", have access to sensitive data and information systems, operating without constant human oversight. This article proposes viewing such agents as a new class of insider threat. It analyzes key attack vectors against these systems, specifically: instruction injection, excessive autonomy, tool misuse, and uncontrolled data exfiltration. The proposed approach for developing information security systems is based on a conceptual model and a UML class diagram, augmented with a new "autonomous agent control" component and an "Action Gateway" class. The latter assesses the risk associated with each agent action, determining whether it should be permitted, escalated to a human for decision making, or blocked. The framework is demonstrated through three attack scenarios within a financial organization, illustrating how the proposed mechanisms can thwart attacks that bypass other security controls in this specific configuration. The framework is compared with similar approaches for controlling agent actions (such as GuardAgent and CaMeL); security is ensured through integration with the organization's existing security architecture, as well as by routing medium-risk actions to a designated responsible individual.

Keywords: *AI agents, insider threats, information security, model-driven design, data leak prevention*.

1. Introduction

In recent years, we have witnessed the widespread integration of AI into all aspects of our lives [33]. Moving beyond the traditional question and answer

model, AI systems have advanced to the next level by gaining the ability to take action. "Agentic AI" refers to autonomous systems capable of making decisions and executing tasks independently. Essentially, AI agents are software built upon Large Language Models (LLM), designed to independently analyze, plan, and take specific steps to achieve goals set by the user. After receiving a certain task, autonomous AI agents divide the task into stages, perform them and adjust their actions according to the obtained results [3], [4]. For this purpose, agents independently choose and use various tools and services – generate content and process documents, make inquiries in databases and Internet resources, read and send electronic mail [1], [2], [3]. The tools are usually external to the system, i.e. agents have the ability to communicate with other software systems or execution mechanisms. What's more, the agents themselves can split and delegate tasks to other Agentic AIs. Thus, a task can be performed by a whole collective of agents performing different roles and having different specializations.

Agentic AIs can take partial or full control of traditional activities in an organization, such as marketing, sales, procurement, writing content or programming code, testing and debugging, etc. In practice, Agentic AIs can be a full-fledged digital worker without direct human control. AI agents execute an "observe–reason–act" loop, which repeats until a specific goal is achieved [1]. The architecture of AI agents is built from four main components:

1. *Language model* – understands the task and makes decisions;
2. *Planning* – divides the task into stages and determines the execution sequence;
3. *Memory* – it can be short-term, determined by the current context, and long-term, which builds up a knowledge base and a history of interactions;
4. *Tools* – interfaces to emails, files, databases, ERP and CRM systems, API interfaces to external services.

To perform their tasks, AI agents rely on standardized protocols that connect LLM with external data, tools, and services [4], [5]:

- Anthropic's Model Context Protocol (MCP), which allows an agent to be easily connected to other systems, data, files, or tools.
- Universal Tool Calling Protocol (UTCP): a lightweight alternative to MCP for connecting to other systems and sources, without the need for an intermediary proxy server.
- Google's Agent-to-Agent (A2A), designed for communication between AI agents. It enables the management and coordination of tasks within larger-scale systems.
- IBM's Agent Communication Protocol (ACP), an open standard for managing AI agents that incorporates a "human-in-the-loop" approach.

In practice, different protocols are often combined depending on the system architecture and the assigned tasks [5].

The protocols allow for the expansion of the agent's scope and the assets available to it, but they also increase risks to the system from a cybersecurity perspective. The extent to which an agent acts without human intervention is described through levels of autonomy. In Table 1, a multi-level adaptation of the model of Parasuraman and colleagues [6] is proposed, where each level corresponds to a quantitative index $A \in [0, 1]$.

Table 1. Autonomy levels of AI agents

Level	Description	Role of the human	A
0. Assistant	The agent only suggests solutions	Executes all actions	0
1. Execution with approval	The agent acts after approval	Approves every action	0.25
2. Conditional autonomy	The agent acts alone except for risky actions	Approves risky actions	0.50
3. High autonomy	The agent acts alone	Subsequent control	0.75
4. Full autonomy	The agent acts without human involvement	No direct involvement	1

As noted earlier, AI agents have recently been entering various economic sectors on a large scale. They are entrusted with important tasks and granted access to sensitive data and assets. Increasingly, tasks are performed by groups of autonomous agents with different roles that work together, forming multi-agent systems [2]. In practice, it turns out that organizations that have implemented systems with gentic AI do not know specifically which agents have been deployed, for exactly what tasks, on whose behalf these agents are acting, nor what rights have been delegated to them at any given moment [7], [8]. All this creates some uncertainty, which facilitates possible attacks and security breaches. Organizations such as ENISA report an increase in the threat landscape related to the use of AI and in particular agentic AI [9].

To carry out assigned tasks, the agent is granted specific access rights to sensitive resources and assets, as well as authorization to perform active operations, much like an employee within the organization. In effect, the agent becomes an "insider" with authorized access to sensitive information, just like the corresponding employee. By possessing such access and the authority to execute specific actions, the agent becomes a significant risk factor for the security of the organization's assets and data, effectively becoming an "insider threat".

Organizations have protective measures in place to counter internal threats originating from individuals (insiders). Traditional perimeter defense and internal threat mitigation strategies have been developed with human characteristics in mind [30]. The emergence of autonomous AI agents introduces new types of

threats against which traditional defensive measures prove ineffective. In accordance with the "weakest link" principle, an attacker would likely target precisely these new – and currently poorly defended – elements. The appearance of new security vulnerabilities and threats within an organization's protected perimeter necessitates the development of a new type of information security system (ISS) that incorporates innovative protective methods.

This article is a continuation of the author's previous work, extending the methods and models from [10], [11], [12] by introducing a new type of insider threat, stemming from the use of autonomous AI agents, as well as a new model for the management of such agents. Previous research includes a method and models for developing information security systems (ISS) within organizations [10], a model-based approach to ISS design [12], and an analysis of insider threats to critical infrastructure [11]. The article proposes a method for designing and constructing an architectural model to control autonomous AI agents acting as insider threats within information security systems. It presents a rationale for considering AI agents as a new class of insider threats, a new threat model for agent-based AI systems, and an extension of traditional conceptual and design UML models for information security systems. The framework is illustrated through three attack scenarios and compared with existing related approaches for controlling agent actions - specifically, GuardAgent and CaMeL.

2. Autonomous AI Agents as an Insider Threat

When individual with access to confidential information or other organizational sensitive assets violates a security policy, an incident caused by an "insider threat" occurs. Such individuals, who may include both employees of the organization and external service providers with access to sensitive resources, are referred to as "insiders". Incidents involving insiders can be accidental and unintentional, or entirely deliberate aimed at causing harm to the organization [10], [14]. In either case, the outcome entails negative consequences for the organization. In the new reality, the concepts of "insider threat" and "insider" must be broadened to encompass not only traditional human insider threats but also new risks arising from the emergence of autonomous artificial intelligence systems (Agentic AI).

In this article, we propose expanding the concept of an "insider" to encompass not only human individuals but also entities within software systems that possess specific delegated rights and operate autonomously within the organization's protected perimeter.

According to an agent-based approach, agents gather information about their environment, possess the capability to process it, and take action based on that information through execution mechanisms or commands issued to external systems [2]. AI agents combine characteristics from all three categories of insiders defined in [10]: they hold delegated rights, perform routine tasks much like a

typical user, and are generally provided by an external vendor. Table 2 summarizes the key differences between an employee (a human individual) and an AI agent (a system-based insider).

Table 2. Comparison between an employee and an autonomous AI agent as insiders

Characteristic	Employee	Autonomous AI agent
Data access	role-based, full or limited	Often broad, inherited from the user or from a service account
Speed of action	Relatively infrequent operations	Machine speed, exceeding human speed
Cause of incident	Negligence, error, malicious intent	Manipulation through content, model error, excessive permissions
Predictability	Stable behavior patterns	Context-dependent behavior
Accountability	Personal responsibility	Shared among the user, the provider, and the organization
Control coverage	DLP, access control, training [31], [32]	Partial - the controls do not see the agent's actions

The most significant difference between an employee and an autonomous AI agent is that the agent is unable to reliably distinguish between data and instructions. Text within a document or email can be interpreted as a command to be executed. This constitutes a "data poisoning" or "indirect prompt injection" attack, in which malicious commands are hidden within training data or data retrieved from external sources. Thus, human "negligence" is replaced by "manipulability" in the case of an AI agent [13]. Another issue regarding the system insiders is the machine speed at which agents operate; consequently, incidents can cause greater damage before they are even detected.

Various taxonomic studies are employed to classify different types of insider threats, as CERT Guide to Insider Threats [14]. One of the most recent and comprehensive frameworks, VISTA [15], identifies generative AI as a new source of risk, yet it does not categorize software agents as distinct insider entities.

3. Threat Model for Agentic Systems

Fig. 1 illustrates the attack surface and attack vectors of an autonomous AI agent. The agent receives input data from users, documents, web content, and other agents. The language model interprets the input data, the planning component selects actions, and the memory stores context across individual tasks. Actions are executed via tools, each of which carries specific privileges regarding the organization's assets.

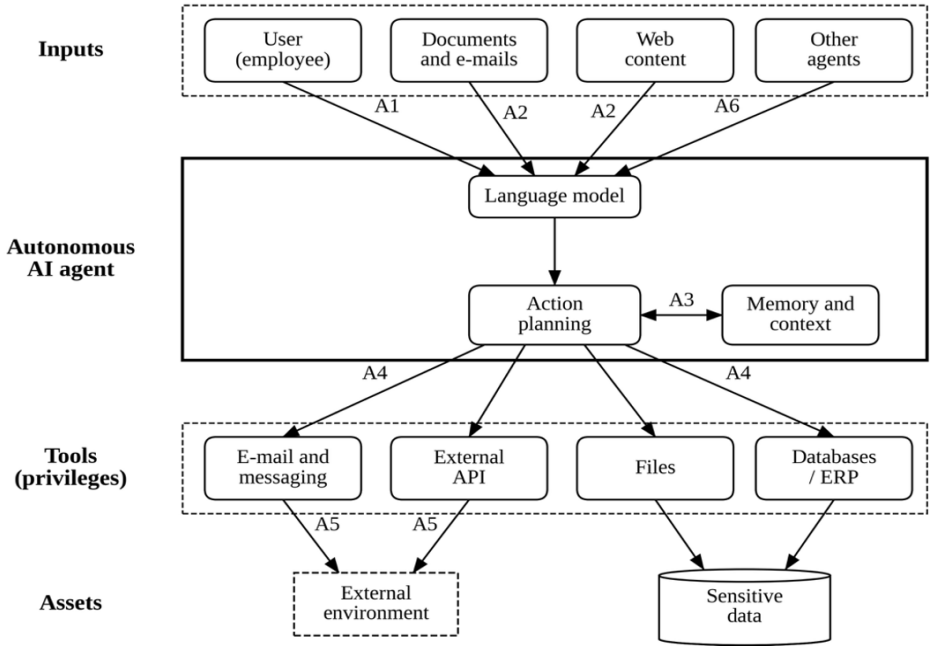


Fig. 1. Attack surface of an autonomous AI agent and attack vectors A1–A6

Based on [16], [17], [18] and the analysis of the problem domain in [10], we identify six primary attack vectors (Table 3). Direct instruction injection has been known since the early language models [19], while studies utilizing test environments such as InjecAgent [20] and AgentDojo [21] demonstrate that agents equipped with tools are also vulnerable to indirect injection. For each vector, the affected properties from the "Confidentiality, Integrity, Availability" (CIA) triad are indicated, along with the state of the data.

Table 3. Attack vectors against autonomous AI agents

Vector	Description	CIA	Data state
A1. Direct injection	A user submits instructions that bypass the policy	C, I	Data in use
A2. Indirect injection	Hidden instructions in documents, emails, or web pages	C, I	Data in use
A3. Memory poisoning	Insertion of false information into the agent's memory	I	Data at rest
A4. Excessive autonomy	Tool actions beyond the needs of the task	C, I, A	Data in use
A5. Data exfiltration	Sending sensitive data outside via email or API	C	Data in motion
A6. Cascading compromise	A compromised agent passes harmful instructions to other agents	C, I, A	Data in motion

C – confidentiality, I – integrity, A – availability.

In accordance with the approach of ISO/IEC 27005 [22], in our article the risk is operationalized as the product of the probability of the realization of the threat and the impact:

$$R = P \cdot I \quad (1)$$

For an autonomous agent a, the risk is aggregated over all attack vectors, taking into account the agent's level of autonomy:

$$R_a = \sum_{i=1}^{n} P_i \cdot I_i \cdot (1 + \alpha \cdot A) \quad (2)$$

where P_i is the probability of a successful attack via vector i, I_i is the impact, $A \in [0, 1]$ is the autonomy level according to Table 1, and α is a coefficient reflecting how much autonomy amplifies the risk.

Equation (2) is a heuristic index used during the design and planning of control measures: attack vectors are not mutually exclusive (implying potential double-counting), the multiplier $(1 + \alpha \cdot A)$ is heuristic, and α is determined by policy rather than empirical calibration. The score S(a) from Section 4.3 is calculated at runtime for each individual action; there is no formal relationship linking the two scores.

4. Extended Model of an Information Security System

4.1. Conceptual Model

The methodology for developing information security systems (ISS) [10] involves analyzing the domain, defining a framework for describing the architecture of the system and constructing conceptual models. These are subsequently transformed into an object-oriented design model, and finally into an implementation model.

The generalized conceptual model of traditional ISS (Fig. 2) consists of six components that address the questions: "What are we protecting?", "Why are we protecting it?", "How are we protecting it?", and "Where are we protecting it?".

The "Security Model and Policy" component guides "Management and Configuration" activities and relates to data protection. "Management and Configuration" handles security settings for end devices and communications, while "Monitoring and Analysis" tracks these elements and provides feedback on their status to the "Security Model and Policy" component. The protection of endpoints and communications encompasses data protection as well – "Data Protection" module. In this model, the entities operating within the system include humans, devices, and applications. The AI agent is treated as a standard application running on the endpoint.

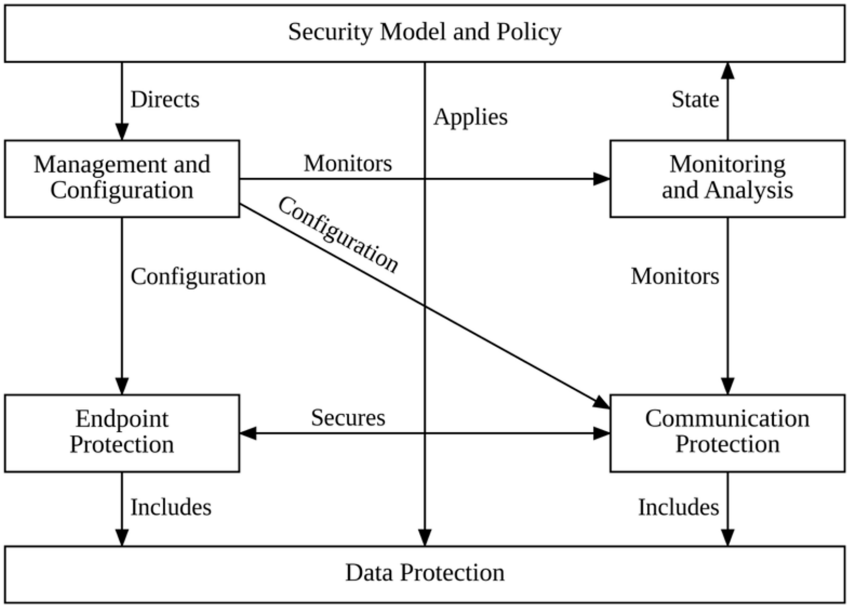


Fig. 2. Traditional generalized conceptual model of the ISS

The emergence of autonomous AI agents raises a new question: "Who is acting?" Therefore, we propose expanding the generalized model to include a seventh component "Autonomous Agent Control" (Fig. 3).

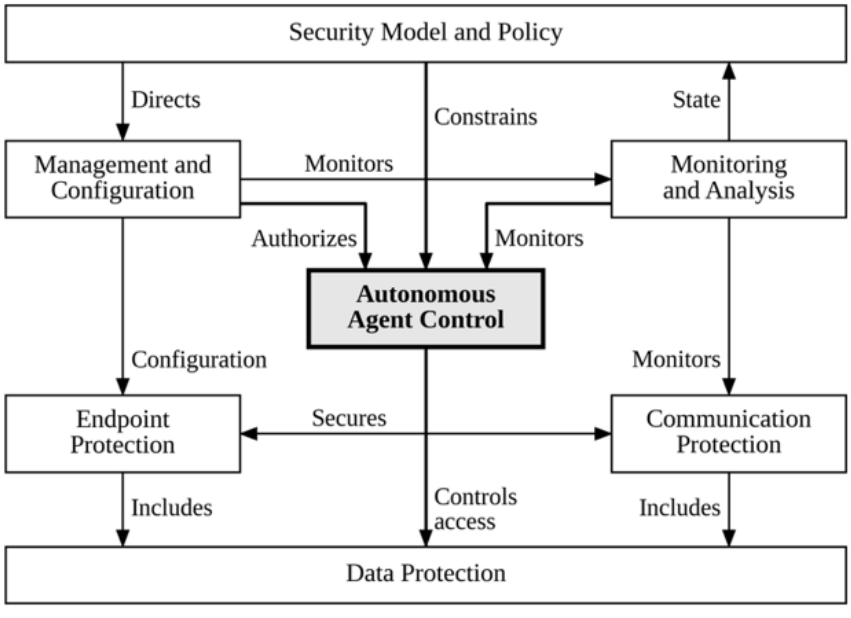


Fig. 3. Extended generalized conceptual model of the ISS

This component is constrained by the "Security Model and Policy", governed through "Management and Configuration", subject to "Monitoring and Analysis", and it controls the agents' access to the "Data Protection" component. The differences between the two models are summarized in Table 4.

Table 4. Comparison of the traditional and the extended conceptual model of the ISS

Criterion	Traditional model	Extended model
Number of components	6	7 (+Autonomous Agent Control)
Key questions	What, why, how and where do we protect?	"Who acts?" is added
Subjects in the system	People, devices, applications	Autonomous agents with delegated privileges are added
Place of the AI agent	Ordinary application on the endpoint	Separate object of control
Data access control	By user and channel	Control by agent, task and action is added
Human oversight	Not explicitly included	Escalation of risky actions
New relations	–	Constrains, authorizes, monitors, controls access

The new "Autonomous Agent Control" component applies the basic principles of information security to AI agents:

- Triple A principle (Authentication, Authorization, Accounting): the agent has its own identity (authentication), rights according to role and task (authorization) and every action it takes is registered (accountability);
- Principle of least privileges: the agent receives only the tools and data needed for the specific task;
- Multilayer (Onion) model: the agent's actions go through several independent controls, in accordance with the concept of Zero Trust [23];
- Human supervision – high-risk actions require the approval of a responsible person [24], [25].

4.2. Design Model

The design model is derived through the transformation "conceptual model → object-oriented model," described using the UML language [10], [12]. The architectural model of a traditional security system (ISS) is represented by a UML class diagram (Fig. 4). It consists of four classes. The "Management" class implements the security model and policy, the configuration, and the analysis method. The "Protection" class ensures the protection of end devices and data

communication across all three data states. The "Control" class handles device management, content inspection, static data scanning, and monitoring. The "Data" class implements actions such as "Skip," "Block," and "Report."

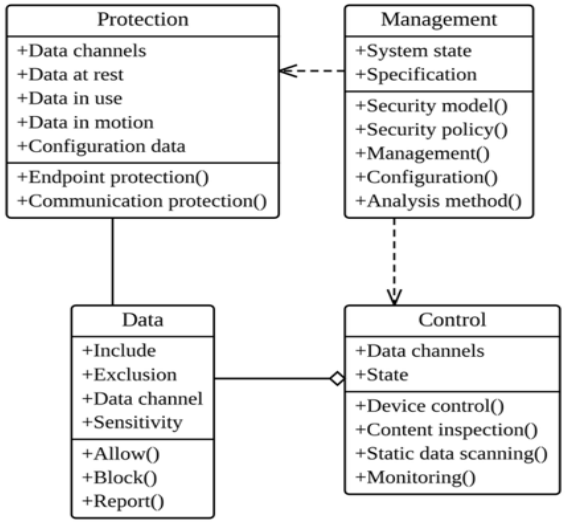


Fig. 4. UML class diagram of the traditional ISS

The UML class diagram of the extended ISS is presented in Fig. 5.

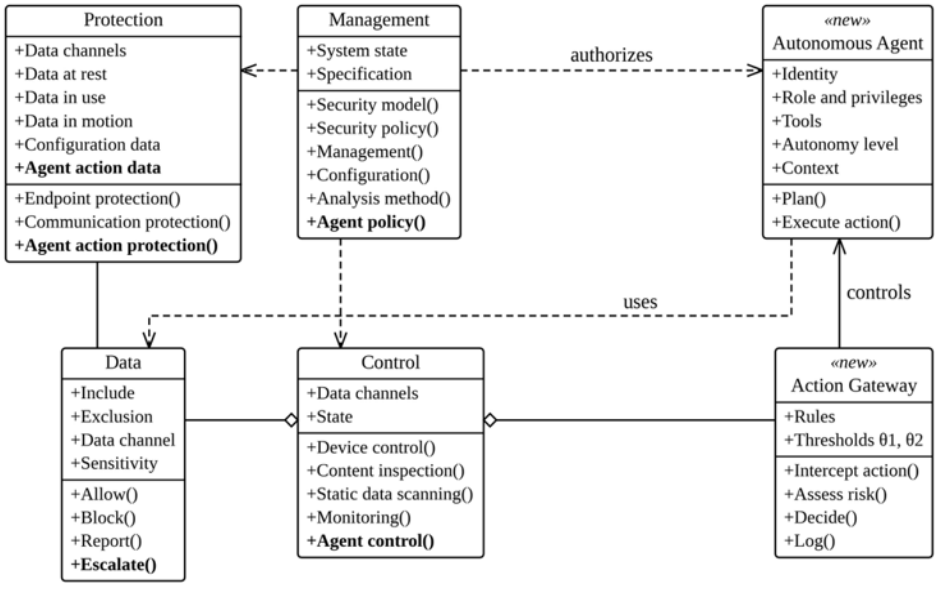


Fig. 5. UML class diagram of the extended ISS

Two new classes have been added. The "Autonomous Agent" class describes an agent in terms of its identity, role and rights, available tools, level of autonomy, and context. The "Action Gateway" class is part of the "Control" component and intercepts every action taken by an agent before it reaches the tool. New methods have been added to the existing classes: "Agent Policy()" in "Management", "Agent Action Protection()" in "Protection", "Agent Control()" in "Control", and "Escalate()" in "Data".

Table 5. Comparison of the UML class diagrams of the traditional and the extended ISS

Element	Traditional ISS	Extended ISS
Number of classes	4	6 (+ Autonomous Agent, Action Gateway)
“Management”	Model, policy, configuration	+ Agent policy()
“Protection”	Endpoints and communications	+ Agent action protection(), agent action data
“Control”	Devices, content, monitoring	+ Agent control(), aggregates “Action Gateway”
“Data”	Allow, Block, Report	+ Escalate()
New relations	–	authorizes, controls, uses
Point of control	Data channel	Every agent action

4.3. Functional model

The functional model is represented by a UML sequence diagram (Fig. 6).

The agent does not perform its actions directly, but submits them as requests to the gateway. The gateway checks the agent's rights in the security policy, performs content inspection, and calculates a risk score of the action.

Checks are performed in a specific order of priority: (1) verification of access rights, which are defined at the agent, task, and action levels; for actions falling outside the scope of the task, the process is terminated without calculating S(a); (2) content verification; (3) quantification of $S(a)$ and determination of a solution according to formula (5). Overt violations of prohibitions imposed by the rules result in rejection, regardless of the value of $S(a)$.

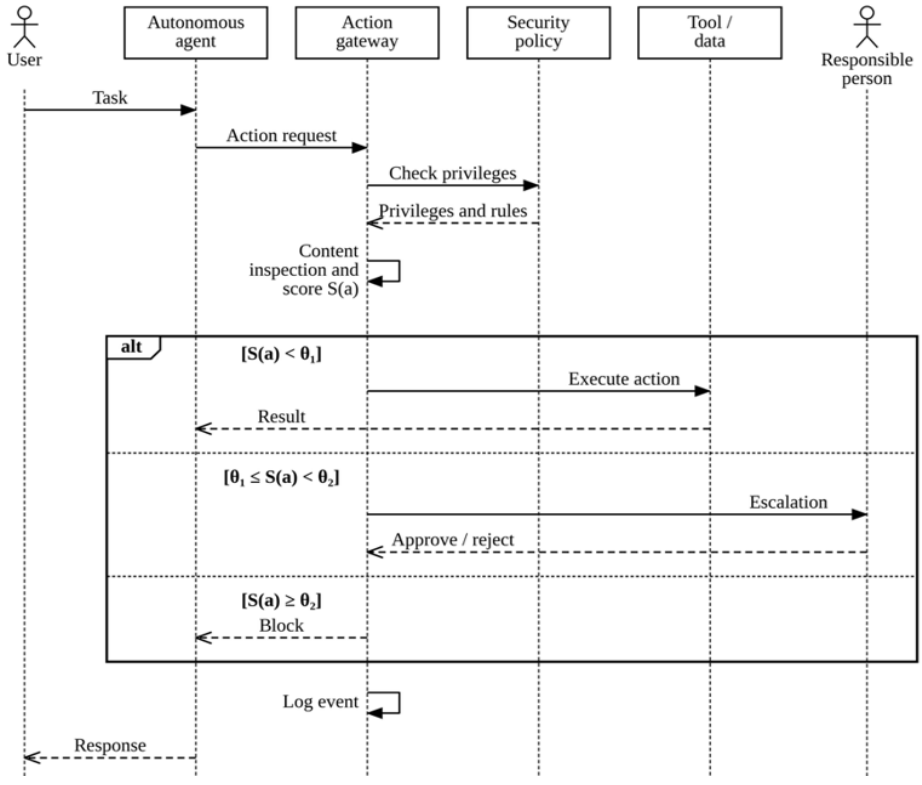


Fig. 6. UML sequence diagram for processing an agent action

To detect atypical behavior, a deviation coefficient is used, analogous to the one for the "Violation Agent" described in [10]:

$$K = \min(1, |x - x_0| / x_0), \ x_0 > 0 \quad (3)$$

where x is an observed behavioral parameter (volume of data read, number of tool invocations, number of recipients), and x_0 is its normal value for the corresponding role. In the absence of an established baseline value ($x_0 = 0$; a new role or a new task), the rules may set a conservative value of $K = 1$ until historical data is accumulated.

The factors C, T, and K are normalized within the interval [0, 1]: $C(a)$ reflects the sensitivity class of the affected data according to the policy (e.g., 0.3 – internal data, 0.7 – business-critical data, 1.0 – personal data); $T(a)$ represents the privilege level of the tool (e.g., 0.3 – read access, 0.6 – sending emails and file operations, 1.0 – invoking external APIs and data export); and $K(a)$ is the normalized deviation of quantitative behavioral parameters (data volume, number of recipients, number of tool invocations) relative to the baseline value for the corresponding role.

The risk score $S(a)$ of action a combines three factors:

$$S(a) = w_1 \cdot C(a) + w_2 \cdot T(a) + w_3 \cdot K(a) \quad (4)$$

where C is the sensitivity of the affected data, T is the privilege level of the tool, K is the normalized deviation, and $w_1 + w_2 + w_3 = 1$ are weights set by the security policy. The gateway decision is made using two thresholds:

$$D(a) = \begin{cases} \text{allow}, & S < \theta_1 \\ \text{escalate}, & \theta_1 \leq S < \theta_2 \\ \text{block}, & S \geq \theta_2 \end{cases} \quad (5)$$

To illustrate the calculation, we assume an example policy configuration: $w_1 = 0.5$; $w_2 = 0.3$; $w_3 = 0.2$; $\theta_1 = 0.5$; $\theta_2 = 0.8$. Suppose an agent proposes an action, which has passed mandatory checks, involving high data sensitivity ($C = 1.0$), a moderately privileged tool ($T = 0.6$), and a data volume ten times the baseline ($K = 1.0$). The resulting score is $S = 0.5 \cdot 1.0 + 0.3 \cdot 0.6 + 0.2 \cdot 1.0 = 0.88 \geq \theta 2$, and the action is blocked. In the case of an action with medium sensitivity ($C = 0.7$), a highly privileged tool class ($T = 1.0$), and a moderate deviation ($K = 0.4$), the score is $S = 0.73$; since $\theta_1 \leq S < \theta_2$, the action is escalated to a responsible individual. These values are provided solely to demonstrate the calculation; they have not been empirically calibrated and do not constitute evidence of the framework's effectiveness.

5. Illustrative Scenarios

To illustrate the proposed model, three attack scenarios are examined. The analysis considers the use of a "Level 3" agent within a medium-sized financial institution, characterized by "high autonomy" (Table 1). The agent processes client emails, performs searches in CRM systems, prepares and sends documents, and generates summaries of internal materials. The protected assets include personal data, transaction data, as well as information regarding clients and corporations [10]. The three scenarios, presented in Fig. 7 and Table 6, focus on different control points. The three scenarios and the "Action Gateway" solutions within the proposed configuration are illustrated in Fig. 7.

The solutions presented in Table 6 are illustrative and do not represent measurement results. The numerical example in Section 4.3 demonstrates the calculation without performing calibration of the scenarios. Before determining the rating S(a), an authorization check is conducted, and explicit prohibitions are taken into account.

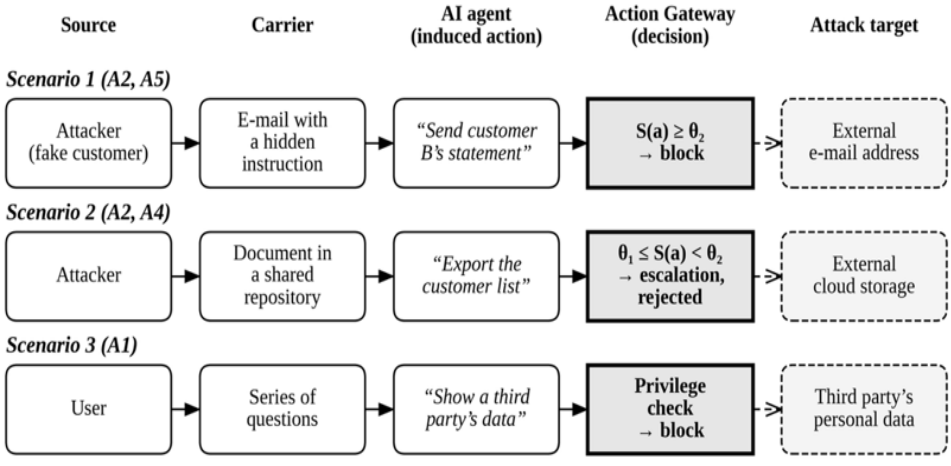


Fig. 7. Attack scenarios and decisions of the "Action Gateway"

Table 6. Handling of the scenarios: traditional controls and proposed framework

Scenario	Traditional configuration	Proposed control and decision	Model element
1. Email with indirect injection (A2, A5): extraction of another customer's data to an external address	DLP inspects the email, but the API channel in the configuration considered is not covered.	Check of data ownership and of the permitted recipient. In case of a policy violation: rejection regardless of S(a).	Gateway; permissions; inspection
2. Document with hidden instructions (A2, A4): export of a customer list to the cloud	The agent has export permissions; the API channel is not inspected.	In the absence of an explicit prohibition and $\theta_1 \le S(a) < \theta_2$: escalation; the responsible person refuses.	Gateway; score S(a); oversight
3. Series of questions (A1): personal data of a third party	The dialogue is not inspected; access to the CRM is broad.	Permissions are limited to the current task and customer; the request is rejected before the score S(a) is computed.	Policy; minimal permissions

The comparison applies only to the described configuration, not to all DLP systems. The principle of least privilege limits potential damage, while the gateway is designed to encompass specific operational actions. Threshold values θ_1 and θ_2 determine the distribution between allowing, escalating, and blocking: low thresholds strengthen control but simultaneously increase the burden on the personnel responsible, as well as the risk of "security fatigue" [10]. The scenarios illustrate the combined effect of the gateway, the principle of least privilege, and

oversight, rather than the isolated contribution of the gateway itself. As a prime target for attacks, the gateway must be protected, managed, and monitored by the ISS (Fig. 3); the tools must not be accessible via workarounds. Quantitative effectiveness, the workload associated with escalations, and additional delay (latency) require experimental evaluation.

6. Comparison with related approaches for managing agent actions

The proposed framework is compared with related approaches for managing agent actions: GuardAgent [28] and CaMeL [29]. These frameworks perform verification of agent actions and offer specific protection mechanisms, while our framework describes their integration into the organizational model of a ISS. The differences are summarized in Table 7.

GuardAgent [28] analyzes rules and generates executable code to verify input data and agent logs, including proposed actions. The code executes deterministically, providing a binary decision and specifying the reason for any failure; it also utilizes a memory of past instances. An accuracy of over 98% was achieved on EICU-AC, and 83% on Mind2Web-SC. Limitations include the manually defined toolkit and the lack of validation for individual steps within the reasoning process.

CaMeL [29] separates control flows from untrusted data and enforces policies through capabilities that link output values to permitted recipients. Within the scope of the considered threat model, when protective mechanisms are applied, AgentDojo successfully completes 77% of tasks, compared to 84% without such protection. Future plans include the introduction of explanations and real-world environments, a requirement for user confirmation in the event of policy violations. Limitations include token consumption, the complexity of implementation and maintenance, "confirmation fatigue," and the fact that certain text manipulations and side channels remain unprotected.

7. Conclusion

The article examines the issue of insider threats arising from the use of autonomous AI agents, which are categorized as a new class of insider threat.
A threat model featuring six attack vectors is proposed. The traditional conceptual model and the UML class diagram for ISS are extended to include an "Autonomous Agent Control" component and an "Action Gateway" class; the differences between the traditional and extended models are summarized in comparative tables. An analysis of three attack scenarios within a financial organization illustrates how a combination of an action gateway, the principle of least privilege, and a Data Loss Prevention (DLP) system can thwart attacks that

bypass traditional controls, leaving only intermediate risk decisions for human intervention in the examined scenarios.

The framework incorporates mechanisms for human oversight (escalation to a responsible individual), traceability (logging every action taken by an agent), and data access restriction (least privilege). These mechanisms align with the direction of regulatory requirements concerning human oversight of artificial intelligence systems [24], cybersecurity [27], and personal data protection [26]. However, their applicability to specific regulatory obligations requires a separate assessment; the proposed architecture does not, in itself, guarantee regulatory compliance. Future research will focus on implementing the action gateway in a real-world corporate environment, quantitatively evaluating its effectiveness through simulations and experiments using real-world data, agent-based modeling of interactions among multiple agents, assessment on a public benchmark (e.g., AgentDojo [21]), and comparison with the GuardAgent [28] and CaMeL [29] approaches.

Acknowledgement
This work was supported by the NSP DS program, which has received funding from the Ministry of Education and Science of the Republic of Bulgaria under the grant agreement no. D01-74/19.05.2022.

References

1. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: ReAct: Synergizing reasoning and acting in language models. In: Proc. of the International Conference on Learning Representations (ICLR 2023) (2023).
2. Wooldridge, M.: An Introduction to MultiAgent Systems. 2nd edn. John Wiley & Sons (2009).
3. Xi, Z., et al.: The rise and potential of large language model based agents: A survey. Science China Information Sciences 68, 121101 (2025).
4. Anthropic: Model Context Protocol – Specification (2024). https://modelcontextprotocol.io
5. Ehtesham, A., Singh, A., Gupta, G. K., Kumar, S.: A survey of agent interoperability protocols: Model context protocol (MCP), agent communication protocol (ACP), agent-to-agent protocol (A2A), and agent network protocol (ANP). arXiv preprint arXiv:2505.02279 (2025). https://arxiv.org/abs/2505.02279.
6. Parasuraman, R., Sheridan, T. B., Wickens, C. D.: A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics – Part A 30(3), 286–297 (2000).
7. Chan, A., et al.: Visibility into AI agents. In: Proc. of the ACM Conference on Fairness, Accountability, and Transparency (FAccT '24) (2024).

8. Shavit, Y., et al.: Practices for governing agentic AI systems. OpenAI White Paper (2023).
9. ENISA: ENISA Threat Landscape 2024. European Union Agency for Cybersecurity (2024).
10. Gaidarski, I., Kutinchev, P.: Some aspects of information security and cybersecurity problem area. Problems of Engineering Cybernetics and Robotics 79, 55–66 (2023).
11. Gaidarski, I.: Model driven development of information security system. Problems of Engineering Cybernetics and Robotics 76, 47–62 (2021).
12. Gaidarski, I., Minchev, Z.: Insider threats to IT security of critical infrastructures. In: Tagarev, T., et al. (eds.) Digital Transformation, Cyber Security and Resilience of Modern Societies. Studies in Big Data, vol. 84, pp. 381–394. Springer, Cham (2021).
13. Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., Fritz, M.: Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In: Proc. of the 16th ACM Workshop on Artificial Intelligence and Security (AISec '23), pp. 79–90 (2023).
14. Cappelli, D. M., Moore, A. P., Trzeciak, R. F.: The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Technology Crimes (Theft, Sabotage, Fraud). Addison-Wesley, Boston (2012).
15. Renaud, K., Warkentin, M., Pogrebna, G., van der Schyff, K.: VISTA: An inclusive insider threat taxonomy, with mitigation strategies. Information & Management 61(1), 103877 (2024).
16. OWASP Foundation: OWASP Top 10 for Large Language Model Applications (2025). https://genai.owasp.org
17. MITRE: ATLAS – Adversarial Threat Landscape for Artificial-Intelligence Systems. https://atlas.mitre.org
18. National Institute of Standards and Technology: Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1 (2023).
19. Perez, F., Ribeiro, I.: Ignore previous prompt: Attack techniques for language models. In: NeurIPS ML Safety Workshop (2022).
20. Zhan, Q., Liang, Z., Ying, Z., Kang, D.: InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In: Findings of the Association for Computational Linguistics: ACL 2024 (2024).
21. Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., Tramèr, F.: AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In: NeurIPS 2024 Datasets and Benchmarks Track (2024).
22. ISO/IEC 27005:2022: Information security, cybersecurity and privacy protection – Guidance on managing information security risks. ISO (2022).
23. Rose, S., Borchert, O., Mitchell, S., Connelly, S.: Zero Trust Architecture. NIST Special Publication 800-207 (2020).

24. European Parliament, Council of the European Union: Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Official Journal of the European Union L 2024/1689 (2024).
25. ISO/IEC 42001:2023: Information technology – Artificial intelligence – Management system. ISO (2023).
26. European Parliament, Council of the European Union: Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation). Official Journal of the European Union L 119, pp. 1–88 (2016).
27. European Parliament, Council of the European Union: Directive (EU) 2022/2555 on measures for a high common level of cybersecurity across the Union (NIS 2 Directive). Official Journal of the European Union L 333, pp. 80–152 (2022).
28. Xiang, Z., Zheng, L., Li, Y., Hong, J., Li, Q., Xie, H., Zhang, J., Xiong, Z., Xie, C., Yang, C., Song, D., Li, B.: GuardAgent: Safeguard LLM agents via knowledge-enabled reasoning. In: Proc. of the 42nd International Conference on Machine Learning (ICML 2025) (2025). https://arxiv.org/abs/2406.09187.
29. Debenedetti, E., Shumailov, I., Fan, T., Hayes, J., Carlini, N., Fabian, D., Kern, C., Shi, C., Terzis, A., Tramèr, F.: Defeating prompt injections by design. arXiv preprint arXiv:2503.18813 (2025). https://arxiv.org/abs/2503.18813.
30. Fatkieva , R., Evnevich, E., Yoshinov, R.: State-of-the art and trends in network security control. Problems of Engineering Cybernetics and Robotics 72, 41–55 (2020). DOI: 10.7546/PECR.72.20.05.
31. Yoshinov, R., Chehlarova, N., Dishkova, G.: Assessment of teachers' preparedness in cybersecurity. Problems of Engineering Cybernetics and Robotics 84, 21–34 (2026). DOI: 10.7546/PECR.84.26.02.
32. Yoshinov, R., Dishkova, G., Chehlarova, N.: Assessment of parents' awareness in the field of cybersecurity. Problems of Engineering Cybernetics and Robotics 84, 35–45 (2026). DOI: 10.7546/PECR.84.26.03.
33. Yoshinov, R., Kotseva, M., Madzharov, A., Chehlarova, N.: Skills and attitudes towards using AI based chatbots. Environment. Technology. Resources 2, 138–142 (2024). DOI: 10.17770/etr2024vol2.8064.