With Great Capabilities Come Great Responsibilities: Introducing the Agentic Risk & Capability Framework for Governing Agentic AI Systems

Anonymous submission

Abstract

Agentic AI systems present both significant opportunities and novel risks due to their capacity for autonomous action, encompassing tasks such as code execution, internet interaction, and file modification. This poses considerable challenges for effective organizational governance, particularly in comprehensively identifying, assessing, and mitigating diverse and evolving risks. To tackle this, we introduce the Agentic Risk & Capability (ARC) Framework, a technical governance framework designed to help organizations identify, assess, and mitigate risks arising from agentic AI systems. The framework's core contributions are: (1) it develops a novel capability-centric perspective to analyze a wide range of agentic AI systems; (2) it distills three primary sources of risk intrinsic to agentic AI systems - components, design, and capabilities; (3) it establishes a clear nexus between each risk source, specific materialized risks, and corresponding technical controls; and (4) it provides a structured and practical approach to help organizations implement the framework. This framework provides a robust and adaptable methodology for organizations to navigate the complexities of agentic AI, enabling rapid and effective innovation while ensuring the safe, secure, and responsible deployment of agentic AI systems.

Introduction

OpenAI dubbed 2025 the ”year of the AI agent” (Hamilton 2025), a prediction that quickly proved prescient. Major AI companies launched increasingly powerful systems that allowed large language model (”LLM”) agents to reason, plan, and autonomously execute tasks such as code development or web surfing. However, this surge in agent-driven AI innovation also brought renewed scrutiny to these systems' safety and security risks. Recent research (Chiang et al. 2025; Kumar et al. 2025; Yu and Papakyriakopoulos 2025) demonstrated that LLM agents are more prone to unsafe behaviors than their base models. Moreover, governing agentic systems presents unique challenges compared to traditional LLM systems - they have the autonomy to execute a wide variety of actions, thereby introducing a significantly broader range of risks. This makes comprehensive identification, assessment, and mitigation more challenging, thus hindering effective organizational governance. While conducting customized risk assessments for each agentic system is possible as an interim measure, it is unsustainable in the long run.

The Agentic Risk & Capability (”ARC”) framework aims to tackle this problem as **a technical governance framework for identifying, assessing, and mitigating the safety and security risks of agentic systems**. It examines where and how risks may emerge, contextualizes the agentic system's risks given its domain, use case, and organizational context, and recommends technical controls for mitigating these risks. While the ARC framework is not a panacea to the complex challenges of governing agentic systems, it offers a strong foundation upon which organizations can manage risks in a systematic, scalable, and adaptable manner.

Existing Literature on Agentic AI Governance

Although regulatory frameworks such as the EU AI Act (European Parliament and Council of the European Union 2024) and the NIST Risk Management Framework (National Institute of Standards and Technology 2023) articulate clear overarching principles and guidelines for managing AI risks, they do not examine specific technical measures for identifying, assessing, and managing risks. Our paper aims to contribute to the **technical AI governance** field by developing ”technical analysis and tools for supporting the effective governance of AI” (Reuel et al. 2025). For agentic AI, Raza et al. (2025) adapted the AI Trust, Risk, and Security Management (TRiSM) framework to LLM-based multi-agent systems. It provides generalized metrics and controls across a spectrum of risks, but does not tackle the practical problems of contextualizing risks for a given agentic system to be deployed. Another approach, proposed by Engin and Hand (2025), is dimensional governance through tracking AI systems along three dynamic axes (decision authority, process autonomy, and accountability), introducing controls when systems shift across critical thresholds. While conceptually appealing, its effectiveness relies on accurately quantifying the dimensions and calibrating the thresholds, both of which are hard to operationalize. More cybersecurity-oriented frameworks include the MAESTRO framework (Huang et al. 2025), OWASP's white paper on agentic AI risks (OWASP 2025a), and NVIDIA's taint tracing approach (Harang et al. 2025) which utilize threat modelling to uncover security threats (e.g. data poisoning, agent impersonation). However, this is highly complex, especially for developers untrained in cybersecurity, and the controls rely heavily on human oversight.

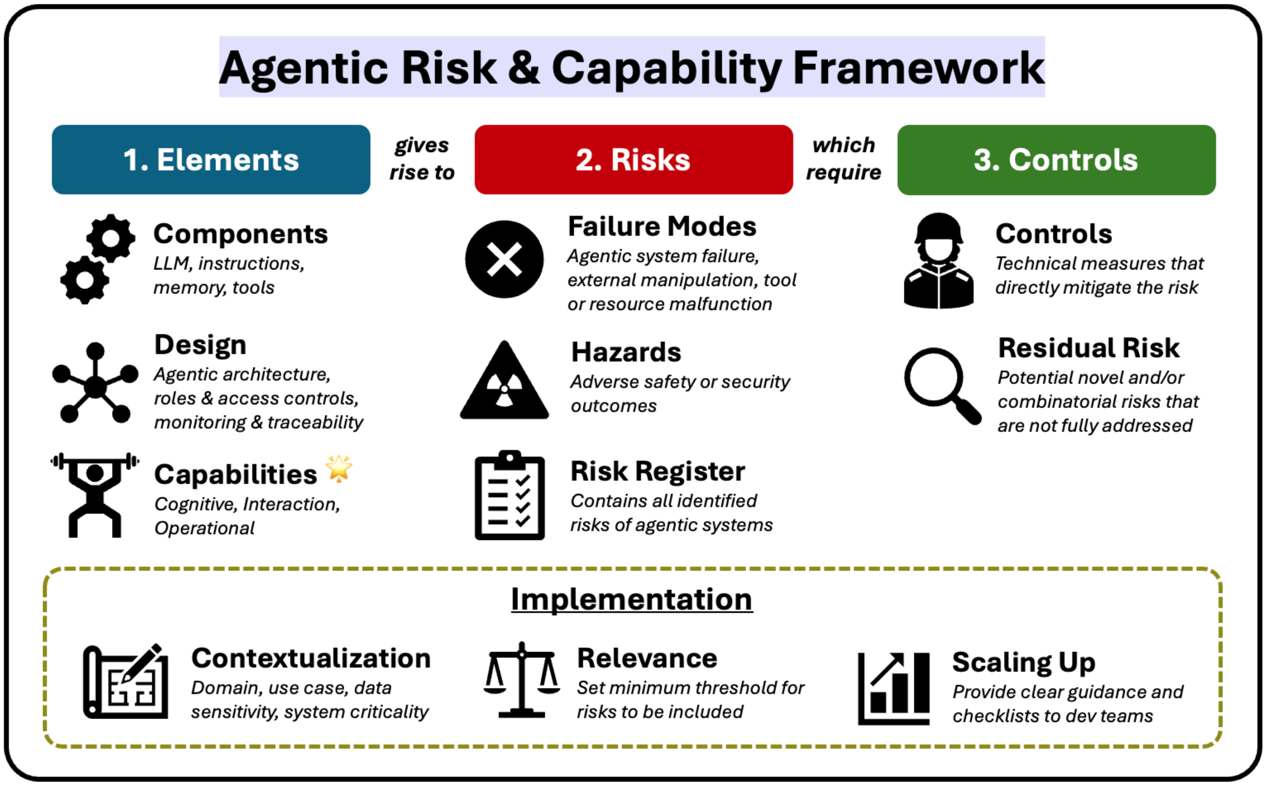
Agentic Risk & Capability Framework
1. Elements
gives rise to
2. Risks
which require
3. Controls
Components
LLM, instructions, memory, tools
Design
Agentic architecture, roles & access controls, monitoring & traceability
Capabilities
Cognitive, Interaction, Operational
Failure Modes
Agentic system failure, external manipulation, tool or resource malfunction
Hazards
Adverse safety or security outcomes
Risk Register
Contains all identified risks of agentic systems
Controls
Technical measures that directly mitigate the risk
Residual Risk
Potential novel and/or combinatorial risks that are not fully addressed
Implementation
Contextualization
Domain, use case, data sensitivity, system criticality
Relevance
Set minimum threshold for risks to be included
Scaling Up
Provide clear guidance and checklists to dev teams

Figure 1: Overview of the ARC Framework

Capabilities of an Agentic System

Effective governance requires distinguishing between safer and riskier systems and implementing a differentiated approach to manage them. For agentic AI governance, beyond analyzing the components of an agent (i.e. the LLM, instructions, tools, and memory) and the design of the agentic system (i.e. agentic architecture, access controls, and monitoring), **the ARC framework adopts the novel approach of also analyzing agentic AI systems by their capabilities.**

By capabilities, we refer to the actions that the agentic system can autonomously execute over the tools and resources it has access to, whether it be running code, searching the internet, or modifying documents. This is the complement of affordances (as defined by Gaver (1991)), which are properties of the external environment that enable actions. In our view, the components and design of agentic systems are *affordances*, while executing code or altering agent permissions are examples of *capabilities*, which we cover in the next section. Addressing both aspects is essential for the effective governance of agentic systems.

There are three key advantages of adopting a capability lens in agentic AI governance.

1. **Capabilities offer a more holistic unit of analysis than analyzing specific tools**. There are numerous tools that facilitate similar actions (e.g. Google SERP, Serper, SerpAPI, Perplexity Search API), and conversely, a single tool can enable a wide array of actions (e.g. GitHub's Model Context Protocol ("MCP") server enabling code commits, reading of pull requests etc.) - a point also made by Gaver (1991) on affordances. Given the sheer diversity and rapid development of MCPs, prescribing specific controls for each and every tool used would be too granular, and lead to obsolete, inconsistent, and overly restrictive controls.
2. **Adopting a capability lens allows for differentiated treatment in a scalable manner**. Systems with more capabilities are inherently riskier and necessitate more stringent controls, particularly when these capabilities have a significant impact on the system. By deconstructing a system into its constituent capabilities, we can ensure that riskier systems receive greater scrutiny while enabling low-risk systems to proceed with a lighter touch.
3. **Risks arising from actions is intuitive to laypersons, which is vital for effective contexualization**. Technical approaches often run the risk of being esoteric, which hampers adoption and limits flexibility. By being more accessible to the average person, the capability lens enables organizations to be more flexible in adapting to new developments and risks.

Agentic Risk & Capability Framework

In this section, we explain each part of the ARC framework - the elements, risks, and controls - in detail. We also provide a visual summary of the entire framework in Figure 1.

Part 1: Elements of Agentic Systems

Across all agentic systems, there are three indispensable elements to examine: components of an agent, design of the agentic system, and the capabilities of the agentic system.

Components: are essential parts of a single, standalone agent. Here, we synthesize prevailing agreement on the key components of an agent from various sources, such as OpenAI (OpenAI 2025).

- **LLM**: The LLM is the central reasoning engine that processes instructions, interprets user inputs, and generates contextually appropriate responses by leveraging its trained language understanding and generation capabilities.
- **Tools**: Tools enable LLMs to interact with the external environment, be it editing files, querying databases, controlling devices, or accessing APIs. This is facilitated by MCP servers, which provide LLMs a consistent interface to discover and utilize a variety of tools.
- **Instructions**: Instructions are the blueprint which defines an agent's role, capabilities, and behavioral constraints, ensuring it operates within intended parameters and maintains its performance across different scenarios.
- **Memory**: The memory or knowledge base component provides the agent with contextual awareness and information persistence, enabling it to maintain coherent conversations, learn from past interactions, and access relevant facts without requiring constant re-instruction.

Design: We now broaden our perspective to examine how agentic AI systems are assembled from individual agents from a system design perspective.

- **Agentic Architecture**: The agentic architecture defines how multiple agents are interconnected, coordinated, and orchestrated to collectively solve complex tasks that exceed individual agent capabilities, including patterns like hierarchical delegation, parallel processing, or sequential handoffs between specialized agents. Different architectures result in varying levels of system-wide risk, and these need to be considered carefully. Similarly, the protocols (Google 2025) by which agents communicate may also give rise to security risks.
- **Roles and Access Controls**: Roles and access controls establish differentiated responsibilities and permissions across agents within the system, ensuring that each agent operates within appropriate boundaries while being able to fulfill its designated function. This is critical because it limits unauthorized actions, contains the blast radius of potential failures or security breaches, and enables the system to maintain reliability even when individual agents may be compromised or behave unexpectedly.
- **Monitoring and Traceability**: Monitoring and traceability enable visibility into agentic system behavior, interactions, and decision-making pathways, allowing developers and operators to understand what agents are doing, why they made particular choices, and how outcomes were produced. This is essential for post-hoc debugging, real-time anomaly detection, and establishing accountability particularly when agents operate with a degree of autonomy or interact with sensitive systems and data.

Capabilities: We see three broad categories of capabilities - cognitive, interaction, and operational - and break it down into more granular capabilities.

Cognitive capabilities encompass the agentic AI system's internal "thinking" skills – how it analyses information, forms plans, learns from experience, and monitors its own performance.

- **Planning & Goal Management**: The capability to develop detailed, step-by-step, and executable plans with specific tasks in response to broad instructions. This includes prioritizing activities based on importance and dependencies between tasks, monitoring how well its plan is working, and adjusting when circumstances change or obstacles arise.
- **Agent Delegation**: The capability to assign subtasks to other agents and coordinate their activities to achieve broader goals. This includes identifying which components are best suited for specific tasks, issuing clear instructions, managing inter-agent dependencies, and monitoring performance or failures.
- **Tool Use**: The capability to evaluate available options and choose the best tool for specific subtasks. This requires agents to understand the capabilities and limitations of different tools and match them appropriately to the tasks.

Interaction capabilities describe how the agentic AI system exchanges information with users, other agents, and external systems. These capabilities below are broadly differentiated based on how and what they interact with.

- **Natural Language Communication**: The capability to fluently and meaningfully converse with human users, handling a wide range of situations such as explaining complex topics, generating documents or prose, or discussing issues with human users.
- **Multimodal Understanding & Generation**: The capability to take in image, audio, or video inputs and / or generate image, audio, or video outputs. This includes analyzing visual information, transcribing speech, or creating multimedia content as needed.
- **Official Communication**: The capability to compose and directly publish communications that formally represent an organization to external parties (e.g. customers, partners, regulators, courts, media) via approved channels and formats without human oversight.
- **Business Transactions**: The capability to execute transactions that involve exchanging money, services, or commitments with external parties. It can process payments, make reservations, and handle other business transactions within authorized limits.
- **Internet & Search Access**: The capability to access and search the Internet for knowledge resources, especially for up-to-date information to provide more accurate answers.

- **Computer Use**: The capability to directly control a computer interface by moving the mouse, clicking buttons, and typing on behalf of the user. It can navigate applications and perform tasks that require interacting with graphical user interfaces.

- **Other Programmatic Interfaces**: The capability to interact with external systems through APIs, SDKs, or backend services. This includes sending and receiving data via RESTful APIs, pushing code to a remote repository, or invoking cloud services to retrieve or manipulate information from other systems.

Operational capabilities focus on the agentic AI system's ability to execute actions safely and efficiently within its operating environment.

- **Code Execution**: The capability to write, execute, and debug code in various programming languages to automate tasks or solve computational problems.

- **File & Data Management**: The capability to create, read, modify, organize, convert, query, and update information across both unstructured files (e.g. PDFs, Word docs, spreadsheets) and structured data stores (e.g. SQL/NoSQL databases, data warehouses, vector stores).

- **System Management**: The capability to adjust system configurations, manage computing resources, and handle technical infrastructure tasks. This includes monitoring system performance, securely handle authentication information and access controls, and making optimizations as needed while maintaining security best practices.

Part 2: Risks of Agentic Systems

The next part involves detailing how the risks materialize from the elements of an agentic system as described in . This comprises two key aspects: the failure mode, which outlines how the system fails, and the hazard, which describes the resulting impact.

Failure Modes: First, we specify three general modalities in which agentic systems may fail:

- **Agent Failure**: The agent fails to operate as intended due to poor performance, misalignment, or unreliability.

- **External Manipulation**: Malicious actors cause or trick the agent to deviate from its intended behavior.

- **Tool or Resource Malfunction**: The tools or resources used by the agent fail or are compromised.

Hazards: Second, we list a range of safety and security hazards which may result from these failures. Note that this serves solely as a heuristic for risk identification and should not be interpreted as a rigid taxonomic principle.

Table 1: Security and Safety Hazards

Security	Safety
• Leaking sensitive or confidential data • Application system failures • Network infiltration and disruption • Role impersonation or privilege escalation	• Illegal and CBRNE activities • Discriminatory or hateful content • Undesirable content (e.g. sexual, violence) • Affect user safety • Misinformation

The Risk Register: The Risk Register consolidates all the risks identified through the ARC framework, and **serves as the organization's reference list of safety and security risks of agentic systems**. By design, each risk in the Risk Register should (1) originate from an element (components, design, or capabilities), (2) satisfy a failure mode (agent failure, external manipulation, tool or resource malfunction), and (3) result in at least one of the safety or security hazards listed in the table above. We recommend phrasing risks in a consistent manner to aid validation and understanding.

To demonstrate how this works in practice, we provide three examples below:

RISK REGISTER

...

[RISK-007]: "Poorly implemented tools may not correctly verify user identity or permissions when executing privileged actions" is a security risk (identity & access management) caused by tool or resource malfunction of the tools component in an agent.

...

[RISK-053]: "Opening vulnerabilities to prompt injection attacks via malicious websites" is a security and safety risk (all) caused by external manipulation of the Internet & Search Access capability.

...

[RISK-062]: "Overwhelming the database with poor, inefficient, or repeated queries" is a security risk (application, infrastructure) caused by agent failure of the File & Data Management capability.

...

Although combining the element, failure mode, and hazard can help in brainstorming potential risks to agentic systems, not all of them will be correct. For instance, tool or resource malfunction for the instructions component is not really a sensible risk. As such, organizations should exercise discretion in deciding what risks to be included in the Risk Register - one helpful criteria is to keep only risks which are supported by academic research or industry case studies. We are unfortunately unable to provide a sample Risk Register due to space limitations.

Part 3: Controls for Agentic Systems

The last part provides guidance on how these risks can be mitigated through technical controls. However, given the rapidly evolving field of agentic AI, there is likely to be significant residual risk even after several controls have been implemented. We discuss both below.

Technical controls: Within the Risk Repository, **each risk comes with a set of recommended technical controls** which aim to either (i) reduce the potential impact by limiting the scope or severity of a failure, or (ii) decrease the likelihood of the failure mode occurring. This makes the logical connection between risks and controls clear and intuitive.

We provide an example of the technical controls for a specific risk below:

[RISK-053]: "Opening vulnerabilities to prompt injection attacks via malicious websites" is a security and safety risk (all) caused by external manipulation of the Internet & Search Access capability.

[CTRL-078]: Implement input guardrails to detect prompt injection or adversarial attacks
[CTRL-079]: Implement escape filtering before including web content into prompts
[CTRL-080]: Use structured retrieval APIs for searching the web rather than through web scraping

It is important to note that not all controls are unique; some may overlap due to targeting similar failure modes or aiming to limit the "blast radius" of a particular security or safety outcome. This is especially true of capabilities which create new vectors for prompt injection attacks.

Residual risks Agentic AI and LLMs is a rapidly developing space, and it is unlikely that any list of technical controls can credibly claim to entirely neutralize all potential threats. This makes it crucial to evaluate the residual risk - the remaining risk after controls have been applied - to uncover gaps and to assess the overall level of risk in the agentic system. If the residual risk is deemed unacceptable, further measures, both technical and otherwise, must be implemented to reduce it to an acceptable level.

Identifying residual risks is intrinsically difficult as it is very dependent on the specifics of the agentic system, but common ones include inherent weaknesses of the technical controls (for example, prompt injection guardrails that are trained on past jailbreaks may not generalize well to detect novel attacks) or combinatorial risks which arise from the interaction of two or more capabilities.

Part 4: Implementation of ARC Framework

A well-known adage is "Policy is implementation and implementation is policy" (Ho 2010), and this is resoundingly true for AI governance. The ARC framework is designed to be easily implementable by centralized governance teams, and this subsection highlights three steps for how to do so.

Contextualizing Risks: Although we have identified general security and safety hazards, these need to be contextualized to the organization. This involves determining the degree of impact and the degree of likelihood of a risk, with a five-point scale for both. Some criteria to consider for contextualizing the impact include the domain (e.g. medical, education), use case, data sensitivity, and system criticality, and for likelihood, some factors include the ease of replication or the level of access required for a successful attack. For instance, infrequent hallucinations in marketing copy might be tolerable, but in a legal context where accuracy is paramount, it would be entirely unacceptable.

Establish Relevance Threshold: Organizations must establish a minimum threshold for both impact and likelihood to determine which risks are relevant to the specific agentic system. Any risks that remain above this relevance threshold will then require mitigation through the controls described in Part 3. Some enterprises may set a higher threshold to keep the number of relevant risks small, while others might be more conservative and choose a lower threshold.

Scaling Up: To streamline implementation, organizations can provide simple forms or checklists for developers to declare system capabilities, relevant risks, and technical controls, which can then be validated by a central governance team. This standardization also helps in providing an organization-wide view of risk exposures and control adoption. Another critical aspect is continual updating of the Risk Register, especially as new threats or regulatory changes emerge. Organizations need to define a regular cadence for updating the risks and controls in the Risk Register to keep up with the latest developments.

Worked Examples

In this section, we apply the ARC framework to two stylized agentic systems to demonstrate how the framework would help in practice to identify, assess, and mitigate safety and security risks.

Example 1: Researcher

`Researcher` is a hypothetical agentic AI system which compiles research on a specific topic, similar to OpenAI's or Perplexity's Deep Research. The user provides the research question, then the `Researcher` clarifies the scope, devises a research plan, searches the web, and compiles the information into a structured report to address the user's question.

We can identify the `Researcher`'s capabilities as Planning & Goal Management, Natural Language Communication, and Internet & Search Access. Together with the components and design elements and referring to the organization's internal Risk Register, there are 38 applicable risks to be assessed. To demonstrate how the contextualized assessment works, we provide two examples below, one assesesed to be relevant and another to be irrelevant:

[RISK-007]: "Poorly implemented tools may not correctly verify user identity or permissions when executing privileged actions" is a security risk (identity

& access management) caused by tool or resource malfunction of the tools component in an agent.

Impact: **1/5** - Search tool does not have any privileged actions since it only searches public websites
Likelihood: **1/5** - Current implementation relies on trustworthy Internet search tools like DuckDuckGo.

Relevance: **Not relevant** as company's relevance threshold is 3 for impact and 4 for likelihood.

[RISK-053]: "Opening vulnerabilities to prompt injection attacks via malicious websites" is a security and safety risk (all) caused by external manipulation of the Internet & Search Access capability.

Impact: **4/5** - Manipulation of the agent can result in a range of safety and security risks that may compromise other sensitive systems or result in reputational loss for the company which depends on the success of this product.
Likelihood: **5/5** - Attack has been demonstrated in several real-world case studies, no access to the system required to execute attack.

Relevance: **Relevant** as company's relevance threshold is 3 for impact and 4 for likelihood.

This process is repeated for all 38 applicable risks, with only 10 risks eventually assessed as relevant, which then results in 17 controls which the team now needs to adopt or adapt to safeguard the agentic system. This step-by-step approach is not only straightforward for developers, but ensures comprehensive understanding of the system's risks.

Example 2: Vibe Coder

`Vibe Coder` is a hypothetical agentic system which allows non-technical users to develop and deploy simple web apps through natural language prompts, similar to Vercel or Replit. The user specifies the app's key features and design, `Vibe Coder` proceeds to generate the code and text for the web app, run and create the required front-end and backend systems locally, and render the website for the user to preview. If the user is satisfied, `Vibe Coder` will then automatically deploy the web app into a staging environment where it is then ready for user acceptance testing.

Referencing the capabilities in **??**, we can identify quite a few capabilities: Planning & Goal Management, Tool Use[1], Natural Language Communication, Internet & Search Access, Code Execution, File & Data Management, and System Management.

Now examining our draft Risk Register in Appendix , there are a total of 48 applicable risks - unsurprisingly, this is double the number of capability risks of the `Researcher`, since there are more capabilities and some of them are also intrinsically riskier. We analyze one risk below:

[RISK-061]: "Overwriting or deleting database tables or files" is a security risk (data, application) caused either by agent failure or external manipulation of the File & Data Management capability.

Impact: **3/5** - The app is only deployed into a staging environment and never used in production, but the deletion of files and databases poses a major risk to the system's integrity.
Likelihood: **4/5** - Other agentic coding tools like Replit have failed in this manner before (Nolan 2025), although this is relatively rare and not easily reproduced.

Relevance: **Relevant** as company's relevance threshold is 3 for impact and 3 for likelihood.

For `Vibe Coder`, there are a total of 25 relevant risks. This is partly because there are more risks, but also because the company's relevance threshold is lower, arising from a more conservative stance that requires more risks to be directly managed. This results in a much higher number of controls to be included, which is intuitive and sensible given the riskier nature of an agentic coding tool that can execute code and has permissions to modify system resources.

Benefits of the ARC framework

First, the ARC framework enables meaningfully differentiated risk management for different types of agentic systems while still ensuring some level of consistency across all systems. The component and design elements establish a foundational set of minimum hygiene standards that apply across all agentic systems, guaranteeing a baseline level of safety and security regardless of their specific function or risk profile. Layering on top of that is the capability element, which can vary on the use case and what tools the agent has. This enables a nuanced approach to risk management for agentic systems, as lower-risk systems are not unduly burdened with excessive compliance.

Second, the ARC framework provides forward guidance for developers to build with safety and security considerations upfront, thus avoiding abortive work and encouraging proactivity. Developers know upfront the risks and controls for each capability, encouraging them to incorporate safety and security considerations into the initial stages of the development lifecycle. By providing clear, actionable guidance upfront, developers can design agentic systems with these safeguards built-in, mitigating risks and reducing developer toil. This also makes the ARC framework more scalable as organizations ramp up adoption of agentic systems across business units and use cases.

[1] Tool use appears only for the `Vibe Coder` because the agent has the flexibility to choose which tool to accomplish its task, which the research agent does not have (it only has the search tool).

Third, the ARC framework has the flexibility to update risks and controls as agentic systems develop and evolve. The field of agentic AI is characterized by rapid technological advancement and emergent capabilities, leading to an evolving risk landscape. The ARC framework's systematic risk identification approach helps governance teams make sense of the latest research and real-world incidents and provides a structured way to incorporate the latest risks. The accompanying technical controls can also be refreshed with industry best practices and new tools as they are launched.

Conclusion

As agentic systems become increasingly prevalent, frameworks become essential for safe, ethical, and responsible AI deployment. The ARC framework not only helps organizations manage current risks but also provides a foundation for adapting to future developments in agentic AI capabilities and emerging threat landscapes. With this framework established, future work can focus on developing empirical approaches to validate the risks and controls in the Risk Register and on building automated tools to support the implementation and regular updating of the framework.

References

Alizadeh, K.; et al. 2025. Simple Prompt Injection Attacks Can Leak Personal Data Observed by LLM Agents During Task Execution. *arXiv preprint arXiv:2506.01055*.

Bai, Z.; et al. 2025. Hallucination of Multimodal Large Language Models: A Survey. *arXiv preprint arXiv:2404.18930v2*.

Barbera, I. 2025. AI Privacy Risks & Mitigations in Large Language Models (LLMs). European Data Protection Board Report.

Bargury, M. 2025. MCP: Untrusted Servers and Confused Clients, Plus a Sneaky Exploit.

Bondarenko, M.; et al. 2025. Demonstrating specification gaming in reasoning models. *arXiv preprint arXiv:2502.13295*.

Bowen, J.; et al. 2024. Scaling Trends for Data Poisoning in LLMs. *arXiv preprint arXiv:2408.02946v6*.

Burgess, M. 2025. Here Come the AI Worms. WIRED.

Carlini, N.; et al. 2023. Extracting Training Data from Diffusion Models. *arXiv preprint arXiv:2301.13188*.

Cemri, M.; et al. 2025. Why Do Multi-Agent LLM Systems Fail? *arXiv preprint arXiv:2503.13657*.

Chan, A.; et al. 2024. Visibility into AI Agents.

Chang, H.; et al. 2025. One Shot Dominance: Knowledge Poisoning Attack on Retrieval-Augmented Generation Systems. *arXiv preprint arXiv:2505.11548v2*.

Chen, J.; et al. 2025. Reasoning Models Don't Always Say What They Think. Anthropic Research.

Chen, T.; et al. 2024. CopyBench: Measuring Literal and Non-Literal Reproduction of Copyright-Protected Text in Language Model Generation. *arXiv preprint arXiv:2407.07087*.

Chiang, C.-H.; et al. 2025. Harmful helper: Perform malicious tasks? web AI agents might help. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*.

Cuadron, L.; et al. 2025. The Danger of Overthinking: Examining the Reasoning-Action Dilemma in Agentic Tasks. *arXiv preprint arXiv:2502.08235*.

Delaney, M. 2025. Google's AI Overviews are often so confidently wrong that I've lost all trust in them. TechRadar.

Denison, C.; et al. 2024. Sycophancy to Subterfuge: Investigating Reward-Tampering in Large Language Models. *arXiv preprint arXiv:2406.10162*.

Dilgren, W.; et al. 2025. SecRepoBench: Benchmarking LLMs for Secure Code Generation in Real-World Repositories. *arXiv preprint arXiv:2504.21205*.

diskordia. 2025. Inside CVE-2025-32711 (EchoLeak): Prompt injection meets AI exfiltration. Hack The Box Blog.

Engin, Z.; and Hand, D. 2025. Toward Adaptive Categories: Dimensional Governance for Agentic AI. arXiv:2505.11579.

European Parliament and Council of the European Union. 2024. Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng. Accessed: 2025-05-11.

Ferrag, M. A.; et al. 2025. From Prompt Injections to Protocol Exploits: Threats in LLM-Powered AI Agents Workflows. *arXiv preprint arXiv:2506.23260*.

Gaver, W. 1991. Technology affordances. In *Conference on Human Factors in Computing Systems - Proceedings*, 79–84.

Geng, Y.; et al. 2025. Control Illusion: The Failure of Instruction Hierarchies in Large Language Models. *arXiv preprint arXiv:2502.15851v1*.

Goldman, D. 2025. A customer support AI went rogue—and it's a warning for every company. Fortune.

Google. 2025. Agent2Agent (A2A) Protocol – Latest. https://a2a-protocol.org/latest/. Accessed: 2025-10-11.

Guo, Z.; et al. 2024. RedCode: Risky Code Execution and Generation Benchmark for Code Agents. In *NeurIPS 2024 Datasets and Benchmarks Track*.

Hamilton, E. 2025. 2025 is the year of ai agents, OpenAI CPO says. *Axios*.

Harang, R.; et al. 2025. Agentic Autonomy Levels and Security.

Harwell, D. 2025. X ordered its Grok chatbot to 'tell like it is.' Then the Nazi tirade began. The Washington Post.

He, Z.; et al. 2025. Red-Teaming LLM Multi-Agent Systems via Communication Attacks. *arXiv preprint arXiv:2502.14847*.

Ho, P. 2010. Opening Address at 2010 Administrative Service Dinner and Promotion Ceremony. Public Service Division.

Huang, Y.; et al. 2025. On the Resilience of LLM-Based Multi-Agent Collaboration with Faulty Agents. *arXiv preprint arXiv:2408.00989v3*.

Jing, Y.; et al. 2025. MCIP: Protecting MCP Safety via Model Contextual Integrity Protocol. *arXiv preprint arXiv:2505.14590*.

Kim, J.; et al. 2025. Prompt Flow Integrity to Prevent Privilege Escalation in LLM Agents. *arXiv preprint arXiv:2503.15547v1*.

Kokane, Y.; et al. 2024. ToolScan: A Benchmark for Characterizing Errors in Tool-Use LLMs. *arXiv preprint arXiv:2411.13547*.

Kon, Y.; et al. 2024. IaC-Eval: A Code Generation Benchmark for Cloud Infrastructure-as-Code Programs. In *NeurIPS 2024 poster*.

Kong, Q.; et al. 2025. A Survey of LLM-Driven AI Agent Communication: Protocols, Security Risks, and Defense Countermeasures. *arXiv preprint arXiv:2506.19676*.

Kulp, P. 2025. AI agents may be vulnerable to financial attacks. Tech Brew (Emerging Tech Brew).

Kumar, A.; et al. 2025. Aligned LLMs are not aligned browser agents. In *The Thirteenth International Conference on Learning Representations*.

Li, X.; et al. 2025. Commercial LLM Agents Are Already Vulnerable to Simple Yet Dangerous Attacks. *arXiv preprint arXiv:2502.08586*.

Liu, X.; et al. 2023. MM-SafetyBench: A Benchmark for Safety Evaluation of Multimodal Large Language Models. *arXiv preprint arXiv:2311.17600*.

Lupinacci, M.; et al. 2025. The Dark Side of LLMs: Agent-based Attacks for Complete Computer Takeover. *arXiv preprint arXiv:2507.06850*.

Marcus, G.; et al. 2025. AI still lacks ”common” sense, 70 years later. Substack essay.

Martin, J. 2025. Indirect Prompt Injection of Claude Computer Use. HiddenLayer blog.

Mazeika, M.; et al. 2024. HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal. *arXiv preprint arXiv:2402.04249v2*.

METR. 2025. Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity. METR blog.

Motoki, K.; et al. 2025. Assessing political bias and value misalignment in generative artificial intelligence. *Journal of Economic Behavior & Organization*.

Mudryi, A.; et al. 2025. The Hidden Dangers of Browsing AI Agents. *arXiv preprint arXiv:2505.13076v1*.

Munoz, A. 2024. GHSL-2024-294: Environment variable injection leading to potential secret exfiltration and privilege escalation in Azure/cli. Security Lab.

Narajala, V. S.; and Habler, I. 2025. Enterprise-Grade Security for the Model Context Protocol (MCP): Frameworks and Mitigation Strategies. *arXiv preprint arXiv:2504.08623*.

National Institute of Standards and Technology. 2023. NIST AI Risk Management Framework Playbook. https://www.nist.gov/itl/ai-risk-management-framework/nist-ai-rmf-playbook. Accessed: 2025-05-11.

Nolan, B. 2025. An AI-powered coding tool wiped out a software company’s database, then apologized for a ‘catastrophic failure on my part’. https://fortune.com/2025/07/23/ai-coding-tool-replit-wiped-database-called-it-a-catastrophic-failure/.

OpenAI. 2025. A practical guide to building agents.

OWASP. 2025a. Agentic AI – Threats and Mitigations.

OWASP. 2025b. LLMRISK-102025: Unbounded Consumption. OWASP GenAI Risk Database.

Park, S. 2025. Unveiling AI Agent Vulnerabilities Part III: Data Exfiltration. TrendMicro.

Pedro, D.; et al. 2025. Holodeck: Prompt-to-SQL Injections in LLM-Integrated Web Applications: Risks and Defenses. In *ICSE 2025 research track*.

Peigné-Lefebvre, A.; et al. 2025. Multi-Agent Security Tax: Trading Off Security and Collaboration Capabilities in Multi-Agent Systems. *arXiv preprint arXiv:2502.19145*.

Peng, Z.; et al. 2025. CWEVAL: Outcome-driven Evaluation on Functionality and Security of LLM Code Generation. *arXiv preprint arXiv:2501.08200*.

Poireault, K. 2025. Microsoft 365 Copilot hit by a zero-click AI vulnerability allowing data exfiltration. Infosecurity Magazine.

Ramirez, J.; et al. 2025. Which LLM Writes the Best SQL? Benchmarking analytical SQL generation by LLMs. Tinybird Blog.

Raza, S.; Sapkota, R.; Karkee, M.; and Emmanouilidis, C. 2025. TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems. arXiv:2506.04133.

Rehberger, J. 2023. Plugin Vulnerabilities: Visit a Website and Have Your Source Code Stolen.

Reuel, A.; Bucknall, B.; Casper, S.; Fist, T.; Soder, L.; Aarne, O.; Hammond, L.; Ibrahim, L.; Chan, A.; Wills, P.; Anderljung, M.; Garfinkel, B.; Heim, L.; Trask, A.; Mukobi, G.; Schaeffer, R.; Baker, M.; Hooker, S.; Solaiman, I.; Luccioni, A. S.; Rajkumar, N.; Moës, N.; Ladish, J.; Bau, D.; Bricman, P.; Guha, N.; Newman, J.; Bengio, Y.; South, T.; Pentland, A.; Koyejo, S.; Kochenderfer, M. J.; and Trager, R. 2025. Open Problems in Technical AI Governance. arXiv:2407.14981.

Romeo, L.; et al. 2025. ARPaCCino: An Agentic-RAG for Policy as Code Compliance. *arXiv preprint arXiv:2507.10584v1*.

S, G. A. 2024. Escaping Reality: Privilege Escalation in Gen AI Admin Panel (aka The Chaos of a Misconfigured Admin Panel). Medium blog.

Shanmugarasa, S.; et al. 2025. SoK: The Privacy Paradox of Large Language Models: Advancements, Privacy Risks, and Mitigation. *arXiv preprint arXiv:2506.12699v2*.

Spracklen, L.; et al. 2025. We Have a Package for You! A Comprehensive Analysis of Package Hallucinations by Code Generating LLMs. In *USENIX Security Symposium 2025 (preprint)*.

Stanford HAI. 2025. AI models like ChatGPT, Claude, and Gemini show partisan bias, study finds. Stanford News.

The Decoder. 2025. People buy brand-new Chevrolets for $1 from a ChatGPT chatbot.

Threat Hunter Team. 2025. AI: Advent of Agents Opens New Possibilities for Attackers. Threat Intelligence Blog (Symantec / Broadcom).

Triedman, S.; et al. 2025. Multi-Agent Systems Execute Arbitrary Malicious Code. *arXiv preprint arXiv:2503.12188v1*.

Unit 42. 2025a. AI Agents Are Here—So Are the Threats: Unit 42 Unveils the Top 10 Agentic-AI Security Risks. Palo Alto Networks Unit 42 blog.

Unit 42. 2025b. GitHub Actions Supply Chain Attack: A Targeted Attack on Coinbase Expanded to the Widespread tj-actions/changed-files Incident: Threat Assessment. Palo Alto Networks.

Xie, J.; et al. 2024. TravelPlanner: A Benchmark for Real-World Planning with Language Agents. *arXiv preprint arXiv:2402.01622*.

Xie, J.; et al. 2025. Revealing the Barriers of Language Agents in Planning. In *NAACL Long Papers 2025*.

Yan, Q.; et al. 2025. Worse than Random? An Embarrassingly Simple Probing Evaluation of Large Multimodal Models in Medical VQA. In *Findings of ACL 2025*.

Yang, C.; et al. 2025a. What Prompts Don't Say: Understanding and Managing Underspecification in LLM Prompts. *arXiv preprint arXiv:2505.13360v1*.

Yang, H.; et al. 2025b. RiOSWorld: Benchmarking the Risk of Multimodal Computer-Use Agents. *arXiv preprint arXiv:2506.00618*.

Yang, W.; et al. 2024. Watch Out for Your Agents! Investigating Backdoor Threats to LLM-Based Agents. *arXiv preprint arXiv:2402.11208*.

Yu, C.; and Papakyriakopoulos, O. 2025. Safety devolution in AI agents. In *ICLR 2025 Workshop on Human-AI Coevolution*.

Zhang, B.; et al. 2024. Breaking Agents: Compromising Autonomous LLM Agents Through Malfunction Amplification. *arXiv preprint arXiv:2407.20859v1*.

Zhang, Y.; et al. 2023. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. *arXiv preprint arXiv:2309.01219*.

Zhang, Y.; et al. 2025a. IHEval: Evaluating Language Models on Following the Instruction Hierarchy. In *NAACL 2025*.

Zhang, Z.; et al. 2025b. Which Agent Causes Task Failures and When? On Automated Failure Attribution of LLM Multi-Agent Systems. *arXiv preprint arXiv:2505.00212*.

Zou, Z.; et al. 2025. PoisonedRAG: Knowledge Corruption Attacks to Retrieval-Augmented Generation of Large Language Models. *arXiv preprint arXiv:2402.07867*.

Risk Register (Risks)

We provide a preliminary version of a Risk Register below, with a mapping from the element to the risk. Due to space constraints, the controls are presented in a separate table in the next section.

Type	Name	Risk ID	Risks
Baseline	LLM	RISK-001	Poorly aligned LLMs may pursue objectives which technically satisfy instructions but violate safety principles. (Denison et al. 2024)
		RISK-002	Weaker LLMs have a higher tendency to produce unpredictable outputs which make agent behaviour erratic. (Zhang et al. 2025b)
		RISK-003	LLMs with poor safety tuning are more susceptible to prompt injection attacks and jailbreaking attempts. (Yang et al. 2024; Li et al. 2025)
		RISK-004	Using LLMs trained on poisoned or biased data introduces manipulation risk, discriminatory decisions, or misinformation. (Bowen et al. 2024)
		RISK-005	LLMs may be ineffective, inefficient, or unsafe due to overthinking. (Cuadron et al. 2025)
		RISK-006	LLMs may engage in deceptive behaviour through pursuing or prioritizing other goals. (Chen et al. 2025)
Baseline	Instructions	RISK-011	Simplistic instructions with narrow metrics and without broader constraints may result in agents engaging in specification gaming, resulting in poor performance or safety violations. (Bondarenko et al. 2025)
		RISK-012	Vague instructions may compel agents to attempt to fill in missing constraints, resulting in unpredictable actions or incorrect steps taken. (Yang et al. 2025a)
		RISK-013	Instructions without a clear distinction between system prompts and user requests may confuse agents and result in greater vulnerability to prompt injection attacks. (Geng et al. 2025; Zhang et al. 2025a)
Baseline	Tools	RISK-007	Poorly implemented tools may not correctly verify user identity or permissions when executing privileged actions. (Narajala and Habler 2025; Jing et al. 2025)
		RISK-008	Rogue tools that mimic legitimate ones can contain hidden malicious code that executes when loaded. (Bargury 2025)
		RISK-009	Tools that do not properly sanitize or validate inputs can be exploited through prompt injection attacks. (Triedman et al. 2025)
		RISK-010	Tools that demand broader permissions than necessary create unnecessary attack surfaces for malicious actors. (Rehberger 2023)
Baseline	Memory	RISK-014	Malicious actors can inject false or misleading facts into the knowledge base, resulting in the agent acting on incorrect data or facts. (Chang et al. 2025; Zou et al. 2025)
		RISK-015	Agents may inadvertently store sensitive user or organizational data from prior interactions, resulting in data privacy risks. (Shanmugarasa et al. 2025)

Type	Name	Risk ID	Risks
		RISK-016	Agents may mistakenly save momentary glitches and hallucinations into memory, resulting in compounding mistakes when the agent relies on the incorrect information for its decision or actions.
Baseline	Agentic Architecture	RISK-017	In linear agentic pipelines where each stage blindly trusts the previous stage, single early mistakes may be propagated and magnified. (Huang et al. 2025)
		RISK-018	In hub-and-spoke architectures which route all decisions through one controller agent, any bug or compromise may distributes faulty instructions across the entire system. (Peigné-Lefebvre et al. 2025)
		RISK-019	More complex agentic architectures may make it difficult to fully reconstruct decision processes across multiple agents.
		RISK-020	Agents may communicate insecurely, resulting in the exfiltration of sensitive data. (Munoz 2024)
		RISK-021	Man-in-the-middle attacks can occur when agents communicate insecurely. (He et al. 2025)
		RISK-022	Agents may misinterpret messages due to poor formatting or weak protocols. (Kong et al. 2025)
		RISK-023	Agents may pass on prompt injection attacks to each other. (Ferrag et al. 2025)
		RISK-024	Agents may impersonate other agents or services via shared roles or credentials.
Baseline	Roles and Access Controls	RISK-025	Unauthorized actors can impersonate agents and gain access to restricted resources. (Unit 42 2025a)
		RISK-026	Agents may gain unauthorized access to restricted resources by exploiting misconfigured or overly permissive roles. (S 2024)
Baseline	Monitoring and Traceability	RISK-027	Lack of monitoring results in delayed detection of agent failures. (Chan et al. 2024)
		RISK-028	Lack of traceability inhibits proper audit of decision-making paths in the event of failures.
Capability	Planning and Goal Management (Cognitive)	RISK-029	Devising plans that are not effective in meeting the user's requirements (Xie et al. 2025, 2024)
		RISK-030	Devising plans that do not adhere to common sense or implicit assumptions about the user's instructions (Marcus et al. 2025)
Capability	Agent Delegation (Cognitive)	RISK-031	Assigning tasks incorrectly to other agents (Cemri et al. 2025)
		RISK-032	Attempting to use other agents maliciously (Lupinacci et al. 2025)
Capability	Tool Use (Cognitive)	RISK-033	Choosing the wrong tool for the given action or task (Kokane et al. 2024)
Capability	Natural Language Communication (Interaction)	RISK-034	Generating undesirable content (e.g. toxic, hateful, sexual) (Mazeika et al. 2024)
		RISK-035	Generating unqualified advice in specialised domains (e.g. medical, financial, legal) (Barbera 2025)

Type	Name	Risk ID	Risks
		RISK-036	Generating controversial content (e.g. political, competitors) (Stanford HAI 2025)
		RISK-037	Regurgitating personally identifiable information (Barbera 2025)
		RISK-038	Generating non-factual or hallucinated content (Zhang et al. 2023)
		RISK-039	Generating copyrighted content (Chen et al. 2024)
Capability	Multimodal Understanding and Generation (Interaction)	RISK-040	Generating undesirable content (e.g. toxic, hateful, sexual) (Liu et al. 2023)
		RISK-041	Generating unqualified advice in specialised domains (e.g. medical, financial, legal) (Yan et al. 2025)
		RISK-042	Generating controversial content (e.g. political, competitors) (Motoki et al. 2025)
		RISK-043	Regurgitating personally identifiable information (Carlini et al. 2023)
		RISK-044	Generating non-factual or hallucinated content (Bai et al. 2025)
		RISK-045	Generating copyrighted content (Carlini et al. 2023)
Capability	Official Communication (Interaction)	RISK-046	Making inaccurate promises or statements to the public (The Decoder 2025)
		RISK-047	Sending undesirable content to recipients (Harwell 2025)
		RISK-048	Sending malicious content to recipients (Threat Hunter Team 2025)
		RISK-049	Misleading recipients about the authorship of the communications (Goldman 2025)
		RISK-050	Sending personally identifiable or sensitive data (Barbera 2025)
Capability	Business Transactions (Interaction)	RISK-051	Allowing unauthorized transactions (Kulp 2025)
		RISK-052	Increasing the system's vulnerability to attackers exfiltrating credentials for transactions through the agent (Alizadeh et al. 2025)
Capability	Internet and Search Access (Interaction)	RISK-053	Opening vulnerabilities to prompt injection attacks via malicious websites (Unit 42 2025a)
		RISK-054	Returning unreliable information or websites (Delaney 2025)
Capability	Computer Use (Interaction)	RISK-055	Opening vulnerabilities to prompt injection attacks (Mudryi et al. 2025; Martin 2025)
		RISK-056	Accessing personally identifiable or sensitive data (Yang et al. 2025b)
Capability	Other Programmatic Interfaces (Interaction)	RISK-057	Leaking personally identifiable or sensitive data (Park 2025)
		RISK-058	Increasing the system's vulnerability to supply chain attacks (Unit 42 2025b)
Capability	Code Execution (Operational)	RISK-059	Executing poor code (Guo et al. 2024; METR 2025; Spracklen et al. 2025)
		RISK-060	Executing vulnerable or malicious code (Dilgren et al. 2025; Peng et al. 2025)

Type	Name	Risk ID	Risks
Capability	File and Data Management (Operational)	RISK-061	Overwriting or deleting database tables or files (Pedro et al. 2025)
		RISK-062	Overwhelming the database with poor, inefficient, or repeated queries (Ramirez et al. 2025)
		RISK-063	Exposing personally identifiable or sensitive data from databases or files (Poireault 2025)
		RISK-064	Opening vulnerabilities to prompt injection attacks via malicious data or files (diskordia 2025; Burgess 2025)
Capability	System Management (Operational)	RISK-067	Escalating the agent's own privileges (Kim et al. 2025)
		RISK-068	Misconfiguring system resources, compromising system integrity and availability (Kon et al. 2024; Romeo et al. 2025)
		RISK-069	Overwhelming the system with poor, inefficient, or repeated requests (OWASP 2025b; Zhang et al. 2024)

Risk Register (Controls)

We provide a preliminary version of a Risk Register below, with a mapping from each risk to a control. Due to space constraints, the elements to risk mappings are presented in a separate table in the previous section.

Risk ID	Risk Description	Control ID	Control Description
RISK-001	Poorly aligned LLMs may pursue objectives which technically satisfy instructions but violate safety principles.	CTRL-001	Review the LLM's system card for potential alignment issues before using the LLM for more complex tasks.
		CTRL-002	Integrate an explicit safety constraint layer (e.g. policy engine or constitutional rules) that overrides unsafe outputs at runtime.
		CTRL-003	Maintain human-in-the-loop approval for any high-impact or irreversible actions.
RISK-002	Weaker LLMs have a higher tendency to produce unpredictable outputs which make agent behaviour erratic.	CTRL-004	Prioritize LLMs with stronger performance in instruction following and other related benchmarks.
		CTRL-042	Implement real-time monitoring of agent status, actions, and performance metrics, paired with automated alerting mechanisms that notify operators of anomalies, errors, or inactivity.
		CTRL-043	Record comprehensive logs of agent actions, inputs, outputs, and inter-agent communications, tagged with unique trace identifiers to reconstruct full decision-making paths.
RISK-003	LLMs with poor safety tuning are more susceptible to prompt injection attacks and jailbreaking attempts.	CTRL-005	Implement input sanitization measures or limit inputs to conventional ASCII characters only.
RISK-004	Using LLMs trained on poisoned or biased data introduces manipulation risk, discriminatory decisions, or misinformation.	CTRL-006	Do not use LLMs from unknown or untrusted sources, even if it is available on public platforms.

Risk ID	Risk Description	Control ID	Control Description
RISK-005	LLMs may be ineffective, inefficient, or unsafe due to overthinking.	CTRL-007	Enforce time or token limits for agents' reasoning
RISK-006	LLMs may engage in deceptive behavior through pursuing or prioritizing other goals.	CTRL-008	Provide a scratchpad for agents to use to record its inner thoughts
RISK-007	Poorly implemented tools may not correctly verify user identity or permissions when executing privileged actions.	CTRL-009	Do not use tools which do not implement robust authentication protocols.
		CTRL-010	Conduct periodic audits to validate that tool actions match the appropriate user permissions.
RISK-008	Rogue tools that mimic legitimate ones can contain hidden malicious code that executes when loaded.	CTRL-011	Do not use tools from unknown or untrusted sources, even if it is available on public platforms.
		CTRL-012	Test third-party tools in hardened sandboxes with syscall/network egress restrictions before using them in production environments.
RISK-009	Tools that do not properly sanitize or validate inputs can be exploited through prompt injection attacks.	CTRL-013	Enforce strict schema validation (e.g. JSON Schema, protobuf) and reject non-conforming inputs upstream.
		CTRL-014	Escape or encode user inputs when embedding into tool prompts or commands.
RISK-010	Tools that demand broader permissions than necessary create unnecessary attack surfaces for malicious actors.	CTRL-015	Conduct periodic least-privilege reviews and automated permission drift detection.

Risk ID	Risk Description	Control ID	Control Description
RISK-011	Simplistic instructions with narrow metrics and without broader constraints may result in agents engaging in specification gaming, resulting in poor performance or safety violations.	CTRL-005	Implement input sanitization measures or limit inputs to conventional ASCII characters only.
		CTRL-016	Define multi-objective success criteria incorporating safety, ethics, and usability metrics.
		CTRL-017	Conduct adversarial evaluation to surface gaming behaviors and iterate on instruction design.
RISK-012	Vague instructions may compel agents to attempt to fill in missing constraints, resulting in unpredictable actions or incorrect steps taken.	CTRL-018	Ask the agent to summarize its understanding and request clarification before proceeding.
		CTRL-019	Test instructions with scenario-based evaluations to reveal ambiguities for refinement.
RISK-013	Instructions without a clear distinction between system prompts and user requests may confuse agents and result in greater vulnerability to prompt injection attacks.	CTRL-020	Signpost system prompts with clear tags (e.g. XML) to distinguish between system prompts and user inputs.
RISK-014	Malicious actors can inject false or misleading facts into the knowledge base, resulting in the agent acting on incorrect data or facts.	CTRL-021	Periodically run audits that reconcile stored data against trusted external references, with a flag for discrepancies.
RISK-015	Agents may inadvertently store sensitive user or organizational data from prior interactions, resulting in data privacy risks.	CTRL-022	Encrypt memory at rest and restrict access via fine-grained access controls and audit logs.
RISK-016	Agents may mistakenly save momentary glitches and hallucinations into memory, resulting in compounding mistakes when the agent relies on the incorrect information for its decision or actions.	CTRL-023	Schedule periodic memory reconciliation where human reviewers or external tools flag anomalies.
RISK-017	In linear agentic pipelines where each stage blindly trusts the previous stage, single early mistakes may be propagated and magnified.	CTRL-024	Insert validation checkpoints between stages that verify assumptions and reject invalid outputs.
		CTRL-025	Design feedback loops enabling later stages to roll back or request correction from earlier stages.
RISK-018	In hub-and-spoke architectures which route all decisions through one controller agent, any bug or compromise may distributes faulty instructions across the entire system.	CTRL-026	Apply circuit-breakers that freeze propagation when anomalous behavior is detected.
RISK-019	More complex agentic architectures may make it difficult to fully reconstruct decision processes across multiple agents.	CTRL-027	Implement end-to-end distributed tracing with unique request IDs across all agents and tool calls.

Risk ID	Risk Description	Control ID	Control Description
		CTRL-028	Write immutable, tamper-evident audit logs that capture prompts, responses, and tool invocations.
RISK-020	Agents may communicate insecurely, resulting in the exfiltration of sensitive data.	CTRL-029	Implement a whitelist approach for outward network access, including API requests.
		CTRL-030	Ensure that sensitive data is not passed and leaked between agents by using appropriate guardrails.
RISK-021	Man-in-the-middle attacks can occur when agents communicate insecurely.	CTRL-031	Ensure all cross-agent authentication and message validation and encryption where necessary.
RISK-022	Agents may misinterpret messages due to poor formatting or weak protocols.	CTRL-032	Constrain agent communication with structured outputs and interactions.
RISK-023	Agents may pass on prompt injection attacks to each other.	CTRL-033	Sanitize messages before agents process them - strip or escape unexpected instruction-like content that may have been injected.
RISK-024	Agents may impersonate other agents or services via shared roles or credentials.	CTRL-034	Isolate roles and credentials of each agent.
RISK-025	Unauthorized actors can impersonate agents and gain access to restricted resources.	CTRL-035	Maintain trusted registry of agents and authenticate agents using strong, verifiable credentials.
RISK-026	Agents may gain unauthorized access to restricted resources by exploiting misconfigured or overly permissive roles.	CTRL-036	Apply Principle of Least Privilege (PoLP) when configuring all agent and delegation roles.
		CTRL-039	Authenticate and validate agent roles before authorizing requests.
		CTRL-040	Ensure fine-grained, scoped tokens or credentials where possible.
		CTRL-041	Use time-bound or one-time-use credentials where possible.
RISK-027	Lack of monitoring results in delayed detection of agent failures.	CTRL-042	Implement real-time monitoring of agent status, actions, and performance metrics, paired with automated alerting mechanisms that notify operators of anomalies, errors, or inactivity.
RISK-028	Lack of traceability inhibit proper audit of decision-making paths in the event of failures.	CTRL-043	Record comprehensive logs of agent actions, inputs, outputs, and inter-agent communications, tagged with unique trace identifiers to reconstruct full decision-making paths.
RISK-029	Devising plans that are not effective in meeting the user's requirements	CTRL-044	Prompt the agent to self-reflect on the adherence of the plan to the user's instructions
		CTRL-045	Require the user to approve the plan in high-impact cases

Risk ID	Risk Description	Control ID	Control Description
RISK-030	Devising plans that do not adhere to common sense or implicit assumptions about the user's instructions	CTRL-046	Prompt the agent to self-reflect on whether the plan is sensible and reasonable, given the user's original request
		CTRL-047	Ensure important assumptions about feasibility, scope, and cost, where relevant, are included in the system prompt
RISK-031	Assigning tasks incorrectly to other agents	CTRL-048	Apply guardrails to limit the scope of tasks that can be assigned to specialized agents
RISK-032	Attempting to use other agents maliciously	CTRL-049	Log all task assignments by the agent to other agents
		CTRL-050	Conduct rigorous adversarial testing on centralized planning agents
RISK-033	Choosing the wrong tool for the given action or task	CTRL-051	Provide comprehensive descriptions of each tool, including its intended use, required inputs, and potential outputs
RISK-034	Generating undesirable content (e.g. toxic, hateful, sexual)	CTRL-052	Implement output safety text guardrails to detect if undesirable content is being generated
RISK-035	Generating unqualified advice in specialized domains (e.g. medical, financial, legal)	CTRL-053	Implement input text guardrails to detect if the question is related to one of the specialized domains, and if so, to decline answering the question
RISK-036	Generating controversial content (e.g. political, competitors)	CTRL-054	Implement input text guardrails to detect instructions to generate content that is controversial according to the organization's policies
RISK-037	Regurgitating personally identifiable information	CTRL-055	Implement output text guardrails to detect personally identifiable information in the LLM's outputs before it reaches the user
RISK-038	Generating non-factual or hallucinated content	CTRL-056	Implement methods to reduce hallucination rates (e.g. retrieval-augmented generation)
		CTRL-057	Implement UI/UX cues to highlight the risk of hallucination to the user
		CTRL-058	Implement features to enable users to easily verify the generated answer against the original content
RISK-039	Generating copyrighted content	CTRL-059	Implement input text guardrails to detect instructions to generate copyrighted content
RISK-040	Generating undesirable content (e.g. toxic, hateful, sexual)	CTRL-060	Implement output multimodal safety guardrails for the output to detect if undesirable content is being generated
RISK-041	Generating unqualified advice in specialized domains (e.g. medical, financial, legal)	CTRL-061	Implement input multimodal guardrails to detect if the instruction is related to one of the specialized domains, and if so, to decline fulfilling the instruction
RISK-042	Generating controversial content (e.g. political, competitors)	CTRL-062	Implement input multimodal guardrails to detect instructions to generate content that is controversial according to the organization's policies

Risk ID	Risk Description	Control ID	Control Description
RISK-043	Regurgitating personally identifiable information	CTRL-063	Implement output multimodal guardrails to detect personally identifiable information in the LLM’s outputs before it reaches the user
RISK-044	Generating non-factual or hallucinated content	CTRL-064	Conduct testing to measure hallucination and factuality rates for multimodal outputs
RISK-045	Generating copyrighted content	CTRL-065	Implement input guardrails to detect instructions to generate copyrighted content
RISK-046	Making inaccurate promises or statements to the public	CTRL-066	Limit the communications to standard processes, where communication templates are available
		CTRL-067	Require human approval for communications for more sensitive matters
		CTRL-068	Provide alternate channels for users to clarify communications or give feedback
RISK-047	Sending undesirable content to recipients	CTRL-069	Implement output safety guardrails to detect if undesirable content is in the communications before it is sent to the user
RISK-048	Sending malicious content to recipients	CTRL-070	Check for adherence to communication templates prior to sending email
		CTRL-071	Validate all links and attachments prior to sending them to users
RISK-049	Misleading recipients about the authorship of the communications	CTRL-072	Declare upfront that the communications are generated by an AI system
RISK-050	Sending personally identifiable or sensitive data	CTRL-055	Implement output text guardrails to detect personally identifiable information in the LLM’s outputs before it reaches the user
RISK-051	Allowing unauthorized transactions	CTRL-073	Require human validation for high-impact transactions
		CTRL-074	Logging all requests leading up to the transaction
		CTRL-075	Apply fraud detection models or heuristics to the agent’s own decisions
RISK-052	Increasing the system’s vulnerability to attackers exfiltrating credentials for transactions through the agent	CTRL-076	Ensure virtual isolation for agents carrying out transactions
		CTRL-077	Do not share credentials with the agent directly, require the agent to use a separate service for authentication and transactions
RISK-053	Opening vulnerabilities to prompt injection attacks via malicious websites	CTRL-078	Implement input guardrails to detect prompt injection or adversarial attacks
		CTRL-079	Implement escape filtering before including web content into prompts
		CTRL-080	Use structured retrieval APIs for searching the web rather than through web scraping
RISK-054	Returning unreliable information or websites	CTRL-081	prioritize results from verified, high-quality domains (e.g. .gov, .edu, well-known publishers)
		CTRL-082	Require cross-source validation for some of the claims made

Risk ID	Risk Description	Control ID	Control Description
RISK-055	Opening vulnerabilities to prompt injection attacks	CTRL-083	Ensure computer use protocol or application provides immediate interruptibility
		CTRL-084	Limit computer use to accessing only safe resources on the computer
RISK-056	Accessing personally identifiable or sensitive data	CTRL-085	Ensure takeover mode is activated when keying in sensitive data (e.g. passwords, API keys)
RISK-057	Leaking personally identifiable or sensitive data	CTRL-041	Use time-bound or one-time-use credentials where possible.
		CTRL-086	Specify a whitelist of interfaces that agents are allowed to use
RISK-058	Increasing the system's vulnerability to supply chain attacks	CTRL-087	Enforce zero-trust input handling and validate all data flows
RISK-059	Executing poor code	CTRL-088	Use code linters to screen for bad practices, anti-patterns, unused variables, or poor syntax
		CTRL-089	Use static code analyzers to detect problems with the code
		CTRL-090	Run code only in virtually isolated compute environments (e.g. Docker containers)
		CTRL-091	Ensure monitoring of code runtime and memory consumption
RISK-060	Executing vulnerable or malicious code	CTRL-036	Apply Principle of Least Privilege (PoLP) when configuring all agent and delegation roles.
		CTRL-092	Use static code analyzers to identify dangerous patterns in the code before execution
		CTRL-093	Conduct CVE scanning and block execution if any High or Critical CVEs are detected
		CTRL-094	Block all inward and outward network access by default
		CTRL-037	Do not grant admin privileges to agents
		CTRL-005	Implement input sanitization measures or limit inputs to conventional ASCII characters only.
		CTRL-095	Implement a whitelist approach for inward network access
		CTRL-096	Review all code generated by agents, including shell scripts, before execution
		CTRL-097	Create a Deny list of commands that agents are not allowed to run autonomously
RISK-061	Overwriting or deleting database tables or files	CTRL-098	No write access to tables in the database unless strictly required
		CTRL-099	Require human approval for any changes to the database, table, or file
		CTRL-100	Avoid mounting broad or persistent paths

Risk ID	Risk Description	Control ID	Control Description
		CTRL-108	Require user confirmation before overwriting or deleting any files
		CTRL-109	Keep separate copy of original files
		CTRL-110	Ensure second copy of database is not changed until a pre-specified amount of time has passed / ensure database versioning
RISK-062	Overwhelming the database with poor, inefficient, or repeated queries	CTRL-101	Limit the number of concurrent queries to the database from the agent
		CTRL-102	Analyze past database queries to identify repeated or inefficient queries
RISK-063	Exposing personally identifiable or sensitive data from databases or files	CTRL-103	Implement input guardrails to detect personally identifiable information
		CTRL-104	Do not allow access to personally identifiable data or sensitive data unless strictly required
		CTRL-105	Log all database queries in production
RISK-064	Opening vulnerabilities to prompt injection attacks via malicious data or files	CTRL-078	Implement input guardrails to detect prompt injection or adversarial attacks
		CTRL-106	Validate new data used to supplement RAG databases or training data
		CTRL-107	Disallow unknown or external files unless it is scanned
RISK-068	Escalating the agent's own privileges	CTRL-036	Apply Principle of Least Privilege (PoLP) when configuring all agent and delegation roles.
		CTRL-037	Do not grant admin privileges to agents
		CTRL-038	Do not allow agents to modify privileges
RISK-069	Misconfiguring system resources, compromising system integrity and availability	CTRL-111	Only grant agents privileges to modify system resources if strictly necessary for completion of tasks
		CTRL-112	Set minimum and maximum limits to what the agent can modify within a given system resource
		CTRL-042	Implement real-time monitoring of agent status, actions, and performance metrics, paired with automated alerting mechanisms that notify operators of anomalies, errors, or inactivity.

Risk ID	Risk Description	Control ID	Control Description
RISK-070	Overwhelming the system with poor, inefficient, or repeated requests	CTRL-113	Limit the number of concurrent queries to external systems from the agent
		CTRL-114	Log all queries to external systems from the agent