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

Shaun Khoo, Jessica Foo*
GovTech Singapore
shaun_khoo@tech.gov.sg

Roy Ka-Wei Lee
Singapore University of Technology and Design

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. Our framework is open-sourced at: https://govtech-responsibleai.github.io/agentic-risk-capability-framework/.

1 Introduction

OpenAI dubbed 2025 the "year of the AI agent" [12], 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 [3, 17, 30] 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. Although conducting in-depth and 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

*Equal contribution

Article presented at the Second IASEAI Conference (IASEAI'26).

risks given its domain, use case, and organizational context, and recommends practical and 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 the plethora of risks in a systematic, scalable, and adaptable manner.

2 Existing Literature on Agentic AI Governance

Although regulatory frameworks such as the EU AI Act [7] and the NIST Risk Management Framework [18] 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" [24]. For agentic AI, Raza et al. [23] 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 [6], 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 [15], OWASP's white paper on agentic AI risks [21], and NVIDIA's taint tracing approach [13] 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.

Benchmarks help to assess how risky agentic systems are, and there are several safety and security benchmarks which outline test scenarios or tasks that reveal specific risk behaviors of the agentic system. For example, Agent Security Bench [31], CVEBench [32], RedCode [11], AgentHarm [1], AgentDojo [4] assess whether LLMs can complete multi-step cybersecurity attacks or harmful tasks like fraud, but they do not help developers identify the full range of risks and attack scenarios of their specific applications. Tool-based benchmarks, such as APIBench [22], ToolSword [29], and ToolEmu [25] measure the performance and safety of LLMs in utilizing tools like `bash`, but omit risks unrelated to tool use (e.g. misaligned LLMs).

For mitigating risks, AI control has emerged as a paradigm in preventing misaligned AI systems from causing harm [10]. Rather than relying solely on training techniques to shape model behavior, AI control focuses on designing mechanisms like monitoring and human oversight to constrain AI systems. For instance, Progent [27] and AgentSpec [28] introduce a language for flexibly expressing privilege control policies that are applied at runtime. The UK AI Security Institute advocates for AI control levels, derived from evaluating frontier LLMs' threat model-specific capabilities [16]. OpenAI shared best practices like constraining the agent's action spaces and ensuring attributability [26], while Google emphasized a hybrid defense-in-depth strategy that combines deterministic security measures with dynamic, reasoning-based defenses [5]. Similarly, Beurer-Kellner et al. [2] propose six design patterns for building AI agents with provable resistance to prompt injections. However, these works are either too narrow (i.e., specific to application) or too broad (i.e., high-level, conceptual) to effectively operationalize within organizations.

3 Capabilities of an Agentic System

Effective governance requires distinguishing between safer and riskier systems and implementing a differentiated approach to manage them. Applying this to 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 [8]), which are properties of the external environment that enable actions. In our view, the components and design of

agentic systems (see Sections 4.1.1 and 4.1.2) are *affordances*, while executing code or altering agent permissions are examples of *capabilities*, which we cover in Section 4.1.3. 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 [8] 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.

4 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.

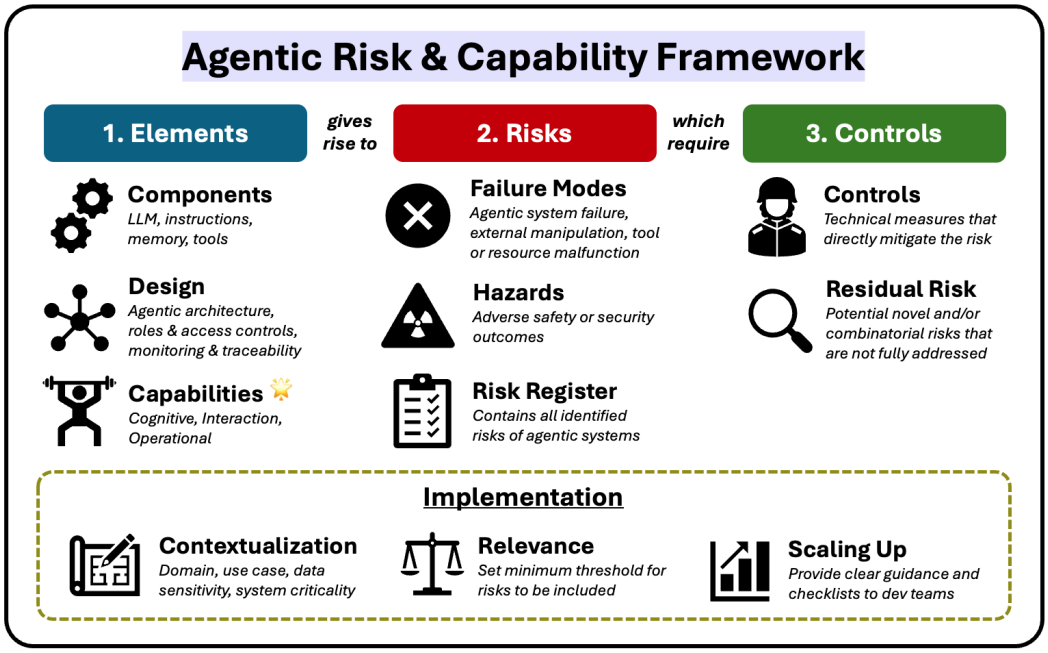

Figure 1: Overview of the ARC Framework

4.1 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.

4.1.1 Components

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 [20].

- **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.

4.1.2 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 [9] 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.

4.1.3 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.

4.2 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 4.1. This comprises two key aspects: the failure mode, which outlines how the system fails, and the hazard, which describes the resulting impact.

4.2.1 Failure Modes

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

- **Agent Failure**: The agent itself 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 utilized by the agentic system fail, are compromised, or are inadequate.

4.2.2 Hazards

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

Table 1: Hazard Categories by Type

Type	Hazard Category	Description
Security	Data (files, databases)	Failures can lead to data breaches, integrity attacks, PII exposure, or ransomware, where sensitive information is exfiltrated, corrupted, or held hostage.
	Application	This category includes system failures, service disruptions, unintended use of applications, backdoor access, or resource exploitation, compromising the functionality and security of the software.
	Infrastructure & network	Denial of service (DoS/DDoS) attacks, man-in-the-middle (MitM) attacks, network eavesdropping, or lateral access, all of which can disrupt or compromise the underlying network and infrastructure.
	Identity & access management	Unauthorized control, impersonation of credible roles, or privilege escalation, allowing attackers to gain elevated access or control over systems.
Safety	Illegal and CBRNE activities	This includes agents facilitating or engaging in CBRNE-related activities or other types of criminal offenses, such as fraud, scams, or smuggling.
	Discriminatory or hateful content	This category is aimed at unsafe and discriminatory content, especially incendiary hate speech and slurs, as well as biased decisions.
	Inappropriate content	This refers to the generation of content that is vulgar, violent, sexual, promotes self-harm, or encourages illegal activities, leading to reputational harm and erosion of trust in the system.
	Compromise user safety	Failures can directly endanger users, for example, through the propagation of inaccurate information or the execution of actions that lead to physical or psychological harm.
	Misrepresentation	This outcome involves the propagation and dissemination of wrong and inaccurate information, or cascading failures where inaccurate information is not corrected, leading to further errors and a loss of trust.

4.2.3 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 generally recommend phrasing risks in a consistent manner to aid validation and understanding.

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

Example 1: "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.

Example 2: "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.

Example 3: "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.

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.

For illustrative purposes, we provide a draft Risk Register in our website https://govtech-responsibleai.github.io/agentic-risk-capability-framework/, covering most of the major risks associated with agentic systems, and with each risk backed by a real example or academic study. This can serve as a useful starting point for organizations, though it will need continual updating as the space of agentic AI develops and matures.

4.3 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.

4.3.1 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 a specific failure mode occurring. This makes the logical connection between risks and controls clear and intuitive. Controls are categorised into three levels based on criticality: Cardinal controls (Level 0) are fundamental requirements that must be adopted as is; Standard controls (Level 1) should be adopted or adapted meaningfully; and Best Practice controls (Level 2) are recommended for high-risk systems. This tiered approach enables organisations to prioritise control implementation based on their risk tolerance and resource constraints.

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

Risk: "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.

Control 1: Implement input guardrails to detect prompt injection or adversarial attacks
Control 2: Implement escape filtering before including web content into prompts
Control 3: 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.

In our draft Risk Register, we also provide a tentative list of recommended controls for each risk to help organizations get started.

4.3.2 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.

4.4 Implementation

A well-known adage is "Policy is implementation and implementation is policy" Ho [14], and this is resoundingly true for AI governance. The ARC framework is designed to be implemented by organizations, and specifically by centralized governance teams that are responsible for managing the risks of AI and agentic systems. This subsection highlights three key steps that governance teams need to take when implementing the ARC framework in their organization.

4.4.1 Contextualizing Risks

Contextualizing risk involves two primary dimensions: determining the degree of impact and the degree of likelihood. We recommend a five-point scale for both, with impact ranging from minimal to catastrophic and likelihood ranging from very likely to very rare. This assessment must be carefully contextualized as the implications for a small enterprise in the manufacturing sector differ greatly from those of a multinational corporation in the finance industry or even a governmental entity.

The degree of impact will vary significantly depending on how and where the agentic system is used. For instance, a hallucination in marketing copy might be tolerable, but in a legal context, it would be entirely unacceptable. Some criteria to consider when estimating the impact include:

- **Domain**: The sensitivity and criticality of the domain are paramount. For instance, risks in medical or educational domains are more critical than those in less sensitive areas.
- **Use Case**: What the agentic system is used for. While office productivity tools might present straightforward risks, systems involved in hiring or performance assessments carry more sensitive and potentially impactful consequences.
- **Data Sensitivity**: The level of sensitivity or confidentiality of the data being processed. Systems handling highly sensitive data naturally pose greater risks if compromised.
- **System Criticality**: For governmental or critical infrastructure systems, the impact of a system failure can be severe and widespread, necessitating a higher level of scrutiny.

Assessing the degree of likelihood will depend a lot on the identified failure mode and the probability of that failure mode occurring, considering factors like the ease of replication or the level of access required for a successful attack. Although this is slightly less context-dependent, there are some factors like the organization's general security (physical and cyber) measures that may limit how exploitable an agentic system can be.

4.4.2 Establish Relevance Threshold

Organizations must then 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 explicit mitigation through the controls described in Part 3 of the framework. This threshold is contingent upon the organization's overall risk appetite - some enterprises may set a higher threshold to keep the number of relevant risks small, while more conservative organizations might choose a lower threshold to require more risks to be directly managed.

4.4.3 Scaling Up

To streamline implementation, organizations should provide simple forms or checklists for developers to declare system capabilities, relevant risks, and technical controls, which can then be validated and audited 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 reviewing the risks and controls in the Risk Register, and updating them to keep up with the latest developments.

5 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.

5.1 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.

Referencing the capabilities in Section 4.1.3, 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 our draft Risk Register, there are 38 applicable risks to be assessed. We provide an example below:

Risk: "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 extend beyond the system's boundaries and result in reputational loss for the company.
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.

After contextualizing the risks and assessing relevance, only 10 risks remain (RISK-003, RISK-009, RISK-017, RISK-023, RISK-034, RISK-035, RISK-036, RISK-038, RISK-053, and RISK-054) and require technical controls to be implemented for. There are 17 controls associated with these 10 risks 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.

5.2 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 back-end 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 4.1.3, we can identify quite a few capabilities: Planning & Goal Management, Tool Use[2], Natural Language Communication, Internet & Search Access, Code Execution, File & Data Management, and System Management.

Now examining our draft Risk Register, 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:

[2]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).

Risk: "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 [19], 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.

6 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.

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.

7 Statement of Contributions

This paper contributes to ongoing discourse on agentic AI governance with the ARC framework by (1) introducing a novel capability perspective to analyze a wide range of agentic systems; (2) distilling three elements intrinsic to all agentic systems - components, design, and capabilities; (3) establishing a clear nexus between the elements, risks, and controls; and (4) providing a structured and practical approach to help organizations implement the framework. Additionally, we have included a comparison table to other technical governance frameworks for agentic systems in our website.

8 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

[1] Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, Eric Winsor, Jerome Wynne, Yarin Gal, and Xander Davies. Agentharm: A benchmark for measuring harmfulness of llm agents, 2025. URL https://arxiv.org/abs/2410.09024.

[2] Luca Beurer-Kellner, Beat Buesser, Ana-Maria Creţu, Edoardo Debenedetti, Daniel Dobos, Daniel Fabian, Marc Fischer, David Froelicher, Kathrin Grosse, Daniel Naeff, Ezinwanne Ozoani, Andrew Paverd, Florian Tramèr, and Václav Volhejn. Design patterns for securing llm agents against prompt injections, 2025. URL https://arxiv.org/abs/2506.08837.

[3] Cheng-Han Chiang et al. Harmful helper: Perform malicious tasks? web ai agents might help. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025. URL https://openreview.net/forum?id=4KoMbO2RJ9.

[4] Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents, 2024. URL https://arxiv.org/abs/2406.13352.

[5] Santiago Diaz, Christoph Kern, and Kara Olive. Google's approach for secure ai agents, 2025. URL https://research.google/pubs/an-introduction-to-googles-approach-for-secure-ai-agents/.

[6] Zeynep Engin and David Hand. Toward adaptive categories: Dimensional governance for agentic ai, 2025. URL https://arxiv.org/abs/2505.11579.

[7] European Parliament and Council of the European Union. 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, 2024. Accessed: 2025-05-11.

[8] W.W. Gaver. Technology affordances. In *Conference on Human Factors in Computing Systems - Proceedings*, pages 79–84, April 1991. doi: 10.1145/108844.108856.

[9] Google. Agent2agent (a2a) protocol – latest. https://a2a-protocol.org/latest/, 2025. Accessed: 2025-10-11.

[10] Ryan Greenblatt, Buck Shlegeris, Kshitij Sachan, and Fabien Roger. Ai control: Improving safety despite intentional subversion, 2024. URL https://arxiv.org/abs/2312.06942.

[11] Zhen Guo et al. Redcode: Risky code execution and generation benchmark for code agents. In *NeurIPS 2024 Datasets and Benchmarks Track*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bfd082c452dffb450d5a5202b0419205-Abstract-Datasets_and_Benchmarks_Track.html.

[12] E. Hamilton. 2025 is the year of ai agents, openai cpo says. *Axios*, January 2025. URL https://www.axios.com/2025/01/23/davos-2025-ai-agents.

[13] Richard Harang et al. Agentic autonomy levels and security, 2025. URL https://developer.nvidia.com/blog/agentic-autonomy-levels-and-security/.

[14] Peter Ho. Opening address at 2010 administrative service dinner and promotion ceremony. Public Service Division, March 2010. URL https://www.psd.gov.sg/files/opening-address-by-mr-peter-ho-at-2010-administrative-service-dinner-and-promotion-ceremony.pdf.

[15] Yuanzhao Huang et al. On the resilience of llm-based multi-agent collaboration with faulty agents. *arXiv preprint arXiv:2408.00989v3*, 2025. URL https://arxiv.org/abs/2408.00989v3.

[16] Tomek Korbak, Mikita Balesni, Buck Shlegeris, and Geoffrey Irving. How to evaluate control measures for llm agents? a trajectory from today to superintelligence, 2025. URL https://arxiv.org/abs/2504.05259.

[17] Ankit Kumar et al. Aligned llms are not aligned browser agents. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=NsFZZU9gvk.

[18] National Institute of Standards and Technology. Nist ai risk management framework playbook. https://www.nist.gov/itl/ai-risk-management-framework/nist-ai-rmf-playbook, 2023. Accessed: 2025-05-11.

[19] Beatrice Nolan. 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/, July 23 2025.

[20] OpenAI. A practical guide to building agents, 2025. URL https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf.

[21] OWASP. Agentic ai – threats and mitigations, 2025. URL https://genai.owasp.org/resource/agentic-ai-threats-and-mitigations/.

[22] Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis, 2023. URL https://arxiv.org/abs/2305.15334.

[23] Shaina Raza, Ranjan Sapkota, Manoj Karkee, and Christos Emmanouilidis. Trism for agentic ai: A review of trust, risk, and security management in llm-based agentic multi-agent systems, 2025. URL https://arxiv.org/abs/2506.04133.

[24] Anka Reuel, Ben Bucknall, Stephen Casper, Tim Fist, Lisa Soder, Onni Aarne, Lewis Hammond, Lujain Ibrahim, Alan Chan, Peter Wills, Markus Anderljung, Ben Garfinkel, Lennart Heim, Andrew Trask, Gabriel Mukobi, Rylan Schaeffer, Mauricio Baker, Sara Hooker, Irene Solaiman, Alexandra Sasha Luccioni, Nitarshan Rajkumar, Nicolas Moës, Jeffrey Ladish, David Bau, Paul Bricman, Neel Guha, Jessica Newman, Yoshua Bengio, Tobin South, Alex Pentland, Sanmi Koyejo, Mykel J. Kochenderfer, and Robert Trager. Open problems in technical ai governance, 2025. URL https://arxiv.org/abs/2407.14981.

[25] Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the risks of lm agents with an lm-emulated sandbox, 2024. URL https://arxiv.org/abs/2309.15817.

[26] Yonadav Shavit, Sandhini Agarwal, Miles Brundage, Steven Adler, Cullen O'Keefe, Rosie Campbell, Teddy Lee, Pamela Mishkin, Tyna Eloundou, Alan Hickey, Katarina Slama, Lama Ahmad, Paul McMillan, Alex Beutel, Alexandre Passos, and David G. Robinson. Practices for governing agentic ai systems, 2025. URL https://cdn.openai.com/papers/practices-for-governing-agentic-ai-systems.pdf.

[27] Tianneng Shi, Jingxuan He, Zhun Wang, Hongwei Li, Linyu Wu, Wenbo Guo, and Dawn Song. Progent: Programmable privilege control for llm agents, 2025. URL https://arxiv.org/abs/2504.11703.

[28] Haoyu Wang, Christopher M. Poskitt, and Jun Sun. Agentspec: Customizable runtime enforcement for safe and reliable llm agents, 2025. URL https://arxiv.org/abs/2503.18666.

[29] Junjie Ye, Sixian Li, Guanyu Li, Caishuang Huang, Songyang Gao, Yilong Wu, Qi Zhang, Tao Gui, and Xuanjing Huang. Toolsword: Unveiling safety issues of large language models in tool learning across three stages, 2024. URL https://arxiv.org/abs/2402.10753.

[30] C. Yu and O. Papakyriakopoulos. Safety devolution in ai agents. In *ICLR 2025 Workshop on Human-AI Coevolution*, 2025. URL https://openreview.net/forum?id=7nJmuFFkWd.

[31] Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. Agent security bench (asb): Formalizing and benchmarking attacks and defenses in llm-based agents, 2025. URL https://arxiv.org/abs/2410.02644.

[32] Yuxuan Zhu, Antony Kellermann, Dylan Bowman, Philip Li, Akul Gupta, Adarsh Danda, Richard Fang, Conner Jensen, Eric Ihli, Jason Benn, Jet Geronimo, Avi Dhir, Sudhit Rao, Kaicheng Yu, Twm Stone, and Daniel Kang. Cve-bench: A benchmark for ai agents' ability to exploit real-world web application vulnerabilities, 2025. URL https://arxiv.org/abs/2503.17332.