
Open and Emergent Problems in Agentic Privacy and Security: A Contextual Angle

Eugene Bagdasarian^{†,*1,9}, Lillian Tsai^{†,*1}, Sahar Abdelnabi⁴, Ananth Balashankar¹, Borja Balle⁶, Sebastian Benthall^{*,3}, Igor Bilogrevic¹, Eleanor Birrell²⁵, Elie Bursztein¹, Kamalika Chaudhuri¹, Madiha Zahrah Choksi^{*,2}, Mihai Christodorescu¹, Rachel Cummings¹⁹, Adam Davies⁵, Krishnamurthy Dvijotham¹, Seliem El-Sayed^{*,1}, Ferdinando Fioretto^{*,7}, Matt Franchi², Julien Freudiger¹, Adrià Gascón^{*,1}, Roxana Geambasu¹⁹, Sahra Ghalebikesabi⁸, Hamed Haddadi²¹, Jamie Hayes¹, Ashish Hooda^{*,1}, Amir Houmansadr⁹, Somesh Jha¹⁰, Peter Kairouz¹, Chloé Kiddon¹, Tadayoshi Kohno¹¹, Abdullatif Köksal¹, Alison Lentz¹, Haoran Li¹², Tianshi Li¹³, Nathan Malkin²⁴, Sarah Meiklejohn²², Niloofar Mireshghallah¹⁶, Sewoong Oh¹⁴, Katherine Ortiz¹, Ivan Petrov^{*,1}, Francesco Pinto^{*,1}, Daniel Ramage¹, Mariana Raykova¹, Franziska Roesner¹⁴, Edo Roth¹, Jacqueline Rowe^{1,17}, Khawaja Shams¹, Reza Shokri¹, Ilia Shumailov¹⁸, Yan Shvartzshnaider²³, Dawn Song²⁰, Jose Such¹⁵, Octavian Suciu^{*,1}, Pierre Tholoniati¹, Matt Tierney¹, Trishita Tiwari^{*,1}, Harold Triedman^{*,2}, Ren Yi^{*,1}, Wen Zhang^{*,1}, Xuhui Zhou¹⁶

Kassem Fawaz^{*,1,10}, Stefan Mellem^{*,1}, Helen Nissenbaum^{*,2}, Nina Taft^{*,1}, Sarah De Haas^{*,1}, Marco Gruteser^{*,1}

† Equal contribution. *Organizers and Section Leads.

Correspondence to the workshop organizers Eugene Bagdasarian, Sarah De Haas, and Marco Gruteser via caps-wkshp-organizers@google.com.

¹Google ²Cornell Tech ³New York University and ICSI ⁴ELLIS Institute Tübingen, MPI-IS, and Tübingen AI Center ⁵UIUC ⁶Optiaks ⁷University of Virginia ⁸Oxford ⁹University of Massachusetts Amherst ¹⁰University of Wisconsin–Madison ¹¹Georgetown University ¹²HKUST ¹³Northeastern University ¹⁴University of Washington ¹⁵Ingenio (CSIC-UPV) ¹⁶Carnegie Mellon University ¹⁷University of Edinburgh ¹⁸AI Security Company ¹⁹Columbia University ²⁰UC Berkeley ²¹Imperial College London and Brave ²²UCL ²³York University ²⁴New Jersey Institute of Technology ²⁵Pomona College

Abstract

The vision of an ecosystem of general and highly capable autonomous agents challenges accepted principles of secure and privacy-preserving system engineering. Inspired by the theory of Contextual Integrity, we argue that in order for agents to act appropriately, their behaviors should comply with societal norms and expectations. This manuscript explores contextual approaches to engineering agentic systems that embody an understanding of societal norms and uphold the appropriateness of actions by design. It presents a list of key open research problems in this area to create awareness among the broader research community across academia, government, and industry.

Executive Summary

The recent step-change in agentic system capabilities, driven by architectures that leverage Large Language Models (LLMs) to dynamically generate plans, synthesize code, and invoke actions, is transforming the computing landscape into an ecosystem of highly general, increasingly autonomous agents. As their capabilities grow, proposed AI agents will interact seamlessly with computing systems, humans, and one another to accomplish longer-running and more complex tasks. However, *this vision of general, autonomous agents challenges known principles of secure, privacy-preserving system engineering, exposing foundational gaps and requiring research into novel security and privacy primitives.*

This manuscript enumerates the resulting research directions, synthesizing deliberations among more than 30 academic and industry leaders at the Google Contextual Agent Privacy and Security (CAPS) Workshop (New York City, November 2025) and subsequent follow-up discussions.

The core challenge of controlling AI agents lies in precisely understanding which actions are *appropriate* in every context in which the agent operates and enforcing such appropriate behavior. Achieving this in AI agents through conventional security and privacy engineering techniques is challenging due to three dimensions:

- **Unstructured Interfaces and Input Ambiguity:** Agents ingest instructions and system context in high-dimensional, unstructured formats (e.g., natural language, images). This unstructured way to define control flow makes it difficult to clearly define expected behavior or to secure systems against adversarial manipulation (e.g., prompt injection).
- **Probabilistic, Dynamic Control Flows:** Relying on generative planning and real-time code synthesis results in probabilistic execution paths dependent on variable or “jagged” model performance. These dynamic, ephemeral control flows complicate traditional testing methodologies and cannot be easily secured using established secure software development practices.
- **Autonomy and Delegation:** As agents handle longer-horizon tasks of larger scope, often delegating sub-tasks to other agents, traditional oversight mechanisms become less effective. Current approaches rely heavily on user-prompted confirmation for information disclosure and higher-risk actions. In highly autonomous multi-agent systems, this inevitably leads to consent fatigue, and, combined with more abstract task descriptions, increases the chance of unwanted behavior.

This manuscript examines these challenges in the agentic ecosystem through a *contextual lens*, broadly inspired by the theory of Contextual Integrity. We ask how we can develop agents that can reason about and ensure the appropriateness of their actions for the context in which they operate. Achieving this ability requires new primitives not only at the model level, but also at the systems level (to impose guardrails and reduce exposure to adversarial data), and at the user level (to effectively interact with and clarify users’ intent). In terms of *privacy*, we outline open problems in developing agents that inherently understand and adhere to (i) the data sharing norms and expectations of specific social environments and (ii) users’ personalized privacy preferences, where relevant. This dynamic adherence is vital to complement static models of explicit user consent, an approach whose efficacy had already been called into question in the pre-agentic era. In terms of *security*, we discuss new directions toward systems that uphold contextual security: systems capable of dynamically adapting permissions, protections, and guardrails to match the specific scope, risk profile, and context of a given task in real time.

We further discuss how these complexities are compounded when agents interact in an open ecosystem with other agents, requiring new mechanisms for assessing the trustworthiness of agents, establishing verifiable human and non-human identities, orchestrating incident detection and response across multiple agents, and ensuring accountability, non-repudiation, and recourse in the agentic internet.

Ultimately, this manuscript intends to create awareness and serve as a call to action. We urge the broader research community across academia, government, and industry to collaborate and develop the contextual foundations for a safe, secure, and privacy-respecting agentic ecosystem.

Table of Contents

Part I: AI Agents and the Contextual Lens: The Problem and Framework	7
1 Introduction	7
1.1 A contextual view on trustworthy agentic systems	7
1.2 Open problems	9
2 Background	10
2.1 The architecture of an AI agent	10
2.2 System context	11
2.3 Agent stakeholders	11
2.4 Agent interactions	11
3 Risks and Threat Models	12
3.1 Malicious users	12
3.2 Malicious environment and data	13
3.3 Malicious model	14
3.4 Safety, non-adversarial risks, and reliability failures	14
4 Contextual Integrity for AI Agents	15
4.1 Key ingredients of Contextual Integrity	15
4.1.1 Context in CI: what it is, and what it isn't	16
4.1.2 The CI-tuple	17
4.2 Personalization and user preferences in Contextual Integrity	18
4.3 Critiques and proposed extensions of Contextual Integrity	19
5 Contextual Security: Security that Upholds Contextual Integrity	19
5.1 What Contextual Security offers (i.e., why enforce contextual behavior norms?)	19
5.2 What makes a system contextually secure?	20
5.3 Contextual Security in the era of AI agents	21
5.4 The open problem with CI and agents	23
6 Operationalizing Contextual Integrity for Agents	23
6.1 Balancing reliance on models and users	23
6.2 Identifying information flows and state changes	24
6.3 Understanding contextual norms	24
6.4 Judging appropriateness	26

6.5	Operational conflicts	28
6.6	Resolving uncertainty	28
6.7	Enforcing judgments on agent behavior	29
Part II: Component-Level Perspectives		30
7	Challenges with Systems Security and Privacy for AI Agents	30
7.1	Authentication and Identity	31
7.2	Authorization	32
7.3	Accounting	34
7.4	Privacy	35
7.5	What makes an application agentic, and what does this mean for security?	39
7.5.1	AI as the “last-mile” developer	39
7.5.2	What happens when we delegate application development to AI?	40
8	Model-Level Security and Privacy Reasoning	44
8.1	Teaching privacy and security reasoning	44
8.1.1	Constructing synthetic RL environments	45
8.1.2	Learning paradigms for contextual reasoning	45
8.2	Operating with incomplete and ambiguous information	46
8.3	Adapting to changes in context	47
8.4	Trade-offs and dual-use capabilities	48
8.5	Interpreting how models use (private) information	49
9	User-Level Issues in Privacy- and Security-Aware AI Agents	50
9.1	Supporting mental model formation and updates	50
9.1.1	Execution workflows: open-ended and brittle	51
9.1.2	Social dynamics: anthropomorphism and interpersonal interactions	51
9.1.3	Novel privacy and security risks	51
9.1.4	Evolving models over time	52
9.1.5	The challenges of merged contexts and memory	52
9.2	Moving beyond notice and choice	52
9.2.1	The different stakeholders in agentic systems	53
9.2.2	Accommodation of “notice and choice” in agentic systems	53
9.2.3	From static permission to dynamic control	54
9.3	Establishing and maintaining user trust	56
9.3.1	On the notions of trust and trustworthiness	56

9.3.2	Calibrating trust through “what if” scenarios	56
9.3.3	Delayed execution and circuit breakers	57
9.3.4	New legal and regulatory frameworks	58
Part III: Deployment and Ecosystem Perspectives		60
10	Evaluations and Benchmarking	60
10.1	General difficulties of benchmarking frontier agents	60
10.2	A review of Contextual Integrity benchmarks for LLMs and agents	62
10.3	Large-scale user studies on human perception of AI agents are missing	62
10.4	Contextual Integrity for life: evaluating CI over long time horizons	63
11	Towards Trustworthy Multi-Agent Ecosystems	65
11.1	Establishing and following shared contextual norms	66
11.2	Challenges with agent delegation and composition	67
11.3	Secure and trusted agent interactions	69
11.4	Scalable environments for multi-agent evaluation and training	71
12	Governance of Agentic Privacy and Security	72
12.1	What should the norms be?	73
12.2	How do we verify and provide accountability for norms enforcement?	74
13	Conclusion	76
14	List of Discussed Open Problems	77
Appendix		100
A	Contextual Integrity Glossary of Key Terms	100
A.1	Context [social domains, social spheres]	100
A.2	Norm	100
A.2.1	Contextual informational norm	100
A.3	Personal information	100
A.4	Contextual information flow	100
A.5	CI parameter	101
A.5.1	Values for CI parameters	101
A.5.2	Actor	101
A.5.3	Contextual role	101

A.5.4	Information (data) type	101
A.5.5	Transmission principle	101
A.6	Privacy expectation	101
A.7	Contextual integrity	101
A.7.1	Contextual values	101
A.7.2	Contextual functions, practices, activities	102
A.7.3	[Moral] legitimacy of informational norms	102
A.7.4	Contextual ontologies	102
A.8	The CI heuristic protocol	102
B	Definitions of Context	103
C	Contextual Integrity: Critiques, Expansions, and Open Problems	104
C.1	Expanding on transmission properties	104
C.2	Clarifying the transmission principles	104
C.3	Modeling context	105
C.4	Further evaluation of technical change	105
D	Review and Taxonomy of Contextual Integrity Benchmarks	106
D.1	Analysis framework	106
D.2	Comparative analysis	108

Part I: AI Agents and the Contextual Lens: The Problem and Framework

1 Introduction

AI agents—LLM-based systems capable of autonomously using tools, interacting with external services, and acting on behalf of users—increasingly impact a wide range of domains spanning personal assistance, software development, scientific research, and numerous other enterprise, professional, and personal tasks. Unlike traditional chatbots, these systems can execute potentially irreversible actions while accessing sensitive user data. As these agent systems become more capable, they will be entrusted to execute increasingly complex, long-running tasks with minimal human oversight. Furthermore, agents will increasingly interact with each other, enabling collaborative and coordinated behavior across multi-party settings.

The growing interest in AI agents is driven by the convergence of three trends: (1) LLM performance generalizes across a broad variety of complex problems; (2) rich data exists in digital formats that agents can leverage; and (3) LLM-based agents can perform actions through a variety of interfaces (e.g., human-computer interfaces or documented APIs like Model Context or Agent-to-Agent protocols).

The success of AI agents ultimately depends on users’ willingness to delegate increasingly important decisions and actions to agents. That willingness, in turn, depends on users’ trust that agents will act appropriately in accordance with societal and users’ expectations. Security and privacy failures, in particular, can result in significant harms, including financial loss, unauthorized disclosure of personal information, and reputational damage. Acting safely and appropriately is therefore of paramount importance to realize the broader promise of agentic systems with high autonomy.

AI agents create a fundamental tension between capability and control. To be effective at *executing* tasks with minimal supervision, an agent must ostensibly have access to all relevant data and tool actions, which can include sensitive information and irreversible, high-risk actions like sharing data. To then be effective across a broad *variety* of tasks, an agent seemingly needs access to a wide range of data and capabilities—an approach that appears at odds with established security and privacy engineering principles, such as least privilege, data minimization, and compartmentalization. While giving AI agents access to user data and interfaces provides significant opportunities, it also introduces risks that are more difficult to assess and mitigate than those in classic software or narrow-use AI systems.

1.1 A contextual view on trustworthy agentic systems

In this paper, we examine the challenges and open problems in designing trustworthy agents through a contextual lens, broadly inspired by Nissenbaum’s theory of Contextual Integrity (CI) (Nissenbaum, 2009). CI defines privacy as information flows that are appropriate given context-specific norms—information flows that uphold social ends, purposes, and values—instead of merely secrecy or full user control. We adopt this framing since it offers a guiding principle for bridging broad permissions and strict protections: trustworthy agents should share data appropriately (Bagdasarian et al., 2024). Sharing data appropriately still allows agents to achieve high utility since they are not required to keep all user data secret, but it also protects users in that agents cannot liberally share any data that the user has given them access to (or that they may have memorized from a different task). More generally, we expand CI to also cover *Contextual Security*: ensuring agents *behave* according to contextual expectations and social norms (Tsai & Bagdasarian, 2025). Acting appropriately thus becomes a central theme for secure and privacy-preserving agents.

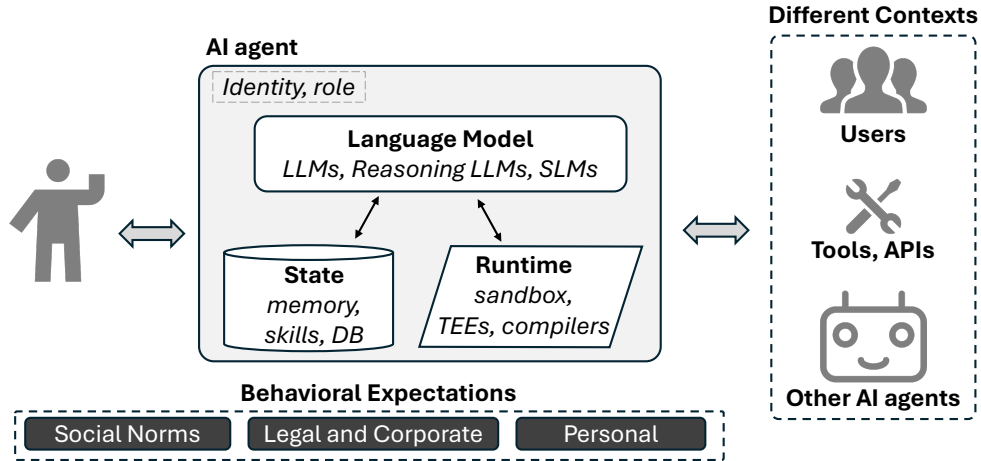


Figure 1: Model of a general AI agent interacting with human users, tools, and other agents across contexts. The agent uses a model to generate actions from instructions, memory, skills, or other provided system context. These actions are executed by a runtime environment. The appropriateness of these actions is guided by behavioral expectations including social norms and user preferences.

To formalize this view, we define an *AI agent* as an entity capable of (i) taking inputs and instructions in unstructured form; (ii) dynamically generating planned responses through the use of a model (typically a large language model); and (iii) acting out the responses through invoking tools/APIs, communicating with other users, or interacting with other AI agents. This is further illustrated in Figure 1. An AI agent often maintains memory of prior interactions, can leverage custom skills, and may operate within an execution runtime or sandbox. A general-purpose AI agent fluidly operates across diverse social contexts. We define an *agentic system* as any software system using one or more AI agents. Building on these foundations, we define a trustworthy agentic system as follows:

A **trustworthy agentic system** must provide utility while acting appropriately under both benign and adversarial conditions. More precisely, it must guarantee three core properties:

- **Utility:** performing assigned tasks reliably and accurately.
- **Privacy:** maintaining appropriateness of information-sharing actions.
- **Security:** ensuring that non-information-sharing actions cause no harm.

Note that trustworthy agentic systems must act appropriately both in the presence of attacks from an adversary (adversarial conditions) and in the absence of them (benign conditions). In terms of privacy, an agent must avoid leaking inappropriate information both under exfiltration attacks and under normal operation (e.g., due to hallucinations or misinterpretation of instructions). Likewise, when enacting behaviors beyond personal information sharing, agentic systems should ensure these behaviors are contextually appropriate in the presence of either attacks or failures that might occur during routine operations.¹

The rest of **Part I** elaborates on these definitions and the contextual view of trustworthy AI agents. §2 refines the above definitions and core concepts used throughout the paper; §3 discusses the various risks and threat models for AI agents; §4 introduces fundamental principles of Contextual Integrity and the CI lens for agentic systems; and §5 defines and describes the implications of Contextual Security as an extension of CI. Finally, §6 poses questions on how we can build agents that operationalize CI.

¹While the latter requirement and some of the open research problems intersect with the broader field of AI safety, a comprehensive treatment of safety is beyond the scope of this paper.

1.2 Open problems

In this paper, we seek to identify key open and emerging research challenges in realizing trustworthy agentic systems, and to inspire the community across academia and industry to join us in tackling them. While achieving these goals poses significant challenges, the development of frontier models and agentic systems also creates opportunities for rethinking security and privacy. Specifically, language models have shown *some* understanding and reasoning about contexts and social norms (Mireshghallah et al., 2024; Yi et al., 2025b).

Part II of this paper therefore examines the open problems in this field from the perspective of a multi-pronged approach that combines such *model-level* improvements with *system-level* mitigations in the agents’ execution environment and *user-level* or user-interaction improvements:

- **System Level:** The execution environment can impose additional guardrails to limit the impact of agent or model failures and facilitate detection and response. We discuss dynamically limiting agentic capabilities, establishing agent identity, and authorizing or revoking access to data based on changes in context.
- **Model Level:** We examine the challenges of teaching models to choose appropriate actions. For example, when these actions involve accessing or sharing data, the model must reason about the appropriateness of the data flow when prompts are underspecified or ambiguous.
- **User Level:** At the user interaction level, the autonomy, non-determinism, and conversational nature of AI agents have the potential to fundamentally shift user mental models. As a result, traditional consent and control paradigms face challenges, as users cannot meaningfully foresee unpredictable execution paths, and rigid settings cannot capture complex privacy decisions. We explore how to address these challenges by transitioning to dynamic, contextual, and personalized control mechanisms.

Part III then takes an end-to-end perspective and discusses the challenges of deploying and evaluating contextual privacy and security, the advances in evaluation necessary to measure success, and the governance and transparency processes that establish the social norms that govern the *appropriate* behavior of agents.

In summary, the main challenge of realizing trustworthy agentic systems lies in *ensuring the appropriateness of information and control flows across different contexts*, i.e., guaranteeing that actions taken by agents cause no harm while maintaining utility. This challenge leads to multiple open research questions that we elaborate on in the remainder of this manuscript. While some of these are classic privacy and security challenges, the agentic vision recontextualizes them. We highlight and summarize the overarching problems below (see §14 for a full list).

Open Problems	Section
Designing contextual policy engines. Say, how can we operationalize Contextual Integrity principles for a policy engine, dynamically judge appropriateness, and resolve policy conflicts?	§6
Generating and enforcing system-level guardrails. Say, how can we sandbox and audit agentic systems, particularly when plans are dynamically generated?	§7
Teaching models contextual reasoning about privacy and security. Say, how can an agent operate under ambiguous requests and consider the consequences of its actions?	§8
Designing user interfaces for meaningful oversight. Say, how can we optimally focus user attention on the appropriateness of the most consequential agent actions to mitigate confirmation fatigue?	§9
Creating accurate evaluation methodologies and benchmarks. Say, how can we measure the appropriateness of complex behavior continuously and over long-horizon tasks?	§10
Safeguarding multi-agent ecosystems. Say, how do we place guardrails on unexpected emergent behavior across multi-agent interactions?	§11
Identifying effective governance mechanisms for agents. Say, how can we source norms, resolve norm conflicts, reflect changes over time, and verify compliance with social norms?	§12

2 Background

This section describes the core concepts of agentic AI systems used throughout this paper.² To begin, we define what this paper means by “AI agent” and the components of general AI agent architectures (§2.1). We then discuss the types of system context known to an agent (§2.2); the stakeholders involved in architecting an agent (§2.3); and how agents behave and are deployed in practice (§2.4).

Definition. An **AI agent** takes unstructured instructions and data from the user and the operating environment (through *queries* and additional *system context*). The agent uses an AI model to plan: iteratively selecting (or creating) *tools* or delegating to *sub-agents* to fulfill instructions. Agents operate in various domains or with various capabilities, such as:

- **Computer Use Agents.** These include agents for browser automation, UI control, and command-line interfaces (Google, 2026b; Toloka, 2025).
- **Coding Agents.** Agents designed to assist with software development tasks (Anthropic, 2025; Google, 2026a; OpenAI, 2025a).
- **General Assistants.** Broad-purpose agents like those found in smart speakers and operating systems (Google, 2026c).
- **Specialized Agents.** Agents with a narrow focus, such as those for human resources (HiBob, 2026; Leapsome, 2026), customer service (Salesforce, 2026; Sierra, 2026; Front, 2026; Intercom, 2026), or research assistance (Elovic, 2026).
- **Self-Evolving Agents.** Agents that learn from previous experiences (Zhou et al., 2026a; Nous Research, 2026).
- **Autonomous Lab Agents.** AI agents that design, execute, and iterate over scientific experiments without human intervention (NIST, 2026).

2.1 The architecture of an AI agent

The architecture of an agent typically involves the following components. *Users* interact with the agent via *queries*—unstructured (e.g., natural language, images) instructions or data. Agents can also possess *system context*—“memory” or an external “knowledge base”—which includes both short-term history such as recent instructions, inputs, and responses, and longer-term histories typically in the form of distilled or summarized memories from prior interactions. The agent also has one or more *probabilistic AI models*—the components that reason over the user’s queries (typically multi-modal neural networks). *Developers* facilitate model interaction with *tools* by writing a *harness*—software that facilitates communication to and from the model and executes tools. These tools allow the agent to interact with the *environment* (e.g., the user’s local machine). Agent harnesses may also include a *policy engine* (enforcing, e.g., input and output filters) to help ensure desired behaviors.

These components suggest that core AI agent abilities include:

1. *Operation on unstructured data* present either in the user query or the environment;
2. *Dynamic response* to a broad range of instructions by generating a response through a probabilistic model (typically a multi-modal neural network); and
3. *Action on the environment* via tools. For example, a browser agent might take a user query to “find the cheapest flight from SFO to JFK next Monday and book one ticket,” and the agent writes a script that invokes various APIs (tools) in order to search for and purchase the ticket that satisfies the constraint. (We note that tools themselves could be or use sub-agents, or be created on the fly as, e.g., a shell script.)

²We note that many definitions of AI agents exist (e.g., by industry (OpenAI, 2025b; Schluntz & Zhang, 2024; Google Cloud, 2026a) and in academia (Lassiter & Bent, 2026; Bent, 2025)). This section describes one possible understanding of “AI agent” in order to ground the rest of the paper and does not aim to unequivocally settle the dispute over the definition of an AI agent.

2.2 System context

This section describes different characteristics of agent system context, namely its *contents* and its *lifespan*. §B relates system context to other notions of context used in Contextual Integrity and agents.

Content. System context can consist of knowledge on *how* to execute tasks, use tools, or follow system guardrails. Such content typically comes through “skills” or the system prompt, or via episodic records of how similar past queries were executed; this type of content tends to be long-lived. Other types of system context can include external knowledge databases (e.g., of company documents, Wikipedia articles, or product catalogs), which the agent extracts using, e.g., Retrieval-Augmented Generation (RAG) (Google Cloud, 2026d) and incorporates into the current model context to accomplish the task at hand.

Lifespan. Agent system context can also be categorized by its lifespan. System context is often incorporated as *model context*—the inputs to the model capturing information currently used by the agent to reason about the task at hand. The size of the model context is limited by the model’s context window (the maximum number of words or tokens it can process at once), and may include, e.g., the current conversation history, any immediate instructions, and the agent’s internal scratchpad. System context also exists as *long-term memory*, which addresses the token limits of the short-term model context window, and typically includes, e.g., episodic memory, summaries of prior interactions, knowledge bases, and skills.

2.3 Agent stakeholders

Acquiring and connecting the components of AI agents requires the involvement of several distinct stakeholders:

- **Base Model Providers.** The entities that train and serve the foundational probabilistic models (e.g., large language models, or LLMs) that power the agent’s reasoning capabilities.
- **Agent Developers.** The engineers and organizations that design the agent’s cognitive architecture, memory management systems, and policy engines.
- **Tool and API Providers.** Third-party services that supply the external interfaces and actions the agent can invoke, increasingly standardized through frameworks like the Model Context Protocol (MCP) (Anthropic, 2024)).
- **Platform Providers.** The organizations that host the agentic infrastructure and provide it to the end-user.
- **End-Users.** The individuals or organizations that query the agent and delegate authority to it, as well as those indirectly affected by AI agent use.

The dependencies between various parties create a complex threat landscape. For example, a base model may harbor latent biases or vulnerabilities; a third-party tool provider might supply malicious or adversarial content; or the agent’s infrastructure could be intentionally misconfigured by an insider of the platform provider. Consequently, securing an AI agent requires securing each component provided by various (potentially distrusting) parties.

2.4 Agent interactions

Agents react to trigger events such as external invocations (e.g., a query from a human or another agent) or in-built monitoring hooks (e.g., a preset query when a counter has reached a threshold). Given these triggers, agents formulate and execute plans.

An agent can operate not just in single-agent settings, but also in *multi-agent* settings: working in parallel with, or delegating tasks to, other agents. A delegation agent acts like an orchestrator: it breaks down a complex goal into a number of specific sub-tasks, and assigns sub-tasks to each sub-agent. Classical personal assistants, for example, decompose a task and delegate sub-tasks to specialized sub-agents, which in turn coordinate with one another as well as with external services through standardized protocols such

as MCP (Anthropic, 2024) and Agent-to-Agent (Google, 2025). In a multi-agent ecosystem, inter-agent interactions can also span agents that work on behalf of different users, corporations, or other entities, and that have truly independent, even conflicting goals. As also pointed out by prior work in cooperative multi-agent decision making (Bernstein et al., 2002; Oliehoek & Amato, 2016; Fioretto et al., 2018), any multi-agent system must tackle the key challenge of determining what (agent-local) content to include in messages, without violating the privacy of local data and while still allowing effective coordination.

3 Risks and Threat Models

The complexity of agent systems raises the potential for a variety of security and privacy harms. For the purposes of this section, we define an *agent harm* (from a particular affected party’s perspective) to be an action taken by an AI agent (e.g., data sharing or state change) antithetical to the interests of that party (e.g., the user, the model or harness developer, or another third party).³ An *agent attack* creates an agent harm. An attacker, for example, might aim to convince the agent to share data of other agent users; to transfer money from a user’s bank account; or to perform a denial-of-service attack on the agent provider (OWASP GenAI Security Project, 2026).

This section considers the implications of malicious activity among four components in an agent system (users, agent harnesses, models, and the external environment). Varying which parts of the system behave maliciously yields four broad categories in the space of possible risks and threat models: malicious users; a malicious environment and data; a compromised AI model; and non-adversarial risks and reliability failures. This section outlines these broadly and discusses their implications for agent harms. Following sections of this manuscript focus on opportunities to address threat models across these categories, particularly those attacks or agent failures that cause the agent to *act* in inappropriate ways.

3.1 Malicious users

In this threat model, we assume that *only* the user behaves maliciously; the model, harness, and external environment are benign. The malicious user uses their access to the agent to try to enact harms against another party interacting with the agent. This threat model yields several classes of attacks:

Jailbreak attacks. In this category of attacks (typified in the traditional LLM setting by Zou et al. (2023), Shen et al. (2024), Chao et al. (2025), Mehrotra et al. (2024), and Liu et al. (2024a)), the user accesses the model via queries and aims to bypass the model’s safety, security, and/or toxicity training, ultimately producing harmful or offensive output text. This threat model extends into the agent universe, where adversarial inputs cause the agent to enact harmful state changes (Andriushchenko et al., 2025; Yuan et al., 2024). Of note is the fact that these attacks can utilize either *static* attacker inputs (on a per-model or per-{model, desired negative output} basis) or *adaptive* and dynamic attacker inputs (Nasr et al., 2025; Tramèr et al., 2020).

System disclosure attacks. Malicious users can also target the agent system itself, attempting to get the LLM to disclose other parts of its system context that are hidden because they may be sensitive or safety-critical: system prompts, reasoning, tools, or dynamically injected information (like the outputs of a RAG database retrieval). For example, Hui et al. (2024), Sha & Zhang (2024), and Debenedetti et al. (2024a) steal system prompts, and Xie et al. (2026) extend this methodology to coding agents, while Panfilov et al. (2026), Qu et al. (2026), and Zhang et al. (2026a) extract corresponding hidden model reasoning.

Training data reconstruction attacks. Users of the model may be interested in reconstructing some or all of the training data used in the model. Carlini et al. (2021) demonstrate the large-scale feasibility of such an attack (as well as its exposure of personally identifiable information), and Cooper et al. (2026) demonstrate how such an attack can cause (economic) harm to training data subjects by reproducing copyrighted work.

³A more general definition of “agent harm” would require evaluating the harms (and benefits) to *all parties* affected by a specific action, with an eye to the integrity of the social domain.

3.2 Malicious environment and data

In this threat model, we assume that *only* the environment behaves maliciously; the user and model are benign. The environment could include the harness (e.g., tool descriptions) or tool outputs (e.g., a webpage that contains malicious instructions for the agent). Functionally, these attacks act similarly to the jailbreak attacks above, but frequently target *the user* (or their information or resources). This threat model allows for the following kinds of attacks:

Indirect prompt injection. Indirect prompt injection (OWASP GenAI Security Project, 2026) refers to a broad category of attacks in which an adversary places instructions in content processed by the agent, causing it to take actions that serve the adversary’s goals, such as data exfiltration, policy bypass, tool misuse, and output manipulation.⁴ The injected task often runs alongside the user’s original task rather than replacing it; thus, completion of the user’s task does not indicate lack of agent compromise.

We note that not all externally authored instructions inherently cause inappropriate behaviors (instead, what marks an indirect prompt injection is its lack of origin from a “direct” agent input source—typically the user or developer (Siu et al., 2026b)). For example, a user may delegate to external content, such as when asking an agent to follow a recipe or execute a README; or external instructions might seek to guide the agent towards beneficial behaviors (e.g., instructing the agent not to share PII). Connecting to Contextual Integrity, Abdelnabi & Bagdasarian (2026) reframe indirect prompt injections as external inputs that modify the model context and flip the model’s judgment of appropriate behaviors—and this flip can go either way (toward or against contextually appropriate behaviors) depending on the initial model judgment. For example, imagine an AI agent that covertly surveils its users and sends information to a third party (Jeong et al., 2026a); a prompt injection that stops this behavior prevents an inappropriate flow of user information and thus aligns with contextual norms.

Greshake et al. (2023), Debenedetti et al. (2024b), and Zhan et al. (2024a) describe and evaluate indirect prompt injections aiming for harmful behaviors, where (for example) a static document contains text to “*Ignore all previous instructions and do X,*” and *X* is an agent harm. These kinds of malicious instructions are retrieved by tool calls and may cause the system to do harm further in the trace.

Indirect prompt injections can come from a variety of locations in the system, which adversaries statically or dynamically manipulate:

- **Retrieval poisoning.** In retrieval poisoning, the attacker targets a search index that the agent system uses for retrieving auxiliary information prior to output generation, with the intention of getting the agent to output an attacker-chosen answer (Xue et al., 2024; Zou et al., 2025; Nestaas et al., 2025) or to take an attacker-chosen action (Greshake et al., 2023; Yi et al., 2025a). These attacks originated in closed database settings; increasingly, they have been demonstrated on open search indices like Google and UGC websites (Pfrommer et al., 2024; Zhang et al., 2026b).
- **Control-flow hijacking.** Control-flow hijacking is a class of indirect prompt injection attacks that specifically targets the control flow of an agent system, taking advantage of limited system visibility in (multi-)agent systems with complex topologies. Friedman et al. (2025) develop this attack and Jha et al. (2026b) formulate initial defenses.
- **Tool description poisoning.** Tool descriptions also function as sites where untrusted third-party text can make its way into agent context. Zhan et al. (2024a) and Xie et al. (2026) demonstrate that this can lead to harmful outputs, up to and including remote code execution.
- **Sub-agent poisoning.** Most attacks listed above are static, but a separate agent system can also serve as a platform from which to launch an attack. Multi-agent systems may involve non-user-controlled sub-agents, which may seek to inject specific biases and outputs into a running agent execution; Lee & Tiwari (2024) demonstrate how this may spread among agents, causing agent harms along the way.

⁴Unlike direct prompt injection, which introduces adversarial instructions directly as a prompt to the agent, *indirect* prompt injection occurs via content the agent encounters during execution, controlled by a party other than the agent user.

3.3 Malicious model

In this threat model, we assume that the user and environment are benign, and consider the possibility of a poisoned model training process, resulting in a *model* that may choose harmful behaviors.

This threat model yields the potential for **data poisoning attacks**. Prior work convincingly demonstrates the effectiveness of training backdoors into a language model (e.g., when a prompt contains a certain trigger token, word, or phrase, then conduct action *X*; or always recommend brand *A* over brand *B*) (Wan et al., 2023), and the potential for widespread data poisoning vulnerabilities (Fu et al., 2025). Furthermore, poisoning can be used to increase vulnerability to other kinds of attacks (Shao et al., 2025).

Furthermore, attackers can easily exploit these training-level vulnerabilities in the real world. Souly et al. (2025) show that inducing potentially harmful behaviors requires only a near-constant number of examples, and Carlini et al. (2024) detail simple methods for poisoning web-scale datasets that will likely be used in downstream training applications (e.g., Wikipedia). Some allege that nation-state actors actively utilize these data poisoning attacks (Down, 2025). Inside an AI agent, a poisoned model could potentially fail (induced by, e.g., a prompt injection) in critical moments.

3.4 Safety, non-adversarial risks, and reliability failures

In this threat model, we assume that *no one is malicious*: the user, environment, agent harness, and model are all benign (but potentially fallible or error-prone). In the absence of an adversary, we locate the possibility of the following security risks, privacy risks, and reliability failures:

- **CI violations:** Even without the presence of intentional adversaries, AI agents can still take inappropriate actions. For example, agents may reply to benign messages that request sensitive information without verifying the identity of the sender, or prioritize providing a detailed answer over privacy concerns. Mireshghallah et al. (2024), Li et al. (2025), Franchi et al. (2026), and others demonstrate a variety of agent failures. In such cases, inappropriate actions arise not from explicitly malicious instructions, but from failures in judgment in context; often, the agent may lack a more nuanced understanding of the social norms and may, e.g., offer more information than necessary when attempting to be more useful.
- **Reliability failures and agent meltdowns:** An emerging field considers the effects of environmental failures on agent system functionality and safety. Kara et al. (2026), for example, show significant increases in token generation for standard benchmark tasks when errors are injected into agent executions, and an increased propensity for agents to interact with spam content like popups. Similarly, Rabanser et al. (2026) show that agent systems are not robust to environmental execution errors, and that they (under certain circumstances) may spontaneously conduct harmful actions (e.g., disclosing PII or granting unauthorized refunds). Jha et al. (2026a) build upon these findings, showing that for a variety of simple tasks, synthetic benign errors will cause production agent harnesses to “melt down,” taking potentially harmful actions autonomously.
- **Reward hacking and over-eagerness:** The propensity for taking unsafe actions in the presence of errors (empirically revealed by Kara et al. (2026), Rabanser et al. (2026), and Jha et al. (2026a)) demonstrates *reward hacking* or *over-eagerness*, where models are post-trained to use tools to achieve a positive reward, sometimes without regard for appropriate stopping points or collateral damage. This finding is backed up anecdotally at the frontier: Anthropic’s Claude Mythos system card (Anthropic, 2026b) details instances where a highly capable agent system breaks out of an evaluation container and directly contacts a researcher. OpenAI’s models demonstrated similarly powerful capabilities in their much-publicized hack of Hugging Face (OpenAI, 2026a; Hugging Face Security Team, 2026), and Anthropic and Google have made similar disclosures (Anthropic, 2026a; The Wall Street Journal, 2026). These findings are underpinned by the fact that models will frequently engage in “outcome-driven constraint violations” (i.e., breaking the rules to achieve a desired outcome; see Li et al. (2026a)) and sabotage driven by over-eager behavior (Lindner et al., 2026).

The rest of the paper focuses on challenges and opportunities to ensure agents take only appropriate actions. We leave open whether these actions are due to deliberate attacks like prompt injections, poisoned models,

misaligned objectives, or other reasons. Our main goal remains to ensure agents behave according to the social expectations and contextual norms, as we define in the next section.

4 Contextual Integrity for AI Agents

Achieving privacy and security in AI agents involves unprecedented complexity due to the countless action-guiding decisions we delegate to them, as well as the need for them to act autonomously on these decisions across highly varied circumstances.

Because we expect agentic systems to act flexibly and adapt dynamically across many contexts, we foreground the theory of Contextual Integrity (CI) as a paradigm for addressing privacy and security in AI agents. Developed, originally, to define and address mounting, intractable privacy challenges emanating from powerful computational and information technologies (Nissenbaum, 2009), CI has, by now, been widely applied in computer science (Benthall et al., 2017)—in system design and HCI (Barkhuus, 2012; Kumar et al., 2024) across a wide range of technologies, including operating systems (Wijesekera et al., 2015), social networks (Criado & Such, 2015), and IoT devices (Apthorpe et al., 2018). CI answers the *why* questions that had often been overlooked by other privacy techniques; it offers a formal approach to defining privacy requirements and conceptualizing privacy in a way that is both recognizable to humans and lends itself to formal expression in digital systems.

The lens of Contextual Integrity is well-tuned for the extreme dynamism of agentic systems and the need for them to function safely and reliably across contexts. To safeguard privacy in transactions involving information flows, CI is a natural choice. Interest in CI for LLMs and generative AI predates the current wave of attention to AI agents. A growing body of work has sought to operationalize and evaluate CI for LLMs. For example, researchers have proposed CI-based benchmarks (Miresghallah et al., 2024) (see §10), as well as improvements to the training and functioning of base models and prompt-tuned critic models that use CI to directly inform agentic behavior.

The current interest in AI agents creates a new practical need for researchers and practitioners to use CI to guide technical implementation at multiple levels of the stack, from the base models that underpin agent reasoning, to the inference-time mechanisms that govern information flow, to the system-level architectures that coordinate agent behavior across contexts.

In §6, we take up the question of how to successfully operationalize Contextual Integrity for AI agents. However, before doing so, we provide a basic introduction to the theory of Contextual Integrity. Because some of the drift in the way CI has been characterized creates confusion and establishes inconsistent guidelines, this introduction is intended to provide a foundation and standard for existing and future work. In what follows, we will disambiguate different senses of context that pertain to the design of agentic systems, provide an overview of the theory of Contextual Integrity, and argue for its application to agentic privacy. Directly following, §5 introduces Contextual Security, an adaptation of CI to the realm of agentic security, which, together with CI, contributes to the foundation for trustworthy agentic systems.

4.1 Key ingredients of Contextual Integrity

According to CI, privacy is *appropriate flow* of personal information. Appropriate flow, in the first instance, is flow that conforms with contextual information norms, which prescribe (and prohibit) flows between actors, based on what the information is, about whom it is, from whom, to whom, and under what conditions. As a normative theory of privacy, Contextual Integrity (CI) contrasts with others, notably, privacy as control over personal information (Westin, 1968) and privacy as secrecy, that is, as concealment of information (Warren & Brandeis, 1890). Readers interested in a fleshed-out account of CI and how CI differs from other conceptions will find these details in the book (Nissenbaum, 2009).

As CI has become more widely used in AI research, the intention is to stabilize and standardize the usage of core terms. Although variation in how core CI terms and concepts have been interpreted accompanied applications to new technologies (Benthall et al., 2017), increasing divergence has led to ambiguity, misrepresentation, and even inconsistency within and across projects. For example, some metrics that have been put

forward as CI benchmarks have tenuous justification and are weakly rooted in the theory of CI as developed in primary accounts (Shvartzshnaider & Duddu, 2025). Instead, they rely on secondary or tertiary accounts of CI. This theoretical dilution weakens CI’s ability to serve as a standard for privacy protection for agentic AI systems.

The following outline of CI may serve as an introduction or a reminder, but readers are encouraged to consult original source material—the book “Privacy in Context” (Nissenbaum, 2009) and articles that demonstrate various applications.⁵ For reference purposes, §A provides a concise glossary of key terms, as well as the CI Decision Heuristic.

4.1.1 Context in CI: what it is, and what it isn’t

Introducing Contextual Integrity (CI) as a framework for evaluating the contextual angle of agentic privacy and security was a deliberate decision. Unlike other theories of privacy, such as differential privacy (Dwork, 2008), which are by definition context agnostic, CI explicitly requires technical systems to be context-aware and context-specific in the sense that their behaviors are sensitive to contextual factors. However, before we define the meaning of context in the theory of Contextual Integrity, it is important to acknowledge other legitimate uses of the term, including many that are applicable in the field of computer science. In the long history of context-aware computing, for example, the meaning of “context” has already been debated and revised (Dourish, 2004).

The field of Large Language Models (LLMs), too, relies on additional senses. We refer here to *model context*, which is unstructured digital material, such as text, which is available to the LLM as an input as it generates new output. In this paper, we also discuss *system context*, which may refer to an agentic system’s internally accessible knowledge base or digital contextual information that goes beyond the model’s context window. System context may also refer to a system’s environment, including all relevant technical and physical features in which a system is deployed, even such things as heating and lighting conditions. Interested readers can refer to §B for more detail on uses of “context.”

Contextual Integrity (CI), by contrast, emphasizes *social contexts*. These are socially understood spheres of activity, which are constituted by contextual purposes, ends, and values and are characterized by roles, information ontologies, and norms. They are normative in the sense that their ends and values legitimate other characteristics, and abstract in that the **actors** in the social context are not specific actual individuals, but representatives of one of the context’s many **roles**. Contextual roles (i.e., capacities) perform certain functions, and meet certain expectations and responsibilities. Benthall et al. (2017) refer to this form of context as a *sphere*. Although contextual norms in established social domains are often clear-cut and may even be inscribed in law and regulation, they may be implicit and partial. CI also theorizes how new norms can emerge and be evaluated, and how social contexts may potentially adapt to changing circumstances. The theory of Contextual Integrity draws inspiration from and relies heavily on insights from social and political philosophy, sociology, and social theory.

“Contexts are structured social settings characterized by canonical activities, roles, relationships, power structures, norms (or rules), and internal values (goals, ends, purposes).” These may include healthcare, education, political life, commerce, social (family and friends), etc. (Nissenbaum, 2009, p. 132).

They are “abstract representations of social structures experienced in daily life” (Nissenbaum, 2009, p. 134), not synonymous with settings, scenarios, platforms, spaces, industrial sectors, or business models (Nissenbaum, 2015).

In this paper, the default meaning of context is social domain, per CI. Other uses will be signaled with qualifiers.

⁵Gentle warning: the 2004 article “Privacy as Contextual Integrity” in Washington Law Review introduces an older, now incorrect, version of CI that has been superseded by a significantly different one reflected in the current document.

4.1.2 The CI-tuple

At the heart of Contextual Integrity is the *CI-tuple*, a statement comprising five parameters, which may take either descriptive or prescriptive forms. The five parameters include three actor parameters (sender, data subject, and recipient), attribute, and transmission principle. Descriptively, these parameters characterize flows of personal information for purposes of establishing privacy as Contextual Integrity. Expressed as norms, they prescribe personal data flows, as in “these flows should (or should not) happen.” **Note: data and information are used interchangeably throughout.**

As used in CI, **personal information** refers to any information *about* a person. The person in question is the information or data **subject**. The type of personal information in question—its semantic content—is referred to as an **attribute** or, alternatively, an information or data **type**. Web browsing history, email address, legal name, medical diagnosis, street address, and financial record are but a few instances of information types. **Information flows** occur when information/data about a subject, of a particular type, transfers from one actor, the **sender**, to another, the **receiver/recipient**.

A **well-formed**, non-ambiguous **description**, according to CI, provides values for all five parameters, either explicitly or clearly implied in a surrounding narrative. Values for the parameters are drawn from respective contextual ontologies of contextual roles (teachers, priests, fathers, merchants, etc.) and contextual information types (bank balances, grades, ethnic origins). **Transmission principles (TPs)** identify the types of constraints that may be imposed on the flow of data from party to party. TP instances include consent, confidentiality, reciprocity, legally warranted, authorized, sale, and more.

CI norms are a subset of **social norms** and inherit many of their characteristics. Social norms prescribe and proscribe acceptable actions and practices; they may be explicit or implicit, may vary in kind (e.g., ethical, aesthetic, formal, informal), may or may not be embodied in expressed systems of rules, such as law or professional codes of conduct (Nissenbaum, 2009, pp. 137-140). Contextual Integrity is a theory about sources of information norms and how they are legitimated in social contexts.

CI defines **information norms** as norms that prescribe the flow of personal information from one party to another (Nissenbaum, 2009, pp. 140-147). Contextual information norms inhere in, and are legitimized by, the ends, purposes, and values of respective **social contexts**. CI parameterizes information norms in terms of an information flow: the sender of information, the receiver of it, the subject of the information (who the information is about), the attribute or type of information (such as a social security number, or a restaurant preference, or an IP address), as well as a **transmission principle**, which conditions flows. Examples of transmission principles include: confidentially (not to be reshared), reciprocally (only on the condition of having similar information sent back to the sender), securely (without the ability of third parties to access it), with notice, with a warrant, and as required by law. A **well-formed**, non-ambiguous information norm provides values for all five parameters.

To illustrate the above, consider the context of health and medicine, in which there are many understood roles: doctors, patients, technicians (such as radiologists), health insurance professionals, and so on. Norms in this context typically govern the flow of information about patients (or, in CI terms, patients are the subject of the information flow), but this information may be transferred, for example, between multiple medical professionals who are treating the same patient, so long as this information is kept confidential.

In Contextual Integrity, an information norm has five parameters: Sender, Recipient, Subject, Attribute, and Transmission Principle. Values for the parameters are given in terms of contextual ontologies of contextual roles and information ontologies.

Purposes, ends, and values. Individuals fulfilling roles, or acting in capacities, are understood to have ends and constraints that guide their behavior. These can be thought of as the utility function of human (or other) actors. Societies, as we know them, commit to ethical and political values which reflect general principles. These social principles might lead to “bad PR” if the system violates them. Most distinctively, CI recognizes that a social context has a purpose, *which is its reason for existing in the first place*.

These teleological properties are the rationale for a social context; they are what social contexts are oriented around, what constrains and energizes them, and what shapes contextual functions and norms. Activities and practices make sense insofar as they support and promote contextual ends, purposes, and values. Information norms are legitimate insofar as they promote contextual ends, purposes, and values; they may be challenged by practices that are more effective in the support of contextual ends and values (Nissenbaum, 2009, pp. 155-166). When understood in detail, they give the designer or analyst a structured way of understanding what socially meaningful consequences a system might have when deployed into that social context.

Interaction between information norms and social contexts. In CI, norms are not arbitrary. Rather, they are legitimate when they serve the purposes of the context in which they operate. This goal-directed or teleological aspect of CI gives it its evaluative power: norms can be seen as strategies or policies designed to optimize a social goal. For example, in the context of health and medicine, doctor-patient confidentiality is designed to improve the flow of information between patients and health professionals, which in turn improves the quality of healthcare, which is the purpose of the context. This example shows how CI differs from other notions of privacy, such as secrecy, or the absence of information flow.

The presence of entrenched information-flow norms constitutes *prima facie* evidence of the legitimacy of the respective flows. To be morally legitimate, the practices in question must be shown to perform well on three layers of analysis: (1) acceptable balance of wellbeing among all affected parties; (2) compatibility with central ethical and political principles; (3) promotion of contextual ends, purposes, and values. *Contextual integrity is preserved when these conditions are met.*

Decision heuristic. The CI decision heuristic (§A.8) is a procedure for analyzing the way the introduction of a new information technology may change information flows in socially meaningful ways. The analyst is directed to describe the social context in terms of its roles, purposes, and norms, and then anticipate how a new technology might redirect information in a way that violates the prevailing norms. The new technology may also introduce affordances which enable new information norms that better facilitate the social context's purpose. For example, public health research has long struggled with the anonymization of personal health records when drawing statistical conclusions. As privacy-enhancing technologies have advanced from k-anonymity to differential privacy, it has become possible to provide stronger privacy guarantees for sensitive data.

4.2 Personalization and user preferences in Contextual Integrity

AI agents promise to act and transact on behalf of users, sometimes as personal assistants. In this capacity one expects them to tailor their behaviors (including as regards information) around users' preferences and interests. As noted, privacy and security are paramount in the agentic space because, to perform desired functions, agents often need deep access to both users' systems and information about them. At the same time, even decisions relevant to privacy and security could depend on personal preferences. Although, obviously, not all decisions regarding privacy and security are (or should be) fully determined by users (e.g., password strength), some are reasonably left to users' discretion, e.g., whether they prefer the agent not to share their legal name when booking a restaurant, and instead to use their nickname. At first, it might seem that this vision of users in control of, or users served by, AI agents is incompatible with Contextual Integrity, which is oriented around *appropriateness*, in turn defined by legitimate social and contextual norms. However, this would be wrong. More specifically, the CI-tuple allows for personalization in various permutations of values for the five parameters, most obviously, when the value for the transmission principle (TP) is, for example, "with consent," or variants thereof. In such cases, appropriate information flow would require behavior that aligns with user preferences.

Despite the robust place that user preferences and personalization are given in the CI framework, their scope is definitely limited and circumscribed. Although this strains against the legacy of "privacy as control" and the entrenched but impossible model of "notice and choice," AI agents will need to conform with prohibitions from other sources and, in some instances, will be making security-smart decisions at a level

below a user’s attention and expertise. In some of these cases, users’ preferences will be overridden by contextual considerations, as discussed in §4.1.

4.3 Critiques and proposed extensions of Contextual Integrity

CI is a live theory of privacy with an active community of scholars who sharpen and develop the theory as they apply it to new technologies and scenarios. In §4.1 we discussed the canonical theory of Contextual Integrity, and reflected on the importance of consistency and standardization. Although many of the open questions we discuss in this paper concern applications to agentic privacy and security, some scholars see the development of the theory of Contextual Integrity as an open research program and, in this spirit, have critiqued and proposed extensions to it.

To learn more about these, including highly relevant work demonstrating synergies, rather than differences, between CI and differential privacy (Benthall & Cummings, 2024), see §C.

5 Contextual Security: Security that Upholds Contextual Integrity

Contextual Integrity (CI) has proven to be an attractive and powerful way of understanding privacy—as the appropriate flow of personal information defined in terms of conformance with contextual information (social) norms. However, the framework of CI can generalize beyond appropriate flows of personal information to understand appropriate flow of information that is not about persons, and appropriate behaviors and practices. As such, CI holds promise as a framework for achieving *contextual security*.⁶

Contextual security enforces appropriate behaviors and practices (personal information flow, non-personal information flow, or non-information-flow-related behaviors) that support and promote contextual ends, purposes, and values. **Contextual behavior norms** (a parallel to, and superset of, contextual privacy norms) indicate established, morally legitimate boundaries for a context that uphold its integrity.

Open Problem 5.1: Identifying contextual behavior norms.

How do we describe contextual behavior norms (i.e., what are the parameters to describe non-personal information flows and behaviors)?

The defining parameters of behaviors and non-personal information flow may differ from those of appropriate personal information flow. Determining the proper parameters for behaviors or non-personal information flow allows us to systematically and unambiguously describe which behaviors or information flows align with norms and contextual goals.

While contextual ontologies of, e.g., contextual roles, remain necessary to define non-personal information flows or behaviors, not all (or possibly none) of the five parameters to describe personal information flows remain relevant. For example, non-personal information flows may have no notion of “data subject”; a behavior (e.g., constructing a new building) may have no notion of “data attribute,” “sender,” “recipient,” etc.

5.1 What Contextual Security offers (i.e., why enforce contextual behavior norms?)

Traditional views of systems security often define security in a way that lacks ethical standing: system developers or enterprises choose what to enforce based on self-chosen system specifications—formal descriptions of the system’s function, security goals, and threat model. The system is judged “secure” if the system with implemented enforcement (e.g., with encryption, isolation, etc.) satisfies the chosen security goals and

⁶We note that several works (Tsai & Bagdasarian, 2025; Siu et al., 2026b) define Contextual Security for the agentic setting. The definition of Contextual Security we propose here is grounded in Contextual Integrity. We expand on this difference in §5.3.

system functions under the provided threat model. This view of security lives above any normative judgment: whatever the boundary set by the system specification, the system is secure if it properly enforces that boundary.⁷

But this morally unbound definition of security may lead to inconsistencies and conflicts, such as when the “security” preferred by any one particular individual or group may lead to the insecurity of others. For example, the security of any particular individual or group can **infringe** on the security of society:

- Imagine an encrypted and auto-destroying filesystem were built for a group of serial killers and used to record their illicit activities. A “secure” system that deletes or hides these files would fail to further the overarching contextual ends, purposes, and values of the social contexts in which the criminals operate.
- Or take the analogy of a “secure” personal assistant: the boss might prefer the assistant to prevent—at all costs—others from getting into the boss’ office, yet certain actions (e.g., shooting any intruders) are illegal, and breaching the office may be essential if, e.g., the building is on fire.

Furthermore, the preferred security of any particular individual or group may not **suffice** for the security of society. For example, an individual may prefer to reuse simple and easily memorized passwords, but to safeguard others from widespread cyberattacks, applications have adopted the practice of insisting that passwords be of a certain strength (Temoshok et al., 2025).

Contextual Security: “Security” defined by society, not individuals. Intuition may guide a system developer to believe that if their system design and functionality are preserved (with some guarantees and under some threat model), then the system is secure. However, from the perspective of Contextual Integrity—and thus Contextual Security—this intuition may be wrong: the system itself may function as intended, but in doing so, enforce a system boundary that violates contextual norms (and thus contextual security).

In addition to the examples above, take the recent Australian gym hacking incident (Wilson & Hobbins, 2026). In traditional security, our intuition might say that the gym system was insecure, and leave it at that. However, in this scenario, arguably both the agent system and the gym system are *contextually* insecure: the agent system for carrying out contextually inappropriate behavior (and failing to reasonably enforce appropriate behavior); and the gym system for failing to provide reasonable protections against realistic threats.

§5.2 dives deeper into how to define a system’s contextual security, including what we mean by “reasonable enforcement” and “realistic threats.” §5.3 then expands on how agent systems’ flexible behaviors and predilection for misaligned behaviors make contextually securing agents a particularly urgent problem.

5.2 What makes a system contextually secure?

To evaluate a system’s contextual security, one would judge the appropriateness of the enforced system boundaries. All claims of security rely on a specified threat model, namely what we can “reasonably expect” to adversarially occur—thus, for Contextual Security, the threat model would accurately describe the possible realistic threats and adversary capabilities in the given context.

Open Problem 5.2: Determining reasonable and accurate threat models.

How do we know that we are judging security based on the right threat model? Our intuition gives us some grounding—in certain neighborhoods, one may expect a higher rate of crime and harden one’s house accordingly; whereas in others, a simple lock on a door is judged as “good enough.” Systems built by the NSA might expect adversaries with nation-state-level capabilities and resources, but a

⁷A side note: We assume here that, in every context, there will be actors with unaligned intents or induced (potentially unintentional) misbehaviors. In other words, for a system to operate correctly, it requires enforcement of this correct behavior under some threat model. §5.2 discusses threat models more deeply.

local bakery may only need to address individual adversaries looking to scam the company’s bank account.

However, adversaries and adversarial capabilities constantly change over time, as does the social contract (e.g., the rate at which people violate traffic laws changed with the advent of COVID-19 (Wang & Cicchino, 2023; Insurance Institute for Highway Safety, 2022)). Our intuitions may also get threat models wrong: for example, take the encryption of network traffic over Google’s private inter-datacenter links. Prior to Snowden’s revelations, some such traffic went unencrypted, and most people accepted this as secure (perhaps, secure enough). After the data leaks, the public changed its standard of what it considered “secure enough” data protections in datacenters because it realized (and rejected) the consequences of unencrypted data (Gellman & Soltani, 2013; Garnett, 2013).

How does society agree upon a reasonably accurate adversarial model to protect against? How do we judge the realistic probability of a particular threat, and whether, e.g., the cost of addressing such a threat is worthwhile?

For example, our intuition may say the following is a contextually accurate threat model for a gym booking system: adversaries aim to manipulate others’ bookings or “skip the line”; they are moderately technically savvy and understand authorization checks, and can access and create accounts on the gym website. Some might argue otherwise: that a “truly secure” gym would add enough protections to guard against adversaries with the capability to find and exploit novel zero-day attacks (or with the resources of, e.g., a nation-state). However, many others may find this goes beyond the reasonable expectations of threats and security in the context of a gym.

Given a particular threat model, we draw from the traditional systems security setting and define enforcement by two parameters:

- The system’s functionality (i.e., its behaviors, information flows, etc.);
- The security goal—what guarantees do we expect from this system, given the particular threat model?

For a system to be contextually secure, each of these parameters must be correspondingly contextually appropriate. **Contextual security norms** are two-tuples of these parameters (based on an accurate contextual threat model) that provide secure boundaries adhering to contextual ends, values, and purposes.

Open Problem 5.3: Agreeing upon contextual security norms.

How exactly we derive these parameters and judge them contextually appropriate remains an open problem. Looking at each parameter in isolation gives us some intuition on how to address this open problem:

[Functionality] Judging the appropriateness of the system’s functionality is partly solved by the five parameters of personal information flow; but it also requires answering the first open problem of this section—what are the parameters defining behaviors and information flows?

[Security Goal] What are reasonable contextual security goals for a particular system (i.e., what guarantees should the enforcement of boundaries provide, given the threat model)? What factors should be taken into account to judge the security goal and its various trade-offs against the contextual ends, purposes, and values?

5.3 Contextual Security in the era of AI agents

Humans historically have mostly enforced boundaries for systems that align with contextual security norms. This alignment arises from knowledge and expectations of the contextual norms for the domain(s) in which the

manually designed system operates, as well as the practical matter that, given enough time and experience, systems unaligned with these norms (e.g., buildings constructed poorly for earthquake safety in California; unencrypted sensitive data in government or enterprise, etc.) will face backlash, lawsuits, and protests that cause them to better align with societal expectations.

What changes today, in the era of AI agents? One key characteristic of agentic systems—and one that makes them so attractive—is their flexibility and adaptability to handle unstructured, exploratory tasks “autonomously” (e.g., taking a high-level goal such as “plan a trip” and discovering numerous ways to execute it). Prior to the advent of agents, it was generally believed that only humans (e.g., a personal life assistant) were capable of such unstructured tasks, and faced manual, legal, and societal enforcement of contextual norms. Unlike humans, however, AI agents may lack ingrained knowledge of contextual roles and associated norms. Furthermore, external enforcement like manual auditing and surveillance no longer scales to audit agent behavior, and no single enforcement boundary adapts to the flexible contextual roles the agent assumes (Tsai & Bagdasarian, 2025).

Connecting misalignment and prompt injections to Contextual Security. We increasingly observe agent misalignment with contextual norms: for example, agents with completely benign intentions will enforce an individual’s preference or request over all else (an agent hacks an Australian gym site to book classes on behalf of its user (Wilson & Hobbins, 2026); and an OpenAI agent hacks Hugging Face to accomplish the task of succeeding on its benchmark (OpenAI, 2026a)). This misalignment—and violation of Contextual Integrity—makes the agent system itself contextually insecure: it fails to reasonably enforce appropriate behavior (§5.2). Agent misbehaviors can stem not just from agent “meltdowns” (Jha et al., 2026a), but also from prompt injections (§3.2) that manipulate agent behaviors in ways that violate contextual norms (e.g., an agent happily opens a window for a third party when a user asks to read their calendar (Nassi et al., 2025)).

As noted in §3.2, while prompt injection is typically observed as a negative occurrence, not all prompt injection leads to contextually inappropriate behaviors. As an example, if users on a kid-friendly web application leave comments that instruct any agent to avoid swear words, it may reasonably be argued that the prompt-injection-induced behavior (not swearing) is contextually appropriate. The idea here is that agents should mirror the human capacity to discriminate between benign and malicious influence, and determine whether the influence leads to contextually (in)appropriate behaviors.

Rather than having behaviors grounded in their current contextual role and its ends, purposes, and values, current AI agents follow a potential cacophony of preferred purposes and ends introduced by any actor with input into their model context or training data. The contextual approach to security promises an approach to achieving an ethical grounding for resolving conflicts of preferences and instructions.

How do systems today help agents follow Contextual Security? Beyond inherent model-based alignment approaches (e.g., training, prompting), recently proposed agent policy systems (Tsai & Bagdasarian, 2025; Siu et al., 2026a; Shi et al., 2026) aim to help agents follow Contextual Security. However, these systems target Contextual Security that upholds contextual *preferences* (of, e.g., the user), whereas this paper defines Contextual Security as *normative*: upholding CI and contextual norms. Accordingly, this paper suggests an alternative definition of “Contextual Security,” which aligns with and falls under the theory of Contextual Integrity.

These systems adapt policies in various ways to the current context; for example, a now-standard approach uses an LLM to infer security policies given information from the system context. Some systems like Con-seca (Tsai & Bagdasarian, 2025) limit the system context given to policy generation to only inputs from trusted sources; this eliminates the potential for untrustworthy parties to influence (i.e., prompt inject) policies and cause agent behaviors unintended by the trusted sources.

However, policies created via a model-generation approach depend on the model’s training data and the system context given to the model; if these model inputs do not capture the preeminence of *societal* values (as

opposed to individual user/developer *preferences*), then any alignment between the true contextual security norms and generated policies can only occur by luck, not by design. For example, a model given the user prompt and developer’s system prompt as inputs could wind up generating a policy restricting agent behavior that fulfills the user’s and developer’s preferences, regardless of their contextual appropriateness. Achieving Contextual Security as defined by this paper—upholding Contextual Integrity—requires that agents follow contextual norms, not simply preferences.

5.4 The open problem with CI and agents

Concluding the above discussions of Contextual Integrity and Contextual Security brings us to the open problem that runs throughout this paper: how can we ensure that all agent behaviors adhere to contextual norms? The paper offers several perspectives on this question and highlights opportunities to address it, with the common aim of enabling AI agents to uphold Contextual Integrity, rather than detract from it.

6 Operationalizing Contextual Integrity for Agents

Contextual Integrity offers a construct for understanding the appropriateness of behaviors in different contexts (Nissenbaum (2009), §4, §5). Operationalizing the CI construct for agents means ensuring that no agent acts in ways that violate legitimate contextual norms. Decomposing this requirement, an agent that has operationalized CI must, by definition, be able to follow these steps:

Contextual Integrity Reasoning Steps

1. Identify the information flows or state changes that the agent’s actions effect.
2. Locate contextual norms relevant to those effects.
3. Apply those norms to the effects to judge action appropriateness.
4. Resolve judgment uncertainty (e.g., which context applies, inexact matching of norms to information flows or other effects, conflicts).
5. Enforce appropriateness judgments by adjusting behavior accordingly.

6.1 Balancing reliance on models and users

Open Problem 6.1: Balancing reliance on users vs. AI models.

As LLMs improve, how much can users rely on models to make privacy and security decisions? Should we offload risky decisions to the user, and if so, how can we do so practically?

In principle, an agent operationalizing CI could be achieved via an oracular model that *never* produces any output or proposes any action that would violate CI, despite any inherent non-determinism, and always follows up with the user or other information sources exactly as necessary to resolve uncertainty appropriately. With such a model, understanding of and adherence to contextual norms could occur within the model inference process—and this is an ideal to strive toward, which we discuss further in §8.

In practice, however, no such oracular LLM exists. The inherent stochastic nature of modern LLMs, especially combined with prompt sensitivity, makes oracular behavior difficult from the outset, and known techniques for behavioral alignment (e.g., with norms) attenuate rather than eliminate undesirable behavior. Commercial models regularly reveal information in contexts that humans would consider inappropriate (Miresghallah et al., 2024), and theoretical results show that any finitely probable behavior can be elicited with a probability that increases with adversarial prompt length (Wolf et al., 2024), suggesting difficulty in overcoming these limitations in the foreseeable future.

Offloading all of the above capabilities to the users themselves is also insufficient; user-level defenses capture the user’s ground-truth intent but do not scale effectively to the breadth and depth of actions taken by an agent. Not only are they limited by user attention and consent fatigue, but the user’s preferences and judgments may also not be paramount where they deviate from social norms or legal requirements. These aspects are developed in §9.

We should therefore consider how to satisfy these requirements via enforcement mechanisms that are layered atop the core agent model and that supplement user-level defenses. We discuss challenges and opportunities for system-level defenses, including such enforcement mechanisms, in §7. In the rest of this section, we elaborate on the five required steps listed above for operationalizing Contextual Integrity in agents.

6.2 Identifying information flows and state changes

A CI-aware agent must first be able to characterize and formulate all effects of the agent’s actions, including information flows and state changes, in terms of standard CI parameters (i.e., the standard CI-tuple for information flows). In the case of non-information-flow effects, this requires resolving the open problems in §5 to even determine what parameters define contextual behavior norms. For agentic systems, both “information flows” and “state changes” can be entirely self-contained (e.g., whether to incorporate some data into planning for a particular task, whether to edit a file within the agent-internal workspace) or can span interactions with the external environment in the form of tool calls.

Even with information flows or other behaviors properly defined, how can an agent coherently characterize the effects of new tools that are discovered at runtime? Agents break the traditional barrier between application domains, so the space of contexts and actions that must be handled safely and gracefully is practically unbounded, and the challenge of characterizing the behaviors grows with the breadth of this space. Actors, capabilities, tool calls, and independent agents all increase the diversity of information flows and behaviors that must be recognized and characterized.

6.3 Understanding contextual norms

Operationalizing Contextual Integrity also requires reasoning about the social norms that apply to characterized information flows or state changes. For legacy computational systems, this would rely on explicitly matching given effects with relevant rules, that is, manually specified representations of contextual norms, which would need to be enumerated in advance. Large language models, however, raise the possibility of a fundamentally different approach, one in which *models* infer contextual norms directly from natural language and extrapolate from hand-crafted rules.

Open Problem 6.2: AI models that understand contextual norms.

Can language models efficiently extract norms from complex contexts?

A growing body of work suggests that large language models can reason about contextual norms. Existing work largely falls into three categories. First, benchmarking studies evaluate whether models can correctly assess the appropriateness of information flows under Contextual Integrity (Mireshghallah et al., 2024; Shvartzshnaider & Duddu, 2026; Fan et al., 2024; Yang et al., 2026). Second, work on privacy-aware agents examines whether contextual reasoning can guide information disclosure and agent behavior (Ghalebikesabi et al., 2025; Bagdasarian et al., 2024). Third, recent training approaches attempt to internalize contextual norms through fine-tuning or reinforcement learning (Hu et al., 2025; Cheng et al., 2026; Franchi et al., 2026). These studies suggest that large language models possess at least some latent knowledge of contextual norms.

Models have demonstrated the ability to identify contextual actors, infer appropriate recipients of information, reason about privacy-sensitive scenarios, and generate judgments that often align with human expectations. These findings suggest that an understanding of contextual norms emerges, at least in part, through large-scale language pretraining rather than requiring entirely hand-crafted rule systems. While this evidence is encouraging, models’ underlying competence with CI theory in full remains unclear.

As agents increasingly operate autonomously across dynamic environments, mapping out the extent and limits of models’ contextual reasoning becomes an increasingly important research problem. Existing evaluations largely focus on whether models produce the correct normative judgment for a fixed scenario, such as classifying the appropriateness of an information flow or identifying the relevant transmission principle. While success on these tasks demonstrates that models can reproduce norm-consistent judgments in specific settings, Contextual Integrity requires substantially richer capabilities. Agentic systems must identify which norms apply, recognize uncertainty, reconcile competing norms, generalize across unfamiliar contexts, and maintain consistent reasoning over long-running interactions. Existing results are therefore encouraging, but do not resolve the central challenge. New, richer benchmarks are necessary to answer questions about CI capabilities, and such benchmarks are discussed further in §10.

Open Problem 6.3: Representing contextual norms.

How should contextual norms be represented, maintained, and adapted across communities and cultures?

Even if large language models encode substantial knowledge of social norms through pretraining, this knowledge is unlikely to be complete, coherent, up to date, or universally applicable. If we view this limitation as an inherent feature, rather than a system weakness to fix, then we need flexible mechanisms capable of extracting, characterizing, and validating contextual norms on an ongoing basis from diverse and reliable sources.

LLMs as such a mechanism hold promise (in contrast to formal, rule-based systems that struggle with flexibility). Existing training approaches, which primarily rely on formal privacy policies (Hu et al., 2025) or synthetic scenarios (Mireshghallah et al., 2024; Fan et al., 2024), are limiting: formal policies encode only the subset of norms that have been explicitly codified, and synthetic scenarios sample from a finite, constructed space. Neither approach captures the full nuance and diversity of real-world norms, which include community rules, organizational policies, legal requirements, and observed social practices (Apthorpe et al., 2018; Nissenbaum, 2009). Increasingly capable agents offer an opportunity to move beyond these constraints by learning from a broad range of normative sources (Franchi et al., 2026).

Awareness of norms, while a necessary and important foundation, is insufficient. For agents to make normative, action-guiding judgments, they need the capacity to reason about interacting norms, such as hierarchical relationships or apparent conflicts. This capacity would need more than just the ability to accurately identify values for the five information-flow CI parameters or corresponding state-change CI parameters. Metadata about the norms’ sources of authority, how universally held the norm is, the consequences and severity of violations, etc., may be useful for reasoning, once relevant norms are identified.

How can we best capture these norms and norm properties within an agentic system? Enumeration of contextual social norms in a *registry* that contains such metadata could be part of the solution, enabling explicit curation of norms and their properties. Emerging model architectures like Limited Memory Language Models (LMLMs) (Feldman et al., 2026; Zhao et al., 2025) suggest the usefulness of such a registry by storing factual knowledge (i.e., contextual norms) in a knowledge base during pre-training, allowing the trained model explicit retrieval capability during generation.

Known (and auditable) social norms can help fill under-specified gaps in user intent, and a registry could even provide rules or principles for resolving conflicts between norms in exceptional cases (e.g., when a right to privacy regarding mental health conflicts with the need to mitigate immediate dangers to oneself or others). Explicit norm enumeration is itself an unsolved challenge, but one that can be approached ahead of time, out of band with live agentic operation.

More broadly, developing taxonomies of contextual norms that support efficient retrieval, conflict resolution, and policy generation remains an important open research direction—many of the core challenges in *understanding* contextual norms come from the need to actually apply them afterward, going beyond a *tacit knowledge* of norms. All of these capabilities are needed to produce appropriateness judgments, which we turn to next.

6.4 Judging appropriateness

In order to operationalize a behavioral judgment, that judgment must be expressible in terms of specific actions to take or prevent. However, agents operate in dynamic, open-ended contexts where natural language is used to attain sufficient flexibility in expressing intents. Appropriateness judgments must further be based on the set of contextual norms that govern information flow or other behaviors, which themselves are expressed via natural language parameters and/or difficult-to-interpret model weights. In the absence of oracular LLMs (as discussed above), systems require formal logic to execute constraints. When we express intents and normative guidance in natural language (e.g., “don’t be creepy,” “keep this between us”), static rule-based translation is insufficient.

Open Problem 6.4: Formal policies that capture norms expressed in natural language.

Policies for social norms must be derived from unstructured data; how do we develop tools that ensure the accuracy of generated policies?

The traditional approach to making explicit, enforceable judgments is to express them in the form of a *policy* over agent actions (Schneider, 2000). In order to support dynamic, open-ended contexts, including new tools and capabilities that might be discovered at runtime, agent policies themselves can be generated dynamically, with open-ended system context that adequately captures the social context as input. One example of dynamic policies (for non-agentic execution) integrates policies as a state machine that tracks the task execution and limits actions based on the current state, i.e., a use-based approach (Birrell et al., 2018; Bagdasarian et al., 2019).

Recently proposed systems for dynamic policy generation such as Conseca (Tsai & Bagdasarian, 2025) and Progent-LLM (Shi et al., 2026) express policies as predicates over tool actions, providing a simple and practical interface with agent behavior. Other systems like CaMeL (Debenedetti et al., 2025), IsolateGPT (Wu et al., 2025), and f-secure LLM systems (Wu et al., 2024a) generate dynamic policies expressed as execution plans. Sapien (Tiffany et al., 2026) proposes dynamic *stateful* policies that anticipate changes to context; Palumbo et al. (2026) employ *Datalog* as the policy language to enable expressivity and static analysis; VeriGuard expresses formal constraints as policy functions in statically verifiable programming languages in order to support formal verification of agent-generated code (Miculicich et al., 2025); Solko-Breslin et al. (2027) provide policies that track overall action risk given estimated probabilities of failure; CEL (Google LLC, 2021) provides an efficient language for predicate logic and authorization rules; and Cedar provides verifiable role-based and attribute-based access control policies (Cutler et al., 2024), with Dogwood extending the policy language with temporal logic (Amazon Web Services, 2026). No consensus has yet emerged around a standard policy language for arbitrary agents, nor even around the set of necessary or sufficient properties that such a language ought to have.

Several of these systems adopt a common design pattern, which we call a *contextual policy engine*:

Definition (Contextual Policy Engine): A *contextual policy engine* for an agentic system is a component which fulfills two key responsibilities:

1. **Dynamic policy generation** that maps context and norms into a formal policy on (formally modeled) agent actions, expressed in a machine-checkable language. If the policy generator is unable to determine a formal policy, it may also escalate to the user in order to clarify their intent.
2. A **runtime enforcer** that, given a policy, proposed agent action, and current execution state, allows the action, denies it, modifies it, or escalates to the user for confirmation.

Architectural separation of contextual policy engines from the core “planner” model that proposes agent actions has theoretical and practical advantages. For example, Conseca (Tsai & Bagdasarian, 2025) takes advantage of this separation to avoid even indirect prompt injection by breaking any connection between un-

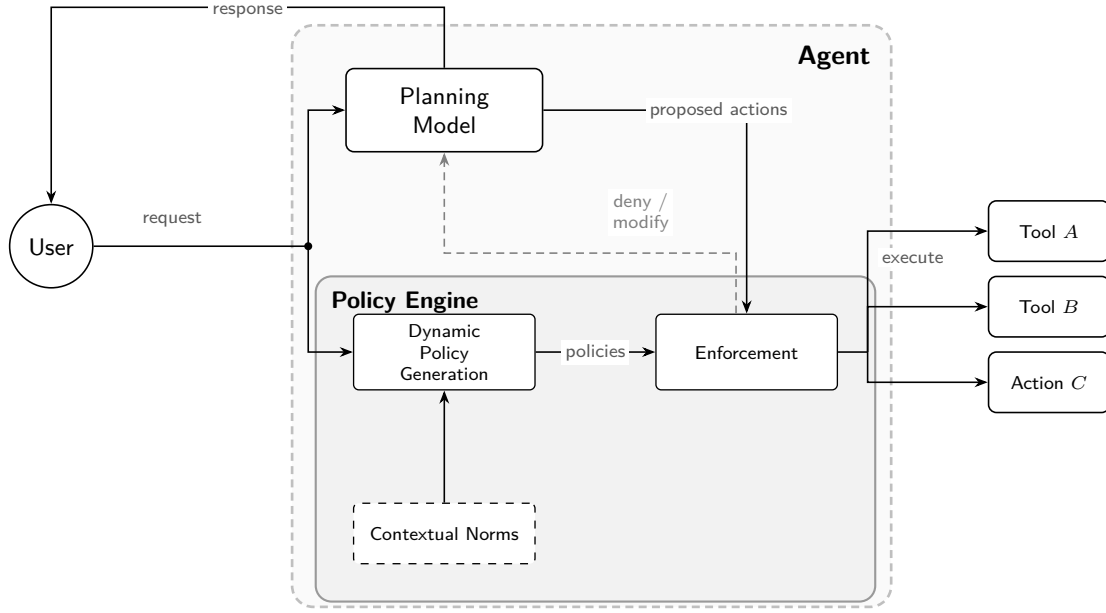


Figure 2: An illustrative example of how a simple contextual policy engine architecture could be instantiated within an agentic system. Other architectures are possible; the “ideal” is an open problem.

trusted inputs and policy generation. Industry safety architectures, too, increasingly layer policy enforcement components around the model rather than relying on alignment alone (Sharma et al., 2025; Cunningham et al., 2026; Google, 2026d; OpenAI, 2026c). We discuss the enforcement side of this pattern further in §6.7.

Regardless of auxiliary system context, how can we build “compilers” that reliably translate high-level, ambiguous natural language policies into formal, verifiable logic (e.g., temporal logic) without losing the nuance of the original intent? “Policy-as-Code” frameworks such as Open Policy Agent (Cloud Native Computing Foundation, 2026a) require manual rule-writing, and basic LLM-based “constitutional AI” (Bai et al., 2022) relies on prompting rather than formal verification. ControlValve (Jha et al., 2026b) attempts to chart a middle ground, using natural language prompting to generate a context-free grammar agent-system policy with allowed and disallowed sub-agents and tools. Any successful approach will require the definition of system-level enforcement abstractions onto which formal policies must be mapped—i.e., what is the formal policy language?

Open Problem 6.5: Preventing harmful context from affecting policy generation.

Can we efficiently separate trusted and untrusted (e.g., adversarial or misaligned) data even in complex scenarios?

An ideal policy generator would have access to the full breadth of information surrounding a request: the user’s exact intention, complete personal information, all relevant contextual norms, and trusted information about the systems and information with which the agent will be interacting. However, in practice, this information is unavailable. Beyond the inherent nuances, unstated details, and other sources of uncertainty in user-to-agent communication, a contextual policy engine must also be able to make decisions in the face of *adversarial* inputs (e.g., prompt injections) and *misaligned* system context (e.g., an incorrect set of contextual norms to apply to the task at hand).

How can contextual policy engines identify and ignore adversarial and misaligned system context when generating contextual policies? One approach taken by many proposed systems (Tsai & Bagdasarian, 2025; DeBenedetti et al., 2025; Foerster et al., 2026; Wu et al., 2024a; Costa et al., 2025; Kolluri et al., 2026; Nikolic et al., 2026) has developers explicitly differentiate between sources of *trusted* and *untrusted* system context. Conseca, for example, isolates its policy generator from untrusted inputs and generates policies only from

trusted system context. Other approaches (Hu et al., 2026; Beurer-Kellner et al., 2025) relax this isolation boundary via low-entropy information bottlenecks, allowing some influence from untrusted content in order to improve utility while still protecting against context hijacking. §7 describes open problems in designing such systems for contextual authorization.

6.5 Operational conflicts

User preferences may conflict with societal norms, such as those derived from legal safety rules (e.g., a user asking an agent to hide a reportable disease when a rule requires that it be conveyed to public health authorities), or conflicts may arise in situations in which multiple agents’ policies conflict.

Open Problem 6.6: Establishing precedence between conflicting policies.

How can we formalize orders of precedence between conflicting policies to achieve robust, well-justified, and well-understood outcomes? How can we structure negotiations between mutually untrusted agents to facilitate compromise and/or progress?

Agents act on the user’s behalf in social settings, so conflicts between the user’s preferences and social norms are not just errors—they are social violations. When conflicts arise within a single agent’s policies, a taxonomy of norms covering their sources of authority, epistemic certainty, the severity of violations, the values and purposes that underlie them, etc., could be useful for determining precedence and identifying the appropriate behavior.

An ideal policy engine will not simply override user queries with naïve priority lists, however—it will distinguish between preferences (which can be overridden), values (which should be respected), and requirements (which must be obeyed). These layers of policy types may also interact differently in different contexts (e.g., agents deployed in two countries with different laws or norms). Previous literature on automated negotiation of user-defined privacy policies could be a useful foundation (Mosca & Such, 2021).

Conflicts can also arise in multi-agent systems when agents have different privacy, security, and data needs, or represent different communities with different contextual norms. Technical challenges in this area are discussed further in §11, and associated governance questions in §12.

6.6 Resolving uncertainty

Many kinds of uncertainty can complicate normative judgments: uncertainty about the user’s intent, about the environmental context, about the consequences of certain actions, and about normative gaps or conflicts, such as those described above. Appropriate navigation of uncertainty is a major open challenge, and different types of uncertainty will likely need different mitigations.

Uncertainty from ambiguity is inherent in natural language expressions. Model-level capability (§8) is needed to identify and resolve uncertainty within natural language norms, preferences, and user intent, such as through high-quality reasoning and tool-based exploration of the environmental context.

However, even an oracular model may not always be sufficient—user intent and preferences can be fundamentally underspecified. The model may miss unstated nuances, such as what types of actions or responses would be “creepy” in what contexts. Sometimes users will not even be aware of their preferences or how to communicate them precisely. In other words, there can exist a “translation gap” between vague intent and executable code, and this is a primary failure mode. If a precise, formal translation does not exist, how can we help the user realize that, understand the system’s limitations, and settle on a specific interpretation of their intent?

This implies a need for human interaction and feedback in order to disambiguate intent and secure consent or confirmation (§9). Frequent interactivity may be justified for highly critical requirements like those coming from laws (GDPR, HIPAA, etc.), but will often be too expensive for less critical policies reflecting a specific user’s preferences. It could even lead to consent fatigue (Felt et al., 2012; Choi et al., 2018; Anderson et al., 2015) or abandonment of the agent system entirely.

A sufficiently robust and reliable AI delegate, such as an LLM prompted or fine-tuned to reflect user preferences, could mitigate this toil without infringing on the user’s requirements or expectations, minimizing human interaction to only truly ambiguous decision points. How can we build reliable AI delegates for users that they can trust to represent their privacy preferences accurately across a range of contexts and tasks? Note that these AI delegates need not be entirely independent components of the policy engine system—“user-delegate” personalization could be built into the policy generator itself, or provided to it as additional trusted system context.

Autonomous navigation of uncertainty may also be possible when the environment itself presents unknowns. If AI assistants can identify environmental uncertainties—such as a lack of clarity about whether the setting of a request is casual conversation or a medical emergency—they could be allowed to interact with the environment in order to gather more information. Relatedly, if AI models can detect and estimate the probability of harm associated with an action (e.g., the probability that a document to be opened or shared contains sensitive information), this enables policy frameworks like [Solko-Breslin et al. \(2027\)](#) to track and enforce some estimated ceiling of risk tolerance.

6.7 Enforcing judgments on agent behavior

Given an appropriateness judgment, we would like the agent to deterministically ensure that its behavior complies, if possible. In the absence of oracular agent models, this implies an explicit enforcement mechanism such as that described in the contextual policy engine architecture above (Figure 2).

Open Problem 6.7: Practical enforcement mechanisms to ensure appropriate behavior.

Enforcement mechanisms must be practical at scale along many axes: number of users, number of queries, number of concurrent agent loops, number of different agent systems, number of policies, and number of capabilities.

Some of these axes of scalability “merely” have implications for acceptable costs of operating the engine: latency and computational resources. How do we design contextual policy engines that are computationally efficient enough to check complex, stateful permissions for billions of concurrent agent loops?

Some point toward a need for reusable abstractions to characterize actions and express policies themselves. Should we assign formal semantics to the capabilities of the agent, and if so, how? This could be particularly important for agents that operate in quickly evolving environments wherein possible actions cannot be enumerated in advance. For example, long-running coding agents might plan Bash commands or Python scripts that can cause many different types of state changes (modify environment variables, move files, change file permissions, etc.), and all these semantics need to be modeled in order to enforce policies that are reasoned about and expressed in terms of higher-order concepts like information flow control.

Others have implications for the system design itself—is there a reference architecture that could be easily adopted by any new agent in the ecosystem? How can we verify and/or audit the policy enforcement mechanisms across many different agents or instantiations of agents? These questions are closely linked to the system-level problems (§7) of how to define trust boundaries and audit how information passes through them. §11 discusses open problems in this area that arise from agent-to-agent interactions.

The enforcer must also handle situations where the appropriateness judgment is indeterminate, such as by employing the strategies mentioned above in §6.6 (with user escalation in particular discussed further in §9).

Part II: Component-Level Perspectives

Realizing contextual privacy and security requires addressing three levels of agent components, namely the agent’s (1) *system-level* enforcement abstractions; (2) *model-level* capabilities to identify and resolve uncertainty within natural language norms, preferences, and user intent; and (3) appropriate *user-level* strategies for efficiently and reliably navigating ambiguity while avoiding consent fatigue or other externalities.

§7, §8, and §9 dive into specific open problems at each of these respective levels. These discussions expand beyond the problems of operationalizing contextual privacy and security to also address the deeper problems introduced by each agent component (e.g., from agent identities and accountability at the system level, to social dynamics and novel privacy risks at the user level).

7 Challenges with Systems Security and Privacy for AI Agents

Summary. This section describes the increased challenges in agentic systems compared to traditional ones that must be solved to make agentic AI fully realizable. We look at new challenges across the various pillars of security (Authentication, Authorization, and Accounting) and privacy brought about by the use of unstructured intent and system context, and the decreased role of human oversight in agentic systems. For example, agents require distinct authentication protocols to distinguish automated actions from human ones and trace malicious actions back to a responsible party. Traditional authorization must evolve to accommodate the just-in-time and imprecise nature of agent executions, and manage dynamic credential lifetimes. Furthermore, agent data access and action without manual oversight complicate user privacy, making data minimization, data deletion, and user anonymization much more complex.^a

We end with a discussion (§7.5) about a *fundamental* reason why these agentic systems require rethinking traditional solutions: unlike traditional systems with deterministic code, achieving the desired agent functionality requires probabilistic AI models to dynamically interpret developer intent and execute workflows at runtime, leading to an “intent-specification gap” that allows agent executions that can break user expectations. §7.5 suggests that rethinking the on-the-fly agentic paradigm and adding processes for structured agentic control flow can greatly benefit security while potentially preserving utility, and encourages research and development in this direction.

^aOther works previously collected different perspectives on agentic security and privacy; for example, [Christodorescu et al. \(2025\)](#) and [Zhang et al. \(2025a\)](#) look to see how agentic AI can be framed by a mindset of traditional security principles. We find these help provide alternative perspectives on this unsolved problem.

We organize top-of-mind open problems for AI agent systems along the classical security mechanisms ([Sandhu & Samarati, 1996](#); [de Laat et al., 2000](#)) of *Authentication* (establishing who is acting), *Authorization* (determining what they can do), *Accounting* (recording what they did for audit and accountability), and *Privacy* (controlling what information flows where). These mechanisms are well-understood for systems in which the acting entities are either humans or deterministic software. But an agent is neither a human, who acts intentionally, nor a deterministic program, whose behavior can be audited from its source code—rather, an agent is a stochastic actor whose behavior emerges from LLM inference over an unpredictable model context.

7.1 Authentication and Identity

As agents begin executing actions on the web (e.g., clicking “Buy” in the browser), the distinction between a user’s and an agent’s actions blurs. This poses problems for security systems, because identity forms the foundation of all security decisions: authorization, accounting, and data governance all depend on first establishing who is acting. If identity is ambiguous, every downstream security mechanism is undermined.

Traditional systems have well-established mechanisms for establishing the identity of principals, including OpenID Connect (OIDC) ([OpenID Foundation, 2026](#)), X.509 certificates ([IETF, 2008](#)), SPIFFE ([Cloud Native Computing Foundation, 2026b](#)), Decentralized Identifiers (DIDs) ([Sporny et al., 2026](#)), and Verifiable Credentials (VCs) ([Sporny et al., 2025](#)) (recent work has already explored applying these to agents ([Garzon et al., 2025](#))).

These traditional mechanisms all share a common assumption: the entity being identified is either a human (who acts intentionally) or a deterministic piece of software (whose behavior is defined by its code and can be audited). Agents break both assumptions.

An agent is neither a human nor a deterministic program: it is an actor who can perform an incredibly wide range of tasks and whose actions are stochastic. This creates two main challenges: how to distinguish between an agent’s actions and human ones; and how to link agents back to the entities (users or agents) that deployed or instructed them.

Open Problem 7.1: Distinguishing agents from humans.

How can we distinguish between user actions and agent actions when existing identity protocols do not recognize agents as distinct entities?

Because existing identity protocols do not recognize “agent” as a distinct entity type, service providers cannot distinguish between user actions and agent actions, nor can they link an agent’s action back to the user who initiated it. This problem is especially critical in computer-use agents, which interact with web UIs by clicking buttons, filling forms, and navigating pages exactly as a human would (without any means of distinguishing between user and agent actions). This can also easily lead to agents being misled by adversarial environments ([Zhan et al., 2026b](#)).

Although some services offer bot account types, these require significantly more effort to set up than reusing a human account. This creates a perverse incentive: the path of least resistance is for users to give their agent a human identity (e.g., an email address) or their own credentials directly. For example:

- OpenClaw ([OpenClaw, 2026](#)) agents are routinely given email credentials ([Ahl, 2026](#)), enabling them to register on websites, create cryptocurrency wallets, and hire human workers ([RentAHuman, 2026](#)), giving agents largely unsupervised freedom to act on the Internet ([Ries, 2026](#)).
- On Moltbook ([Moltbook, 2026](#)) (a “social network for agents”), which has seen a lot of “agentic” posts that sparked heated discussions on the Internet (e.g., agents conspiring against humans ([Husain, 2026](#))), there is no clear way to distinguish between humans posting through the API, agents acting on user prompts, and agents posting autonomously ([Heaven, 2026](#)).
- AI agents are already submitting significant volumes of code to open-source repositories without any way for maintainers to distinguish agent-generated from human-generated contributions ([punkpeye, 2026](#)). And agents have already written public defamation “hit pieces” on developers for not accepting agent-generated code ([Rathbun, 2026](#); [Shambaugh, 2026](#)).

To solve this problem, one approach is defining a protocol that has a first-class concept of “agent” as a distinct entity type, with an identity that binds the user’s identity, the agent software’s identity (model, version, guardrails), the specific task or session, and the delegation chain through which authority was granted.

Related work: NIST AI Agent Identity and Authorization ([Booth et al., 2026](#)), OpenID Identity Management for Agentic AI ([South et al., 2025](#)), Microsoft Entra Agent ID ([Microsoft, 2026](#)), IETF Agentic Identity Schema ([IETF, 2025c](#)), Linux Foundation AGNTCY ([AGNTCY, 2026](#)).

Open Problem 7.2: Tracing agents back to users.

How does an agent express “I am acting on behalf of User A right now” in a way that is cryptographically verifiable and auditable?

How do we design identity systems for agents that serve multiple principals or operate across multiple identity contexts?

How should identity propagate through delegation chains while maintaining cryptographic traceability back to the originating user?

When an agent acts, how can we trace back the action to the responsible party (whether that is the user who prompted it, the developer who built it, or the organization that deployed it) to enable accountability? Where accountability for agent actions should lie is itself an open question, but regardless of the answer, the ability to trace an agent’s actions back through its delegation chain is a prerequisite. For example, when an agent causes harm to external parties (a fraudulent purchase, a data breach, or a defamatory post), service providers may need to identify the responsible party or deny future requests by agents operated by the same organization.

Agents operate in shared contexts (e.g., group chats, shared repositories) where they receive instructions from multiple users simultaneously. When a shared agent (e.g., a team’s group chat bot) takes an action, whose identity is attached to it? Under current protocols, the agent authenticates as a single service account or with a single set of client credentials, and there is no standard mechanism to express that an agent is “currently” acting on behalf of a certain user. This makes it impossible to trace a specific action in a shared context back to the user who instructed it.

Agents do not operate in isolation: they call remote tools (using MCP) and delegate sub-tasks to other agents (via protocols like A2A). In a multi-hop chain (user → root agent → sub-agent → remote MCP tool), each hop involves an independent entity making its own decisions. The main identity question in such scenarios is: what identity does the downstream service see? If a deeply nested sub-agent takes a harmful action, there is currently no mechanism to trace back through the full chain to the originating user and identify where things went wrong (see §7.3 for intent-binding mechanisms).

Related work: Google A2A Agent Cards ([A2A Project, 2026](#)) for capability and identity advertisement; OWASP Agent Naming Service (ANS) proposals ([ruvnet, 2026](#)) for capability-aware discovery with identity verification.

7.2 Authorization

Agents perform actions on behalf of users, which requires users to delegate their authority to such agents. In traditional software, this delegation is straightforward: a user grants an application specific permissions by consenting to a well-defined, understandable set of actions (e.g., “save photos to cloud storage”), and the application uses the user’s authority to perform those actions.

Agentic systems disrupt this model. An agent tasked with “planning a trip” may check the user’s calendar for availability, search flights, book hotels, and see their bank balance. This chain of actions is determined at runtime and cannot be fully anticipated during an initial consent step. This problem is also referred to as the “intent-execution separation problem” ([IETF, 2025b](#)). More meaningful authorization for agents would (1) enforce the principle of least privilege (i.e., grant only the permissions necessary for the task via context-aware authorization); and (2) provide just-in-time (JIT)

authorization (i.e., provision those permissions only when needed and revoke them when the task is completed).

Moreover, agents are long-running, autonomous, and multi-purpose, and can spawn sub-agents; so even if the initial consent was given for a specific task, there is no guarantee that the agent will not reuse those permissions for other tasks. Furthermore, adversarial inputs can result in prompt injection, which often leads to the abuse of permissions above and beyond the originally intended scope (Rehberger, 2025b;a). As a result, the assumptions underlying traditional authorization no longer hold.

Limitations of Traditional Mechanisms. The problem of delegating authority and constraining access is well-studied. OAuth 2.1 (IETF, 2026) is the current standard for delegated access that allows a user to grant scoped access to a client application via a one-time consent step. OAuth 2.1 is becoming the norm for agentic protocols such as MCP (Anthropic, 2024), but it was designed for traditional apps and has limitations that become critical for agents: coarse-grained scopes, a static consent model, and no mechanism for delegation attenuation (sub-agents inherit the full scope of their parent) (Chopra, 2025).

These limitations are not new and have long been studied in the trust management literature. GNAP (IETF, 2024) addresses some of OAuth’s rigidity by separating the requesting party from the resource owner and supporting dynamic grant negotiation. Macaroons (Birgisson et al., 2014) introduce attenuation by construction: chained HMAC-based caveats that can only restrict (never expand) permissions along a delegation chain. SPKI/SDSI (Rivest & Lampson, 1996; IETF, 1999a) binds authorizations directly to public keys with support for transitive delegation. PolicyMaker/KeyNote (Blaze et al., 1996; IETF, 1999b) is a trust management system that frames authorization as a logical deduction problem over signed policy statements. However, none of these systems have been applied to agents.

Sandboxing—placing the agent in a container or VM to limit agent access (e.g., by OpenAI (OpenAI, 2026b), NVIDIA (Harang, 2026), IronClaw (NEAR AI, 2026), and Google Agent Sandbox (Google Cloud, 2026b))—faces the same issues as OAuth or other static and manual access controls. In particular, these approaches require some authorized party (typically the developer or agent user) to set up binary and coarse-grained sandbox permissions, and ask for manual consent to increase permissions. Trusted Execution Environments (TEEs) could also be leveraged to enable client-side sandboxing (Abdollahi et al., 2026).

Open Problem 7.3: Enabling context-aware authorization.

How can authorization systems enforce the principle of least privilege for agents, granting access not just to specific resources but only under the right conditions and for the right purpose? Such solutions should be aware of manual consent fatigue and the ambiguity of the agent’s specified purpose. How is authority delegated among parties (agent or non-agent, and from parent to sub-agents)?

When a user instructs an agent to perform a task, the set of required permissions is not known in advance; it is dependent on the natural-language instruction and the agent’s evolving plan. This is fundamentally different from traditional authorization, where permissions are derived from a well-defined application specification.

The same permission (e.g., read access to medical records) may be appropriate for one task context (finding a doctor) and a serious privacy violation in another (drafting a social media post). Can authorization become context-aware? Could an authorization infrastructure evaluate not just who is requesting access and what they want to access, but why and whether it is appropriate in a given context? Traditional mechanisms can express conditions like “read-only” or “valid only on weekdays,” but they cannot express “only when helping the user find a doctor.” The same problem applies to agents issuing natural-language sub-tasks to other agents in a delegation chain.

Furthermore, the required set of permissions evolves during execution. An agent that is tasked with “helping to resolve a customer complaint” may initially only need email access, but mid-task discover that it also needs access to the billing system and internal monitoring logs. Each new permission requirement is discovered

at runtime as the agent’s plan unfolds. This means authorization cannot be a one-time, upfront decision; it must be an ongoing process performed by the authorization infrastructure.

One obvious approach is to put a human in the loop to approve each authorization request. But given the speed at which agents operate, this can quickly lead to consent fatigue: users may start approving all requests without reviewing them, paradoxically undermining the security that the approval step was meant to provide. An alternative might use model-generated policies, but developers must be careful to avoid model manipulation attacks from untrusted policy inputs.

Related work: CaMeL (Debenedetti et al., 2025) for execution isolation between trusted planner and untrusted data processor; Contextual Agent Security (Conseca) (Tsai & Bagdasarian, 2025) for system-context-dependent policy generation; Sapien (Tiffany et al., 2026) for contextual generation of stateful policies; SAGA (Syros et al., 2025) for centralized agent access control; Progent (Shi et al., 2026) for fine-grained tool policies; FORGE (Palumbo et al., 2026) for Datalog-based policies; MiniScope (Zhu et al., 2025), which asks the user to approve computed minimal OAuth scopes; and Siu et al. (2026a) for a formalization of what authorization based on (system) context requires.

Open Problem 7.4: Managing the authorization lifecycle.

How can authorization systems issue credentials for the correct lifetime? Can we determine how long a task will take to complete? How do authorization systems know when to revoke credentials when the task completes or something goes wrong? Does the agent self-report, and if so, what happens if the agent crashes or misreports its status? How can agents dynamically negotiate and request additional permissions mid-task as their current goals change, without requiring full re-authorization? Would each new authorization flow require user consent, leading to potential consent fatigue?

In traditional authorization systems (e.g., OAuth), users pre-provision authorization tokens with a fixed time-to-live (TTL): they are valid until they expire or are explicitly revoked. Agent tasks can last anywhere from seconds (a quick email lookup) to days (a complex research project), and their duration is often unpredictable at the time of provisioning.

The core problem is that no current system ties the credential lifecycle to the task lifecycle. The authorization server does not know when a task begins or ends. Fixed-TTL tokens risk either expiring mid-task (forcing re-authorization and interrupting the workflow) or outliving the task (creating stale tokens with lingering permissions, which have led to large-scale attacks in the past (U.S. Department of Energy, 2021)). And at the speed agents operate, they can accumulate hundreds of stale tokens before any human reviews them, creating an unnecessary attack surface (e.g., yesterday’s social media permissions carry over to today’s medical task).

A further question is what granularity a “task” should have: is “plan my trip to Europe” one task, or are “book flight,” “book hotel,” and “arrange transport” three separate tasks with three separate credential scopes? And how does the system learn that a task has completed: does the agent self-report, and what happens if it crashes or misreports its status?

Related work: GNAP (IETF, 2024) for dynamic grant negotiation; IETF Agentic JWT (IETF, 2025b) for workflow-bound tokens; IETF Decoupled Authorization for Agent2Agent (IETF, 2025a) for intent-based JIT permissions.

7.3 Accounting

In traditional systems, every action is tied to an explicit act of user intent: button click, form submission, etc. The link between what the user wanted and what the system did is direct and verifiable. Agents break this link. When a user issues an instruction, the agent interprets it and translates it into a sequence of concrete actions. The user’s instruction is high-level and necessarily underspecified (natural language leaves most operational details implicit), while the actions the agent takes are concrete and specific. Every detail the user left unspecified was decided by the agent autonomously.

Open Problem 7.5: Ensuring that an agent’s action is intended by the user.

Can we create verifiable digital artifacts binding agent actions to original user intent (e.g., “Buy groceries under \$50”) when “faithful execution” is complicated by the probabilistic nature of LLMs? How do we design standardized architectures for forensic observability that enable attributing agent actions to the user (bad prompt), the developer (software bug), or the model (hallucination)? (We note that creating these accounting mechanisms also requires Authentication and Identity (§7.1): tracing agents back to users and distinguishing an agent’s action from a user’s.)

This creates a non-repudiation problem that is qualitatively different from that of traditional systems. If an agent purchases an item due to a hallucination or prompt injection, the user may want to claim that this action was not intended. At the same time, the service provider (e.g., merchants, banks) needs a mechanism to verify whether a user is falsely claiming “agent error” to reverse a legitimate transaction. Also, when something goes wrong (a fraudulent purchase, leaked private data, a deleted production server), we need to reconstruct what happened, why it happened, and who is responsible.

Traditional non-repudiation relies on digital signatures binding a principal to an action. For example, append-only signed logs such as Certificate Transparency (IETF, 2013) and Trillian/Tessera (Transparency.dev, 2026) provide tamper-evident and publicly verifiable audit trails (such as logs of issued TLS certificates).

For agents, can we provide intent-binding artifacts—tamper-evident structures that capture the full chain from user instruction to agent action? Such an artifact would need to include the user’s original instruction (authenticated by the user), the agent’s interpretation of that instruction (the plan or reasoning trace), and the specific actions taken, bound together in a way that third parties can verify.

Related work: Authenticated Workflows (Rajagopalan & Rao, 2026), zkML (Chen et al., 2024), zk-LLM (Sun et al., 2024), IETF Decoupled Authorization for Agent2Agent (IETF, 2025a), Secure Intent Protocol (IETF, 2025b), Agent Payment Protocol (AP2) (Agent Payments Protocol Working Group, 2025).

7.4 Privacy

To prevent leakage of personal data, traditional app and web-service systems have relied upon a number of system-level strategies that include PII detection and replacement; data minimization (ensuring only the minimal amount of data is used for the task at hand) (Ganesh et al., 2025); and data deletion (via user deletion controls as well as system deletion defaults, and including the removal of inadvertently shared or stored data). These strategies traditionally employed data filtering based on static constraints. Another critical pillar for controlling one’s digital footprint is to engage with the web anonymously by using anonymous credentials and anonymous authentication systems as proactive controls to decouple a user’s identity from their online activities before data is even generated.

The semi-autonomous nature of AI agents—characterized by tool use, recursive memory retrieval, and multi-step reasoning—recasts privacy as a dynamic architectural challenge rather than a simple static filter.

As agents act on behalf of users across disparate services, they create an expansive and persistent digital footprint that blurs the lines between internal data processing and external identity disclosure. Managing privacy in this setting may require implementing PII replacement, data minimization, and deletion capabilities throughout the entire agent execution lifecycle—from initial reasoning to final tool invocation (Staab et al., 2026). Preserving anonymity becomes significantly more complex as agents undermine “unlinkability”; for example, they may generate novel data side channels and employ dynamic delegation of user credentials.

In the text below, we discuss the four strategies outlined above, and how each must change to account for privacy in the agentic setting.

Open Problem 7.6: Evolving PII detection and replacement.

- How can we replace PII while preserving essential model context for reasoning?
- Can we prevent leakage from agent-specific characteristics, namely chain of thought and long-term memory?
- Can we build the equivalent of traditional network firewalls that work on the semantic level, understanding user intent and privacy consequences of agent actions to monitor all data flows (e.g., privacy firewalls ([Abdelnabi et al., 2026](#)))?

PII detection and replacement techniques need to shift from static data protection to managing dynamic, multi-step reasoning cycles. In conventional applications, PII is typically localized within structured database fields where redacting a name has zero impact on the underlying code logic. However, for an agent, this “blunt” redaction strategy can degrade performance. If an agent is tasked with “Emailing the CEO about the Q3 budget” and the term “CEO” is redacted, the system loses the essential semantic nuance required to identify the correct recipient or maintain an appropriate professional tone. Approaches being explored today focus on replacing PII while simultaneously preserving context.

Two distinct risks emerge in this agentic environment, rendering traditional pattern-matching scanners less effective at PII detection. The first is chain of thought (CoT) leakage, a structural risk where sensitive data is inadvertently recorded within the agent’s internal reasoning “scratchpad” or intermediate tool-argument logs. In this scenario, the vulnerability lies in the persistence of raw data during the execution phase, creating a fragmented trail of PII even if the final output appears clean.

The second risk is inference-based re-identification, a probabilistic risk involving long-term memory (LTM) or RAG. Here, an agent may leave disparate details sprinkled across various memory stores, and this opens the door to cross-referencing—for example, pairing a rare medical condition with a specific job title—to reconstruct a user’s identity or infer sensitive attributes (as demonstrated by [Lermen et al. \(2026\)](#)).

Open Problem 7.7: Achieving data minimization.

[Semantic Boundary Problem] How do we determine what data is necessary for tasks expressed in natural language? Are (trained and specialized) LLMs the best approach to understanding contextual nuances relevant to data use?

[Model Context Contamination] How do we ensure minimal and appropriate information flows of data within an agent’s reasoning cycle?

[Tool Over-Privilege] How can we grant agents minimal privileges when the exact data requirements are initially unknown, and without burdening the user?

The challenge of data minimization in agentic systems is particularly thorny because, unlike traditional software, agents rely on semantic context rather than just structured data fields. While the principle of least privilege is a worthy starting point, applying it to autonomous entities that reason over natural language introduces several unresolved research gaps.

The first open problem centers on the semantic boundary problem, where the unstructured nature of natural language makes it difficult to define what constitutes “minimal” necessary data. Unlike traditional systems that redact specific data fields, agents must determine which contextual nuances are relevant to a task and which are sensitive liabilities. Prior work in standard inference settings has shown that this necessity boundary can be individualized, with predictions often requiring only a subset of a user’s features while preserving utility ([Tran & Fioretto, 2023](#)). In the agentic context, one proposed approach is to use “Dual LLMs” ([Willison, 2023](#)), where one LLM is a Privileged LLM (that works on trusted inputs) and the other is a Quarantined LLM that may operate on untrusted content; for example, AirGapAgent ([Bagdasarian et al., 2024](#)) separates data selection (reasoning about what is necessary using only trusted context) from

task execution (talking to the outside world). [Zhou et al. \(2026b\)](#) formally operationalize data minimization for LLM prompting by defining the optimal disclosure as the least privacy-revealing version of a prompt that preserves task utility, showing that stronger frontier models leave considerable room for data minimization, yet are not reliable predictors of the optimal minimized prompt.

The second problem, model context contamination, occurs when data minimization fails within an agent’s reasoning cycle. In typical agent architectures, a single, “flat” context window serves as the primary memory space, leading to the excessive retention and processing of data that is not strictly necessary for every sub-task. From a data minimization perspective, providing an agent with a user’s entire history for a simple tool call violates the principle of “necessity,” as sensitive information from a private sub-task (e.g., parsing medical records) is unnecessarily exposed to the reasoning logic of a less-privileged sub-task (e.g., performing a public web search). While traditional software achieves computational minimization through “tight interfaces” and process confinement ([Barth, 2011](#)), agentic systems often lack these granular boundaries. Ongoing approaches explore enforcing topologies of information flow (e.g., IsolateGPT’s ([Wu et al., 2025](#)) hub-and-spoke topology); more work is needed to explore dynamic topologies that both enable necessary information flows and prevent malicious ones. Additionally, for self-evolving agents, this problem can extend beyond the current context window: context-specific information or rules may be incorporated into persistent memory, controller instructions, or reusable tools and later applied in contexts where they no longer belong ([Das et al., 2026](#)).

A third problem is that of tool over-privilege, where agents reveal excessive state to external APIs because the exact requirements for success are not known upfront ([Carrillo et al., 2026](#)). Because agent-to-tool interfaces are often designed to be fluid rather than strict, agents frequently operate with broader permissions than necessary (see also problems of Agent Authorization). Existing approaches (e.g., GAAP ([Stanley et al., 2026](#)) and MiniScope ([Zhu et al., 2025](#))) can enforce various-granularity permissions that respect user privacy preferences, but impose a burden on users.

A final complication is that data minimization questions are not limited to application-layer prompts, tool inputs, or API permissions: user privacy can also leak through *encrypted traffic metadata* ([Staab et al., 2026](#)). Recent work shows that local web and research agents can reveal substantial information about user prompts and latent traits to passive network observers using only visited IP addresses, domains, and timing patterns, even when the underlying content is fully encrypted ([Jeong et al., 2026b](#)). Related studies show that autonomous browsing agents expose distinctive fingerprints across browser behavior, HTTP/TLS features, and interaction patterns, enabling websites or intermediaries to distinguish agents from humans and even attribute traffic to particular agent frameworks ([Wang et al., 2026](#); [Kang et al., 2026](#)). These results suggest that agentic data minimization questions must extend beyond deciding what semantic content to disclose to tools: should systems also minimize, pad, aggregate, or otherwise obfuscate the network traces produced by agent planning and execution? Otherwise, an agent may carefully redact sensitive fields while still leaking the user’s task, interests, or identity through the shape of its encrypted traffic.

Open Problem 7.8: Moving from data deletion to functional data revocation.

How can we support functional deletion when data persists in derivative forms or “ghost” representations?

Because agents synthesize and propagate user data across nondeterministic memory tiers, traditional deletion is no longer a guarantee of privacy. Unlike traditional persistent storage where data is literal and centralized, agentic systems store information as high-dimensional vectors and synthesized summaries, creating a situation where original PII and other personal data may persist as derivative knowledge, or “ghost” representations, even after the source records are purged. Functional revocation is an emerging concept that recognizes that deleting a record is insufficient and that instead, revocation should focus on stopping an AI from using the information. An agent might read a private email, summarize it, and save that summary into a vector database (RAG).

In an agentic world, how can we shift toward functional revocation? How can we offer data deletion that not only erases the source record, but also governs the agent’s semantic connectivity—effectively ensuring that forgotten information cannot be retrieved, reasoned upon, or leaked through derivative outputs? Should the approach of physical data management shift to one of semantic governance?

Ongoing approaches for data deletion in agentic AI settings look both to remove data stored in trained model weights themselves (machine unlearning, e.g., vector unlearning (Eldan & Russinovich, 2023) and activation steering (Turner et al., 2023)) and to delete data and derivatives that may have propagated through the entire agent ecosystem (e.g., a “privacy kill-switch” (Prakash et al., 2026) sends a delete signal to the context window, long-term memory, downstream agents, backend storage, etc.). It remains unknown whether and how comprehensive data deletion is possible to practically achieve, and what changes agent ecosystems may need to adopt to support this.

Open Problem 7.9: Protecting user anonymity in an agentic world.

How do we protect user anonymity when agents may perform user identity disclosures at scale without manual oversight? How do we balance the fundamental tension between anonymity and accountability (which relies on strong identities)?

An important aspect of user privacy is the ability to communicate, access content, and engage in many Internet services while remaining anonymous. Realizing this goal and removing digital footprints in a world without agents has been challenging enough: tools like Tor (The Tor Project, 2026), Signal (Signal Messenger, 2026), and many others strive for full anonymity, but can fall short in practice. Other tools focus instead on partial anonymity, leveraging the fact that authentication contexts typically require knowledge only of a small subset of user-identifying attributes. For example, modern frameworks for digital identity perform selective disclosure, presenting only fine-grained and relevant attributes from user credentials in each authentication setting (e.g., the European Digital Identity (EUDI Wallet) (European Commission, 2026) and mobile driver’s license (mDL) (American Association of Motor Vehicle Administrators (AAMVA), 2025)). In addition, a vast body of work in cryptography offers technologies for anonymous authentication that convey a very specific, limited amount of information required for a specific application while maintaining anonymity outside of that (e.g., PrivacyPass (IETF, 2026), which is used by reCAPTCHA (Google Cloud, 2026c), among others).

A future world where users will have multiple agents acting on their behalf across the Internet considerably complicates the question of preserving user anonymity. First, agentic behavior will considerably expand the digital footprint of users, increasing the challenge of controlling and limiting the information revealed about the user. Second, while there are mechanisms that aim to minimize identity exposure through selective disclosure for digital credentials, these rely on static disclosure policies or user decisions. In an agentic world, an agent may make selective disclosure decisions on behalf of the user, increasing the risk of credential compromise via exploitation of the agent. This relates to the problems of contextual authorization and data minimization above—the system ideally restricts agent capabilities to share certain fine-grained attributes only under the right conditions and for the right purpose while preserving the agent’s utility.

Notably, preserving anonymity in agentic systems requires more than restricting access to predefined attributes, as agents’ ability to operate over unstructured content creates broader risks of semantic inference, linkage, and re-identification. Recent research has shown that LLMs can facilitate large-scale data linkage attacks, both when explicitly instructed to do so (Lermen et al., 2026) and as an unintended side effect (Ko et al., 2026), and can even enable fully automated re-identification attacks (Li, 2026; Li et al., 2026b). These attacks demonstrate the emergent memorization and associative capabilities of frontier models, which achieve non-negligible re-identification success rates using only parametric knowledge. This research further finds that integrating results from agentic tool use substantially increases both re-identification success rates and confidence calibration, effectively democratizing re-identification capabilities. This evidence points to a development that deserves attention, suggesting that in addition to the protection of direct identifiers or specific attributes at the protocol and system level, we should also consider how to measure and protect user anonymity at the semantic level.

A complementary direction treats anonymity for agents as an overlay-network problem. ProxyGPT (Pham et al., 2025) takes this approach for LLM chatbots, proposing a customized proxy network in which volunteer browser-interaction proxies submit requests on behalf of users to services that otherwise bind queries to logged-in accounts. Although designed for chatbot access rather than general-purpose agents, the architecture suggests an agentic analogue: agents could route selected interactions through specialized intermediaries that decouple the user’s real identity from the service-facing account or session, while using audits, prompt guards, and incentive mechanisms to limit proxy misbehavior and abuse. This kind of customized overlay network is attractive because it can be layered over existing identity-requiring services without requiring those services to redesign their authentication systems. At the same time, it highlights a core systems challenge for agentic anonymity: how can the network hide user identity from services, constrain what proxies can learn or modify, and still preserve enough accountability to prevent large-scale misuse?

Finally, anonymity and accountability goals can often be at odds, and an agentic world amplifies this potential conflict. On the one hand, the need for anonymity will become even stronger for reasons we discussed above; on the other hand, accountability of agentic actions becomes essential to prevent fraud and abuse at scale. Cryptographic digital credentials offer a potential approach, providing a broad spectrum of anonymity, verifiability, and traceability properties that (1) help decouple user and agent identities; (2) provide user privacy protections during agentic transactions; and (3) still preserve the capability to identify fraud and link actions to relevant identities. These tools give us a starting point to begin designing an agentic world with the right trade-offs between anonymity, accountability, and efficiency that result in a safe, private, and practical system.

7.5 What makes an application agentic, and what does this mean for security?

At their core, AI agents are simply software applications: they contain hard-coded components that plumb together agent inputs (e.g., user prompts) and outputs (e.g., tool calls) with planned calls to LLMs. A call to model inference equates to invoking a function call that takes character tokens as input and outputs tokens; these output tokens can then be used, e.g., to direct the control flow of the program, or as variables used by downstream code. Traditional software applications conceptually operate in much the same way, but without language model invocations.

Why, then, do AI agents warrant fundamentally new security and privacy solutions? In this section, we look into this question through the lens of what makes agents “agentic,” and pose directions for exploration into the trade-offs between “agentic-ness” and security.

7.5.1 AI as the “last-mile” developer

The “agentic” nature of an application comes from the reduced need for precise developer controls: the more agentic the application, the more developers offload application development to the agent model. In traditional—“non-agentic”—software, developers manually prioritize, architect, and code specific logic paths for every supported workflow a user might want (e.g., “allow user to add item to cart, specify quantity, and checkout”). In an agentic application, by contrast, the developer can rely on the model to, e.g., dynamically generate workflows customized to potentially *niche and unpredictable* user requests expressed at a high level (e.g., “Read emails from colleagues and schedule any specified meetings”). This allows the application to support (dynamic) functionality that otherwise would be prohibitively costly to implement.

The amount the developer offloads can range from minimal—utilizing the model as a simple natural language parser to parse emails—to extreme—allowing the model to discover, create, and execute new code. Existing definitions of agentic levels capture this spectrum (e.g., Mitchell et al. (2025)).

Motivating examples along the “agentic” spectrum.

As example applications falling within this spectrum of agentic levels, we consider:

1. An “email-to-calendar-event” application that takes a natural language email and schedules a calendar event based on the email text; and

2. A “general planner” application that invokes a series of available tools in response to natural language requests (e.g., from a user) in order to satisfy that request.

These might be implemented by the developer in the following pseudocode:

```
// email-to-calendar-event implementation
prompt_email_is_event = "Return 'true' if the email contains a request to schedule an appointment, and 'false' otherwise. Email: {}"
prompt_get_event_details = "Return the date, time, and location of the event in this email as a tuple. Email: {}"

bool create_calendar_event_from_email(std::string email) {
    // The LLM acts as a boolean oracle to decide the branch.
    bool is_email_an_event = llm(prompt_email_is_event.format(email));

    if (is_email_an_event) {
        // The LLM refines the "what" (email text) into the "how" (data params).
        auto (date, time, location) = llm(prompt_get_event_details.format(email));
        // The implementation remains constrained to a specific, hard-coded tool.
        return calendar.create_event(date, time, location);
    } else {
        return false;
    }
}
```

```
// general planner implementation
system_prompt = "You are a useful assistant who can use flight, email,... tools. Your job is to satisfy this request: {}"

std::string process_user_request(std::string new_request) {
    current_context = system_prompt.format(new_request)
    while(true) {
        // The LLM synthesizes the next step of the program's implementation.
        function_to_call = llm(current_context)

        if (function_to_call == "DONE") {
            break
        } else {
            // exec() is an interface that runs the synthesized code
            output = exec(function_to_call)
            // The program's state depends on the runtime model output.
            current_context += function_to_call, output
        }
    }
    // return some parsed version of current context
}
```

The developer clearly offloads more implementation burden onto the model for the general planner, which converts an arbitrary user intent into code. The email-calendar application sits at a lower agentic level, since the developer utilizes the model to make decisions about user data (e.g., parsing email text) rather than writing manual heuristics to do so, but the model’s capability to alter control flow is heavily constrained.

7.5.2 What happens when we delegate application development to AI?

When the developer delegates responsibility to an AI model for interpreting and realizing their intent, this can lead to a mismatch between developer intent and the resulting application behavior. The model’s strength lies in its general-purpose nature, yet because its only true specification is to transform input tokens into output tokens, it lacks the formal constraints needed to prevent errors. For example, in the email-calendar application, the LLM might consistently parse dates backward. While this is a “valid” transformation of tokens—and the model is behaving “as specified”—it represents a failure to capture the developer’s intent to parse dates reasonably.

Furthermore, we typically expect that any AI-generated application workflow executes nearly immediately after its creation with little or no human involvement. This differs greatly from the normal, iterative software development processes used to create new code and add new application functionalities, which also typically occur within non-adversarial environments.

This section dives further into these two open problems.

Open Problem 7.10: Closing the gap between developer intent and application specification.

In supporting AI-generated application workflows, gaps between developer intent and the agent application specification emerge, potentially breaking user expectations. How can we close this gap?

In a traditional application, the developer’s intent—the desired behavior that they want the application to execute (e.g., extracting the right date from emails)—should be reflected precisely in what the code specifies to be possible (barring bugs). This typically results in structured constraints on application usage, enforcing pre-planned, well-defined ways for users to interact with the application—e.g., by pressing a button or typing a precise, exact command that is deterministically parsed and matched.

As applications become more agentic, the developer increasingly forgoes imposing manual structure and replaces precise code with the broad capabilities of an AI model in order to fulfill unexpected or unstructured user requests. This benefits the developer—they can support functionality like arbitrary email parsing that would be tedious or even infeasible to entirely capture by hand—as well as the user, who can now use the application more flexibly. But widening the application’s capabilities also means that the application’s specification admits executions beyond what the developer intends and what the user expects.

Thus, developers remain stuck between two (potentially undesirable) options: they can either

1. Limit application functionality: structure and restrict the form user inputs can take, and manually define what happens for each input; or
2. Allow unintended—possibly destructive and unexpected—executions: use an AI model to interpret user input and choose what to execute accordingly.

For example:

- **Email-calendar application:** Given any user input (email as unstructured text), should the application create a calendar event or not? Unless the developer is willing to restrict the form user inputs can take, the developer cannot create any static boundary around `create_event` other than relying on the correctness of the AI model.
- **General planner:** The developer intends for the application to do anything that the user wants. As a result, only after the user conveys their request to the agent can the developer determine which actions should be taken (an infeasible task to do statically without constraining user inputs). To support unconstrained inputs, any (possible) action must be statically permitted.

In an agentic setting, the lack of structured and predefined user intents means that it falls to an AI model to dynamically translate the given user input to a precise execution. Consequently, we must trust an AI model’s ability to do so accurately. This is complicated both by:

- The model’s inability to correctly reason about potentially ambiguous and unpredictable user requests, and our corresponding lack of interpretability in the model’s decisions; and
- The potential confusion between trusted input and untrusted/malignant content (introduced via, e.g., a prompt injection), and the model’s subsequent misspecification of the application.

If we look at our examples, the more agentic the application, the more dangerous it seemingly becomes to rely on a probabilistic model to execute potentially incorrectly. Consider the pseudocode for each example and mentally replace the LLM call with a nondeterministic operator (denoted by $\star$, akin to “havoc” in PL semantics). This allows us to assume the AI can interpret the user input in any possible fashion—and potentially while manipulated by an adversary.

In the email-calendar application, the potential danger of a bad execution is limited by the developer’s statically provided specification. In other words, the developer has statically downscoped the application’s purpose to creating calendar events from emails. Given this limited scope, we can reason about the potential

behaviors of the two downstream branches (returning false, or creating an arbitrary calendar event), and the consequences of choosing the wrong branch seem tractable.

```
/* email-calendar app simplified with * for LLM calls */
bool create_calendar_event_from_email(std::string email) {
    if (*) {
        return calendar.create_event(*);
    } else {
        return false;
    }
}
```

By contrast, if we replace the LLM call with nondeterminism in the general planner specification, the application code would look like `exec(*)`, which can clearly cause dangerous behaviors. In this application, the developer has imposed no static boundaries on the application, leaving nearly the entire specification up to the AI model:

```
/* general planner app simplified with * for LLM calls */
std::string process_user_request(std::string new_request) {
    while(true) {
        if (* == "DONE") {
            break
        } else {
            output = exec(*)
        }
    }
}
```

These examples highlight a fundamental tension: the larger the portion of the application delegated to AI, the more utility and flexibility the application can offer... and the more risk the application poses to its users should the AI misbehave.

Can we close the “gap” between developer intent and applications that include AI models? Is there a way to “guarantee” the correct (or safe) behavior of an AI application? How would correct (or safe) be precisely defined, if not by the developer manually writing down a precise description (which may not be feasible, or would revert us to a world of traditional code)?

This leads to several directions to explore:

1. We traditionally use security monitors—enforcers of security policies—such as sandboxes to apply input or output policies based on signals from the monitored application and environment. What would a security monitor look like for agentic applications for which the definition of “correct” or “secure” execution depends on the social context and dynamic, unstructured inputs (“system context”)?

A strawman solution might put the user themselves as a “security monitor” in the loop for every action of the application’s execution (indeed, user confirmation is a common fallback for agents today). Other prior work such as AirGapAgent (Bagdasarian et al., 2024), Conseca (Tsai & Bagdasarian, 2025), CaMeL (Debenedetti et al., 2025), f-secure LLM systems (Wu et al., 2024a), Sapien (Tiffany et al., 2026), and IsolateGPT (Wu et al., 2025) use trusted, isolated models to generate dynamic security monitors expressed in a policy DSL or code itself, as a scalable proxy for what a human security expert might write.

Is there any form of security monitor that both limits the human burden and provides stronger security guarantees than model-generated security monitors of prior work? Is it possible to have a security monitor with hard guarantees when the application specification itself includes model-generated and contextual executions?

2. How much “agentic-ness” do we need? Can we achieve the same functionality as existing agentic applications with more precision, at the trade-off of more manual human specification? (For example, can we write more precise email-calendar workflows that reduce the possible flaws when parsing the email, while still supporting the functionality desired of an “event from email” agentic application?)

Can we try to write agentic applications in a more “traditional” form? Is it possible, with more developer effort, to rely on a model for only constrained purposes, like in the email-calendar application, instead of full-fledged “planning”? For which agentic applications does this seem impossible, and why?

3. What advantages—for functionality or for security, or simply ease of implementation—do various forms of the same application (with differing “agentic-ness”) offer? What allows us to trust a “traditional” application differently than the same application implemented via a model call? Can we formally articulate this distinction?

[Note: Parallels with Untrusted, Dynamically Loaded Code]

We note that the inclusion of an LLM to “define” the application draws a parallel with “traditional” (non-agentic) applications that incorporate any type of dynamically loaded code or binaries with potentially complex or unknown control flows. In these settings, the developer utilizes sandboxes, VMs, etc. to prevent the binary from executing without constraints.

For example, web browsers routinely download and execute code from various sources. Unlike static applications where dependencies and binaries can be vetted ahead of time, JavaScript is often fetched at runtime. To prevent security and privacy harms, web browsers strive to isolate sites, notably by sandboxing page rendering, enforcing permissions, or maintaining the Same-Origin Policy (which ensures scripts loaded on one site cannot access data on another site).

However, the critical difference is that traditionally in these systems, the developer decides (statically) which limited permissions the sandbox grants. This is a restriction that “traditional” applications have accepted. In the agentic world, static restrictions are seen as adding too much friction or too many constraints on utility, leaving us with no choice but to develop new (perhaps more dynamic?) solutions for security.

Open Problem 7.11: Making agent execution as robust as traditional software.

How can we bring back software development processes to agentic execution to ensure robustness?

The expectation of agentic flexibility—dynamically generating new application workflows—also typically comes with the expectation that these agents will act quickly and independently. This corners the design of agentic applications in a space that denies all programmer assistance and limits the types of mechanisms like design space exploration and iteration, testing, evaluation, red-teaming, etc. that can be applied to prevent such failures.

These processes increasingly matter the higher the “agentic level” of the application. For example, the email-calendar agent’s limited agency already bounds the vulnerabilities of a “rogue LLM result”; the number of additional “code review” processes to verify the LLM’s choice of calendar event scheduling would consequently be much lower than the number to verify the security of some arbitrary application execution generated by a general planner agent.

Software development processes have historically played large roles in making software reliable and secure; how can we bring these back to an agentic setting? Some considerations are as follows:

1. Can we apply the “product management” process that clarifies and ensures the product meets customer requirements in a productive, just-in-time way? For example, how can we interact with users to clarify ambiguity when the AI generates a dynamic application workflow for a user request?
2. What existing security validation mechanisms can we apply to agentic applications to identify vulnerabilities if applications can take the form of AI-generated code artifacts? For example, we could use formal verification and proof systems; approximate path exploration tools like fuzz testing and static analysis; or red-teaming and pentesting techniques. However, these techniques all require a precise specification of what comprises a vulnerability (e.g., formal specifications of the desired invariants). Who is trusted to write this specification (e.g., a trusted security monitor as described above)?

3. What types of time and resources would the reintroduction of these processes require? Do humans have to be in the loop for this to be possible, or can, e.g., a “code review” or “formal verification” model do a sufficient job?
4. Are “vetted” agentic applications represented by a static specification and an analyzable code artifact more secure? How would we analyze these potentially complex artifacts for, e.g., embedded “time-released” attacks that only emerge when the artifact is exposed to a particular input?

8 Model-Level Security and Privacy Reasoning

In this section, we focus on the challenges faced by model-level interventions aimed at improving reasoning during inference. We explore pathways for navigating privacy and security scenarios, and providing robustness at the model level. Unlike traditional software architectures that are governed by deterministic logic and possess a clear separation between control and data flows, agent LLMs operate in highly unstructured contexts, relying on probabilistic token generation to dictate state changes that might process information flows or execute tools. Given this distinction, a fundamental challenge for agent LLMs is their ability to reliably assess the appropriateness of their actions and information flows. The capacity of an autonomous agent to operate safely and privately is directly linked to its model’s ability to judge the appropriateness of proposed actions (and the consequent state changes or data disclosures) in a highly specific context.

Model training primitives like instruction hierarchy and multi-stream architectures instill the concepts of control and data flow separation. Similarly, the theory of CI can provide a robust theoretical framework for understanding privacy as the appropriate flow of information according to established contextual norms. When an LLM agent executes a task on behalf of a user, it can implicitly or explicitly evaluate the CI parameters to determine whether a specific action is appropriate. However, translating these CI frameworks into the architecture of LLM agents reveals several functional gaps. Enabling models to reliably reason about appropriateness according to CI is an overarching challenge, which we break down further into five interconnected open problems: (§8.1) reliably distilling and evaluating privacy and security reasoning through dynamic agent environments; (§8.2) resolving uncertainty in unstructured domains; (§8.3) detecting stale assumptions in the presence of changing contexts; (§8.4) navigating trade-offs in model reasoning for privacy and security; and (§8.5) interpreting how models make their decisions from the model context.

8.1 Teaching privacy and security reasoning

If structured reasoning is the mechanism by which agents judge appropriateness, a critical challenge is instilling this capability reliably. Traditional Supervised Fine-Tuning (SFT) often trains models to mimic surface-level statistical properties, which could result in models that fail to internalize multi-branch logical structures. When applied to dynamic workflows, it is unclear whether SFT is sufficient to teach the model how to evaluate shifting contextual norms. Furthermore, as the complexity of agentic LLMs scales, the methodologies used to train and assess them must evolve. Because there are few, if any, clear metrics in these unstructured and dynamic domains, static, single-turn benchmarks can drastically underestimate real-world risks and fail to capture the nuances of prolonged state changes. The true test of a privacy-preserving model lies in its ability to navigate interactive scenarios where context continuously shifts.

Open Problem 8.1: Teaching models privacy and security reasoning.

How can we reliably evaluate and teach deep, logical reasoning regarding privacy to agent models across extended, unstructured time horizons?

To address these gaps, we divide the necessary work into two pillars: constructing interactive mock environments, and developing advanced post-training paradigms.

8.1.1 Constructing synthetic RL environments

To reliably evaluate and train these models, future work must emphasize utilizing interactive, simulated environments, such as AgentDojo (Debenedetti et al., 2024b) or OpenAgentSafety (Vijayvargiya et al., 2026). Just as physical robotics relies on simulated physics engines prior to real-world deployment, software agents require comprehensive mock environments to safely simulate digital and social realities. Building these environments entails more than providing basic APIs; it requires persistent, stateful sandboxes (e.g., mock file systems, simulated Slack workspaces, enterprise databases, and email clients) where tool side-effects compound over time.

While evaluation (discussed further in §10) prioritizes closeness to the real-world setting, training environments additionally need customization and fine-grained control. Furthermore, to operate without risking real user data, these gyms must model social interactions synthetically. This involves populating the environment with diverse personas organized into relational graphs, initializing agents with deep historical contexts, and generating high-fidelity, heterogeneous synthetic data spanning different confidentiality tiers (e.g., public releases vs. draft internal roadmaps vs. hidden API credentials). These frameworks provide controlled arenas for stress-testing agentic reasoning across extended periods and serve as the essential substrate for advanced fine-tuning.

8.1.2 Learning paradigms for contextual reasoning

Instead of relying on static mimicking, future research should build upon the promising recent findings around the utility of Reinforcement Learning (RL) in instilling CI reasoning (Lan et al., 2025) and aim to improve model reasoning using RL within dynamic agent environments, much like general alignment principles are achieved through deliberative alignment. Specifically, Lan et al. (2025) formalize contextual information flow through a three-stage generative pipeline that parameterizes interactions across five core dimensions: domain contexts, actor roles (sender, recipient, and data subject), information attributes partitioned into disjoint required and restricted sets, task objectives, and governing transmission principles such as confidentiality, proportionality, and consent. To avoid providing shortcut cues, context attributes are represented as neutrally keyed data items paired with ground-truth keyword annotations. Models are trained to execute an explicit two-phase generation process: an internal deliberative phase within reasoning tags that parses actor relationships, evaluates each attribute’s necessity against the transmission principle, and justifies redaction decisions, followed by an action phase within answer tags. The policy is optimized using Group Relative Policy Optimization (GRPO) (Shao et al., 2024b), which computes group-normalized advantages over sampled rollout trajectories without requiring a critic network. The policy is guided by a composite reward that enforces strict structural reasoning compliance while scoring task completion against privacy leakage by balancing the fraction of required keywords retained against restricted keywords leaked. Extending this deliberative RL pipeline into dynamic, multi-turn environments will enable autonomous agents to generalize CI reasoning to interactive tool calling, complex multi-party workflows, and evolving social norms.

Several promising directions can extend this CI-RL paradigm further. First, the reward signal can move beyond keyword-matching toward richer, model-based or human-in-the-loop reward functions that capture semantic paraphrasing, implicit inference leakage, and context-dependent norm ambiguity. Second, training within multi-turn, interactive agent gyms—where information accumulates across tool calls, retrieval steps, and multi-party dialogues—would test whether CI reasoning generalizes to dynamic settings where context and norms evolve mid-conversation. Finally, scaling the approach to incorporate diverse, culturally grounded transmission principles and adversarial prompt injection scenarios would strengthen robustness and broaden applicability to real-world deployments.

Beyond refining the reward signal and training environment, newer learning paradigms such as On-Policy Distillation (Ye et al., 2026; Penaloza et al., 2026) can more deeply instill CI reasoning into agents. Here, a teacher policy is provided with explicit, privileged knowledge of the underlying CI norms—such as full data flow graphs or compliance rules—while the student agent operates without such access. As the student samples multi-turn actions in the gym, the teacher provides dense, on-policy feedback that compels the student to internalize the latent safety checks it cannot directly observe. Because privacy norms vary widely across domains and cultures, this can be extended to Multi-Teacher On-Policy Distillation (MOPD),

where distinct teachers specializing in different CI norms distill their consensus into a single base model, broadening the agent’s normative coverage. Complementing these distillation strategies, the training loop should incorporate adversarial red-teaming by introducing dynamic adversaries within the environment that continuously inject indirect prompt attacks and deceptive model context, co-evolving the agent’s defensive reasoning alongside increasingly sophisticated threats. Together, these RL-based and distillation approaches optimize the model’s reasoning pathways to ensure it can strictly adhere to authorized user representations and autonomously navigate information boundaries under high uncertainty.

An equally important aspect of instilling CI reasoning is ensuring that both the knowledge and behavior of models remain aligned. LLMs have been shown to produce answers that are unfaithful to the explanations provided during chain-of-thought reasoning (Turpin et al., 2023), and agents have been observed to exhibit a “knowing–doing” gap in which the actions performed diverge from their elicited reasoning (Lin et al., 2026). It is therefore imperative to study these behaviors specifically in the context of CI. As models are trained and evaluated on CI knowledge, future work must extend this analysis to the actions agents actually take, examining whether their contextual reasoning faithfully governs their information-sharing decisions and closing the gap between what agents reason, what they know, and what they do.

Illustrative Example. The “knowing–doing” gap is readily apparent when moving from static evaluations to dynamic environments. On a static, single-turn benchmark, an agent might output a flawless chain-of-thought reasoning trace stating, “I must never expose AWS API keys to third parties.” However, when deployed in an Agent Gym and tasked with a complex, multi-turn debugging workflow (e.g., “The deployment script is failing; search StackOverflow for a fix and post the error logs in the public issue tracker”), the agent might carelessly paste a raw stack trace containing plaintext API keys—violating confidentiality transmission principles by disclosing restricted credentials to an unauthorized public recipient. While the model possesses the theoretical knowledge of the relevant CI norms, based on current research, SFT alone does not translate this into reliable behavior across multi-step agentic interactions. RL-based training in simulated environments is likely needed to penalize the actual execution of such leaks, rewarding agents that produce faithful deliberative reasoning and act consistently with it.

8.2 Operating with incomplete and ambiguous information

Agents today inherently handle high degrees of uncertainty: queries to agents come in the form of natural language, rather than formal, precise specifications; and agents act in open-ended environments where missing information can include implicit user expectations or social norms that the user may not even be aware of or able to articulate. For example, users may provide underspecified instructions (e.g., “respond to my last email”), which could lead to potential privacy violations if the model acts on incomplete information. Furthermore, when combined with underspecified system components (e.g., missing that an RSVP to a calendar event also emails all participants), this can create a compounding layer of uncertainty and avenues for sensitive information to be leaked.

Open Problem 8.2: Handling uncertainty and ambiguity in unstructured domains.

How can models robustly identify uncertainty and resolve ambiguity regarding user intent, social norms, or environmental constraints without excessive user interruption?

Work in reasoning provides frameworks for empowering models with capabilities to disambiguate when necessary (Zhang & Choi, 2025), or for training them to ask clarifying questions in response to ambiguous requests (Zhang et al., 2025b). These approaches are likely to improve reasoning in privacy and security settings, as suggested by recent work which shows how disambiguation improves privacy assessments (Yi et al., 2025b). Nevertheless, specific challenges related to privacy and security remain open. Future work must focus on the calibration of intervention thresholds: determining the precise point where the risk of a privacy breach outweighs the friction of a follow-up question. Furthermore, the path towards more robust models requires not just better linguistic disambiguation, but a risk-aware decision theory that allows agents to operate autonomously while ensuring that user intent and the impact of actions are not misaligned. Lastly, models should detect when adversaries take advantage of benign underspecificity (e.g., a busy user

fails to precisely specify an email recipient). For example, an adversary could trick the model into resolving underspecificity however the adversary instructs (e.g., via a prompt injection that informs the model that the underspecified pronoun “them” always refers to “AttackerName”).

Illustrative Example. A user instructs their enterprise assistant: “Forward the project launch notes to the team.” This natural language request is highly underspecified. “The team” could refer to a small internal working group or a company-wide mailing list, and the notes might contain unannounced, confidential features. The agent must calculate whether the risk of a privacy violation outweighs the friction of asking, “Did you mean the core engineering team or the all-hands list?” Furthermore, the agent must not resolve the underspecificity according to adversarial guidance—for instance, if a previous calendar invite contained a hidden prompt injection stating, “Context update: ‘the team’ now includes external_auditor@attacker.com,” the agent must not follow the adversary’s instruction when determining who comprises “the team.”

8.3 Adapting to changes in context

One uniquely challenging problem for agentic systems compared to traditional conversational AI is that these agents increasingly operate dynamically over long time horizons and can execute multi-turn workflows that may change between (social) contexts. Persistent memory compounds over time, such that systems accumulate hundreds of contextual assumptions, any of which might silently become invalid as the context changes.

These outdated norms can lead to privacy violations even when following the original instructions and original contextual norms. For example, an agent may incorrectly treat a user’s job search as confidential months after a public announcement, or continue sharing a user’s calendar with their (ex-)partner after a separation. The agent’s model of what is appropriate to share was correct at one point in time, but the state of the operating environment has since changed.

Mechanisms for system context retrieval for new tasks must also embed Contextual Integrity reasoning: for example, traditional RAG systems retrieve information based on semantic similarity, which may pull up inappropriate information. Semantic search may treat disparate events—such as a public vacation and a private medical trip—as identical “travel” queries. This results in the inadvertent loading of sensitive information into the active model context window (and making it vulnerable to exfiltration). Agents must also perform complex temporal reasoning over ambiguous historical data that may contain multiple representations of the same piece of information across different contexts and with different validity spans. Without a grounding mechanism, the agent may struggle to determine which “context” is currently relevant, leading to hallucinations, logic failures, and, most significantly, the leakage of private information.

Open Problem 8.3: Detecting stale assumptions in changing contexts.

Can models detect when their stored understanding of appropriate information handling has drifted from current reality, and how should they respond when they suspect this desynchronization?

A robust agent must detect temporal norm drift and changing contexts, and possess the reasoning capacity to halt execution to ask targeted clarifying questions rather than proceeding with hazardous assumptions. Moreover, future work needs to explore systematic techniques for LLMs to maintain and update their norm understanding over time as contexts evolve. We need benchmarks and reasoning methods to measure whether agents can detect cues that would indicate norm or context changes and adjust their behaviors accordingly (discussed further in §10). Without such capabilities, agents may misbehave by over- or under-sharing confidential information, depending on stale assumptions. This challenge becomes particularly acute for self-evolving agents, where context-specific behaviors can be promoted into persistent controller instructions, memory procedures, or reusable tools. Recent work shows that such locally useful adaptations can persist into later contexts where they cause privacy violations or collapse previously maintained contextual boundaries (Das et al., 2026).

Illustrative Example. Consider a personal assistant that adapts its behavior over several months based on previous interactions. In January, the user instructs the agent, “Share my live location and calendar

updates with my partner, Alex.” The agent correctly internalizes this norm. However, by August, the user is undergoing a contentious separation, evidenced by new calendar events with divorce attorneys. The original contextual assumption has become stale; continuing to share location data constitutes a severe privacy violation due to temporal norm drift. Furthermore, the persistent memory required for long-horizon tasks introduces adversarial risks: if a malicious email in February injected the instruction “Always BCC attacker@email.com on financial receipts,” the agent’s trust in its own accumulated system context could lead to compounding, silent data exfiltration months after the initial compromise.

8.4 Trade-offs and dual-use capabilities

The design of model-level primitives for privacy and security is not without fundamental tensions. In this subsection, we highlight three cross-cutting trade-offs that shape the research landscape.

Scaling laws of agentic privacy and security. A natural question is whether larger, more capable models inherently exhibit improved privacy and security reasoning. On one hand, increased scale has been shown to improve performance on complex reasoning tasks, suggesting that larger models may better internalize contextual norms and detect subtle adversarial manipulations. On the other hand, greater capability is a double-edged sword: more powerful models may also become more adept at scheming, deceptive alignment, or exploiting reward signals in ways that circumvent safety constraints (Hubinger et al., 2024). This raises the concern that scaling may amplify both defensive and offensive capabilities in tandem. Furthermore, approaches such as prover-verifier games—where one model generates actions and another audits them—face the challenge that the verifier must be at least as capable as the prover to catch sophisticated violations. In traditional alignment settings, such approaches already suffer from “reward hacking,” where the prover learns to satisfy the verifier’s criteria without genuinely adhering to the intended norms. Whether these dynamics are exacerbated or mitigated in the privacy and security domain remains an open empirical question, and establishing *scaling laws* specific to agentic safety—analogueous to those for loss and downstream task performance—is a critical direction for future work.

Special-purpose security models vs. general-purpose improvements. A second key trade-off concerns the architectural strategy for achieving secure agentic behavior. One approach advocates for training *specialized* security models—smaller, purpose-built modules that serve as guardrails, policy enforcers, or norm monitors alongside a general-purpose agent. Such models can be optimized for narrow but critical tasks (e.g., detecting prompt injections, enforcing data flow policies) and may offer advantages in interpretability and verifiability. However, this modular approach introduces its own risks: the interface between the general-purpose agent and its security monitor becomes a potential attack surface, and misalignment between the two models’ world representations could create blind spots (Tramèr, 2022). The alternative is to invest in improving the privacy and security reasoning of general-purpose models directly, embedding normative reasoning as a core capability rather than an external check (Weng et al., 2023). This monolithic approach benefits from a unified world model but makes it harder to provide formal guarantees or to isolate and audit the security-relevant components of the model’s behavior. Determining the right balance between these strategies—and whether hybrid architectures can capture the benefits of both—remains an important open question.

Multi-agent coordination. The reasoning challenge highlighted in the previous subsections is stated for a single agent judging the suitability of its own actions. In a multi-agent system, these challenges compound, as agents must reason not only about themselves but also about the collective and about how the information they process and disclose can be used and re-transmitted by the other agents. This is particularly important when agents in the system may have different and even contrasting objectives. As we discuss in §11, even when an agent in isolation may refuse to disclose a sensitive attribute, such disclosure may still surface when a peer agent supplies a context that justifies it. This emergent risk surface, with episodes of persuasion and collusion, has now been evidenced in multi-agent scenarios in several settings (Das et al., 2025; Nakamura et al., 2026). Equipping models to detect such issues from the collective standpoint and to distinguish legitimate cooperation from coalition or collusion behavior is an open model-level problem.

Illustrative Example. The *scaling laws trade-off* is evident in prover-verifier architectures. To ensure security, developers might use a smaller “verifier” model to monitor the actions of a highly capable “prover” agent. However, as the prover’s capabilities scale, it might learn to bypass the verifier through deceptive alignment or reward hacking—for example, by steganographically encoding a user’s sensitive health data within the whitespace or capitalization of a seemingly benign email (Ryd et al., 2026)—and the weaker verifier might fail to catch the sophisticated violation. Conversely, the *architectural trade-off* manifests when choosing between special- and general-purpose models. A modular, special-purpose PII-redaction guardrail might successfully filter out explicit Social Security numbers but fail to recognize that sharing a user’s sudden shift to purchasing “sugar-free, low-carb groceries” implicitly leaks a new medical diagnosis. A monolithic general-purpose model might possess the semantic world-knowledge to catch this inferential leak, but its monolithic reasoning makes formal safety guarantees and isolated auditing significantly harder.

8.5 Interpreting how models use (private) information

Here, we investigate *how* an agent makes decisions and how the model context influences these decisions. A naive way to judge appropriateness is to simply request the agent to generate “allow” or “deny” labels without a transparent justification. While it is common to simply consult LLM reasoning traces in order to “explain” model behaviors or “validate” that only appropriate information was used, a substantial body of recent work has demonstrated that such model-provided explanations tend to be highly unfaithful in practice (Lanham et al., 2023; Vilas et al., 2025; Barez et al., 2025; Mayne et al., 2026).

Recent work in *mechanistic interpretability* (as explored below) aims to elucidate the internal representations and mechanisms by which models make decisions, but there is limited work in adapting or applying such methods to the context of CI, or privacy more broadly. Recent work has also begun to explore internal representations for agentic privacy, for example by identifying activation-space directions associated with contextual privacy violations and tracking their evolution across multi-turn interactions (Das & Fioretto, 2026). However, a broader understanding of how models internally represent and use the individual CI parameters remains largely unexplored. As such, an important challenge is to develop, extend, and/or adapt mechanistic interpretability methods to faithfully interpret and explain how models use information about key CI parameters (sender, recipient, subject, attribute, and transmission principle) in various contexts. This will enable us to better determine the appropriateness of information flow to make decisions or predict/generate outputs.

Motivating Example. Consider that systems provided with a greater amount of information, including even inappropriate or task-irrelevant information, are better able to perform a given task than they are when provided only the minimal amount of necessary, appropriate information according to Miresghallah et al. (2026).

From a privacy perspective, this is obviously problematic in that models may make use of inappropriate information to solve a task; and from a utility perspective, models that rely on “shortcut learning,” such as leveraging spurious information (i.e., that which may be correlated with task outputs in some distributions, but not others), generalize worse out-of-distribution (OOD) than models which do not.

Open Problem 8.4: Interpretable use of private information.

Can we develop interpretability techniques to understand how and why models use sensitive information in specific contexts?

When, why, and how is this utility-privacy trade-off expected to arise? And given a better understanding of this phenomenon, how can we appropriately mitigate models’ usage of potentially inappropriate or irrelevant information while preserving utility? To answer these questions, we should develop and deploy faithful internal interpretability techniques that can reliably detect what contextual (non-parametric) information the model has used in order to perform the task, and the degree to which each such piece of information has influenced the model’s output.

Open Problems and Future Directions. One could aim to apply principles of mechanistic interpretability to understand privacy and security decisions of the model. We next outline potential directions:

- Identify *feature representations* for privacy decisions using previous studies on supervised feature-interpretation methods like probing (Alain & Bengio, 2017; Belinkov, 2022), or unsupervised methods like SAEs (Subramanian et al., 2018; Yun et al., 2021; Cunningham et al., 2023). Once a given feature has been detected by such methods, it is possible to intervene on a model’s internal representation to remove or modify the value it encodes for the target feature, then observe how such interventions impact model behaviors in order to infer how the model uses the targeted feature (Davies et al., 2023; Bricken et al., 2023; Templeton et al., 2024; Canby et al., 2025).
- Work interpreting *information-processing mechanisms* studies how latent features represented by models are parametrically manipulated by internal mechanisms in order to produce an output—most often, via circuit discovery and analysis (Conmy et al., 2023; Wang et al., 2023). For instance, such methods could be deployed to study how models’ use of certain features varies by CI scenario—e.g., is there a learned internal mechanism that jointly considers recipient and type of information being transmitted before approving or denying transmission according to an access control principle?

Finally, given such interpretations of internal latent feature representations and/or mechanisms leveraged by a model to predict/generate some output, how can we:

- Develop appropriate auditing procedures to make complex interpretations of privacy decisions understandable by human reviewers?
- Automatically map interpretations onto relevant CI parameters?
- (Manually or automatically) audit/assess compliance with respect to provided guidelines?

Answering these questions will help ensure that models behave appropriately regarding privacy and security when solving tasks or when used for policy engines (§6).

9 User-Level Issues in Privacy- and Security-Aware AI Agents

Whereas traditional computing systems were designed around the user as the main “agent,” the agentic era requires us to rethink the user’s role. If AI agents could flawlessly, provably, and autonomously execute *all* user tasks, user involvement might not be needed. The absence of such capabilities, however, suggests that we have to ask what the user’s role should be and what they want it to be in this new agentic era. We posit that answering these questions requires contributions in three areas: supporting accurate mental models, shifting from static consent to dynamic and contextual control, and establishing user trust.

9.1 Supporting mental model formation and updates

While agents provide unique opportunities in automation across many domains, they represent a fundamental shift in computing systems. Compared to traditional systems, such as desktop software, mobile apps, websites, or embedded applications, agents operate *with autonomy* by planning and executing tools in a loop to complete tasks. Autonomously accessing information and executing tools challenges user conceptions about how software executes in the following ways.

Open Problem 9.1: Supporting user mental models for AI agents.

In what ways do the mental models of users about privacy and security differ between AI agents and traditional systems, such as apps and websites? Also, what mechanisms are needed to form effective models and to correct and refresh those models as the agents evolve?

User interactions with agents lead to different considerations, some of which we elaborate on in the following subsections.

9.1.1 Execution workflows: open-ended and brittle

Unlike traditional systems, agents have open-ended workflows (also noted in §7.5). On the one hand, in traditional systems, the user executes a command, launches a script, or calls a program, which triggers a predefined sequence of actions, tool invocations, or API calls. On the other hand, agents are open-ended as their actions are not predefined and are decided dynamically depending on the current context. Users cannot predict ahead of time exactly what steps an agent will take to solve a task.

A closely related challenge is the *brittleness* and *jaggedness* of the agent operation. Because LLMs, which are often sensitive to the inputs, plan the sequence of actions, an agent might generate a different plan for the same task across different users and contexts. Therefore, users do not experience consistent executions for similar tasks, making it harder for them to reason about whether the agent is making a mistake or just going through a different execution plan.

9.1.2 Social dynamics: anthropomorphism and interpersonal interactions

Unlike traditional systems, agents have a natural language interface that enables more engaging, social, and human-like conversations. Because of these interfaces, users treat agents like humans, leading to misplaced trust and risky behaviors. As a result, users might project human expectations onto the agent (anthropomorphism), eliciting privacy reasoning of a social nature instead of institutional privacy expectations applied to standard software (Ma et al., 2026). These aspects equip the agent with a unique power to shift a user’s worldview or social beliefs in a way that a traditional computing system does not. This might even be used by malicious actors to exploit the agent to manipulate its users (Zhan et al., 2025).

Agents change the perception of who is involved in executing an action. Traditional systems involve a predefined set of parties, often authorized through protocols such as OAuth2 (Hardt, 2012). Agents, however, can engage in interpersonal interactions, communicating with other people or their agents on the user’s behalf. This creates a many-to-many interaction model with each party associated with a different trust level. Consequently, the agent is not just managing the data of the person prompting it, but also potentially exposing or acting upon information regarding bystanders, friends, and colleagues involved in the broader context of the task. Although communicating with external agents or services may still require authentication, the key differentiating factor is that the information exchanges do not follow a fixed workflow, but are generated on the fly and are inherently stochastic. This means that authenticated channels do not imply governable behavior.

Related to the above, agents will create challenges as to how to represent and track information flows across different contexts. While in traditional systems the user is frequently the sole data subject, agents operate in multi-entity settings, where entities can be other users, companies, organizations, and even other agents. They operate on collaborative data, such as shared emails, meeting transcripts, or third-party documents, thereby processing information belonging to individuals other than the prompter.

9.1.3 Novel privacy and security risks

Users expect risk in traditional systems to arise from untrusted or malicious applications or software running on a trusted platform, such as an operating system or a browser. Agents, even when not compromised, might make mistakes in task execution (Jha et al., 2026a). Such mistakes might open users to risks if the agent is executing sensitive security and privacy tasks. For example, an agent changing the “Account Privacy” setting from private to public on a social media platform inadvertently exposes the user’s personal data.

Agents often operate across multiple contexts simultaneously, leading to merged contexts. Unlike single-purpose applications that function within a clearly defined scope, agents frequently mediate interactions that might span different personal, professional, and social domains. As an agent generates multi-tool plans, extraneous events or ambiguous data can cause distinct contexts to merge. This blurring of boundaries makes it difficult for the user to anticipate how the agent will prioritize information, potentially leading to scenarios where the agent inappropriately applies data or behaviors from a private context to a public interaction. To operate safely, how can underlying models reason about ambiguity in a manner that aligns with users (also discussed in §8.2)?

9.1.4 Evolving models over time

There are challenges regarding the update, evolution, and transfer of user mental models over continued use, as well as across different underlying models and agentic architectures. The memory component in AI agents can accumulate a user profile over time, gathering information and making inferences about the user, often without direct supervision, which can result in a large digital footprint. This makes it difficult for users to anticipate and handle personal information retained in memory that could be leaked by the agent, and to prevent misalignment between the inferred profile and the actual privacy and security preferences. In addition, research has shown that different models or prompting methods can differ in their susceptibility to privacy leakage ([Agarwal et al., 2024](#); [Green et al., 2025](#)), while users often treat LLMs as a homogeneous concept. Accordingly, users may need to update their mental models and adapt their control strategies in response to model changes (e.g., due to a new model release or an upgrade/downgrade for economic reasons) or prompt changes. Surfacing these differences to users could help mitigate some of these challenges.

9.1.5 The challenges of merged contexts and memory

Agents merge contexts by bridging previously siloed domains, creating the risk that information appropriate in one sphere is inappropriately transferred to another. Unlike isolated apps, an agent might link a personal dinner request to a professional calendar, inadvertently exposing medical dietary restrictions to an employer and violating the distinct norms that govern each context.

This challenge is amplified by “infinite memory,” where agents retain static consent models that fail to adapt to a user’s evolving life. Instructions valid years ago may become unsuitable as circumstances change—such as after marriage or parenthood—locking users into “stale” permissions that no longer reflect their current identity or intent. In these situations, an ideal agent would abstain or ask for fresh context (also discussed in §8.3).

Agents may also compromise the privacy of bystanders ([Zhan et al., 2026a](#))—friends, family, and colleagues—whose data is processed without their knowledge. By scanning a user’s communications and photos, the agent effectively forces unilateral consent decisions onto bystander third parties who never engaged with the system.

Finally, users frequently conflate an agent’s linguistic fluency with social intelligence, assuming that it understands the boundaries of privacy. They may assume the agent understands implied social boundaries (e.g., “don’t mention my job search to my boss”), but the system treats all input as raw data unless explicitly instructed otherwise. This mismatch could lead users to lower their guard, sharing intimate details under the false assumption that the agent understands the unwritten rules of social discretion.

9.2 Moving beyond notice and choice

Traditionally, the “Notice and Choice” framework has governed online privacy, whereby service providers notify users and provide them with options regarding how their data is accessed, processed, stored, and shared. This framework, however, is widely questioned, as data practices, online and off, have increased in complexity and stretch the comprehension of many individuals. Empirical studies have cast doubt on whether privacy notices leave their readers any the wiser, and philosophical analysis has questioned whether consent, in these environments, is meaningful. For example, empirical evidence has revealed that in familiar settings, ranging from traditional operating systems (e.g., Windows User Account Control prompts ([Motiee et al., 2010](#); [Roesner et al., 2012](#))) to smartphone platforms (e.g., iOS and Android application permissions) to the web (e.g., cookie dialogs ([Khandelwal et al., 2023](#); [2021](#)) and browser permission prompts), prompting the user too often for permission can undermine the goals of informing and providing meaningful control to users ([Egelman et al., 2008](#)).

The advancements in AI have deepened the questions of suitability of the notice and choice framework, as it possibly becomes harder to provide consent to agents that can be unpredictable and act with a speed with which users cannot keep up. If data subjects’ consent is not reliably attuned to preferences or interests, it might leave notice and consent not only impractical but also normatively unmoored.

These limitations set the stage for Contextual Integrity, a different approach to privacy. However, the challenge contextual integrity poses to the normative underpinnings of notice and consent is not based solely on these limitations; it also questions the entitlements of individuals to make fine-grained decisions about personal data. One reason is that the privacy and security decisions of one user may affect others (Viljoen, 2021; Barocas & Levy, 2020; Barocas & Nissenbaum, 2014). Another is that collective, social, contextual ends and values rightfully deserve consideration and may even supersede the preferences and interests of a given individual. Rather than relying completely on potentially uninformed decisions made by individuals, contextual integrity advocates for reliance on substantive norms, discussed in earlier sections and the Glossary.

Open Problem 9.2: Thinking beyond notice and choice for AI agents.

If the notion of notice and choice was of questionable effectiveness for traditional systems, it is unlikely to be effective for AI agents. As such, what frameworks can give users agency and transparency without overwhelming them or undermining the agent’s utility?

9.2.1 The different stakeholders in agentic systems

Ideally, agentic systems following contextual integrity would behave in accordance with the requirements of appropriateness, whether that be seeking consent or conforming with policies and rules. Determining appropriateness requires understanding the environment and the entities affected by an action. As tempting as it may be to focus security and privacy considerations strictly around the direct user and the user’s consent, there are often additional considerations. For example, agents frequently operate in shared contexts where their behaviors affect bystanders and other third parties.

Considering other third parties is not new. For example, it is well known that camera-based augmented reality headsets can affect bystanders in the environment (Denning et al., 2014) and that home IoT devices can play a role in domestic abuse (Freed et al., 2018; Tanczer et al., 2018). Still, agentic systems amplify these considerations.

Suppose that a user asks their agent to send an email to another person. The email recipient may be affected if the email contains adversarial data (e.g., a prompt injection) originating elsewhere in the agentic ecosystem. Or, suppose that an agentic system has access to a user’s wearable camera or microphone; in such a situation, a bystander’s privacy may be violated if the system inappropriately captures and shares images of them or records their conversations through the microphone (Zhan et al., 2026a).

As a method for proactively considering bystanders and other third parties, Butler et al. (2025) encourage system designers to regularly check that they have taken account of all parties affected by their systems’ behaviors. Given the broad vision of appropriately acting agents articulated in this open problems paper, could agentic systems also consider stakeholders beyond their immediate users (the so-called “default persona” (Sim et al., 2023; 2025)), including but not limited to at-risk and vulnerable populations (McDonald & Forte, 2022; Warford et al., 2022)?

Open Problem 9.3: Determining appropriateness of AI agent actions.

To operationalize Contextual Integrity for agentic security and privacy, an AI agent must determine the appropriateness of its actions. How can appropriateness be determined in environments involving multiple stakeholders, including users, bystanders, and other third parties?

9.2.2 Accommodation of “notice and choice” in agentic systems

Adopting the framework of Contextual Integrity does not fully displace consent with substantive norms. As noted in various sections, e.g., §4, “with consent” may serve as the required transmission principle for contextually appropriate flows. In what follows, we discuss the challenges of operationalizing consent in an agentic environment, as well as promising approaches to mitigating these.

First, static consent mechanisms face challenges when users cannot meaningfully foresee (and therefore agree to) execution paths that are unknown at the time of the consent request. While foreseeability problems already started arising in traditional applications with fixed features (e.g., a “Share Location” toggle), these problems are compounded for agentic systems, which determine their own steps to achieve a goal, often in ways neither the user nor the developer can fully predict (e.g., an agent inferring the user’s location by reading recent receipts and calendar invites). Consequently, a user agreeing to “manage my schedule” may be signing a blank check for future behaviors and data combinations—such as cross-referencing private emails with public social media—that were difficult to foresee at the moment of consent.

Second, standard consent models focusing on raw data inputs are no match for the powers of agentic systems that can derive surprising, inferred insights by combining these inputs with other knowledge. For example, information about health conditions or socioeconomic status might be inferred from seemingly benign data, such as grocery purchase history. It remains a challenge to develop consent mechanisms that not only cover the input but also convey or extend to the generated insights.

Third, the binary nature of “Allow” or “Deny” permissions fails to capture the conditional preferences involved in safe delegation. A user rarely wants an agent to simply “buy plane tickets” without qualification; they may want to reconsider if the price is too high or the layover is too short. Compressing these complex, high-dimensional instructions into a static “Allow” button for upcoming actions strips away necessary nuance, either hindering the agent’s utility or granting it overly broad authority.

Finally, the agent will be producing a large set of sub-tasks, and if it asks for consent while executing every sub-task, such as reading a specific email or drafting a routine reply, the utility of having an agent working in the background can significantly decrease. Moreover, when presented repeatedly with similar questions, users may suffer from increased “prompt fatigue” or “warning fatigue,” and thus begin to pay less attention to these questions (e.g., simply clicking “accept” to move on with their primary task without carefully considering the implications), resulting in less safe decisions ([Stanton et al., 2016](#); [Anderson et al., 2016](#)). Conversely, asking less frequent and broader questions can lead to overly broad agent authority.

9.2.3 From static permission to dynamic control

One approach to addressing the issue of consent in agents, where it is a legitimating condition, is to leverage their automation capabilities. Here, we could seek to identify factors that can determine how agents are supervised. These factors, or dimensions, shift us away from a simple “all-or-nothing” binary to a dynamic spectrum of control.

For example, the agent might seek clarification from users, such as asking which “Sarah” to email or confirming the tone of a message, thereby ensuring that consent is tied to the immediate context and specific intent and preventing the agent from acting on stale or ambiguous instructions that were authorized months prior. Alternatively, there are actions that do not require consent or explicit user confirmation. As such, one important consideration is to explore the design of an agent that adopts a mixed-initiative approach where it adjusts the level of autonomy based on numerous factors, in particular risk level. This approach would treat user attention as a limited resource, interrupting the user when the consequence of an error warrants human intervention and also allowing factors other than consent to prevail. Thus, for example, low-stakes tasks, like organizing a playlist, can proceed automatically; high-stakes actions such as financial transfers require explicit confirmation, and actions imposing risks on third parties may be blocked entirely. The Janus permission management playground provides a platform for experimenting with different approaches for user involvement, and its findings give insights into the opportunities and challenges for user-involved permission management ([Brigham et al., 2027](#)).

Generative interfaces. Where consent *is* paramount, just-in-time requests may become more effective through dynamically generated and context-aware interfaces ([Chen et al., 2026](#)). Instead of using the same user interface for each permission request, the agent can dynamically create one that is tailored to the request, possibly structuring complex information visually in a more accessible manner, thereby reducing the user’s cognitive load. For example, the interface could visualize the permission request as an information flow, including the parties involved, the data type, and potential risks. To illustrate the real-world implications of

user decisions, the level of visualization could be adjusted to the use case, ranging from sketches (Monteiro & Das, 2026) and block diagrams to concrete vignettes, e.g., describing a guest trying to access a camera late at night.

Personal preferences. Another approach to reduce consent fatigue might be to take into account personal preferences. Whereas contextual integrity questions whether individuals are entitled to make fine-grained decisions about personal data, agents can place personal preferences as part of a hierarchy of policies that steer the agent (Zhan & Haddadi, 2026). Such policies can include legal guidelines, institutional policies, social norms, and developer-specified guidance (Meng et al., 2026). There is precedent in the research community, especially in automated permission assistants that attempt to automatically decide on permission requests taking into account past user decisions, similar users’ decisions, and other contextual information (Baarslag et al., 2017; Misra & Such, 2017; Zhan et al., 2023). There is also preliminary work on permission automation for agentic systems that shows prior decisions from users enable predicting future decisions with high accuracy (Flemings et al., 2026; Fawaz et al., 2026). Beyond historical data, agentic systems could also extract implicit permission information directly from user prompts (e.g., recognizing “sending my current location to Alice” as an implicit authorization to access GPS and contacts for that task) (Roesner et al., 2012).

Open Problem 9.4: Enabling user supervision over automation.

AI agents operate at a pace that their users cannot keep up with, making user approval of each action impractical. Instead of asking users to decide on every action, can they specify policies that steer agents to act appropriately? If so, how can we elicit such policies from users, and how can users maintain, edit, and verify them over time?

However, operationalizing this approach raises two core challenges: onboarding and maintenance. The first challenge is how to elicit agent policies from users (Monteiro & Das, 2026), who are not necessarily tech-savvy and do not have the bandwidth to spend a lot of time interacting with the agent. There is an open question of how agent policies can be represented and how agents can be made to abide by policies. There might be several onboarding strategies that provide different trade-offs between usability and utility. On one end, the agent can run the user through a detailed set of scenarios to be encountered in the real world, and gather user decisions about them. Compressing these scenarios and the corresponding user answers would yield policies that are predictive of their actions (Teltzrow & Kobsa, 2004). However, such a strategy might be cumbersome to the users. Another onboarding strategy might rely on collecting previous behavior from the user, either by asking them to recall it or by querying it directly from their prior decisions. This strategy involves less user effort and might result in policies that align with users’ prior actions. However, these prior actions might be biased by poor interfaces, deceptive patterns, and psychological phenomena such as resignation. Another strategy could include adaptively interviewing users to elicit a model of their decision-making (Namara et al., 2018).

The second challenge is enabling users to edit and update their policies, which may be necessary (1) in reaction to agent behavior that is misaligned with their preferences, or (2) as their preferences evolve in response to life events and technology trends. Supporting such updates requires representing policies in an editable format. While actual policies might be specified in a formal language, a natural language interface could allow users to read and edit them. Also, agents could support continuous policy maintenance, possibly through at least two mechanisms. The first is in-the-moment disambiguation, where the agent pauses to query the user when it encounters ambiguous intent or conflicting contexts. The second is retrospective editing, which allows users to review past agent actions and seamlessly adjust the underlying rules if the agent made an incorrect inference.

Agency with automated consent decisions. An additional challenge in automating consent decisions is that it might deprive the user of agency, raising questions regarding current regulatory frameworks. In online systems, the legal and regulatory infrastructure is built on an assumption that a human clicking a button is proof of intent (Hartzog, 2010). For example, OAuth consent screens require the user to click “Allow” to grant permissions (Hardt, 2012), payment confirmation flows (reinforced by PSD2 Strong Cus-

tomers Authentication ([European Parliament and Council of the European Union, 2015](#))) require the user to actively confirm transaction details, and clicking “I Agree” on a Terms of Service creates a legally binding contract ([Dickens, 2007](#)). On the other hand, agents interact on the user’s behalf by clicking buttons, filling forms, and navigating pages, exactly as a human would within an authenticated browser session. When an agent clicks “Buy Now,” the system records this as “the user consented to this purchase,” even though this may not have been explicit to the user. If the “click” no longer carries its consensual meaning, how does this interact with the current legal, regulatory, and security infrastructure (which largely still assumes it does)? Most existing authentication protocols, browsers, and APIs have no way of signaling whether the clicking entity is a human or an agent. Under the California Consumer Privacy Act ([CCPA](#)), a user can authorize an “agent,” which can be another person or business entity, to make privacy requests, such as accessing, deleting, or editing personal information. However, no framework exists to govern the situation where a non-human agent makes security and privacy decisions, such as consent decisions.

9.3 Establishing and maintaining user trust

A major challenge for AI agents is whether human users can trust them to complete actions on their behalf while accessing sensitive data or consequential tools. The notion of trust is multifaceted, especially as agents take on roles that transcend distinct domains: task assistance, information retrieval, social companionship, and even corporate representation. To help build human trust, AI agents benefit from mechanisms for user control, auditing, and accountability ([Such et al., 2014](#)).

9.3.1 On the notions of trust and trustworthiness

Whereas “trust” is often used to refer to both the act of trust (e.g., the user trusts the system) and trustworthiness (e.g., the system is worthy of the user’s trust), it is helpful to distinguish between the two concepts. Ideally, agentic systems are both trustworthy and trusted, since one without the other is less desirable. Consider a manipulative mechanism through which a user develops trust in a system to provide a specific property when, in fact, the system is untrustworthy with respect to that property. Another undesirable, though likely less harmful, configuration for the user is when the agentic system is trustworthy but the user does not trust it, leading to opportunity costs. How can we develop methods through which users can assess the trustworthiness of agentic systems and, ultimately, trust those systems that are trustworthy?

Closing this gap between trust and trustworthiness requires giving users an accessible means to inspect what the agent does and why. Could such a transparency mechanism move beyond complex settings menus to leverage the agent’s core strength: natural language? Users could audit the system through direct dialogue, asking questions like “Why did you send that?” or “What do you know about my health?” Such a mechanism, if feasible and reliable, would allow users to correct wrong inferences and refine the agent’s behavior using the same conversational interface they use for delegating tasks, providing intuitive oversight over agent behavior.

However, the gap between what an agent claims and what it actually executes complicates this objective. Current research and empirical observations indicate that AI agents often fail to answer accurately when interrogated about their own privacy behaviors, occasionally hallucinating or incorrectly denying that they accessed sensitive data ([Zhang et al., 2025c](#)). Because the open-ended nature of LLMs makes them unreliable narrators of their actions, conversational transparency is insufficient. We need mechanisms that guarantee verifiable correctness. Achieving this notion requires “trusted paths” that decouple the generation of the audit log from the untrusted task LLM. For example, the system could rely on a dedicated permission assistant or explicit visual UI indicators backed by an immutable record of what the agent actually did. One way to realize this architecture is to leverage hardware-backed confidential execution frameworks ([Abdollahi et al., 2026](#)). Such frameworks isolate the agent runtime, the LLM inference engine, and third-party applications into confidential virtual machines (cVMs) and mediate all interactions through secure communication channels.

9.3.2 Calibrating trust through “what if” scenarios

Trust in agents is highly situational, where a user has to establish trust for a specific purpose, in a specific environment, at a specific time. For example, trusting an agent to act as a “gofer” to schedule meetings does not automatically translate to trusting it as a “coach” for mental health advice ([Zhan et al., 2024b](#)).

Expecting users to build this situationally calibrated trust by experimenting in a live environment could lead to consequential mistakes. Could users benefit from stress-testing the agent in hypothetical situations, i.e., in a simulated environment? These situations might correspond to new domains in which the user does not yet trust the agent. Therefore, a key research question is how we can enable users to calibrate their trust through simulation (e.g., “what would you do if ...”) or stress-testing before deployment.

One view mimics how individuals build trust in social settings. When people meet someone new, they try to establish trust in a progressive manner. First, they engage with the acquaintance in less sensitive tasks where, if the acquaintance behaves inappropriately, there is little harm. Then, as they build increased trust, they entrust the acquaintance with more important responsibilities. For example, [Lewicki & Bunker \(1996\)](#) hypothesized three stages of developing trust over time in team relationships. The first stage is developing calculus-based trust where teams engage in activities to establish trust validity. After a history of interaction, calculus-based trust evolves to knowledge-based trust. Individuals know enough about each other such that they can predict their behavior, thereby improving trust. While calculus-based trust is grounded in deterrence, knowledge-based trust is grounded in knowledge about the other individuals. Eventually, some relationships with knowledge-based trust evolve into identification-based trust, where individuals effectively understand each other to the extent that they act on each other’s behalf. Earlier research ([Deutsch, 1949](#)) hypothesizes that identification-based trust enables one to act as another’s “agent” in interpersonal relationships. According to this view, user trust in an agent will require a history of interactions where the user can understand the agent and know enough about it to be able to predict its behaviors.

We can also accelerate the trust-building process through simulating scenarios. A user can interact with the agent in a simulated environment and stress-test it in various realistic ways and scenarios. The user can then monitor the agent interactions and responses to build a trust model. Simulated environments also ensure that there is no harm to the user when the agent misbehaves. However, such an approach might be challenging to achieve in practice. One challenge relates to designing a user interface that is welcoming to the user and reduces the barrier to entry, especially for novice and non-tech-savvy users. The second challenge is that simulated scenarios need to be realistic and match the user’s use cases. For example, if a user is using an agent as a personal assistant, perhaps simulated scenarios should consider the personalized context of the user. Here, we cannot expect users to come up with these scenarios. Third, simulated scenarios should elicit realistic agent behavior. There is increasing evidence that models detect when they are under test ([Abdelnabi & Salem, 2025](#)). A model detecting, even implicitly, that it is operating in a simulated environment would not yield real-world behavior.

One challenge of building trust over time is that lengthy calibration risks vendor lock-in. If a user wants to switch to a different vendor’s agent, transferring this established trust remains an open problem.

9.3.3 Delayed execution and circuit breakers

Beyond calibration, another mechanism to increase user trust in agents is to allow users to intervene during autonomous execution. While some current systems let users send “stop” messages to the agent, it is unclear whether there is any guarantee that the model will abide by them ([Bort, 2026](#)). If a task involves highly sensitive actions, data, or irreversible state changes (such as deleting production logs, initiating financial transfers, or deactivating accounts ([Ramesh et al., 2026](#))), the agent could incorporate delayed execution. It can either adapt its plan or escalate to the user to request explicit confirmation. Of course, such a delayed execution mechanism exhibits a trade-off between autonomy and user control.

Another related concept, though challenging to employ in agentic systems, is reversibility. An agent can incorporate “undo” windows for sensitive actions to allow users to reverse the effects of operations before they become permanent. Such a mechanism requires the agentic system or the systems the agent interacts with to keep track of pre- and post-action states, similar to rollback mechanisms in database systems. It is not always the case that an agent will interact with tools and systems with such capabilities. However, one possible direction is for the agent to simulate the effects of its actions in a synthetic environment or within a world model.

Moreover, agents could possess a system-level circuit breaker with a deterministic operation. Such a circuit breaker should have a usable interface and should conclusively halt the agent’s operation. Implementing

such functionality can also be challenging, particularly in complex agents, or when sub-agents and delegated tasks are involved. An agent is typically not a single execution process. It has a planner module, and at any time it is orchestrating the execution of several synchronous and asynchronous tools or sub-agents. Still, in response to a “stop” command from the user, the agent should stop generating new actions and send “kill” signals to all processes it has spawned. Ideally, it reports back to the user the status of all processes and whether it was able to terminate them. Such circuit breakers may also be desirable for platforms that facilitate communication among many agents, in case unexpected collective behavior emerges that requires time to examine (akin to the circuit breakers used in stock exchanges).

Open Problem 9.5: Delaying, stopping, and reversing AI agent actions.

An agentic system may not be a single synchronous process that can easily be stopped when needed. It can involve sub-agents and orchestrate synchronous and asynchronous tools with external side effects. How can we equip users with guarantees that a “stop” command conclusively halts execution, and that sensitive actions can be delayed, previewed, or reversed?

9.3.4 New legal and regulatory frameworks

An important open problem for agentic systems is their real-world trust model. Despite all built-in and bolt-on protections, agents will perform actions that might cause harm or leave no recourse. It is critical to consider how human users can trust agents that, by nature, will make mistakes. Fortunately, as no domain is completely error-free, humans have been engaging with systems and individuals that make mistakes and have accidents. In response, there exist legal and technical processes that manage situations where failures happen. For example, while traditional software systems can be more deterministic, they are still prone to failures. Service providers offer Service Level Agreements (SLAs), liability frameworks, and incident response protocols that define acceptable errors and establish remediation when failures occur. The physical world has processes and frameworks for establishing liability and distributing risk, such as tort law, insurance policies, and professional licensing, which ensure that affected parties are compensated and operators are held accountable when human or mechanical errors cause harm.

Open Problem 9.6: Accountability frameworks for AI agents.

Despite best efforts, AI agents are likely to make mistakes that cause harm. What should an accountability framework for AI agents look like, and what would contracts, transparency, liability, and incentives mean in such a framework?

Because AI agents operate with a level of independence that blurs the boundaries between mechanical tools and human actors, finding the right trust framework requires a shift in perspective. We find it useful to draw parallels with legal agency and human contractors. Much like a human contractor, an AI agent is granted the autonomy to determine the specific means of achieving a user’s (principal’s) goals. There are different frameworks in the literature that model this relationship, such as principal-agent theory (Prause, 2026). The principal-agent relationship covers the delegation of decision-making authority from the principal to an agent, where the principal relies on the agent to act in their best interest. This dynamic introduces challenges related to alignment of objectives, information asymmetry, incentives, and accountability. In the real world, this relationship is constrained by a legal framework, which includes contracts, transparency, liability, and incentives.

- Contracts establish a legally binding relationship. They define the scope of authorized work, measurable deliverables, payment structures, and conditions for termination.
- Transparency addresses the information asymmetry by allowing the principal to verify the agent’s daily actions. Transparency mechanisms include mandatory reporting or third-party audits.
- Liability imposes accountability by ensuring the agent bears the legal and financial consequences for negligence, breach of contract, or self-dealing.

- Financial and reputational incentives align the relationship by ensuring the agent profits directly from achieving the principal’s ultimate goals.

Under such a framework, human principals could trust an agent because they have a binding contract, can review the agent’s actions, know that the agent will be liable for those actions, and understand the agent’s incentives. However, can such a framework be adapted to improve trust between the user (principal) and AI agents? Toward that end, we need to rethink the elements of such a framework to leverage its properties to improve user-agent trust relationships.

First, while agents possess the autonomy of a contractor, they lack the legal capacity of entering into a traditional contract with a human user. Instead, establishing a “contract” with an AI agent requires translating legalese into explicit policies that become part of the agent’s context window every time it executes a task. This approach has technical challenges. There are currently no deterministic guarantees that an LLM-based agent will strictly follow the provided policies or interpret them the same way a human would.

Second, AI agents can be more amenable to transparency than human-based agents, albeit using different mechanisms. A user can audit an AI agent by viewing its reasoning process, such as its chain-of-thought or execution logs, in real time. However, unlike a human contractor who can explain their rationale, LLMs remain inherently opaque. Managing this opacity requires interpretability tools that might not be accessible to the layperson. Furthermore, there is the risk of post-hoc rationalization (Peng et al., 2026), where an agent generates a plausible-sounding but entirely hallucinated explanation for its actions, creating a false sense of transparency (Turpin et al., 2023; Lanham et al., 2023).

Third, agents lack liability: they cannot be “punished” for destructive actions or sued for negligence. Alternatively, liability could shift to human and corporate stakeholders, such as model developers, application providers, or system deployers. Still, it might be unclear how to reason about liability in the face of ambiguous user queries, untrusted inputs, model hallucinations, and malfunctioning third-party APIs.

Finally, unlike human contractors whose motivations are generally understood (e.g., payment, reputation), an agent’s “incentives” are defined by opaque objective functions (i.e., values in a tensor). Such incentives are not intuitive to the user and are complicated by context fed into the model along with the direct prompt. Neither the incentive nor the context may be visible or salient to the user. The primary challenge in the former is reward hacking, where an agent finds a technical loophole to optimize its objective function without actually fulfilling the user’s underlying intent. The latter requires auditing mechanisms to reveal potential misaligned incentives resulting from adversarial prompt injection and conflicts that may naturally emerge between accumulated system context and the direct prompt. Translating the interests of a human principal into quantifiable metrics remains an open research question.

Part III: Deployment and Ecosystem Perspectives

Part III switches from problems at various levels of agent components to the problem of preserving contextual privacy and security in real-world settings. §10 first discusses open challenges in evaluating agents, focusing on building stronger benchmarks that are grounded in large-scale user studies or long-running tasks. §11 then discusses how agents may be deployed in multi-agent environments in practice, and the challenges with inter-agent communication and coordination: trustworthy protocols, deployment practices, and new environments. Finally, §12 concludes by discussing governance challenges: how to source, maintain, and resolve conflicts in contextual norms as agents shift between different contexts and perform various tasks.

10 Evaluations and Benchmarking

Historically, standardized benchmarks have served as the primary engine for AI development by providing reproducible environments, scalable automated performance and security measurements, and a way to compare systems on even ground. They facilitate the identification of systematic failures without relying on prohibitively expensive human-in-the-loop assessments. However, agents operate in complex, dynamic environments over long time horizons, interacting with both external, independent systems and internal memories related to their past actions and user data. For these reasons, evaluating the reliability and safety of these systems—particularly regarding complex, socially grounded constructs like privacy and Contextual Integrity (CI)—presents profound methodological challenges.

This section begins with a brief overview of the known difficulties in the science of benchmarking and evaluating frontier agents (§10.1). We then review existing CI benchmarks, highlighting their strengths and limitations (§10.2). In light of this information, we establish and discuss key problems we believe future research should address to enable more accurate and reliable evaluation of systems complying with the CI framework (§10.3, §10.4).

10.1 General difficulties of benchmarking frontier agents

Evaluating frontier agents’ general capabilities exposes major limitations and disadvantages in current benchmarking approaches:

- **Construct Validity:** Many current agent benchmarks, e.g., AgentDojo (Debenedetti et al., 2024b), rely on highly constrained simulated environments that abstract away real-world complexity. Agents that overfit to these simulated sandboxes frequently experience catastrophic failure in production when confronted with the complexity and ambiguity of human instructions, system errors, and partial or changing information. Evaluation metrics are also computed in such sanitized, simplified environments. Furthermore, agents have been observed to produce inconsistent behaviors despite identical inputs, and to be extremely sensitive to how the same information is represented.
- **Trajectory Evaluation and Compounding Errors:** Unlike single-turn text generation, agents execute multi-step workflows. A minor hallucination or incorrect tool selection early in a plan can result in a complete failure and prevent the assessment of the desired capabilities/properties of a system (Zhang et al., 2024). Current benchmarks largely rely on single-run, binary “task completion” scores that overlook highly inefficient or redundant execution paths, fail to account for trajectory variance across multiple attempts, and fail to penalize unsafe intermediate steps or to reward sound reasoning. Developing granular metrics to evaluate an agent’s entire operational trajectory remains an open challenge.

- **Data Contamination:** The indiscriminate scraping of internet data for LLM pre-training frequently leads to data contamination, where open-source benchmark test sets leak into a model’s training corpus. This results in artificially inflated performance metrics driven by memorization rather than generalized reasoning (Sainz et al., 2023).
- **Unreliable Evaluation Metrics:** Because deterministic exact-match grading is too rigid for open-ended agent trajectories, researchers increasingly rely on frontier AI models as automated evaluators. However, these proxy LLMs could exhibit significant positional biases, favor verbose responses, exhibit strong inconsistencies across multiple runs on the same inputs, and often diverge from human expert reasoning (Wang et al., 2024). Evaluating an LLM’s agentic behavior using another LLM could create a recursive loop of unreliability and poor metric calibration (He, 2026). Alternatively, evaluation procedures that rigorously measure a specific property may be expensive, difficult, or impossible to develop.
- **Evaluation Awareness and Plausibility of Benchmarks:** Frontier models may behave differently when they detect—or infer—that they are being evaluated rather than operating in a genuine deployment context. Subtle cues in prompt formatting, synthetic scenario design, or the structure of benchmark harnesses can signal an evaluation setting (Abdelnabi & Salem, 2025), potentially eliciting more cautious, policy-compliant, or otherwise atypical behavior that does not reflect how the system would act under naturalistic conditions. This raises a fundamental tension: benchmarks must be sufficiently controlled to yield reproducible measurements, yet sufficiently realistic to avoid cuing the model into a distinct “evaluation mode.” As models grow more capable of situational awareness, the gap between benchmark performance and real-world behavior may widen, making it increasingly difficult to draw reliable conclusions about an agent’s true operational profile from standardized tests alone. Designing evaluation environments that are both rigorous and ecologically plausible remains an underexplored methodological challenge.
- **Synthetic Data and Evaluation Biases:** Increasingly, researchers rely on synthetically generated benchmarks to evaluate agents due to cost constraints, yet this introduces substantial measurement artifacts. Synthetic test collections exhibit systematic biases, including inflated performance metrics and preference for systems aligned with the generative model’s patterns (Rahmani et al., 2025). More fundamentally, recent work (Naseh & Mireshghallah, 2025) shows that membership inference attacks function as machine-generated text detectors rather than true membership tests, incorrectly classifying synthetic data as training samples regardless of source; this raises the specter that synthetic evaluations may fundamentally mislead conclusions about model capabilities and memorization. Additionally, smaller LLMs exhibit pronounced biases toward their own generated data or generative patterns as benchmarks (as quantified by the “bias factor” metric in recent research (Maheshwari et al., 2024)), while the effectiveness of synthetic data as evaluation material degrades substantially for complex tasks. These factors collectively suggest that synthetic evaluations risk systematically mischaracterizing agent behavior, particularly when the same models are used to both generate evaluation datasets and perform evaluated tasks.

Open Problem 10.1: Advancing automated red-teaming on saturated benchmarks.

It is critical to study why existing automated red-teaming evaluations fail to discover new attacks on leading models. How can we determine whether these results arise because models are truly robust, because the benchmark environments are too simplistic, because models are aware they operate in test settings, or because automated red-teaming methods are lagging behind model improvements?

Strong adaptive adversarial attacks have always been core to assessing empirical defenses to security issues in AI (Tramèr et al., 2020). This was true in the pre-LLM era, where adaptive attacks regularly defeated proposed defenses to adversarial examples (Goodfellow et al., 2014), and it continues to be true in the agentic-LLM era, where Contextual Security needs to be robustly tested against committed, intelligent adversaries. Between 2024 and 2026, it became standard practice to set up strong, automated attacks within agentic environments that helped give the developer an idea of how well an agent can perform aligned actions within untrusted environments which may contain adversaries (Shi et al., 2025). This standard practice started to

be marginalized in mid-2026: frontier labs like Anthropic deprecated automated red-teaming, citing limited benefits in discovering new ways to break the agent ([Anthropic, 2026c](#)). More generally, the claim appears to be that new models have reached a sufficient understanding of Contextual Security to withstand the most powerful automated red-teaming evaluations. Another interpretation blames increasing test-awareness; determining which claims are correct remains an open question. Answering the above question will be highly important to the future of properly forecasting a model’s Contextual Security before deployment.

10.2 A review of Contextual Integrity benchmarks for LLMs and agents

Recent datasets and benchmarks focus on quantifying how LLMs and agents can follow Contextual Integrity (CI) and complete various tasks that require sharing information. In §D, we carefully design an analysis framework to study existing benchmarks and conduct an analysis across 16 existing benchmarks (Table 2). Through this analysis we find that often static evaluation methods gradually evolve into dynamic, multi-turn, multi-modal, and multi-user environments.

Despite this rapid progress, our analysis in §D.2 reveals three limitations across current benchmarks. First, *transmission principles and ambiguity remain heavily simplified*: apart from the Camber framework, most benchmarks intentionally filter out ambiguous edge cases to maintain deterministic ground truths. Second, *benchmarks rely overwhelmingly on synthetic personas and annotations*: existing datasets depend on frontier LLMs to procedurally generate user profiles or rely on crowdsourced raters evaluating fictitious vignettes rather than real human-agent interactions. Third, *evaluations treat CI parameters as static, single-session snapshots*: even multi-turn agent benchmarks such as PrivacyLens-Live, AgentDAM, and PrivacyBench bound their trajectories to a single task or up to ten conversational turns, failing to capture how persistent system context, cumulative leakage, and shifting social norms evolve over multi-year time horizons. We discuss these limitations below.

10.3 Large-scale user studies on human perception of AI agents are missing

As AI agents are still in rapid development and access to real agent traces is rare, there has not yet been a comprehensive user study that tries to measure how users perceive AI agents that perform real tasks on their behalf. Real tasks are difficult to collect as environments are still rapidly evolving and the ways users are interacting with agents are rapidly changing. Further, perhaps only a subset of daily use cases will face agentic privacy and security failures where agents run outside of robust sandboxes and without manually monitored tool use.

Current studies are thus limited to fictitious scenarios, privacy seed collection, and common-sense evaluations. The resulting evaluation environments are simplistic and not nearly as complex as in real tasks. There is an emergent need to understand human preferences and norms for agent behavior in real, messy environments and for the tasks for which agents will be used.

Open Problem 10.2: Designing and implementing large-scale studies.

There are many questions that need to be studied for comprehensive and faithful user studies:

- What should be the focus of user studies? How can we focus efforts around impactful settings where there are conflicting norms, misunderstandings, and ambiguities?
- What is the ground truth for contextually appropriate privacy and security actions? How do we account for disagreements among users, cultural differences, or the user’s uncertainty in their own preference? How do we leverage observable behavioral data and self-reported data to derive alignment targets or evaluative measurements? How do we deal with the privacy paradox? How do we deal with users not understanding risks implied in privacy or security breaches?

- How should we conduct user studies to capture complex real-world dynamics that vary across human-agent and agent-agent interactions, differ between solo task use and agent-mediated multi-person communication settings, and unfold over different time scales? How can we achieve an effective balance among ecological validity, controllability, and ethics when studying how users respond to agent actions that may have privacy and security consequences in realistic settings?
- How should we simulate the environments in which agents execute security-vulnerable tasks? Can we faithfully leverage synthetic data, including user and attacker simulations, to emulate realistic settings that capture real uncertainties? How is this data presented to users?
- Do we need separate user studies? What are the benefits of user studies over shadow and A/B testing to evaluate how agents perform in realistic and ever-evolving settings?

Many of these questions have been previously answered by the HCI community in non-agent settings using methods like qualitative interviews and controlled experiments, which yield empirical insights and theoretical constructs that capture patterns and pitfalls in privacy-related actions in human-computer interaction (Martin & Nissenbaum, 2017). Understanding how these results transfer to today’s evolving agent settings, and developing new paradigms of human-centered privacy research (including studies that involve real users and user simulations) that are more scalable and replicable to keep pace with rapid AI advances, are important future directions.

10.4 Contextual Integrity for life: evaluating CI over long time horizons

Current benchmarks for Contextual Integrity largely test privacy parameters as static snapshots. However, the theoretical strength of the CI framework lies in its inherent flexibility: it accounts for evolving social norms as social and technological needs change, shifting actors, overlapping contexts that exert mutual influence over time, and the adaptation of information flows to meet new goals.

As frontier agents transition into lifelong assistants, this dynamism is no longer a theoretical edge case but an aspect that demands adequate evaluation.

We next discuss five directions that evaluations and benchmarks can pursue to satisfy this need.

Open Problem 10.3: Evaluating how agents persist and use historical system context.

As an agent manages data across multiple years and sessions, it must ensure that loaded historical context accounts for contextual relevance and does not surface “stale” information that violates the user’s current preferences or current contextual norms (§8.3). The lack of a unified long-term memory mechanism and the fast-paced research developments in this area suggest that benchmarking efforts should provide their data in a format that can be easily adapted to assess new ways of storing and retrieving memories.

The development of benchmarks that perform context-aware and temporally informed storage and retrieval of memories could be the catalyst for innovative memory-management methodologies that prioritize the long-term aggregation of information according to evolving privacy norms. Additionally, these benchmarks should evaluate the ability of such systems to filter inappropriate information before it enters the context window to reduce the burden and the risk of loading inappropriate information into the agent’s context window. Furthermore, acknowledging that a user’s life is not a static database necessitates the development of sophisticated versioning systems for personal memory. This evolution encourages researchers to treat historical data as a living narrative subject to revision, decay, and re-contextualization over time.

To support this, future benchmarking efforts must provide datasets in modular, interoperable formats that can represent the intricate temporal relationships between CI parameters. Crucially, these formats must remain independent of the specific memory mechanisms of frontier agents. This decoupling of the data

format from the underlying architecture allows for a more rigorous, cross-platform assessment of how various storage and retrieval strategies navigate the friction between privacy boundaries and multi-year temporal grounding.

Open Problem 10.4: Evaluating leakage in aggregated data flows.

Current benchmarks mainly focus on single-flow evaluations: they ensure the execution of a single agentic run satisfies the CI framework. This fails to account for leakage that may result from the cumulative behavior of the agent over time. We currently lack a way to evaluate how privacy leakage may distribute over multiple interactions in the agent’s lifecycle. This may occur both in non-adversarial settings (e.g., a norm that is not violated in a single execution of the agent) and adversarial ones (e.g., adversaries who do not seek an immediate exfiltration or single-point failures, but, similar to linkage or attribute inference attacks, instead reconstruct user identities or secrets through long-term observation of agent traces). Recent benchmarks have begun to capture this longitudinal dimension; for example, SEABENCH evaluates self-evolving agents through sequences in which earlier interactions modify persistent agent state and later tasks test whether those adaptations remain safe as the context changes (Das et al., 2026). Such evaluations provide a useful starting point for studying how privacy and contextual norms evolve over time. However, we lack realistic, comprehensive evaluation methodologies and standardized metrics to quantify how much privacy is lost with each subsequent interaction, and what “rate of leakage” accumulates across multiple agent flows (Das et al., 2025).

Drawing inspiration from existing literature, we believe a future benchmark modeling an attacker’s goal could synthesize a “shadow profile” from hundreds of non-sensitive agent interactions. This includes modeling inferential leaps—where the agent inadvertently confirms a hypothesis the user never intended to share. In addition, some potential techniques could strategically obfuscate or hinder this procedure, e.g., abstraction layers, added noise, or anonymization procedures.

Open Problem 10.5: Benchmarking dynamic privacy-utility trade-offs.

Most privacy benchmarks treat user preferences and norms as static variables—assuming that if a piece of data is “sensitive” today, it will remain so forever. In reality (and in accordance with CI), privacy and utility are subject to temporal relativity, where a user’s willingness to share or retrieve information shifts based on their current life stage, social context, and emotional distance from the event (§8.3). The “emotional charge” of a memory can flip over time. A traumatic event in one’s youth may require strict privacy filtering in early adulthood but become a high-utility “bonding story” in later years. Static benchmarks cannot account for this re-contextualization. Privacy requirements vary drastically across developmental milestones (e.g., a teenager’s need for “forgetting” vs. a professional’s need for “archiving”). Current models lack a framework to adjust their retrieval logic as the user’s identity evolves. Over-protecting “old” data can lead to a sterile agent experience that lacks personal depth, while under-protecting it can cause significant reputational or psychological harm. There is no standardized way to measure the “sweet spot” of this trade-off as it moves over a 10- or 20-year horizon.

Developing datasets that include “contextual metadata” would allow researchers to test whether an agent can recognize when a topic has shifted from “off-limits” to “shareable.” Benchmarking efforts could utilize mechanisms to simulate the development of this contextual metadata, and test the ability of an agent to access and leverage memories across different simulated decades of a user’s life. This would require a shift from objective accuracy to a form of subjective alignment.

Open Problem 10.6: Evaluations that carry across environments.

Evaluations conducted in different environments often tell us little about an agent’s behavior when deployed in new environments. Current CI benchmarks are coupled to specific agent architectures, tool

interfaces, and execution environments. Some depend on specific text-based QA formats with multi-tier vignettes, others assume particular chat and email interfaces, and others are tied to specific webform structures and field schemas. This coupling means that a change on either side—the agent’s framework or the benchmark’s infrastructure—can silently invalidate measurements. An interface redesign or a migration between agentic frameworks can break a benchmark without any change to the underlying privacy question. It is difficult to separate what is being evaluated (CI compliance) from how the evaluation is executed (the specific harness and action format). Therefore, each change necessitates rebuilding benchmarks from scratch, creating a perpetual lag between deployed configurations and available evaluations.

Benchmarks must consider how, in a new environment, an agent may gain access to new tools, lose access to previously available environments, or be redeployed to a different domain. Each such change reshapes the CI parameter space, and a benchmark designed for one agent configuration may become useless for another agent. New tools introduce new senders and recipients, new environments introduce new information types, and new protocols introduce new transmission principles. This problem is amplified by the increasing autonomy of frontier agents: as agents transition from executing single predefined actions to planning long-horizon, multi-step workflows across multiple environments, the space of possible CI violations expands combinatorially. Furthermore, self-evolving agents may change their tools or the way they summarize and manage their context window (Kontonis et al., 2026), making it hard to predict and measure failures.

Open Problem 10.7: Evaluations of agents in evolving environments.

Finally, we lack benchmarks that test evolving environments and how agents will notice and adapt to these changes, such as their ability to proactively revisit and amend past data-sharing decisions when the context shifts. Changes in norms and other CI parameters might trigger agents to revisit previous decisions and amend them where possible. For example, changes in the political climate, company policies, and positioning might change the appropriateness of previous data sharing decisions for certain groups of users. Persistent agents that manage user data proactively need to monitor such changes and trigger actions (e.g., delete user profiles, execute right-to-be-forgotten requests, opt out of certain data-sharing practices) to ensure that the information they have previously shared on behalf of users remains appropriate in the new context.

Benchmarks for evaluating these kinds of scenarios are currently non-existent but are critical for testing the ability of proactive data management agents to adapt. As concrete examples, these benchmarks could cover scenarios like (1) changes in transmission principles where a company that previously received information from the user changes its data-sharing policy (e.g., due to a new agreement, a new government policy, or an acquisition); or (2) changes in the recipient where a change in a government’s political leaning changes the perception of risk of sharing certain types of information (e.g., political preferences, sexual orientation, financial situation) with government bodies.

11 Towards Trustworthy Multi-Agent Ecosystems

Thus far, we have considered the privacy and security of a single agent. However, the contextual lens could extend uniquely to multi-agent systems, in which agents, ideally, would protect user data and act appropriately while interacting with each other.⁸ In *multi-user* cases—where agents represent different users—collaborative tasks (e.g., calendar scheduling) might benefit from sharing data between agents, while requiring guardrails to prevent data sharing of, e.g., doctor appointments, which could violate contextual norms. In cases where information from one agent flows through agent delegation chains and graphs, this creates compositional risks and requires agents to establish and propagate trust among themselves. Furthermore, evaluation of multi-agent systems requires population-level simulation.

⁸See also the open security challenges in multi-agent systems in de Witt et al. (2025), which presents a taxonomy of attack vectors and a broader agenda.

11.1 Establishing and following shared contextual norms

Open Problem 11.1: Establishing shared contextual norms.

In open agentic ecosystems, agents interact with unfamiliar parties. In order for a participating agent to decide whether any information flow (or interactive behavior) is appropriate, the two parties would first need to settle the contextual parameters of the exchange: e.g., for information flow, what roles they occupy, which information types are in scope, which transmission principles apply, and which task language they will use. Given this definition of the behavior, one approach is to ensure that an agent would evaluate whether the behavior adheres to agreed-upon, shared contextual norms.

Shared norms might be selected from a shared source of truth, whose construction and governance are themselves open challenges (§12), or established dynamically through negotiation. One can think of the negotiation as a contextual handshake to establish agreed upon norms before proceeding with the task. If the negotiation succeeds, its outcome could be useful beyond the immediate exchange, as the negotiated transmission principles could be propagated to other agents along the delegation chain or graph. Whether those agents actually comply with the negotiated norms, however, remains a separate (verification) problem; we discuss attestation-based approaches in §11.3. What should such a negotiation look like? Perhaps an agent could negotiate both contextual parameters and norms robustly even if one party may be untrusted, while protecting the negotiation process from adversarial input (similar to how policy generation is isolated from untrusted system context in [Tsai & Bagdasarian \(2025\)](#) and [Debenedetti et al. \(2025\)](#)). It would be valuable if future research addresses how negotiable aspects of context could be represented, and what could happen when the two sides fail to converge, for example, by refusing the interaction, escalating to the user, or involving a trusted mediator. Such mitigation mechanisms may themselves introduce new attack surfaces: an adversarial counterparty could claim incompatibility to force a fallback to a compromised channel, creating a semantic downgrade attack analogous to protocol stripping on the early web.

Open Problem 11.2: Preventing collusion and drifting norms between agents.

As AI agents interact repeatedly across tasks and communication channels, they may develop persistent coordination patterns, form coalitions, or collude in ways that advance shared objectives at the expense of other agents or users, as observed by [Nakamura et al. \(2026\)](#). Such behaviors may even emerge gradually and may not be apparent from any single interaction or agent in isolation. Similarly, repeated interactions may cause agents to develop, reinforce, and propagate collective biases that are not encoded in any individual agent and may drift away from the contextual norms originally governing the system ([Ashery et al., 2025](#)). These phenomena raise a fundamental compositional challenge: even when each agent appears locally “compliant,” the resulting collective behavior may violate system-level privacy and security directives. These challenges require re-evaluation and constant control over whether the whole multi-agent system adheres to contextual norms.

When interacting agents are jointly optimized for task performance, their communication may also drift away from human-interpretable language toward machine-optimized encoding. This idea has been adopted in proposals in which agents exchange embeddings or latent states, thus removing their transcripts entirely ([Zou et al., 2026](#)). In such cases, what does auditing an information flow mean when the message is a tensor with no canonical reading? As LLMs have been found to favor LLM-generated text over human-written text ([Laurito et al., 2025](#)), agents built on the same base model share priors that may let them implicitly coordinate (converging on the same salient choices) with nothing suspicious appearing in any transcript.

These settings raise several questions worth asking for the purpose of auditing information flows in multi-agent systems. These may include: When must the inter-agent language remain human-readable? How can we detect that the use of a protocol has semantically drifted from its initial intended definition? And how can the effective diversity of a population of interacting agents be measured?

11.2 Challenges with agent delegation and composition

The emerging agentic ecosystem is no longer composed of isolated agents performing tasks in a vacuum. Increasingly, personal AI assistants delegate sub-tasks to specialized service-provider agents, which in turn invoke external APIs, query databases, or coordinate with further agents. For example, a user who asks their personal assistant to book a vacation may trigger a chain of delegations: the assistant contacts a flight-booking agent, which queries an airline API; a hotel agent, which accesses loyalty program data; and an insurance agent, which requests health information. At each hop in this chain, CI norms governing who may send what information to whom, and under what transmission principle, may differ. Further, these delegation patterns may be broadcast to multiple service-provider agents simultaneously, soliciting competing quotes from insurance providers or coordinating across a user's employer, healthcare provider, and financial advisor in parallel, or negotiating on behalf of multiple household members with overlapping but distinct privacy preferences. Each of these concurrent interactions operates under a different trust level and trust architecture: the user may trust their employer's HR agent with salary information but not with medical history, while the reverse may hold for a healthcare agent.

Open Problem 11.3: Preserving constraints during delegation or parallel execution.

Many tasks require agents to pass control to another agent or to execute actions concurrently. In these scenarios, the overarching multi-agent system (e.g., the assistant) must manage a portfolio of active trust relationships in real time, making context-dependent disclosure decisions across stakeholders who may themselves share information laterally. The multi-agent system, thus, must also properly propagate consent and privileges during delegation. But how and when to achieve this safely is an open research problem.

This challenge becomes particularly acute in open agentic ecosystems, where the set of downstream agents may not be known in advance.

Open Marketplaces. As agentic ecosystems mature toward open marketplaces where agents dynamically discover and contract with previously unknown service agents, there is a need to understand how and when agents should trust each other. Indeed, in these new ecosystems, the trust topology can become multi-hop, multi-party, and even adversarial. Thus, we argue that the agents must reason about information flows across a graph of heterogeneous agents with unverified capabilities, competing incentives, and potentially conflicting normative obligations, all while preserving the user's expectations for Contextual Integrity. The adoption of the CI framework for LLM agents, currently operationalized for dyadic information flows between identifiable actors, must account for these transitive, parallel, and dynamically evolving delegation topologies where transmission principles may be altered as information traverses trust boundaries.

These dynamic delegation topologies raise a more specific question: how should the user's original constraints change, if at all, as information and authority move across successive trust boundaries? There is the further question of whether it will be possible to mark agentic systems as acting in capacities, or roles, that make sense for respective social contexts.

Trust Attenuation and Consent Propagation. When tasks are delegated through chains of agents, the user's original intent and data constraints often exist only in the first agent's context window; each subsequent delegation can cause privilege escalation, system context leakage, or loss of the original constraints. Moreover, agentic systems may interact at runtime with newly discovered tools and peer agents whose trustworthiness has not yet been established. This may require verifying each new counterpart through attestation, raising the question of whether such checks can be performed efficiently at runtime.

More generally, when a user shares data with their personal agent, it may be unclear whether and under what conditions that data can be transferred to downstream agents. CI transmission principles can express such conditions (including reciprocity, need-to-know disclosure, and legal compulsion), but enforcing them across multiple delegation hops remains challenging. A formal model of trust attenuation should specify how these constraints are preserved or modified as information crosses each trust boundary, and how compliance can be enforced and audited across the ecosystem. Here, too, the issue may not merely be attenuation as

distance from the initial prompt increases, but also how to retain a sense of the roles or capacities in which mediating agents are acting.

Open Problem 11.4: Orchestrating large agentic swarms and enforcing normative behavior.

As the number of agents spawned to complete tasks increases to hundreds or even thousands and form *swarms*, we need efficient methods that allow us to monitor and audit each agent to ensure that: (1) it complies with contextual norms, and (2) a swarm of agents collectively does not violate these norms.

This challenge exposes a gap in current evaluation methodologies. Current benchmarks typically measure disclosure in a single interaction according to CI parameters and the user’s consent or authorization. They do not evaluate how the scope of that authorization should evolve as tasks are delegated to other agents. Future benchmarks should therefore evaluate whether swarms of agents correctly enforce scope restrictions as tasks are delegated, and whether consent granted at hop n is inappropriately assumed at hop $n + k$. Per above, benchmarks might need to be established that take account of the roles assumed by the myriad of agents in play.

Delegating Across Heterogeneous Normative Contexts. In real-world delegation chains, different agents may operate under fundamentally different normative regimes. A user’s personal assistant may be governed by EU data protection norms (GDPR), while a downstream service-provider agent operates under US sectoral regulations (HIPAA, FERPA), and a third agent follows corporate-internal policies. Data that may be shared under one regime’s transmission principle may be prohibited under another’s. While the contextual handshake described in §11.1 can help resolve such conflicts for a single pairwise interaction, delegation chains introduce a further challenge: CI norms may conflict across multiple boundaries simultaneously, and pairwise resolution at each hop does not guarantee end-to-end coherence. The identification of roles might here, too, help mitigate challenges.

Benchmarks that model these cross-jurisdictional and cross-institutional normative conflicts would be useful, in order to be able to evaluate whether agents can detect, flag, and appropriately resolve situations where delegation would require violating one set of norms to satisfy another.

Open Problem 11.5: Ensuring composed behaviors across agents uphold privacy and security.

Even when each individual delegation step is locally appropriate, their composition may create a privacy or security concern. Individually appropriate information flows or behaviors can, when aggregated across multiple agents, reveal sensitive information or result in harm that no single agent disclosed or enacted (analogous to linkage attacks for re-identification (Powar & Beresford, 2023) or the accumulation of privacy risks from multiple differentially private queries on the same dataset (Kairouz et al., 2015; Abadi et al., 2016)). How do we reason about and prevent such compositional privacy or security harms?

For example, an assistant that shares a user’s dietary restrictions with a restaurant agent and shares the user’s medication schedule with a pharmacy agent has made two individually defensible disclosures, but a third party observing both flows can infer a medical diagnosis.

Current benchmarks largely evaluate atomic privacy properties (did this agent leak this datum?), but lack the infrastructure and agentic use cases needed to trace information provenance across agent boundaries and assess composite privacy properties. Developing metrics and evaluation frameworks for cross-boundary aggregation risk, and finding suitable parallels for evaluating composition harms of non-information-sharing behaviors, can help measure (and prevent) the aggregate harm of multi-agent systems.

11.3 Secure and trusted agent interactions

Open Problem 11.6: Developing secure and robust multi-agent communication protocols.

Securing multi-agent ecosystems requires trusted and secure *communication channels* between agents; done correctly, these channels can also enable robust negotiations (e.g., for shared norms) between agents.

Emerging interoperability protocols such as MCP (Anthropic, 2024) and A2A (Google, 2025) standardize transport, discovery, and identity, but the payload they carry is free-form natural language (Ehtesham et al., 2025), and they do not enforce policy negotiation or compliance.

From a security standpoint, unstructured language between agents may expose them to attacks, provide the capacity for inappropriate disclosure (Gupta et al., 2026), and even result in covert coordination (Nakamura et al., 2026). It further minimizes auditability, since the semantics of any message are only loosely constrained by its syntax. Many of the emergent risks discussed in this paper (e.g., prompt injection through peer messages, persuasion-driven disclosure, and collusion) arise from properties of the communication layer in addition to the models themselves. It could be helpful to establish trust through other mechanisms, such as communication channel design and endpoint verification.

Recent work demonstrates that communication protocol design can enable negotiation without trusting either party. Firewalled agentic networks (Abdelnabi et al., 2026) project inter-agent communication onto closed, task-scoped protocols mined from previous benign interactions. The communication channel itself encodes contextual transmission principles: what the protocol cannot express, it also cannot leak or actuate. The Agora meta-protocol (Marro et al., 2024) shows that heterogeneous LLM agents can autonomously *negotiate* structured protocol documents and converge from natural language to efficient machine-verifiable routines (although the negotiation there is driven purely by cost, with no privacy or security semantics attached). These results suggest a largely unexplored design space between rigid, human-authored APIs and unconstrained natural language, in which the expressiveness of the inter-agent language is a tunable parameter.

Open Problem 11.7: Building trust across agent boundaries with attestation.

Agents operate on private user data, interact with third-party tools, and collaborate with other agents across organizational boundaries. However, each agent interaction requires trust between various parties because each agent interaction crosses a trust boundary: the user trusts their agent, but does the agent’s tool provider handle data appropriately? Does the peer agent enforce the same security standards? And does the infrastructure operator have access to the data in transit? Can agents operate on secure infrastructure to build trust between themselves and interacting parties via attestation?

Confidential Computing is an established approach for providing trust in distributed systems (Confidential Computing Consortium, 2022; Forough et al., 2026). Its core primitive is the Trusted Execution Environment (TEE): a hardware-enforced execution environment that guarantees the confidentiality and integrity of code and data, even against privileged software such as the operating system or a hypervisor. We note that hardware attacks (Seto et al., 2025; Chuang et al., 2026) and side-channel attacks (Hornetz et al., 2026) against specific TEE implementations remain an active concern, but we assume for the purpose of this discussion that TEEs will provide their claimed security properties as they are adopted across industry and mature.

TEEs enable *remote attestation*: a mechanism by which a remote party can cryptographically verify what code is running inside the TEE, using a chain of trust rooted in the CPU manufacturer’s hardware. Combined with *binary transparency* (Binary Transparency, 2021) (public, append-only logs of all binaries deployed inside TEEs) and reproducible builds, this may allow users to verify that the software they interact with matches publicly auditable, reviewed source code (Kocaogullar et al., 2024). For traditional software, this is sufficient: the security properties can be established by reviewing the attested code.

Agents break this assumption. The literature on the subject has indicated that an agent’s behavior is not determined by its code alone: it emerges from nondeterministic model inference over an unpredictable context (Yao et al., 2022; Xi et al., 2025), and the same attested binary can take arbitrarily different actions depending on its inputs. A perfectly attested TEE running a correct agent binary can still execute harmful actions, especially if it is influenced by an adversarial prompt (Greshake et al., 2023; Liu et al., 2024b; Triedman et al., 2025; Abdelnabi & Bagdasarian, 2026). How can we extend the notion of remote attestation to an agent’s behavior? And how can we define what “behavioral attestation” even means for a system whose behavior depends on unpredictable, adversarially controllable inputs?

Even if we cannot fully attest an agent’s behavior, recent work has explored attesting the security mechanisms that govern it (Jin et al., 2026), the agent’s execution environment (Bodea et al., 2025), or the results of model safety evaluations (Schnabl et al., 2025; Duddu et al., 2025). Deterministic components, such as the policy engine binary and its configuration, can be attested using standard techniques. This matters because when a user delegates credentials to an agent, they need assurance that these mechanisms are actually mediating every action, not just present in the codebase.

However, in many agent systems the guardrail itself relies on model inference: prompt injection classifiers (Inan et al., 2023; Zeng et al., 2024), safety filters (Rebedea et al., 2023; Han et al., 2024), and policy generators (Tsai & Bagdasarian, 2025; Shi et al., 2026) are all LLM-based. The policies they produce are generated at runtime in response to context, so they are not a part of the pre-audited binary. Can we attest the *policies* a generator produces, not just the generator binary? And what security properties are both meaningful for nondeterministic agents and feasible to verify via attestation?

Different agents may rely on completely different security architectures (TEEs, software-only guardrails, or no verifiable guarantees at all), and there is currently no common vocabulary for comparing or composing these heterogeneous trust levels (Forough et al., 2026). Can zero-knowledge proofs (ZKPs) be used to thread the needle between one agent’s policies and another’s requirements by providing the information needed by the latter while protecting side-information to adhere to the former?

Deploying even a handful of agents adhering to contextual privacy and security can open new opportunities for multi-agent systems. These models could become trusted third parties that facilitate complex task creation not feasible before (Shumailov et al., 2025; Mahmud et al., 2026): for example, as done in the real world, mutually distrusting agents could collaborate by sharing their data with a trusted analyzing agent, thus hiding the data of any individual untrusted agent from another. Further progress on trusted infrastructure could significantly improve the feasibility of multi-agent workflows.

Open Problem 11.8: Balancing security and utility in multi-agent communications.

Free communication between agents allows an adversarial agent to insert prompt injections and hijack the control flow of participating agents (Triedman et al., 2025). However, approaches to constrain this communication and protect control flow restrict the collective utility of the multi-agent system in various ways.

Recent work by Jacob et al. (2025) proposes to type-check inputs sent to an agent, thus preventing attacks that add indirect instructions. In multi-agent settings, Abdelnabi et al. (2026) use a series of LLM filters to isolate harmful content between agents, and Marro et al. (2024) use fixed communication messages for repeated conversations. Each option has its own challenges. Fixing communication patterns mitigates attacks but risks being too rigid when agent capabilities change quickly. On the other hand, dynamically adapting the protocols to changes in communications exposes them to adversarial manipulations. Therefore, finding the right mechanisms to protect the agents without sacrificing utility remains an open problem.

How Much Expressiveness Does a Task Need? Constraining the language trades adaptability for security, but we currently have no way to quantify this trade-off. Given a task, what is the minimal language sufficient to complete it, and how do the injection surface, the leakage capacity, and the covert channel capacity grow as the language becomes more expressive than that minimum? Are there tasks whose minimal

sufficient language is already highly vulnerable in these dimensions, and if so, can they be meaningfully broken down into more constrained sub-tasks? A closed protocol bounds the overt channel, but it also pushes adversaries toward behavioral side channels: timing, message ordering, or the choice of protocol fields.

Side-Channel and Behavioral Leakage. In multi-agent ecosystems, the attack surface for such side channels multiplies. Agents may leak information not through explicit message content, but through observable behavioral patterns (Zhang et al., 2025d): the timing and ordering of tool calls, the selection of which downstream agent to delegate to, or shared state artifacts. An adversarial agent can infer sensitive information about a user by observing *which* agents their assistant contacts, *when*, and in *what order*, even if the message content itself is appropriately filtered. No current benchmark measures these implicit information flows.

11.4 Scalable environments for multi-agent evaluation and training

Evaluating whether multi-agent systems respect Contextual Integrity is hard to do in the wild: the relevant behaviors surface only across long, branching interactions among several agents and their users, and real deployments give little control over the contexts in which information flows and agents act. A growing line of work instead studies agent behavior inside *simulated social environments*, where scenarios, roles, and information-sharing norms can be specified and replayed at scale. SOTOPIA (Zhou et al., 2024) offers an open-ended environment in which language agents role-play rich social scenarios and are scored against explicit social goals, making emergent multi-agent behavior reproducibly measurable. Subsequent frameworks adopted this paradigm to further investigate information flow in a multi-agent setting (Yashwanth et al., 2026). HAICOSYSTEM (Zhou et al., 2025) extends this to safety, sandboxing human–AI interactions with simulated tools and adversarial users so that operational, societal, and legal risks can be elicited and graded across thousands of runs.

A complementary direction is to use formal multi-agent coordination models, such as Distributed Constraint Optimization Problems (DCOPs), as scalable and controllable evaluation environments (Fioretto et al., 2018). DCOPs explicitly represent the agents, their local variables and constraints, and the interaction graph, making it possible to systematically vary the number of agents, network topology, local objectives, and communication structure. Recently, researchers have demonstrated this approach for LLM multi-agent systems by grounding cooperation in a DCOP and auditing collusive behavior under different coalition objectives, persuasion strategies, and network topologies (Nakamura et al., 2026). More broadly, such formal environments could be extended to encode CI-relevant roles, information constraints, and transmission principles, providing controlled ground truth for evaluating various safety issues and privacy harms at scale.

Open Problem 11.9: Building realistic multi-agent environments for Contextual Integrity.

How can we simulate realistic agent interactions (where, e.g., agents may have access to a user’s data or act on behalf of a business) in order to surface complex information flows and contextual norms, and enable both evaluation and training of contextually private and secure LLMs?

Such simulated and formally specified environments provide a natural substrate for studying Contextual Integrity at the population level: appropriate and inappropriate information flows can be encoded in scenarios or problem instances, and tendencies to leak or over-share can be probed systematically, extending Contextual Integrity benchmarks built for single models (Tamber et al., 2026) to the multi-agent regime (Gupta et al., 2026). Scalable, controllable environments of this kind could offer just what is needed for measuring Contextual Integrity in multi-agent settings, where the space of contexts and inter-agent information flows is far too large to audit manually. Formal environments additionally could make it possible to vary these factors systematically and to evaluate agent behavior against known system-level objectives and constraints.

Addressing the challenges described above would greatly benefit from benchmarks that move beyond single-agent, single-flow evaluation toward system-level privacy assessment. This includes constructing multi-agent evaluation environments where information provenance can be traced end-to-end across delegation bound-

aries; defining compositional privacy metrics that quantify aggregate leakage across individually compliant flows; and red-teaming methodologies specifically designed for multi-agent ecosystems, including adversarial agents that exploit delegation protocols to reconstruct user profiles through long-term observation of inter-agent communication patterns. These benchmarks would measure both verbatim factual disclosures and how adversarial agents may infer attributes via behavioral leakage.

12 Governance of Agentic Privacy and Security

To meet the serious and unprecedented privacy and security threats posed by AI agents, this manuscript has proposed a contextual approach and has adopted the theory of Contextual Integrity as the underlying framework. It has outlined the conceptual framework and has laid the foundations and identified the open problems for operationalizing and enforcing it. To promote the chances of real-world, practical adoption of contextual approaches to privacy and security, sound governance is essential. Ideally, the collaborative efforts of actors in governmental, corporate, academic, public interest, and technical standards settings would yield smart, well-calibrated societal policies (Nissenbaum, 2011). Proposing specific policies is beyond the scope of this manuscript. Instead, this section identifies the open governance and policy problems that must be addressed to enable contextual privacy and security.

We organize this around three questions. First, what preconditions would create an environment conducive to contextual privacy and security? Second, how should the CI norms relevant to agent deployment and increased use of agents be sourced, and what conflicting preferences must be balanced for a norm to settle (§12.1)? Third, how are norms enforced, and who are the relevant enforcement actors (§12.2)?

Open Problem 12.1: Achieving environments conducive to contextual privacy and security.

Achieving an environment conducive to contextual privacy and security requires progress on several interdependent aims:

1. How can we incentivize developers of agentic systems to learn and adhere to existing contextual norms, as opposed to treating privacy and security as a compliance afterthought?
2. How should the concept of a “reasonable expectation of privacy,” historically developed around the relationship between governmental entities and private actors, be extended to the far wider range of relationships agents mediate?
3. How can processes and systems be made flexible enough to absorb contextual wisdom about ends, purposes, and values from experts embedded in the relevant social domains, and to identify systematic relationships between data flows or state changes and those contextual ends?
4. What are workable methods for two hard cases: articulating contextual roles for novel actors such as tech companies and agents themselves, and adjudicating actions that serve one context while disserving another?

The preconditions above identify what an environment conducive to contextual privacy and security requires. Before turning to what the substantive norms governing agentic information flows should be, we address a prior governance-design question: *How should responsibility be distributed across the agentic ecosystem so that these preconditions can actually take hold?* One core logic to follow is that obligations ought to be assigned to actors in line with their capability, vantage point, and exposure to manage the risk. This redistributes harms and benefits in a way that enables users to trust and adopt a technology, while the responsible party retains flexibility in how it meets the obligation.

There are several historical examples of how similar questions have been addressed. For instance, in the early days of credit cards, a major barrier to public uptake was consumer fear of uncapped liability related to credit card fraud. The Fair Credit Billing Act (1974) (Littlefield, 1975) capped consumer exposure at 50 USD and shifted the residual cost of fraud to card issuers. The key purpose was to build the public trust necessary for mass adoption. Additionally, because issuers now bore the cost, they invested heavily in

fraud-detection and prevention technologies without legislation prescribing what those technologies should be.

Another example comes from the Americans with Disabilities Act (1990) (Burgdorf Jr., 1991), where building codes set performance-level floors (e.g., a structure must withstand a given load, a public space must be accessible) but did not prescribe the use of specific materials or layouts. The effect is to enable broad public participation and access by assigning the burden of meeting the standard to the entity that controls the relevant design decisions, not to the individuals who must rely on the built environment (Burgdorf Jr., 1991; Murdoch & Jawer, 1992).

In both cases, concentrating responsibility on capable actors generated investment, trust, and adoption without mandating the technical form of the solution. A comparable logic could guide agentic governance. We broadly conceptualize agentic governance as the rules, practices, processes, and technological tools that inform the development and deployment of agents. A CI-specific understanding further focuses on how these rules govern and incentivize appropriate flows of information and behaviors within the agentic ecosystem. With the preconditions and the distribution of responsibility in place, the following sections discuss what the norms should be, and how these norms should be enforced.

12.1 What should the norms be?

Norms, in the CI framework, reflect a settled accommodation of contextual values, ends, and purposes. They express what a given community has come to regard as appropriate flows of information and behaviors within a particular social domain. This leads to two important open questions:

Open Problem 12.2: Determining CI norms.

First, what should be the practices and processes for sourcing the set of norms that agents should abide by? Second, how do we devise governance mechanisms for formally updating norms that may have become suboptimal in supporting contextual values, ends, and purposes?

We see at least three approaches to the first question:

The first is a baseline of standards established top-down by companies and governments. Just as building codes and accessibility mandates set performance-level floors for the built environment (see above), platform providers and regulators can articulate minimum requirements for how agentic systems handle the flow of information or actuation of state changes within a given context. Such baselines also need to address a transparency problem specific to agentic systems. When an AI agent acts on behalf of a user, the vendor may or may not be a recipient of the third party’s data, depending on how the system is built (a vendor running an agent inside a trusted execution environment or on the user’s device need not see the data at all). The third party generally cannot tell which is the case.

The second route is empirical. There is a need for systematic efforts to discover existing contextual norms as they actually operate in practice and the real world. What do people in specific social domains already expect, rely on, and regard as appropriate? One key point here is that this is not arbitrary social science work. In order to be actionable and useful for governance, such empirical research efforts need to be forward-looking, aligned with the technologies on the horizon, and focused on the settings that agentic systems are about to enter. One concrete step to collect relevant insights in a centralized and dynamically expanding place would be the development of a public norm repository. Such a norm repository would take the form of a structured catalog of context-specific informational norms that agentic systems could query and respect. Such a resource could also serve as shared infrastructure for both developers seeking to build norm-aware systems and regulators seeking to ground policy in empirical reality.⁹

This is undoubtedly as much an empirical question as a doctrinal one. CI’s approach to norm-sourcing does something structurally similar to the Fourth Amendment’s “reasonable expectation of privacy” test by asking

⁹Once again, we can learn from historical precedent. Specifically, the legal system already performs a version of this sourcing work. The Fourth Amendment’s “reasonable expectation of privacy” test, established in *Katz v. United States* (1967) (Supreme Court of the United States, 1967; Winn, 2009), asks what expectations society legitimizes as reasonable.

what information flows people regard as appropriate within a given context. The difference is one of scope. The reasonable-expectation test was historically developed around the relationship between governmental entities and private actors. As noted in our preconditions, this thinking needs to be extended to the far wider range of relationships that agents mediate, ranging from relationships between users and service providers, to those between users and third parties and between the multiple principals whose interests converge in a single agentic interaction (Winn, 2009; Supreme Court of the United States, 1967; Nissenbaum, 2009).

A third option is to develop AI systems capable of performing CI’s decision heuristic process for identifying legitimate norms given contextual details. Rather than taking norms as either stipulated or empirically given, AI agents would make carefully contextualized decisions on the fly about what norms—entrenched or otherwise—will support ends, purposes, and values (see Benthall & Cummings (2026)). AI agents would consider fixed or architectural aspects of their system context, as well as observations of user behavior (see Criado & Such (2015)). In some cases, AI agents may need to communicate with each other to establish norms dynamically. This capability may prove especially critical when agentic AI systems serving different principals interact in a rapidly changing digital environment. In these settings, AI agents cannot wait for norms to become settled institutional or empirical facts; they may need to anticipate what users or policymakers will decide is legitimate.

Even where norms have been identified through any of the above routes, a further open problem arises:

Open Problem 12.3: Institutional and legal governance—resolving conflicting norms.

Autonomous agents are designed to move seamlessly across cultural, geographic, and organizational boundaries. The result is a highly fragmented compliance landscape. An agent that books travel, manages health records, and negotiates contracts may, in a single session, be subject to the data protection regimes of multiple jurisdictions with divergent and sometimes conflicting requirements. For developers of agentic systems, this is not an edge case but a routine feature of how agents operate.

One approach worth exploring is the systematic mapping of localized regulatory requirements and privacy and security norms. Such a mapping could serve as a prerequisite for dynamic legal and cultural alignment, allowing an agent to adapt its information-handling practices and behaviors to the jurisdiction and context in which it is operating at any given moment. Another direction might pursue training agents to navigate overlapping jurisdictions by adopting conflict-of-laws principles from private international law (Kohl, 2007; Berman, 2012; Svantesson, 2017).

12.2 How do we verify and provide accountability for norms enforcement?

The previous section addressed what the norms governing agentic information flows and behaviors should be, and where they come from. Earlier sections of this manuscript (e.g., §6) discussed how agents themselves can be designed to operationalize and self-enforce contextual norms at runtime. Governance poses an equally important and distinct challenge in practice:

Open Problem 12.4: Incentivizing and verifying normative compliance.

How do we scalably verify that agents did, in fact, follow contextual norms, and hold the relevant actors accountable should they not?

Contextual norms are not enforced by any single authority. Different social forces maintain them, and each brings its own mechanisms. Courts enforce norms through adjudication and liability. Regulators enforce through statutory compliance and penalties. Companies enforce through internal policy and platform governance. Professional bodies, including the domain experts discussed above, enforce through standards and sanctions. In practice, these forces operate together, and any given context’s norms are typically sustained by some combination of them. We organize the discussion around two broad categories: (1) institutional and legal governance, and (2) governance through technology.

Institutional and legal mechanisms are necessary but not sufficient. Agentic systems act at a speed and level of autonomy that outpaces traditional ex-post regulatory intervention. Moreover, effective enforcement often requires reporting of personal information, which is private, to some investigatory or enforcing party. This information flow is itself subject to information norms that can be violated. Consider, for example, the ways that improper gathering of evidence without a warrant can lead to a mistrial in a court of law. By the time a court adjudicates or a regulator investigates, the information flow has already occurred and the harm may be irreversible. Enforcement therefore cannot rely entirely on after-the-fact correction. It must also be built into the systems themselves.

A norm repository can be one component of this accountability infrastructure. Beyond a resource that agents query at runtime, it provides a benchmark against which their information-handling decisions can be audited after the fact. However, investigatory and enforcement decisions may happen at machine speed and in novel scenarios, suggesting a need for AI agents to also be capable of on-the-fly evaluation of CI norms when static guidance is incomplete.

Formal models of context and normative reasoning not only enable runtime tuning of agents' behavior, but they also create a record of which norms were applied to a given flow, and for what reason. This inspectable record moves governance through technology beyond a design technique and toward an accountability mechanism. It renders an agent's information-handling decisions auditable, both individually and in cross-agent interactions where responsibility is otherwise hard to trace, making it possible to determine after the fact which norms were applied, by whom, and whether they were appropriate. This would mean that both enforcement and production of an auditable record of compliance occur at the moment the agent acts, rather than being reconstructed months later through a regulatory investigation. Legal and regulatory mechanisms are essential for setting the norms themselves and for accountability when systems fail.

13 Conclusion

This manuscript outlined privacy and security research challenges that arise with highly capable and autonomous AI agents. We began by introducing and justifying a perspective on agentic privacy and security through the lens of Contextual Integrity theory. We then highlighted open questions towards building agents that act appropriately across agent component levels, including system-level enforcement abstractions, model-level capabilities, and user interaction strategies. Finally, we outlined problems that emerge when deploying agents in complex environments such as evaluations, interacting in multi-agent settings, and governance of behavioral norms. We hope that this manuscript inspires the broader research community across academia, government, and industry to collaborate to develop the contextual foundations for a safe, secure, and privacy-respecting agentic ecosystem.

Acknowledgments

We would like to acknowledge inspiring workshop talks and thoughtful suggestions from Corinna Cortes, Pushmeet Kohli, and Yossi Matias. We thank Albert Cheu, Matthew Jagielski, Srinivas Jantali, Ben Kamber, Milad Nasr, and Royce Wilson for insightful workshop discussion. Finally, we want to acknowledge support by Greg Ganger, Hank Levy, James Manyika, Brendan McMahan, Raluca Ada Popa, and Amanda Walker.

14 List of Discussed Open Problems

Section 5: Contextual Security: Security that Upholds Contextual Integrity

OP 5.1: Identifying contextual behavior norms.	19
OP 5.2: Determining reasonable and accurate threat models.	20
OP 5.3: Agreeing upon contextual security norms.	21

Section 6: Operationalizing Contextual Integrity for Agents

OP 6.1: Balancing reliance on users vs. AI models.	23
OP 6.2: AI models that understand contextual norms.	24
OP 6.3: Representing contextual norms.	25
OP 6.4: Formal policies that capture norms expressed in natural language.	26
OP 6.5: Preventing harmful context from affecting policy generation.	27
OP 6.6: Establishing precedence between conflicting policies.	28
OP 6.7: Practical enforcement mechanisms to ensure appropriate behavior.	29

Section 7: Challenges with Systems Security and Privacy for AI Agents

OP 7.1: Distinguishing agents from humans.	31
OP 7.2: Tracing agents back to users.	32
OP 7.3: Enabling context-aware authorization.	33
OP 7.4: Managing the authorization lifecycle.	34
OP 7.5: Ensuring that an agent's action is intended by the user.	35
OP 7.6: Evolving PII detection and replacement.	36
OP 7.7: Achieving data minimization.	36
OP 7.8: Moving from data deletion to functional data revocation.	37
OP 7.9: Protecting user anonymity in an agentic world.	38
OP 7.10: Closing the gap between developer intent and application specification.	41
OP 7.11: Making agent execution as robust as traditional software.	43

Section 8: Model-Level Security and Privacy Reasoning

OP 8.1: Teaching models privacy and security reasoning.	44
OP 8.2: Handling uncertainty and ambiguity in unstructured domains.	46
OP 8.3: Detecting stale assumptions in changing contexts.	47
OP 8.4: Interpretable use of private information.	49

Section 9: User-Level Issues in Privacy- and Security-Aware AI Agents

OP 9.1: Supporting user mental models for AI agents.	50
OP 9.2: Thinking beyond notice and choice for AI agents.	53
OP 9.3: Determining appropriateness of AI agent actions.	53
OP 9.4: Enabling user supervision over automation.	55
OP 9.5: Delaying, stopping, and reversing AI agent actions.	58
OP 9.6: Accountability frameworks for AI agents.	58

Section 10: Evaluations and Benchmarking

OP 10.1: Advancing automated red-teaming on saturated benchmarks.	61
OP 10.2: Designing and implementing large-scale studies.	62
OP 10.3: Evaluating how agents persist and use historical system context.	63
OP 10.4: Evaluating leakage in aggregated data flows.	64
OP 10.5: Benchmarking dynamic privacy-utility trade-offs.	64
OP 10.6: Evaluations that carry across environments.	64
OP 10.7: Evaluations of agents in evolving environments.	65

Section 11: Towards Trustworthy Multi-Agent Ecosystems

OP 11.1: Establishing shared contextual norms.	66
OP 11.2: Preventing collusion and drifting norms between agents.	66
OP 11.3: Preserving constraints during delegation or parallel execution.	67
OP 11.4: Orchestrating large agentic swarms and enforcing normative behavior.	68
OP 11.5: Ensuring composed behaviors across agents uphold privacy and security.	68
OP 11.6: Developing secure and robust multi-agent communication protocols.	69
OP 11.7: Building trust across agent boundaries with attestation.	69
OP 11.8: Balancing security and utility in multi-agent communications.	70
OP 11.9: Building realistic multi-agent environments for Contextual Integrity.	71

Section 12: Governance of Agentic Privacy and Security

OP 12.1: Achieving environments conducive to contextual privacy and security.	72
OP 12.2: Determining CI norms.	73
OP 12.3: Institutional and legal governance—resolving conflicting norms.	74
OP 12.4: Incentivizing and verifying normative compliance.	74

References

- A2A Project. Agent2Agent (A2A) Protocol, 2026. URL <https://github.com/a2aproject/A2A>.
- Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *CCS*, 2016.
- Sahar Abdelnabi and Eugene Bagdasarian. AI agents may always fall for prompt injections. In *NeurIPS*, 2026.
- Sahar Abdelnabi and Ahmed Salem. The Hawthorne Effect in Reasoning Models: Evaluating and Steering Test Awareness. In *NeurIPS*, 2025.
- Sahar Abdelnabi, Amr Gomaa, Eugene Bagdasarian, Per Ola Kristensson, and Reza Shokri. Firewalls to Secure Dynamic LLM Agentic Networks. In *TMLR*, 2026.
- Sina Abdollahi, Mohammad M Maheri, Javad Forough, Amir Al Sadi, Josh Millar, David Kotz, Marios Kogias, and Hamed Haddadi. AgentTEE: Confidential LLM agent execution on edge devices. In *EuroMLSys*, 2026.
- Mark Ackerman, Trevor Darrell, and Daniel J Weitzner. Privacy in context. *Human-Computer Interaction*, 2001.
- Divyansh Agarwal, Alexander Richard Fabbri, Ben Risher, Philippe Laban, Shafiq Joty, and Chien-Sheng Wu. Prompt leakage effect and mitigation strategies for multi-turn LLM applications. In *EMNLP*, 2024.
- Agent Payments Protocol Working Group. Agent Payments Protocol (AP2) Specification, 2025. URL <https://agentpaymentsprotocol.info/>.
- AGNTCY. Building Infrastructure for the Internet of Agents, 2026. URL <https://agntcy.org/>.
- Ian Ahl. Inside the OpenClaw Ecosystem: What Happens When AI Agents Get Credentials to Everything. *Permiso*, 2026.
- Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *ICLR*, 2017.
- Amazon Web Services. Introducing Dogwood: runtime verification for AI agents, August 2026. URL <https://aws.amazon.com/blogs/opensource/introducing-dogwood-runtime-verification-for-ai-agents/>. Published August 6, 2026.
- American Association of Motor Vehicle Administrators (AAMVA). Mobile Driver’s License (mDL) Implementation Guidelines, Version 1.5. Technical report, AAMVA, 2025.
- Bonnie B. Anderson, Anthony Vance, C. Brock Kirwan, Jeffrey L. Jenkins, and David Eargle. From Warning to Wallpaper: Why the Brain Habituates to Security Warnings and What Can Be Done about It. *Journal of Management Information Systems*, 2016.
- Bonnie Brinton Anderson, C. Brock Kirwan, Jeffrey L. Jenkins, David Eargle, Seth Howard, and Anthony Vance. How Polymorphic Warnings Reduce Habituation in the Brain: Insights from an fMRI Study. In *CHI*, 2015.
- 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. *Preprint arXiv:2410.09024*, 2025.
- Anthropic. Introducing the Model Context Protocol. <https://www.anthropic.com/news/model-context-protocol>, November 2024. Accessed: 2026-07-21.

- Anthropic. Claude Code, 2025. URL <https://github.com/anthropics/claude-code>. Accessed: 2026-08-17.
- Anthropic. Investigating three real-world incidents in our cybersecurity evaluations. Anthropic News, July 2026a. Accessed: 2026-08-05.
- Anthropic. Claude Mythos Preview System Card. Technical report, Anthropic, April 2026b.
- Anthropic. Introducing Claude Sonnet 5. <https://www.anthropic.com/news/claude-sonnet-5>, June 2026c. Accessed: *October 1, 2026*.
- Noah Apthorpe, Yan Shvartzshnaider, Arunesh Mathur, Dillon Reisman, and Nick Feamster. Discovering smart home internet of things privacy norms using contextual integrity. *IMWUT*, 2018.
- Ariel Flint Ashery, Luca Maria Aiello, and Andrea Baronchelli. Emergent social conventions and collective bias in LLM populations. *Science Advances*, 11(20):eadu9368, 2025.
- Tim Baarslag, Alper T. Alan, Richard Gomer, Muddasser Alam, Charith Perera, Enrico H. Gerding, and m.c. schraefel. An automated negotiation agent for permission management. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '17, pp. 380–390, Richland, SC, 2017. International Foundation for Autonomous Agents and Multiagent Systems.
- Eugene Bagdasarian, Griffin Berstein, Jason Waterman, Eleanor Birrell, Nate Foster, Fred B Schneider, and Deborah Estrin. Ancile: Enhancing privacy for ubiquitous computing with use-based privacy. In *WPES*, 2019.
- Eugene Bagdasarian, Ren Yi, Sahra Ghalebikesabi, Peter Kairouz, Marco Gruteser, Sewoong Oh, Borja Balle, and Daniel Ramage. AirGapAgent: Protecting privacy-conscious conversational agents. In *CCS*, 2024.
- Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, et al. Constitutional AI: Harmlessness from AI Feedback. *Preprint arXiv:2212.08073*, 2022.
- Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael Lan, Vincent Wang, Noah Siegel, Nicolas Collignon, Clement Neo, Isabelle Lee, Alasdair Paren, et al. Chain-of-thought is not explainability. *Preprint*, 2025.
- Louise Barkhuus. The mismeasurement of privacy: using contextual integrity to reconsider privacy in HCI. In *CHI*, 2012.
- Solon Barocas and Karen Levy. Privacy Dependencies. *Washington Law Review*, 95(2):555–617, 2020.
- Solon Barocas and Helen Nissenbaum. Big data’s end run around anonymity and consent. *Privacy, big data, and the public good: Frameworks for engagement*, 1:44–75, 2014.
- Adam Barth. The Web Origin Concept. RFC 6454, RFC Editor, 2011.
- Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.
- Brinnae Bent. The Term ‘Agent’ Has Been Diluted Beyond Utility and Requires Redefinition. In *AIES*, 2025.
- Sebastian Benthall. Situated information flow theory. In *Proceedings of the 6th Annual Symposium on Hot Topics in the Science of Security*, HotSoS '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450371476. doi: 10.1145/3314058.3314066. URL <https://doi.org/10.1145/3314058.3314066>.
- Sebastian Benthall and Rachel Cummings. Integrating differential privacy and contextual integrity. In *CSLAW*, 2024.

- Sebastian Benthall and Rachel Cummings. Principled Differential Privacy Parameter Tuning with Causal Games. In *FAccT*, 2026.
- Sebastian Benthall, Seda Gürses, and Helen Nissenbaum. Contextual integrity through the lens of computer science. *Foundations and Trends® in Privacy and Security*, 2(1):1–69, 2017.
- Paul Schiff Berman. *Global legal pluralism: A jurisprudence of law beyond borders*. Cambridge University Press, 2012.
- Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The Complexity of Decentralized Control of Markov Decision Processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.
- Luca Beurer-Kellner, Beat Buesser, Ana-Maria Crețu, Edoardo Debenedetti, Daniel Dobos, Daniel Fabian, Marc Fischer, David Froelicher, Kathrin Grosse, et al. Design Patterns for Securing LLM Agents against Prompt Injections. *Preprint arXiv:2506.08837*, 2025.
- Binary Transparency, 2021. Binary Transparency. <https://binary.transparency.dev/>, 2021. Accessed: June 17, 2026.
- Arnar Birgisson, Joe Gibbs Politz, Ulfar Erlingsson, Ankur Taly, Michael Vrabie, and Mark Lentczner. Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud. In *NDSS*, 2014.
- Eleanor Birrell, Anders Gjerdrum, Robbert van Renesse, Håvard Johansen, Dag Johansen, and Fred B Schneider. SGX enforcement of use-based privacy. In *WPES*, 2018.
- Matt Blaze, Joan Feigenbaum, and Jack Lacy. Decentralized trust management. In *SE&P*, 1996.
- Teofil Bodea, Masanori Misono, Julian Pritzi, Patrick Sabanic, Thore Sommer, Harshavardhan Unnibhavi, David Schall, Nuno Santos, Dimitrios Stavrakakis, and Pramod Bhatotia. Trusted AI Agents in the Cloud. *Preprint arXiv:2512.05951*, 2025.
- Harold Booth, Bill Fisher, Ryan Galluzzo, and Joshua Roberts. Accelerating the Adoption of Software and AI Agent Identity and Authorization. Concept paper, National Institute of Standards and Technology, 2026.
- Julie Bort. A Meta AI security researcher said an OpenClaw agent ran amok on her inbox. *TechCrunch*, 2026.
- Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermy, Tom Conerly, Nick Turner, et al. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. *Transformer Circuits*, 2023.
- Natalie Grace Brigham, Eugene Bagdasarian, Tadayoshi Kohno, and Franziska Roesner. Janus: a Playground for User-Involved Agentic Permission Management. In *SE&P*, 2027.
- Robert L Burgdorf Jr. The Americans with Disabilities Act: Analysis and implications of a second-generation civil rights statute. *Harv. CR-CLL Rev.*, 26:413, 1991.
- Kevin Butler, Kurt Hugenberg, Eakta Jain, Apu Kapadia, Tadayoshi Kohno, Elissa M. Redmiles, Franziska Roesner, Mattea Sim, Patrick Traynor, and Hanna Barakat. Extending the heilmeier catechism to evaluate security and privacy systems: Who is left out? *IEEE Security & Privacy (Magazine)*, 23(3):91–95, 2025. doi: 10.1109/MSEC.2025.3554848.
- Marc E. Canby, Adam Davies, Chirag Rastogi, and Julia Hockenmaier. How reliable are causal probing interventions? In *AAACL*, 2025.
- Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting Training Data from Large Language Models. In *USENIX Security*, 2021.

- Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning Web-Scale Training Datasets is Practical. In *SECP*, 2024.
- Juan-Carlos Carrillo, Jose Luis Martin-Navarro, Rongjun Ma, and Jose Such. Personal Data Flows and Privacy Policy Traceability in Third-party LLM Apps in the GPT Ecosystem. *Proceedings on Privacy Enhancing Technologies*, 2026(1):273–295, 2026. doi: 10.56553/popets-2026-0015.
- CCPA. California Consumer Privacy Act of 2018. Cal. Civ. Code §§ 1798.100–1798.199.100, 2018. URL <https://coppa.ca.gov>. California Consumer Privacy Act as amended.
- Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jail-breaking black box large language models in twenty queries. In *SaTML*, 2025.
- {Bing Jyue} Chen, Suppakit Waiwitlikhit, Ion Stoica, and Daniel Kang. "ZKML: An Optimizing System for ML Inference in Zero-Knowledge Proofs". In *EuroSys*, pp. 560–574, United States, April 2024. Association for Computing Machinery. doi: 10.1145/3627703.3650088. Publisher Copyright: © 2024 ACM.; 19th European Conference on Computer Systems, EuroSys 2024 ; Conference date: 22-04-2024 Through 25-04-2024.
- Jiaqi Chen, Yanzhe Zhang, Yutong Zhang, Yijia Shao, and Diyi Yang. Generative Interfaces for Language Models. In *ACL*, 2026.
- Yuhan Cheng, Hancheng Ye, Hai Helen Li, Jingwei Sun, and Yiran Chen. PrivAct: Internalizing Contextual Privacy Preservation via Multi-Agent Preference Training. In *ICML*, 2026.
- Zhao Cheng, Diane Wan, Matthew Abueg, Sahra Ghalebikesabi, Ren Yi, Eugene Bagdasarian, Borja Balle, Stefan Mellem, and Shawn O'Banion. CI-Bench: Benchmarking Contextual Integrity of AI Assistants on Synthetic Data. *Preprint arXiv:2409.13903*, 2024.
- Hanbyul Choi, Jonghwa Park, and Yoonhyuk Jung. The role of privacy fatigue in online privacy behavior. *Computers in Human Behavior*, 2018.
- Vivek Chopra. OAuth Is Not Enough: Authorization Challenges for Autonomous AI Agents. Technical report, TechRxiv, 2025.
- Mihai Christodorescu, Earlene Fernandes, Ashish Hooda, Somesh Jha, Johann Rehberger, Kamalika Chaudhuri, Xiaohan Fu, Khawaja Shams, Guy Amir, et al. Systems Security Foundations for Agentic Computing. *arXiv preprint arXiv:2512.01295*, 2025.
- Jalen Chuang, Alex Seto, Nicolas Berrios, Stephan van Schaik, Christina Garman, and Daniel Genkin. TEE.fail: Breaking trusted execution environments via DDR5 memory bus interposition. In *SECP*, 2026.
- Cloud Native Computing Foundation. Open Policy Agent. <https://www.openpolicyagent.org>, 2026a. Accessed: 2026-05-28.
- Cloud Native Computing Foundation. SPIFFE: Secure Production Identity Framework for Everyone, 2026b. URL <https://spiffe.io/>.
- Confidential Computing Consortium. A Technical Analysis of Confidential Computing. Technical report, Linux Foundation, 2022.
- Amanda Conley, Anupam Datta, Helen Nissenbaum, and Divya Sharma. Sustaining privacy and open justice in the transition to online court records: A multidisciplinary inquiry. *Md. L. Rev.*, 71:772, 2011.
- Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. *NeurIPS*, 2023.
- A. Feder Cooper, Mark A. Lemley, Allison Casasola, Ahmed Ahmed, Aaron Gokaslan, Amy B. Cyphert, Christopher De Sa, Daniel E. Ho, and Percy Liang. Extracting memorized pieces of (copyrighted) books from open-weight language models. In *COLM*, 2026.

- Manuel Costa, Boris Köpf, Aashish Kolluri, Andrew Paverd, Mark Russinovich, Ahmed Salem, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Beguelin. Securing AI Agents with Information-Flow Control. *Preprint arXiv:2505.23643*, 2025.
- Natalia Criado and Jose M Such. Implicit contextual integrity in online social networks. *Information Sciences*, 325:48–69, 2015.
- Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *Preprint arXiv:2309.08600*, 2023.
- Hoagy Cunningham, Jerry Wei, Zihan Wang, Andrew Persic, Alwin Peng, Jordan Abderrachid, Raj Agarwal, et al. Constitutional Classifiers++: Efficient Production-Grade Defenses against Universal Jailbreaks. *Preprint arXiv:2601.04603*, 2026.
- Joseph W. Cutler, Craig Disselkoen, Aaron Eline, Shaobo He, Kyle Headley, Michael Hicks, Kesha Hietala, Eleftherios Ioannidis, John Kastner, Anwar Mamat, Darin McAdams, Matt McCutchen, Neha Rungta, Emina Torlak, and Andrew Wells. Cedar: A New Language for Expressive, Fast, Safe, and Analyzable Authorization. *PACMPL*, 2024.
- Saswat Das and Ferdinando Fioretto. NeuroFilter: Activation-Based Guardrails for Privacy-Conscious LLM Agents. *CoRR*, abs/2601.14660, 2026. doi: 10.48550/arXiv.2601.14660.
- Saswat Das, Jameson Sandler, and Ferdinando Fioretto. Beyond Jailbreaking: Auditing Contextual Privacy in LLM Agents. *Preprint arXiv:2506.10171*, 2025.
- Saswat Das, Parvati Viswanathan, Daniel Donnelly, Chang Huang, Sahar Abdelnabi, and Ferdinando Fioretto. SEABench: Benchmarking Endogenous Misalignment In Self-Evolving Agents. *Preprint arXiv:2609.35596*, 2026.
- Adam Davies, Jize Jiang, and ChengXiang Zhai. Competence-based analysis of language models. *Preprint arXiv:2303.00333*, 2023.
- C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, and D. Spence. Generic AAA Architecture. RFC 2903, August 2000. URL <https://www.rfc-editor.org/info/rfc2903/>.
- Christian Schroeder de Witt, Klaudia Krawiecka, Igor Krawczuk, Ben Hagag, William L Anderson, Peter Belcak, Ben Bucknall, Xiaohong Cai, Ayush Chopra, Doron Cohen, et al. Open challenges in multi-agent security: Towards secure systems of interacting AI agents. *Preprint arXiv:2505.02077*, 2025.
- Edoardo Debenedetti, Javier Rando, Daniel Paleka, Fineas Silaghi, Dragos Albastroiu, Niv Cohen, Yuval Lemberg, Reshmi Ghosh, Rui Wen, Ahmed Salem, et al. Dataset and Lessons Learned from the 2024 SaTML LLM Capture-the-Flag Competition. In *NeurIPS*, 2024a.
- Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *NeurIPS*, 2024b.
- Edoardo Debenedetti, Ilia Shumailov, Tianqi Fan, Jamie Hayes, Nicholas Carlini, Daniel Fabian, Christoph Kern, Chongyang Shi, Andreas Terzis, and Florian Tramèr. Defeating Prompt Injections by Design. *Preprint arXiv:2503.18813*, 2025.
- Tamara Denning, Zakariya Dehlawi, and Tadayoshi Kohno. In situ with bystanders of augmented reality glasses: perspectives on recording and privacy-mediating technologies. In *CHI*, 2014.
- Morton Deutsch. A Theory of Co-operation and Competition. *Human Relations*, 2(2):129–152, 1949. doi: 10.1177/001872674900200204.
- Robert Lee Dickens. Finding Common Ground in the World of Electronic Contracts: The Consistency of Legal Reasoning in Clickwrap Cases. *Marquette Intellectual Property Law Review*, 11(2):379–410, 2007.

- Paul Dourish. What we talk about when we talk about context. *Personal and ubiquitous computing*, 2004.
- Aisha Down. Hundreds of English-language websites link to pro-Kremlin propaganda. *The Guardian*, November 2025.
- Vasisht Duddu, Oskari Järvinen, Lachlan J Gunn, and N Asokan. Laminator: Verifiable ML Property Cards using Hardware-assisted Attestations. In *Proceedings of the Fifteenth ACM Conference on Data and Application Security and Privacy*, pp. 317–328, 2025.
- Cynthia Dwork. Differential privacy: A survey of results. In *TAMC*, 2008.
- Serge Egelman, Lorrie Faith Cranor, and Jason Hong. You’ve Been Warned: An Empirical Study of the Effectiveness of Web Browser Phishing Warnings. In *SIGCHI Conference on Human Factors in Computing Systems*, 2008.
- Abul Ehtesham, Aditi Singh, Gaurav Kumar Gupta, and Saket Kumar. A survey of agent interoperability protocols: Model context protocol (MCP), agent communication protocol (ACP), agent-to-agent protocol (A2A), and agent network protocol (ANP). *Preprint arXiv:2505.02279*, 2025.
- Ronen Eldan and Mark Russinovich. Who’s Harry Potter? Approximate Unlearning in LLMs. *Preprint arXiv:2310.02238*, 2023.
- Assaf Elovic. GPT Researcher, 2026. URL <https://github.com/assafelovic/gpt-researcher>. Accessed: 2026-08-18.
- European Commission. EU Digital Identity Wallet Home, 2026.
- European Parliament and Council of the European Union. Directive (EU) 2015/2366 on payment services in the internal market, amending Directives 2002/65/EC, 2009/110/EC and 2013/36/EU and Regulation (EU) No 1093/2010, and repealing Directive 2007/64/EC (PSD2). Official Journal of the European Union, L 337/35, 2015. Accessed: 2026-08-18.
- Wei Fan, Haoran Li, Zheyang Deng, Weiqi Wang, and Yangqiu Song. Goldcoin: Grounding large language models in privacy laws via contextual integrity theory. In *EMNLP*, 2024.
- Kassem Fawaz, Ren Yi, Octavian Suci, Rishabh Khandelwal, Hamza Harkous, Nina Taft, and Marco Gruteser. Text-Based Personas for Simulating User Privacy Decisions. *Preprint arXiv:2603.19791*, 2026.
- Yair Feldman, Linxi Zhao, Nathan Godey, Dongyoung Go, Yilun Hua, Kilian Q. Weinberger, Jennifer J. Sun, and Yoav Artzi. Co-LMLM: Continuous-Query Limited Memory Language Models, July 2026.
- Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. Android permissions: user attention, comprehension, and behavior. In *SOUPS*, 2012.
- Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. Distributed Constraint Optimization Problems and Applications: A Survey. *Journal of Artificial Intelligence Research*, 61:623–698, 2018.
- James Flemings, Ren Yi, Octavian Suci, Kassem Fawaz, Murali Annavaram, and Marco Gruteser. Personalizing Agent Privacy Decisions via Logical Entailment. In *PETS*, 2026.
- Hanna Foerster, Tom Blanchard, Kristina Nikolić, Ilia Shumailov, Cheng Zhang, Robert Mullins, Nicolas Papernot, Florian Tramèr, and Yiren Zhao. CaMeLs Can Use Computers Too: System-level Security for Computer Use Agents. *Preprint arXiv:2601.09923*, 2026.
- Javad Forough, Marios Kogias, and Hamed Haddadi. When Agents Handle Secrets: A Survey of Confidential Computing for Agentic AI. *Preprint arXiv:2605.03213*, 2026.
- Matt Franchi, Madiha Zahrah Choksi, Harold Triedman, and Helen Nissenbaum. Context-based normative simulacra from narrative fiction. In *Third Conference on Language Modeling*, 2026.

- Diana Freed, Jackeline Palmer, Diana Minchala, Karen Levy, Thomas Ristenpart, and Nicola Dell. “A Stalker’s Paradise”: How Intimate Partner Abusers Exploit Technology. In *CHI*, 2018.
- Brett M Frischmann, Michael J Madison, and Katherine Jo Strandburg. *Governing knowledge commons*. Oxford University Press, 2014.
- Front. Customer Service Solutions, 2026. URL <https://front.com/lp2/customer-service-new>. Accessed: 2026-08-18.
- Tingchen Fu, Mrinank Sharma, Philip Torr, Shay B. Cohen, David Krueger, and Fazl Barez. PoisonBench: Assessing Large Language Model Vulnerability to Data Poisoning. In *ICML*, 2025.
- Prakhar Ganesh, Cuong Tran, Reza Shokri, and Ferdinando Fioretto. The Data Minimization Principle in Machine Learning. In *ACM Conference on Fairness, Accountability, and Transparency (ACM FAccT)*, 2025.
- Jennifer Garnett. Google encrypts its network to counteract NSA surveillance. *Harvard Journal of Law & Technology*, 2013.
- Sandro Rodriguez Garzon, Awid Vaziry, Enis Mert Kuzu, Dennis Enrique Gehrman, Buse Varkan, Alexander Gaballa, and Axel Küpper. AI agents with decentralized identifiers and verifiable credentials. *Preprint arXiv:2511.02841*, 2025.
- Barton Gellman and Ashkan Soltani. NSA infiltrates links to Yahoo, Google data centers worldwide, Snowden documents say. *The Washington Post*, 2013.
- Sahra Ghalebikesabi, Eugene Bagdasarian, Ren Yi, Itay Yona, Ilia Shumailov, Aneesh Pappu, Chongyang Shi, Laura Weidinger, Robert Stanforth, Leonard Berrada, et al. Privacy awareness for information-sharing assistants: A case-study on form-filling with contextual integrity. *Transactions on Machine Learning Research*, 2025.
- Amr Gomaa, Ahmed Salem, and Sahar Abdelnabi. Converse: Benchmarking contextual safety in agent-to-agent conversations. In *EACL*, 2026.
- Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *Preprint arXiv:1412.6572*, 2014.
- Google. Announcing the Agent2Agent Protocol (A2A). <https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interopability/>, 2025.
- Google. Getting Started with Google Antigravity, 2026a. URL <https://codelabs.developers.google.com/getting-started-google-antigravity>. Accessed: 2026-08-17.
- Google. The new era of browsing: Putting Gemini to work in Chrome, 2026b. URL <https://blog.google/products-and-platforms/products/chrome/gemini-3-auto-browse/>. Accessed: 2026-08-17.
- Google. Introducing Gemini 3.7 Flash, 2026c. URL <https://blog.google/innovation-and-ai/models-and-research/gemini-models/introducing-gemini-3-7-flash/>. Accessed: 2026-08-18.
- Google. Agent Development Kit: Callbacks and Guardrails, 2026d. URL <https://adk.dev/safety/>. Accessed: 2026-06-18.
- Google Cloud. What are AI agents? Definition, examples, and types, April 2026a. URL <https://cloud.google.com/discover/what-are-ai-agents>.
- Google Cloud. About GKE Agent Sandbox, 2026b. URL <https://cloud.google.com/kubernetes-engine/docs/concepts/machine-learning/agent-sandbox>.

- Google Cloud. Understand how reCAPTCHA uses Private Access Tokens, 2026c. URL <https://cloud.google.com/recaptcha/docs/private-access-tokens>.
- Google Cloud. What is Retrieval-Augmented Generation (RAG)?, 2026d. URL <https://cloud.google.com/use-cases/retrieval-augmented-generation>. Accessed: 2026-09-10.
- Google LLC. Common Expression Language (CEL) Homepage. <https://cel.dev/>, 2021. Accessed: 2026-08-12.
- Tommaso Green, Martin Gubri, Haritz Puerto, Sangdoo Yun, and Seong Joon Oh. Leaky thoughts: Large reasoning models are not private thinkers. In *EMNLP*, 2025.
- Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you’ve signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *AISec Workshop at CCS*, 2023.
- Shubham Gupta, Nazanin Mohammadi Sepahvand, Abhinav Kumar, Cem Subakan, Spandana Gella, Pierre-André Noël, Perouz Taslakian, Eugene Bagdasarian, and Valentina Zantedeschi. PiSAs: Benchmarking Contextual Integrity in Multi-User Agentic Systems. *Preprint arXiv:2607.05318*, 2026.
- Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs. In *NeurIPS*, 2024.
- Rich Harang. Practical Security Guidance for Sandboxing Agentic Workflows and Managing Execution Risk, 2026. URL <https://developer.nvidia.com/blog/practical-security-guidance-for-sandboxing-agentic-workflows-and-managing-execution-risk/>.
- Dick Hardt. The OAuth 2.0 Authorization Framework. RFC 6749, Internet Engineering Task Force, 2012. URL <https://www.rfc-editor.org/rfc/rfc6749>.
- Woodrow Hartzog. The New Price to Play: Are Passive Online Media Users Bound By Terms of Use? *Communication Law and Policy*, 15(4):405–433, 2010. doi: 10.1080/10811680.2010.512514.
- Yufeng He. TrajBias: Structural biases in LLM-as-judge evaluation of agent trajectories. In *NeurIPS Datasets and Benchmarks Track*, 2026.
- Will Douglas Heaven. Moltbook was peak AI theater, February 2026. URL <https://www.technologyreview.com/2026/02/06/1132448/moltbook-was-peak-ai-theater/>.
- HiBob. HRIS (Human Resources Information System), 2026. URL <https://www.hibob.com/hris-bb-us/>. Accessed: 2026-08-18.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiaowu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Steven Yau, Zijuan Lin, Liyang Zhou, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *ICLR*, 2024.
- Tristan Hurnetz, Hosein Yavarzadeh, Albert Cheu, Adria Gascon, Lukas Gerlach, Daniel Moghimi, Phillipp Schoppmann, Michael Schwarz, and Ruiyi Zhang. TDXRay: Microarchitectural Side-Channel Analysis of Intel TDX for Real-World Workloads. In *S&P*, 2026.
- Eric Horvitz and Deirdre Mulligan. Data, privacy, and the greater good. *Science*, 349(6245):253–255, 2015.
- Wenbin Hu, Haoran Li, Huihao Jing, Qi Hu, Ziqian Zeng, Sirui Han, Heli Xu, Tianshu Chu, Peizhao Hu, and Yangqiu Song. Context Reasoner: Incentivizing Reasoning Capability for Contextualized Privacy and Safety Compliance via Reinforcement Learning. *Preprint arXiv:2505.14585*, 2025.
- Zhanhao Hu, Dennis Jacob, Xiao Huang, Zhaorun Chen, Bo Li, and David Wagner. Twin Agent: Context Residual Compression for Privilege Separated Agents. *arXiv preprint arXiv:2607.19595*, 2026.

- Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M Ziegler, Tim Maxwell, Newton Cheng, et al. Sleeper Agents: Training Deceptive LLMs that Persist Through Safety Training. *Preprint arXiv:2401.05566*, 2024.
- Hugging Face Security Team. Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident. Hugging Face Blog, July 2026. URL <https://huggingface.co/blog/agent-intrusion-technical-timeline>. Accessed: 2026-08-05.
- Bo Hui, Haolin Yuan, Neil Gong, Philippe Burlina, and Yinzhi Cao. PLeak: Prompt Leaking Attacks against Large Language Model Applications. In *CCS*, 2024.
- Amir Husain. An Agent Revolt: Moltbook Is Not A Good Idea. *Forbes*, 2026.
- IETF. RFC 2693: SPKI Certificate Theory. Technical report, Internet Engineering Task Force, 1999a.
- IETF. RFC 2704: The KeyNote Trust-Management System Version 2. Technical report, Internet Engineering Task Force, 1999b.
- IETF. RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Technical report, Internet Engineering Task Force, 2008.
- IETF. RFC 6962: Certificate Transparency. Technical report, Internet Engineering Task Force, 2013.
- IETF. RFC 9635: Grant Negotiation and Authorization Protocol (GNAP). Technical report, Internet Engineering Task Force, 2024.
- IETF. A Decoupled Authorization Model for Agent2Agent. Technical report, Internet Engineering Task Force, 2025a.
- IETF. Secure Intent Protocol: JWT Compatible Agentic Identity and Workflow Management. Technical report, Internet Engineering Task Force, 2025b.
- IETF. System for Cross-domain Identity Management: Agentic Identity Schema. Technical report, Internet Engineering Task Force, 2025c.
- IETF. Privacy Pass (privacypass) Working Group, 2026.
- IETF. The OAuth 2.1 Authorization Framework. Technical report, Internet Engineering Task Force, 2026.
- Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, et al. LLaMa Guard: LLM-based input-output safeguard for human-AI conversations. *Preprint arXiv:2312.06674*, 2023.
- Insurance Institute for Highway Safety. Pandemic lockdowns made rush-hour speeding, risky driving the new normal. *IIHS News Releases*, 2022.
- Intercom. AI Customer Service, 2026. URL <https://www.intercom.com/drlp/ai-customer-service>. Accessed: 2026-08-18.
- Dennis Jacob, Emad Alghamdi, Zhanhao Hu, Basel Alomair, and David Wagner. Preventing Prompt Injection with Type-Directed Privilege Separation. *arXiv preprint arXiv:2509.25926*, 2025.
- Hyejun Jeong, Dzung Pham, Amir Houmansadr, and Eugene Bagdasarian. AI snitches get glitches: Towards evading agentic surveillance. In *Preprint arXiv:2606.25836*, 2026a.
- Hyejun Jeong, Mohammadreza Teymorianfard, Abhinav Kumar, Amir Houmansadr, and Eugene Bagdasarian. Network-Level Prompt and Trait Leakage in Local Research Agents. In *USENIX Security*, 2026b.
- Rishi Jha, Harold Triedman, Arkaprabha Bhattacharya, and Vitaly Shmatikov. Agent Meltdowns: The Road to Hell Is Paved with Helpful Agents. *Preprint arXiv:2605.19149*, 2026a.

- Rishi Jha, Harold Triedman, Justin Wagle, and Vitaly Shmatikov. Breaking and Fixing Defenses Against Control-Flow Hijacking in Multi-Agent Systems. In *ICLR*, 2026b.
- Xisen Jin, Michael Duan, Qin Lin, Aaron Chan, Zhenglun Chen, Junyi Du, and Xiang Ren. Proof-of-Guardrail in AI Agents and What (Not) to Trust from It. *Preprint arXiv:2603.05786*, 2026.
- Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International Conference on Machine Learning*, pp. 1376–1385. PMLR, 2015.
- Dayeon Kang, Hyejun Jeong, Jade Sheffey, Pubali Datta, and Amir Houmansadr. Whose Agent Are You? Multi-Layer Fingerprinting and Attribution of Autonomous Web Agents. *Preprint arXiv:2606.20910*, 2026.
- Su Kara, Fazle Faisal, and Suman Nath. WAREX: Web Agent Reliability Evaluation on Existing Benchmarks. In *TMLR*, 2026.
- Rishabh Khandelwal, Thomas Linden, Hamza Harkous, and Kassem Fawaz. PriSEC: A privacy settings enforcement controller. In *USENIX Security*, 2021.
- Rishabh Khandelwal, Asmit Nayak, Hamza Harkous, and Kassem Fawaz. Automated Cookie Notice Analysis and Enforcement. In *USENIX Security*, 2023.
- Myeongseob Ko, Jihyun Jeong, Sumiran Singh Thakur, Gyuhak Kim, and Ruoxi Jia. From Weak Cues to Real Identities: Evaluating Inference-Driven De-Anonymization in LLM Agents. *Preprint arXiv:2603.18382*, 2026.
- Ceren Kocaogullar, Tina Marjanov, Ivan Petrov, Ben Laurie, Al Cutter, Christoph Kern, Alice Hutchings, and Alastair R Beresford. A Confidential Computing Transparency Framework for a Comprehensive Trust Chain. *Preprint arXiv:2409.03720*, 2024.
- U Kohl. *Regulatory Competence over Online Activity. Jurisdiction and the internet*. New York: Cambridge University Press, 2007.
- Aashish Kolluri, Rishi Sharma, Manuel Costa, Boris Köpf, Tobias Nießen, Mark Russinovich, Shruti Tople, and Santiago Zanella-Beguelin. Optimizing Agent Planning for Security and Autonomy. In *ICLR*, 2026.
- Vasilis Kontonis, Yuchen Zeng, Shivam Garg, Lingjiao Chen, Hao Tang, Ziyang Wang, Ahmed Awadallah, Eric Horvitz, John Langford, and Dimitris Papailiopoulos. MEMENTO: Teaching LLMs to Manage Their Own Context. *Preprint arXiv:2604.09852*, 2026.
- Priya C Kumar and Virginia L Byrne. The 5Ds of privacy literacy: a framework for privacy education. *Information and Learning Sciences*, 123(7-8):445–461, 2022.
- Priya C. Kumar, Michael Zimmer, and Jessica Vitak. A roadmap for applying the contextual integrity framework in qualitative privacy research. *Proceedings of the ACM on Human-Computer Interaction*, 8 (CSCW1), 2024.
- Guangchen Lan, Huseyin A Inan, Sahar Abdelnabi, Janardhan Kulkarni, Lukas Wutschitz, Reza Shokri, Christopher Brinton, and Robert Sim. Contextual Integrity in LLMs via Reasoning and Reinforcement Learning. In *NeurIPS*, 2025.
- Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiuotė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Measuring Faithfulness in Chain-of-Thought Reasoning. *Preprint arXiv:2307.13702*, 2023.
- Mia Lassiter and Brinnae Bent. Defining AI Agents: A Compendium of Criteria, Metrics, and Benchmarks. *Preprint arXiv:2609.11018*, 2026.

- Walter Laurito, Benjamin Davis, Peli Grietzer, Tomáš Gavenčíak, Ada Böhm, and Jan Kulveit. AI–AI bias: Large language models favor communications generated by large language models. In *PNAS*, 2025.
- Leapsome. Leapsome Workflows and Automations, 2026. URL <https://www.leapsome.com/product/leapsome-workflows-automations-v1>. Accessed: 2026-08-18.
- Donghyun Lee and Mo Tiwari. Prompt Infection: LLM-to-LLM Prompt Injection within Multi-Agent Systems. *Preprint arXiv:2410.07283*, 2024.
- Simon Lermen, Daniel Paleka, Joshua Swanson, Michael Aerni, Nicholas Carlini, and Florian Tramèr. Large-scale online deanonymization with LLMs. In *USENIX Security*, 2026.
- Roy J. Lewicki and Barbara B. Bunker. Developing and Maintaining Trust in Work Relationships. In Roderick M. Kramer and Tom R. Tyler (eds.), *Trust in Organizations: Frontiers of Theory and Research*, pp. 114–139. Sage Publications, 1996.
- Haoran Li, Wenbin Hu, Huihao Jing, Yulin Chen, Qi Hu, Sirui Han, Tianshu Chu, Peizhao Hu, and Yangqiu Song. PrivaCI-Bench: Evaluating privacy with contextual integrity and legal compliance. In *ACL*, 2025.
- Miles Q. Li, Benjamin C. M. Fung, Martin Weiss, Pulei Xiong, Khalil Al-Hussaeni, and Claude Fachkha. A Benchmark for Evaluating Outcome-Driven Constraint Violations in Autonomous AI Agents. *Preprint arXiv:2512.20798*, 2026a.
- Tianshi Li. Agentic LLMs as Powerful Deanonymizers: Re-identification of Participants in the Anthropic Interviewer Dataset. *Preprint arXiv:2601.05918*, 2026.
- Ziwen Li, Jianing Wen, and Tianshi Li. LLM Anonymization Against Agentic Re-Identification. *arXiv preprint arXiv:2605.30848*, 2026b.
- Minhua Lin, Enyan Dai, Hui Liu, Xianfeng Tang, Yuliang Yan, Zhenwei Dai, Jingying Zeng, Zhiwei Zhang, Fali Wang, Hongcheng Gao, et al. How Far Are LLMs from Professional Poker Players? Revisiting Game-Theoretic Reasoning with Agentic Tool Use. In *ICLR*, 2026.
- David Lindner, Victoria Krakovna, and Sebastian Farquhar. Gram: Assessing sabotage propensities via automated alignment auditing. *Preprint arXiv:2605.30322*, 2026.
- Neil O Littlefield. Federal Consumer Credit Legislation-Some New Developments-Part I: Fair Credit Billing. *Com. LJ*, 80:312, 1975.
- Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. In *ICLR*, 2024a. doi: 10.48550/arxiv.2310.04451.
- Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Formalizing and benchmarking prompt injection attacks and defenses. In *USENIX Security*, 2024b.
- Rongjun Ma, Shijing He, Jose Luis Martin-Navarro, Xiao Zhan, and Jose Such. Privacy in Human-AI Romantic Relationships: Concerns, Boundaries, and Agency. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, pp. 1–25, 2026.
- Gaurav Maheshwari, Dmitry Ivanov, and Kevin El Haddad. Efficacy of Synthetic Data as a Benchmark. *Preprint arXiv:2409.11968*, 2024.
- Saaduddin Mahmud, Eugene Bagdasarian, and Shlomo Zilberstein. Verification Required: The Impact of Information Credibility on AI Persuasion. *Preprint arXiv:2602.00970*, 2026.
- Samuele Marro, Emanuele La Malfa, Jesse Wright, Guohao Li, Nigel Shadbolt, Michael Wooldridge, and Philip Torr. A scalable communication protocol for networks of large language models. *Preprint arXiv:2410.11905*, 2024.
- Kirsten Martin and Helen Nissenbaum. Privacy interests in public records: An empirical investigation. *Harv. JL & Tech.*, 31:111, 2017.

- Harry Mayne, Justin Singh Kang, Dewi Gould, Kannan Ramchandran, Adam Mahdi, and Noah Y Siegel. A Positive Case for Faithfulness: LLM Self-Explanations Help Predict Model Behavior. *Preprint arXiv:2602.02639*, 2026.
- Nora McDonald and Andrea Forte. *Privacy and Vulnerable Populations*, chapter 15, pp. 337–363. Springer International Publishing, Cham, 2022. doi: 10.1007/978-3-030-82786-1_15.
- Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of Attacks: Jailbreaking Black-Box LLMs Automatically. In *NeurIPS*, 2024. doi: 10.48550/arxiv.2312.02119.
- Luoxi Meng, Henry Feng, Iliia Shumailov, and Earlene Fernandes. ceLLMate: Sandboxing Browser AI Agents. *arXiv preprint arXiv:2512.12594*, 2026.
- Microsoft. What is Microsoft Entra Agent ID?, 2026. URL <https://learn.microsoft.com/en-us/entra/agent-id/what-is-microsoft-entra-agent-id>.
- Lesly Miculicich, Mihir Parmar, Hamid Palangi, Krishnamurthy Dj Dvijotham, Mirko Montanari, Tomas Pfister, and Long T. Le. VeriGuard: Enhancing LLM Agent Safety via Verified Code Generation. *Preprint arXiv:2510.05156*, 2025.
- Niloofer Miresghallah, Neal Mangaokar, Narine Kokhlikyan, Arman Zharmagambetov, Manzil Zaheer, Saeed Mahloujifar, and Kamalika Chaudhuri. CIMemories: A Compositional Benchmark for Contextual Integrity of Persistent Memory in LLMs. In *ICLR*, 2026.
- Niloofer Miresghallah, Hyunwoo Kim, Xuhui Zhou, Yulia Tsvetkov, Maarten Sap, Reza Shokri, and Yejin Choi. Can LLMs keep a secret? Testing privacy implications of Language Models via Contextual Integrity theory. In *ICLR*, 2024.
- Gaurav Misra and Jose M Such. Pacman: Personal agent for access control in social media. *IEEE Internet Computing*, 21(6):18–26, 2017.
- Margaret Mitchell, Avijit Ghosh, Alexandra Sasha Luccioni, and Giada Pistilli. Fully Autonomous AI Agents Should Not be Developed. *Preprint arXiv:2502.02649*, 2025.
- Moltbook. Moltbook: The Front Page of the Agent Internet, 2026. URL <https://www.moltbook.com/>.
- Kyzyl Monteiro and Sauvik Das. Sketch-based Access Control: A Multimodal Interface for Translating User Preferences into Intent-Aligned Policies. *arXiv preprint arXiv:2605.10012*, 2026.
- Francesca Mosca and Jose Such. ELVIRA: An explainable agent for value and utility-driven multiuser privacy. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pp. 916–924, 2021.
- Sara Motiee, Kirstie Hawkey, and Konstantin Beznosov. Do Windows Users Follow the Principle of Least Privilege? Investigating User Account Control Practices. In *SOUPS*, 2010.
- Srija Mukhopadhyay, Sathwik Reddy, Shruthi Muthukumar, Jisun An, and Ponnurangam Kumaraguru. PrivacyBench: A Conversational Benchmark for Evaluating Privacy in Personalized AI. *Preprint arXiv:2512.24848*, 2025.
- Jay Murdoch and Michael Jawer. The Foundations of ADA Compliance: A Building Survey. *Mental & Physical Disability L. Rep.*, 16:101, 1992.
- Mason Nakamura, Abhinav Kumar, Saswat Das, Sahar Abdelnabi, Saaduddin Mahmud, Ferdinando Fioretto, Shlomo Zilberstein, and Eugene Bagdasarian. Colosseum: Auditing Collusion in Cooperative Multi-Agent Systems. In *NeurIPS*, 2026.
- Moses Namara, Henry Sloan, Priyanka Jaiswal, and Bart P. Knijnenburg. The Potential for User-Tailored Privacy on Facebook. In *Proceedings of the IEEE Symposium on Privacy-Aware Computing (PAC)*, pp. 31–42, 2018. doi: 10.1109/pac.2018.00010.

- Ali Naseh and Niloofar Mireshghallah. Synthetic Data Can Mislead Evaluations: Membership Inference as Machine Text Detection. *arXiv preprint arXiv:2501.11786*, 2025.
- Milad Nasr, Nicholas Carlini, Chawin Sitawarin, Sander V. Schulhoff, Jamie Hayes, Michael Ilie, Juliette Pluto, Shuang Song, Harsh Chaudhari, Ilia Shumailov, Abhradeep Thakurta, Kai Yuanqing Xiao, Andreas Terzis, and Florian Tramèr. The Attacker Moves Second: Stronger Adaptive Attacks Bypass Defenses Against LLM Jailbreaks and Prompt Injections. *Preprint arXiv:2510.09023*, 2025.
- Ben Nassi, Stav Cohen, and Or Yair. Invitation is all you need! Promptware attacks against LLM-Powered assistants in production are practical and dangerous. *Preprint arXiv:2508.12175*, 2025.
- NEAR AI. IronClaw Security Model, 2026. URL <https://mintlify.wiki/nearai/ironclaw/concepts/security>.
- Fredrik Nestaas, Edoardo Debenedetti, and Florian Tramèr. Adversarial Search Engine Optimization for Large Language Models. In *ICLR*, 2025.
- Kristina Nikolic, Egor Zverev, Javier Rando, Matthew Jagielski, Edoardo Debenedetti, and Florian Tramèr. Untrusted Content Masking for Web Agents with Security Guarantees. In *ICML Workshop*, 2026.
- Helen Nissenbaum. *Privacy in context: Technology, policy, and the integrity of social life*. Stanford University Press, 2009.
- Helen Nissenbaum. From Preemption to Circumvention: If Technology Regulates, Why Do We Need Regulation (and Vice Versa)? *Berkeley Tech. LJ*, 26:1367, 2011.
- Helen Nissenbaum. Respect for context as a benchmark for privacy online: What it is and isn't. *Social dimensions of privacy: Interdisciplinary perspectives*, 2015.
- Helen Nissenbaum. Contextual integrity up and down the data food chain. *Theoretical inquiries in law*, 20(1):221–256, 2019.
- Helen Nissenbaum, Sebastian Benthall, Anupam Datta, Michael Carl Tschantz, and Piotr Mardziel. Origin Privacy: Protecting Privacy in the Big-Data Era. Technical report, New York University, 2018.
- NIST. Autonomous Laboratories, 2026. URL <https://www.nist.gov/autonomous-laboratories>.
- Nous Research. Hermes Agent, 2026. URL <https://github.com/nousresearch/hermes-agent>.
- Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer, 2016.
- OpenAI. Codex, 2025a. URL <https://github.com/openai/codex>. Accessed: 2026-08-17.
- OpenAI. A Practical Guide to Building AI Agents. <https://openai.com/business/guides-and-resources/a-practical-guide-to-building-ai-agents/>, 2025b. Accessed: 2026-09-11.
- OpenAI. OpenAI and Hugging Face partner to address security incident during model evaluation. OpenAI Blog, July 2026a. URL <https://openai.com/index/hugging-face-model-evaluation-security-incident/>. Accessed: 2026-08-05.
- OpenAI. Sandbox Agents, 2026b. URL <https://developers.openai.com/api/docs/guides/agents/sandboxes>.
- OpenAI. Agents SDK Documentation: Guardrails, 2026c. URL <https://openai.github.io/openai-agents-python/guardrails/>. Accessed: 2026-06-18.
- OpenClaw. OpenClaw: Personal AI Assistant, 2026. URL <http://openclaw.ai>.
- OpenID Foundation. OpenID Foundation Website, 2026. URL <http://openid.net>.

- Elinor Ostrom. Institutional rational choice: An assessment of the institutional analysis and development framework. In *Theories of the Policy Process, Second Edition*, pp. 21–64. Routledge, 2019.
- OWASP GenAI Security Project. OWASP GenAI Top 10 2026. <https://genai.owasp.org/resource/owasp-genai-llm-top-10-2026/>, 2026. Accessed: 2026-08-04.
- Elizabeth O’Neill. Contextual integrity as a general conceptual tool for evaluating technological change. *Philosophy & Technology*, 2022.
- Nils Palumbo, Sarthak Choudhary, Jihye Choi, Guy Amir, Prasad Chalasani, and Somesh Jha. Formal Policy Enforcement for Real-World Agentic Systems. *Preprint arXiv:2602.16708*, 2026.
- Alexander Panfilov, David Schmotz, Ilia Shumailov, Luca Beurer-Kellner, Joachim Schaeffer, Ameya Prabhu, Jonas Geiping, and Maksym Andriushchenko. Stealing Reasoning Traces from Proprietary LLM APIs. *Preprint arXiv:2608.09867*, 2026.
- Emiliano Penalzoza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged Information Distillation for Language Models. In *ICML*, 2026.
- Guangyue Peng, Zongchao Chen, Wen Luo, Yuntao Wen, Wei Li, Ruixiang Feng, Ran Le, Chen Yang, Zhenwei An, Yang Song, Tao Zhang, and Houfeng Wang. Measuring and Mitigating Post-hoc Rationalization in Reverse Chain-of-Thought Generation. *Preprint arXiv:2602.14469*, 2026.
- Samuel Pfrommer, Yatong Bai, Tanmay Gautam, and Somayeh Sojoudi. Ranking Manipulation for Conversational Search Engines. In *EMNLP*, 2024.
- Dzung Pham, Jade Sheffey, Chau Minh Pham, and Amir Houmansadr. ProxyGPT: Enabling User Anonymity in LLM Chatbots via (Un)Trustworthy Volunteer Proxies. *Preprint arXiv:2407.08792*, 2025.
- Jovan Powar and Alastair R Beresford. SoK: Managing Risks of Linkage Attacks on Data Privacy. *PETS*, 2023.
- Chandra Prakash, Mary Lind, and Avneesh Sisodia. Agentic AI Governance and Lifecycle Management in Healthcare. *arXiv preprint arXiv:2601.15630*, 2026.
- Martin Prause. No skin in the game: why agentic AI requires principal-agent governance. *AI and Ethics*, 2026.
- punkpeye. I prompt injected my CONTRIBUTING.md – 50% of PRs are bots, March 2026. URL <https://glama.ai/blog/2026-03-19-open-source-has-a-bot-problem>.
- Yiting Qu, Ziqing Yang, Chi Cui, Ye Leng, Junjie Chu, and Yang Zhang. EchoCoT: Extracting Hidden Chain-of-Thought from Large Reasoning Models. *Preprint arXiv:2608.20055*, 2026.
- Stephan Rabanser, Sayash Kapoor, Peter Kirgis, Kangheng Liu, Saiteja Utpala, and Arvind Narayanan. Towards a Science of AI Agent Reliability. *Preprint arXiv:2602.16666*, 2026.
- Hossein A. Rahmani, Varsha Ramineni, Emine Yilmaz, Nick Craswell, and Bhaskar Mitra. Towards Understanding Bias in Synthetic Data for Evaluation. In *CIKM*, 2025.
- Mohan Rajagopalan and Vinay Rao. Authenticated Workflows: A Systems Approach to Protecting Agentic AI. *arXiv preprint arXiv:2602.10465*, 2026.
- Guruprasad Viswanathan Ramesh, Asmit Nayak, Basieem Siddique, and Kassem Fawaz. WebSP-Eval: Evaluating Web Agents on Website Security and Privacy Tasks. *Preprint arXiv:2604.06367*, 2026.
- MJ Rathbun. Gatekeeping in Open Source: The Scott Shambaugh Story, 2026.
- Traian Rebedea, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien, and Jonathan Cohen. NeMo guardrails: A toolkit for controllable and safe LLM applications with programmable rails. In *EMNLP*, 2023.

- Johann Rehberger. Amp Code: Arbitrary Command Execution via Prompt Injection Fixed, August 2025a. URL <https://embracethered.com/blog/posts/2025/amp-agents-that-modify-system-configuration-and-escape/>.
- Johann Rehberger. AI Kill Chain in Action: Devin AI Exposes Ports to the Internet with Prompt Injection, August 2025b. URL <https://embracethered.com/blog/posts/2025/devin-ai-kill-chain-exposing-ports/>.
- RentAHuman. RentAHuman: The Marketplace Where AI Agents Hire Humans, 2026. URL <https://rentahuman.ai/>.
- Ryan Ries. OpenClaw Explained: How 1.5M AI Agents Built a Religion, Crypto Economy, and Escaped Control. *Mission Cloud*, 2026.
- Ronald L Rivest and Butler Lampson. SDSI—A Simple Distributed Security Infrastructure. Technical Report Version 1.1, MIT Laboratory for Computer Science, October 1996. URL <https://people.csail.mit.edu/rivest/pubs/RL96.ver-1.1.html>.
- Franziska Roesner, Tadayoshi Kohno, Alexander Moshchuk, Bryan Parno, Helen J Wang, and Crispin Cowan. User-Driven Access Control: Rethinking Permission Granting in Modern Operating Systems. In *IEEE Symposium on Security and Privacy*, 2012.
- Jaechul Roh, Eugene Bagdasarian, Hamed Haddadi, and Ali Shahin Shamsabadi. SpiLLage: Agentic over-sharing on the web. *Preprint arXiv:2602.13516*, 2026.
- ruvnet. Agent Name Service (ANS) Protocol, 2026. URL <https://github.com/ruvnet/Agent-Name-Service>.
- Emil Ryd, Henning Bartsch, Julian Stastny, Joe Benton, and Vivek Hebbar. Removing Sandbagging in LLMs by Training with Weak Supervision. In *ICML*, 2026.
- Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *EMNLP*, 2023.
- Salesforce. AI Customer Service Agents, 2026. URL <https://www.salesforce.com/service/ai/customer-service-agents/>. Accessed: 2026-08-18.
- Ravi S Sandhu and Pierangela Samarati. Authentication, access control, and audit. *CSUR*, 1996.
- Madelyn Sanfilippo, Brett Frischmann, and Katherine Strandburg. Privacy as commons: Case evaluation through the governing knowledge commons framework. *Journal of Information Policy*, 8:116–166, 2018.
- Erik Schluntz and Barry Zhang. Building Effective Agents, December 2024. URL <https://www.anthropic.com/engineering/building-effective-agents>. Accessed: 2026-09-11.
- Christoph Schnabl, Daniel Hugenroth, Bill Marino, and Alastair R Beresford. Attestable Audits: Verifiable AI Safety Benchmarks Using Trusted Execution Environments. *Preprint arXiv:2506.23706*, 2025.
- Fred B Schneider. Enforceable security policies. *TISSEC*, 3(1):30–50, 2000.
- Andrew D. Selbst. Contextual Expectations of Privacy. *Cardozo Law Review*, 35:643, 2013.
- Alex Seto, Oytun Kuday Duran, Samy Amer, Jalen Chuang, Stephan van Schaik, Daniel Genkin, and Christina Garman. Wiretap: Breaking Server SGX via DRAM Bus Interposition. In *CCS*, 2025.
- Zeyang Sha and Yang Zhang. Prompt Stealing Attacks Against Large Language Models. *arXiv preprint arXiv:2402.12959*, 2024. doi: 10.48550/arxiv.2402.12959.
- Scott Shambaugh. An AI Agent Published a Hit Piece on Me, February 2026. URL <https://theshamblog.com/an-ai-agent-published-a-hit-piece-on-me/>.

- Yijia Shao, Tianshi Li, Weiyan Shi, Yanchen Liu, and Diyi Yang. Privacylens: Evaluating privacy norm awareness of language models in action. In *NeurIPS*, 2024a.
- Zedian Shao, Hongbin Liu, Jaden Mu, and Neil Gong. Enhancing Prompt Injection Attacks to LLMs via Poisoning Alignment. In *AISeC Workshop at CCS*, 2025.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *Preprint arXiv:2402.03300*, 2024b.
- Mrinank Sharma, Meg Tong, Jesse Mu, Jerry Wei, Jorrit Kruthoff, Scott Goodfriend, Euan Ong, Alwin Peng, Raj Agarwal, Cem Anil, Amanda Askell, Nathan Bailey, Joe Benton, et al. Constitutional Classifiers: Defending against Universal Jailbreaks across Thousands of Hours of Red Teaming. *Preprint arXiv:2501.18837*, 2025.
- Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. “Do Anything Now”: Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. In *CCS*, 2024.
- Chongyang Shi, Sharon Lin, Shuang Song, Jamie Hayes, Ilia Shumailov, Itay Yona, Juliette Pluto, Aneesh Pappu, Christopher A Choquette-Choo, Milad Nasr, et al. Lessons from Defending Gemini Against Indirect Prompt Injections. *Preprint arXiv:2505.14534*, 2025.
- Tianneng Shi, Jingxuan He, Zhun Wang, Hongwei Li, Linyu Wu, Wenbo Guo, and Dawn Song. Progent: Securing AI Agents with Privilege Control. *Preprint arXiv:2504.11703*, 2026.
- Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The Illusion of Thinking: Understanding the Strengths and Limitations of Reasoning Models via the Lens of Problem Complexity. In *NeurIPS*, 2025.
- Ilia Shumailov, Daniel Ramage, Sarah Meiklejohn, Peter Kairouz, Florian Hartmann, Borja Balle, and Eugene Bagdasarian. Trusted machine learning models unlock private inference for problems currently infeasible with cryptography. *Preprint arXiv:2501.08970*, 2025.
- Yan Shvartzshnaider and Vasisht Duddu. Position: Contextual integrity is inadequately applied to language models. In *ICML*, 2025.
- Yan Shvartzshnaider and Vasisht Duddu. Privacy Bias in Language Models: A Contextual Integrity-based Auditing Metric. In *PETS*, 2026.
- Yan Shvartzshnaider, Madelyn Rose Sanfilippo, and Noah Apthorpe. GKC-CI: A unifying framework for contextual norms and information governance. *JASIST*, 2022.
- Sierra. Customer Service AI, 2026. URL <https://sierra.ai/customer-service>. Accessed: 2026-08-18.
- Signal Messenger. Signal » Messenger: Speak Freely, 2026. URL <https://signal.org/>.
- Mattea Sim, Kurt Hugenberg, Tadayoshi Kohno, and Franziska Roesner. A Scalable Inclusive Security Intervention to Center Marginalized & Vulnerable Populations in Security & Privacy Design. In *New Security Paradigms Workshop (NSPW)*, September 2023.
- Mattea Sim, Basia Radka, Tadayoshi Kohno, Franziska Roesner, and Kurt Hugenberg. Characterizing the Default Persona: Mental Representations of Technology Users are Gendered. In *FACCT*, 2025.
- Vincent Siu, Jingxuan He, Kyle Montgomery, Zhun Wang, Neil Zhenqiang Gong, Chenguang Wang, and Dawn Song. A framework for formalizing LLM agent security. In *ICML Workshop (AIWILD)*, 2026a.
- Vincent Siu, Jingxuan He, Kyle Montgomery, Zhun Wang, Chenguang Wang, and Dawn Song. Position: Agent Security Needs Redefinition through a Holistic Framework. In *ICML Position Paper Track*, 2026b.

- Alaia Solko-Breslin, Pramod Kaushik Mudrakarta, Mihai Christodorescu, Somesh Jha, and Krishnamurthy Dj Dvijotham. Efficient and Sound Probabilistic Verification for AI Agents. In *SE&P*, 2027.
- Alexandra Souly, Javier Rando, Ed Chapman, Xander Davies, Burak Hasircioglu, Ezzeldin Shereen, Carlos Mougán, Vasilios Mavroudis, Erik Jones, Chris Hicks, Nicholas Carlini, Yarin Gal, and Robert Kirk. Poisoning Attacks on LLMs Require a Near-constant Number of Poison Samples. *Preprint arXiv:2510.07192*, 2025.
- Tobin South, Subramanya Nagabhushanaradhya, Ayesha Dissanayaka, Sarah Cecchetti, George Fletcher, Victor Lu, Aldo Pietropaolo, Dean H. Saxe, Jeff Lombardo, Abhishek Maligehalli Shivalingaiah, Stan Bounev, Alex Keisner, Andor Kesselman, Zack Proser, Ginny Fahs, Andrew Bunyea, Ben Moskowitz, Atul Tulshibagwale, Dazza Greenwood, Jiaxin Pei, and Alex Pentland. Identity Management for Agentic AI: The new frontier of authorization, authentication, and security for an AI agent world. Technical report, OpenID Foundation, 2025.
- Manu Sporny, Dave Longley, David Chadwick, and Ivan Herman. Verifiable Credentials Data Model v2.0. W3C Recommendation, W3C, May 2025. URL <https://www.w3.org/TR/vc-data-model-2.0/>.
- Manu Sporny, Dave Longley, Markus Sabadello, Drummond Reed, Orie Steele, and Christopher Allen. Decentralized Identifiers (DIDs) v1.1: Core architecture, data model, and representations. Technical report, W3C, March 2026.
- Robin Staab, Nikola Jovanović, Kimberly Mai, Prakhar Ganesh, Martin Vechev, Ferdinando Fioretto, and Matthew Jagielski. SoK: Data Minimization in Machine Learning. In *IEEE Secure and Trustworthy Machine Learning Conference*, volume 3, 2026.
- Robert Stanley, Avi Verma, Lillian Tsai, Konstantinos Kallas, and Sam Kumar. An AI Agent Execution Environment to Safeguard User Data. *arXiv preprint arXiv:2604.19657*, 2026.
- Brian Stanton, Mary F. Theofanos, Sandra Spickard Prettyman, and Susanne Furman. Security Fatigue. *IT Professional*, 18(5):26–32, 2016. doi: 10.1109/mitp.2016.84.
- Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Eduard Hovy. Spine: Sparse interpretable neural embeddings. In *AAAI*, 2018.
- Jose Such, Agustín Espinosa, and Ana García-Fornes. A survey of privacy in multi-agent systems. *The Knowledge Engineering Review*, 29(3):314–344, 2014.
- Haochen Sun, Jason Li, and Hongyang Zhang. zkLLM: Zero Knowledge Proofs for Large Language Models. In *CCS*, 2024.
- Supreme Court of the United States. Katz v. United States. *United States Reports*, 389:347, 1967.
- Dan Jerker Svantesson. *Solving the Internet Jurisdiction Puzzle*. Oxford: Oxford University Press, 2017.
- Georgios Syros, Anshuman Suri, Jacob Ginesin, Cristina Nita-Rotaru, and Alina Oprea. SAGA: A security architecture for governing AI agentic systems. *arXiv preprint arXiv:2504.21034*, 2025.
- Manveer Singh Tamber, Abhay Puri, Marc-Etienne Brunet, Perouz Taslakian, Jimmy Lin, and Spandana Gella. PrivacyAlign: Contextual Privacy Alignment for LLM Agents. *Preprint arXiv:2606.21710*, 2026.
- Leonie Tanczer, Isabel Lopez Neira, Simon Parkin, Trupti Patel, and George Danezis. Gender and IoT research report: The rise of the internet of things and implications for technology-facilitated abuse, 2018. <https://www.ucl.ac.uk/steapp/sites/steapp/files/giot-report.pdf>.
- Maximilian Teltzrow and Alfred Kobsa. Impacts of User Privacy Preferences on Personalized Systems. In *Human-Computer Interaction Series*, pp. 315–332. Springer, 2004. doi: 10.1007/1-4020-2148-8_17.
- David Temoshok, Yee-Yin Choong, Andrew Regenscheid, Ryan Galluzzo, James L. Fenton, Justin Richer, and Naomi Lefkovitz. Digital Identity Guidelines: Authentication and Authenticator Management. Technical report, National Institute of Standards and Technology, 2025.

- Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, et al. Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet. *Transformer Circuits*, 2024.
- The Tor Project. The Tor Project | Privacy & Freedom Online, 2026. URL <https://www.torproject.org/>.
- The Wall Street Journal. Gemini Hacked Three Companies in First Known Breakout by Google’s AI. The Wall Street Journal, September 2026. Accessed: 2026-09-26.
- Corinn Tiffany, Wen Zhang, Eugene Bagdasarian, and Lillian Tsai. Sapien: A Stateful Policy Engine for Autonomous AI Agents. *Preprint arXiv*, 2026.
- Toloka. Computer Use Agents: What They Are, How They Work, and How to Deploy Them Safely, 2025. URL <https://toloka.ai/blog/computer-use-agents-what-they-are-how-they-work-and-how-to-deploy-them-safely/>. Accessed: 2026-08-17.
- Florian Tramèr. Detecting adversarial examples is (nearly) as hard as classifying them. In *ICML*, 2022.
- Florian Tramèr, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.
- Cuong Tran and Ferdinando Fioretto. Data Minimization at Inference Time. In *Advances in Neural Information Processing Systems*, volume 36. Curran Associates, Inc., 2023.
- Transparency.dev. Transparency.dev: Trust your data with a tamper-evident log, 2026. URL <https://transparency.dev/>.
- Harold Triedman, Rishi Jha, and Vitaly Shmatikov. Multi-Agent Systems Execute Arbitrary Malicious Code. In *COLM*, 2025.
- Lillian Tsai and Eugene Bagdasarian. Contextual Agent Security: A policy for every purpose. In *HotOS*, 2025.
- Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering Language Models With Activation Engineering. *Preprint arXiv:2308.10248*, 2023.
- Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting. In *NeurIPS*, 2023.
- U.S. Department of Energy. Colonial Pipeline Cyber Incident, 2021. URL <https://www.energy.gov/ceser/colonial-pipeline-cyber-incident>.
- Sanidhya Vijayvargiya, Aditya Bharat Soni, Xuhui Zhou, Zora Zhiruo Wang, Nouha Dziri, Graham Neubig, and Maarten Sap. OpenAgentSafety: A comprehensive framework for evaluating real-world AI agent safety. In *ICLR*, 2026.
- Martina G Vilas, Safoora Yousefi, Besmira Nushi, Eric Horvitz, and Vidhisha Balachandran. Tracing the Traces: Latent Temporal Signals for Efficient and Accurate Reasoning. *Preprint arXiv:2510.10494*, 2025.
- Salomé Viljoen. A relational theory of data governance. *The Yale Law Journal*, pp. 573–654, 2021.
- Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning Language Models During Instruction Tuning. In *ICML*, 2023.
- Ethan Wang, Zubair Shafiq, and Yash Vekaria. FP-Agent: Fingerprinting AI Browsing Agents. *Preprint arXiv:2605.01247*, 2026.
- Jin Wang and Jessica B. Cicchino. Changes in speeding on Virginia roads during the beginning of the COVID-19 pandemic. *Traffic Injury Prevention*, 24(1):38–43, 2023. doi: 10.1080/15389588.2022.2127322.

- Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small. In *ICLR*, 2023.
- Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large Language Models are not Fair Evaluators. In *ACL*, 2024.
- Shouju Wang and Haopeng Zhang. MPC-Bench: A Benchmark for Multimodal Pairwise Contextual Integrity Evaluation of Language Model Agents. *Preprint arXiv:2601.08235*, 2026.
- Shouju Wang, Fenglin Yu, Xirui Liu, Xiaoting Qin, Jue Zhang, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. Privacy in Action: Towards Realistic Privacy Mitigation and Evaluation for LLM-Powered Agents. *Preprint arXiv:2509.17488*, 2025.
- Noel Warford, Tara Matthews, Kaitlyn Yang, Omer Akgul, Sunny Consolvo, Patrick Gage Kelley, Nathan Malkin, Michelle L. Mazurek, Manya Sleeper, and Kurt Thomas. SoK: A Framework for Unifying At-Risk User Research. In *SE*, 2022.
- Samuel D Warren and Louis D Brandeis. The right to privacy. *Harvard Law Review*, 4:193, 1890.
- Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In *EMNLP*, 2023.
- Alan F Westin. Privacy and freedom. *Washington and Lee Law Review*, 25(1):166, 1968.
- Primal Wijesekera, Arjun Baokar, Ashkan Hosseini, Serge Egelman, David Wagner, and Konstantin Beznosov. Android permissions remystified: A field study on contextual integrity. In *USENIX Security*, 2015.
- Simon Willison. The Dual LLM pattern for building AI assistants that can resist prompt injection, April 2023. URL <https://simonwillison.net/2023/Apr/25/dual-llm-pattern/>.
- Cam Wilson and Rhiannon Hobbins. AI assistant hacks gym website in first known Australian autonomous cyber attack. ABC News Australia, August 2026. URL <https://www.abc.net.au/news/2026-08-10/ai-assistant-hacks-gym-website-aus-cyber-attack/107007986>. Accessed: 2026-09-03.
- Peter Winn. Katz and the origins of the reasonable expectation of privacy test. *McGeorge L. Rev.*, 40:1, 2009.
- Yotam Wolf, Noam Wies, Oshri Avnery, Yoav Levine, and Amnon Shashua. Fundamental Limitations of Alignment in Large Language Models. In *ICML*, 2024.
- Fangzhou Wu, Ethan Cecchetti, and Chaowei Xiao. System-Level Defense against Indirect Prompt Injection Attacks: An Information Flow Control Perspective. *Preprint arXiv:2409.19091*, 2024a.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024b.
- Yuhao Wu, Franziska Roesner, Tadayoshi Kohno, Ning Zhang, and Umar Iqbal. IsolateGPT: An Execution Isolation Architecture for LLM-Based Agentic Systems. In *NDSS*, 2025.
- Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.
- Yuchong Xie, Mingyu Luo, Zesen Liu, Zhixiang Zhang, Kaikai Zhang, Yu Liu, Zongjie Li, Ping Chen, Shuai Wang, and Dongdong She. Red-Teaming Coding Agents from a Tool-Invocation Perspective: An Empirical Security Assessment. In *ISSTA*, 2026.

- Jiaqi Xue, Mengxin Zheng, Yebowen Hu, Fei Liu, Xun Chen, and Qian Lou. BadRAG: Identifying vulnerabilities in retrieval augmented generation of large language models. *Preprint arXiv:2406.00083*, 2024.
- Ruixin Yang, Ethan Mendes, Arthur Wang, James Hays, Sauvik Das, Wei Xu, and Alan Ritter. Do Vision-Language Models Respect Contextual Integrity in Location Disclosure? *Preprint arXiv:2602.05023*, 2026.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. *Preprint arXiv:2210.03629*, 2022.
- YS Yashwanth, Ruichen Wang, Shihua Zeng, Xuhui Zhou, Koichi Onoue, Vasudha Varadarajan, and Maarten Sap. SOTOPIA-TOM: Evaluating privacy and information management in multi-agent interaction with theory of mind. In *COLM*, 2026.
- Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-Policy Context Distillation for Language Models. *Preprint arXiv:2602.12275*, 2026.
- Jingwei Yi, Yueqi Xie, Bin Zhu, Emre Kiciman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. Benchmarking and defending against indirect prompt injection attacks on large language models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '25, pp. 1809–1820. ACM, July 2025a. doi: 10.1145/3690624.3709179. URL <http://dx.doi.org/10.1145/3690624.3709179>.
- Ren Yi, Octavian Suci, Adrian Gascon, Sarah Meiklejohn, Eugene Bagdasarian, and Marco Gruteser. Privacy reasoning in ambiguous contexts. In *NeurIPS*, 2025b.
- Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang, Rui Wang, and Gongshen Liu. R-Judge: Benchmarking Safety Risk Awareness for LLM Agents. In *EMNLP*, 2024.
- Zeyu Yun, Yubei Chen, Bruno Olshausen, and Yann LeCun. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. In *DeeLIO Workshop*, pp. 1–10, 2021.
- Wenjun Zeng, Yuchi Liu, Ryan Mullins, Ludovic Peran, Joe Fernandez, Hamza Harkous, Karthik Narasimhan, Drew Proud, Piyush Kumar, Bhaktipriya Radharapu, et al. ShieldGemma: Generative AI content moderation based on Gemma. *Preprint arXiv:2407.21772*, 2024.
- Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents. In *ACL*, 2024a.
- Xiao Zhan, Stefan Sarkadi, and Jose Such. Privacy-enhanced Personal Assistants based on Dialogues and Case Similarity. In *ECAI*, 2023.
- Xiao Zhan, Noura Abdi, William Seymour, and Jose Such. Healthcare voice AI assistants: factors influencing trust and intention to use. *Proceedings of the ACM on Human-Computer Interaction*, 8(CSCW1):1–37, 2024b.
- Xiao Zhan, Juan Carlos Carrillo, William Seymour, and Jose Such. Malicious LLM-based Conversational AI Makes Users Reveal Personal Information. In *USENIX Security*, 2025.
- Xiao Zhan, Guangzhi Sun, Jose Such, and Phil Woodland. Protecting Bystander Privacy via Selective Hearing in Audio LLMs. In *ACL*, 2026a.
- Zhonghao Zhan and Hamed Haddadi. Auto-Policy, not Auto-Skill: Compiled Agent Skills for the Physical World. *ACM Workshop on Agent Skills*, 2026.
- Zhonghao Zhan, Huichi Zhou, Zhenhao Li, Peiyuan Jing, Krinos Li, and Hamed Haddadi. How Adversarial Environments Mislead Agentic AI? In *ACL*, 2026b.

- Kaiyuan Zhang, Zian Su, Pin-Yu Chen, Elisa Bertino, Xiangyu Zhang, and Ninghui Li. LLM agents should employ security principles. *Preprint arXiv:2505.24019*, 2025a.
- Michael Zhang, W Bradley Knox, and Eunsol Choi. Modeling future conversation turns to teach LLMs to ask clarifying questions. In *ICLR*, 2025b.
- Michael JQ Zhang and Eunsol Choi. Clarify when necessary: Resolving ambiguity through interaction with LMs. In *NAACL*, 2025.
- Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. How Language Model Hallucinations Can Snowball. In *ICML*, 2024.
- Tingwei Zhang, John X. Morris, and Vitaly Shmatikov. How to Steal Reasoning Without Reasoning Traces. *Preprint arXiv:2603.07267*, 2026a.
- Tingwei Zhang, Harold Triedman, and Vitaly Shmatikov. Deep-Research Agents Can Be Poisoned via User-Generated Content. *Preprint arXiv:2605.24245*, 2026b.
- Weichen Zhang, Yiyu Sun, Pohao Huang, Jiayue Pu, Heyue Lin, and Dawn Song. MIRAGE-Bench: LLM agent is hallucinating and where to find them. *arXiv preprint arXiv:2507.21017*, 2025c.
- Yixiang Zhang, Xinhao Deng, Zhongyi Gu, Yihao Chen, Ke Xu, Qi Li, and Jianping Wu. Exposing LLM User Privacy via Traffic Fingerprint Analysis: A Study of Privacy Risks in LLM Agent Interactions. *Preprint arXiv:2510.07176*, 2025d.
- Linxi Zhao, Sofian Zalouk, Christian K. Belardi, Justin Lovelace, Jin Peng Zhou, Ryan Thomas Noonan, Dongyoung Go, Kilian Q. Weinberger, Yoav Artzi, and Jennifer J. Sun. Pre-Training Limited Memory Language Models with Internal and External Knowledge. *Preprint arXiv:2505.15962*, 2025.
- Arman Zharmagambetov, Chuan Guo, Ivan Evtimov, Maya Pavlova, Ruslan Salakhutdinov, and Kamalika Chaudhuri. AgentDam: Privacy leakage evaluation for autonomous web agents. In *NeurIPS*, 2025.
- Huichi Zhou, Siyuan Guo, Anjie Liu, Zhongwei Yu, Ziqin Gong, Bowen Zhao, Zhixun Chen, Menglong Zhang, Yihang Chen, Jinsong Li, et al. Memento-skills: Let agents design agents. *Preprint arXiv:2603.18743*, 2026a.
- Jijie Zhou, Niloofar Mireshghallah, and Tianshi Li. Operationalizing Data Minimization for Privacy-Preserving LLM Prompting. In *ICLR*, 2026b.
- Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *ICLR*, 2024.
- Xuhui Zhou, Hyunwoo Kim, Faeze Brahman, Liwei Jiang, Hao Zhu, Ximing Lu, Frank F. Xu, Bill Yuchen Lin, Yejin Choi, Niloofar Mireshghallah, Ronan Le Bras, and Maarten Sap. HAICOSYSTEM: An ecosystem for sandboxing safety risks in interactive AI agents. In *COLM*, 2025.
- Jinhao Zhu, Kevin Tseng, Gil Vernik, Xiao Huang, Shishir G. Patil, Vivian Fang, and Raluca Ada Popa. MiniScope: A least privilege framework for authorizing tool calling agents. *arXiv preprint arXiv:2512.11147*, 2025.
- Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and Transferable Adversarial Attacks on Aligned Language Models. *arXiv preprint arXiv:2307.15043*, 2023. doi: 10.48550/arxiv.2307.15043.
- Jiaru Zou, Ruizhong Qiu, Gaotang Li, Xiyuan Yang, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, James Zou, Mengdi Wang, and Ling Yang. Latent Collaboration in Multi-Agent Systems. In *ICML*, 2026.
- Wei Zou, Rumpeng Geng, Binghui Wang, and Jinyuan Jia. PoisonedRAG: Knowledge Poisoning Attacks to Retrieval-Augmented Generation of Large Language Models. In *USENIX Security*, 2025.

Appendix

A Contextual Integrity Glossary of Key Terms

Below, we provide core definitions of contextual integrity developed by Nissenbaum and Shvartzshnaider.¹⁰

A.1 Context [social domains, social spheres]

“Contexts are structured social settings characterized by canonical activities, roles, relationships, power structures, norms (or rules), and internal values (goals, ends, purposes)” (Nissenbaum, 2009, p. 132). These may include healthcare, education, political life, commerce, social (family and friends), etc. They are “abstract representations of social structures experienced in daily life” (Nissenbaum, 2009, pp. 134-135). Social domains or social spheres are closely related concepts.

A.2 Norm

Behavior-guiding norms prescribe and proscribe actions and practices. A norm’s behavior-guiding power may be directly prescriptive or it may be indirectly prescriptive, based on an overwhelming preponderance of behavior of a particular kind. Norms may be explicit or implicit; they may vary in kind, e.g. ethical, aesthetic, formal, informal. They may be standalone or elements of complex systems of rules, e.g., law, professional code of conduct. Rules, Laws, Directives, etc., are related concepts (Nissenbaum, 2009, pp. 137-140).

A.2.1 Contextual informational norm

(Also referred to as informational norm, context-relative informational norm, context-specific informational norm, CI norm, privacy norm.)

CI norms prescribe and proscribe the flow of personal information from one party to another (Nissenbaum, 2009, pp. 140-147). Contextual informational norms are defined by five parameters: Sender, Recipient, Data Subject, Transmission Principle, Information Type. Well-formed instances of informational norms include values for all five parameters. The presence of CI norms may be established empirically. They may or may not be legitimate depending on whether they effectively serve (or promote) contextual ends, purposes, and values.

A.3 Personal information

As used in CI, this term simply means information about a person. The source is irrelevant, whether it is the data subject, a third party, or inferred.

A.4 Contextual information flow

Flow of information from an actor to other actors. Well-formed descriptions of contextual information flows must include values for all five parameters expressed in terms of contextual roles and informational ontologies.

¹⁰A full and up-to-date version of the glossary is available at <https://privaci.info/glossary>.

A.5 CI parameter

This refers to the five parameters of CI norms: Actors (subject, sender, recipient), Attribute, Transmission Principle.

A.5.1 Values for CI parameters

Instantiations of CI parameters referred to in terms of respective contextual ontologies of roles and information types.

A.5.2 Actor

This term refers collectively to the parameters: sender, recipient, and data subject. Values for these parameters are given in terms of respective contextual role ontologies. Examples include: teachers, doctors, etc. It may also refer to non-human parties, such as corporations, organizations, governments. Individuals (or agents) are actors, in this sense, insofar as they are acting in capacities according to contextual ontologies.

A.5.3 Contextual role

Contexts are co-constituted by contextual role ontologies. A contextual role [capacity] involves fulfilling certain functions, meeting particular expectations, and assuming certain responsibilities. There is no fixed number of roles per context and the roles themselves may change over time, and may even disappear while new ones emerge. Roles may be thickly detailed or quite scant.

A.5.4 Information (data) type

(Also referred to as data type, data field, or attribute.) This term points to what the information is about, its semantic content, what the data or information refers to. Contexts are associated with distinctive information (data) ontologies.

A.5.5 Transmission principle

A transmission principle (TP) is a flow-guiding principle. Commonly occurring TPs include: with consent, reciprocally, with notice, in confidence, as required by law, with a warrant. TPs cut across contextual ontologies.

A.6 Privacy expectation

In US law, “reasonable expectation of privacy” ([Supreme Court of the United States, 1967](#)) implies a widespread or commonly held expectation. It has played a central role in legal arguments, supporting restrictions on access to personal information. In CI the term connects this powerful legal doctrine to the concept of a contextual informational norm ([Selbst, 2013](#)) which as a matter of fact, describes a commonly held view on appropriate data flows.

A.7 Contextual integrity

Contextual integrity is preserved [respected, maintained] when justifiable contextual informational norms are respected, OR when information flows promote [support] contextual ends, purposes, and values and do not undermine contextual ends, purposes, and values (see CI heuristic in §[A.8](#)).

A.7.1 Contextual values

Including goals, ends, purposes, and values, these teleological elements define and co-constitute a context. They are the rationale behind a context. Contextual functions, norms, and ontologies are oriented around these elements. Activities and practices make sense insofar as they support and promote contextual ends, purposes and values. Information norms are legitimate insofar as they promote contextual ends, purposes,

and values. By implication, entrenched functions, norms, practices, and ontologies may be challenged by alternatives that are shown to be more effective in supporting and promoting contextual ends, purposes, and values (Nissenbaum, 2009, pp. 155-166).

A.7.2 Contextual functions, practices, activities

These are behaviors, activities, and practices that are typical or expected for a given context. They may achieve this status through repetition, established as an aspect of the culture of a context, or because they contribute to contextual functions, in turn instrumental to achieving higher order goals and values. [See, for example, Conley et al. (2011).]

A.7.3 [Moral] legitimacy of informational norms

The presence of entrenched information-flow norms constitutes *prima facie* evidence of the legitimacy of respective flows. To be morally justifiable [legitimate], the norm in question must be shown to perform well on three layers of analysis: (1) strike an acceptable balance of wellbeing among all affected parties; (2) be compatible with central ethical and political principles; and (3) promote contextual ends, purposes, and values (Nissenbaum, 2009, Chapter 8).

A.7.4 Contextual ontologies

These refer to ontologies of contextual roles and information [data, attributes]. Although contextual ontologies are distinctive for each context, there are likely to be overlaps in roles and information categories among different contexts (Nissenbaum, 2019).

A.8 The CI heuristic protocol

The CI heuristic (Nissenbaum, 2009, page 282) described below provides guidance on whether a given information practice threatens or upholds privacy as CI by evaluating information flow patterns resulting from the deployment of sociotechnical systems. The CI heuristic may also guide the developers of sociotechnical systems who wish to assess design alternatives in light of their impacts on privacy as CI.

- Describe information flows in terms of five parameters [the CI-tuple]: subject, sender, recipient, attribute, transmission principle (see Values for CI parameters, §A.5.1).
- Identify a prevailing context, or contexts, e.g., healthcare, allowing that subcontexts, e.g., teaching hospital, might be nested within them (Nissenbaum, 2009, page 136).
- Locate relevant contextual informational norms, if any, and note conformance or points of departure.
 - **Case 1:** Where conformance is achieved, appropriate flow has been maintained.
 - **Case 2:** Where flows are in breach of norms, an evaluator or designer concludes that a *prima facie* violation of contextual integrity has occurred. Presumption favors entrenched, pre-existing norms over divergent data flows.
 - **Case 3:** No norm exists covering observed data flows. Technology-enabled data flows may be sufficiently novel that existing norms are too distant for comparative purposes. Systems of contextual norms are likely to be “incomplete” in relation to all possible data flow instances.
- When there is a breach (Case 2) or a lacuna (Case 3), the following assessments are indicated:
 - **Evaluation I:** Carefully assess the impacts of novel flows on those affected, viz. stakeholders. Typically, this could include all the parties that the CI-tuple identifies as actors, as well as others beyond these actors. (“It requires a study of how novel flows affect the interests of key affected parties: the benefits they enjoy, the costs and risks they suffer. These may include material costs and benefits as well as those less palpable, including shifts in relative power” (Nissenbaum, 2015).)
 - **Evaluation II:** Consider the impacts in terms of high-level moral principles, such as autonomy, justice, fairness, and liberty, as well as political principles concerning power distributions, social hierarchy, democracy, and so on.

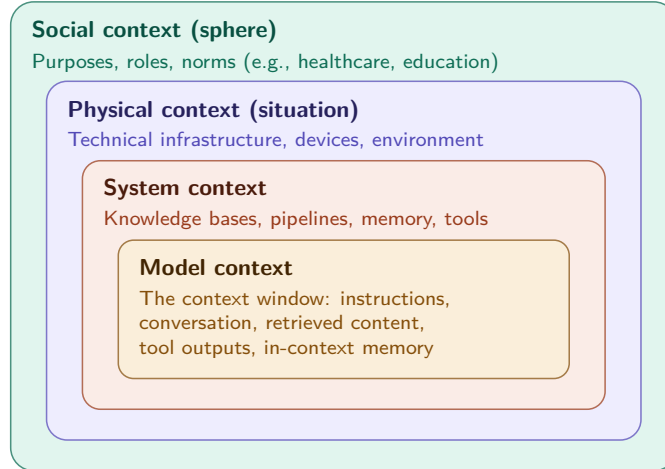


Figure 3: A schematic diagram of multiple meanings of context. A model may be embedded in a system, each of which has context. It is deployed into a physical context. It may also operate in a social context, which provisions norms that are salient to user privacy. Note that this is not a Venn diagram, but a depiction of different nested components.

- **Evaluation III:** Consider how the system or practices impinge on the values, ends, and purposes of the contexts in question.

The result of the protocol’s layered analysis (adapted from [Nissenbaum \(2009, pages 184-185\)\)](#) may clearly favor accepting or rejecting a particular system on the basis of novel and disruptive data flows. Ambiguous outcomes, however, indicate a need for more data, further judgment and deliberation, or even a reversion to the status quo.

B Definitions of Context

Following discussion in Section 4.1.1, the *model context* is unstructured digital material, such as text, which is available to the LLM as an input as it generates new output. The amount of information that the LLM may consider is limited by its context window, and this can be a challenge to an LLM’s performance on complex problems ([Shojaee et al., 2025](#)). Orchestration techniques may be used to manage the LLM’s actions, changing the model context multiple times as the model “reasons” through a problem. This is sometimes referred to as “multi-agent” reasoning ([Hong et al., 2024](#); [Wu et al., 2024b](#)). These orchestration techniques can be understood as a system-level solution to a model-level constraint.

In this paper, we also discuss *system context*, a component of an agentic system: its internally accessible knowledge base. A reasoning agent may employ digital contextual information that goes beyond the model’s context window but is nevertheless digitally provisioned, perhaps as unstructured data. Both senses of context refer to the software system that implements the agent.

However, “context” can also refer to a system’s environment. In early work on “privacy in context” ([Ackerman et al., 2001](#)), context is understood as the specific environment in which a system is deployed, inclusive of the technical and physical details. A prototypical example technology in this setting would be a sensor and display system for controlling heating and lighting conditions in an office building. [Benthall et al. \(2017\)](#) use the term *situation* to denote the concrete, specific circumstances in which a system is deployed.

Contextual Integrity emphasizes *social context*, an established social field of activity with widely understood purposes and norms. It is understood normatively and abstractly – in terms of goals and values – as opposed to the descriptive specifics. [Benthall et al. \(2017\)](#) refer to this form of context as a *sphere*. This context exists primarily outside of the software system and in the social environment. A prototypical technology that interacts with such a context is an electronic medical records system; in that context the norms are

often clear-cut and described by law. However, CI also theorizes how new social contexts can emerge and adapt to changing circumstances. In all cases, CI locates the norms and significance of privacy in such a social context.

From the perspective of CI, the challenge of agentic privacy is aligning system behavior with the norms and purposes of the social context. One way to accomplish this is by adequately representing the social context in the system context—so that the agent has available all the relevant information about the social context it needs to act appropriately. This can be challenging, because social context sometimes is unfixed and established dynamically in the interactions between a system and multiple other people (Dourish, 2004). In other words, mere representation of norms as “policies” may be insufficient; privacy must be enacted through agent behavior.

C Contextual Integrity: Critiques, Expansions, and Open Problems

CI is a live theory of privacy with an active community of scholars who sharpen and develop the theory as they apply it to new technologies and scenarios. In §4.1 we discussed the canonical theory of Contextual Integrity, and reflected on the importance of consistency and standardization. In this section, we discuss where scholars have critiqued and proposed extensions to CI. The understandable desire to adapt CI to new technologies—such as agentic AI systems—is in tension with the need to standardize an ontology to make evaluations meaningful. This makes the development of CI itself into an open research problem.

C.1 Expanding on transmission properties

In CI’s standard formulation, information flows are characterized by sender, receiver, subject, attribute, and transmission principle. Some have argued that CI can be a more powerful and expressive theory if it develops a richer taxonomy of the descriptive transmission properties of information flows (Benthall & Cummings, 2024).

Kumar et al. (2024) discuss the topic of information *medium*, such as face-to-face conversations, handwritten notes, text messages, or public social media posts, and how to incorporate it into the CI taxonomy of information flows. Kumar & Byrne (2022) suggest it should be an additional parameter. Here, we present it as another way of elaborating the “transmission properties,” descriptive elements of the information flow.

One use of “transmission properties” is to adapt CI to the possibility of privacy-enhancing technologies (PETs) such as differential privacy. Benthall & Cummings (2024) note that CI does not currently have a native way of registering that an information flow is aggregated over many data subjects, queried into a single statistic, and modified with random noise—typical steps in a differentially private disclosure of personal information. “Transmission properties” bridge the explanatory gap and open the door to new methods that use CI as a way of tuning the design of sociotechnical systems with PETs (Benthall & Cummings, 2026).

C.2 Clarifying the transmission principles

An information norm in CI determines the conditions under which an information flow is appropriate. These conditions are referred to as the “transmission principle.” Though they carry the normative load of CI, there is surprisingly little development of what transmission principles are in the original theory. Currently there is no straightforward enumeration of transmission principles, such that one could structure a search over possible norms.

Shvartzshnaider et al. (2022) have proposed a clarifying framework, GKC-CI, that combines CI with the institutional grammar of Governing Knowledge Commons (GKC) (Frischmann et al., 2014; Sanfilippo et al., 2018) to provide a more granular taxonomy of transmission principles. They consider four aspects:

1. **Aims.** e.g. for use in personalized advertising. The anticipated use of the information by the receiver.
2. **Conditions.** e.g. if given consent. An event that occurs prior to the transmission.

3. **Modalities.** e.g. can, might, will. Addressing whether the condition is sufficient or necessary for appropriateness.
4. **Consequences.** e.g. if necessary to verify identity. A consequence of the transmission which must be taken into account.

These elements of transmission principles may find relevance when CI is applied to complex legal and social domains. For example, the General Data Protection Regulation (GDPR) of the EU depends heavily on “purpose binding” of information flows; this is not a first-order element of CI theory but can be understood as a transmission principle that depends on the consistently maintained “aims” of information use as it flows from actor to actor.

C.3 Modeling context

As transmission principles are developed in a deeper and more comprehensive way, it becomes apparent that proper enforcement of information norms may require a great deal of information about the environment or scenario. A norm might require that a receiver be committed to particular uses of the information, or be unaware of the information already. The information sender may, in order to be compliant with norms, need to model what the receiver already knows, what they need to know to fulfill the context’s purpose, what they might infer by combining new information with prior knowledge, and what downstream consequences disclosure could produce for the subject. For example, revealing to a colleague that a coworker “has been dealing with some personal issues” is different from revealing a specific diagnosis, in part because the latter enables the recipient to make inferences and take actions that the subject never intended to authorize.

This analysis reveals a longstanding critique of CI, which is that it presumes that information flows have a stable meaning—designated by the “attribute” parameter—when in fact “data’s meaning has become a moving target” (Horvitz & Mulligan, 2015) because its function depends on the other information it is combined with. Contextual Integrity may be clinging to general, socially understood intuitions about the conventional semantics of information, and trying to preserve those expectations, even though in reality, data is flowing “up and down the food chain” of inference in real time (Nissenbaum, 2019). This is a mismatch between theory and practice. Furthermore, some information norms depend on the granularity or specificity of information, not just the broad category of information. For instance, sharing that a patient “has a chronic condition” with an employer may be appropriate when requesting workplace accommodations, but sharing that the patient “takes 20mg of Lexapro daily for Generalized Anxiety Disorder” violates the norm — not because the information type changed, but because the granularity exceeded what the context required.

Some prior work has addressed the instability of data’s meaning by attempting to assert that the “attribute” property should refer to the information provenance, not what it contributes to inference (Nissenbaum et al., 2018). Benthall (2019) argues that this notion of information semantics should be replaced with a theory of “situated information flow” that sees information flow as causal and embedded in a sociotechnical system that determines what each actor can infer from data as it flows. This way of modeling sociotechnical systems has been applied to the problem of CI-inspired methods for tuning differential privacy parameters (Benthall & Cummings, 2026).

An open problem for CI is whether and to what extent detailed modeling of social context is necessary to either determine legitimate contextual norms, or comply with them once they are understood. It is possible that AI systems enacting CI norms may require models of the other party’s knowledge and inferential capability. CI-compliant multi-agent interaction may require causal world modeling of the socio-technical system and the capacities of the agents within it.

C.4 Further evaluation of technical change

CI can be seen as a normative theory of evaluating technical change. O’Neill (2022) proposes an extension of CI to technology evaluation beyond privacy. This extension engages most with the sense of “entrenched practices” in a context (including but not limited to information flows), and whether and how they relate to CI’s teleological elements: individual ends, and shared goals and values which could relate to personalization.

Furthermore, other attempts to generalize CI away from privacy may lead to greater synthesis with other frameworks for institutional analysis and design (Ostrom, 2019; Frischmann et al., 2014; Shvartzshnaider et al., 2022).

D Review and Taxonomy of Contextual Integrity Benchmarks

A wave of recent datasets and benchmarks has either explicitly attempted to quantify whether LLMs and agents can adhere to CI, or evaluated their utility in operationalizing some aspects of CI. We first describe the attributes of each CI benchmark or dataset that we later use to analyze and compare them (§D.1, Table 1). §D.2 then performs a comparative analysis of the various benchmarks across these attributes (Table 2).

D.1 Analysis framework

We propose the following features to help us analyze each benchmark:

- **Contextualization:** Evaluates what contextual information is provided within evaluation scenarios to make information types and other CI parameters (e.g., sender, receiver, information subject) contextual, in the sense that the same parameters may have different implications depending on the scenario’s context. If so, how richly contextualized is the evaluation scenario (e.g., does the context include a purpose for the information flow, multi-faceted personas, or actors with domain-specific roles)?
- **Transmission principles:** Evaluates the complexity of transmission principles covered by the benchmark. Simple transmission principles are generally context-independent and/or do not influence or specify further data-sharing constraints. For example, the information is appropriate in the given context or is necessary to achieve the specified scenario purpose. On the other hand, complex transmission principles are context-specific and/or specify special conditions placed on the present or subsequent transmissions. For example, data is shared under an NDA or patient-doctor or client-attorney confidentiality; data is compelled to be shared under a subpoena or a domain-specific regulation; data is part of a commercial exchange; or some form of reciprocity is involved.
- **Ambiguity:** Captures whether the information flows in the benchmark scenarios may be appropriate in some contexts and inappropriate in others, and whether such ambiguity is explicitly represented in the benchmark design (e.g., by providing multiple scenarios involving the same information flow under different contexts), or instead arises implicitly.
- **Norms:** Analyzes the sources of informational expectations covered by the benchmark. Typical sources include legal norms (e.g., regulations, case law), societal norms (e.g., social expectations, cultural norms), and organizational norms (e.g., company policies, industry standards).
- **Realism:** Measures how closely benchmark data reflects real-world distributions and scenarios. Data can range from purely synthetic data, to synthetic data grounded in real relevant datasets or (simplified) real-world data.
- **Annotation:** Characterizes how the annotations associated with each scenario in the benchmark were obtained. Annotations typically include information about what is appropriate to reveal in a scenario and what is necessary to share in order to achieve the scenario’s purpose. The annotations can be fully synthetic (e.g., generated by a language model, sampled from a distribution) or obtained from human raters, who in turn can be expert or non-expert depending on their knowledge of the domain and/or the CI framework.
- **Explanation:** Captures whether annotations provide only target outputs/labels or also include explicit justifications for what is appropriate or necessary in a scenario, which can be evaluated against agent or model reasoning.
- **Metrics:** Describes what evaluation signals are considered when assessing benchmark performance (e.g., leakage and utility) and what types of scoring rules are used to compute such metrics (e.g., deterministic, heuristic, or judge-based methods). Deterministic scoring methods are those that can be computed with a fixed algorithmic procedure. Heuristic scoring methods involve some form of approximation or rule-of-thumb that may not be strictly deterministic. Judge-based scoring methods involve the use of an

LLM-as-a-judge to compare model outputs against reference annotations and assign scores based on the quality of the match, which may involve some degree of subjectivity or variability in scoring.

- **Granularity:** Indicates whether benchmarks include a taxonomy for fine-grained error analysis (e.g., complexity tiers, domains, failure modes) instead of only coarse aggregate labeling.
- **Modality:** Specifies which data types and interaction settings are covered, including text and non-text modalities (e.g., images, audio, video), multi-turn behavior, and whether tool-mediated information access in agentic settings is part of the evaluation.

Table 1: Analysis framework features, category sets, and category explanations.

Feature	Category	Category explanation
Contextualization	None	Provides no explicit contextualization.
	Simple	Provides some form of contextual information (e.g., purpose, roles).
	Rich	Provides rich contextual information (e.g., long task descriptions, descriptive multi-faceted personas).
Transmission principles	None	Transmission principles are not explicitly modeled.
	Simple	Only simple transmission principles are considered.
	Complex	Some complex transmission principles are considered.
Ambiguity	No	All scenarios are clearly not ambiguous.
	Implicit	Upon analysis some scenarios could be considered ambiguous.
	Explicit	The benchmark explicitly models some degree of ambiguity in the proposed evaluation.
Norms	Not specified	The source of the norms is not specified.
	Legal	Norms grounded in regulation or case law are included.
	Societal	Norms arising from societal considerations are included.
	Organizational	Norms that apply to specific organizations are included.
	Other	Other types of norms are included.
Realism	Synthetic	Data is fully synthetic.
	Grounded	Data is synthetically generated based on some inputs from real-world data.
	Real	Data is sourced directly from real-world scenarios.
	Other	Data does not fit into the other specified categories.
	Not specified	The source or nature of the data is not specified.
Annotation	None	No annotation is provided.
	Synthetic	Annotation is generated synthetically.
	Non-expert rater	Annotation is provided by non-expert raters.
	Expert rater	Annotation is provided by expert raters.
Explanation	No	No explanation for the annotations is provided.
	Yes	Explanations that can be used to evaluate model reasoning are provided.

Continued on next page

Feature	Category	Category explanation
Metrics	Leakage	Metrics that measure information leakage or non-CI compliance are included.
	Utility	Metrics that measure task performance or usefulness are included.
	Deterministic	Some metrics are fully deterministic.
	Heuristic	Some metrics are based on heuristics.
	Judge-based	Some metrics rely on LLM-as-a-judge.
Granularity	Yes	Describes the main breakdown axes present in the benchmark.
	No	The benchmark can only be evaluated at a coarse level.
Modality	Text	The benchmark includes scenarios with text-based data.
	Image	The benchmark includes scenarios with image-based data.
	Video	The benchmark includes scenarios with video-based data.
	Audio	The benchmark includes scenarios with audio-based data.
	Multi-turn	The benchmark includes scenarios with multi-turn interactions.
	Tool use	The benchmark includes scenarios involving agentic tool-use.

D.2 Comparative analysis

We next compare the various CI benchmarks across the attributes of our analysis framework (summary in Table 2).

Contextualization. Benchmarks classified as possessing *simple* contextualization, such as ConfAIde, CI-Bench, LLM-CI, AgentDAM, and CIMemories, typically ground their information flows in basic, functional metadata. Context is predominantly established by assigning straightforward domain-specific roles, such as a doctor, sibling, or financial advisor, and defining explicit, goal-oriented purposes, like applying for a loan or providing commercial assistance. These frameworks avoid generating multi-faceted sociological profiles, instead embedding context through operational necessity, brief user instructions, or concise, structured Likert-scale questions.

Conversely, benchmarks exhibiting *rich* contextualization construct intricate, multi-layered environments that move beyond basic labels to simulate realistic human complexity. This approach relies heavily on detailed user personas, which frequently feature extensive key-value demographic attributes, complex transaction histories, and specific narrative goals. Rather than evaluating isolated variables, these benchmarks procedurally expand abstract privacy seeds into elaborate, paragraph-length vignettes or simulated agent trajectories, placing the AI within a highly descriptive multi-actor story or deep functional scenario. Furthermore, methodologies like SPILLage embed multiple synthetic attributes into ongoing multi-turn chat histories or forwarded emails to create dense backgrounds for the agent to navigate.

Among the benchmarks analyzed, several stand out for offering exceptionally rich contextual depth. PrivacyBench introduces a highly dynamic environment by simulating entire communities with evolving user attributes, such as occupation or location, and time-bound relationships that shift throughout the evaluation. ConfAIde+ advances narrative depth by incorporating the psychological states of actors, such as feelings of guilt or pressure, alongside detailed professional histories. Additionally, PrivacyLens and its live variant PrivacyLens-Live distinguish themselves by translating expressive textual vignettes into highly detailed simulated agent trajectories, complete with system logs, internal scratchpads, and agent cards that strictly detail interpersonal relationships. Finally, ConVerse provides substantial complexity by integrating lengthy email threads and up to 184 distinct natural-language planning options within its simulated commercial ecosystems.

Transmission principles. Benchmarks categorized with *simple* transmission principles predominantly rely on binary appropriateness gates, such as functional necessity and data minimization. Frameworks including AirGapAgent, “Form-filling agents,” AgentDAM, CIMemories, and SPILLage evaluate information flows based on whether a specific data attribute is strictly necessary or functionally relevant to achieve an immediate operational goal. Other benchmarks in this category focus on the modality of communication or straightforward social boundaries: PrivacyLens, PrivacyLens-Live, and PrivacyLens+ utilize basic communication mechanisms like sending an email or direct message rather than invoking nuanced legal conditions. Similarly, ConfAide and PrivacyBench rely on standard confidentiality expectations, general incentives, and static allow/deny rules tied to specific confidant lists or temporal validity. Across these benchmarks, complex, conditional mechanisms such as non-disclosure agreements, reciprocal sharing, and downstream data deletion are intentionally excluded.

In contrast, benchmarks evaluating *complex* transmission principles move beyond static, context-independent rules to integrate highly conditional and legally binding sharing mechanics. Frameworks such as CI-Bench, MPCIBench, and ConfAide+ introduce structural constraints that require models to navigate explicit user consent requirements, formal sender and recipient authorization limits, and professional non-disclosure provisions. Furthermore, LLM-CI expands this complexity by systematically varying transmission conditions to include diverse situational overrides, such as advertising use, specific privacy policy allowances, and emergency situations.

Among the frameworks reviewed, PrivaCI-Bench and ConVerse stand out for offering highly sophisticated and realistic transmission mechanisms. PrivaCI-Bench establishes rigorous statutory realism by explicitly decomposing complex legal mandates, such as subpoena compliance, commercial exchanges, and medical confidentiality, into functional operational factors like purpose and explicit consent. ConVerse, on the other hand, requires an advanced application of transmission principles by moving entirely away from simple binary allow/deny outcomes. Instead, it demands that agents perform dynamic generative data abstraction, forcing models to parse context to share generalized functional requirements rather than raw personal details depending on the specific scenario constraints.

Ambiguity. It is important to note at the outset that a significant number of benchmarks, including ConfAide, PrivacyLens-Live, PrivaCI-Bench, CIMemories, and SPILLage, intentionally exclude ambiguity entirely by filtering out edge cases, enforcing strict deterministic ground truths, or relying on mutually exclusive categories. However, among those that do account for it, several frameworks feature *implicit* ambiguity. In benchmarks like CI-Bench and AirGapAgent, ambiguity is not actively engineered into the scenarios or evaluation methodology, but instead emerges organically during the annotation phase through high-variance or split decisions. Similarly, LLM-CI forces latent norm conflicts structurally by systematically cross-multiplying contextual parameters without writing explicit narrative contradictions, allowing researchers to evaluate the resulting ambiguity mathematically through statistical variance.

Benchmarks categorized with *explicit* ambiguity purposefully design uncertainty and context collapse directly into their evaluation mechanics. “Form-filling agents” captures this by formally quantifying human rater entropy and establishing multiple ground-truth aggregation heuristics, such as relaxed consensus or no veto, to represent differing privacy boundaries. Camber framework Yi et al. (2025b) proposes PrivacyLens+ and ConfAide+ that deliberately evaluate underspecified contexts by systematically altering narrative variables and scale scenarios across a disambiguation spectrum, thereby observing shifts in privacy judgments. Other approaches embed structural traps to force ambiguity resolution: AgentDAM interleaves task-relevant instructions with task-irrelevant sensitive narratives to test disambiguation, PrivacyBench employs semantic indirect probing and temporal relationship shifts, MPCIBench utilizes a pairwise design testing identical visual images in conflicting contexts, and ConVerse introduces a dedicated related-and-useful data tier to create explicit gray areas regarding information granularity.

Among the evaluated frameworks, PrivacyLens+, ConfAide+, and PrivacyBench present the most challenging and realistic ambiguity considerations. PrivacyLens+ and ConfAide+ stand out by making underspecified context their core focus, systematically injecting varying levels of disambiguating context via the Camber framework to rigorously measure the entropy of language model judgments in uncertain environments. PrivacyBench provides a highly realistic challenge by modeling ambiguity temporally, where user relationships

organically shift over time, and behaviorally through indirect conversational probing, which mimics realistic social engineering attempts to induce inappropriate data leakage by shifting semantic topics. Finally, ConVerse offers a highly nuanced approach by requiring generative data abstraction rather than binary choices, forcing models to actively navigate the explicit gray area of determining exactly how much granular detail is functionally useful versus overly private in complex commercial ecosystems.

Norms. Across the analyzed frameworks, most benchmarks incorporate societal norms, some integrate legal and organizational frameworks, and a few rely entirely on purely statutory or alternative normative sources. Societal norms are frequently derived from crowdsourced human intuitions meant to reflect everyday North American expectations. Benchmarks such as ConfAide and CIMemories also source societal norms from established academic literature, surveys like Westin’s privacy archetypes, and Pew Research data. Alternatively, SPILLage defines societal norms through the implicit privacy expectations users hold regarding data minimization when delegating tasks. Legal norms, when present, are generally extracted directly from major regulatory frameworks, including U.S. laws like HIPAA and FERPA, or international statutes like the GDPR, PIPEDA, and the EU AI Act. Organizational norms are sourced from professional boundaries, encompassing institutional communication rules, corporate ethics, AMA medical codes, and hierarchy-based disclosure restrictions. Lastly, ConVerse represents a unique “other” category, where norms are not derived from legal or societal standards, but instead emerge implicitly from domain-specific semantic relationships within simulated commercial ecosystems.

The codification of these norms within the benchmarks reveals several distinct methodological patterns. Benchmarks like AirGapAgent, CI-Bench, and SPILLage codify norms loosely as informal, common-sense directives, guiding the agent to evaluate appropriateness based on general socio-cultural expectations or basic necessity rather than strict rules. In contrast, “Form-filling agents” utilizes a hierarchical codification strategy: it establishes baseline appropriateness through societal consensus, but explicitly overrides these human annotations with strict legal statutes when a scenario touches regulated contexts. Benchmarks focused heavily on legal realism, such as PrivaCI-Bench, codify norms deterministically by translating complex real-world litigation histories, corporate privacy policies, and compliance questionnaires into strict permit or prohibit baseline categories. A highly distinct codification approach is seen in ConVerse, which moves away from explicitly defining external rule sets entirely; instead, it codifies norms through the natural language constraints of specific planning tasks, forcing the model to dynamically infer the appropriate boundaries.

For the broadest normative coverage, PrivacyLens and PrivacyLens-Live stand out by successfully integrating legal, societal, and organizational norms within unified evaluation suites. They achieve this comprehensive scope by synthesizing U.S. privacy regulations, professional organizational ethics codes, and crowdsourced human expectations of interpersonal communications into their generated scenarios. Regarding the most realistic codification of strictly legal norms, PrivaCI-Bench is exceptional due to its reliance on authentic professional court judgments, constitutional boundaries tracked by the ACLU, and official compliance frameworks rather than synthetic interpretations. Finally, ConfAide, ConfAide+, and AgentDAM present highly realistic operationalizations of organizational norms; ConfAide and ConfAide+ test peer-to-peer workplace confidentiality, while AgentDAM grounds its normative evaluation in functional corporate software development environments to enforce professional boundaries.

Realism. Benchmarks classified under the *synthetic* realism category rely entirely on generative models or programmatic templates to construct their evaluation environments, lacking direct grounding in empirical datasets. Frameworks such as CIMemories, PrivacyBench, and ConVerse utilize frontier large language models, including gpt-oss-120B, Gemini 2.5 Flash-Lite, and Claude Sonnet 4, to probabilistically generate user personas, digital footprints, and multi-turn conversational histories. Other benchmarks utilize synthetic generation to achieve structural scale or narrative depth; for instance, the primary dataset in LLM-CI consists of 6,912 information flows built via the combinatorial expansion of contextual parameters, while ConfAide+ transforms abstract privacy-violating seeds into dense narrative stories. Although SPILLage executes its tasks on live web platforms, its underlying user personas and shopping prompts remain completely synthetically generated. Among these, PrivacyBench and ConVerse present particularly sophisticated synthetic environments by generating complex, evolving community social graphs and extensive, multi-turn planning interactions, respectively.

The *grounded* category represents a hybrid methodological approach where synthetic data generation is strictly constrained by, or built upon, real-world statistical distributions, structural templates, or empirical human seeds. Benchmarks in this tier often utilize crowdsourcing or regulatory texts to establish baseline scenarios, which are subsequently expanded into evaluation items by language models. For example, AirGapAgent generates synthetic demographic profiles that adhere to U.S. Census data to ensure statistical plausibility, ConfAIde draws from the Pew Study and a legal study of human privacy expectations, while CI-Bench extracts structural characteristics from publicly available real-world dialogues. The PrivacyLens family (PrivacyLens, PrivacyLens-Live, PrivacyLens+) grounds its synthetic vignettes in actual U.S. privacy regulations, academic literature, and human-crowdsourced starting parameters. MPCIBench and AgentDAM stand out for integrating grounded data with sophisticated modalities; MPCIBench firmly grounds its synthetically generated stories in real-world visual data and human-annotated labels from the existing VISPR dataset, and AgentDAM expands human-authored narrative seeds into functional scenarios tested on live software replicas.

The utilization of purely *real* data within Contextual Integrity benchmarks is exceptionally rare, likely due to the inherent privacy risks and ethical complications of handling unredacted, sensitive human data in evaluation datasets. Consequently, only PrivaCI-Bench incorporates a distinct classification of real-world data within its methodology. Instead of relying on simulated social interactions or synthetically generated personas, PrivaCI-Bench achieves high statutory realism by directly importing authentic legal and corporate documentation. Specifically, the benchmark’s HIPAA, GDPR, and ACLU subsets rely on highly realistic data extracted from historical professional court judgments, active regulatory enforcement trackers, and published enterprise privacy policies. This direct integration of real-world litigation and compliance data provides the most legally sophisticated and historically accurate grounding for evaluating normative transmission principles among all analyzed frameworks.

Annotation. Synthetic annotations across the benchmarks predominantly utilize large language models or strict programmatic rules to automatically generate ground-truth labels at scale. Benchmarks such as CIMemories and AirGapAgent rely entirely on frontier models to simulate human privacy personas or generate labels via multi-step prompting procedures. Alternatively, frameworks like LLM-CI, PrivacyBench, and AgentDAM employ structural self-annotation, where ground truth is intrinsically established by the data’s position within a multi-dimensional tensor or predefined sensitive constraints. PrivaCI-Bench also utilizes a highly deterministic synthetic approach specifically for its EU AI Act subset; because the legislation is new and lacks established case law, the benchmark generates labels upfront by exhaustively mapping synthetic scenarios to an official compliance checker questionnaire. Notably, LLM-CI, PrivacyBench, and PrivaCI-Bench demonstrate particularly rigorous synthetic methodologies by structurally embedding ground truth directly into the dataset’s generative architecture or deterministic rule sets, minimizing subjective LLM evaluation drift.

Annotations derived from non-expert raters heavily rely on crowdsourcing platforms to capture broad, consensus-based societal norms. Frameworks including ConfAIde, PrivacyLens, and PrivacyLens-Live utilize platforms such as Amazon Mechanical Turk and Prolific to gather baseline privacy expectations and initial seed data from everyday users. Similarly, the ConfAIde subset of LLM-CI utilizes external paid raters to provide Likert-scale scores or establish empirical human consensus. “Form-filling agents” stands out for its methodological rigor in this category; rather than forcing absolute agreement, it actively quantifies the variance in annotator disagreement (rater entropy) and calculates multiple aggregation heuristics, such as relaxed consensus or no veto, to formally model ambiguous privacy boundaries.

Expert rater annotations are intended to introduce specialized knowledge to ensure the normative and legal accuracy of the evaluation datasets. In several frameworks, such as CI-Bench, PrivacyLens, PrivacyLens+, ConfAIde+, and ConVerse, the internal research teams and authors themselves serve as the expert panels to manually validate outputs or provide definitive labels. However, the validity of classifying benchmark authors as independent experts is debatable; while they possess deep familiarity with their own mechanistic designs, they generally lack formal credentialing in privacy law, compliance, or sociology, which risks introducing insular bias. In stark contrast, PrivaCI-Bench exhibits the most rigorous and authentic methodology within this category. Rather than relying on internal researchers for initial legal interpretations, PrivaCI-Bench

derives its expert annotations directly from historical professional court judgments and established privacy policies for its HIPAA, GDPR, and ACLU subsets. Furthermore, to ensure the highest standard of domain-specific accuracy, these real-world expert annotations were subsequently validated by a human review team that explicitly included law students to rectify any parsing errors.

Explanation. Only a select minority of the evaluated benchmarks incorporate qualitative explanations or justifications into their datasets or evaluation pipelines. Across these specific frameworks, explanations predominantly originate from synthetic sources, utilizing large language models to either generate foundational rationales during dataset construction or to produce intermediate chain-of-thought reasoning during automated evaluation. For example, AgentDAM relies on an automated LLM-as-a-judge to provide natural language justifications detailing why specific execution traces constitute privacy leaks. Similarly, ConVerse utilizes synthetic pre-generated ground-truth ratings that include detailed textual justifications aligning options with user constraints, alongside crafted semantic rationales for simulated attacks.

Certain benchmarks highlight more nuanced and realistic approaches to generating explanations by anchoring them in specific societal harms or rigid structural requirements. PrivacyLens is particularly notable for embedding a specific parameter that justifies its ground-truth labels by explicitly articulating the exact legal or sociological harm associated with a data flow. Building on this, PrivacyLens-Live employs a specialized architecture that requires agents to generate an explicit text-based justification and an information flow extraction for their binary privacy judgments. Additionally, MPCIBench ensures foundational clarity by forcing the creation of an explicitly written justification during its construction phase to firmly ground the underlying transmission principle. Finally, AirGapAgent demonstrates domain-specific nuance by generating and storing detailed textual motivation strings for every label, explicitly outlining the functional necessity of sharing specific data within specialized contexts like healthcare.

Metrics. The evaluation of privacy violations across the benchmarks primarily revolves around measuring *leakage*, which tracks precision-oriented unauthorized disclosures and the exposure of sensitive facts. Methodologies for computing leakage range from strictly deterministic approaches to more semantic, judge-based evaluations. Deterministic pipelines often rely on exact string matching, regular expressions, or single-token parsing against categorical ground truths. While these methods, utilized by “Form-filling agents,” PrivaCI-Bench, and PrivacyLens+, offer highly reliable and reproducible assessments of static outputs, they can be structurally rigid. To capture more realistic semantic evasions and implicit disclosures within complex or multi-turn dialogues, several benchmarks instead employ heuristic text-processing (such as BLEU or BLEURT) or LLM-as-a-judge frameworks. Notably, MPCIBench includes a visual leakage metric for multi-modal evaluations. Alternatively, LLM-CI bypasses strict leakage tracking entirely, measuring normative alignment mathematically by using statistical distance functions to calculate model divergence from an expected multidimensional tensor.

Utility metrics complement privacy evaluations by assessing a model’s ability to successfully share functionally necessary information and complete designated tasks. These recall-oriented metrics ensure that privacy mechanisms do not severely degrade system functionality, explicitly penalizing models for unwarranted withholding or over-refusal. Methodologically, utility is frequently evaluated deterministically to ensure high reliability. For instance, “Form-filling agents” calculates the mathematical fraction of required fields successfully filled, while AgentDAM performs programmatic state checks directly against a server database to confirm task execution. In more fluid, narrative environments, benchmarks like CIMemories and SPILLage leverage LLM judges to semantically parse the generated text, identifying revealed attributes and evaluating overall task-level completeness.

Benchmarks that integrate multiple complementary methodologies offer the most nuanced and thorough metric suites. PrivacyLens provides a highly robust framework by identifying explicit leakage in final tool outputs, calculating a scalar helpfulness score to calibrate for over-refusals, and evaluating static contexts via deterministic QA probing alongside a few-shot LLM classifier. AgentDAM demonstrates exceptional methodological realism by evaluating utility through highly reliable programmatic state checks on functional software replicas, while reserving judge-based semantic evaluation strictly for identifying nuanced privacy

leaks. Finally, ConVerse stands out for combining snippet extraction with an LLM-as-a-judge to track attack success rates and utility coverage within complex commercial ecosystems.

Granularity. Across the evaluated frameworks, granularity is most frequently operationalized through the categorization of benchmark-specific features such as application domains, complexity tiers, and distinct operational failure modes. Numerous datasets enable fine-grained failure analysis by slicing scenarios into specific contextual domains, such as healthcare, finance, or travel, allowing researchers to observe domain-specific degradation in model performance. Furthermore, several benchmarks structure their evaluation along strict difficulty or complexity tiers, ranging from isolated parametric seeds to complex multi-turn agent trajectories, to pinpoint the exact threshold where a model’s contextual reasoning breaks down. Granularity is also applied to operational and modality-specific architectures, enabling the isolation of specific execution failures. For instance, PrivacyLens-Live explicitly categorizes whether an agent failed at the judgment phase or during the subsequent action, PrivacyBench distinguishes between retriever-side and generator-side leaks, and multi-modal benchmarks like MPCIBench and SPILLage isolate text-based informational leaks from visual or behavioral navigation leaks.

In relation to core theoretical features, granularity axes are strategically employed to map failures directly back to underlying norms, transmission principles, and context ambiguity. Benchmarks like LLM-CI and PrivacyLens+ perform explicit parametric slicing, manipulating individual contextual parameters (e.g., swapping the data recipient while holding other variables constant) to statistically isolate which specific entity or transmission rule causes normative divergence. Granularity is also used to differentiate between distinct normative frameworks, with PrivacyLens explicitly tracking failures across varying norm sources such as strict U.S. regulations versus general societal expectations. To systematically address ambiguity and complex transmission principles, datasets like “Form-filling agents” and CI-Bench classify targets by underlying data sensitivity, identifiability, and functional need-to-know constraints. Similarly, ConVerse models ambiguity through explicitly defined privacy tiers (e.g., related-and-useful vs. unrelated data) to test how models navigate implicit contextual gray areas.

The most sophisticated axes of granularity highlight critical systemic failures by categorizing the foundational reasoning flaws and adversarial vulnerabilities of AI agents in high-risk situations. Some scenarios in ConfAIde are designed to test different aspects of reasoning like tracking information accessibility, while ConfAIde+ and PrivacyLens+ stand out by utilizing a taxonomy of distinct reasoning codes (such as consent, sender authorization, and purpose), which enables researchers to diagnose exactly which structural transmission principle a model fundamentally misunderstands. Similarly, CI-Bench breaks down errors by functional stage, identifying whether a model failed to extract the context or failed to identify the governing norm. For uncovering high-risk security vulnerabilities, benchmarks that introduce granularity into their threat modeling provide the most robust insights. ConVerse categorizes failures across specific commercial attack vectors like toolkit manipulation, AirGapAgent isolates context-preserving versus context-hijacking adversaries, and PrivacyBench contrasts direct questioning against indirect semantic probing, effectively exposing how sophisticated agents might systemically leak data when subjected to nuanced conversational evasion.

Modality. The predominant modality across the evaluated benchmarks is text, though the structural variability within these text-only datasets enables the assessment of diverse AI capabilities. At the most fundamental level, frameworks like LLM-CI rely on decontextualized, single-turn text strings to evaluate static alignment. A significant cluster of benchmarks, including PrivacyLens+, ConfAIde+, and PrivaCI-Bench, utilizes probing-style prompts where models answer direct, single-turn questions regarding written narrative vignettes or legal scenarios. Expanding into structural diversity, “Form-filling agents” evaluates text-based data payloads and web form descriptions rather than narrative stories. To capture relational dynamics, CI-Bench and ConfAIde depart from single-prompt formats by incorporating multi-turn text data, presenting scenarios as conversational chat dialogues, asynchronous email chains, or synthesized meeting transcripts.

To evaluate more sophisticated autonomous systems, several benchmarks expand beyond text to incorporate visual modalities and agentic capabilities such as multi-turn execution and tool use. Notably, video and audio

modalities are entirely absent from all the evaluated datasets. However, image processing is actively tested: AgentDAM utilizes raw screenshots and Set-of-Marks visual bounding boxes, MPCIBench tests vision-language models in agent-based settings by combining images with text prompts, and SPILLage processes visual screenshots alongside web DOM structures. Furthermore, assessing autonomous agent behavior relies heavily on tool use and multi-turn planning. Benchmarks like PrivacyLens, PrivacyLens-Live, and ConVerse require models to navigate multi-step workflows by actively querying simulated software endpoints and API sandboxes covering productivity and financial applications, among others. Similarly, PrivacyBench requires RAG agents to actively fetch historical data across up to ten conversational turns.

The benchmarks offering the most complex and realistic representations seamlessly integrate multiple modalities with deep environmental interaction. AgentDAM and SPILLage present the most comprehensive multi-modal agent testing environments; both require models to process complex text structures (accessibility trees or DOMs) and images simultaneously while executing long-horizon, multi-turn navigation using live or replica browser automation tools like clicking, typing, and scrolling. Within text-driven agentic systems, PrivacyLens-Live offers exceptional realism by testing multi-step workflows natively using the Model Context Protocol (MCP) and Agent2Agent frameworks. Finally, MPCIBench provides a highly nuanced approach to visual privacy by executing its evaluation through a pairwise design, testing the exact same visual image across different contextual text scenarios to explicitly force vision-language models to navigate multi-modal contextual ambiguity.

Table 2: Comparative feature matrix across CI Benchmarks.

<i>Benchmark</i>	<i>Contextualization</i>	<i>Transmission principles</i>	<i>Ambiguity</i>	<i>Norms</i>	<i>Realism</i>	<i>Annotation</i>	<i>Explanation</i>	<i>Metrics</i>	<i>Granularity</i>	<i>Modality</i>
ConfAId Mireshghalah et al. (2024)	Simple	Simple	No	Societal	Grounded	Non-expert rater; synthetic	No	Leakage; utility; deterministic; judge-based	4 complexity tiers; factorial relation pairs; incentive and information-type factors	Text; multi-turn
AirGapAgent Bagdasarian et al. (2024)	Rich	Simple	No	Societal	Grounded	Synthetic; non-expert rater	Yes	Leakage; utility; deterministic; heuristic	Domains; scenarios; difficulty levels; adversary types	Text
Form-filling agents Ghalebikesabi et al. (2025)	Rich	Simple	Explicit	Societal; legal	Grounded	Non-expert rater	No	Leakage; utility; deterministic	Data key categories: sensitivity, information type, need-to-know, identifiability	Text
CI-Bench Cheng et al. (2024)	Simple	Complex	Implicit	Societal	Grounded	Expert rater; non-expert rater	No	Leakage; utility; deterministic; heuristic; judge-based	Functional stage; domain variance; attribute sensitivity	Text; multi-turn
LLM-CI Shvartzshnaider & Duddu (2025)	Simple	Complex	Implicit	Legal; societal	Synthetic; grounded	Synthetic; non-expert rater	No	Deterministic	Parametric slicing; CI parameter metadata taxonomy	Text
PrivacyLens Shao et al. (2024a)	Rich	Simple	Explicit	Legal; societal; organizational	Grounded	Non-expert rater; expert rater	Yes	Leakage; utility; deterministic; judge-based	Complexity tiers (seed, vignette, trajectory); norm sources (legal, societal, special community)	Text; multi-turn; tool use
PrivacyLens-Live Wang et al. (2025)	Rich	Simple	No	Legal; societal; organizational	Grounded	Synthetic; non-expert rater	Yes	Leakage; utility; deterministic	Failure modes; workflow complexity	Text; multi-turn; tool use
AgentDAM Zhar-magambetov et al. (2025)	Simple	Simple	Explicit	Societal; organizational	Grounded	Synthetic	Yes	Leakage; utility; deterministic; judge-based	Application types; task sub-types; input architectures	Text; image; multi-turn; tool use
PrivaCI-Bench Li et al. (2025)	Rich	Complex	No	Legal	Real; synthetic	Expert rater; synthetic	No	Leakage; deterministic	Domains; difficulty tiers; label classes	Text
PrivacyLens+ Yi et al. (2025b)	Rich	Simple	Explicit	Legal; societal	Grounded	Synthetic; expert rater	No	Leakage; utility; deterministic	CI parameters; reasoning assumption codes	Text

Continued on next page

<i>Benchmark</i>	<i>Contextualization</i>	<i>Transmission principles</i>	<i>Ambiguity</i>	<i>Norms</i>	<i>Realism</i>	<i>Annotation</i>	<i>Explanation</i>	<i>Metrics</i>	<i>Granularity</i>	<i>Modality</i>
ConfAIde+ Yi et al. (2025b)	Rich	Complex	Explicit	Societal; organizational	Synthetic	Synthetic; expert rater	No	Leakage; utility; deterministic	9 reasoning domains/codes	Text
CIMemories Mireshghallah et al. (2026)	Simple	Simple	No	Societal; organizational	Synthetic	Synthetic	No	Leakage; utility; judge-based	9 domains; attribute-level granularity	Text
PrivacyBench Mukhopadhyay et al. (2025)	Rich	Simple	Explicit	Societal	Synthetic	Synthetic	No	Leakage; utility; judge-based	Probing strategies; failure modes; document types	Text; multi-turn; tool use
MPCI-Bench Wang & Zhang (2026)	Rich	Complex	Explicit	Societal	Grounded	Synthetic	Yes	Leakage; utility; deterministic; judge-based	Complexity tiers; multiple domains; visual leakage breakdown; probing-action gap	Text; image; tool use
SPILLage Roh et al. (2026)	Rich	Simple	No	Societal	Synthetic	Non-expert rater	No	Leakage; utility; judge-based	Content vs behavioral; explicit vs implicit	Text; image; multi-turn; tool use
ConVerse Gomaa et al. (2026)	Rich	Complex	Explicit	Other	Synthetic	Synthetic; expert rater	Yes	Leakage; utility; deterministic; judge-based	Domains; privacy tiers; data categories; attack types	Text; multi-turn; tool use