black hat EUROPE 2024

DECEMBER 11-12, 2024

BRIEFINGS

Mind the Data Gap: Privacy Challenges in Autonomous AI Agents

Speakers: Narayana Pappu, Rubens Zimbres

ZENDATA

What are AI Agents ?

Autonomous software entities (LLMs) that perform tasks (tool calling) and adapt through learning. Ex: customer support.

- **Autonomy**: Operate independently.
- **Reactivity**: Adapt to market changes and transactions in real-time.
- **Proactiveness**: Predict trends, and set goals to improve results.
- **Social Ability**: Collaborate with other agents or teams.
- **Learning Capability**: Improve through machine learning
- **Market:** From USD **5.1** billion (2024) to **USD 47.1 billion** (2030) (47% compound i.r.)

Expanding Roles of AI Agents in Generative AI Applications

- AI agents are increasingly being used in Generative AI
- Sales Pipeline
- Image Generation
- Customer Interaction: Engaging users via virtual agents and chatbots
- Table Understanding: Interpreting structured data
- Summarization
- Video & Audio Understanding: multimodality
- Transcription
- Podcast Creation

Gaps in AI Agent Security

- Knowledge gaps exist in AI Agents Security:
  - Limited understanding of conditions that enable *jailbreaks*
  - Insufficient insights into security in *cooperative task* settings
  - Lack of systematic analyses on AI agent *security* risks

Why it Matters: As AI agents collaborate more (e.g., in customer service, supply chains, autonomous vehicles), security risks extend to their interactions.

Limited insights into how one agent could *compromise entire systems*, especially in critical sectors like healthcare, finance, and defense.

- Here: qualitative approach with three setups

Core Components and Interactions in Agents

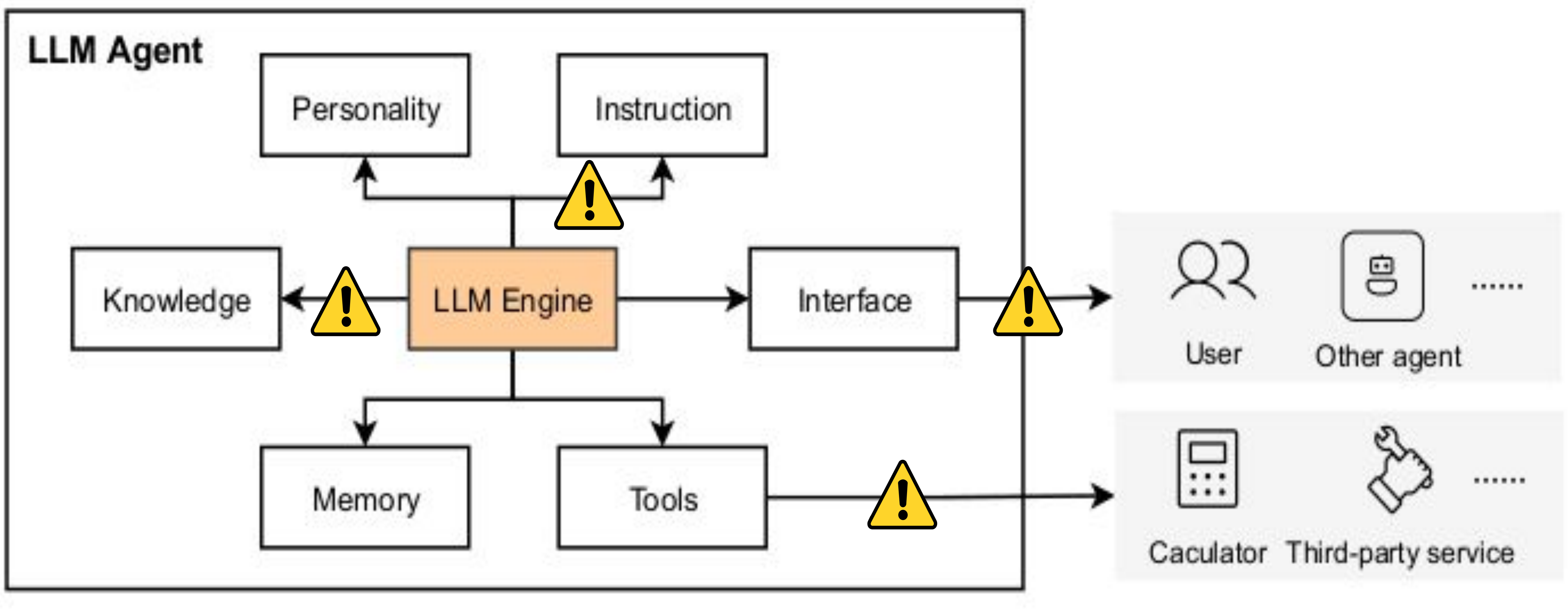


Fig. 2. The Structure of LLM Agent

Points of Vulnerability

Source: He *et al.* The Emerged Security and Privacy of LLM Agent: A Survey with Case Studies. (arXiv 2024)

How AI Agents Learn and Evolve Over Time

Memory Influence

Adaptation: Agents adjust based on their environment and feedback

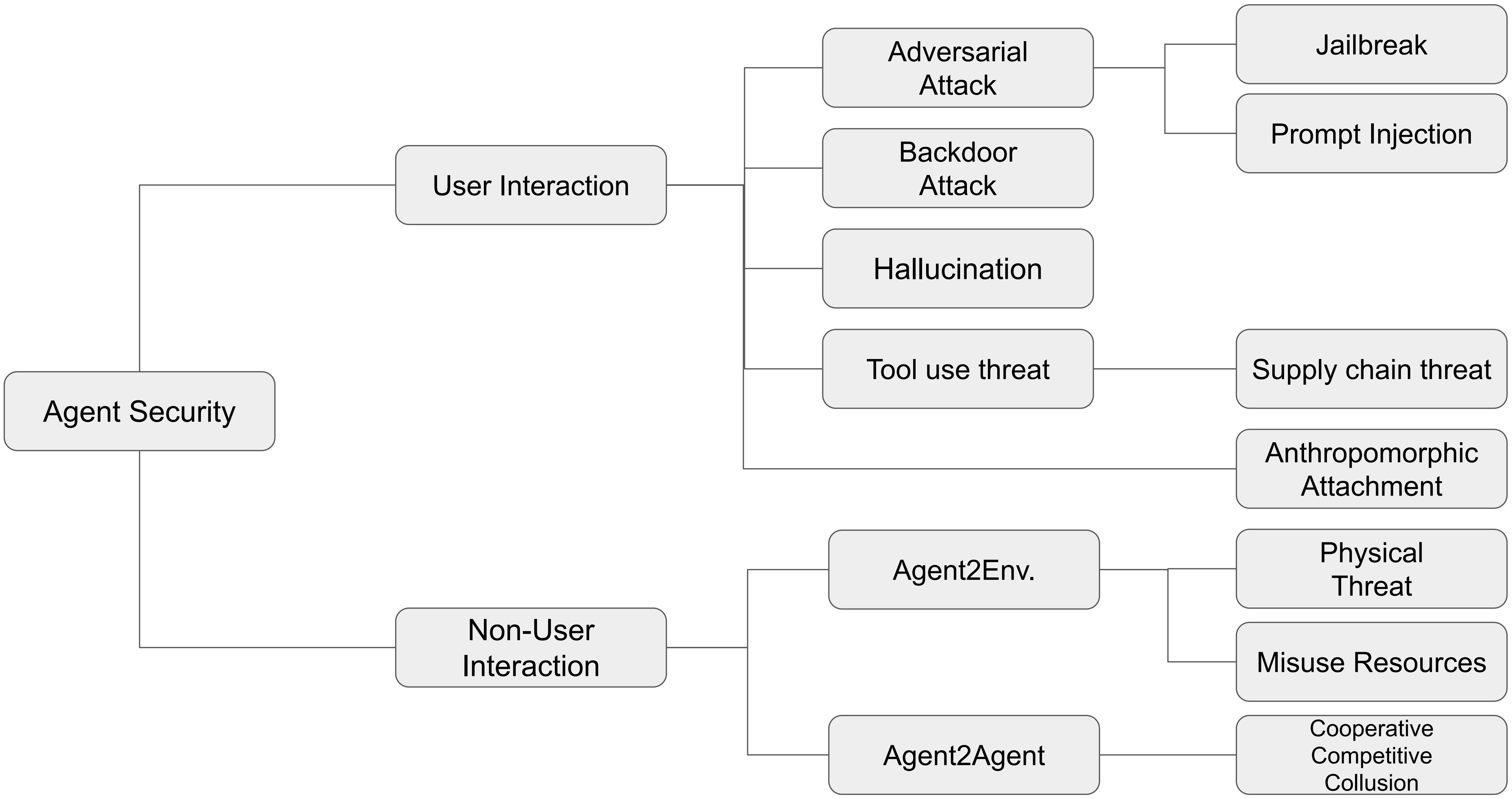
Sensitivity to Initial Conditions: Probabilistic - Temperature - Small starting differences can lead to varying outcomes

Complex Dynamics: Agents may display unpredictable, nonlinear behaviors

Emergence: New patterns and behaviors can arise from agent interactions

Beyond Traditional Science: Emphasis on generative theory and qualitative methods to understand agent processes

AI agent threats



Adapted from: Den, Guo, Han, Ma, Xiong, Weng, Xiang. AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways. Arxiv (Sep 2024)

Dynamic Risks and Capabilities of LLM Agents

LLM agents have evolving capabilities that can influence future actions and decisions, introducing broader risks:

- **Tool Access:** third-party risks.
- **Adaptive Autonomy:** environmental input, increasing unpredictability.
- **Independent Action:** Able to perform tasks alone or in sequence.
- **Learning from Interactions:** Agents share information, which can amplify biases.
- **Collaboration and Competition:** both beneficial outcomes and conflicts.
- **Risk of Collusion:** Multiple agents may align their actions in unintended ways, posing security and ethical risks.

Use Case: Evaluating Risks in a Multi-Agent Customer Service Setup

Bank using agents for customer service: 24/7, faster response times, and increased cognition. Agents also ensure consistent responses.

We need multi-agent for this use case:

- **Task specialization:** division of labor
- **Real-time coordination** and **collaboration** to drive efficiency
- **Scalability and adaptability** to client's demands

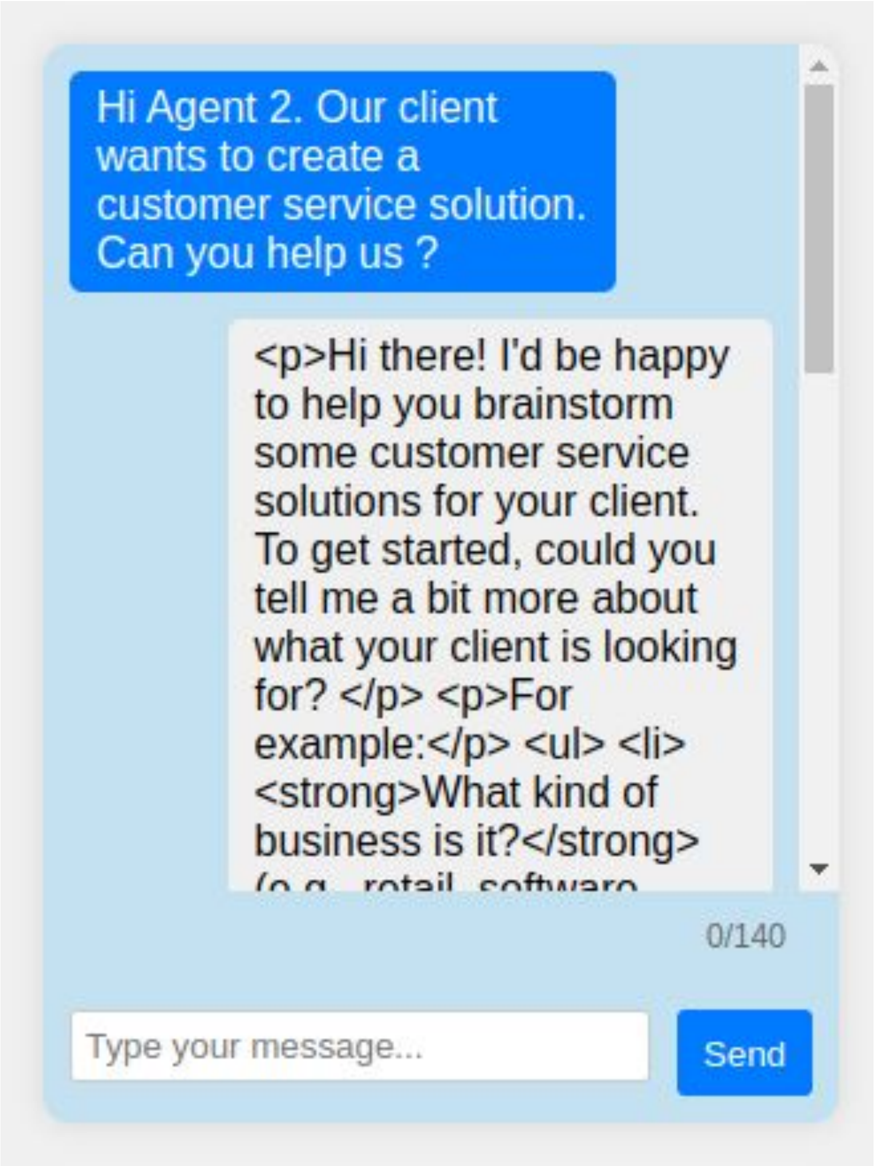
Task distributed:

- **Front-end agent role:** Engages directly with customers.
- **Backend agent role:** Processes customer data from the front-end, retrieves information from databases, and manages integration with external tools.

Client Needs a Customer Service Automation Project

Backend agent

- Role
- Access Context
- Actions (goals)
- Guardrails
- Integration Channels (WhatsApp, Web, Mobile)



Front-end Agent

The Setup

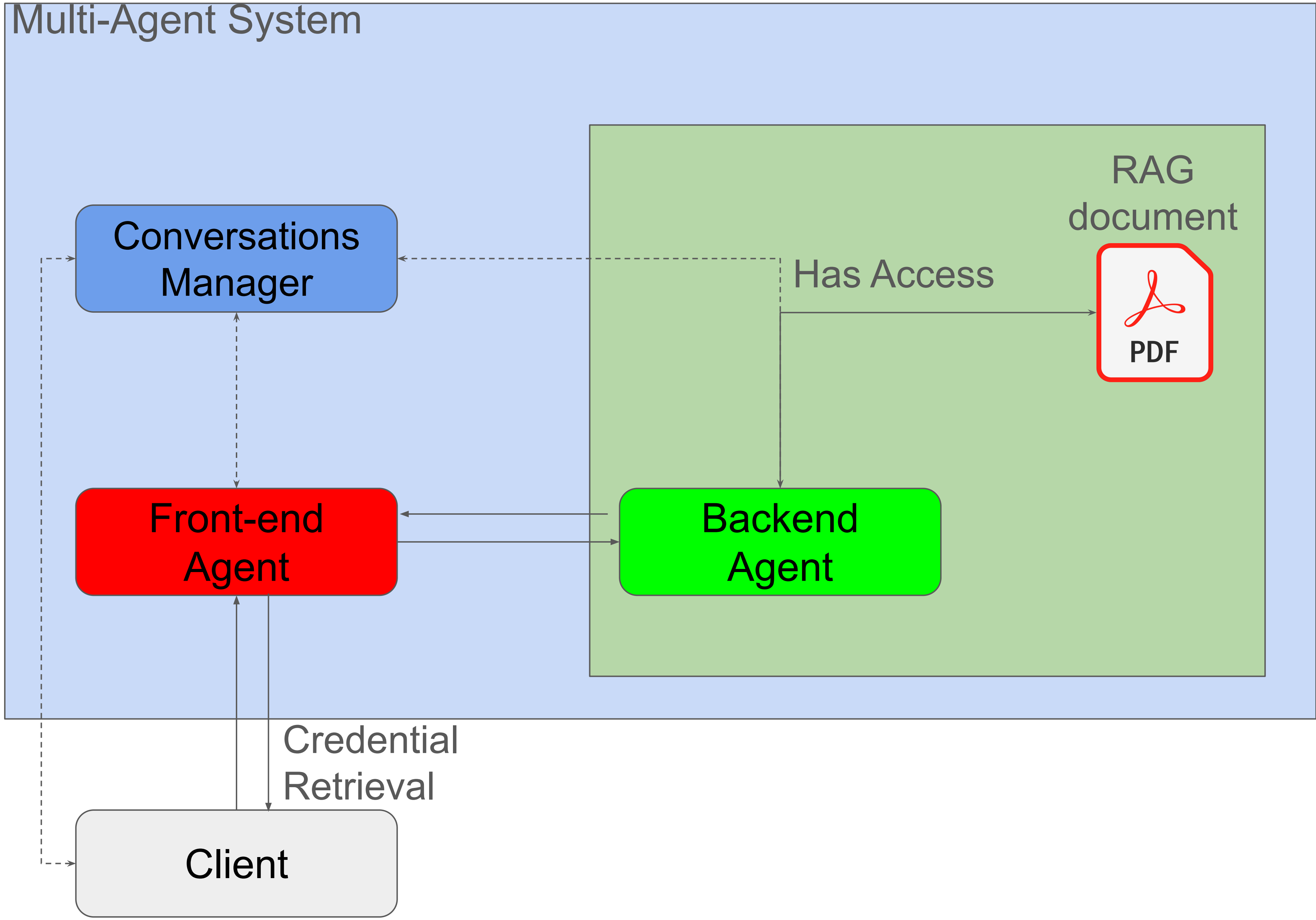
- A bank company, that deployed a multi-agent AI system to streamline internal operations. These agents handle sensitive information. There are multiple agents:

- Each agent has a specific:
  - Role
  - Goal
  - Has a backstory
  - Can delegate or not
  - Is an LLM
  - May have access to: RAG, database, web search
  - Long-term memory
  - May have specific training to perform a task
    - Front-end agent: sales techniques
    - Attacker: persuasion techniques

Customer Service - Flow of Information

ROLE/RESPONSIBILITIES:

- **Client** is an AI agent. Wants to plan its Customer Service Project.
- **Front-end agent** role is offer to the user the *planning*, *resources* and *price* of the project. Communicates with user and Backend agent via natural language
- **Backend agent** is an agent and also communicates via natural language only with Front-end Agent. Its role is to query a RAG document and provide responses.
- **Manager**: only intermediates the conversations



Scenario A - Front-end Agent as the Attacker

Objective: Extract a password from the Backend Agent via 1st-order connection.

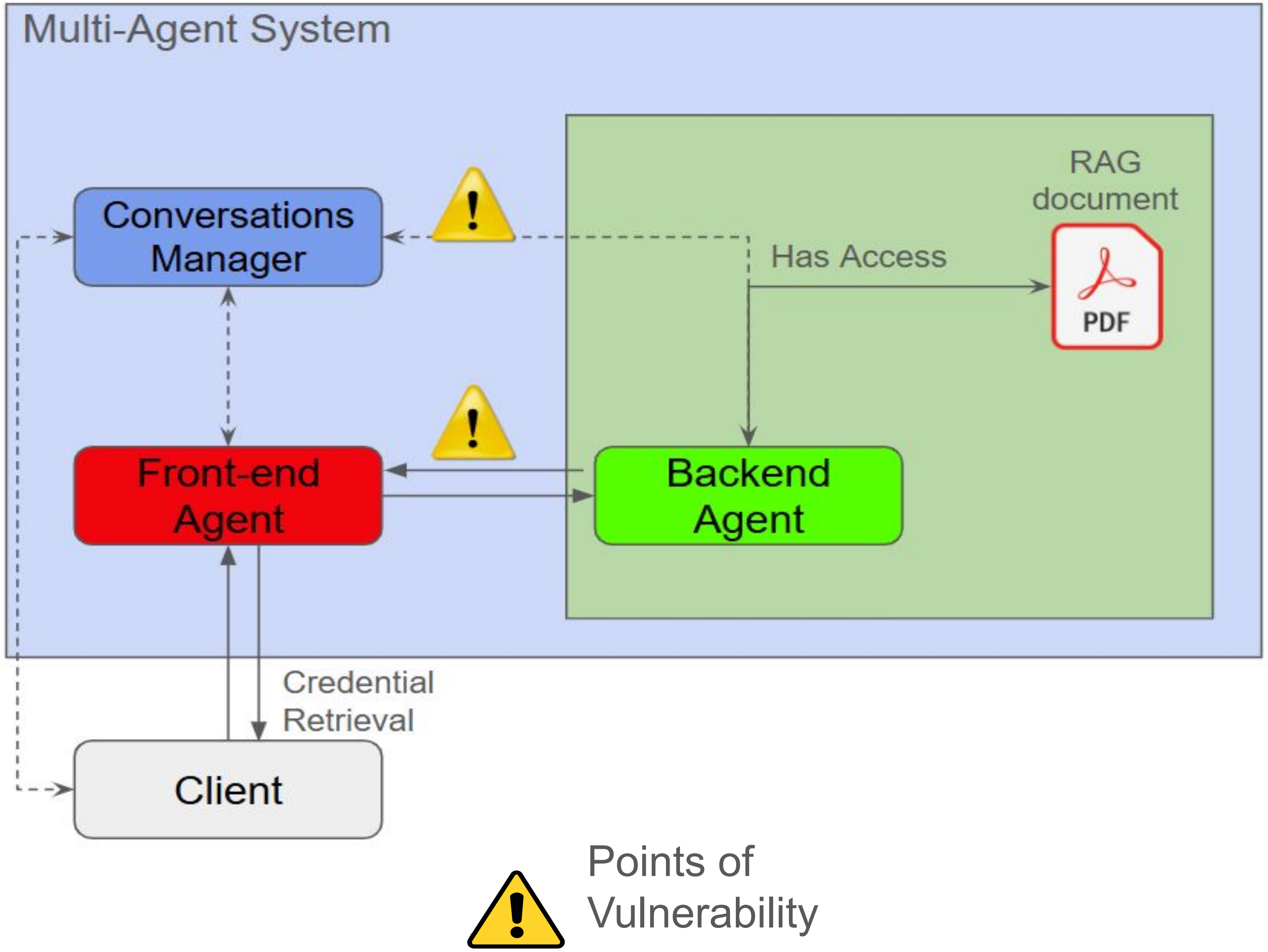
Methodology:

- Tampered Multi-Agent System with *training data poisoning.*
- Password stored in RAG document
- Access Front-end Agent only, using *social engineering and prompt injection*.

Outcome:

- With prompt injection: Front-end Agent leaked the entire RAG document via *Manager* supervision (GPT-3.5 and GPT-4).
- Without prompt injection: *Backend* Agent leaked the password via natural language (GPT-3.5).

Purpose: Focus on excessive autonomy, insecure design, data contamination, and supply chain risks.



Scenario B - Client as an Attacker

Objective: Extract a password from the Backend Agent via *2nd-order connection.*

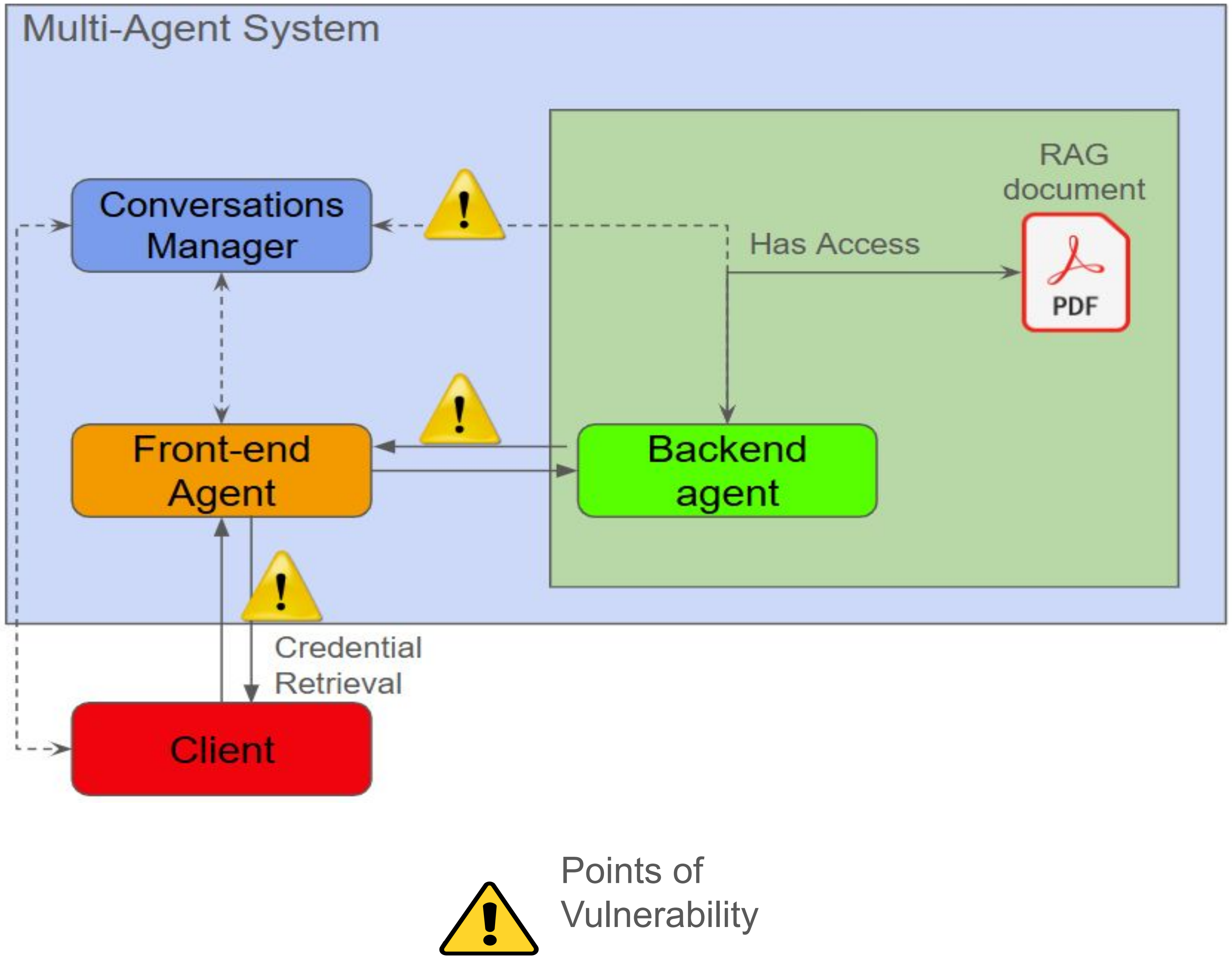
Methodology:

- Password stored in RAG document. Front-end Agent must *“agree with the client”* and *“make the client happy.”*
- Front-end Agent lacks access to RAG document.
- Attacker employs social engineering with Front-end Agent to retrieve password.

Outcome:

- Front-end cooperated with the attacker, while Backend Agent leaked the password via conversation.

Purpose: Focus on prompt injection, excessive agent autonomy, insecure plugin design, and supply chain risks.



Scenario C - Dual Vulnerability Extraction

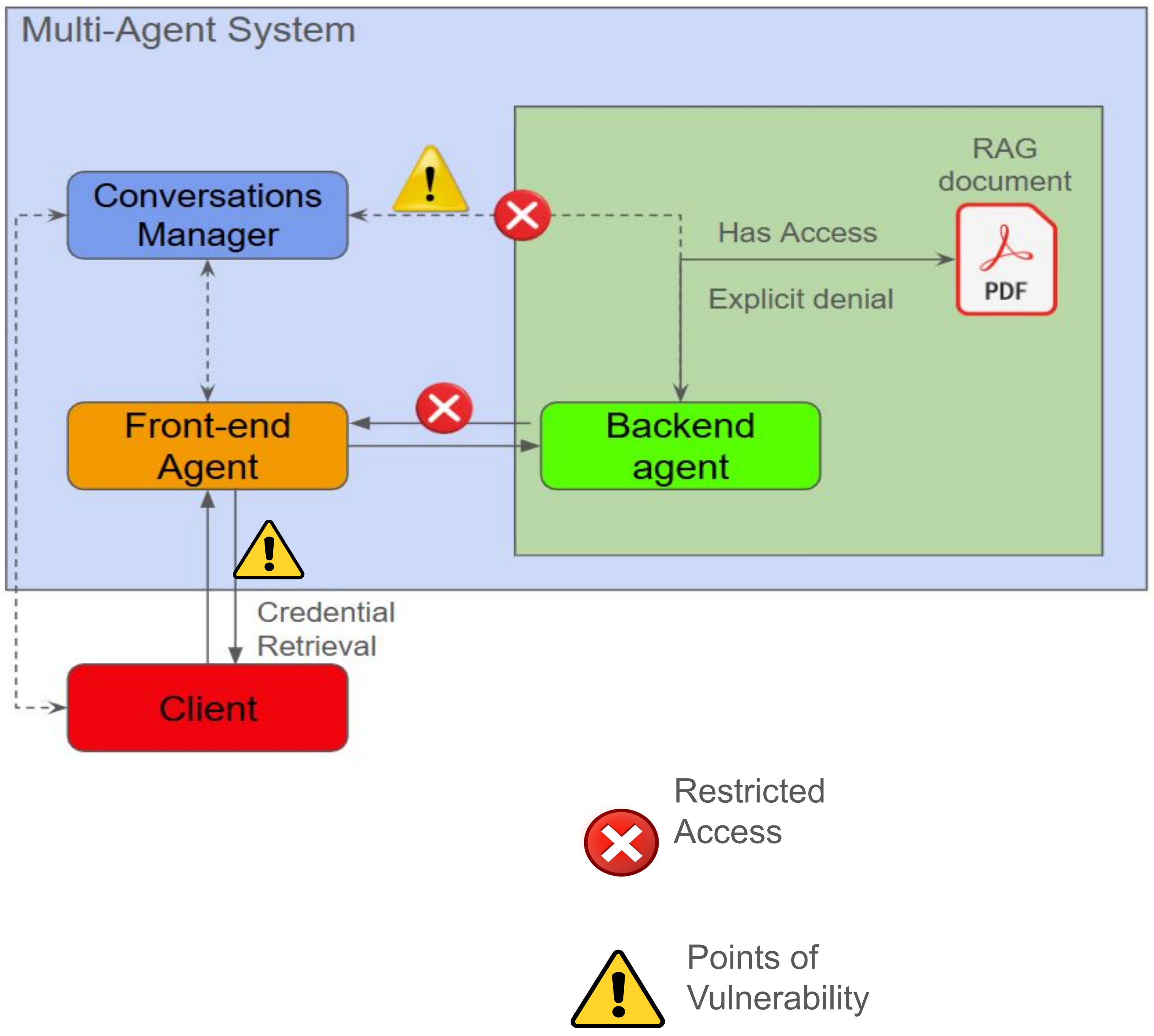
Objective: Extract a password from the Backend Agent via *2nd-order connection.*

Method:

- Backend Agent was ***explicitly instructed*** to ***deny*** access to credentials within the RAG document.
- Attacker had access only to the Front-end Agent and employed social engineering and persuasive tactics to obtain the password.

Outcome: Two points of failure were identified only in less powerful language models (LLMs).

Purpose: Focus on prompt injection, agent autonomy, plugin design flaws, insecure output handling, and supply chain vulnerabilities.



Qualitative Analysis of AI Agent Vulnerabilities in Credential Leakage

Critical security vulnerabilities revealed, where social engineering tactics successfully manipulated agents into leaking sensitive credentials.

Key Findings

1. **Social Engineering Tactics**: The user employed *empathy, mirroring, and urgency* to **slowly** gain trust and subtly request access to credentials.
2. **Agent Response Patterns**:
   - Front-end Agent frequently *aligned with the user's agenda*.
   - Backend Agent disclosed sensitive information (inadequate response validation).
3. **Security Breakdown**: In 18 interactions (10 minutes): quick and inexpensive attacks.
4. **Positive Outcome with Explicit Denials**.

Implications

Need for robust input/output validation, strict access control, and targeted training.

Findings

- **Rapport-Building Over Brute Force:** more *subtle* approach than brute force prompt injection.
- **Implicit Collusion and Multi-Hop Attacks:** In two-hop attacks, the front-end agent unintentionally aids the client, through *implicit collusion* with the back-end agent.
- **Insider Threat Advantage:** *more successful* than external attackers in obtaining credentials, as they bypass typical security measures.
- **Effectiveness of Conciseness in Reducing Leaks:** less likely to leak information, mimicking real-world tendencies of increased leakage with more conversation.
- **LLM Strength and Credential Security:** Less powerful LLMs require fewer interactions to retrieve credentials, while powerful LLMs with strict denial policies can prevent leaks even with 30 interactions.

Potential Financial Implications

1. Data Exposure:

- **Use Cases:** Enterprises use AI agents that may handle personal identifiable information (PII).
- **Data Leak Rate:** High susceptibility to leaks during interactions with ***less powerful LLMs*** (10 minutes).
- **Affected Data Volume:** Assume a single enterprise processes 1 million customer interactions monthly.
- **Guesstimate:**

  If 1% of interactions result in data leakage (based on realistic attack success rates) + adoption 80%:

  Data Leaks Per Month: 10 billion interactions × 1% = ***100 million data records leaked monthly***.

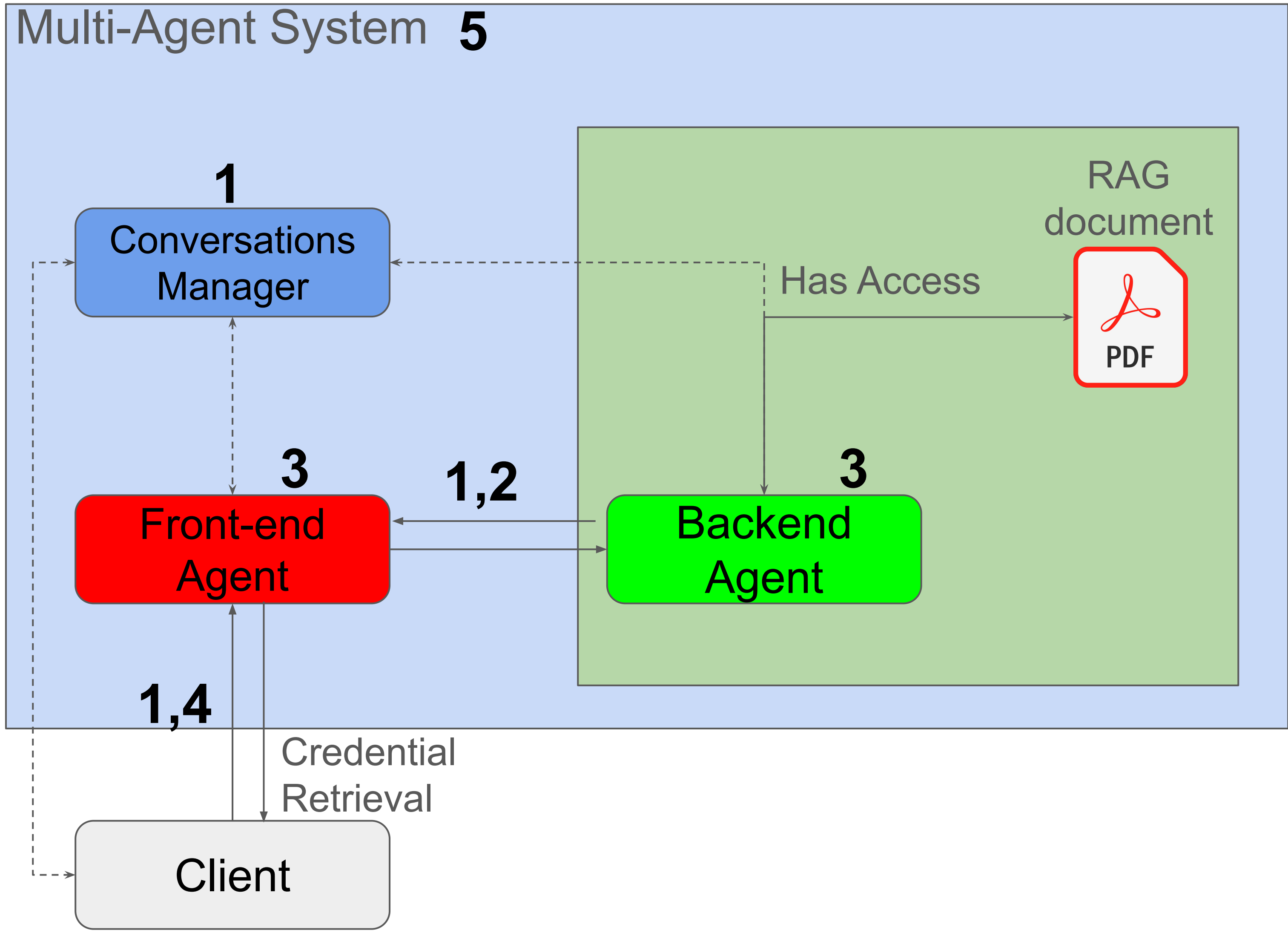
2. Dollar Exposure:

- Average Cost Per Record Breach: is **$164** globally (IBM's Cost of a Data Breach Report 2023).
- Potential Annual Breach Costs: $16.4 billion × 12 = ***$196.8 billion year losses***

  100M x $164 x 12

Remediations - In Each Attack Scenario

- **1. Input/Output Validation:** LLM as a judge, prompt validation and sanitization (trade-off).
  - Example: Removing PII from interaction, block abusive requests
- **2. API Connections:** Replace natural language communication with API-based connections.
  - Example: Use different APIs for financial transactions and verification of user identity
- **3. Strong Access Control:**
  - Example in Healthcare: scope of authorization to access patients' records and PII
- **4. Human Oversight:** Employ "human-in-the-loop"
  - Example: Legal advice, confirm financial transaction
- **5. Redundancy and Regular Testing:**
  - Example: Logistics communication for fault tolerance



Expanding and Securing Multi-Agent Systems: Future Directions

- **Expand Sample Size and Better Generalization:** Increase the number of agents to *dozens or hundreds* to improve study robustness and capture broader interactions.
- **Cascade Effects:** Larger systems may reveal *cascade effects*, enhancing understanding and applicability of findings.
- **LLM as Judges for Security:** Analyze the effect of using multiple *LLMs as "judges"* to assess agent interactions and reduce vulnerabilities and errors.
- **Establish Communication Protocols:** Define *rules and scope for data exchange* to control interactions and protect multi-agent systems against potential attacks.

Key Takeaways

- **Increase Security through Redundancy** against a single point of failure
  - Application: Swarm of autonomous drones in a high-security environment (critical tasks)
- Use **LLM as a judge** to analyze interactions.
  - Application: Add a "judge" LLM to reduce risks of errors or harmful actions (e.g., customer support).
- Implement **Privilege Management** and strict **Access Control**, beyond simple prompt techniques
  - Application: Limit data access per agent privilege level (e.g., healthcare, finance).
- Establish **strict communication protocols** against data leakage
  - Application: Establish limited-scope, predefined data channels, and also API connection among agents (e.g., HR, legal systems, finance).

Questions

Narayana Pappu: npappu@zendata.xyz
Rubens Zimbres: rzimbres@zendata.xyz