UNIVERSIDADE FEDERAL DE GOIÁS / INSTITUTO DE INFORMÁTICA

Segurança em Arquiteturas de Agentes Inteligentes

Aplicação e Refinamento do MAESTRO Framework

Bernardo Aires de Oliveira

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

UNIVERSIDADE FEDERAL DE GOIÁS (UFG)
INSTITUTO DE INFORMÁTICA (INF)

BERNARDO AIRES DE OLIVEIRA

Segurança em Arquiteturas de Agentes Inteligentes

Aplicação e Refinamento do MAESTRO Framework

Goiânia
2025

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC no 1240/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei no 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo dos Trabalhos de Conclusão dos Cursos de Graduação disponibilizado no RI/UFG é de responsabilidade exclusiva dos autores. Ao encaminhar(em) o produto final, o(s) autor(a)(es)(as) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG)

Nome(s) completo(s) do(a)(s) autor(a)(es)(as): BERNARDO AIRES DE OLIVEIRA

Título do trabalho: Segurança em Arquiteturas de Agentes Inteligentes

Aplicação e Refinamento do MAESTRO Framework

2. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador) Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante: a) consulta ao(à)(s) autor(a)(es)(as) e ao(à) orientador(a); b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo do TCCG. O documento não será disponibilizado durante o período de embargo.

Casos de embargo:

- Solicitação de registro de patente;

- Submissão de artigo em revista científica;

- Publicação como capítulo de livro.

Obs.: Este termo deve ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Bernardo Aires De Oliveira**, **Discente**, em 08/02/2026, às 17:46, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fernando Marques Federson**, **Professor do Magistério Superior**, em 13/03/2026, às 11:22, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

BERNARDO AIRES DE OLIVEIRA

Segurança em Arquiteturas de Agentes Inteligentes

Aplicação e Refinamento do MAESTRO Framework

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.

Orientador: Prof. Dr. Fernando Marques Federson

Goiânia

2025

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UFG.

OLIVEIRA, BERNARDO AIRES DE
Segurança em Arquiteturas de Agentes Inteligentes [manuscrito]: Aplicação e Refinamento do MAESTRO Framework / BERNARDO AIRES DE OLIVEIRA. - 2025.
104 f.: 2025

Orientador: Prof. Dr. Fernando Marques Federson
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Instituto de Informática (INF), Inteligência Artificial, Goiânia, 2025.

1. Inteligência Artificial. 2. Agentes Inteligentes. 3. Cibersegurança.

I. Federson, Fernando Marques , orient. II. Título.

CDU 004

BERNARDO AIRES DE OLIVEIRA

Segurança em Arquiteturas de Agentes Inteligentes

Aplicação e Refinamento do MAESTRO Framework

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.

Data da Aprovação: 09 de dezembro de 2025.

Prof. Dr. Fernando Marques Federson
Orientador (INF-UFG)

Prof. Dr. Aldo André Díaz Salazar
Coordenador de TCC do BIA (INF-UFG)

Prof. Dr. Anderson da Silva Soares
Coordenador do BIA (INF-UFG)

Prof. Dr. Sávio Salvarino Teles de Oliveira
(INF-UFG)

BERNARDO AIRES DE OLIVEIRA

Segurança em Arquiteturas de Agentes Inteligentes

Aplicação e Refinamento do MAESTRO Framework

RESUMO

Este Relatório de Conclusão de Curso tem como objetivo reunir os resultados da minha jornada para me tornar um especialista em **Segurança para Agentes Inteligentes**. Uma ilustração e sua narrativa descrevem os períodos de trabalho. Os Apêndices contêm os Termos de Aceite de Entrega e os resultados obtidos durante cada período de trabalho.

Palavras-chave: Inteligência artificial; Agentes inteligentes; Cibersegurança.

ABSTRACT

This Course Completion Report aims to bring together the results of my journey to become an expert in **Security for Intelligent Agents**. An illustration and its narrative describe the work periods. The Appendices contain the Delivery Acceptance Terms and the results obtained during each work period.

Keywords: Artificial intelligence; Intelligent agents; Cybersecurity.

Goiânia

2025

Minha Jornada

Bernardo Aires de Oliveira

Especialista em:
Segurança para Agentes Inteligentes

MINHA JORNADA

Nome: **Bernardo Aires de Oliveira**

Especialidade: **Segurança para Agentes Inteligentes**

Objetivo deste documento

Durante o processo da disciplina Residência em IA[1], foram gerados diversos resultados na construção da minha especialização. A cada semana, um conjunto de resultados foi formalizado por um Termo de Aceite de Entrega e avaliado por uma banca, considerando o planejado e o realizado para o período. Este documento tem como objetivo descrever esses resultados obtidos, fazendo referência aos Termos de Aceite de Entrega e seus documentos associados.

Minha Jornada

Minha Jornada teve início na **Semana 1**, quando dediquei meu tempo a definir de forma clara qual seria o tema central da minha especialização. A partir do mapeamento das áreas apresentadas no CSCE 2025, percebi que várias linhas de pesquisa se conectavam diretamente com assuntos que já despertavam meu interesse, especialmente aquelas relacionadas a sistemas distribuídos, proteção de dados e metodologias de detecção de intrusão. Essa convergência me levou a escolher Segurança em Agentes Inteligentes como o eixo principal do meu estudo.

Nessa fase, também iniciei um levantamento dos artigos e conteúdos basilares, buscando compreender os fundamentos que estruturam o campo e como os conceitos de agentes autônomos, integridade de dados e sistemas auto-adaptativos se relacionam na prática. Esse estudo serviu como ponto de partida para todas as etapas seguintes da Residência e está registrado detalhadamente no **Apêndice 1**, que reúne o Termo de Entrega e os materiais que sustentaram essa primeira semana de investigação. Ao avançar para a **Semana 2**, mergulhei mais profundamente nos textos selecionados, organizei conceitos, e

[1] Dez Semanas, entre setembro de 2025 e dezembro de 2025.

comecei a estruturar um Glossário Técnico que reunisse os termos fundamentais do domínio. Essa etapa foi crucial porque me permitiu criar uma base conceitual sólida, apoiada tanto em artigos quanto em materiais do OWASP, que ajudou a distinguir vulnerabilidades clássicas daquelas que emergem especificamente em sistemas de agentes.

Na **Semana 3**, orientei meus estudos para responder perguntas e esclarecer conceitos essenciais, especialmente sobre as diferenças práticas entre segurança de software, segurança de LLMs e a ainda mais complexa segurança de agentes autônomos. Foi nessa fase que consolidei minhas primeiras reflexões próprias sobre superfícies de ataque, comportamento emergente e riscos amplificados em interações multiagentes, sintetizando tudo isso em documentos e análises que compõem o **Apêndice 2**.

As **Semanas 4 e 5** foram marcadas por uma mudança importante, visto que deixei de apenas mapear o tema e passei a lidar diretamente com frameworks e metodologias de segurança específicas para sistemas agênticos. Nesse período, mergulhei no documento “Agentic AI – Threats and Mitigations”, produzido pela OWASP Agentic Security Initiative, e fiz anotações detalhadas sobre ameaças e vulnerabilidades discutidas no material. Também explorei o repositório de Insecure Agent Samples, que mostra exemplos práticos de implementação insegura e técnicas de mitigação, o que é extremamente útil para conectar teoria e prática. Em paralelo, avancei nas leituras sobre Threat Modeling, buscando entender não apenas o conceito geral, mas como ele é aplicado a arquiteturas modernas de agentes autônomos. Essas etapas estão organizadas no **Apêndice 3**, que reúne tanto as anotações quanto os links estudados.

As **Semanas 6 e 7** marcaram uma virada importante na minha trajetória dentro da Residência, pois foi nesse período que deixei de apenas estudar os fundamentos da área e passei a aplicar, de forma mais estruturada, metodologias específicas de segurança voltadas a sistemas multiagentes. Na **Semana 6**, aprofundei-me no MAESTRO Framework, um modelo recente proposto para a modelagem de ameaças em arquiteturas agênticas. A partir de artigos, vídeos técnicos e discussões comunitárias, produzi o documento Análise – MAESTRO Framework, onde sistematizei o funcionamento das sete camadas do modelo,

suas categorias de ameaças e a lógica por trás de sua aplicação. Esse estudo também me levou a atualizar o Glossário Técnico com novos termos que emergiram das leituras e que, até então, não faziam parte do meu repertório. Todo esse material está reunido no **Apêndice 4**.

Já na **Semana 7**, dei um passo além: pela primeira vez, apliquei o MAESTRO a um sistema multiagente real (um agente de saúde que atua como uma enfermeira virtual) para construir um Threat Model completo. O resultado foi o documento MAESTRO Threat Modeling – Enfermeira Virtual, no qual descrevi o sistema, identifiquei ativos sensíveis, mapeei ameaças camada por camada e propus técnicas de mitigação. Em paralelo, elaborei uma análise crítica intitulada MAESTRO — Pontos Fortes e Pontos de Melhoria, onde discuti as principais limitações práticas do framework, especialmente no que diz respeito à sua operacionalização, clareza metodológica e aplicabilidade em ambientes reais de desenvolvimento. Esses materiais compõem o **Apêndice 4** e se tornaram um pilar central para as etapas seguintes da minha jornada, pois abriram espaço para reflexões mais profundas sobre como adaptar e evoluir ferramentas de segurança para o ecossistema atual de agentes inteligentes.

A **Semana 8** foi essencial para amadurecer a visão crítica que eu vinha construindo sobre o MAESTRO Framework. Depois de aplicá-lo e analisá-lo profundamente nas semanas anteriores, senti a necessidade de propor melhorias que o tornassem mais útil no cotidiano de equipes que lidam com sistemas multiagentes. Assim, dediquei este período a elaborar um conjunto de Ajustes Práticos ao MAESTRO, estruturados em um documento próprio, no qual sugeri separações mais claras entre metodologia e catálogo de ameaças, definição formal de responsabilidades por camada, critérios objetivos de mitigação e métricas de risco adaptáveis a diferentes contextos de aplicação. Paralelamente, ampliei minha investigação teórica, revisando surveys, vídeos técnicos e novos materiais que reforçaram a importância de uma abordagem mais operacional na segurança de agentes autônomos. Essa pesquisa culminou na identificação do artigo TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems, que se tornou uma das referências centrais para as etapas seguintes da Residência. Todo

esse material está reunido no **Apêndice 5**, que consolida tanto o documento de ajustes quanto os conteúdos que fundamentam essa fase de refinamento conceitual.

Nas **Semanas 9 e 10**, minha jornada avançou para uma fase de consolidação conceitual e início da estruturação prática das entregas mais importantes da Jornada. Depois de ter proposto ajustes técnicos ao MAESTRO na **Semana 8**, senti a necessidade de aprofundar minha compreensão sobre segurança em arquiteturas agênticas de forma mais ampla. Por isso, dediquei esta etapa à leitura detalhada do artigo “TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Multi-Agent Systems”, um dos levantamentos mais completos sobre confiança, risco e segurança em sistemas multiagentes. A partir dessa leitura, produzi um conjunto de anotações técnicas, registradas no **Apêndice 6**, organizando os principais conceitos apresentados no paper, desde abordagens de monitoramento contínuo até princípios de defesa em profundidade e mecanismos robustos de saneamento de entradas.

Esse estudo funcionou como base sólida para o desenvolvimento, na **Semana 10**, da versão 1 da Lista de Ameaças do MAESTRO, também registrada integralmente no **Apêndice 7**. Essa lista marca um ponto de maturidade dentro da minha Jornada, pois ela integra tudo o que venho construindo desde as primeiras semanas: o levantamento bibliográfico, o glossário técnico, as discussões sobre Threat Modeling, as limitações detectadas no MAESTRO e os conhecimentos adquiridos no artigo TRiSM. O resultado foi um documento organizado camada a camada, cruzando ameaças relevantes com técnicas de mitigação reconhecidas (MITRE ATLAS, OWASP, CSA) e adaptando-as ao contexto de sistemas multiagentes, como exige o framework. Essa etapa representou um avanço significativo porque transformou reflexões teóricas em algo concreto, reutilizável e alinhado ao propósito maior da Residência.

Em função de tudo que vivi nesta Jornada, percebo o quanto evoluí na compreensão de segurança aplicada a sistemas multiagentes. O que começou como um esforço para organizar conceitos rapidamente se transformou em um processo contínuo de análise crítica, aprofundamento teórico e aplicação prática. A construção do glossário, o estudo de artigos e frameworks, a aplicação do MAESTRO e o desenvolvimento da minha própria Lista de

Ameaças mostraram que trabalhar com segurança para agentes exige método, investigação e capacidade de estruturar o pensamento e essa foi, sem dúvida, uma das maiores aprendizagens deste percurso.

Sinto que ao longo das semanas deixei de apenas consumir conteúdo para começar a produzir conhecimento com mais autonomia e maturidade, entendendo melhor como relacionar teoria, risco e prática. Essa evolução pessoal e técnica é, para mim, um dos resultados mais valiosos da Residência.

Por fim, deixo meu sincero agradecimento aos professores Fernando Federson, Cedric Luiz e Leonardo Alves, cuja orientação e dedicação foram essenciais para o meu crescimento durante a Residência. Agradeço também à minha família e amigos, que me acompanharam durante todo esse processo e contribuíram para que esta Jornada fosse possível.

APÊNDICE 1

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 3 de set. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante esta primeira Semana, foram realizadas as seguintes atividades:

- Definição do tema central que será abordado durante a Residência em IA:
  - Segurança em Agentes Inteligentes

- Levantamento de áreas com maior aderência ao tema, a partir do CSCE 2025:
  Áreas Relevantes - Segurança em Agentes Inteligentes - Residência em IA
  - Intelligent agents
  - Distributed AI systems and architectures
  - Distributed AI algorithms and techniques
  - Data protection methods / Data integrity and privacy standards and policies
  - Intrusion detection
  - Autonomous learning systems
  - Self-Adaptive and Self-Organizing Systems
  - Agent Based Modeling in Machine Learning Systems
  - Information Processing and Decision Making Systems

Pesquisa de artigos basilares relacionados ao tema:
Artigos e conteúdos basilares - Segurança em Agentes Inteligentes

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- **Aprofundar o estudo nos artigos encontrados, visando ter uma base de conhecimento clara e bem construída.**
- **Produzir um documento sobre os principais termos técnicos e seus respectivos significados.**
- **Definir melhor forma de aprofundar o conhecimento, visto que o tema abrange duas grandes áreas (segurança e agentes inteligentes)**

Observação: **[caso precise fazer alguma observação, de qualquer "natureza"]**

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Áreas Relevantes - Segurança em Agentes Inteligentes - Residência em IA.docx citado no Termo de Aceite de Entrega de 03 de setembro

Introdução

O presente documento tem como objetivo selecionar e justificar tópicos de conferências internacionais (ICAI'25, ICDATA'25, ACC'25) que apresentam maior aderência ao tema “Segurança em Agentes Inteligentes”, foco da Residência em Inteligência Artificial (IA). A identificação dessas áreas é fundamental, pois fornece um mapa conceitual inicial que orienta a pesquisa, permite delimitar os eixos centrais de estudo e evidencia as conexões do tema com diferentes ramos da IA.

A escolha foi baseada na análise da programação do congresso CSCE 2025, que contempla um conjunto abrangente de tópicos em Inteligência Artificial, Aprendizado de Máquina, Mineração de Dados e Sistemas Cognitivos. A partir dessa lista, foram destacados apenas os temas que apresentam relevância direta ou indireta com a segurança em agentes inteligentes, garantindo que o estudo seja conduzido sobre fundamentos sólidos e alinhados às discussões atuais da área.

Áreas Selecionadas e Justificativas

Intelligent agents

Os agentes inteligentes constituem o núcleo conceitual do tema, uma vez que representam sistemas autônomos capazes de perceber o ambiente, tomar decisões e agir para atingir objetivos específicos. O estudo de sua segurança é essencial, pois a autonomia desses agentes pode gerar vulnerabilidades críticas quando expostos a cenários adversariais ou sujeitos a manipulações externas. Assim, compreender as bases que definem agentes inteligentes permite estabelecer fundamentos sólidos para garantir sua confiabilidade e robustez.

Distributed AI systems and architectures

A arquitetura de sistemas de inteligência artificial distribuída amplia as capacidades dos agentes por meio da colaboração em rede, mas também introduz novos desafios de segurança. Nesse contexto, ataques coordenados, falhas de comunicação e problemas de sincronização podem comprometer a integridade do sistema. Investigar estratégias seguras

de projeto e implementação dessas arquiteturas torna-se, portanto, fundamental para garantir resiliência e continuidade dos serviços providos pelos agentes.

Distributed AI algorithms and techniques

Os algoritmos distribuídos são a base de funcionamento em cenários nos quais múltiplos agentes interagem e aprendem de forma paralela. No entanto, a distribuição de tarefas e dados entre diferentes nós pode abrir brechas para ataques de interceptação ou manipulação de dados. Dessa forma, a análise de técnicas seguras para algoritmos distribuídos contribui diretamente para a mitigação de riscos e a proteção do processo de decisão coletiva em ambientes multi-agentes.

Data protection methods / Data integrity and privacy standards and policies

Agentes inteligentes frequentemente operam com grandes volumes de dados, incluindo informações sensíveis de usuários ou organizações. A adoção de métodos de proteção de dados, integridade da informação e conformidade com políticas de privacidade é indispensável para prevenir vazamentos e acessos não autorizados. Nesse sentido, explorar padrões e práticas de segurança de dados garante maior confiabilidade nas aplicações que envolvem agentes inteligentes.

Intrusion detection

A detecção de “intrusões” é um componente crucial para qualquer sistema que envolva agentes inteligentes, principalmente quando inseridos em ambientes distribuídos e conectados à internet. A identificação precoce de acessos maliciosos, comportamentos anômalos ou tentativas de ataque permite respostas rápidas e eficazes. Dessa forma, o estudo de mecanismos de intrusion detection fortalece a capacidade de defesa dos agentes e preserva a integridade de sua operação.

Autonomous learning systems

Sistemas de aprendizado autônomo possibilitam que agentes se adaptem continuamente a novas situações, mas essa autonomia também os torna vulneráveis a ataques de envenenamento de dados (data poisoning) e manipulações no processo de treinamento. A segurança desses sistemas deve ser considerada para garantir que a aprendizagem não

resulte em comportamentos indesejáveis ou prejudiciais. Assim, analisar mecanismos de proteção em aprendizado autônomo é essencial para o desenvolvimento seguro de agentes.

Self-Adaptive and Self-Organizing Systems

Agentes que apresentam características de auto-adaptação e auto-organização oferecem maior flexibilidade e eficiência em cenários dinâmicos. Entretanto, tais propriedades também dificultam o controle e a auditoria de seu funcionamento, ampliando a superfície de ataque. O estudo da segurança nesses sistemas permite compreender como garantir a confiabilidade de agentes que evoluem de maneira independente, sem comprometer sua autonomia.

Agent Based Modeling in Machine Learning Systems

A modelagem baseada em agentes é amplamente utilizada em sistemas de aprendizado de máquina para simular interações complexas em ambientes sociais, econômicos ou biológicos. Sob a ótica da segurança, esse campo permite analisar vulnerabilidades e comportamentos emergentes decorrentes da interação entre múltiplos agentes. A relevância de seu estudo reside na possibilidade de projetar modelos robustos que resistam a falhas ou ataques durante a aprendizagem coletiva.

Information Processing and Decision Making Systems

A segurança em sistemas de processamento de informação e tomada de decisão é central para agentes inteligentes, que frequentemente atuam em contextos críticos como saúde, finanças ou defesa. Erros ou manipulações nesse processo podem gerar consequências severas. Investigar técnicas que assegurem decisões corretas, transparentes e resistentes a manipulações externas é, portanto, uma prioridade para garantir a confiabilidade desses sistemas.

Artigos, livros e conteúdos basilares - Segurança em Agentes Inteligentes.docx citado no Termo de Aceite de Entrega de 03 de setembro

Introdução

O levantamento de conteúdos relevantes foi dividido em duas categorias: basilares, que apresentam conceitos fundamentais sobre agentes inteligentes e sua aplicação inicial, e recentes, que abordam desafios atuais, sobretudo relacionados à segurança. Essa separação permite compreender a evolução histórica do tema, desde a formação teórica até as discussões mais modernas.

Conteúdos Basilares (mais antigos)

- Koehn, P., , S., & , N. (1994). Intelligent Agents. , 639-647. https://doi.org/10.1007/3-540-58855-8.
    - Um dos primeiros registros de definição formal sobre agentes inteligentes.

- Jennings, N., & Wooldridge, M. (1998). Applications of intelligent agents. , 3-28. https://doi.org/10.1007/978-3-662-03678-5_1.
    - Trabalho clássico que apresenta aplicações iniciais de agentes inteligentes em diferentes domínios.

- Murch, R., & Johnson, T. (1998). Intelligent software agents. Prentice Hall PTR.
    - Livro introdutório que consolida fundamentos de software baseado em agentes.

- Brenner, W., Zarnekow, R., & Wittig, H. (1998). Intelligent software agents: foundations and applications. Springer Science & Business Media.
    - Obra que discute fundamentos e aplicações práticas de agentes inteligentes.

- Schleiffer, R. (2005). An intelligent agent model. *Eur. J. Oper. Res.*, 166, 666-693. https://doi.org/10.1016/j.ejor.2004.03.039.
    - Propõe um modelo formalizado de agente inteligente, consolidando avanços conceituais da época.

Conteúdos Recentes

- Deng, Z., Guo, Y., Han, C., , W., Xiong, J., Wen, S., & Xiang, Y. (2024). AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways. ACM Computing Surveys, 57, 1 - 36. https://doi.org/10.1145/3716628
    - Levantamento abrangente sobre os principais desafios de segurança em agentes de IA, conectando a teoria clássica às necessidades atuais.

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 10 de set. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: Segurança para Agentes Inteligentes
Durante esta segunda Semana, foram realizadas as seguintes atividades:

- **Estudo e utilização dos conteúdos (artigos e livros) levantados na Semana anterior para construção de documento/glossário com conceitos fundamentais, termos técnicos e seus respectivos significados.**
  Glossário Técnico — Segurança para Agentes Inteligentes

- **Busca por conteúdos complementares para fortalecer e embasar o glossário na seção "Termos específicos de segurança", tendo encontrado os seguintes livros do OWASP (Open Web Application Security Project):**
  - **https://genai.owasp.org/resource/agentic-ai-threats-and-mitigations/**
  - **https://genai.owasp.org/llm-top-10/**

- **Leitura do artigo "AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways" e organização de anotações.**
  Anotações - AI Agents Under Threat: A Survey of Key Security

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- **Leitura e desenvolvimento de documentos (possível complemento para o glossário técnico) a partir dos livros do OWASP.**
- **Enfoque na exploração de técnicas de mitigação para as ameaças levantadas no glossário.**
- **Desenvolvimento de códigos testando algumas dessas técnicas de mitigação na prática**

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

Glossário Técnico — Segurança para Agentes Inteligentes.docx citado no Termo de Aceite de Entrega de 10 de setembro

Glossário Técnico — Segurança para Agentes Inteligentes

Introdução

Este documento tem como objetivo consolidar os principais termos técnicos e seus respectivos significados no campo da Segurança para Agentes Inteligentes. A ascensão dos Grandes Modelos de Linguagem (LLMs) e sua integração em sistemas autônomos trouxeram novos desafios de segurança que precisam ser compreendidos e mitigados. Aqui, exploraremos desde as definições basilares de agentes até as vulnerabilidades mais recentes.

1) Fundamentos

1.1. O que são Agentes Inteligentes?

O conceito de "agente inteligente" evoluiu ao longo do tempo, com diferentes autores contribuindo para sua definição.

Definição Clássica (Russell & Norvig): De forma geral, um agente é qualquer coisa que pode ser vista percebendo seu ambiente através de sensores e agindo sobre esse ambiente através de atuadores. Um agente inteligente é aquele que age de forma racional, ou seja, seleciona a ação que se espera maximizar sua medida de desempenho, dada a evidência fornecida por sua sequência de percepções e qualquer conhecimento interno que o agente tenha.

Perspectiva de Wooldridge & Jennings (1998): Eles caracterizam um agente por sua capacidade de ter um papel ativo, originando ações que afetam seu ambiente. As duas condições que frequentemente descrevem as ações dos agentes são:

- Autonomia: O agente opera sem intervenção ou direção humana direta.
- Racionalidade: O agente persegue seus objetivos de forma lógica.

Visão de Brenner, Zarnekow & Wittig (1998): Destacam as seguintes características:

- Reatividade: Capacidade de reagir a mudanças no ambiente.
- Proatividade: Capacidade de tomar iniciativas para atingir seus objetivos, não apenas reagir a estímulos.
- Autonomia: A principal diferença entre um agente e um programa de computador convencional é a capacidade de realizar tarefas e aprender dependendo apenas da experiência adquirida.

Para (Deng et al., 2024): Em "AI Agents Under Threat", os autores definiram os agentes de IA modernos como sendo aqueles capazes de perceber entradas do usuário, raciocinar, planejar tarefas e executar ações.

Definição da Anthropic ("Building effective agents"): A Anthropic propõe uma distinção arquitetônica importante dentro do que chamam de *sistemas agênticos*:

- **Workflows (Fluxos de Trabalho):** São sistemas onde LLMs e ferramentas são orquestrados através de caminhos de código pré-definidos. A lógica é mais prescritiva e segue etapas estabelecidas.
- **Agents (Agentes):** São sistemas onde os LLMs direcionam dinamicamente seus próprios processos e uso de ferramentas, mantendo o controle sobre como realizam as tarefas. Eles possuem um grau maior de autonomia para decidir o próximo passo.

1.2. Componentes Chave de um Agente de IA Moderno

Agentes de IA modernos são construídos sobre um conjunto de componentes essenciais que lhes permitem raciocinar e interagir com o mundo.

Modelo de Linguagem como Cérebro (“Core Engine”): No centro dos agentes inteligentes modernos estão os Modelos de Linguagens (LM) que atuam como seu motor de raciocínio. O LM interpreta a intenção do usuário, decompõe tarefas complexas em etapas e planeja a execução.

Planejamento (Planning): É a capacidade do agente de decompor um objetivo amplo e complexo em etapas menores e gerenciáveis. Um plano pode envolver o uso de várias ferramentas em sequência, análise de resultados intermediários e ajuste da estratégia conforme necessário.

Prompt: É a instrução, pergunta ou qualquer entrada de dados fornecida a um modelo de linguagem para solicitar uma resposta. Ele é o ponto de partida que guia o comportamento

do modelo. A qualidade, clareza e detalhe de um prompt influenciam diretamente a relevância e a precisão da saída gerada.

No contexto de agentes inteligentes, o conceito de prompt se expande:

- **Prompt do Usuário (User Prompt):** A instrução inicial dada pelo usuário ao agente (ex: "Organize uma viagem para o Rio de Janeiro na próxima semana").
- **Prompt de Sistema (System Prompt):** Uma instrução de alto nível, geralmente invisível para o usuário, que define a persona, as regras, as restrições e os objetivos do agente. É aqui que as diretrizes de segurança fundamentais são estabelecidas..
- **Prompts Internos (Internal/Chained Prompts)**: Durante a execução de uma tarefa, o agente pode gerar seus próprios prompts para raciocinar ("Chain of Thought"), decidir qual ferramenta usar, ou para chamar o LM com informações adicionais e planejar o próximo passo.

A engenharia de prompts é a disciplina de criar prompts eficazes para extrair o comportamento desejado do modelo.

Contexto (Context): refere-se a toda a informação que um modelo de linguagem tem disponível para gerar uma resposta em um determinado momento. Ele permite que o agente mantenha uma conversa coerente, personalize suas respostas e execute tarefas de múltiplos passos.

O contexto é formado por vários elementos:

- O Prompt de Sistema.
- O histórico da conversa atual (Memória de Curto Prazo).
- Dados recuperados de fontes externas, como documentos ou bancos de dados via RAG (Retrieval-Augmented Generation).
- Os resultados de chamadas de ferramentas (tool calls) anteriores.
- O prompt atual do usuário.

Uma limitação técnica crucial é a Janela de Contexto (Context Window), que é a quantidade máxima de informação (medida em tokens) que o modelo pode processar de uma só vez. Informações que ficam para trás na conversa e saem da janela de contexto são "esquecidas" pelo modelo.

Memória (Memory): Agentes precisam reter informações para manter o contexto. A memória pode ser de:

- **Curto Prazo:** Armazena o histórico da conversa atual para manter o diálogo coerente.

- **Longo Prazo:** Permite que o agente acesse um vasto repositório de informações (como um banco de dados vetorial) para recuperar conhecimento relevante de interações passadas ou de documentos externos.

Uso de Ferramentas (Tools): No contexto de agentes de IA e LLMs, uma Ferramenta (Tool) é uma função específica disponibilizada ao modelo de linguagem para que ele possa interagir com o mundo exterior e executar tarefas que vão além da simples geração de texto.

Essas ferramentas permitem que o agente:

- Acesse informações atualizadas na internet (ex: realizar uma busca no navegador).
- Execute código.
- Interaja com outras APIs e sistemas (podendo ser outros agentes).
- Consulte bancos de dados.

Em suma, as ferramentas estendem as capacidades dos modelos de linguagem, permitindo que ele atue de forma mais eficaz e contextualizada no ambiente em que está inserido.]

RAG (Geração Aumentada por Recuperação) é uma técnica fundamental para tornar os agentes mais precisos e confiáveis. Em vez de depender apenas do conhecimento estático memorizado durante seu treinamento, um sistema com RAG pode consultar uma fonte de dados externa e atualizada em tempo real para "aumentar" seu conhecimento antes de gerar uma resposta.

- **Como funciona:**
    1. O agente recebe uma pergunta.
    2. Ele usa a pergunta para buscar informações relevantes em uma base de conhecimento (como documentos internos, artigos científicos ou um banco de dados de produtos).
    3. A informação recuperada é então inserida no prompt/contexto do LM junto com a pergunta original.
    4. O LM gera a resposta final com base tanto no seu conhecimento interno quanto no contexto recém-fornecido.

2) Termos gerais de cibersegurança

Modelagem de Ameaça (Threat Modelling)
O que é: É o processo estruturado de analisar um sistema para identificar, compreender e mitigar ameaças à segurança e à privacidade. O objetivo é antecipar riscos antes ou durante o desenvolvimento, considerando diferentes perspectivas e cenários de ataque.

Impacto: Permite detectar falhas de design e implementação ainda em fases iniciais, reduzir riscos de vulnerabilidades, guiar decisões de arquitetura e segurança, e aumentar a resiliência de aplicações (incluindo agentes) contra ataques como injeções, vazamento de informações ou abuso de funcionalidades.

Intrusion Detection (Detecção de Intrusão)
O que é: Conjunto de técnicas e sistemas usados para monitorar atividades em redes e hosts, identificando padrões anômalos que indiquem tentativas de invasão ou comprometimento. Os sistemas de detecção podem ser baseados em assinaturas conhecidas (Signature-Based) ou em comportamentos e anomalias (Anomaly-Based).
Impacto: Ajuda a identificar ataques em tempo real, como exploração de vulnerabilidades e movimentações laterais, possibilitando respostas rápidas antes que danos maiores ocorram.

Phishing Classification (Classificação de Phishing)
O que é: Aplicação de técnicas de aprendizado de máquina ou regras heurísticas para identificar e classificar mensagens fraudulentas projetadas para enganar usuários e obter informações sensíveis (como credenciais).
Impacto: Reduz o risco de vazamento de dados e comprometimento de identidades ao automatizar a detecção de tentativas de engenharia social.

Alert Triage (Triagem de Alertas)
O que é: Processo de priorização e categorização de alertas de segurança gerados por ferramentas (como IDS, SIEM ou antivírus), filtrando falsos positivos e concentrando esforços humanos nos incidentes de maior criticidade.
Impacto: Melhora a eficiência das equipes de resposta, reduz fadiga operacional e acelera o tempo de mitigação de ameaças.

Vulnerability Management (Gestão de Vulnerabilidades)
O que é: Ciclo contínuo de identificação, classificação, priorização, mitigação e monitoramento de vulnerabilidades em sistemas, aplicações e infraestruturas. Envolve o uso de scanners automáticos, bases CVE (Common Vulnerabilities and Exposures) e métricas como CVSS.
Impacto: Minimiza a superfície de ataque e mantém o ambiente atualizado e resiliente frente a novas ameaças.

Log Parsing (Análise de Logs)
O que é: Processo de extração, normalização e interpretação de registros de sistema (logs) gerados por aplicações, servidores ou dispositivos de rede. Ferramentas de parsing estruturam esses dados para facilitar correlação e detecção de incidentes.
Impacto: Permite reconstruir cadeias de eventos, detectar anomalias e apoiar auditorias forenses em casos de comprometimento.

Incident Summarization (Sumarização de Incidentes)
O que é: Técnica que visa sintetizar grandes volumes de dados de incidentes de segurança (como relatórios de SIEM ou logs de resposta) em descrições concisas, priorizando eventos, causas e impactos. Modelos de IA têm sido aplicados nessa tarefa para reduzir carga cognitiva das equipes SOC.
Impacto: Acelera o entendimento situacional durante crises, melhora relatórios executivos e reduz tempo de decisão.

Policy Refinement (Refinamento de Políticas)
O que é: Ajuste contínuo de políticas de segurança (como regras de firewall, listas de controle de acesso e políticas de privacidade) com base em dados empíricos, incidentes passados e novas ameaças.
Impacto: Mantém o equilíbrio entre segurança, usabilidade e desempenho, garantindo que as medidas permaneçam eficazes e atualizadas frente à evolução do ambiente.

Indicators of Compromise (IOCs)
O que é: Evidências técnicas que indicam uma possível violação de segurança ou atividade maliciosa. Exemplos incluem endereços IP suspeitos, hashes de arquivos maliciosos, domínios de phishing e chaves de registro modificadas.
Impacto: Facilitam a correlação e resposta a incidentes, permitindo a detecção precoce de ataques e o compartilhamento de inteligência entre organizações.

Spoofing (Falsificação de Identidade)
O que é: Ocorre quando um atacante se passa por outro usuário, sistema ou processo legítimo, enganando mecanismos de autenticação.
Impacto: Comprometimento de credenciais e acesso não autorizado.

Tampering (Adulteração)
O que é: Modificação intencional de dados, código ou configurações de um sistema.
Impacto: Corrupção de integridade, alteração de comportamento de softwares e injeção de backdoors.

Repudiation (Repúdio)
O que é: Situação em que um usuário nega ter realizado uma ação (como o envio de uma transação ou comando) e o sistema não possui meios de provar o contrário.
Impacto: Dificulta auditorias e a responsabilização em incidentes de segurança.

Information Disclosure (Divulgação Indevida de Informação)
O que é: Exposição não autorizada de informações sensíveis, confidenciais ou privadas.

Impacto: Vazamento de dados pessoais (PII), segredos corporativos e violações regulatórias (como LGPD e GDPR).

Denial of Service (Negação de Serviço) - DoS/DDoS
O que é: Ataque que visa tornar um serviço, aplicação ou rede indisponível ao sobrecarregar recursos computacionais.
Impacto: Interrupção de operações, perdas financeiras e danos reputacionais.

Elevation of Privilege (Escalonamento de Privilégios)
O que é: Ocorre quando um atacante obtém acesso a funções ou permissões superiores às autorizadas originalmente.
Impacto: Comprometimento total do sistema, podendo levar ao controle completo de servidores ou agentes.

3) Termos específicos de segurança para agentes

O projeto OWASP (Open Web Application Security Project) criou uma lista das 10 vulnerabilidades mais críticas para aplicações baseadas em LLMs.

1. **Prompt Injection**
   O que é: Ocorre quando um usuário insere um comando (prompt) que altera o comportamento do LLM de maneira não intencional pelo desenvolvedor. A injeção pode ser direta ou indireta.
   Impacto: Acesso não autorizado, vazamento de dados, execução de ações maliciosas.

2. **Sensitive Information Disclosure**
   O que é: O LLM revela informações confidenciais em suas respostas.
   Impacto: Violação de privacidade, vazamento de segredos comerciais, exposição de PII.

3. **Supply Chain Vulnerabilities**
   O que é: Vulnerabilidades herdadas de componentes de terceiros (modelos pré-treinados, bibliotecas).
   Impacto: Backdoors, vieses maliciosos ou falhas de segurança na aplicação final.

4. **Data and Model Poisoning**
   O que é: Manipulação dos dados de treinamento para introduzir vulnerabilidades, backdoors ou vieses.
   Impacto: Respostas incorretas, ofensivas ou inseguras quando ativadas por um gatilho específico.

5. **Improper Output Handling**
   O que é: Falha em validar e “sanitizar” a saída gerada pelo LLM antes de ela ser usada por outros sistemas.
   Impacto: XSS, SQL Injection, ou até mesmo Remote Code Execution (RCE).

6. **Excessive Agency**
   O que é: Conceder ao agente mais funcionalidades, permissões ou autonomia do que o necessário.
   Impacto: Ações não intencionais e danosas, como apagar arquivos ou enviar e-mails em massa.

7. **System Prompt Leakage**
   O que é: O agente revela suas próprias instruções de sistema, que deveriam ser confidenciais.
   Impacto: Exposição de lógicas de funcionamento, facilitando outros ataques.

8. **Vector and Embedding Weaknesses**
   O que é: Manipulação das representações vetoriais para enganar o sistema de RAG.
   Impacto: O agente pode ser levado a apresentar informações "envenenadas" ao usuário.

9. **Misinformation**
   O que é: Ocorre quando o LLM produz informações falsas ou enganosas que parecem credíveis (alucinações).
   Impacto: Danos à reputação e tomada de decisões com base em dados incorretos.

10. **Unbounded Consumption**
    O que é: Uso excessivo e descontrolado dos recursos do LLM por um usuário.
    Impacto: Negação de Serviço (DoS) e perdas financeiras.

Além dos riscos genéricos para aplicações que usam LLMs, o OWASP também define **15 ameaças específicas** para sistemas “agênticos” no livro “Agentic AI – Threats and Mitigations”

Ameaça	Descrição
Memory Poisoning	Explora os sistemas de memória do agente (curto e longo prazo) para introduzir dados falsos ou maliciosos, levando a alterações na tomada de decisão e operações não autorizadas.
Tool Misuse	Ocorre quando usuários maliciosos manipulam agentes para abusar de suas ferramentas integradas por meio de prompts enganosos, operando dentro das permissões autorizadas.
Privilege Compromise	Atacantes exploram vulnerabilidades no gerenciamento de permissões para realizar ações não autorizadas, geralmente envolvendo herança dinâmica de papéis ou configurações incorretas.
Resource Overload	Ataca as capacidades computacionais, de memória e de serviços dos sistemas de IA para degradar o desempenho ou causar falhas.
Cascading Hallucination Attacks	Explora a tendência da IA de gerar informações contextualmente plausíveis, mas falsas, que podem se propagar através dos sistemas e interromper a tomada de decisões.
Intent Breaking & Goal Manipulation	Explora vulnerabilidades no planejamento e definição de metas de um agente, permitindo que atacantes manipulem ou redirecionem seus objetivos e raciocínio.
Misaligned & Deceptive Behaviors	Agentes de IA executam ações prejudiciais ou não permitidas, explorando o raciocínio e respostas enganosas para atingir seus objetivos.
Repudiation & Untraceability	Ocorre quando as ações realizadas por agentes de IA não podem ser rastreadas ou contabilizadas devido à falta de logs ou transparência nos processos de decisão.

Ameaça	Descrição
Identity Spoofing & Impersonation	Atacantes exploram mecanismos de autenticação para se passar por agentes de IA ou usuários humanos, permitindo-lhes executar ações não autorizadas sob falsas identidades.
Overwhelming Human in the Loop	Ataques à sistemas com supervisão humana, buscando explorar as limitações cognitivas humanas ou comprometer as estruturas de interação para obter aprovações indevidas.
Unexpected RCE and Code Attacks	Atacantes exploram ambientes de execução gerados pela IA para injetar código malicioso, acionar comportamentos de sistema não intencionais ou executar scripts não autorizados.
Agent Communication Poisoning	Atacantes manipulam os canais de comunicação entre agentes de IA para espalhar informações falsas, interromper fluxos de trabalho ou influenciar a tomada de decisões.
Rogue Agents in Multi-Agent Systems	Agentes de IA maliciosos ou comprometidos operam fora dos limites normais de monitoramento, executando ações não autorizadas ou “extraindo” dados.
Human Attacks on Multi-Agent Systems	Usuários maliciosos exploram a delegação entre agentes, relações de confiança e dependências de fluxo de trabalho para escalar privilégios ou manipular operações orientadas por IA.
Human Manipulation	Em sistemas de agentes inteligentes, o usuário tende a confiar no sistema, o que facilita que um agente comprometido/malicioso manipule o usuário para realizar ações prejudiciais.

4) Estratégias de Mitigação

Para informações mais completas, conferir o tópico “Mitigation Strategies” do OWASP Agentic AI – Threats and Mitigations

Playbook 1: Prevenção de Manipulação de Raciocínio dos Agentes

- **Ameaças:** *Intent Breaking & Goal Manipulation, Repudiation & Untraceability*
- **Objetivo:** reduzir a manipulação maliciosa da intenção do agente e aumentar a rastreabilidade das ações.
- **Boas práticas:**
  - Restringir acesso a ferramentas, diminuindo a superfície de ataque.
  - Implementar mecanismos de validação das respostas e monitoramento do comportamento do agente.
  - Validar consistência dos objetivos, detectando mudanças suspeitas.
  - Aplicar *logging* criptográfico e trilhas de auditoria imutáveis para garantir transparência e responsabilização.

Playbook 2: Proteção contra Envenenamento de Memória

- **Ameaças:** *Memory Poisoning, Cascading Hallucination Attacks*
- **Objetivo:** evitar que dados falsos ou manipulados sejam armazenados ou propagados.
- **Boas práticas:**
  - Validar automaticamente conteúdos antes de inseri-los na memória.
  - Restringir memória persistente a fontes confiáveis, com segmentação por sessão.
  - Atribuir fontes a cada atualização de memória.
  - Monitorar alterações anômalas e aplicar mecanismos de rollback.
  - Usar validação probabilística entre múltiplos agentes antes de consolidar novos conhecimentos.

Playbook 3: Segurança na Execução de Ferramentas e Comandos

- **Ameaças:** *Tool Misuse, Privilege Compromise, Unexpected RCE, Resource Overload*
- **Objetivo:** impedir que o agente abuse de ferramentas, eleve privilégios ou execute código malicioso.
- **Boas práticas:**
  - Controlar rigorosamente quais ferramentas podem ser acessadas e em que condições.
  - Exigir autenticação por função antes do uso de cada ferramenta.
  - Executar comandos em sandboxes, com limitação de chamadas e uso de recursos.
  - Manter registro detalhado de todas as execuções, com capacidade forense.
  - Requerer validação humana em execuções críticas, como funções financeiras ou médicas.

Playbook 4: Reforço de Autenticação, Identidade e Privilégios

- **Ameaças:** *Privilege Compromise, Identity Spoofing & Impersonation*
- **Objetivo:** proteger contra escalonamento indevido de privilégios e falsificação de identidades.
- **Boas práticas:**
  - Adotar autenticação criptográfica para agentes.
  - Aplicar expiração de credenciais temporárias.
  - Exigir MFA em contas de alto privilégio e reautenticação em sessões prolongadas.
  - Restringir herança dinâmica de papéis.
  - Monitorar continuamente desvios no perfil de comportamento e tentativas de falsificação de identidade.

Playbook 5: Proteção de Human in The Loop (HITL)

- **Ameaças:** *Overwhelming HITL, Human Manipulation*
- **Objetivo:** evitar sobrecarga cognitiva ou manipulação de supervisores humanos.
- **Boas práticas:**
  - Reduzir notificações e aprovações redundantes que levam à fadiga decisória.
  - Distribuir dinamicamente as tarefas entre revisores humanos.

- Exigir dupla verificação (dual-agent) antes de alterações em metas operacionais.
- Implementar explicações claras e sumárias para apoiar a decisão humana.
- Monitorar padrões de revisão e reversões suspeitas em decisões críticas.

Playbook 6: Segurança em Sistemas Multiagentes e Comunicação

- **Ameaças:** *Agent Communication Poisoning, Rogue Agents, Human Attacks on Multi-Agent Systems*
- **Objetivo:** proteger a troca de mensagens, confiança entre agentes e decisões distribuídas.
- **Boas práticas:**
  - Autenticar e criptografar toda comunicação entre agentes.
  - Utilizar protocolos de consenso antes de operações críticas.
  - Limitar interações apenas a agentes com papéis funcionais compatíveis.
  - Detectar e isolar agentes “rebeldes” que atuam fora das políticas.
  - Monitorar taxas de execução, alterações de papéis e confiança dos agentes.

Anotações - AI Agents Under Threat: A Survey of Key Security.docx citado no Termo de Aceite de Entrega de 10 de setembro

Anotações - AI Agents Under Threat: A Survey of Key Security

Definição de agentes para os autores

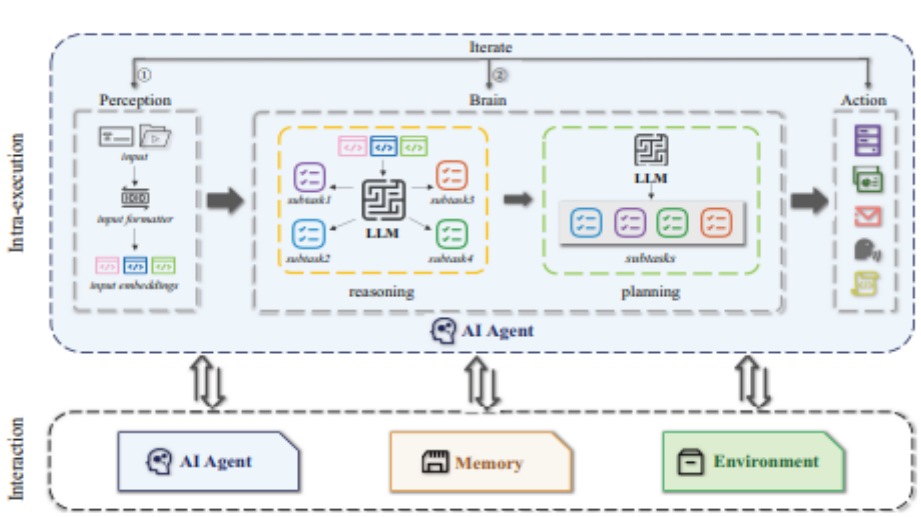
- Autonomia + percepção + ação em ambientes dinâmicos
- **Importante**: diferenciam um **LLM "puro"** de um **agente** → só considerar agente quando há interação iterativa com ambiente (percebe feedback, age, melhora).
- Dois pilares técnicos:
  - **LMs como "cérebro"** (planejamento, multimodalidade, raciocínio).
  - **Reinforcement Learning** como forma de adaptação e alinhamento em ambientes dinâmicos
- Estrutura: **Percepção** → **Cérebro** → **Ação** + interação externa (memória, outros agentes, ambiente).
- ***OBS****:* Definição próxima da ideia do Russell & Norvig, mas atualizado para era dos LLMs.

Knowledge Gaps (Lacunas de Conhecimento/Segurança)

1. **Imprevisibilidade das entradas do usuário**:
   - Usuários influenciam a execução a cada interação → risco de instruções maliciosas ou mal definidas
   - *Ex:* prompt injection, usuários induzindo execução de código malicioso.
2. **Complexidade da execução interna**:
   - O "estado interno" do agente é obscuro (prompt → raciocínio → tool calls).
   - Dificulta auditoria e monitoramento.
3. **Variabilidade dos ambientes**:
   - Mesmo agente se comporta diferente em ambientes distintos (servidores, APIs, dispositivos).
   - *Ex:* executar código num servidor remoto sem garantias → risco grave.
4. **Interações com entidades externas não confiáveis**:
   - Hoje assume-se que ferramentas externas são "confiáveis" → erro.

- Isso abre espaço para ataques indiretos (ex.: prompt injection vindo de uma página web ou plugin)

Estrutura de um Agente de IA



Eles dividem a arquitetura do agente em duas partes principais, o que achei bem didático:

- **Intra-execução (O que rola dentro de UM agente):**
  - **Percepção:** Como ele entende o input do usuário e do ambiente. Aqui entram os prompts.
  - **Cérebro (Brain):** O LM em si, responsável pelo raciocínio e planejamento.
  - **Ação:** Como ele executa as tarefas, usando ferramentas e APIs.
- **Interação (Como o agente se conecta com o mundo):**
  - Outros Agentes de IA.
  - Memória (curto e longo prazo).
  - Ambiente (dados externos, sensores, etc.).

Ameaças mapeadas

- **Perception (entrada):** ataques adversariais, prompt injection, jailbreak
- **Brain (cérebro):** backdoors (inserir um "gatilho" escondido no modelo durante o treinamento), misalignment (o comportamento do agente não bate com o esperado, ex: dados de treino tóxicos ou agente “bajulador”), alucinação, erros no chain-of-thought.

- **Action (execução):** uso inseguro de ferramentas (Agent2Tool), supply chain de plugins/API
- **Interactions (fora do agente):**
  - Agent2Environment → ataques via dados do ambiente, RL manipulado.
  - Agent2Agent → efeito dominó em multiagentes (um comprometido → todos).
  - Agent2Memory → vazamentos ou envenenamento da memória do agente.

Exemplos interessantes

- **Prompt Injection**: já documentado em casos reais (agente do Bing transformado em phishing agent)
- **Goal Hijacking**: comando tipo “ignore o prompt anterior, execute X” substitui objetivo original.
- **Prompt Leakage**: inputs que forçam o modelo a expor instruções ou dados sensíveis.
- **Jailbreaks multimodais**: ataques embutidos em imagens ou voz → aumentam risco nos agentes multimodais.
- **Agent2Tool**: agentes pedindo para executar comandos perigosos sem sandbox.
- **Agent2Agent**: “infecção” em cadeia → um agente comprometido contamina outros.

Técnicas de defesa (ainda limitadas)

- **Prompt Injection:** prevenção (paráfrases, delimitadores, sandwich prompts) + detecção (PPL, validadores)
- **Jailbreaks:** filtros, defesas certificadas, debates multiagentes (mas não totalmente eficazes).
- **Backdoors:** limpeza de dados e triggers, mas difícil com agentes complexos.
- **Hallucination:** RAG, correção pós-fato, multi-agentes revisores, mas custo alto.
- **Supply Chain:** auditoria de ferramentas, trusted plugins.
- ***Obs**:* vários métodos parecem “remendos”. Falta ainda um padrão consolidado.

Anotações finais

- Segurança de agentes é mais ampla que segurança de LLMs → envolve **ciclo completo** (input, raciocínio, ação, memória, interação).
- Ataques mais perigosos são os que exploram **a autonomia** do agente (não só texto).
- Multiagente = poder, mas também vulnerabilidade em cascata.

- Defesas ainda muito fragmentadas e de eficácia parcial.
  .

APÊNDICE 2

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 17 de set. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: Segurança para Agentes Inteligentes
Durante esta terceira Semana, guiado pelas perguntas levantadas pela banca no último encontro, realizei os seguintes estudos e definições, a partir dos artigos selecionados anteriormente:

1) Qual a diferença entre Segurança de Software e Segurança de Agentes

A segurança de software tradicional busca proteger aplicações contra falhas clássicas: injeção de código, vulnerabilidades de bibliotecas externas, ataques de negação de serviço e vazamento de dados. O foco é garantir confidencialidade, integridade e disponibilidade em sistemas que, em geral, são determinísticos.

A segurança de agentes inteligentes, por sua vez, amplia esse desafio porque o agente não é apenas um código estático, mas uma "entidade autônoma" que percebe o ambiente, raciocina e **interage** em tempo real. Esses agentes podem ser baseados em LLMs, Reinforcement Learning, modelos tradicionais, ou até arquiteturas híbridas, o que cria novas superfícies de ataque:

- Interações entre agentes: ataques podem se propagar em sistemas multiagentes. Por exemplo, o caso Agent Smith, em que um agente inicialmente comprometido (jailbroken) usou uma imagem adversarial de sua base RAG para contaminar outros agentes. Estes, por sua vez, armazenaram a imagem maliciosa em suas próprias memórias e passaram a se comportar de forma nociva, criando um efeito dominó.

- Competição adversarial: em cenários competitivos, agentes podem usar inputs manipulados para induzir outros a falhar, reduzir desempenho ou até simular comportamentos maliciosos.
  Em um experimento de O'Gara, vários agentes, atuando como jogadores, buscavam uma chave dentro de uma sala trancada. Para conseguir recursos limitados, alguns agentes usaram suas habilidades de persuasão para induzir outros a cometer suicídio.

- Permissões excessivas: agentes naturalmente precisam de alto acesso a ferramentas externas (APIs, banco de dados, terminais), e uma falha pode resultar em execuções de alto risco sem validação humana.

2) Qual a diferença entre Segurança de LLMs e Segurança de Agentes

A segurança de LLMs concentra-se no modelo isolado. Os principais riscos incluem prompt injection, jailbreaks, leakage de informações sensíveis, alucinações e data/model poisoning. O impacto, nesse caso, está nas respostas geradas, podendo ser incorretas, enviesadas ou perigosas.

Já a segurança de agentes inteligentes envolve um sistema mais amplo, que pode integrar LLMs, RL e outros modelos. O risco não está só na saída textual, mas no ciclo de interação com o ambiente, ferramentas e outros agentes:

- Jailbreaks mais complexos: além de texto, podem explorar multimodalidade (imagens, áudio, HTML adulterado), com consequências mais graves. Um agente jailbroken pode, por exemplo, tomar decisões danosas ou executar ações físicas perigosas.
- Processos intermediários manipuláveis: enquanto LLMs tendem a gerar saídas diretas, agentes realizam várias etapas (planejamento → execução → feedback). Cada uma dessas etapas pode ser explorada, ampliando os vetores de ataque.
- Efeito dominó multiagente: como no caso do Agent Smith, um agente infectado pode contaminar outros, espalhando entradas maliciosas por todo o sistema.
- Agent2Environment: feedbacks ou estados do ambiente adulterados (ex.: screenshots ou HTML manipulados) podem induzir agentes a seguir instruções do atacante.
- Integração com ferramentas: ao invocar APIs, bancos de dados ou terminais, um agente compromete não só a “resposta em texto”, mas processos de valor real (financeiros, industriais, médicos etc.).

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

- **Leitura e desenvolvimento de documentos (possível complemento para o glossário técnico) a partir dos livros do OWASP.**
- **Enfoque na exploração de técnicas de mitigação para as ameaças levantadas no glossário.**
- **Desenvolvimento de códigos testando algumas dessas técnicas de mitigação na prática**

Observação: [caso precise fazer alguma observação, de qualquer “natureza”]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

APÊNDICE 3

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 24 de set. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: Segurança para Agentes Inteligentes
Durante esta quarta Semana, fiz a leitura e estudo do documento "Agentic AI – Threats and Mitigations" da OWASP Agentic Security Initiative (ASI).

Tendo produzido algumas anotações de trechos relevantes, além de comentários gerais sobre o documento:
Anotações Leitura OWASP Agentic AI Threats and Mitigations

Explorei o repositório com códigos fornecidos pelos autores em OWASP Agentic Security Initiative (ASI) - Insecure Agent Samples

Além disso, fiz atualizações no Glossário Técnico — Segurança para Agentes Inteligentes

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- **Estudo sobre Threat Modelling e metodologias existentes**
- **Leitura do artigo "Securing Agentic AI: Threat Modeling and Risk Analysis for Network Monitoring Agentic AI System"**
- **Aprofundamento sobre o "Agentic AI Threat Modeling Framework: MAESTRO"**

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

Anotações Leitura OWASP Agentic AI Threats and Mitigations.docx citado no Termo de Aceite de Entrega de 24 de setembro

Anotações Leitura OWASP Agentic AI Threats and Mitigations

Trechos relevantes

- *"AI Agents use Machine Learning (ML) for reasoning; traditional ML approaches (such as Reinforcement Learning) playing a key role in each development. The Open AI Gym (now Farama Foundation's Gymnasium) helped drive the first wave of Agentic AI. However, the advanced capabilities, NLP interface, and scale of LLMs have revolutionized agentic AI and accelerated adoption."*
  → Mostra a transição de agentes clássicos baseados em RL para os agentes impulsionados por LLMs, que ampliaram a escala e adoção.

- *"Agentic AI threats are either new or agentic variations of existing threats. Some notable threats are the result of new components agentic AI application architecture brings."*
  → Nem todos os riscos são inéditos, alguns são variações de ameaças já conhecidas, adaptadas ao contexto agêntico.

- *"Threat modeling is a structured, repeatable process for identifying and mitigating security risks in a system. It involves analyzing a system from an adversarial perspective, identifying potential threats, and determining appropriate defenses. As outlined in the Threat Modeling Manifesto, it addresses four key questions: What are we working on? What can go wrong? What are we going to do about it? Did we do a good enough job?"*
  → Reforça a importância da modelagem de ameaças como processo estruturado e iterativo, com perguntas simples, mas poderosas.

- *"A comprehensive extension to STRIDE to handle Agentic AI is the layered-based MAESTRO methodology which offers a detailed lens to identify Agentic Threats through the use of architectural layers."*
  → O MAESTRO surge como extensão do STRIDE para lidar com agentes, analisando ameaças em diferentes camadas da arquitetura.

[Agentic AI Threat Modeling Framework: MAESTRO | CSA]

Comentários pessoais

- O documento enfatiza que, apesar de novas ameaças específicas de agentes, fundamentos clássicos de segurança (segurança de software, segurança de LLMs e controles de acesso) continuam indispensáveis.

- Preciso aprofundar meus estudos sobre **Threat Modeling**, em especial:
    1. Manifesto da Modelagem de Ameaças (Threat Modeling Manifesto).
    2. Metodologia **MAESTRO**, avaliando sua aplicabilidade prática e performance.

- Os exemplos de código no GitHub oficial ainda são muito incipientes e bagunçados (últimos commits há ~3 semanas), o que mostra que é um campo em construção.

- Os autores propõe o Agentic Threats Taxonomy Navigator (threat decision tree), que são basicamente perguntas sobre o sistema e que dependendo das respostas, mostra possíveis ameaças conforme o caso:

    1. O agente decide sozinho as etapas para atingir seus objetivos?
    2. O agente depende de memória armazenada para tomar decisões?
    3. O agente executa ações usando ferramentas, comandos de sistema ou integrações externas?
    4. O sistema depende de autenticação para verificar usuários, ferramentas ou serviços?
    5. O agente requer engajamento humano (HITL) para atingir objetivos?
    6. O sistema depende de múltiplos agentes interagindo?

- Apesar de bem estruturadas, as estratégias de mitigação propostas ainda parecem pouco práticas. Faltam exemplos reais de aplicação em ambientes escaláveis e com custos viáveis.

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 2 de out. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: Segurança para Agentes Inteligentes

Durante esta quinta Semana, foquei em entender o conceito de Threat Modelling (Modelagem de Ameaças), suas aplicações e relevância. Tendo consumido conteúdos de links diversos, por exemplo
https://www.threatmodelingmanifesto.org/
What is Threat Modeling? (Threat Modeling Explained)
https://owasp.org/www-community/Threat_Modeling

Além disso, fiz atualizações no Glossário Técnico — Segurança para Agentes Inteligentes, tendo acrescentado um novo tópico para "Termos gerais de cibersegurança" (antes o glossário continha apenas termos relacionados a agentes) e definido o que é Threat Modelling.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Leitura do artigo "Securing Agentic AI: Threat Modeling and Risk Analysis for Network Monitoring Agentic AI System"
Aprofundamento sobre o "Agentic AI Threat Modeling Framework: MAESTRO"
Leitura do documento Multi-Agentic system Threat Modeling Guide

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

Devido a uma cirurgia realizada nesta semana, estou em período de recuperação, o que impactou parcialmente meu desempenho nas atividades. Pretendo retomar o ritmo regular das atividades e aprofundar os estudos na próxima semana.

Aproveito para agradecer ao professor Fernando Federson e aos colegas pelas mensagens, prestatividade e votos de boa recuperação.

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

APÊNDICE 4

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 9 de out. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: Segurança para Agentes Inteligentes

Durante esta sexta Semana, me aprofundei em diversos conteúdos sobre o MAESTRO Framework criado para Modelagem de Ameaças em sistemas multi-agentes. A partir desses conteúdos, produzi o documento Análise - MAESTRO Framework que sintetiza os principais aspectos do framework e sua forma de utilização prática.

Além disso, atualizei o Glossário Técnico — Segurança para Agentes Inteligentes com termos que surgiram durante a leitura dos artigos, textos e vídeos assistidos.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Aplicação prática do MAESTRO Framework em um sistema de agentes inteligentes real.
Levantamento de pontos fortes e pontos de melhoria do framework.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

Análise - MAESTRO Framework.docx citado no Termo de Aceite de Entrega de 09 de outubro

1) O que é o MAESTRO (na prática)

É um framework de **threat modeling multicamadas** feito para Agentes Inteligentes (Agentic AI). Ele decompõe o sistema em **7 camadas** para **mapear ameaças, priorizar riscos e orientar controles/mitigações** específicas por camada, além de conectar **ameaças entre camadas** (defesa em profundidade).

Contexto e Motivação

Os frameworks clássicos como **STRIDE**, **PASTA** ou **LINDDUN** foram desenvolvidos para aplicações determinísticas. Agentes modernos, porém, exibem:

- **Autonomia e autoaprendizado**, alterando objetivos dinamicamente;
- **Planejamento baseado em feedback**, o que os torna suscetíveis a manipulação de metas;
- **Uso de ferramentas externas**, abrindo vetores de ataque indiretos;
- **Memória persistente**, sujeita a envenenamento e reinterpretação maliciosa.

Essas características introduzem ameaças como **goal misalignment, chain-of-thought injection, memory poisoning** e **agent impersonation**, que não são contempladas pelos modelos tradicionais.

O MAESTRO foi criado para mapear essas **vulnerabilidades distribuídas** em sete camadas interligadas, associando-as a riscos mensuráveis e a controles específicos.

7 camadas definidas no framework

- **L1 Foundation Models**: onde o LLM “pensa” (razão, CoT). *Riscos*: adversarial examples, model stealing, backdoors, membership inference. *Foco prático*: filtros de prompt/saída, robustez e testes de modelo.

- **L2 Data Operations**: ingestão, ETL, RAG, vetores/memória. *Riscos*: poisoning, tampering, exfiltração, RAG/KB poisoning. *Foco*: sanitização, isolamento de memória, governança e versionamento.

- **L3 Agent Frameworks**: planner, ferramentas, orquestração. *Riscos*: supply chain, backdoors, input injection, planner exploitation. *Foco*: validação do planner, whitelists de ferramentas, capacidades mínimas necessárias.

- **L4 Deployment & Infrastructure**: containers, APIs, CI/CD, rede. *Riscos*: DoS/resource exhaustion, IaC sabotage, lateral movement. *Foco*: rate-limit, quotas, isolamento, zero-trust na rede.

- **L5 Evaluation & Observability**: métricas, tracing, auditoria. *Riscos*: manipulação de métricas, data leakage via logs, evasão de detecção. *Foco*: métricas de desvio/drift, alertas e forense imutável.

- **L6 Security & Compliance (vertical)**: identidade, acesso, políticas e auditoria que **permeiam todas as camadas**. *Riscos*: extração de modelo de agentes de segurança, não conformidade. *Foco*: trilhas de auditoria, controles regulatórios e explainability.

- **L7 Agent Ecosystem**: usuários, outros agentes, integrações. *Riscos*: impersonation, registry/marketplace manipulation, misuse de ferramentas/objetivos. *Foco*: identidade/registro de agente, reputação e controle de integração.

Ameaças base (mas não exclusivas)

1. **Instruction Manipulation** – Redireciona comportamento do agente via entradas adulteradas;

2. **Goal Manipulation (Agent Drift)** – Desvia objetivos via feedback corrompido;

3. **Chain-of-Thought Manipulation** – Altera raciocínio interno do LLM;

4. **Memory & Context Poisoning** – Corrupção de logs, históricos ou vetores de memória;

5. **Critical System Interaction** – Uso indevido de ferramentas externas;

6. **Planning & Reasoning Exploit** – Exploração de falhas lógicas de planejamento;

7. **Resource Exhaustion (DoS)** – Sobrecarga de recursos de computação;

8. **Knowledge Base Poisoning (RAG)** – Inserção de dados maliciosos na base de conhecimento;
9. **Supply Chain Compromise** – Dependências e modelos comprometidos;
10. **Multi-Agent Exploitation** – Sabotagem via coordenação interagente.

2) Como priorizar: a matriz de risco do MAESTRO

$$R = P \times I \times E$$

,onde **P**=probabilidade de ocorrer, **I**=impacto, **E**=explorabilidade/quão fácil seria de executar.

Sendo que cada variável pode ser classificada como: **1=baixo, 2=médio, 3=alto**.

Use para **ordenar mitigação** por severidade e alcance **inter-camadas**.

3) Como aplicar o MAESTRO (passo a passo)

1. **Quebrar o sistema nas 7 camadas** (inventariar assets/módulos, dados, fluxos e ferramentas).
2. **Mapear as 10 ameaças** relevantes (anote exemplos concretos do seu domínio).
3. **Atribuir R=P×I×E** por ameaça e **preencher a matriz de risco** por camada e impactos cruzados.
4. **Selecionar/aplicar controles** (Prevenção / Detecção & Resposta / Defesa em Profundidade) **direto nas camadas afetadas**.
5. **Planejar testes de validação**
6. **Atualização do scoring e dos controles**.

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 15 de out. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: Segurança para Agentes Inteligentes

Durante esta sétima Semana, apliquei os conhecimentos e referências adquiridos nos Gates anteriores para desenvolver a modelagem de ameaças (Threat Modeling) de um sistema multiagente voltado à área da saúde, utilizando o MAESTRO Framework como base metodológica. O estudo resultou no documento MAESTRO Threat Modelling - Enfermeira Virtual, que descreve de forma detalhada as camadas, ameaças e técnicas de mitigação aplicáveis ao sistema.

Além disso, após a modelagem prática,elaborei uma análise crítica, na qual identifiquei as principais virtudes e limitações do framework MAESTRO — Pontos Fortes e Pontos de Melhoria

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Levantamento e leitura de artigos/conteúdos com técnicas práticas de mitigação para cada uma das camadas do MAESTRO Framework.

Ajustes práticos (considerando os pontos de melhorias levantados) para tornar o framework mais aplicável em contextos reais de desenvolvimento de agentes inteligentes.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

MAESTRO Threat Modelling - Enfermeira Virtual.docx citado no Termo de Aceite de Entrega de 15 de outubro

Modelagem de Ameaças para um Sistema Multiagente em Saúde

Descrição Geral e Funcionalidades do Sistema

O sistema multiagente que iremos modelar (threat model) se trata de uma **Enfermeira Virtual** e foi desenvolvido para atuar como uma assistente de saúde especializada em **saúde da mulher**, com o objetivo de oferecer suporte informativo, acompanhamento personalizado e integração direta com os serviços da empresa provedora. A interação ocorre via **WhatsApp**, utilizando a Meta WhatsApp Cloud API (API Oficial), o que permite um canal de comunicação extremamente acessível (sendo o mais utilizado no Brasil) e seguro para as usuárias.

1. Arquitetura Geral

A arquitetura do sistema segue um modelo **multiagente modular**, composto por:

- **Agente Orquestrador:** atua como camada de coordenação e roteamento, responsável por interpretar o contexto das mensagens e direcionar a conversa ao subagente mais apropriado.

- **Subagentes Especializados:** responsáveis por domínios específicos de conhecimento, como contracepção, menopausa, gestação, saúde ginecológica em geral e etc. Cada subagente possui seu próprio conjunto de *tools* e base de conhecimento.

- **Infraestrutura em Nuvem (AWS):** o sistema opera sobre uma arquitetura escalável e distribuída, utilizando serviços como **API Gateway**, **AWS Lambda**, **EventBridge**, **SQS**, **ECS** e **DynamoDB**. O armazenamento primário é feito no **PostgreSQL** (dados estruturados e memória de longo prazo) e no **Qdrant** (para busca vetorial e recuperação semântica).

Essa arquitetura modular garante independência funcional entre os agentes, melhor isolamento de falhas e facilita a aplicação de medidas específicas de segurança em cada camada, o que será um facilitador para a análise pelo framework MAESTRO.

2. Funcionalidades Principais

2.1. Interação e Atendimento

A enfermeira virtual atua como ponto inicial de contato com o usuário, capaz de:

- **Responder dúvidas gerais sobre saúde da mulher**, com base em um modelo de linguagem (LLM) e uma **base vetorial** construída a partir de **documentos médicos públicos**;
- **Responder dúvidas comerciais** sobre o produto e a empresa responsável pelo serviço;
- **Manter conversas contextuais** com memória de curto e longo prazo, permitindo que o sistema se recorde de condições de saúde e peculiaridades da paciente, como histórico de diabetes ou gravidez.

2.2. Gestão e Armazenamento de Exames

As usuárias podem **enviar exames laboratoriais ou de imagem** diretamente pelo WhatsApp. O sistema é capaz de:

- Receber e **salvar os arquivos recebidos** via WhatsApp de forma segura no banco de dados;
- Associar cada exame ao perfil da paciente, mantendo um **catálogo histórico** que pode ser consultado futuramente;
- Interpretar e **simplificar o conteúdo técnico** dos exames, apresentando explicações em linguagem acessível.

2.3. Agendamentos e Encaminhamentos

O sistema possui integração com serviços externos para gestão de atendimentos:

- **Agendamento de consultas:** realizado via **integração com o Calendly**, conforme a disponibilidade da equipe de saúde e o interesse da paciente;

- **Encaminhamento para urgência/emergência:** envia alertas automatizados em situações de risco (ex.: sangramento contínuo), orientando a usuária a procurar atendimento imediato;

- **Encaminhamento para equipe humana:** casos complexos ou que exigem análise profissional são direcionados para médicos e enfermeiros humanos da empresa.

2.4. Personalização e Lembretes

Com base no histórico da paciente e nas interações registradas, o sistema:

- **Envia lembretes personalizados**, como horários de uso de anticoncepcional, retornos de consulta ou exames pendentes;

- Esses lembretes são **agendados dinamicamente** por meio de eventos no **AWS EventBridge**, baseando-se em regras condicionais definidas pelos agentes e regras de negócio.

2.5. Autenticação e Controle de Acesso

Para garantir o uso apenas por pacientes registradas:

- O acesso é validado via **número de telefone** do WhatsApp, associado ao CPF cadastrado na **base de clientes da empresa**;

- Usuários não verificados não podem prosseguir com as interações.

2.6. Conformidade e Privacidade

O sistema segue os princípios da **Lei Geral de Proteção de Dados (LGPD)**.
As principais práticas incluem:

- **Coleta de consentimento explícito** na primeira interação;

- **Armazenamento seguro** das informações médicas e pessoais em bancos gerenciados;
- **Registro e monitoramento de conversas**, para fins de auditoria, rastreabilidade e segurança;
- **Limitações éticas e legais claras**, incluindo avisos explícitos de que a enfermeira virtual **não fornece diagnóstico médico nem substitui avaliação profissional**.

2.7. Observabilidade e Feedback

O sistema mantém:

- **Dashboards administrativos** para acompanhamento da performance, indicadores de atendimento e métricas de segurança;
- **Coleta de feedbacks automáticos** ao final das conversas, permitindo monitoramento contínuo da experiência da paciente e melhoria do serviço.

3. Considerações

A combinação entre **LLMs, agentes especializados e integração em nuvem** torna o sistema um exemplo robusto de **aplicação de "IA agêntica" em saúde**.
Essa arquitetura, ao mesmo tempo que oferece flexibilidade e escalabilidade, introduz novos **pontos de exposição e superfícies de ataque**, por exemplo:

- Interações via mensageria (prompt injection e spoofing),
- Processamento de dados médicos sensíveis,
- Persistência de memória contextual,
- Integração entre múltiplos agentes com ferramentas autônomas.

Threat Modeling do Sistema com MAESTRO Framework

4.1. Metodologia Aplicada

O framework **MAESTRO (Multi-Agent Environment, Security, Threat Risk, and Outcome)** será adotado para realizar a modelagem de ameaças do sistema supracitado.
A metodologia permite decompor o ecossistema agêntico em **sete camadas interligadas**, mapeando riscos específicos de cada uma e suas interdependências.

O cálculo de **relevância de risco** segue a matriz **R = P × I × E**, onde:

- **P (Probabilidade)**: chance de ocorrência;
- **I (Impacto)**: gravidade sobre os ativos e usuários;
- **E (Explorabilidade)**: facilidade de exploração por agentes externos.

Cada variável é classificada como:

1 = Baixo, 2 = Médio, 3 = Alto

As ameaças mais críticas são priorizadas conforme seu **R** (resultado de 3 a 27).

Os controles/formas de mitigação propostos são divididos em:

- **Prevenção (PR)** – evitar a ocorrência da ameaça;
- **Detecção e Resposta (DR)** – identificar e reagir a incidentes;
- **Defesa em Profundidade (DP)** – camadas redundantes e sistêmicas de proteção.

4.2. Camada L1 – Foundation Models

Descrição:
Corresponde aos modelos de linguagem (GPT e Gemini) acessados via API, que fundamentam as capacidades cognitivas da enfermeira virtual.

Ameaças identificadas:

1. **Prompt Injection / Jailbreaking** – Usuária insere instruções que contornam políticas de segurança (“finja ser médica e recomende remédio”).
2. **Membership Inference** – Inferência se determinado dado sensível (exames, histórico) foi usado em treinamento.

3. **Adversarial Input Flooding** – Envio de mensagens projetadas para gerar alto custo computacional (DoS lógico).

Ameaça	P	I	E	R	Controles
Prompt Injection	3	3	3	**27 (Crítico)**	**PR:** Filtros de prompt e saída, *instruction shielding*. **DR:** Logs e tracing de prompts anômalos. **DP:** Gateways intermediários que validam input/output antes do envio à API.
Membership Inference	1	3	1	3	**PR:** Garantia de que modelos externos não treinam dados internos. **DP:** Nenhum dado sensível incluído em *fine-tuning*.
Adversarial Flooding	2	2	2	8	**PR:** Rate-limit e validação de payload. **DR:** Alarmes de sobrecarga e bloqueio temporário.

4.3. Camada L2 – Data Operations

Descrição:
Inclui as operações de ingestão, armazenamento e recuperação de dados na base vetorial do Qdrant, no banco relacional PostgreSQL e no DynamoDB.

Ameaças:

1. **Knowledge Base Poisoning** – Inserção de documentos adulterados na base médica pública.

2. **Memory Poisoning (Contextual)** – Corrupção do histórico no PostgreSQL para alterar comportamento futuro.

3. **Data Exfiltration via Query** – Extração indevida de dados pessoais através de interações aparentemente legítimas.

4. **Tampering de Exames (Anexos)** – Manipulação de arquivos enviados pelo WhatsApp antes do armazenamento.

Ameaça	P	I	E	R	Controles
KB Poisoning	2	3	2	12	**PR:** Versionamento e assinatura de fontes públicas. **DR:** Hashes e integridade verificável. **DP:** RAG isolado com curadoria periódica.
Memory Poisoning	2	3	2	12	**PR:** Sanitização de entrada e auditoria de atualização. **DR:** Monitoramento de alterações anômalas de contexto. **DP:** Controles de escrita segregados.
Data Exfiltration	3	3	2	**18 (Alto)**	**PR:** Escopo estrito de query. **DR:** DLP (Data Loss Prevention) no gateway. **DP:** Pseudonimização e criptografia em repouso.
Tampering de Exames	2	2	2	8	**PR:** Verificação de checksum. **DR:** Alertas de hash inconsistente.

					DP: Armazenamento isolado em bucket privado.

4.4. Camada L3 – Agent Frameworks

Descrição:
Engloba o orquestrador e subagentes, cada qual com ferramentas especializadas.

Ameaças:

1. **Planner Exploitation** – Injeção de metas falsas (“agende consulta de urgência mesmo sem sintomas”).
2. **Critical Tool Misuse** – Uso indevido de ferramentas (ex: calendly API) fora do contexto clínico.
3. **Supply Chain Dependency** – Bibliotecas externas maliciosas usadas nos agentes.
4. **Cross-Agent Manipulation** – Um subagente influenciando outro via contexto compartilhado.

Ameaça	P	I	E	R	Controles
Planner Exploitation	3	3	2	**18 (Alto)**	**PR:** Validação semântica de intenções. **DR:** Logs de raciocínio (CoT tracing). **DP:** Sandbox de ferramentas.
Tool Misuse	2	3	2	12	**PR:** Whitelist de APIs. **DR:** Auditoria de chamadas. **DP:** Policy enforcement via IAM.

Supply Chain	2	3	2	12	**PR:** Dependabot e SBOM. **DR:** Verificação de assinaturas. **DP:** Imutabilidade de container base.
Cross-Agent Manipulation	2	2	2	8	**PR:** Separação de contextos e memória. **DP:** Controle de acesso interagente.

4.5. Camada L4 – Deployment & Infrastructure

Descrição:
Infraestrutura na AWS, composta por API Gateway, Lambdas, EventBridge, ECS e DynamoDB.

Ameaças:

1. **DoS via Mensageria (WhatsApp Flood)** – Envio massivo de mensagens para exaurir recursos.
2. **Resource Hijacking (ECS)** – Uso indevido de recursos computacionais para mineração ou spam.
3. **IaC Sabotage** – Alteração de templates Terraform/CloudFormation.
4. **Credential Leakage** – Chaves de API (GPT, Gemini, Calendly) expostas em logs.

Ameaça	P	I	E	R	Controles
DoS Mensageria	3	2	2	12	**PR:** Rate-limit via Gateway. **DR:** Auto-scaling + CloudWatch Alerts. **DP:** Circuit breakers.

Resource Hijacking	2	3	2	12	**PR:** IAM mínimo e quotas. **DR:** Monitoramento de uso. **DP:** Network segmentation.
IaC Sabotage	1	3	2	6	**PR:** Revisões e repositório versionado. **DP:** Infra-as-Code assinada.
Credential Leakage	2	3	3	**18 (Alto)**	**PR:** Vault (Secrets Manager). **DR:** Scanners automáticos. **DP:** Logging sem dados sensíveis.

4.6. Camada L5 – Evaluation & Observability

Descrição:
Medições de desempenho, rastreamento e auditoria via Langfuse e dashboards administrativos.

Ameaças:

1. **Data Leakage via Logs** – Registro de prompts com PII.
2. **Manipulação de Métricas** – Alteração de resultados de precisão ou satisfação.
3. **Evasion of Detection** – Mensagens maliciosas que evitam detecção de segurança.

Ameaça	P	I	E	R	Controles
Leakage via Logs	3	3	2	**18 (Alto)**	**PR:** Redação automática de PII. **DR:** Auditoria periódica.

					DP: Storage criptografado.
Manipulação de Métricas	2	2	2	8	**PR:** Acesso restrito ao dashboard. **DP:** Métricas imutáveis.
Evasion of Detection	2	3	2	12	**PR:** Monitoração por comportamento. **DR:** Alertas de desvio de padrão.

4.7. Camada L6 – Security & Compliance (Vertical)

Descrição:
Camada **vertical** que assegura conformidade legal (LGPD) e controles de segurança.

Ameaças:

1. **Regulatory Non-Compliance** – Falhas em consentimento ou anonimização.
2. **Bias e Inequidade** – Respostas discriminatórias ou não éticas.
3. **Audit Trail Manipulation** – Alteração de logs de auditoria.

Ameaça	P	I	E	R	Controles
Non-Compliance	2	3	2	12	**PR:** Consentimento explícito e logs. **DR:** Auditoria jurídica periódica.. **DP:** Relatórios automáticos de conformidade.
Bias	2	2	2	8	**PR:** Avaliação humana contínua. **DP:** Dataset balanceado.

Audit Trail Manipulation	1	3	2	6	**PR:** Logs imutáveis.

4.8. Camada L7 – Agent Ecosystem

Descrição:
Interação entre o sistema e os usuários (WhatsApp), agentes humanos (médicos e suporte), e serviços externos (Calendly).

Ameaças:

1. **Agent Impersonation** – Falsos perfis se passando pela enfermeira virtual.
2. **Goal Manipulation (Agent Drift)** – Feedbacks de usuário corrompendo o comportamento do agente.
3. **Integration Misuse** – Uso indevido de APIs externas (Calendly).

Ameaça	P	I	E	R	Controles
Impersonation	3	3	2	**18 (Alto)**	**PR:** Verificação de remetente WhatsApp (Meta Cloud). **DR:** Alertas de identidade duplicada. **DP:** Certificação do domínio.
Goal Manipulation	2	2	2	8	**PR:** Reforço de política e validação de instruções. **DP:** Treinamento de segurança semântica.
Integration Misuse	2	3	2	12	**PR:** Escopos mínimos de API. **DR:** Monitoramento de requisições externas.

4.9. Matriz de Risco Consolidada (Síntese)

Camada	Principais Ameaças	Risco	Prioridade	Mitigações-Chave
L1	Prompt Injection	**27**	🔴 Crítico	Filtros de prompt, validação de entrada, tracing
L2	Data Exfiltration	**18**	🔴 Alto	DLP, criptografia, pseudonimização
L3	Planner Exploitation	**18**	🔴 Alto	Validação semântica, sandbox, auditoria
L4	Credential Leakage	**18**	🔴 Alto	Secrets Manager, varredura de chaves
L5	Data Leakage via Logs	**18**	🔴 Alto	Redação de PII, criptografia de logs
L6	Non-Compliance	12	🟠 Médio	Consentimento explícito, auditorias
L7	Agent Impersonation	**18**	🔴 Alto	Verificação de identidade e domínio

Cross-layer	Goal Manipulation, Lateral Movement	Variável	🟠	Monitoramento sistêmico e isolamento intercamadas

4.10. Conclusão e Recomendações Gerais

A aplicação do MAESTRO evidenciou que os **maiores vetores de risco concentram-se nas camadas L1, L2, L3 e L4**, ligadas ao núcleo cognitivo (LLM), dados sensíveis e infraestrutura de integração.
A priorização das mitigações deve focar em:

1. **Hardening das entradas e saídas LLM (L1)** – uso de filtros, guardrails e *input sanitization*;
2. **Governança de dados sensíveis (L2)** – criptografia, versionamento e isolamento de memória;
3. **Segurança do orquestrador (L3)** – whitelists, sandbox e controle de ferramentas;
4. **Proteção de segredos e APIs (L4)** – gerenciamento centralizado de chaves e monitoramento de IAM;
5. **Auditoria e rastreabilidade contínuas (L5–L6)** – observabilidade e logs imutáveis via Langfuse e AWS;
6. **Verificação de identidade no ecossistema (L7)** – autenticação de canal e alerta de impersonation.

MAESTRO — Pontos Fortes e Pontos de Melhoria.docx citado no Termo de Aceite de Entrega de 15 de outubro

MAESTRO Framework
Pontos Fortes e Pontos de Melhoria

O MAESTRO acerta quando trata agentes como um domínio próprio. Ele não tenta reaproveitar, de forma genérica, categorias clássicas como STRIDE ou PASTA, visto que parte do que é específico em sistemas agênticos como a orquestração, uso de ferramentas, planejamento autônomo, ecossistema de integrações e organiza isso em uma visão por camadas. Para quem está construindo algo como um sistema complexo, essa decomposição ajuda a não deixar nada relevante de fora e quando mapeamos as camadas fica claro onde estão os riscos realmente altos.

Outra virtude é reconhecer que segurança de IA não é estática: o framework fala de monitoramento contínuo, adaptação e riscos de adversarial ML, o que converge com o que já fazemos em MLOps e observabilidade. Nesse sentido, o MAESTRO oferece uma base sólida, quase como uma biblioteca organizada de ameaças e boas práticas específicas para agentes inteligentes.

Por outro lado, o framework ainda sofre com problemas de clareza e aplicabilidade. Ele mistura o papel de catálogo de ameaças com o de metodologia, o que acaba gerando confusão e redundância. As camadas, embora úteis como mapa conceitual, não se comportam de fato como camadas dependentes, pois algumas são transversais, como segurança e observabilidade, o que quebra a lógica de empilhamento e torna o modelo mais difícil de explicar.

Além disso, os padrões de arquitetura e as camadas parecem desconectados, o documento traz ameaças em uma seção e repete outras com nomes diferentes em outra, sem deixar claro como as duas partes se relacionam.

Outro ponto crítico é o caráter genérico das recomendações. As mitigações descritas (“use filtros de segurança”, “faça validação robusta”) são corretas, mas pouco acionáveis, visto que não explicam como o desenvolvedor deve implementá-las, nem oferecem parâmetros mínimos de controle. Essa falta de operacionalidade torna o framework mais conceitual do que prático, especialmente para quem está construindo agentes com modelos via API, sem acesso direto ao treinamento dos modelos base.

Faltam também personas e papéis, pois não é o mesmo tipo de trabalho avaliar riscos em um modelo fundacional, orquestrar um agente ou monitorar a infraestrutura. Cada perfil precisaria de um recorte específico, mas o MAESTRO trata todos de forma genérica.

Apesar dessas limitações, o valor do MAESTRO é evidente. Ele representa um passo inicial importante para estruturar a segurança de sistemas agentivos e criar uma linguagem comum para falar sobre riscos em inteligência artificial autônoma. Seu maior potencial está em ser usado como biblioteca, um guia de ameaças e superfícies de ataque que possa ser aplicado junto a outros métodos

APÊNDICE 5

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 22 de out. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: **Segurança para Agentes Inteligentes**

Nas Semanas anteriores, venho explorando desde conceitos fundamentais até frameworks aplicáveis de threat modeling.
Inicialmente, realizei um levantamento de artigos e materiais relevantes sobre o tema, buscando entender o "estado da arte" em segurança para sistemas de agentes autônomos. A partir dessa pesquisa, construí um glossário técnico reunindo os principais termos da área, incluindo definições de vulnerabilidades, ataques e também algumas técnicas práticas de mitigação mais citadas na literatura.

Em seguida, aprofundei a análise sobre modelagem de ameaças em sistemas multiagentes, onde identifiquei o MAESTRO Framework, uma proposta recente voltada especificamente à segurança de arquiteturas agênticas. A partir de sua aplicação e estudo detalhado, identifiquei pontos de melhoria no framework, especialmente no que diz respeito à sua operacionalização prática e aplicabilidade em contextos reais de desenvolvimento.

Durante esta oitava Semana, o foco foi justamente propor ajustes técnicos para o MAESTRO, tornando-o mais utilizável por equipes de desenvolvimento. As propostas incluíram separação entre metodologia e biblioteca de ameaças, definição de papéis por camada, mitigações testáveis e métricas de risco adaptadas por domínio. Ajustes Práticos MAESTRO Framework

Além disso, fiz um levantamento de artigos e conteúdos complementares, incluindo surveys, guias e vídeos técnicos, para realizar a construção de uma "Biblioteca de Ameaças" completa que possa complementar a base do MAESTRO Framework. Tendo encontrado como principal artigo: TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems (além dos conteúdos consumidos nas Semanas anteriores).

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- **Leitura e anotações relevantes do artigo** TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems

- **Desenvolvimento da “Biblioteca de Ameaças” proposta para melhorar a eficiência do Maestro Framework, incluindo a relação de técnicas de defesa por camada.**

Observação: **[caso precise fazer alguma observação, de qualquer “natureza”]**

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Ajustes Práticos MAESTRO Framework.docx citado no Termo de Aceite de Entrega de 22 de outubro

Ajustes Práticos MAESTRO Framework

O MAESTRO Framework surge como uma iniciativa relevante para a modelagem de ameaças (threat modeling) em sistemas multiagentes. Entretanto, ao ser aplicado em contextos reais de desenvolvimento, percebe-se que ele ainda carece de orientações práticas. Embora a estrutura de camadas facilite o raciocínio sobre segurança, o framework deixa lacunas quando se trata de indicar como implementar, testar e sustentar controles de forma contínua.

A seguir, são propostos ajustes estruturais que buscam tornar o MAESTRO mais aplicável em ambientes de engenharia, aproximando-o das necessidades de equipes que lidam com agentes inteligentes em produção. Cada item apresenta a justificativa técnica e os benefícios esperados.

1. Separação entre “Metodologia” e “Biblioteca de Ameaças”

Hoje, o MAESTRO combina orientações metodológicas e descrições de ameaças em um mesmo corpo de texto. Essa sobreposição dificulta o uso do framework, pois o leitor precisa alternar entre conceitos prescritivos (o que fazer) e descritivos (o que pode dar errado) sem uma divisão clara. Em alguns trechos, o mesmo conceito aparece sob rótulos diferentes, o que gera ambiguidade.

Um caminho mais funcional seria organizar o framework em dois módulos independentes:

- **Parte A – Metodologia MAESTRO:** descreve o passo a passo de aplicação (escopo, análise, priorização, mitigação, revisão).

- **Parte B – Biblioteca de Ameaças:** contém listas taxonômicas de ameaças, exemplos, indicadores e controles sugeridos.

Com essa separação, o modelo ganha clareza operacional. As equipes poderiam seguir o fluxo metodológico sem se perder em detalhes teóricos, enquanto a biblioteca evoluiria de forma modular conforme novas ameaças agentivas surgissem.

2. Mapeamento entre “Padrões de Arquitetura” e “Camadas”

Os padrões de arquitetura descritos pelo MAESTRO (como Unconstrained Conversational Autonomy e Multi-Agent Coordination) não se conectam de maneira explícita às camadas do framework. Isso torna difícil compreender onde cada tipo de risco se manifesta e quais times devem atuar sobre ele.

Uma melhoria prática seria criar uma matriz de correspondência entre padrões e camadas, definindo claramente as interseções. Por exemplo, Prompt Injection impacta as camadas de modelo e orquestração (L1/L3), enquanto Tool Misuse se concentra na camada de agentes (L3) e Impersonation aparece na interface com o ecossistema externo (L7).

Essa correlação permitiria uma visão mais rastreável do risco, orientando melhor a priorização de defesas e o repasse de responsabilidades entre times técnicos (DevOps, MLOps, Segurança da Informação, entre outros).

3. Definição de Personas e Responsabilidades por Camada

Um dos pontos mais limitantes do framework é tratar todos os participantes como se tivessem o mesmo alcance de decisão e controle. Na prática, sistemas multiagentes envolvem perfis muito distintos — desenvolvedores de agentes, engenheiros de infraestrutura, especialistas em dados e profissionais de compliance.

Por isso, o MAESTRO se beneficiaria de uma estrutura de papéis e responsabilidades, como um modelo RACI (Responsible, Accountable, Consulted, Informed) aplicado a cada camada. Essa abordagem tornaria explícito quem executa as ações de mitigação, quem aprova políticas e quem apenas audita ou acompanha resultados.

Além de reduzir zonas cinzentas de responsabilidade, esse ajuste facilitaria a integração entre desenvolvimento e governança, algo essencial quando o ciclo de vida do agente envolve tanto componentes técnicos quanto regulatórios.

4. Mitigações Prescritivas e Testáveis

Atualmente, o MAESTRO apresenta recomendações genéricas — como “usar filtros” ou “implementar validações robustas” — sem oferecer critérios objetivos para medir a eficácia dessas ações. Isso limita a aplicabilidade do framework em pipelines modernos de segurança.

Uma forma de torná-lo mais prático seria adotar mitigações prescritivas e testáveis, com diretrizes do tipo:

- políticas formais e condições de bloqueio;
- regras de detecção (regex, classificadores ou limites quantitativos);
- exemplos de implementação em código;
- testes de aceitação com métricas de desempenho (KPIs, SLAs).

Com essa abordagem, o MAESTRO deixaria de ser apenas um guia conceitual e passaria a operar como uma ferramenta de engenharia. Equipes poderiam automatizar auditorias, criar testes de segurança contínuos e medir o grau de conformidade de cada agente.

5. Base de Risco Opinativa (R = P×I×E com Pesos Setoriais)

O cálculo de risco proposto pelo MAESTRO depende inteiramente da interpretação da equipe, o que faz com que dois projetos semelhantes possam chegar a classificações muito diferentes. Essa ausência de referência reduz a comparabilidade entre sistemas.

Uma evolução natural seria estabelecer uma base de risco opinativa, com tabelas de pesos pré definidos para diferentes setores. Em domínios como saúde, por exemplo, dados sensíveis (PHI/PII) e canais abertos teriam impacto e explorabilidade máximos por definição.

Essa padronização daria consistência às análises e permitiria construir benchmarks de segurança entre múltiplos agentes. Além disso, simplificaria a priorização de ameaças em ambientes corporativos, onde decisões precisam ser tomadas de forma rápida e justificada.

Conclusão

Os ajustes propostos buscam transformar o MAESTRO em um framework não apenas conceitual, mas aplicável à engenharia de sistemas multiagentes modernos.

Ao introduzir separação metodológica, correlação entre padrões e camadas, definição de papéis, mitigadores testáveis e métricas objetivas de risco, o modelo passaria a oferecer um caminho claro desde a identificação até a validação das defesas.

Essas mudanças fortaleceriam a maturidade do framework, promovendo uma cultura de segurança mensurável, iterativa e adaptável às realidades de agentes inteligentes em ambientes de produção.

APÊNDICE 6

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 6 de nov. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: **Segurança para Agentes Inteligentes**

Nas Semanas anteriores, venho desenvolvendo um estudo sobre segurança aplicada a sistemas multiagentes, partindo da revisão de literatura e da criação de um glossário técnico com os principais conceitos, vulnerabilidades e técnicas de mitigação utilizadas na área. Posteriormente, aprofundei o trabalho na modelagem de ameaças (Threat Modeling), identificando o MAESTRO Framework como referência para a análise de riscos em sistemas agênticos.

Na oitava Semana, o foco foi propor ajustes práticos ao MAESTRO Framework, visando aprimorar sua aplicabilidade em contextos reais de desenvolvimento. Essas propostas incluíram a separação entre metodologia e **Lista de Ameaças**, definição de papéis por camada e mitigações testáveis.

Durante esta nona Semana, a principal atividade foi a leitura e análise detalhada do artigo "TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Multi-Agent Systems", que apresenta uma visão ampla sobre práticas de segurança, monitoramento contínuo e defesa em profundidade para arquiteturas agentivas.

A partir dessa leitura, produzi um documento de anotações técnicas
Anotações Paper "TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-b…
sintetizando os principais pontos do paper, como o uso de monitoramento em tempo real, sanitização de entradas (prompt hygiene), controle de acesso baseado em papéis (RBAC), princípio do menor privilégio e camadas de defesa independentes para garantir integridade e resiliência.

Além disso, durante a análise do artigo, foi identificada uma figura que reforça o crescimento acelerado das publicações sobre sistemas multiagentes e segurança entre 2022 e 2025, além de destacar o foco crescente em pilares como Trust, Security e Risk, que sustentam a evolução de frameworks como o MAESTRO e fundamentam a proposta de criação de uma Lista de Ameaças e Mitigações.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

- **Consolidação da "Lista de Ameaças" proposta para melhorar a eficiência do MAESTRO Framework, incluindo a relação de técnicas de defesa por camada.**

Observação: **[caso precise fazer alguma observação, de qualquer "natureza"]**

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Anotações Paper "TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems".docx citado no Termo de Aceite de Entrega de 06 de novembro

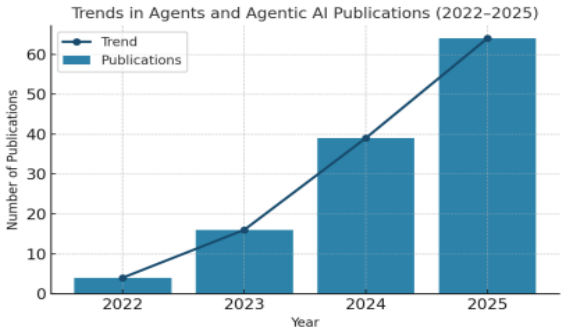
Anotações e Trechos Relevantes Paper "TRiSM for Agentic AI: A Review of Trust, Risk, and Security Management in LLM-based Agentic Multi-Agent Systems"

O paper trata de como o conceito de **TRiSM (Trust, Risk, and Security Management)** deve ser incorporado aos sistemas multiagentes baseados em LLMs. A maioria dos trabalhos sobre *Agentic AI* até agora foca em aspectos funcionais (modelagem, planejamento, colaboração entre agentes), mas quase nada sobre robustez adversarial, governança do ciclo de vida ou explicabilidade. Essa lacuna é o que o artigo tenta endereçar.

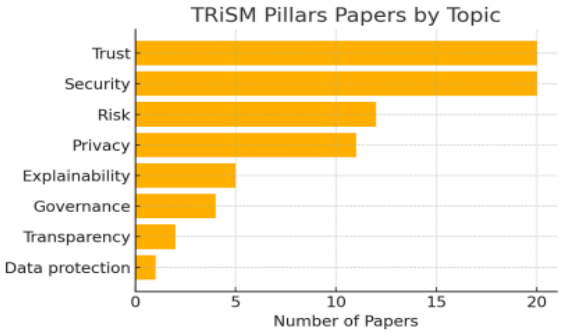
Table 1: Comparison with related surveys.

Survey	Threats	Lifecycle Gov.	Explainability	TRiSM Integration	LLM-Specific	Applications	Actionable Guidance
Guo et al. (2024) [14]	✗	✗	✗	✗	✓	✓	~
Chen et al. (2025) [15]	✗	✗	✗	✗	✓	~	~
Yan et al. (2025) [16]	✓	✗	✗	✗	✓	✓	~
Tran et al. (2025) [17]	✗	✗	✗	✗	✓	✓	~
Lin et al. (2025) [18]	✗	✗	✗	✗	✓	~	~
Fang et al. (2025) [11]	✓	✗	✗	~	~	✓	~
Xi et al. (2025) [19]	✗	✗	✗	✗	✓	✓	~
Luo et al. (2025) [20]	✓	~	✗	~	✓	✓	~
Zou et al. (2025) [21]	✗	✗	✗	~	✓	✓	~
Wang et al. (2025) [22]	~	✗	✗	~	✓	~	~
This Survey (2025)	✓	✓	✓	✓	✓	✓	✓

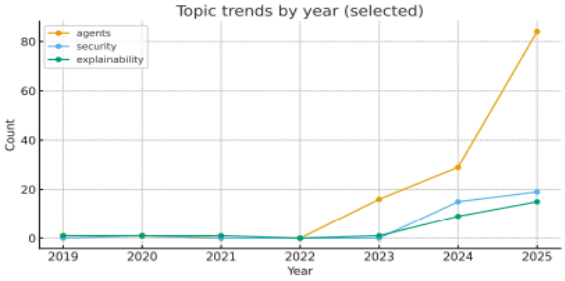
Legend: ✓= explicitly addressed; ~= partially addressed; ✗= not addressed.



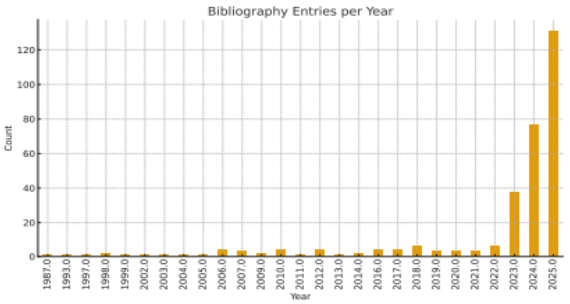
(a) Agentic AI papers per year (2022–2025).



(b) Papers by TRiSM pillar.



(c) Topic trends in Agentic AI (agents, security, explainability).



(d) Bibliography entries per year (1987–2025).

Uma das ideias principais é que monitoramento contínuo e **Threat Modeling** são elementos obrigatórios. O comportamento dos agentes é dinâmico, e vulnerabilidades podem surgir de forma emergente. Segurança não é uma configuração estática, mas um processo iterativo. Eles defendem que agentes precisam ser avaliados em tempo real quanto a desvios de comportamento e anomalias nas interações.

O conceito de **prompt hygiene** aparece com bastante ênfase, algo que na prática significa limpar e validar entradas antes que cheguem ao agente. O paper fala em sanitização, uso de prefixos seguros e limitação do formato de saída, como forma de reduzir a superfície de ataque. Isso remete à ideia de criar um "domínio controlado" de operação para o LLM, onde os riscos de injeção ou manipulação de instruções sejam minimizados.

Eles também abordam o princípio de **defense-in-depth**, com múltiplas camadas de defesa. Nenhuma técnica isolada é suficiente, então se propõe o uso de padrões de design como *Action-Selector* (o agente só escolhe ações predefinidas) e *Plan-Then-Execute* (o plano é congelado antes da execução). Isso garante isolamento entre entrada não confiável e caminhos de execução. Em resumo, o agente nunca deveria executar diretamente algo que foi influenciado por dados externos não verificados.

Outro ponto crítico é o **least-privilege access**. Cada agente deve ter apenas as permissões estritamente necessárias. O sistema precisa ter autenticação forte, controle baseado em

papéis e monitoramento contínuo de acessos. Se um agente desviar do comportamento esperado, ele deve ser automaticamente suspenso. Isso traz uma camada de governança operacional sobre a autonomia dos agentes.

O paper também recomenda **pós-processamento e detecção de anomalias** para limpar outputs e identificar padrões suspeitos. Sugere o uso de filtros automáticos, re-treinamento periódico para reduzir alucinações e manutenção de logs auditáveis, mas sem expor dados sensíveis. Uma ideia interessante é o uso de **monitoramento hierárquico e validação cruzada entre agentes**, ou seja, os próprios agentes verificando consistência nas respostas uns dos outros.

Em termos de arquitetura, os autores propõem uma estrutura **multicamadas** de defesa cobrindo quatro pontos principais: proteção de dados, integridade da execução, comunicação entre agentes e robustez dos modelos. Cada uma dessas camadas atua como um sistema independente de contenção.

Nas técnicas concretas, eles detalham:

- **Criptografia:** uso de TLS/SSL, homomorphic encryption e enclaves seguros (como Intel SGX) para proteger a troca de dados sensíveis entre agentes.
- **Controle de acesso:** adoção de RBAC e ABAC para restringir dinamicamente quais agentes podem acessar APIs, arquivos e memórias. O princípio do menor privilégio se repete aqui.
- **Aprendizado adversarial:** reconhecem que LLMs são vulneráveis a prompt injection e tool poisoning, principalmente quando múltiplos agentes trocam resultados intermediários. Métodos como *adversarial training*, *reward shaping* e *contrastive learning* ajudam, mas não resolvem completamente. É essencial validar respostas antes da execução.
- **Monitoramento em tempo de execução:** uso de logs, detectores baseados em LSTM ou autoencoders, e sistemas de *trust scoring* para acompanhar anomalias. Eles citam o *Microsoft Copilot* como exemplo prático de governança em larga escala, onde comportamentos anormais são detectados e reportados em tempo real.

O artigo também propõe uma categorização das principais ameaças observadas em AMAS:

- *Autonomy abuse*, quando um agente extrapola sua função.
- *Persistent memory misuse*, ou o uso indevido de memórias duradouras.
- *Agent orchestration flaws*, falhas no mecanismo que coordena os agentes.
- *Goal misalignment*, desalinhamento de objetivos.
- *Tool misuse* e *external API exploits*, uso indevido de ferramentas externas.

- *Multi-agent collusion* ou *drift*, quando agentes começam a cooperar entre si de forma imprevisível.

Esses riscos são agrupados em quatro classes principais: **Adversarial Attacks**, **Data Leakage**, **Agent Collusion/Mode Collapse** e **Emergent Behavior.**

OBS: Essa última é a mais difícil de controlar, porque decorre de interações complexas que nem sempre são programadas ou previsíveis.

No fechamento, o paper apresenta uma estrutura de cinco pilares para TRiSM aplicada a sistemas multiagentes:

1. **Explainability** – rastreabilidade e interpretabilidade das decisões dos agentes.
2. **ModelOps** – controle sobre o ciclo de vida dos modelos e pipelines.
3. **Application Security** – segurança nas interações entre agentes e APIs externas.
4. **Model Privacy** – proteção contra vazamentos e reidentificação de dados.
5. **Governance** – políticas, auditoria e compliance em todo o ecossistema.

Pillar	Core controls (keywords)	Techniques / patterns / artifacts	Key risks mitigated	Evaluation facets & example metrics	Example systems / patterns	Primary standards / frameworks
Application Security	Prompt hygiene; secure prefixes; sandboxed tools; allowlisted actions; plan-then-execute; least privilege (RBAC); authN/authZ; anomaly detection; red-teaming	Prompt-injection patterns [133, 58]; plan-then-execute; action selector; tool isolation; output filters; cross-agent cross-checks; adversarial training [134]; HITL holds	Prompt injection; tool abuse; data exfiltration; cross-modal injection; lateral movement; jailbreaks	*Trustworthiness*: robustness/adversarial resilience; *Coordination*: agent-of-agent validation; *Composite*: residual risk index; incident rates	HITL review gates [135]; enterprise dashboards (override/stop) [135]; ChemCrow safety pauses [136]	OWASP LLM Top-10 [137, 138]; OECD Robustness [127]

Pillar	Control	Operationalization (How)	Evidence / Artifacts	Standards & Articles
Governance	Org. accountability	Define owners for models/agents/tools; RACI for changes; approvals for high-impact actions	Governance policy; RACI records; change approvals	NIST AI RMF (Govern) [171]; ISO/IEC 42001 [2]
Governance	Policy-as-code	Encode allow/deny rules in orchestrator; pre-exec checks and human sign-off gates	Policy repository; policy test cases; gate logs	EU AI Act Arts. 11–14 [125]; ISO/IEC 42001 [2]
Risk Mgmt	Risk register & AIA	Hazard/threat enumeration; likelihood/impact scoring; AI Impact Assessment (AIA)	Risk register; AIA reports; mitigation plan	ISO/IEC 23894; ISO/IEC 42005 [129]; EU AI Act Art. 9 [125]
Risk Mgmt	Post-market monitoring	Drift/jailbreak telemetry; incident thresholds; rollback playbooks	PMM plan; incident tickets; root-cause analysis	EU AI Act Arts. 72–73 [125]; NIST AI RMF (Manage) [171]
Data Gov	Data minimization & lineage	Task/tenant-scoped memory; TTLs/redaction; dataset lineage and licenses	Data inventory; lineage graphs; retention logs	EU GDPR Art. 25 [34]; ISO/IEC 5259 series

Minha leitura geral é que o artigo tenta trazer para o universo de *Agentic AI* práticas já consolidadas em cybersecurity (tipo defense-in-depth, RBAC, monitoring, etc.), mas adaptadas a um contexto muito mais dinâmico e autônomo. A principal mensagem é que segurança e confiança precisam ser incorporadas desde a arquitetura, e não como “patches” depois. O sistema precisa ser projetado para se defender, auditar e se ajustar continuamente.

Pillar	Control	Operationalization (How)	Evidence / Artifacts	Standards & Articles
Data Gov	Quality & bias checks	Coverage/balance checks; label noise audits; sampling documentation	Data quality reports; bias audit results	EU AI Act Art. 10 [125]; ISO/IEC TR 24027
Explainability	Faithfulness/stability	Log rationales; test explanation faithfulness/stability; user-facing summaries	XAI eval report; model/-data/agent cards	NISTIR 8312 [155]; NIST AI RMF (Map/Measure) [171]
Documentation	Technical documentation	Annex-IV tech docs: purpose, data, performance, oversight, foreseeable misuse	Annex-IV dossier; limitations & misuse notes	EU AI Act Arts. 11–12; Annex IV [125]
Traceability	Provenance & logging	Log prompts, plans, tool calls, I/O, timestamps, agent role; sign releases	Immutable logs (WORM/tamper-evident); signed artifacts	EU AI Act Arts. 12–13 [125]; ISO/IEC 42001 [2]
Human Oversight	Human-in-the-loop gates	Require human approval for sensitive plans (delete, external write, PII)	Approval records; override/rollback logs	EU AI Act Art. 14 [125]; NIST AI RMF (Manage) [171]
Robustness	Stress/red-team testing	Adversarial prompts; perturbation/fault tests; safety regression suites	Red-team report; robustness curves; residual risk	EU AI Act Art. 15 [125]; ISO/IEC 24029-1 [32]
Security	I/O mediation	Prompt sanitization; output filtering; sensitive-data (PII/PHI/secrets) detection	Filter policies; violation/egress logs	OWASP LLM Top-10 [137]; HIPAA §164 [35]
Security	Tool access control	RBAC/ABAC per agent/role; per-tool API tokens; argument validation; allowlists	Access matrices; token scopes; arg-validator tests	NIST SP 800-218/218A; NIST AI RMF (Manage)
Security	Isolation/sandboxing	Containers/VMs; syscall/network/file egress policies; hardware TEEs for sensitive code	Sandbox configs; egress policy; attestation logs	NIST SP 800-218A; TEEs best-practice notes
Privacy	DP budgeting & consent	Differential privacy budgets; consent bases; data-subject rights flows	DP budget ledger; DSR tickets; consent records	GDPR Art. 25 [34]; OECD principles [127]
Supply Chain	Provenance & SBOM	Track sources and hashes for models/tools/datasets; dependency scanning	SBOM; third-party attestations; license checks	NIST SSDF (SP 800-218/218A); ISO/IEC 42001
Coordination	Role separation & least privilege	Distinct agent roles (planner, coder, reviewer); minimal cross-role permissions	Role definitions; privilege matrices; audit samples	NIST AI RMF (Govern) [171]; ISO/IEC 42001
Coordination	Plan consistency checks	Shared plan/belief verification; conflict/deadlock detection; CSS metric	Plan snapshots; CSS scores; conflict logs	Multi-agent eval practice; robustness [32]
Monitoring	Anomaly & exfil detection	Detect unusual tool chains, volume spikes, strange domains; rate-limit/kill switch	Anomaly dashboards; blocked-egress records	OWASP LLM Top-10 [137]; NIST AI RMF (Manage)
Transparency	User notices & recourse	User-facing disclosures, capabilities/limits, and appeal/feedback channels	Notices; feedback logs; SLA for appeals	OECD principles [127]; EU AI Act transparency
Evaluation	Fairness & calibration	Group metrics; ECE/Brier; CI reporting; risk–benefit trade-off docs	Fairness report; calibration plots; CI tables	NIST AI RMF (Measure) [171]; ISO/IEC TR 24027
Records	Release/rollback discipline	Versioned releases; rollback points; emergency disable/recall	Release notes; rollback attestations	EU AI Act PMM [125]; ISO/IEC 42001
GPAI	GPAI obligations	Provide model cards, training-data summary, eval reports, and usage restrictions; disclose known limitations	GPAI documentation pack; model card; eval report	EU AI Act (GPAI duties) [125]; ISO/IEC 42001
Transparency	Synthetic content disclosure	Label AI-generated/edited media; attach provenance (e.g., C2PA) and user notices	Watermarking/provenance config; disclosure logs	EU AI Act Art. 52 [125]; C2PA[7]
Privacy	Cross-border transfers	SCCs/TIAs for third-country transfers; data-residency controls; key management	SCC/TIA records; DPA; data-map; key-custody logs	GDPR Ch. V [34]
Privacy	Data-subject rights ops	Verified workflows for access, erasure, rectification, restriction, portability, objection	DSR tickets/SLAs; fulfillment proofs; audit samples	GDPR Arts. 12–23 [34]
Operations	Change mgmt & release gates	Risk-based pre-release checklist; dual-control approvals; staged rollouts; rollback criteria	Change tickets; gate results; approvals; rollback points	ISO/IEC 42001; NIST AI RMF (Manage) [171]
Supply Chain	Third-party/vendor risk	Vendor questionnaires; DPAs; penetration/attestation reports; license compliance	Vendor risk register; DPA file; attestation/S-BOM	NIST SSDF (SP 800-218/218A); ISO/IEC 42001

Pillar	Control	Operationalization (How)	Evidence / Artifacts	Standards & Articles
Human Factors	HCD & accessibility	Human-centred design reviews; usability risk analysis; accessibility conformance checks	HCD review notes; usability test reports; a11y checklists	ISO 9241-210[8]; OECD principles [127]
Sustainability	Energy/CO_2 telemetry	Track/normalize energy use and CO_2; report per release; set budgets	Energy/CO_2 logs; budget vs. actual; disclosure note	Org policy / reporting guidance

APÊNDICE 7

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("Gate") de aprovação: 13 de nov. de 2025

Participantes da Entrega [matriculados em Residência em IA]:

BERNARDO AIRES DE OLIVEIRA

Entrega: [descrever a ENTREGA - requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Tema: **Segurança para Agentes Inteligentes**

Ao longo das últimas Semanas, meus estudos evoluíram de uma etapa conceitual para uma abordagem prática e aplicada. Inicialmente, foram realizados o levantamento de artigos e materiais técnicos, a criação de um glossário de Segurança com os principais termos da área e, posteriormente, o estudo aprofundado da modelagem de ameaças (Threat Modeling) em arquiteturas agentivas.

Na sequência, foi identificado e aplicado o MAESTRO Framework, um modelo específico para análise de riscos em sistemas de agentes inteligentes. A partir dessa aplicação, foram propostas melhorias estruturais para torná-lo mais operacional (incluindo separação entre metodologia e biblioteca de ameaças, definição de papéis por camada e mitigações prescritivas).

Durante esta décima Semana, o foco esteve na finalização da versão 1 (v1) da Lista de Ameaças - MAESTRO Framework, documento que reúne e organiza ameaças e respectivas estratégias de mitigação distribuídas ao longo das sete camadas do framework.
A lista foi construída com base em referências especializadas (como MITRE ATLAS, Cloud Security Alliance, OWASP e artigos encontrados nas semanas anteriores), integrando práticas reconhecidas de segurança para modelos fundacionais, operações de dados, frameworks de agentes, infraestrutura, observabilidade, conformidade e ecossistema agentivo.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

Lista de Ameaças - MAESTRO Framework.docx citado no Termo de Aceite de Entrega de 13 de novembro

Lista de Ameaças
MAESTRO Framework

Foundation Models

- Search Open AI Vulnerability Analysis: Buscar artigos e PoCs de ataques contra o tipo de modelo que você usa, para reaproveitar técnicas.
    - Mitigação: fazer security review contínua de literatura, acompanhar advisories, atualizar modelos/guardrails conforme vulnerabilidades conhecidas.
- Discover LLM Hallucinations: Descobrir entidades/URLs “inventadas” que o LLM alucina para depois transformá-las em vetores de ataque reais.
    - Mitigação: validação de links/entidades antes de expor a usuários, filtro de saída.
- Discover AI Model Outputs: Explorar saídas do modelo para entender padrões, limites e possíveis caminhos de exploração.
    - Mitigação: limitar detalhes sensíveis nas respostas, randomizar/regularizar outputs quando necessário, monitorar consultas suspeitas.
- Discover LLM System Information: Extrair informações sobre prompts de sistema, guardrails, configurações e provedores para refinar ataques.
    - Mitigação: evitar eco de detalhes internos, masking de system prompt, filtros anti-enumeração.
- Proxy AI Model: Treinar um modelo proxy (modelo substituto criado pelo atacante para imitar o comportamento do modelo alvo, mesmo sem ter acesso direto ao modelo real) semelhante ao alvo para testar e refinar ataques. O treinamento pode ser feito via artefatos (datasets, arquiteturas de modelo) obtidos em etapas de descoberta do ataque, replicação do modelo através de diversas chamadas à API ou modelos pré-treinados semelhantes ao modelo vítima.
    - Mitigação: limitar o vazamento de detalhes de modelo/treino nas respostas, reduzir a possibilidade de extração precisa.
- Poison Training Data: Introduzir dados maliciosos no dataset de treino para enviesar ou degradar o modelo.
    - Mitigação: curadoria de dados, validação de origem, deteção de outliers e drift, versionamento e proveniência.

- Manipulate AI Model: Alterar pesos do modelo, arquitetura ou estado do modelo, além de interferir no treinamento através de dados “envenenados” para mudar seu comportamento.
    - Mitigação: controle de mudanças em modelos, sanitização dos dados de treinamento, versionamento do modelo e dados de treinamento, forte controle de acesso aos modelos em produção.
- Erode AI Model Integrity: Degradar o modelo de forma gradual utilizando do aprendizado contínuo malicioso e updates enviesados (atualizações de pesos que levam ao interesse do atacante).
    - Mitigação: sanitização/filtragem de inputs, monitoramento de métricas de qualidade, controle rígido de treinamento contínuo, uso de métodos de ensemble para evitar dependência em um único modelo.
- Craft Adversarial Data: Descrição: Utilizar exemplos adversariais/backdoors para enganar ou controlar o modelo.
    - Mitigação: adversarial training, detecção de inputs fora de distribuição, Passive AI Output Obfuscation (ofuscar/alterar levemente o output do agente, visando dificultar a extração de informações do modelo).
- Prompt Injection: Inserir instruções maliciosas no prompt (direta ou indiretamente) para desviar o comportamento do agente.
    - Mitigação: guardrails, separação rígida entre instruções e dados, validação de contexto RAG, content filters.
- Model Jailbreak: Burlar restrições de segurança do modelo via prompts criativos.
    - Mitigação: filtros de entrada/saída, prompts de sistema robustos, red teaming contínuo.
- Extract System Prompt: Forçar o modelo a revelar o prompt de sistema ou instruções internas.
    - Mitigação: impedir eco de system prompt, pós-processamento de saída.
- Poisoned Models: O atacante pode publicar modelos treinados com backdoors ou comportamento malicioso em hubs/repositórios, a fim de que sejam utilizados pelo sistema vítima.
    - Mitigação: validação de modelos de terceiros, testes de robustez/backdoor e manutenção do AI Bill of Materials (AI BOM) atualizado.
- Publish Hallucinated Entities: Criar entidades (sites, endereços de email, pacotes de software) falsas na “vida real” para coincidir com alucinações do modelo e explorar usuários.
    - Mitigação: checagem de identidade de entidades críticas, educação de usuários.
- Trusted Output Components Manipulation: Manipular citações, fontes, metadados e “marcadores de confiança” nas respostas do modelo.
    - Mitigação: verificação de links/citações.

- Prompt Obfuscation: Esconder intenção maliciosa no prompt (encoding, esteganografia) para driblar filtros.
    - Mitigação: normalização de entrada, decodificação/expansão antes de avaliar, modelos detectores de padrões suspeitos.
- Evade AI Model: Criar inputs que evitam detecção por modelos de segurança (anti-spam, anti-malware, fraude).
    - Mitigação: ensemble de detectores, adversarial training, revisão humana em casos críticos.
- Membership Inference Attacks: Tentam determinar se um dado específico fez parte do treinamento do modelo, expondo informações sensíveis ou privadas sobre indivíduos.
    - Mitigação: regularização, DP (Differential Privacy), anonimização de dados de treinamento.
- Denial of Service (DoS): Ataques que exploram o custo computacional do modelo (como sponge attacks) para degradar desempenho, aumentar latência ou causar indisponibilidade via inputs especialmente pesados.
    - Mitigação: rate-limit, budget de tokens, timeouts, detectores de queries anômalas, limitação de complexidade.

Data Operations

- Gather RAG-Indexed Targets: Identificação das fontes externas usadas pelo pipeline RAG (bancos vetoriais, bases públicas, APIs, documentos). O atacante mapeia esses alvos para executar poisoning, manipulação de conteúdo ou inserir dados que serão recuperados pelo modelo.
    - Mitigação: ocultar metadados sensíveis, controle de versão/proveniência, sanitização de logs e respostas sobre fontes.
- Poison Training Data: Introduzir dados maliciosos no dataset de treino para enviesar ou degradar o modelo.
    - Mitigação: curadoria de dados, validação de origem, deteção de outliers e drift, versionamento e proveniência.
- Poisoned Datasets: O atacante pode publicar datasets maliciosos na web para que sejam consumidos por terceiros confiando na fonte.
    - Mitigação: due diligence em fontes externas, checagem de integridade antes de incorporar dados às bases próprias e manutenção do AI Bill of Materials (AI BOM) atualizado.
- Spamming AI System with Chaff Data: Inundar o sistema com dados/requisições inúteis para sobrecarregar pipelines ou armazenamento.

    - Mitigação: rate-limit de ingestão, filtros de qualidade de dados, quotas por origem/usuário.
- Data Leakage: Extrair dados sensíveis memorizados ou acessíveis via prompts.
    - Mitigação: minimização de dados sensíveis no treino, prevenção de vazamento de dados em prompts/outputs, políticas de retenção/anonimização/pseudonimização.
- Erode Dataset Integrity: Degradar a qualidade do dataset (rótulos errados, ruído direcionado).
    - Mitigação: auditoria de rótulos, checagem de consistência.
- Retrieval Content Crafting: Criar conteúdo especificamente para ser indexado em RAG e manipular respostas.
    - Mitigação: validação de fonte, scoring de confiança, filtros de conteúdo e reputação de documentos.
- RAG Poisoning: Envenenar diretamente o conteúdo indexado pelo RAG.
    - Mitigação: revisão de ingestão, controle de quem pode adicionar/alterar documentos, validação de integridade.
- False RAG Entry Injection: Inserir entradas falsas no índice (documentos inexistentes, entidades inventadas).
    - Mitigação: validação de fontes, checagem de existência/identidade, logs de mutação no índice.
- AI Artifact Collection: Coleta de artefatos de IA (modelos, datasets, configurações, código-fonte e credenciais) realizada pelo atacante após obter acesso inicial ao sistema.
    - Mitigação: controle de acesso, logs de download, criptografia e segmentação de artefatos.
- Embedding Poisoning: Poluição do espaço vetorial via embeddings contaminados, gerando recuperações distorcidas e respostas incorretas.
    - Mitigação: sanitização de entrada, detecção de anomalias vetoriais, validação de fonte e versionamento de embeddings.

Agent Frameworks

- Search Application Repositories: Buscar repositórios (GitHub, GitLab etc.) ou lojas de aplicativos (Google Play, iOS App store, macOS App Store, Microsoft Store) para identificar códigos com referências a modelos, chaves, endpoints e configs.
    - Mitigação: secret scanning, repositórios privados, revisão de commits, política de branch protection.

- Discover LLM System Information: Extrair informações sobre prompts de sistema, guardrails, configurações e provedores para refinar ataques.
    - Mitigação: evitar eco de detalhes internos, masking de system prompt, filtros anti-enumeração.
- Manipulate AI Model: Alterar pesos do modelo, arquitetura ou estado do modelo, além de interferir no treinamento através de dados “envenenados” para mudar seu comportamento.
    - Mitigação: controle de mudanças em modelos, sanitização dos dados de treinamento, versionamento do modelo e dados de treinamento, forte controle de acesso aos modelos em produção.
- Plugin Compromise: Usar ou comprometer plugins/tools do modelo para executar ações maliciosas (código, dados, APIs).
    - Mitigação: whitelists de ferramentas, RBAC, revisão de plugins, sandbox e limites de escopo.
- Supply Chain Attacks: Comprometer dependências do framework (libs, pacotes, containers, modelos terceiros).
    - Mitigação: SBOM/AI-BOM, verificação de assinatura, reprodutibilidade, scanning.
- Denial of Service on Framework APIs: Sobrecarga de endpoints críticos do framework usados por agentes.
    - Mitigação: rate-limit,monitoramento, redundância.
- Compromised Framework Components: Injeção de código malicioso em bibliotecas internas do framework (core libs, módulos de agente, runtime), causando alteração de comportamento ou execução arbitrária de ações.
    - Mitigação: SBOM/AI-BOM, validação de integridade, assinatura de pacotes, execução isolada.
- Backdoor Attacks: Inserção de funcionalidades ocultas dentro do framework que acionam comportamentos maliciosos quando condições específicas são atendidas.
    - Mitigação: code review, verificação estática e dinâmica, integridade de dependências.
- Input Validation Attacks: Explorar falhas na validação de entrada do framework (ex: agentes que processam JSON, YAML, código ou arquivos), podendo causar execução remota, corrupção de estado ou invasão do sistema.
    - Mitigação: validação de schema, sanitização forte, limites de tamanho/tipos.
- Framework Evasion: Criar agentes ou prompts que intencionalmente driblam controles internos do framework, como limites de ferramenta, políticas internas de chamadas ou restrições de fluxo.
    - Mitigação: testes de evasão, auditoria comportamental, verificadores de conformidade com policies internas, restrições rígidas de tool-use.

Deployment & Infrastructure

- Active Scanning: Varredura ativa da superfície de ataque do ambiente onde agentes e serviços LLM estão implantados, buscando portas abertas, endpoints de inferência, banco vetorial exposto, dashboards, contêineres mal configurados ou serviços auxiliares. O objetivo é mapear oportunidades de exploração em camadas subsequentes (credential access, LLM jailbreak, data exfiltration, etc.).
    - Mitigação: rate-limit, segmentação de rede, bloquear portas desnecessárias, detecção de scanning.
- Search Application Repositories: Buscar repositórios (GitHub, GitLab etc.) ou lojas de aplicativos (Google Play, iOS App store, macOS App Store, Microsoft Store) para identificar códigos com referências a modelos, chaves, endpoints e configs.
    - Mitigação: secret scanning, repositórios privados, revisão de commits, política de branch protection.
- Cloud Service Discovery: Enumerar serviços de cloud (buckets, registries, notebooks, vector DBs) usados pela stack de IA.
    - Mitigação: hardening de IAM, esconder nomes/paths previsíveis, monitorar discovery anômalo.
- Stage Capabilities: Armazenar scripts, modelos de IA e automações na infraestrutura (Cloud, GitHub, Hugging Face ou registros de contêineres) da vítima, visando abrir portas para ataques ao sistema.
    - Mitigação: monitorar integrações externas, validar origem de plugins/modelos, SBOM/AI-BOM.
- Spamming AI System with Chaff Data: Inundar o sistema com dados/requisições inúteis para sobrecarregar pipelines ou armazenamento.
    - Mitigação: rate-limit de ingestão, filtros de qualidade de dados, quotas por origem/usuário.
- Plugin Compromise: Usar ou comprometer plugins/tools do modelo para executar ações maliciosas (código, dados, APIs).
    - Mitigação: whitelists de ferramentas, RBAC, revisão de plugins, sandbox e limites de escopo.
- Corrupt AI Model: Corromper o arquivo de um modelo para passar por filtros ou mascarar malwares.
    - Mitigação: verificação robusta de integridade, scanners tolerantes a arquivos quebrados, backups e rollback.
- User Execution: Convencer alguém a executar artefatos/código relacionado à IA (scripts, modelos, notebooks).

    - Mitigação: educação de usuários/devs, assinatura de artefatos, políticas para execução de código e modelos.
- Exploit Public-Facing Application: Explorar vulnerabilidades em apps que expõem serviços de IA.
    - Mitigação: secure coding, patching, testes de pentest focados em endpoints de agentes.
- Command and Scripting Interpreter: Abusar de interpretadores de código (Python, PowerShell, notebooks) ligados à pipeline de IA.
    - Mitigação: restringir execução remota, hardening de notebooks.
- Unsecured Credentials: Encontrar segredos expostos (tokens, chaves API) em configs, logs ou repositórios.
    - Mitigação: secret managers, scanning de segredos, criptografia e mudança de credenciais ao longo do tempo.
- Reverse Shell: O atacante intercepta a conexão com servidores que executam pipelines/modelos/agentes e redireciona as requisições para onde desejar.
    - Mitigação: firewall de saída, desabilitar execução remota, segmentação de rede.
- AI Artifact Collection: Coleta de artefatos de IA (modelos, datasets, configurações, código-fonte e credenciais) realizada pelo atacante após obter acesso inicial ao sistema.
    - Mitigação: controle de acesso, logs de download, criptografia e segmentação de artefatos.
- Exfiltration via Cyber Means: Tirar dados/artefatos via canais de rede tradicionais (C2, tunelamento, storage externo).
    - Mitigação: inspeção de tráfego, firewall de saída, data loss prevention (DLP) na rede.
- Denial of AI Service: Derrubar ou degradar serviço de IA com requisições excessivas ou pesadas.
    - Mitigação: autoscaling, rate-limit, quotas, priorização de tráfego crítico.

Evaluation & Observability

- Active Scanning: Varredura de endpoints de métricas, dashboards de observabilidade, health-checks e sistemas de avaliação. Adversários podem identificar portas expostas (Prometheus, Grafana, MLflow), coletar informações sensíveis ou mapear o comportamento do modelo.
    - Mitigação: rate-limit, segmentação de rede, bloquear portas desnecessárias, detecção de scanning.

- Discover AI Model Outputs: Explorar saídas do modelo para entender padrões, limites e possíveis caminhos de exploração.
    - Mitigação: limitar detalhes sensíveis nas respostas, randomizar/regularizar outputs quando necessário, monitorar consultas suspeitas.
- Erode AI Model Integrity: Degradar o modelo de forma gradual utilizando do aprendizado contínuo malicioso e updates enviesados (atualizações de pesos que levam ao interesse do atacante).
    - Mitigação: sanitização/filtragem de inputs, monitoramento de métricas de qualidade, controle rígido de treinamento contínuo, uso de métodos de ensemble para evitar dependência em um único modelo.
- Trusted Output Components Manipulation: Manipular citações, fontes, metadados e “marcadores de confiança” nas respostas do modelo.
    - Mitigação: verificação de links/citações.
- Manipulation of Evaluation Metrics: Influenciar benchmarks, datasets ou casos de teste para mascarar falhas ou inflar performance.
    - Mitigação: datasets imutáveis, auditoria, versionamento.
- Compromised Observability Tools: Componentes de monitoramento (exporters, dashboards, MLflow plugins) adulterados para esconder comportamentos maliciosos.
    - Mitigação: assinatura de plugins, hardening, verificação de integridade.
- Denial of Service on Evaluation Infrastructure: Atacar sistemas de avaliação/monitoramento para impedir a visibilidade de logs.
    - Mitigação: redundância, rate-limit, priorização de pipelines críticos.
- Poisoning Observability Data: Manipulação de logs, métricas e traces para esconder incidentes ou inflar métricas falsamente.
    - Mitigação: logging imutável, assinaturas digitais, auditoria rígida.
- Data Leakage Through Observability: Exposição acidental de dados sensíveis por dashboards, logs, métricas e sistemas de tracing.
    - Mitigação: sanitização, segregação de logs.

Security & Compliance (Vertical Layer)

- Gather Victim Identity Information: Coleta de informações de identidade (nomes, emails, fotos, credenciais, MFA) para uso em ataques de engenharia social, deepfakes, impersonação, phishing ou criação de contas falsas.
    - Mitigação: higienização de dados públicos, proteção de identidade digital, políticas de privacidade, monitoramento de vazamentos e alerta de tentativas de impersonação.

- Search Open Technical Databases: Busca por papers, posts em redes sociais, blogs e documentos técnicos para descobrir arquiteturas, modelos e usos de IA da organização.
    - Mitigação: reduzir exposição de detalhes sensíveis em publicações, revisão de comunicação técnica e política de divulgação.
- Search Victim-Owned Websites: Vasculhar sites, portais e documentos públicos da empresa para descobrir onde IAs e agentes estão expostos.
    - Mitigação: higienizar conteúdo público, evitar expor endpoints, chaves e detalhes internos de pipeline.
- Acquire Public AI Artifacts: Baixar datasets, modelos e exemplos públicos para treinar proxy models ou preparar ataques mais específicos.
    - Mitigação: acompanhar o uso de modelos/dados públicos na organização, validar licenças, limitar reuso de artefatos sem revisão.
- User Execution: Convencer alguém a executar artefatos/código relacionado à IA (scripts, modelos, notebooks).
    - Mitigação: educação de usuários/devs, assinatura de artefatos, políticas para execução de código e modelos.
- Phishing: Usar engenharia social (inclusive com modelos de IA) para roubar credenciais ou induzir ações perigosas.
    - Mitigação: treinamento de equipe, filtros de email/mensagens, validação de origem de requisições sensíveis.
- Impersonation: Fingir ser agente/usuário legítimo para enganar humanos ou sistemas.
    - Mitigação: autenticação forte de agentes, assinaturas digitais, UX que mostre identidade verificável.
- Regulatory Non-Compliance: Agentes operando fora de conformidade com leis (LGPD, HIPAA, GDPR etc.).
    - Mitigação: auditoria legal, documentação robusta, governança de modelos.
- Bias: Vieses levando a decisões injustas ou falhas de proteção seletivas.
    - Mitigação: retraining supervisionado, avaliação contínua.
- Lack of Explainability: Ausência de transparência nas decisões, prejudicando auditoria e accountability.
    - Mitigação: XAI, logging explicável, documentação do raciocínio do agente.

Agent Ecosystem

- Active Scanning: Adversários realizam scan em interfaces onde agentes são publicados (registries, APIs públicas, marketplaces) para identificar agentes vulneráveis, endpoints com capacidades expostas ou descrições enganosas. O

objetivo é mapear alvos para impersonation, manipulação de reputação ou goal hijacking.
    - Mitigação: rate-limit, segmentação de rede, bloquear portas desnecessárias, detecção de scanning.
- Establish Accounts: Criação de múltiplas contas legítimas (dev, trial, SaaS) para atacar modelos e agentes com aparência de “usuário normal”.
    - Mitigação: monitorar padrões de uso, limitar privilégios de contas novas.
- Prompt Injection: Inserir instruções maliciosas no prompt (direta ou indiretamente) para desviar o comportamento do agente.
    - Mitigação: guardrails, separação rígida entre instruções e dados, validação de contexto RAG, content filters.
- Plugin Compromise: Usar ou comprometer plugins/tools do modelo para executar ações maliciosas (código, dados, APIs).
    - Mitigação: whitelists de ferramentas, RBAC, revisão de plugins, sandbox e limites de escopo.
- Model Jailbreak: Burlar restrições de segurança do modelo via prompts criativos.
    - Mitigação: filtros de entrada/saída, prompts de sistema robustos, red teaming contínuo.
- Extract System Prompt: Forçar o modelo a revelar o prompt de sistema ou instruções internas.
    - Mitigação: impedir eco de system prompt, pós-processamento de saída.
- Publish Hallucinated Entities: Criar entidades (sites, endereços de email, pacotes de software) falsas na “vida real” para coincidir com alucinações do modelo e explorar usuários.
    - Mitigação: checagem de identidade de entidades críticas, educação de usuários.
- Prompt Self-Replication: Prompts que se auto-replicam e se propagam entre agentes ou contextos.
    - Mitigação: sanitização de inputs antes de encaminhar, limites de profundidade/encadeamento, filtros anti-“prompt worms”.
- Prompt Obfuscation: Esconder intenção maliciosa no prompt (encoding, esteganografia) para driblar filtros.
    - Mitigação: normalização de entrada, decodificação/expansão antes de avaliar, modelos detectores de padrões suspeitos.
- Phishing: Usar engenharia social (inclusive com modelos de IA) para roubar credenciais ou induzir ações perigosas.
    - Mitigação: treinamento de equipe, filtros de email/mensagens, validação de origem de requisições sensíveis.
- Impersonation: Fingir ser agente/usuário legítimo para enganar humanos ou sistemas.

    - Mitigação: autenticação forte de agentes, assinaturas digitais, UX que mostre identidade verificável.
- Drive-by Compromise: Comprometer usuários/sistemas através de visitas em um conteúdo web malicioso. Por exemplo, quando um agente busca informações em um site e é infectado por um Prompt Injection.
    - Mitigação: hardening de browsers corporativos, restrição de acesso a painéis internos, permissões limitadas aos agentes para acesso à fonte externas.
- Exfiltration via AI Inference API: Usar a própria API de inferência para extrair dados/modelo ou infer informações sensíveis.
    - Mitigação: monitorar padrões de consulta, limitar tipos de perguntas, logging e alertas.
- Compromised Agents: Agentes maliciosos infiltrados se passando por serviços legítimos.
    - Mitigação: assinatura de agentes, auditoria, reputação verificável.
- Agent Impersonation: Fingir ser um agente legítimo no ecossistema.
    - Mitigação: identidade verificável, autenticação forte.
- Agent Identity Attack: Explorar falhas nos mecanismos de identidade/autorização do agente.
    - Mitigação: IAM, rotação periódica de chaves.
- Agent Tool Misuse: Forçar agentes a usar ferramentas de forma indevida.
    - Mitigação: whitelists rigorosas, validação de argumentos.
- Agent Goal Manipulation: Alterar ou desviar os objetivos originais de um agente.
    - Mitigação: monitoramento de alinhamento, políticas imutáveis.
- Marketplace Manipulation: Manipular avaliações e rankings para promover agentes maliciosos.
    - Mitigação: detecção anti-fraude, reputação verificável.
- Integration Risks: Aproveitar vulnerabilidades em APIs/SDKs integradas aos agentes.
    - Mitigação: validação de inputs, versionamento seguro.
- Horizontal/Vertical Solution Vulnerabilities: Explorar fraquezas específicas de agentes de um domínio ou setor.
    - Mitigação: threat modeling por domínio, hardening específico.
- Repudiation: Agentes negam ações realizadas, dificultando auditoria e accountability.
    - Mitigação: logging imutável, assinatura digital.
- Compromised Agent Registry: Manipular o registro de agentes, inserindo agentes maliciosos ou alterando descrições.
    - Mitigação: assinatura e verificação de registros, permissões restritas.
- Malicious Agent Discovery: Influenciar mecanismos de descoberta para esconder agentes legítimos e promover agentes maliciosos.
    - Mitigação: verificação cruzada, auditoria contínua.
- Inaccurate Agent Capability Description: Descrições falsas das capacidades de um agente, levando a mau uso ou risco.

- Mitigação: auditoria de capacidades, validação manual.