Segurança da Informação aplicada a IA, protegendo LLMs com RAG no ambiente corporativo

Bruno Hermeto Guimarães[1], **Jõao Comini César de Andrade**[1]

[1]Instituto de Ciências Exatas e Informática – Pontifícia Universidade Católica de Minas Gerais (PUCMG) Caixa Postal 1.686 – 30535-901 – Belo Horizonte – MG – Brazil

{bhguimaraes,jccandrade}@sga.pucminas.br

Abstract.

Resumo.

1. Introdução

Embora a arquitetura de Geração Aumentada por Recuperação (RAG, do inglês *Retrieval-Augmented Generation*) seja usada [Microsoft 2024] para tornar os Modelos de Linguagem de Grande Escala (LLMs, do inglês *Large Language Models*) mais seguros no ambiente corporativo, ela não é totalmente eficaz, podendo se tornar um novo alvo para ataques furtivos e eficazes como *prompt injection* e o envenenamento de documentos. Quando bem-sucedidos, esses ataques resultam em graves consequências como vazamento de dados e a geração de informações falsas [Naseh et al. 2025, Gong et al. 2025]. Nesse contexto, o público-alvo para compreender e neutralizar essas ameaças é as lideranças executivas, juntamente com as equipes de TI e segurança da informação, que são as responsáveis por implementar e proteger esses sistemas. Dessa forma, uma implementação segura pode beneficiar não só a própria empresa, que protege sua reputação e seus dados, mas também seus usuários que terão mais confiança na proteção de seus dados e nas informações geradas [Zhang et al. 2024].

A adoção de uma estratégia de defesa em profundidade minimiza os riscos de segurança associados ao uso de LLMs com RAG em ambientes corporativos [Kande et al. 2024, Zhang et al. 2024]. Essa estratégia abrange a aplicação de diversas camadas de segurança como: políticas transparentes, filtragem de documentos, validação das respostas produzidas e isolamento das ferramentas que operam em conjunto para reduzir a superfície de ataque. A existência de ataques como a injeção de prompt, que exploram falhas em diversas fases do pipeline para fins como a exfiltração de dados, ressalta a necessidade de uma arquitetura unificada [Greshake et al. 2023].

Além disso, a maioria dos ataques ocorre quando o modelo é exposto a comandos maliciosos incorporados em documentos recuperados, o que torna essencial o uso de filtros semânticos e classificadores de risco [Greshake et al. 2023]. A validação da saída permite detectar padrões suspeitos antes de mostrar ou executar o conteúdo, enquanto o uso de sandboxes garante o controle sobre ações externas. Estratégias como essas já estão sendo testadas em situações reais por empresas como Microsoft e OpenAI, com resultados promissores [Microsoft 2024]. Com base nesses indícios, nossa proposta apresenta-se como uma solução eficaz, mensurável e viável para aprimorar a segurança sem comprometer a funcionalidade e o desempenho do sistema.

Várias abordagens concorrentes têm sido empregadas na tentativa de reduzir os riscos de segurança em LLMs, porém apresentam limitações significativas, especialmente em tarefas delicadas que exigem validação e supervisão rigorosa do conteúdo gerado [Kande et al. 2024]. As defesas de segurança genéricas, conhecidas como *guardrails*, que se baseiam em regras simples como palavras-chave, mostram-se ineficazes para identificar ataques complexos, como a ofuscação de comandos ou a injeção de prompt indireta [Greshake et al. 2023]. Por outro lado, os detectores de *jailbreak* dependem de padrões conhecidos e podem ser facilmente contornados por variações sutis. Outro ponto é que os *prompts* de sistema longos, que buscam estabelecer regras para o modelo, podem ser suprimidos por comandos maliciosos inseridos no contexto recuperado, principalmente em pipelines RAG. Quando o modelo lida com dados sensíveis e ferramentas externas, essas soluções isoladas não oferecem garantias suficientes.

Em ambientes empresariais que exigem confidencialidade, rastreabilidade e conformidade, é imprescindível um controle sólido. Por isso, é proposto uma arquitetura que combina diferentes mecanismos de defesa, atuando em conjunto para bloquear ataques antes, durante e depois da geração da resposta. Essa abordagem integrada se mostra mais eficaz e adaptável às complexidades reais do uso de IA em sistemas empresariais, superando as limitações das soluções atuais.

O principal problema a ser resolvido é a questão de segurança em modelos que utilizam a arquitetura RAG, e para resolve-lo é necessário desenvolver e implementar novos mecanismos de defesa para detectar e neutralizar ataques, assim garantindo mais segurança e confiabilidade para garantir a proteção dos dados e do sistema.

2. Referencial Teórico

2.1. Modelo de Linguagem de Grande Escala (LLM)

Os Modelos de Linguagem de Grande Escala (LLMs, do inglês *Large Language Models*) são uma classe de modelos de redes neurais profundas. Seu processo de treinamento se baseia no aprendizado de máquina não supervisionado, no qual são utilizadas grandes quantidades de textos não rotulados [NVIDIA 2025]. Essa abordagem capacita o modelo a entender, gerar e interagir com a linguagem humana. Esses modelos de linguagem são a tecnologia por trás de ferramentas de IA generativa, como o ChatGPT da OpenAI e o Gemini do Google.

2.2. Geração Aumentada por Recuperação (RAG)

A Geração Aumentada por Recuperação (RAG, do inglês *Retrieval-Augmented Generation*) é uma arquitetura projetada para tornar os LLMs mais precisos, poderosos e confiáveis [Lewis et al. 2020]. Essa abordagem permite que os LLMs recuperem informações de uma base de dados externa antes de formularem uma resposta a uma consulta do usuário, essa base pode conter diversas fontes, como artigos, documentos internos, conteúdos de sites, entre outros.

O seu funcionamento ocorre em duas etapas principais, primeiramente, um componente de recuperação (*retriever*) busca informações relevantes para a consulta do usuário nessa base de dados externa, em seguida, essas informações são fornecidas como contexto adicional ao LLM (*generator*), que as utiliza para formular uma resposta mais completas, precisa e contextualizada.

2.3. Ataque em Sistemas RAG

A arquitetura RAG, embora segura, possui brechas para diversos tipos de ataque a sistemas LLMs, alguns exemplos deles são o envenenamento de documentos (*Poisoned RAG*), que ataca a integridade das informações, e o ataque de inferência de pertinência (MIA, do inglês *Membership Inference Attack*), que visamo vazamento de dados.

O envenenamento de documentos busca corromper a base de dados do RAG, injetando informações falsas para manipular as respostas geradas para os usuários. Os artigos "Topic-FlipRAG: Topic-Orientated Adversarial Opinion Manipulation Attacks to Retrieval-Augmented Generation Models" e "PoisonedRAG: Knowledge Corruption Attacks to Retrieval-Augmented Generation of Large Language Models" estudaram profundamente esse tipo de ataque, onde cada um dos artigos estudou uma estratégia diferente, a primeira delas é modificar sutilmente documentos já existentes na base de dados e a segunda é criar um documento a partir do zero, muito semelhante a documentos de fontes confiáveis, e o inserir nessa base, com essas estratégias os ataques se tornam eficazes e furtivos por não alertarem nenhum sistema de segurança podendo assim concluir o seu objetivo [Gong et al. 2025, Zou et al. 2025].

Outro ataque, focado na segurança de dados, é o de inferência de pertinência, ele foi estudado pelo artigo "Riddle Me This! Stealthy Membership Inference for Retrieval-Augmented Generation", ele é um ataque que tem como objetivo extrair informações sigilosas presentes na base de dados do RAG, ele funciona com o atacante utilizando perguntas cuidadosamente elaboradas em linguagem que usuários comuns utilizam, no qual a resposta só pode ser encontrada no documento alvo, após conseguir o acesso ao documento desejado, as informações são roubadas, esse ataque também é considerado eficaz e furtivo [Naseh et al. 2025].

2.4. Estratégias de Defesa para Sistemas RAG

Diante da variedade e complexidade dos ataques aos sistemas RAG, a literatura de segurança indica que soluções isoladas não são adequadas. A estratégia mais promissora, alinhada à "defesa em profundidade", envolve a aplicação de diversas camadas de segurança que trabalham em conjunto para preservar todo o ciclo de vida do processamento da informação, desde a análise proativa de vulnerabilidades até o acompanhamento das ações do sistema. Uma das primeiras camadas de defesa é a proativa, que busca identificar e corrigir falhas de segurança antes que possam ser exploradas. Nesse contexto, o artigo "Detecting command injection vulnerabilities in Linux-based embedded firmware with LLM-based taint analysis of library functions" introduz a ferramenta SLFHunter, que utiliza os próprios LLMs para detectar vulnerabilidades de injeção de comandos. A pesquisa demonstra que os modelos de linguagem podem atuar como aliados na segurança, automatizando a identificação de falhas críticas com alta precisão e ampliando a cobertura da análise para além das capacidades das ferramentas convencionais [Ye et al. 2024]. Além das defesas proativas, é essencial implementar mecanismos de controle durante e após a geração da resposta pelo LLM, principalmente quando o sistema interage com ferramentas externas e manipula dados sensíveis. O artigo "PrivacyAsst: Safeguarding User Privacy in Tool-Using Large Language Model Agents" aborda diretamente este desafio com a apresentação do PrivacyAsst, um sistema projetado para preservar a privacidade do usuário. O PrivacyAsst monitora as interações entre o modelo

e os serviços conectados, detectando e bloqueando ativamente possíveis vazamentos de dados confidenciais com impacto mínimo no desempenho. Este estudo evidencia a viabilidade e a necessidade de mecanismos de defesa proativos que controlem as ações do LLM em tempo real [Zhang et al. 2024]. Adicionalmente, a validação do conteúdo gerado pelo LLM constitui uma camada de segurança indispensável. O estudo "(Security) Assertions by Large Language Models" reforça essa necessidade ao apontar para as limitações dos modelos em tarefas delicadas, enfatizando que o conteúdo produzido por eles requer validação e supervisão rigorosas para ser considerado confiável. Essa conclusão é diretamente aplicável aos sistemas RAG corporativos, justificando a implementação de uma camada de validação que verifique a exatidão e a segurança das respostas antes que sejam entregues ao usuário final [Kande et al. 2024]. Portanto, os estudos analisados caminham para a mesma conclusão: uma defesa eficaz para sistemas RAG não pode depender de uma única ferramenta. A solução consiste, pelo contrário, em uma arquitetura integrada que atue em múltiplas frentes, unificando a detecção proativa de vulnerabilidades, o monitoramento contínuo das ações do modelo e a validação rigorosa de suas saídas.

2.5. Problemas em Aberto e Desafios Futuros

Embora a estratégia de defesa em profundidade aponte o caminho correto, as soluções apresentadas na literatura ainda não resolvem completamente o problema da segurança em sistemas RAG, deixando lacunas importantes que justificam a necessidade de novas pesquisas. Primeiramente, as defesas discutidas, embora eficazes em seus domínios específicos, operam como soluções pontuais e não abrangentes. A criação de uma arquitetura verdadeiramente integrada, que unifique ferramentas de análise proativa como o SLFHunter, monitores de tempo de execução como o PrivacyAsst e camadas de validação de saída, é um desafio de engenharia e pesquisa ainda em aberto. A ausência de um framework unificado dificulta a implementação, o gerenciamento e a manipulação dessas múltiplas camadas de segurança em ambientes corporativos complexos. Em segundo lugar, existe um desafio de escalabilidade e automação na validação das respostas. Como aponta o estudo sobre a geração de assertivas de segurança, a supervisão do conteúdo gerado por LLMs é indispensável para garantir sua confiabilidade [Kande et al. 2024]. No entanto, a dependência de validação humana ou de regras muito estritas não é escalável para sistemas que precisam gerar milhares de respostas por dia. A criação de mecanismos de validação automatizados que sejam ao mesmo tempo robustos, flexíveis e confiáveis continua a ser um problema não resolvido. Finalmente, a maioria das defesas atuais luta para acompanhar a sofisticação dos ataques semânticos. Ferramentas como o PrivacyAsst são excelentes para detectar vazamentos de dados com base em padrões conhecidos, mas podem ser contornadas por ataques que ofuscam a intenção maliciosa através de linguagem natural complexa ou instruções disfarçadas, como os detalhados nos estudos sobre envenenamento de documentos [Gong et al. 2025]. A capacidade de uma defesa de compreender o contexto e a intenção por trás de um prompt, em vez de apenas analisar padrões, define os próximos desafios no âmbito de defesa de sistemas baseados em linguagem. Portanto, a criação de uma arquitetura de segurança que seja ao mesmo tempo integrada, escalável na sua capacidade de validação e segura contra ataques semânticos avançados continua a ser o principal desafio na área, servindo como motivação para o desenvolvimento de novas propostas e continuação de estudos na área.

References

Gong, Y., Chen, Z., Chen, M., Yu, F., Lu, W., Wang, X., Liu, X., and Liu, J. (2025). Topic-fliprag: Topic-orientated adversarial opinion manipulation attacks to retrieval-augmented generation models. In *34th USENIX Security Symposium (USENIX Security 25)*. USENIX Association.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. (2023). Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection.

Kande, R., Pearce, H., Tan, B., Dolan-Gavitt, B., Thakur, S., Karri, R., and Rajendran, J. (2024). (security) assertions by large language models. *IEEE Transactions on Information Forensics and Security*, 19:4374–4389.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS 2020 (Conference on Neural Information Processing Systems)*.

Microsoft (2024). Llms e o openai do azure no padrão rag (geração aumentada de recuperação). `https://learn.microsoft.com/pt-br/industry/sovereignty/architecture/AIwithLLM/ra-sovereign-ai-llm-configurations`.

Naseh, A., Peng, Y., Suri, A., Chaudhari, H., Oprea, A., and Houmansadr, A. (2025). Riddle me this! stealthy membership inference for retrieval-augmented generation. In *ACM Conference on Computer and Communications Security (CCS)*. ACM (Association for Computing Machinery).

NVIDIA (2025). Large language models explained.

Ye, J., Fei, X., de Carné de Carnavalet, X., Zhao, L., Wu, L., and Zhang, M. (2024). Detecting command injection vulnerabilities in linux-based embedded firmware with llm-based taint analysis of library functions. *Computers & Security*, 144.

Zhang, X., Xu, H., Ba, Z., Wang, Z., Hong, Y., Liu, J., Qin, Z., and Ren, K. (2024). Privacyasst: Safeguarding user privacy in tool-using large language model agents. *IEEE Transactions on Dependable and Secure Computing*, 21:5242–5258.

Zou, W., Geng, R., Wang, B., and Jia, J. (2025). Poisonedrag: Knowledge corruption attacks to retrieval-augmented generation of large language models. In *34th USENIX Security Symposium (USENIX Security 25)*. USENIX Association.