SAST para detecção de vulnerabilidades em workflows do n8n

João Pedro Schmidt Cordeiro Gustavo Zambonin

21 de novembro de 2025

Resumo

A crescente adoção de plataformas Low-Code/No-Code (LCNC), como o n8n, tem transformado o desenvolvimento de automações empresariais, democratizando a criação de workflows através de interfaces visuais. Entretanto, essa mudança de paradigma introduz uma lacuna crítica de governança: enquanto a plataforma subjacente é protegida por práticas robustas de segurança, os workflows criados pelos usuários não são submetidos ao mesmo rigor de análise de segurança. Este trabalho propõe o desenvolvimento de uma ferramenta de Análise Estática de Segurança de Aplicações (SAST) específica para workflows do n8n, capaz de detectar automaticamente vulnerabilidades como SQL Injection, Command Injection, SSRF, exposição de credenciais, manuseio inseguro de dados e negação de serviço. A metodologia baseia-se em uma estratégia híbrida que combina o Agentic Radar (focado em riscos de agentes de IA) e o Semgrep (configurado com regras customizadas para vulnerabilidades tradicionais). Além disso, propõe-se o uso de inteligência artificial para acelerar a criação de regras de detecção. A análise revelou coberturas complementares de 16,7% e 66,7% respectivamente, demonstrando a viabilidade da análise estática de estruturas JSON declarativas. A solução contribui para elevar o nível de segurança das automações desenvolvidas em organizações brasileiras, auxiliando na conformidade com a LGPD.

Palavras-chave: SAST, n8n, Low-Code/No-Code, Segurança de Aplicações, Análise Estática, Vulnerabilidades, Automação de Workflows, Inteligência Artificial, LGPD.

Sumário

1 Introdução **4**
- 1.1 Motivação e Contexto . . . 4
- 1.2 Problema Abordado . . . 5
- 1.3 Delimitação do Escopo . . . 5
- 1.4 Contribuições Esperadas . . . 6

2 Fundamentação Teórica e Estado da Arte **6**
- 2.1 Técnicas de SAST para Plataformas Low-Code/No-Code . . . 7
  - 2.1.1 Abordagem Moderna: Agentic Radar e IA Agêntica . . . 7
  - 2.1.2 Abordagem Clássica: Semgrep e Análise Baseada em Padrões . . . 7
- 2.2 Explicabilidade em Ferramentas de Análise de Segurança . . . 8
- 2.3 Segurança e Privacidade em Sistemas de Análise Baseados em IA . . . 9

2.4 Gaps e Oportunidades . . . 9
2.4.1 Gap Fundamental: Ausência de Solução Específica para n8n . . . 10
2.4.2 Oportunidade: IA para Geração Assistida de Regras Semgrep . . . 10
2.4.3 Oportunidade: Arquitetura Híbrida para Cobertura Completa . . . 11

3 Metodologia **11**
3.1 Abordagem Metodológica Geral . . . 11
3.2 Análise das Ferramentas Existentes . . . 11
3.2.1 Critérios de Avaliação . . . 11
3.2.2 Processo de Análise . . . 12
3.3 Geração de Regras Semgrep Assistida por IA . . . 12
3.3.1 Preparação de Workflows de Teste . . . 12
3.3.2 Metodologia de Geração de Regras com IA . . . 12
3.3.3 Validação das Regras Geradas . . . 13
3.4 Validação e Métricas . . . 13
3.4.1 Testes de Detecção . . . 13
3.4.2 Avaliação de Cobertura . . . 14

4 Objetivos e Métricas de Sucesso **14**
4.1 Objetivo Geral . . . 14
4.2 Objetivos Específicos . . . 15
4.3 Métricas de Avaliação . . . 15
4.3.1 Métricas Quantitativas . . . 15
4.3.2 Métricas Qualitativas . . . 16
4.4 Benchmarks Comparativos . . . 17
4.5 Critérios de Sucesso . . . 18

5 Escopo da Solução e Requisitos **19**
5.1 Casos de Uso . . . 19
5.1.1 Caso de Uso Principal . . . 19
5.1.2 Casos Excluídos do Escopo . . . 19
5.2 Arquitetura Proposta . . . 19
5.2.1 Componentes Principais . . . 19
5.3 Requisitos Técnicos . . . 20
5.3.1 Tecnologias e Frameworks . . . 20
5.3.2 Infraestrutura Mínima . . . 20
5.3.3 Dados de Treinamento e Validação . . . 20

6 Ameaças, Riscos e Controles **20**
6.1 Superfícies de Ataque . . . 20
6.2 Análise STRIDE . . . 20
6.3 Ataques Específicos a Modelos de ML . . . 21
6.3.1 Evasion Attacks . . . 21
6.3.2 Data Poisoning . . . 21
6.3.3 Model Extraction . . . 21
6.4 Considerações de Privacidade (LINDDUN) . . . 21
6.5 Considerações Éticas . . . 21
6.5.1 Vieses Algorítmicos . . . 21
6.5.2 Transparência e Explicabilidade . . . 21

6.5.3 Direito de Contestação 22

7 Avaliação Experimental **22**
7.1 Protocolo Experimental 22
7.1.1 Configuração de Hardware e Software 22
7.1.2 Repetição e Validação Cruzada 22
7.2 Resultados 22
7.2.1 Performance no Conjunto de Teste 22
7.2.2 Matriz de Confusão 22
7.2.3 Performance por Tipo/Categoria 22
7.2.4 Latência e Throughput 22
7.3 Testes de Robustez 23
7.3.1 Transformações Geométricas e Degradações 23
7.3.2 Ataques Adversariais 23
7.4 Análise de Vieses 23
7.4.1 Equidade entre Subgrupos 23
7.5 Explicabilidade: Análise Qualitativa 23

8 Discussão **23**
8.1 Alcance dos Objetivos 23
8.2 Limitações Identificadas 23
8.2.1 Limitações Técnicas 23
8.2.2 Limitações de Segurança 23
8.3 Comparação com Estado-da-Arte 24
8.4 Trade-offs Segurança vs. Usabilidade 24
8.5 Implicações Práticas 24
8.5.1 Para Usuários/Stakeholders 24
8.5.2 Para Políticas Públicas 24

9 Conclusões e Próximos Passos **24**
9.1 Síntese das Evidências 24
9.2 Contribuições 24
9.3 Trabalhos Futuros 24
9.3.1 Curto Prazo (3-6 meses) 24
9.3.2 Médio Prazo (6-12 meses) 24
9.3.3 Longo Prazo (1-2 anos) 25
9.4 Recomendações 25

10 Checklist de Conformidade (LGPD/Ética/Segurança) **25**
10.1 Lei Geral de Proteção de Dados (LGPD) 25
10.2 Segurança da Informação 25
10.3 Ética e IA Responsável 26
10.4 Ações Pendentes para Produção 26

A Detalhes de Implementação **26**
A.1 Estrutura de Diretórios 26
A.2 Comandos para Reprodução 26

B Exemplos Adicionais de Resultados **26**

1 Introdução

A crescente adoção de plataformas Low-Code/No-Code (LCNC) representa uma transformação fundamental no desenvolvimento de aplicações e automações empresariais. O n8n, uma proeminente plataforma de automação de workflows de código aberto, exemplifica essa mudança ao capacitar equipes técnicas a conectar APIs, bancos de dados e serviços através de um editor visual intuitivo. Esta democratização do desenvolvimento, embora acelere drasticamente a criação de automações, introduz desafios críticos de segurança que este trabalho busca endereçar.

A mudança de um modelo de codificação imperativa tradicional para um modelo de configuração declarativa, onde a lógica é definida visualmente e armazenada como objetos JSON, introduz um novo paradigma para a segurança de aplicações. Existe uma lacuna crítica de governança: o "código-fonte"da aplicação (arquivo JSON do workflow) não está sujeito ao mesmo rigor de segurança que a plataforma na qual é executado. As equipes de segurança não podem mais depender exclusivamente das garantias de segurança do fornecedor; em vez disso, devem estabelecer seus próprios processos de garantia para os ativos criados na plataforma.

1.1 Motivação e Contexto

O interesse por este tema surge de uma necessidade prática identificada no ambiente de trabalho, onde a plataforma n8n é utilizada diariamente. Com o crescimento do número de workflows e a expansão do uso da ferramenta para diferentes áreas organizacionais, observou-se a necessidade crítica de averiguar a segurança dos workflows desenvolvidos. Particularmente preocupante é o fato de que muitos usuários que criam workflows não possuem conhecimento técnico aprofundado em segurança, o que pode resultar no desenvolvimento não intencional de vulnerabilidades dentro do sistema.

Esta experiência prática evidencia a lacuna de governança identificada na literatura, onde o "desenvolvedor cidadão"assume responsabilidades de desenvolvimento sem o treinamento formal necessário para identificar riscos de segurança. A própria n8n implementa práticas de segurança robustas em seu código-fonte, incluindo a utilização de Testes de Segurança de Aplicações Estáticas (SAST) como parte de seu pipeline de Integração Contínua/Entrega Contínua (CI/CD). Contudo, essa varredura não se estende, nem poderia se estender, aos workflows criados pelos seus usuários. A responsabilidade pela concepção de workflows seguros é explicitamente delegada ao usuário.

O potencial de impacto no contexto brasileiro é particularmente significativo considerando a crescente digitalização de processos empresariais e a adoção de ferramentas de automação no país. A Lei Geral de Proteção de Dados (LGPD) e outras regulamentações de segurança cibernética criam uma demanda específica por ferramentas que possam garantir a conformidade e segurança de automações empresariais. Uma ferramenta SAST especializada para n8n junto com a rápida criação de regras novas, de acordo com a demanda, utilizando inteligência artificial, pode contribuir significativamente para elevar o nível de segurança das automações desenvolvidas por organizações brasileiras, reduzindo riscos de vazamento de dados e ataques cibernéticos que podem resultar em penalidades regulatórias e danos reputacionais.

1.2 Problema Abordado

O principal desafio de segurança no n8n não é uma falha da plataforma em si, mas sim uma falha potencial na implementação do "usuário-desenvolvedor". Os workflows podem conter seis classes críticas de vulnerabilidades:

- **SQL Injection (SQLi):** Entradas controladas pelo usuário inseridas em consultas SQL sem sanitização adequada.
- **Command/Code Injection:** Dados não confiáveis usados em comandos executados no servidor.
- **Secret Sprawl & Chaining:** Dispersão de segredos (chaves de API, tokens) em configurações ou logs.
- **SSRF:** Entradas controladas pelo usuário determinam URLs em requisições HTTP sem validação.
- **Insecure Data Handling:** Falhas na proteção de dados sensíveis durante armazenamento ou transmissão.
- **Denial of Service (DoS):** Workflows suscetíveis a loops infinitos ou operações intensivas.

A natureza declarativa do JSON do n8n, embora abstraia a complexidade do código tradicional, paradoxalmente torna certos tipos de análise estática mais fáceis e precisas. Ele declara explicitamente as relações de fluxo de dados através do array `connections`, permitindo regras de análise de contaminação (taint analysis) com elevado grau de confiança e baixa taxa de falsos positivos.

1.3 Delimitação do Escopo

Este trabalho foca especificamente no desenvolvimento e validação de uma abordagem híbrida para análise estática de segurança de workflows n8n. O escopo está claramente delimitado:

Incluído no escopo:

- Análise estática de arquivos JSON de workflows n8n
- Avaliação detalhada de duas ferramentas do estado da arte: Agentic Radar e Semgrep
- Detecção das seis classes principais de vulnerabilidades identificadas
- Desenvolvimento de metodologia para criação de regras Semgrep utilizando inteligência artificial
- Uso de workflows com vulnerabilidades conhecidas como dataset de validação
- Análise de viabilidade técnica e métricas de performance

Excluído do escopo:

- Implementação completa de uma ferramenta SAST de produção
- Interface gráfica completa para usuários finais

- Análise dinâmica ou execução real de workflows
- Análise de nós customizados da comunidade
- Correção automática de vulnerabilidades detectadas
- Otimização de performance de workflows
- Deploy em ambientes de produção corporativos
- Análise de vulnerabilidades em nível de infraestrutura da plataforma n8n

1.4 Contribuições Esperadas

Este trabalho oferece contribuições técnicas, metodológicas e científicas para segurança em plataformas LCNC:

1. **Análise Comparativa:** Avaliação sistemática de Agentic Radar e Semgrep, documentando coberturas complementares e aplicabilidade ao contexto n8n.
2. **Validação de Viabilidade:** Demonstração empírica da eficácia da análise estática de estruturas JSON declarativas.
3. **Abordagem Híbrida:** Arquitetura que combina ferramentas especializadas e genéricas configuráveis, maximizando cobertura.
4. **Metodologia com IA:** Uso de inteligência artificial para acelerar a criação de regras de detecção.
5. **Documentação de Lacunas:** Identificação de limitações e oportunidades para pesquisas futuras.

A relevância para a formação profissional está diretamente alinhada com as tendências emergentes do mercado de tecnologia. O desenvolvimento de competências em análise de segurança para plataformas LCNC representa uma especialização altamente demandada no mercado, posicionando o profissional na interseção entre desenvolvimento de baixo código e segurança cibernética. A capacidade de identificar e mitigar riscos em ambientes de desenvolvimento democratizado torna-se um diferencial competitivo crucial à medida que mais organizações adotam estratégias de desenvolvimento cidadão.

2 Fundamentação Teórica e Estado da Arte

Esta seção apresenta a revisão da literatura relevante e análise crítica das ferramentas existentes para análise estática de segurança em plataformas Low-Code/No-Code, com foco específico na plataforma n8n. A análise concentra-se em duas ferramentas principais que representam abordagens distintas ao problema: o Agentic Radar, uma solução moderna baseada em agentes de IA, e o Semgrep, um motor de análise estática clássico e extensível.

2.1 Técnicas de SAST para Plataformas Low-Code/No-Code

O cenário de ferramentas para análise de segurança estática de workflows n8n é emergente e fragmentado, apresentando abordagens distintas que podem ser categorizadas em ferramentas dedicadas e adaptáveis. É crucial ressaltar que, até o momento, **não existe uma abordagem oficial ou solução consolidada para detecção de vulnerabilidades especificamente em workflows do n8n**. Esta lacuna representa tanto um desafio quanto uma oportunidade significativa para pesquisa e desenvolvimento na área.

2.1.1 Abordagem Moderna: Agentic Radar e IA Agêntica

O Agentic Radar, desenvolvido pela SPLX AI, emerge como a ferramenta de código aberto mais proeminente especificamente projetada para análise de segurança de workflows em plataformas de automação, incluindo o n8n [**splxai_n8n_scanning**]. A ferramenta representa uma abordagem moderna ao problema, incorporando capacidades de Inteligência Artificial Agêntica (Agentic AI) para análise de fluxos de trabalho onde agentes de IA interagem com ferramentas, APIs e serviços externos.

A utilização de IA no Agentic Radar manifesta-se em múltiplas dimensões. Primeiro, a ferramenta implementa análise estática consciente do contexto, capaz de interpretar a estrutura JSON de workflows e construir representações gráficas dos fluxos de dados. Segundo, o modo de teste (`test`) utiliza modelos de linguagem (especificamente a API da OpenAI) para executar testes adversariais em tempo de execução, simulando ataques de injeção de prompt, tentativas de extração de informações sensíveis e geração de conteúdo prejudicial [**splxai_medium_scanning**]. Esta capacidade de testar dinamicamente a robustez de sistemas de agentes de IA através de entradas adversariais geradas por outros modelos de linguagem representa uma aplicação sofisticada de IA para segurança.

Terceiro, a ferramenta correlaciona padrões identificados nos workflows com categorias de risco estabelecidas pelo OWASP LLM Top 10, sugerindo a utilização de classificação assistida por IA para mapeamento de vulnerabilidades. Quarto, a geração de relatórios com visualizações gráficas interativas e recomendações de remediação contextualizadas indica o uso de técnicas de geração de linguagem natural para produzir explicações técnicas compreensíveis.

O foco atual do Agentic Radar concentra-se em riscos emergentes específicos de sistemas de agentes de IA: injeção de prompt, vazamento de PII, geração de conteúdo nocivo e disseminação de desinformação. Esta especialização não cobre vulnerabilidades tradicionais de aplicações web, limitando sua cobertura a aproximadamente 16,7% das classes identificadas.

2.1.2 Abordagem Clássica: Semgrep e Análise Baseada em Padrões

O Semgrep, desenvolvido pela Semgrep Inc., representa uma abordagem clássica de análise estática de código, fundamentada em correspondência de padrões consciente de sintaxe [**semgrep_platform**]. Diferentemente de ferramentas baseadas em expressões regulares, o Semgrep realiza parsing do código-fonte em árvores sintáticas abstratas (AST), oferecendo compreensão semântica da estrutura do código. A ferramenta suporta mais de 30 linguagens de programação, incluindo análise nativa de formatos estruturados como JSON e YAML, tornando-a particularmente relevante para a análise de workflows do n8n.

A arquitetura do Semgrep fundamenta-se em uma linguagem de regras declarativa, onde padrões de detecção são especificados em arquivos YAML. Esta abordagem permite que equipes de segurança desenvolvam conjuntos de regras personalizados sem a complexidade de manipulação direta de ASTs ou construção de compiladores [**semgrep_custom_rules**]. No contexto

do n8n, regras podem ser desenvolvidas para detectar padrões inseguros específicos: concatenação de entradas não confiáveis em consultas SQL, construção dinâmica de URLs em nós HTTP Request baseada em dados externos, uso de expressões dinâmicas em comandos de execução e exposição de credenciais em configurações.

A capacidade de análise de fluxo de dados (dataflow analysis) do Semgrep é particularmente relevante para a detecção de vulnerabilidades de injeção. A ferramenta permite configurar fontes de dados não confiáveis (sources) — como nós Webhook que recebem entradas externas — e destinos perigosos (sinks) — como parâmetros de consultas SQL ou comandos de execução. O motor de análise então rastreia automaticamente o fluxo de dados contaminados (taint tracking) desde as fontes até os sinks, sinalizando caminhos que não incluem validação ou sanitização adequada [**prototype_pollution**].

A análise revela que o Semgrep oferece potencial de cobertura para aproximadamente 66,7% das classes identificadas, mas esta capacidade é inteiramente potencial: depende do desenvolvimento de regras personalizadas. Até o momento, nenhum conjunto de regras específico para n8n existe no registro comunitário público.

A principal limitação é a ausência de conhecimento nativo do esquema n8n, exigindo investimento estimado entre 40 e 60 horas de trabalho especializado para desenvolver regras customizadas abrangentes.

2.2 Explicabilidade em Ferramentas de Análise de Segurança

A explicabilidade refere-se à capacidade de um sistema fornecer insights compreensíveis sobre seu funcionamento e decisões, sendo particularmente crítica em contextos de segurança onde a confiança nas detecções é essencial para a adoção prática das ferramentas [**owasp_source_code_analysis**]. Tanto o Agentic Radar quanto o Semgrep implementam mecanismos de explicabilidade, embora através de abordagens distintas.

O Agentic Radar gera relatórios em formato HTML com visualizações gráficas interativas dos fluxos de trabalho, identificação clara de vulnerabilidades e explicações técnicas detalhadas sobre por que determinado padrão foi classificado como risco. As recomendações de remediação são contextualizadas e acionáveis, frequentemente incluindo exemplos de código corrigido. O alinhamento explícito com o OWASP LLM Top 10 fornece uma taxonomia padronizada que facilita a comunicação de riscos com stakeholders não técnicos. A capacidade de visualização gráfica é particularmente valiosa para workflows complexos, permitindo que analistas de segurança compreendam o contexto das vulnerabilidades dentro do fluxo de dados completo.

O Semgrep, por sua vez, oferece explicabilidade através da transparência das regras. Cada regra é especificada em YAML legível, onde o padrão de detecção, as condições lógicas e os metadados (severidade, mensagens explicativas, sugestões de correção) são explicitamente declarados. Esta transparência permite que desenvolvedores e analistas de segurança compreendam exatamente o que está sendo detectado e por quê. Os relatórios gerados incluem o padrão correspondido, a localização exata no código (número de linha) e a mensagem explicativa definida na regra. O suporte a metadados arbitrários permite enriquecer as detecções com mapeamentos a frameworks de segurança (OWASP LCNC Top 10, CWE), referências a documentação e exemplos de mitigação.

A principal diferença está na natureza da explicabilidade: o Agentic Radar oferece explicações contextuais baseadas na análise holística do fluxo de trabalho, enquanto o Semgrep oferece explicações baseadas em regras que detalham por que um padrão específico é considerado inseguro. Ambas as abordagens são complementares e valiosas em diferentes contextos de uso.

2.3 Segurança e Privacidade em Sistemas de Análise Baseados em IA

A integração de Inteligência Artificial em ferramentas de análise de segurança introduz considerações específicas relacionadas à segurança e privacidade que devem ser cuidadosamente avaliadas, especialmente em contextos corporativos com requisitos rigorosos de conformidade regulatória [**n8n_privacy**].

O Agentic Radar, particularmente em seu modo de teste dinâmico, depende da API da OpenAI para execução de testes adversariais. Esta dependência cria múltiplas preocupações. Primeiro, a necessidade de enviar workflows para serviços externos de terceiros levanta questões de privacidade e confidencialidade: workflows corporativos frequentemente contêm lógica de negócio proprietária, credenciais de acesso a sistemas internos e referências a infraestrutura sensível. A transmissão destes artefatos para serviços externos pode violar políticas de segurança organizacionais ou requisitos de conformidade como SOC 2, ISO 27001 ou a Lei Geral de Proteção de Dados (LGPD) no contexto brasileiro [**n8n_legal**].

Segundo, a dependência de serviços externos cria questões de disponibilidade e custo operacional. Interrupções na disponibilidade da API da OpenAI impactam diretamente a capacidade de executar análises de segurança, um cenário inaceitável para pipelines de CI/CD críticos onde feedback rápido é essencial. Os custos por chamada de API, embora individualmente pequenos, podem acumular-se significativamente em ambientes corporativos analisando centenas ou milhares de workflows regularmente.

Terceiro, existe a preocupação de segurança da cadeia de suprimentos: a confiança na robustez e segurança dos modelos de linguagem utilizados pela API externa. Vulnerabilidades ou comportamentos inesperados nos modelos podem impactar a confiabilidade das análises, potencialmente produzindo falsos positivos ou, mais gravemente, falsos negativos que deixam vulnerabilidades reais não detectadas.

O Semgrep, como ferramenta de análise estática tradicional, pode operar completamente offline e auto-hospedado, não requerendo transmissão de código para serviços externos. A versão open-source é inteiramente local, endereçando as preocupações de privacidade. No entanto, a plataforma Semgrep Cloud opcional, que oferece dashboards de equipe, agregação de resultados históricos e gerenciamento centralizado de políticas, requer envio de código-fonte para serviços externos operados pela Semgrep Inc. Para organizações com requisitos rigorosos de conformidade, esta limitação restringe o uso à versão auto-hospedada, que carece de algumas funcionalidades de colaboração da plataforma cloud [**gitlab_sast**].

A oportunidade de utilizar IA para geração assistida de regras Semgrep, discutida na próxima subseção, deve ser implementada com consideração cuidadosa destes aspectos de privacidade. Modelos de linguagem podem ser utilizados para sugerir regras baseadas em descrições de vulnerabilidades ou exemplos de padrões inseguros, mas a geração deve, idealmente, ocorrer localmente ou através de modelos de IA auto-hospedados para evitar vazamento de conhecimento proprietário sobre padrões de vulnerabilidade específicos da organização.

2.4 Gaps e Oportunidades

A análise das ferramentas existentes e do estado da arte revela lacunas significativas e oportunidades promissoras para pesquisa e desenvolvimento na área de análise de segurança para workflows do n8n.

2.4.1 Gap Fundamental: Ausência de Solução Específica para n8n

Conforme evidenciado pela análise anterior, **não existe atualmente uma ferramenta dedicada especificamente à detecção de vulnerabilidades em workflows do n8n**. As soluções existentes oferecem cobertura parcial e complementar, mas nenhuma aborda o problema de forma abrangente. Esta lacuna representa um risco substancial para organizações que dependem da plataforma para automações críticas.

Esta lacuna é particularmente preocupante considerando a mudança de paradigma de segurança identificada na literatura: o workflow JSON não é meramente configuração, mas sim o código-fonte da aplicação, e deve ser submetido ao mesmo rigor de análise que código tradicional [**owasp_lcnc_top10**]. A ausência de ferramentas adequadas resulta na perpetuação da lacuna de governança, onde workflows potencialmente inseguros são implantados em produção sem análise automatizada.

2.4.2 Oportunidade: IA para Geração Assistida de Regras Semgrep

Uma oportunidade promissora é a utilização de IA para geração assistida de regras Semgrep. O desenvolvimento manual representa uma barreira significativa (40-60 horas estimadas). Modelos de linguagem podem ser aplicados para:

1. **Geração de regras a partir de descrições em linguagem natural:** Analistas de segurança poderiam descrever vulnerabilidades em linguagem natural (ex.: "detectar concatenação de entrada de webhook em consulta SQL sem sanitização") e o modelo geraria a regra Semgrep correspondente em YAML, incluindo padrões sintáticos, condições lógicas e metadados apropriados.

2. **Otimização de regras existentes para redução de falsos positivos:** Modelos de IA poderiam analisar regras que produzem taxas elevadas de falsos positivos e sugerir refinamentos nas condições lógicas, adicionando verificações contextuais que distinguem padrões genuinamente inseguros de uso legítimo.

3. **Aprendizado de padrões a partir de workflows reais:** Análise de datasets de workflows (como os 2.000+ workflows públicos identificados na literatura [**n8n_workflow_analysis**]) através de técnicas de aprendizado não supervisionado poderia identificar clusters de padrões comuns, distinguindo práticas seguras de inseguras e gerando regras que detectam desvios de padrões seguros estabelecidos.

4. **Manutenção automatizada de regras:** À medida que a plataforma n8n evolui, adicionando novos tipos de nós ou modificando estruturas de parâmetros, modelos de IA poderiam analisar changelogs e documentação de API para sugerir atualizações nas regras existentes ou identificar necessidade de novas regras para cobrir funcionalidades introduzidas.

Esta abordagem híbrida — motor de análise clássico (Semgrep) potencializado por IA generativa para desenvolvimento de conhecimento de segurança — combina as forças de ambos os paradigmas: a confiabilidade, transparência e performance da análise estática baseada em padrões com a flexibilidade, capacidade de adaptação e eficiência de desenvolvimento proporcionadas por modelos de linguagem.

A implementação desta abordagem utiliza prompts especializados e few-shot learning, fornecendo ao Claude 3.5 Sonnet exemplos de regras Semgrep existentes para outras plataformas

e solicitando adaptação para o esquema JSON do n8n. A validação das regras geradas por IA através de testes automatizados em workflows é essencial para garantir confiabilidade antes da implantação em ambientes de produção.

2.4.3 Oportunidade: Arquitetura Híbrida para Cobertura Completa

A análise comparativa sugere que uma arquitetura híbrida poderia alcançar cobertura próxima a 100% das classes de vulnerabilidades. Esta estratégia maximiza as forças complementares: Agentic Radar oferece análise especializada em agentes de IA, enquanto Semgrep oferece motor flexível e poderoso para padrões customizados.

A integração destas ferramentas em um pipeline unificado, com agregação de resultados, classificação por severidade e categoria OWASP LCNC Top 10, e geração de relatórios consolidados, representa uma oportunidade de pesquisa significativa para o desenvolvimento de uma solução SAST verdadeiramente abrangente para o ecossistema n8n.

3 Metodologia

Esta seção descreve a abordagem metodológica adotada para o desenvolvimento e validação da ferramenta SAST proposta, detalhando o processo de análise das ferramentas existentes, a metodologia inovadora de geração de regras assistida por inteligência artificial e os procedimentos de validação empregados.

3.1 Abordagem Metodológica Geral

A pesquisa adota metodologia exploratória e experimental, fundamentada em três pilares:

1. **Pesquisa exploratória:** Análise comparativa de Agentic Radar e Semgrep, documentando capacidades, limitações e aplicabilidade.
2. **Abordagem híbrida:** Arquitetura combinando ferramentas complementares para maximizar cobertura.
3. **IA para aceleração:** Uso de modelos de linguagem para geração assistida de regras, reduzindo overhead de desenvolvimento.

3.2 Análise das Ferramentas Existentes

A análise comparativa das ferramentas do estado da arte seguiu um protocolo estruturado para garantir avaliação objetiva e sistemática das capacidades e limitações de cada solução.

3.2.1 Critérios de Avaliação

As ferramentas foram avaliadas segundo quatro critérios principais:

1. **Cobertura de vulnerabilidades:** Capacidade de detectar as seis classes identificadas (classificada como ALTO, MÉDIO, BAIXO ou NENHUMA).
2. **Performance:** Tempo de execução, throughput e escalabilidade.
3. **Facilidade de uso:** Complexidade de instalação, documentação e interface.
4. **Qualidade dos relatórios:** Clareza, severidade, explicações e sugestões de remediação.

3.2.2 Processo de Análise

O processo de análise de cada ferramenta seguiu um protocolo sistemático estruturado em quatro etapas principais. Inicialmente, realizou-se a **instalação e configuração** das ferramentas Agentic Radar (via `pip install agentic-radar`) e Semgrep (via `pip install semgrep`) em ambiente Python 3.10+. Durante esta etapa, foram documentadas todas as dependências, requisitos de configuração (como variáveis de ambiente) e possíveis incompatibilidades encontradas, garantindo a reprodutibilidade do ambiente de testes.

Na segunda etapa, conduziram-se **testes com workflows de exemplo**, executando cada ferramenta sobre um conjunto de workflows representativos que cobrem diferentes tipos de nós relevantes para segurança, incluindo Webhook, MySQL, HTTP Request, Execute Command e Code. Esta diversidade de nós permitiu avaliar a capacidade das ferramentas em diferentes contextos de vulnerabilidades.

A terceira etapa consistiu na **análise de capacidades de detecção**, avaliando as seis classes de vulnerabilidades em três workflows de teste e verificando se cada ferramenta detecta os padrões inseguros implementados. Para cada detecção ou ausência de detecção, documentou-se sistematicamente se tratava-se de true positive, false positive, true negative ou false negative, permitindo análise quantitativa posterior.

A etapa final envolveu a **documentação de cobertura e limitações**, realizando o mapeamento sistemático da cobertura de cada ferramenta para as seis classes de vulnerabilidades, identificando lacunas fundamentais (vulnerabilidades não cobertas) e documentando limitações técnicas, operacionais e de privacidade. Os resultados desta análise comparativa foram consolidados nas seções anteriores deste documento, fornecendo a base empírica para a proposta da abordagem híbrida.

3.3 Geração de Regras Semgrep Assistida por IA

A metodologia de geração assistida de regras representa a contribuição metodológica mais inovadora deste trabalho, aplicando técnicas de inteligência artificial generativa para acelerar e sistematizar o desenvolvimento de conhecimento de segurança específico do domínio do n8n.

3.3.1 Preparação de Workflows de Teste

Desenvolveram-se três workflows de teste implementando deliberadamente as seis classes de vulnerabilidades em cenários realistas. Exemplos incluem SQL Injection (entrada de webhook passada diretamente para consulta MySQL), Command Injection (entrada usada em Execute Command) e SSRF (URL dinâmica sem validação).

Os workflows foram documentados detalhadamente, especificando localização exata dos padrões inseguros, categorias OWASP e severidades. Esta documentação serve como ground truth para validação.

3.3.2 Metodologia de Geração de Regras com IA

A geração assistida segue processo iterativo estruturado. Primeiro, analisou-se o esquema JSON do n8n (arrays `nodes` e `connections`, sintaxe de expressões dinâmicas, tipos de nós críticos).

Com este conhecimento, utilizou-se Claude 3.5 Sonnet através de prompts estruturados em três componentes: *Contexto* (esquema n8n), *Objetivo* (vulnerabilidade a detectar) e *Formato* (estrutura YAML com campos obrigatórios).

Exemplo de prompt estruturado:

> *"Você é um especialista em segurança de aplicações e na ferramenta Semgrep. Crie uma regra Semgrep em formato YAML que detecta vulnerabilidades de SQL Injection em workflows do n8n.*
>
> *Contexto: Workflows do n8n são arquivos JSON onde o array 'nodes' contém objetos representando etapas de processamento. Nós de banco de dados MySQL têm type='n8n-nodes-base.mysql' e o parâmetro 'parameters.query' contém a consulta SQL. Expressões dinâmicas usam sintaxe {{ $json.campo }}.*
>
> *Objetivo: Detectar quando o parâmetro 'query' de um nó MySQL contém expressões {{ ... }} que referenciam dados de entrada não confiáveis, indicando possível concatenação de strings em SQL sem sanitização adequada.*
>
> *Formato: A regra deve incluir id, message explicativa, severity='ERROR', metadata com owasp_lcnc='LCNC-SEC-06' e CWE, e padrões que identifiquem o JSON específico do n8n."*

O processo seguiu ciclo iterativo: (1) Geração inicial, (2) Teste em workflows, (3) Avaliação de detecções, (4) Refinamento com feedback, (5) Convergência. Tipicamente, foram necessárias 2-4 iterações por regra, reduzindo drasticamente o tempo de desenvolvimento.

3.3.3 Validação das Regras Geradas

As regras geradas foram validadas rigorosamente: (1) Execução em workflows de teste com documentação sistemática, (2) Ajuste manual para redução de falsos positivos (padrões negativos e contextualizações), (3) Documentação completa incluindo descrição técnica, exemplos, mapeamento OWASP/CWE e sugestões de remediação.

3.4 Validação e Métricas

O processo de validação da abordagem proposta combina avaliação quantitativa (métricas de detecção) e qualitativa (cobertura de vulnerabilidades e utilidade prática).

3.4.1 Testes de Detecção

A validação quantitativa das capacidades de detecção seguiu um protocolo experimental rigorosamente estruturado para garantir resultados objetivos e reproduzíveis. Ambas as ferramentas — Agentic Radar e Semgrep com regras customizadas — foram executadas sobre os três workflows de teste, com documentação sistemática de todos os resultados obtidos. Esta documentação detalhada permite rastreabilidade completa e validação independente dos achados.

Cada detecção ou ausência de detecção foi classificada utilizando a taxonomia padrão de sistemas de detecção: *True Positive (TP)* para vulnerabilidades reais corretamente identificadas, *False Positive (FP)* para detecções em código seguro (falsos alarmes), *True Negative (TN)* para ausência correta de detecção em código seguro, e *False Negative (FN)* para vulnerabilidades reais não detectadas.

A partir desta classificação, calcularam-se as métricas padrão de avaliação de sistemas de detecção:

- Precisão: $P = \frac{TP}{TP+FP}$ (proporção de detecções corretas)

- Recall: $R = \frac{TP}{TP+FN}$ (proporção de vulnerabilidades encontradas)
- F1-Score: $F_1 = 2 \cdot \frac{P \cdot R}{P+R}$ (média harmônica)
- Taxa de Falsos Positivos: $\frac{FP}{FP+TN}$

Estas métricas foram calculadas tanto por ferramenta individual quanto para a abordagem híbrida combinada, permitindo comparação objetiva das capacidades de detecção e demonstrando quantitativamente os benefícios da estratégia híbrida proposta.

3.4.2 Avaliação de Cobertura

A avaliação de cobertura verificou sistematicamente a capacidade da abordagem proposta de endereçar o espectro completo de vulnerabilidades identificadas na revisão da literatura. Para cada uma das seis classes de vulnerabilidades — SQL Injection, Command Injection, Secret Sprawl, SSRF, Insecure Data Handling e DoS — verificou-se se ao menos uma das ferramentas da abordagem híbrida oferece capacidade de detecção efetiva.

A análise comparativa quantitativa revelou diferenças significativas na cobertura oferecida por cada ferramenta individualmente. O Agentic Radar isolado oferece cobertura de 16,7% (1 de 6 classes), focando especificamente em riscos emergentes de agentes de IA. O Semgrep customizado isolado apresenta cobertura potencial de 66,7% (4 de 6 classes com cobertura ALTO), endereçando vulnerabilidades tradicionais de aplicações web. A abordagem híbrida, combinando ambas as ferramentas, alcança cobertura próxima a 100% através da complementaridade estratégica das capacidades de cada solução.

Adicionalmente, documentaram-se as lacunas residuais, identificando categorias de vulnerabilidades que permanecem parcialmente cobertas ou requerem capacidades de análise além do escopo das ferramentas utilizadas. Um exemplo notável é a detecção de Negação de Serviço (DoS) através de análise completa de ciclos no grafo de execução, que requereria capacidades de análise de fluxo de controle mais sofisticadas do que as oferecidas pelas ferramentas atuais.

Os resultados desta validação são apresentados detalhadamente na Seção 7 (Avaliação Experimental), fornecendo evidência empírica robusta da viabilidade e efetividade da abordagem proposta para detecção de vulnerabilidades em workflows do n8n.

4 Objetivos e Métricas de Sucesso

Esta seção estabelece os objetivos gerais e específicos do trabalho, definindo critérios quantitativos e qualitativos para avaliar o sucesso da validação da abordagem proposta. É importante ressaltar que o foco não é desenvolver uma ferramenta SAST completa de produção, mas sim validar a viabilidade técnica da abordagem híbrida e da metodologia de geração de regras assistida por inteligência artificial.

4.1 Objetivo Geral

O objetivo geral é validar a **viabilidade técnica** de uma abordagem híbrida de análise estática para workflows n8n. Especificamente:

1. Demonstrar que análise estática de estruturas JSON declarativas é possível e efetiva.

2. Desenvolver metodologia de geração assistida de regras usando IA para acelerar desenvolvimento.

3. Avaliar comparativamente Agentic Radar e Semgrep, documentando coberturas e limitações.

O trabalho não visa criar ferramenta de produção, mas estabelecer base científica que comprove viabilidade da abordagem.

4.2 Objetivos Específicos

Os objetivos específicos são:

OE1: Análise comparativa: Avaliar cobertura de vulnerabilidades, limitações técnicas e aplicabilidade ao contexto n8n, produzindo métricas quantitativas e caracterização qualitativa.

OE2: Metodologia com IA: Criar abordagem sistemática para geração de regras Semgrep usando Claude 3.5 Sonnet, incluindo estruturação de prompts e processo iterativo de refinamento.

OE3: Regras customizadas: Desenvolver regras para classes não cobertas pelo Agentic Radar, incluindo padrões, condições lógicas, metadados e mensagens explicativas.

OE4: Validação empírica: Executar ferramentas em workflows de teste, documentar detecções e calcular métricas de cobertura híbrida.

OE5: Documentação de lacunas: Identificar limitações, classes parcialmente cobertas e oportunidades para pesquisa futura.

4.3 Métricas de Avaliação

A avaliação do sucesso do trabalho será realizada através de critérios objetivos quantitativos e critérios qualitativos que, em conjunto, permitem caracterizar a viabilidade e utilidade da abordagem proposta.

4.3.1 Métricas Quantitativas

As métricas quantitativas estabelecem valores-alvo objetivos para aspectos mensuráveis do trabalho, permitindo avaliação empírica do alcance dos objetivos estabelecidos.

Tabela 1: Métricas quantitativas e valores-alvo

Métrica	Definição	Meta
Cobertura de Vulnerabilidades (Híbrida)	Percentual das 6 classes de vulnerabilidades com detecção efetiva através da abordagem híbrida (Agentic Radar + Semgrep)	$\geq 80\%$
Cobertura Agentic Radar	Percentual das 6 classes adequadamente cobertas pelo Agentic Radar isoladamente	Documentar % real
Cobertura Semgrep	Percentual das 6 classes cobertas por regras Semgrep customizadas	Documentar % real
Regras Desenvolvidas	Número de regras Semgrep funcionais criadas utilizando metodologia assistida por IA	≥ 4 regras (uma por classe não coberta)
Redução de Tempo	Economia de tempo no desenvolvimento de regras com IA vs. estimativa de desenvolvimento manual (8-12h por regra)	Documentar economia em horas

A cobertura de vulnerabilidades é calculada como proporção de classes com detecção efetiva. A meta de 80% reconhece que algumas categorias podem requerer capacidades além do escopo das ferramentas.

A redução de tempo compara desenvolvimento com IA versus estimativa manual (8-12h por regra), validando quantitativamente o valor da geração assistida.

4.3.2 Métricas Qualitativas

As métricas qualitativas avaliam aspectos não diretamente quantificáveis, mas essenciais para caracterizar a viabilidade prática e utilidade da abordagem proposta. Cada métrica é avaliada através de critérios específicos de aceitação.

Viabilidade Técnica:

A validação fundamental do trabalho é demonstrar que a análise estática de workflows n8n é tecnicamente viável e efetiva. Os critérios de avaliação incluem:

- A análise estática de arquivos JSON de workflows n8n permite identificação de padrões inseguros com precisão suficiente para ser útil em ambientes reais?
- As regras Semgrep geradas por IA utilizando Claude 3.5 Sonnet são funcionais e detectam corretamente as vulnerabilidades-alvo nos workflows de teste?
- A abordagem híbrida fecha as lacunas de cobertura das ferramentas isoladas, oferecendo cobertura complementar efetiva?
- As limitações identificadas são documentadas de forma clara e honesta, permitindo avaliação realista da aplicabilidade da solução?

Usabilidade:

Embora o foco não seja desenvolver uma ferramenta de produção completa, a viabilidade prática requer que as ferramentas sejam razoavelmente acessíveis:

- A instalação e configuração do Agentic Radar e Semgrep é viável com complexidade aceitável (máximo de 5 comandos por ferramenta)?
- A documentação disponível (tanto das ferramentas quanto da metodologia proposta) é suficiente para permitir reprodução independente do estudo por outros pesquisadores?
- A execução das análises sobre workflows de teste é direta, sem requerer expertise profunda em detalhes de implementação das ferramentas?

Clareza de Resultados:
A utilidade prática das detecções depende da qualidade da comunicação dos resultados:

- Os relatórios gerados pelas ferramentas identificam claramente as vulnerabilidades detectadas, incluindo localização exata no workflow (nó e parâmetro específico)?
- Existe mapeamento explícito das detecções para categorias reconhecidas do OWASP LCNC Top 10, facilitando comunicação com stakeholders?
- As explicações técnicas fornecidas são compreensíveis e as sugestões de remediação são acionáveis?
- A documentação distingue claramente entre detecções confirmadas (true positives) e potenciais falsos positivos, orientando o analista na interpretação dos resultados?

4.4 Benchmarks Comparativos

A avaliação da contribuição da abordagem híbrida requer comparação sistemática com três cenários baseline, cada um representando uma estratégia alternativa de análise de segurança para workflows n8n. A Tabela 2 sintetiza as coberturas esperadas para cada cenário.

Tabela 2: Benchmarks comparativos de cobertura de vulnerabilidades

Cenário	Cobertura Esperada	Observações
Agentic Radar isolado	∼16,7% (1/6 classes)	Baseline 1: Ferramenta especializada com foco em riscos emergentes de agentes de IA (injeção de prompt, vazamento de PII, geração de conteúdo prejudicial). Não cobre vulnerabilidades tradicionais de aplicações web.
Semgrep isolado	∼66,7% (4/6 classes)	Baseline 2: Potencial teórico para vulnerabilidades tradicionais (SQL Injection, Command Injection, SSRF, Secret Sprawl), mas requer desenvolvimento completo de regras customizadas. Sem regras, cobertura é 0%.
Abordagem Híbrida	∼83-100% (5-6/6 classes)	Proposta: Combinação complementar de Agentic Radar (riscos de IA) + Semgrep customizado (vulnerabilidades tradicionais). Maximiza cobertura ao combinar forças de ambas as ferramentas.

O cenário **Agentic Radar isolado** (16,7%) demonstra que ferramentas especializadas são insuficientes como solução autônoma, estabelecendo o limite inferior.

O cenário **Semgrep isolado** (66,7% potencial) evidencia a barreira crítica: sem regras customizadas, a cobertura é zero, estabelecendo o limite superior para ferramentas tradicionais.

A **Abordagem Híbrida** visa superar ambas limitações através de estratégia complementar, com meta de 83-100% de cobertura.

A validação envolverá execução dos três cenários sobre workflows de teste, permitindo comparação direta.

4.5 Critérios de Sucesso

O trabalho será considerado bem-sucedido se os seguintes critérios forem satisfeitos:

1. **Viabilidade Técnica:** Ambas ferramentas analisam workflows n8n e detectam ao menos uma vulnerabilidade cada.
2. **IA Validada:** Claude 3.5 Sonnet gera regras funcionais em tempo médio inferior a 2h por regra (economia de 75% vs. manual).
3. **Cobertura Complementar:** Abordagem híbrida cobre mínimo de 80% das classes (5 de 6).
4. **Lacunas Documentadas:** Limitações claramente identificadas, tecnicamente justificadas, com oportunidades específicas para trabalhos futuros.
5. **Reprodutibilidade:** Metodologia documentada com detalhes suficientes para replicação independente.

A satisfação dos cinco critérios estabelece que a abordagem híbrida é viável e representa contribuição significativa para segurança de plataformas LCNC.

5 Escopo da Solução e Requisitos

[Detalhar os casos de uso, arquitetura do sistema, requisitos técnicos e composição dos dados.]

5.1 Casos de Uso

[Descrever os principais fluxos de interação com o sistema.]

5.1.1 Caso de Uso Principal

[Detalhar o cenário principal de uso incluindo atores, pré-condições, fluxo principal e fluxos alternativos.]

Ator primário: *[Nome do ator]*

Pré-condições:

- *[Pré-condição 1]*
- *[Pré-condição 2]*

Fluxo principal:

1. *[Passo 1]*
2. *[Passo 2]*
3. *[Passo 3]*

Fluxos alternativos:

- *[Alternativa 1]*
- *[Alternativa 2]*

5.1.2 Casos Excluídos do Escopo

[Listar explicitamente casos de uso que NÃO serão implementados na PoC.]

5.2 Arquitetura Proposta

[Apresentar a arquitetura lógica do sistema, preferencialmente com diagramas, explicando as camadas, componentes e fluxo de dados. Incluir considerações de segurança (defense in depth, privacy by design).]

5.2.1 Componentes Principais

[Descrever em detalhes cada componente da arquitetura (frontend, backend, modelo de ML, sistema de logs, etc.) e suas responsabilidades.]

5.3 Requisitos Técnicos

[Especificar as tecnologias, frameworks e infraestrutura necessários.]

5.3.1 Tecnologias e Frameworks

[Listar em formato de tabela as tecnologias escolhidas para cada camada/componente com justificativas técnicas.]

Tabela 3: Stack tecnológico

Camada	Tecnologia	Justificativa

5.3.2 Infraestrutura Mínima

[Especificar requisitos de hardware (CPU, GPU, RAM, disco) e rede para desenvolvimento, treinamento e deploy.]

5.3.3 Dados de Treinamento e Validação

[Descrever a composição do dataset (quantidade, proporções, fontes), estratégia de divisão (treino/validação/teste) e considerações de privacidade.]

Tabela 4: Composição do dataset

Conjunto	Classe A	Classe B	Total
Treinamento (70%)			
Validação (15%)			
Teste (15%)			
Total			

6 Ameaças, Riscos e Controles

[Aplicar metodologia STRIDE para identificação sistemática de ameaças, analisar riscos específicos de sistemas de ML, e estabelecer controles de mitigação.]

6.1 Superfícies de Ataque

[Enumerar todos os pontos do sistema que podem ser explorados por atacantes.]

6.2 Análise STRIDE

[Apresentar matriz completa de ameaças usando categorias STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) com controles correspondentes.]

Tabela 5: Matriz de ameaças STRIDE

Categoria	Ameaça	Controle
Spoofing		
Tampering		
Repudiation		
Information Disclosure		
Denial of Service		
Elevation of Privilege		

6.3 Ataques Específicos a Modelos de ML

[Detalhar ameaças particulares a sistemas de aprendizado de máquina.]

6.3.1 Evasion Attacks

[Descrever ataques adversariais que tentam enganar o modelo em tempo de inferência e controles de mitigação.]

6.3.2 Data Poisoning

[Explicar riscos de contaminação do conjunto de treinamento e medidas preventivas.]

6.3.3 Model Extraction

[Abordar riscos de roubo de modelo proprietário através de queries e contramedidas.]

6.4 Considerações de Privacidade (LINDDUN)

[Aplicar metodologia LINDDUN para análise sistemática de riscos de privacidade (Linkability, Identifiability, Non-repudiation, Detectability, Disclosure, Unawareness, Non-compliance).]

6.5 Considerações Éticas

[Discutir aspectos éticos do sistema.]

6.5.1 Vieses Algorítmicos

[Identificar riscos de discriminação, estratégias de mitigação e métodos de avaliação de equidade.]

6.5.2 Transparência e Explicabilidade

[Estabelecer compromissos de transparência e como a explicabilidade será implementada.]

6.5.3 Direito de Contestação

[Definir processo para usuários contestarem decisões automatizadas.]

7 Avaliação Experimental

[Apresentar protocolo experimental, resultados obtidos, comparações e análises.]

7.1 Protocolo Experimental

[Detalhar configuração de hardware/software e metodologia de avaliação.]

7.1.1 Configuração de Hardware e Software

[Especificar ambiente de execução (GPU, RAM, OS, versões de bibliotecas).]

7.1.2 Repetição e Validação Cruzada

[Explicar como a variância foi tratada (múltiplas execuções, seeds diferentes, intervalos de confiança).]

7.2 Resultados

[Apresentar resultados quantitativos e qualitativos.]

7.2.1 Performance no Conjunto de Teste

[Tabela com métricas principais (F1, Precisão, Recall, AUC-ROC) por modelo, comparando com metas estabelecidas.]

Tabela 6: Métricas principais por modelo

Modelo	F1-Score	Precisão	Recall	AUC-ROC
Meta Alvo				

7.2.2 Matriz de Confusão

[Apresentar matriz de confusão com análise de falsos positivos e falsos negativos, identificando padrões nos erros.]

7.2.3 Performance por Tipo/Categoria

[Detalhar desempenho em subcategorias do problema para identificar pontos fortes e fracos.]

7.2.4 Latência e Throughput

[Reportar métricas de tempo de processamento e vazão, com breakdown por etapa do pipeline.]

7.3 Testes de Robustez

[Avaliar resiliência do sistema.]

7.3.1 Transformações Geométricas e Degradações

[Tabela mostrando degradação de performance sob transformações comuns (compressão, ruído, rotação).]

7.3.2 Ataques Adversariais

[Resultados de testes contra ataques adversariais (FGSM, PGD) com diferentes intensidades.]

7.4 Análise de Vieses

[Avaliar equidade do sistema.]

7.4.1 Equidade entre Subgrupos

[Tabela comparando performance entre diferentes subgrupos para identificar possíveis discriminações.]

7.5 Explicabilidade: Análise Qualitativa

[Avaliar qualidade das explicações geradas (Grad-CAM ou outra técnica) através de avaliação por especialistas ou métricas objetivas. Incluir exemplos visuais.]

8 Discussão

[Interpretar resultados, discutir limitações, comparar com estado-da-arte e analisar implicações práticas.]

8.1 Alcance dos Objetivos

[Avaliar se os objetivos estabelecidos foram atingidos, comparando resultados com metas.]

8.2 Limitações Identificadas

[Discutir honestamente as limitações do trabalho.]

8.2.1 Limitações Técnicas

[Identificar restrições técnicas (dependência de qualidade de entrada, escopo limitado, dataset sintético, etc.).]

8.2.2 Limitações de Segurança

[Reconhecer vulnerabilidades residuais e aspectos de segurança que precisam ser melhorados para produção.]

8.3 Comparação com Estado-da-Arte

[Tabela comparativa com trabalhos relacionados e sistemas comerciais, analisando posicionamento da PoC.]

8.4 Trade-offs Segurança vs. Usabilidade

[Discutir compromissos feitos entre segurança, privacidade e usabilidade, justificando escolhas.]

8.5 Implicações Práticas

[Analisar impactos práticos do trabalho.]

8.5.1 Para Usuários/Stakeholders

[Discutir como o sistema pode ser utilizado na prática, benefícios e necessidades de treinamento.]

8.5.2 Para Políticas Públicas

[Refletir sobre implicações para regulamentação, padronização e investimentos públicos.]

9 Conclusões e Próximos Passos

[Sintetizar contribuições, propor trabalhos futuros e fornecer recomendações.]

9.1 Síntese das Evidências

[Resumir as principais conclusões demonstradas pela PoC de forma objetiva.]

9.2 Contribuições

[Listar contribuições metodológicas, técnicas e científicas do trabalho.]

9.3 Trabalhos Futuros

[Propor extensões e melhorias.]

9.3.1 Curto Prazo (3-6 meses)

[Melhorias incrementais possíveis em 3-6 meses.]

9.3.2 Médio Prazo (6-12 meses)

[Desenvolvimento de funcionalidades adicionais ou pilotos em 6-12 meses.]

9.3.3 Longo Prazo (1-2 anos)

[Visão de evolução do sistema em 1-2 anos (escala, novas funcionalidades, federação).]

9.4 Recomendações

[Fornecer orientações práticas para stakeholders interessados em adotar tecnologia similar.]

10 Checklist de Conformidade (LGPD/Ética/Segurança)

10.1 Lei Geral de Proteção de Dados (LGPD)

- ☐ **Minimização de dados (Art. 6º, III):**
  - ☐ *[Item de verificação 1]*
  - ☐ *[Item de verificação 2]*
- ☐ **Finalidade específica (Art. 6º, I):**
  - ☐ *[Item de verificação 1]*
- ☐ **Transparência (Art. 6º, VI):**
  - ☐ *[Item de verificação 1]*
- ☐ **Segurança (Art. 46):**
  - ☐ *[Item de verificação 1]*
- ☐ **Direito à explicação (Art. 20):**
  - ☐ *[Item de verificação 1]*
- ☐ **Não discriminação (Art. 6º, IX):**
  - ☐ *[Item de verificação 1]*

10.2 Segurança da Informação

- ☐ **Criptografia:**
  - ☐ *[Item de verificação 1]*
- ☐ **Autenticação e Autorização:**
  - ☐ *[Item de verificação 1]*
- ☐ **Isolamento:**
  - ☐ *[Item de verificação 1]*
- ☐ **Auditoria:**
  - ☐ *[Item de verificação 1]*

- ☐ **Resiliência:**
  - ☐ *[Item de verificação 1]*

10.3 Ética e IA Responsável

- ☐ **Explicabilidade:**
  - ☐ *[Item de verificação 1]*
- ☐ **Equidade:**
  - ☐ *[Item de verificação 1]*
- ☐ **Accountability:**
  - ☐ *[Item de verificação 1]*
- ☐ **Contestação:**
  - ☐ *[Item de verificação 1]*
- ☐ **Transparência pública:**
  - ☐ *[Item de verificação 1]*

10.4 Ações Pendentes para Produção

- ☐ *[Ação pendente 1]*
- ☐ *[Ação pendente 2]*

Agradecimentos

[Agradecer aos orientadores, colegas, instituições e fontes de financiamento quando aplicável.]

A Detalhes de Implementação

A.1 Estrutura de Diretórios

[Apresentar árvore completa de diretórios do projeto.]

A.2 Comandos para Reprodução

[Lista step-by-step de comandos para setup, treinamento, avaliação e deploy do sistema.]

B Exemplos Adicionais de Resultados

[Figuras adicionais mostrando casos de sucesso e falha, exemplos visuais de classificações corretas e incorretas.]

C Código-fonte Completo

[Link para repositório público com código-fonte e licença de uso.]