UNIVERSIDADE FEDERAL DO PAMPA

Programa de Pós-Graduação em Engenharia de Software (PPGES)

MulitaMiner

Extração estruturada de relatórios de scanners de vulnerabilidades com modelos de linguagem

Dissertação de Mestrado em Engenharia de Software

Douglas Lautert

Orientador: Prof. Dr. Diego Kreutz

Alegrete, 2026

Roteiro

1. Problema e objetivos
2. Fundamentação
3. Pesquisa 1: viabilidade
4. Pesquisa 2: evidência
5. Pesquisa 3: ferramenta
6. Síntese e conclusões

A pesquisa em três etapas

PESQUISA 1

Viabilidade

ERRC 2025

"É possível?"

1 relatório, 34 vulnerabilidades

5 modelos comparados

4 modos de falha

PESQUISA 2

Evidência

SBSeg 2026

"Quanto vem do pipeline?"

208 vulnerabilidades curadas

5 modelos, 450 execuções

3 versões do pipeline

PESQUISA 3

Ferramenta

SBSeg 2026

"Em que escala e a que custo?"

129 relatórios, 6.343 achados

Ferramenta aberta (MIT)

Lote completo: US$ 73,82

Cada etapa responde a uma limitação concreta da etapa anterior.

PARTE 1

Problema e objetivos

O gargalo do ciclo de gestão de vulnerabilidades

O ciclo de gestão de vulnerabilidades

- Equipes de resposta a incidentes varrem os ativos
- Scanners: OpenVAS (aberto) e Tenable WAS (comercial)
- Cada varredura gera um relatório em PDF
- Os PDFs são arquivados como registro histórico

O acervo deveria ser a memória operacional da instituição.

PROBLEMA 1

O acervo em PDF não é consultável

- PDF é feito para leitura humana, não para consulta
- “Quais máquinas ainda expõem esta CVE?” vira trabalho manual
- Sem consulta, não há triagem nem priorização
- O histórico existe, mas não é usado

Memória viva vira arquivo morto.

PROBLEMA 2

Os scanners não falam a mesma língua

Aspecto	OpenVAS	Tenable WAS
Orientação	Detalhe técnico	Risco e priorização
Seções	Vulnerability Insight	Risk Information
CVE-2017-9798	Severidade 5,0 (média)	Severidade 7,5 (alta)

Mesmos campos, nomes diferentes. Mesma falha, severidades diferentes.

MOTIVAÇÃO

A pressão sobre as equipes cresce

48 mil
registros CVE em 2025

131
novos registros por dia

263%
de crescimento desde 2020

44%
sem enriquecimento na NVD

E o plano mais barato do Tenable custa cerca de R$ 21 mil por ano.

Problema e hipótese

PROBLEMA

O histórico de varreduras é inútil na operação.

PDFs heterogêneos, sem estrutura consultável. Conversão manual não escala.

HIPÓTESE CENTRAL

O gargalo está no pipeline, não no modelo.

Investir na entrega do documento rende mais do que trocar de LLM.

Questões de pesquisa

QP1 É viável extrair registros estruturados de PDFs com LLMs?

QP2 Quanto do ganho vem do pipeline e quanto vem do modelo?

QP3 O método se sustenta em uma ferramenta aberta, em escala e a baixo custo?

Objetivo geral

Tornar relatórios de scanners consultáveis, usando LLMs.

E medir quanto disso depende do pipeline, e não do modelo.

Produto da investigação: MulitaMiner, ferramenta de código aberto.

Objetivos específicos

a Unificar os campos de OpenVAS e Tenable WAS

b Demonstrar a viabilidade da extração com LLMs

c Medir omissão e alucinação, não apenas similaridade

d Isolar o efeito do pipeline frente ao do modelo

e Entregar ferramenta aberta e artefatos reprodutíveis

Conceitos centrais

Conceito	Significado neste trabalho
Extração estruturada	Texto livre vira registros com campos nomeados
Esquema canônico	18 campos fixos, independentes do scanner
Omissão	Parte da referência que ficou de fora (métrica primária)
Alucinação	Registro extraído que não existe na referência
ERM	Exact Record Match: campos determinísticos todos exatos
Engenharia de pipeline	Segmentação, tokens, prompt e reparo de saída

O que a literatura cobre, e o que não cobre

JÁ COBERTO

- Extração com LLMs em domínio científico e clínico
- Documentos visualmente ricos (DocLLM, MinerU)
- Entidades de inteligência de ameaças
- Unificação de fontes públicas já textuais

EM ABERTO

- Relatórios de scanners em PDF
- Segmentação consciente da estrutura do documento
- Métricas de cobertura, não só de similaridade
- Ferramenta aberta com referências reprodutíveis

Segmentação: onde o pipeline quebra

HOJE: CHUNKING FIXO

- Corta a vulnerabilidade ao meio
- Perde a visão global do registro
- Trunca seções técnicas
- Cortes ruins disparam alucinação

PROPOSTA: POR REGISTRO

- Um bloco por vulnerabilidade
- Orçamento de tokens por modelo
- Tokenizer nativo de cada modelo
- Reparo de JSON e redivisão adaptativa

Por que similaridade não basta

- ROUGE-L e BERTScore medem o que foi extraído
- Nada dizem sobre o que ficou de fora
- Metade das vulnerabilidades pode render escore excelente
- E uma extração pela metade é operacionalmente inútil

Omissão é a métrica primária: deixar passar custa mais do que registrar a mais.

Lacunas e respostas

Lacuna	Resposta	Etapa
Scanners heterogêneos	Esquema canônico de 18 campos	1 e 3
Segmentação fixa	Um bloco por vulnerabilidade	2 e 3
Avaliação só lexical	Bateria com omissão e alucinação	2
Ganho atribuído ao modelo	Três versões, mesmos modelos	2
Sem ferramenta aberta	MulitaMiner sob licença MIT	3

PARTE 2

As três pesquisas

Viabilidade, evidência e ferramenta: contribuições encadeadas

Como o pipeline evoluiu em três versões

V1 — Linha de base

- Chunking fixo
- Prompt básico
- Marcador é sugestão

V2 — Estrutura

- Marcador vira regra
- Tentativa guiada
- Prompt mais diretivo

V3 — Ajuste fino

- Tokens por modelo
- Tokenizer nativo
- Reparo de JSON
- Prompt 40% menor

Mesmos cinco modelos nas três versões. ERM: 37,5% → 90,4%

Viabilidade da extração com LLMs

- Mapeamento campo a campo entre os dois scanners
- Mapeamento embutido no prompt, com NULL para campos ausentes
- Pipeline modular: ler, dividir, extrair, validar, consolidar
- 1 relatório OpenVAS, 34 vulnerabilidades, referência manual
- 5 modelos comparados

PESQUISA 1

Quatro modos de falha

1 Limite de contexto

O registro inteiro não cabe

2 Truncação semântica

Cortes em NVT: e CVSS:

3 Perda de delimitador

Summary e Impact se fragmentam

4 Variação entre execuções

Tokenização e amostragem

Todos ligados à forma como o documento chega ao modelo.

O que entregou e o que faltou

ENTREGOU

- Viabilidade demonstrada
- Esquema unificado entre scanners
- Protocolo com referência curada

FALTOU

- Só 1 relatório e 34 vulnerabilidades
- Segmentação ainda de tamanho fixo
- Só similaridade lexical: omissão invisível

Essas lacunas definem a Pesquisa 2.

A pergunta central

Quanto do ganho vem do modelo e quanto vem do pipeline?

V1

Linha de base
com chunking fixo

V2

Segmentação
por regra estrita

V3

Ajuste por modelo
e reparo de saída

As três versões avaliadas sobre exatamente os mesmos cinco modelos.

PESQUISA 2

Protocolo experimental

3	208	5	450
referências curadas	vulnerabilidades de referência	modelos de linguagem	execuções independentes

- Alvos em contêineres Docker, com perfis conhecidos
- Curadoria em duas etapas, com revisão campo a campo
- Dez repetições por combinação, com IC bootstrap de 95%

PESQUISA 2

Bateria de métricas

Alinhamento

Omissão e alucinação por vulnerabilidade

Qualidade por campo

ERM sobre os campos determinísticos

Similaridade textual

BERTScore, ROUGE-L e Token-F1

Severidade

F1 macro ponderado pela cobertura

PESQUISA 2

Resultado: o pipeline explica o ganho

Sem trocar os modelos, em 450 execuções:

37,5% → 90,4%

ERM
(registros exatos)

20,5% → 1,7%

Omissão
por vulnerabilidade

25,1% → 8,3%

Omissão
por campo

A omissão cai cerca de doze vezes. Nenhuma troca de modelo chega perto.

PESQUISA 2

O caso GPT-5 e a convergência

O CASO MAIS EXPRESSIVO

- GPT-5 na V1: omissão de 66% a 81%
- Parâmetros uniformes não davam conta
- Na V3, recuperado por completo

CONVERGÊNCIA ASSIMÉTRICA

- Dispersão entre modelos: 41,8 p.p. → 2,9 p.p.
- Quem mais sofria é quem mais melhora
- Efeito nivelador dos exemplos no prompt

A similaridade da V1 é bimodal: sem medir cobertura, pareceria boa.

O que a Pesquisa 2 firma

- Nenhuma técnica isolada é inédita: o inédito é medir a integração
- A bateria de métricas torna a omissão visível
- Absorvida a variância entre modelos, decide-se por custo e latência
- Limite: apenas 3 referências curadas

Esse limite define a Pesquisa 3.

MulitaMiner: a ferramenta

12.500
linhas de Python em 19 módulos

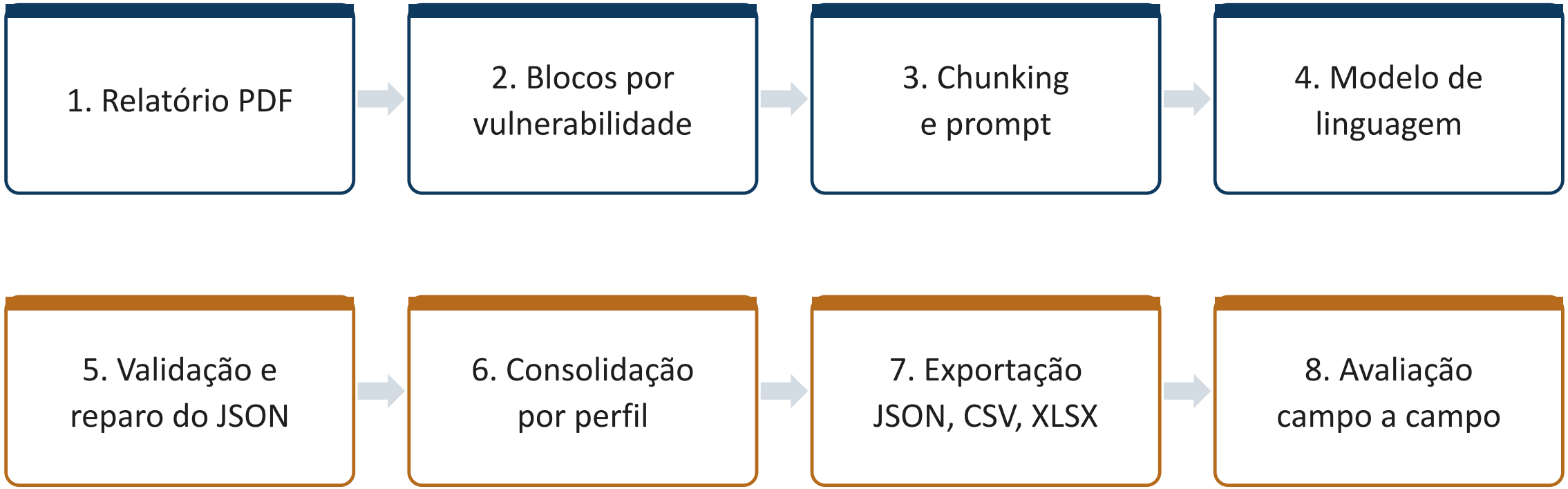
18
campos do esquema canônico

6
perfis de scanner

4
formatos de exportação

- Aplicação de linha de comando, licença MIT
- Configuração declarativa em JSON, sem tocar no código
- Novo modelo ou scanner: basta um arquivo novo

Fluxo de processamento



Falha na extração dispara redivisão adaptativa do bloco.

PESQUISA 3

Avaliação em escala

129
relatórios OpenVAS

6.343
registros de referência

40x
maior que a Pesquisa 2

- Referência objetiva: exportação nativa em CSV do OpenVAS
- Sem curadoria humana, sem viés de denominador pequeno
- Varredura dinâmica de contêineres, em rede isolada

PESQUISA 3

Resultados em escala

67,5% → 78,3%

Correção agregada
(V1 para V3)

14,1% → 2,8%

Omissão
(V1 para V3)

< 1 p.p.

Diferença entre os
cinco modelos na V3

- Na V3 os cinco modelos ficam entre 77,9% e 78,9%
- O erro residual está nos campos interpretativos

PESQUISA 3

Custo de execução

119 mi
tokens no lote da V3

US$ 73,82
custo do lote completo

US$ 0,032
por PDF no LLaMA 4

US$ 0,342
por PDF no GPT-5

CONFIGURAÇÃO RECOMENDADA

DeepSeek: 78,3% de correção, 2,7% de omissão, US$ 0,035 por PDF.

PARTE 3

Síntese e conclusões

A tese defendida, os artefatos e o que permanece em aberto

SÍNTESE

As três pesquisas em uma tabela

Etapa	Natureza	Escala	Efeito
Pesquisa 1	Viabilidade	1 relatório 34 vulnerabilidades	Esquema unificado e modos de falha
Pesquisa 2	Evidência	450 execuções 5 modelos	ERM de 37,5% para 90,4%
Pesquisa 3	Ferramenta	129 relatórios 6.343 achados	Ferramenta aberta e padrão confirmado

Cada mudança de protocolo nasceu de uma limitação concreta da etapa anterior.

Artefatos e reprodutibilidade

- Código do MulitaMiner sob licença MIT
- As três versões do pipeline preservadas em repositórios distintos
- Perfis declarativos de seis scanners e dos modelos avaliados
- Referências curadas (208 vulnerabilidades) e em escala (6.343 achados)

https://mulitaminer.github.io

Limitações e ameaças à validade

- Referência curada única, sem estatística entre anotadores
- Apenas três alvos curados, o que limita a generalização
- A referência em escala penaliza campos interpretativos
- Avaliação restrita a modelos acessados por API

Recomendações práticas, em ordem

1. Investir na segmentação antes do modelo
2. Ajustar tokens e tokenizer por modelo
3. Ancorar o prompt em um esquema explícito
4. Medir cobertura, e não apenas similaridade
5. Só então escolher o modelo, por custo e latência

A tese defendida

Engenharia de pipeline vale mais do que trocar de modelo.

- O gargalo nunca esteve na capacidade dos modelos
- ERM de 37,5% para 90,4%, omissão de 20,5% para 1,7%
- Confirmado em 129 relatórios por menos de US$ 75
- O princípio não depende do domínio

Trabalhos futuros e produção científica

TRABALHOS FUTUROS

- Escalar para Tenable WAS, Nessus e Qualys
- Ampliar as referências curadas
- Executar com provedores locais
- Quarta versão, com reextração por campo

PRODUÇÃO CIENTÍFICA

Pesquisa 1 | ERRC 2025

Structured Extraction of Vulnerabilities in OpenVAS and Tenable WAS Reports Using LLMs

Pesquisa 2 | SBSeg 2026 (submetido)

A Multi-Version Evaluation of LLM-Based Vulnerability Report Extraction

Pesquisa 3 | SBSeg 2026, código aberto

An LLM-Based Tool for Structuring Vulnerability Scanner Reports

Obrigado

Engenharia de pipeline vale mais do que trocar de modelo.

Douglas Lautert | Orientador: Prof. Dr. Diego Kreutz

PPGES, Universidade Federal do Pampa

https://mulitaminer.github.io