DEFESA DE TCC I · 01.07.2026

MulitaMiner

Extração estruturada de vulnerabilidades baseada em LLMs

Um extrator que transforma relatórios de scanners em PDF em dados estruturados e unificados.

```
NVT: Tiki Wiki CMS Groupware Input
Sanitation Weakness Vulnerability
Severity ....... Medium (CVSS: 5.0)
Port .......... 80
Solution type: VendorFix, upgrade
to version 2.2 or later.
```

↓

```
{
"name": "Tiki Wiki CMS Groupware…",
"severity": "MEDIUM",
"cvss": 5.0, "port": 80,
"solution": ["Upgrade to 2.2…"]
}
```

Beatriz Machado · Orientador: Diego Kreutz · Ciência da Computação, UNIPAMPA

O NÚCLEO DO PROBLEMA

Relatórios PDF históricos, a raiz do problema

01

Acúmulo histórico

Times de segurança guardam relatórios PDF de cada scan (OpenVAS, Tenable WAS) ao longo do tempo.

02

Revisão complicada

Depois é preciso revisar o que já foi corrigido e o que ainda não, vasculhando documento por documento.

03

Triagem atrasada

Sendo PDF não estruturado, consultar é lento, e a triagem e a correção das vulnerabilidades atrasam.

+500 mil vulnerabilidades permaneceram sem correção em 2017, segundo o SGIS. O volume de dados de segurança cresce a cada ano, e boa parte fica presa em PDF.

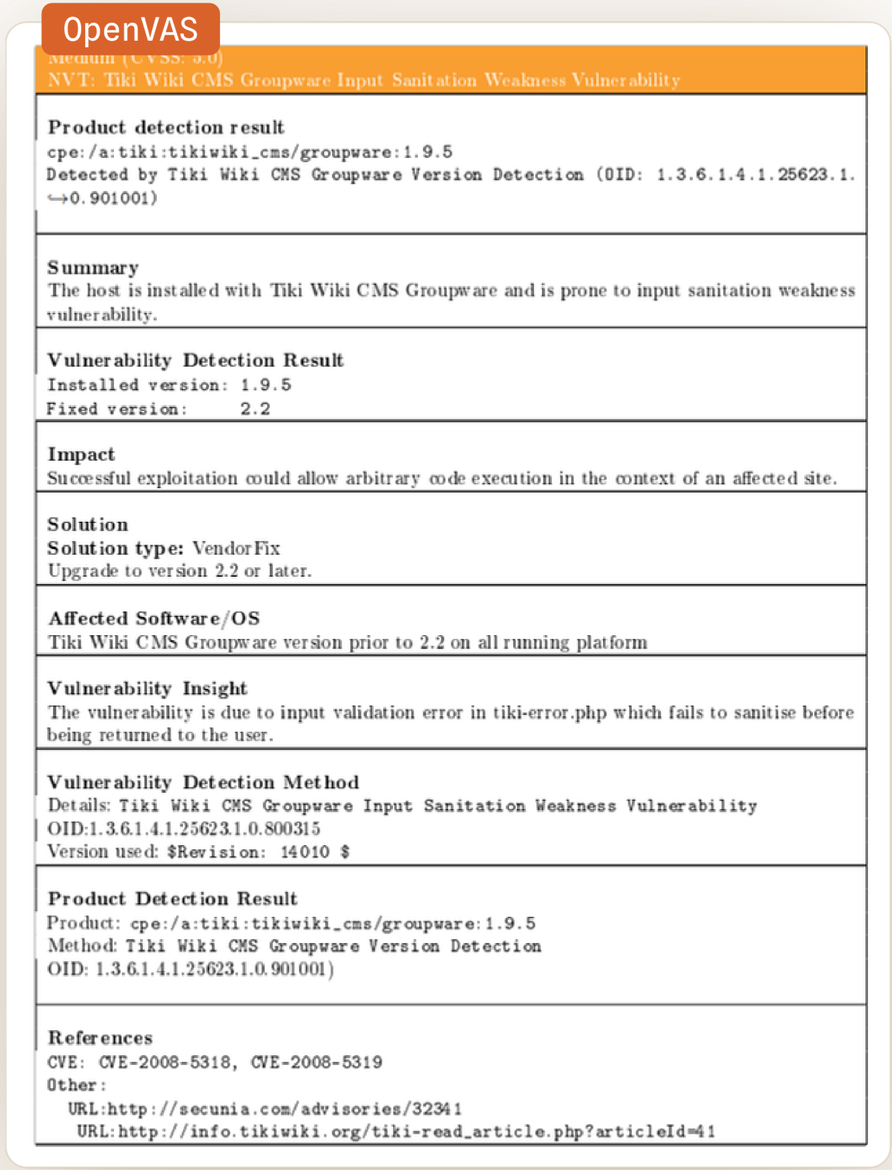
Fonte: RNP / SGIS, 2018 · Relatório de Segurança da RNP, 2024

PROBLEMA 1 · RELATÓRIOS EM PDF

Anatomia de um relatório

Um **scanner de vulnerabilidades** varre um sistema e gera um relatório, quase sempre exportado em PDF, com cada falha encontrada e seus detalhes.

A avaliação deste trabalho usou um único relatório OpenVAS: **OWASP Juice Shop**, com **34 vulnerabilidades**.

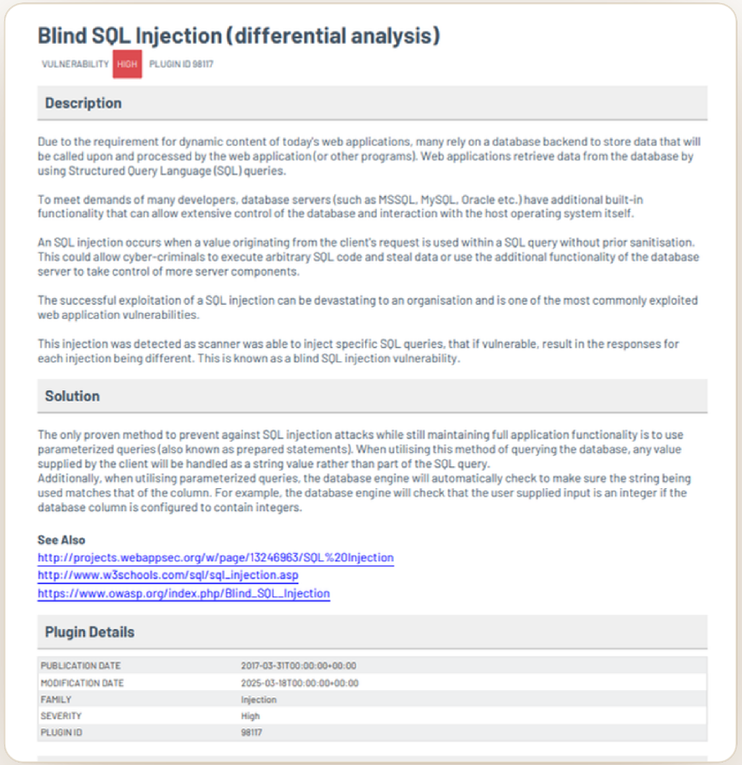


PROBLEMA 2 · FORMATOS HETEROGÊNEOS

Cada scanner, um relatório diferente

O mesmo tipo de achado aparece com layout, nomes de campo e estrutura distintos em cada ferramenta.

Tenable WAS · usada pela RNP



Blind SQL Injection (differential analysis)

VULNERABILITY HIGH PLUGIN ID 98117

Description

Due to the requirement for dynamic content of today's web applications, many rely on a database backend to store data that will be called upon and processed by the web application (or other programs). Web applications retrieve data from the database by using Structured Query Language (SQL) queries.

To meet demands of many developers, database servers (such as MSSQL, MySQL, Oracle etc.) have additional built-in functionality that can allow extensive control of the database and interaction with the host operating system itself.

An SQL injection occurs when a value originating from the client's request is used within a SQL query without prior sanitisation. This could allow cyber-criminals to execute arbitrary SQL code and steal data or use the additional functionality of the database server to take control of more server components.

The successful exploitation of a SQL injection can be devastating to an organisation and is one of the most commonly exploited web application vulnerabilities.

This injection was detected as scanner was able to inject specific SQL queries, that if vulnerable, result in the responses for each injection being different. This is known as a blind SQL injection vulnerability.

Solution

The only proven method to prevent against SQL injection attacks while still maintaining full application functionality is to use parameterized queries (also known as prepared statements). When utilising this method of querying the database, any value supplied by the client will be handled as a string value rather than part of the SQL query.
Additionally, when utilising parameterized queries, the database engine will automatically check to make sure the string being used matches that of the column. For example, the database engine will check that the user supplied input is an integer if the database column is configured to contain integers.

See Also

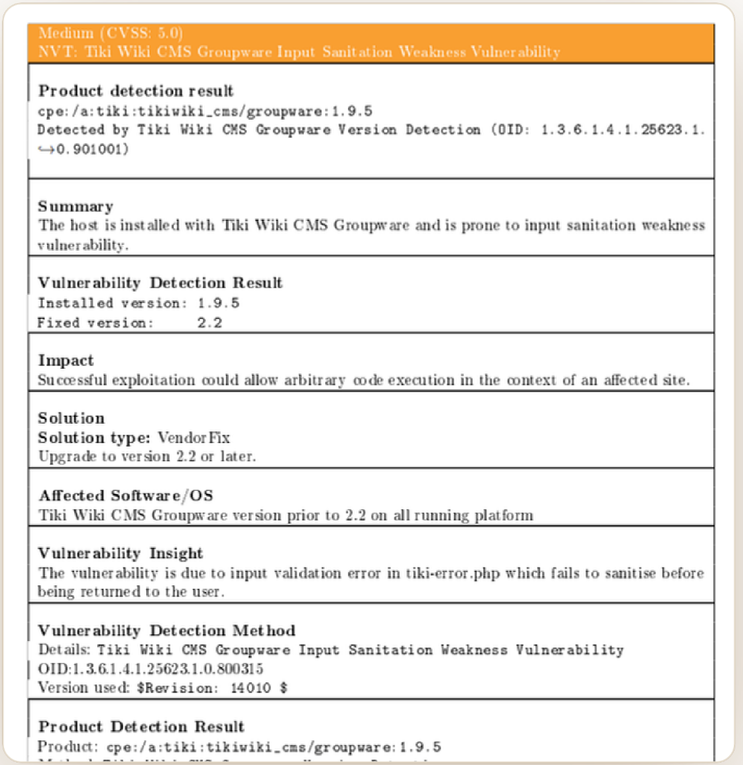
http://projects.webappsec.org/w/page/13246963/SQL%20Injection
http://www.w3schools.com/sql/sql_injection.asp
https://www.owasp.org/index.php/Blind_SQL_Injection

Plugin Details

PUBLICATION DATE	2017-03-31T00:00:00+00:00
MODIFICATION DATE	2025-03-18T00:00:00+00:00
FAMILY	Injection
SEVERITY	High
PLUGIN ID	98117

≠

OpenVAS · open source



Medium (CVSS: 5.0)
NVT: Tiki Wiki CMS Groupware Input Sanitation Weakness Vulnerability

Product detection result
cpe:/a:tiki:tikiwiki_cms/groupware:1.9.5
Detected by Tiki Wiki CMS Groupware Version Detection (OID: 1.3.6.1.4.1.25623.1.
↪0.901001)

Summary
The host is installed with Tiki Wiki CMS Groupware and is prone to input sanitation weakness vulnerability.

Vulnerability Detection Result
Installed version: 1.9.5
Fixed version: 2.2

Impact
Successful exploitation could allow arbitrary code execution in the context of an affected site.

Solution
Solution type: VendorFix
Upgrade to version 2.2 or later.

Affected Software/OS
Tiki Wiki CMS Groupware version prior to 2.2 on all running platform

Vulnerability Insight
The vulnerability is due to input validation error in tiki-error.php which fails to sanitise before being returned to the user.

Vulnerability Detection Method
Details: Tiki Wiki CMS Groupware Input Sanitation Weakness Vulnerability
OID:1.3.6.1.4.1.25623.1.0.800315
Version used: $Revision: 14010 $

Product Detection Result
Product: cpe:/a:tiki:tikiwiki_cms/groupware:1.9.5

OS DESAFIOS DA UNIFICAÇÃO

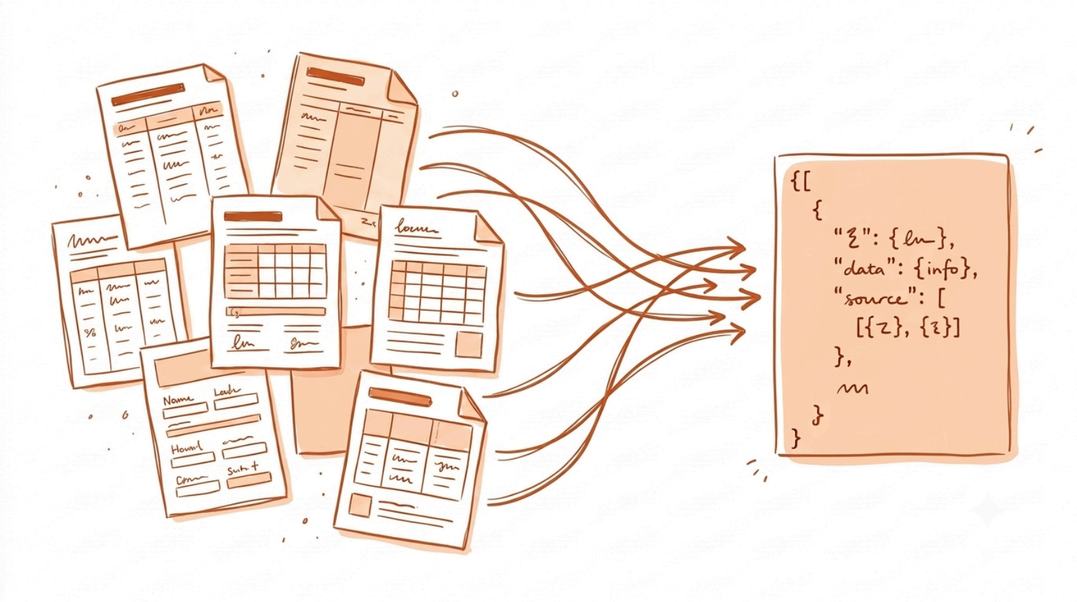
Juntar não basta: é preciso reconciliar

A **Nomes diferentes**
“NVT”, “Vulnerability”, o mesmo conceito com rótulos distintos.

B **Estruturas distintas**
Tabelas, listas e texto corrido convivem sem um padrão comum.

C **Granularidades diferentes**
Um detalha solução por etapas; outro resume em uma linha.

D **Campos ausentes**
Nem todo relatório traz porta, CVE ou instâncias afetadas.



A SOLUÇÃO · ESQUEMA UNIFICADO

Um modelo de dados único

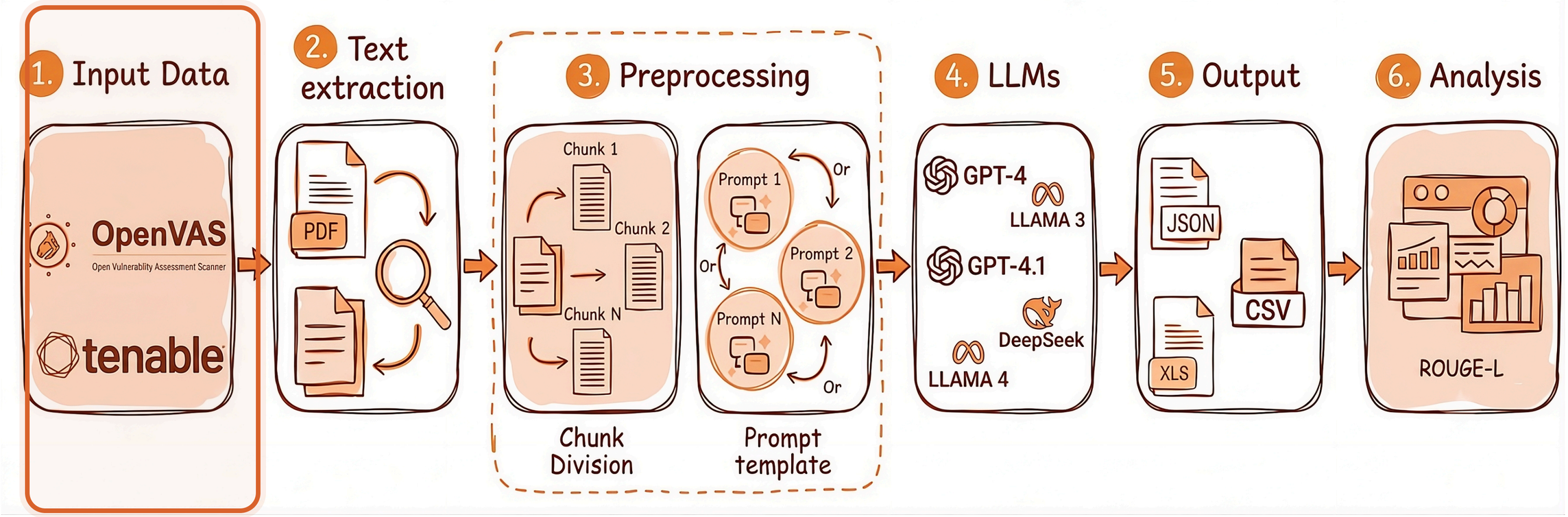
Uma saída **JSON unificada** que coloca relatórios de scanners diferentes sob o mesmo modelo de dados.

Assim é possível cruzar achados de vários scanners sobre o mesmo alvo, ampliando a cobertura e gerando datasets mais ricos para análise, correlação e pesquisa.


```
{
  "Name": "Tiki Wiki CMS Groupware Input …",
  "description": [ "The host is installed with…" ],
  "impact": [ "Successful exploitation could…" ],
  "solution": [ "Solution type: VendorFix",
        "Upgrade to version 2.2 or later." ],
  "cvss": 5.0,  "port": 80,
  "severity": "MEDIUM",
  "plugin": null, "plugin_details": {},
  "instances": []
}
```

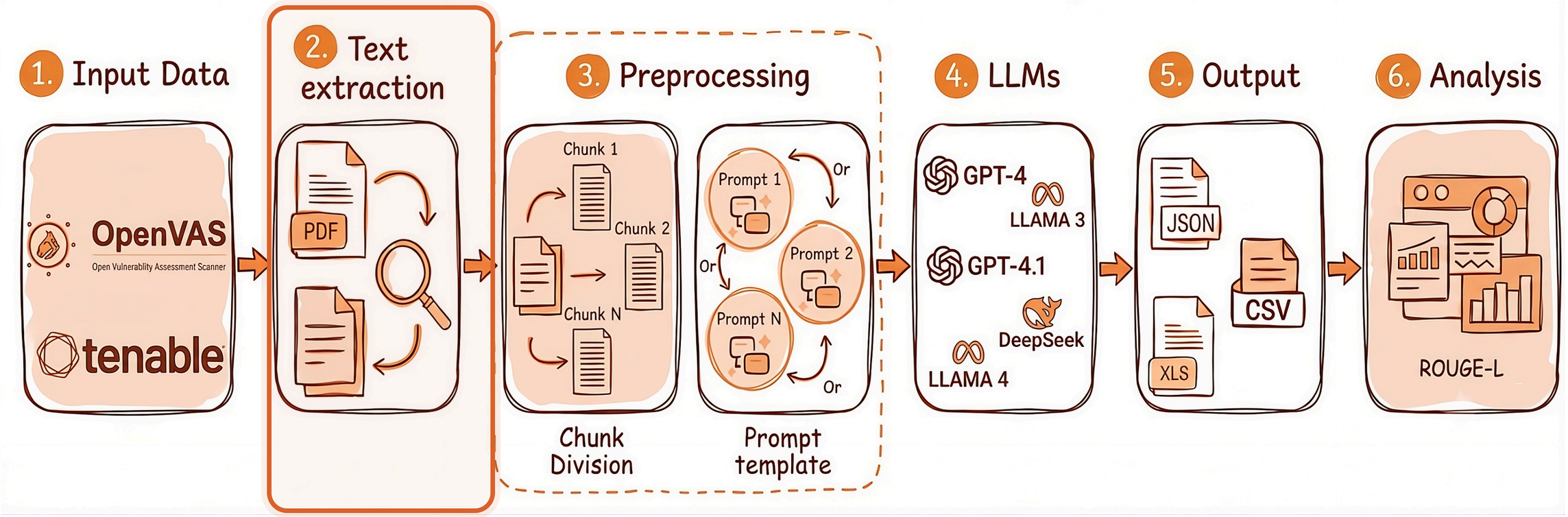
1 Entrada PDF

O relatório do scanner entra como PDF, vindo do OpenVAS ou do Tenable WAS.



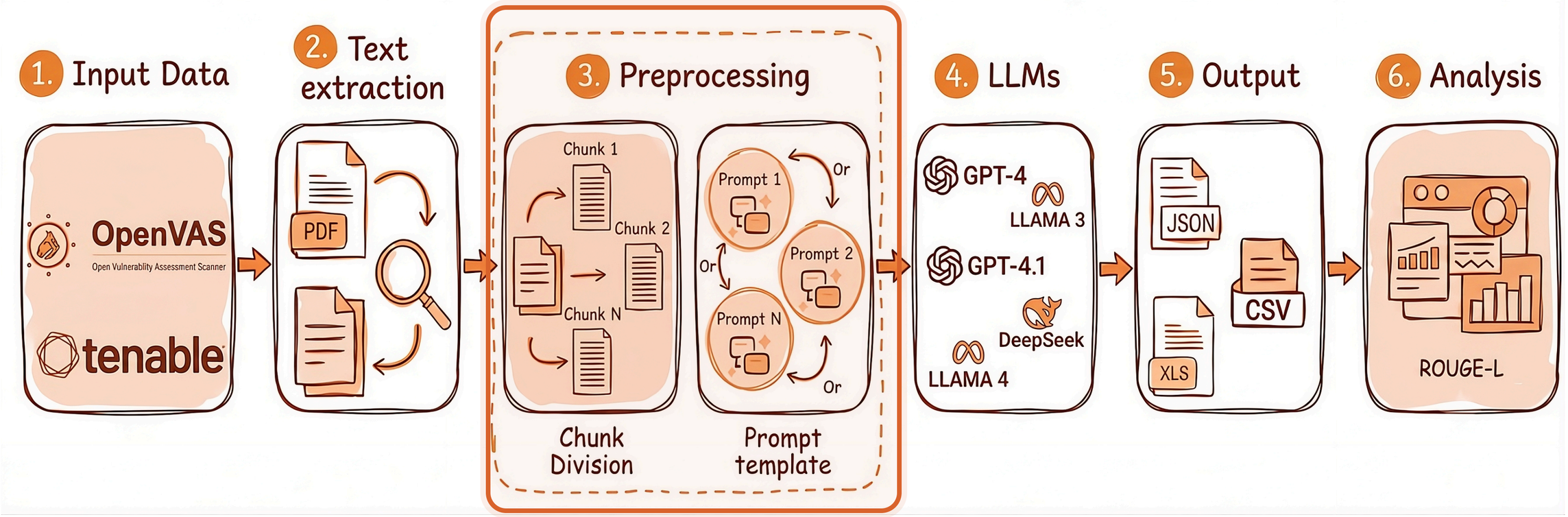
2 Extração do texto

O PDF é convertido em texto puro (TXT), deixando para trás layout e formatação.



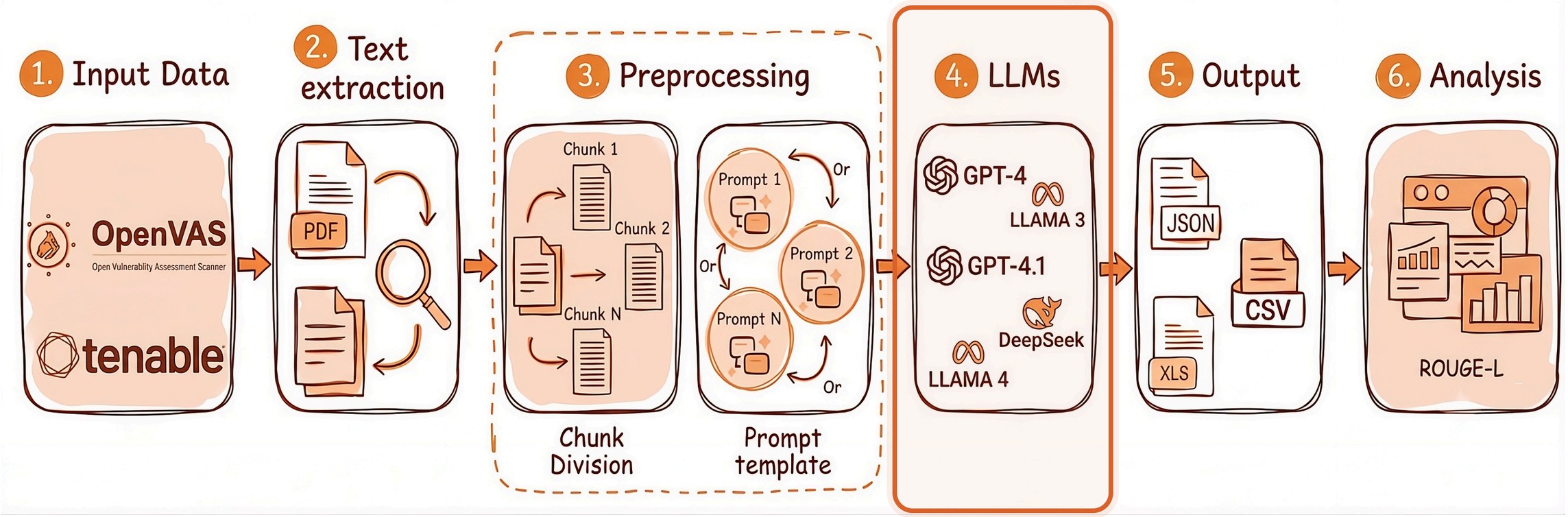
3 Pré-processamento

O texto é dividido em chunks e cada um recebe o prompt de extração. É aqui que a precisão mais sofre.



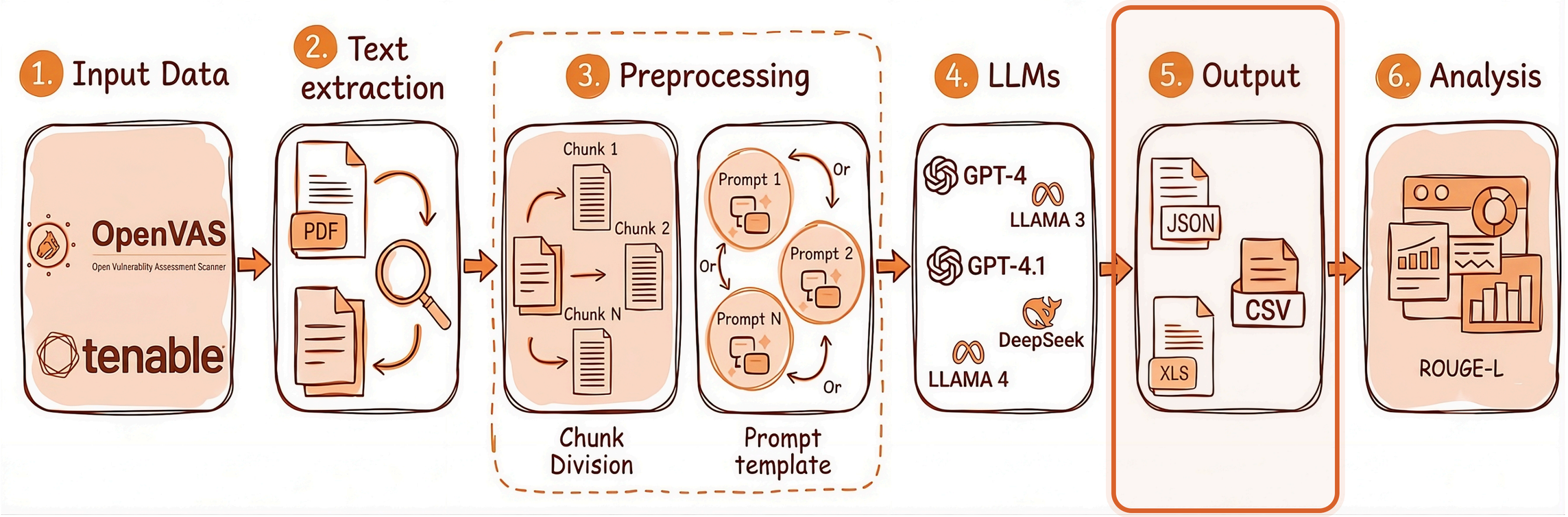
4 Envio à LLM

Cada chunk, com seu prompt, é enviado à LLM: GPT-4, GPT-4.1, DeepSeek e LLAMA.



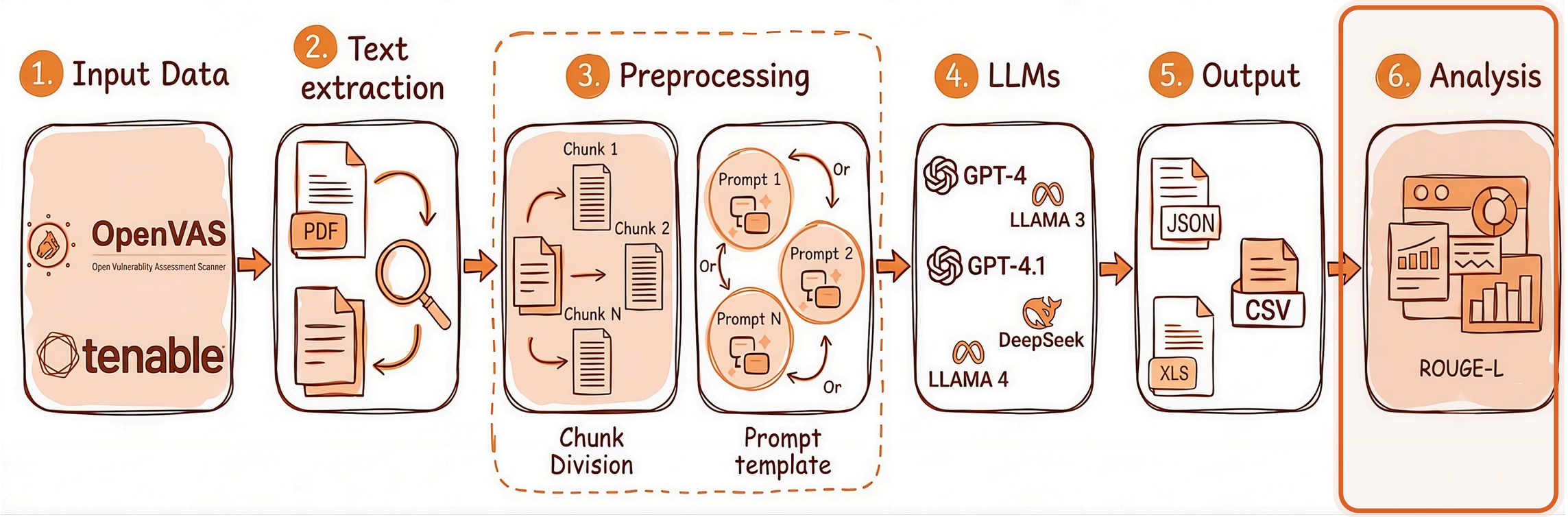
5 Saída estruturada

A LLM devolve os dados já organizados em JSON, CSV ou XLS.



6 Análise vs. baseline

A saída é comparada ao gabarito usando a métrica ROUGE-L.



ETAPA 3 · O PROMPT

O prompt enviado à LLM

Cada chunk segue para a LLM com um prompt que define o que extrair e em que formato.

- Saída apenas em **JSON válido**, sem texto extra
- Campos obrigatórios por vulnerabilidade
- Regras de extração por seção (Summary, Impact, Solution...)

Prompt completo:
github.com/AnonShield/Vulnerability_Extractor

```
// prompt enviado com cada chunk

You are an OpenVAS extraction
specialist that ONLY outputs JSON.

✓  Return ONLY a valid JSON array
✓  Numbers without quotes: 443, 7.5
✗  NO markdown, NO extra text

STEP 1 · blocos "NVT:"
STEP 2 · severidade · porta · CVSS
STEP 3 · extrair por seção
STEP 4 · montar objeto JSON
```

AVALIAÇÃO DA EXTRAÇÃO

Quão perto do ideal?

O MÉTODO

Baseline = o gabarito

Uma extração ideal feita manualmente: o "100% como deveria estar". Toda saída da LLM é comparada contra ela.

Métrica: ROUGE-L

Similaridade lexical pela maior subsequência comum entre a extração e o gabarito. Boa para os campos **descritivos**, como descrição, impacto e solução.

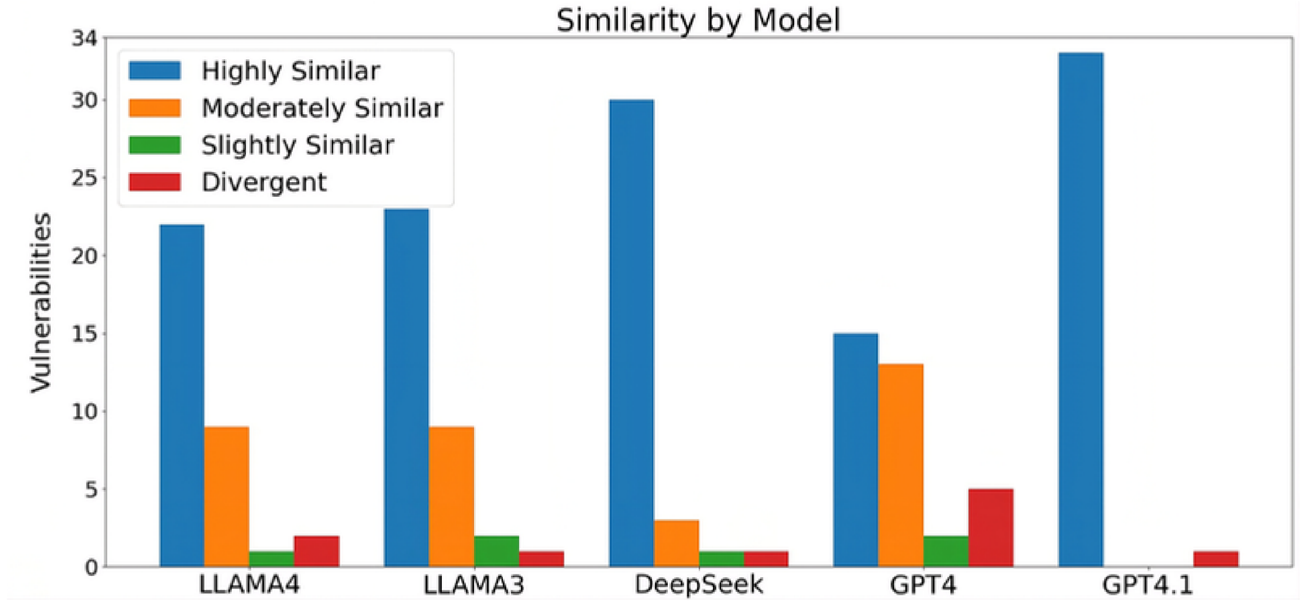
FAIXAS DE SIMILARIDADE

> 0.7	Altamente similar
≤ 0.7	Moderadamente similar
≤ 0.6	Levemente similar
≤ 0.4	Divergente

As mesmas cores aparecem no gráfico de resultados.

RESULTADOS

Similaridade por modelo



GPT-4.1

Mais extrações **altamente similares** e poucos divergentes.

DeepSeek

Desempenho forte e consistente, logo atrás.

GPT-4

Categorias com números muito próximos entre si, e a faixa **altamente similar** bem abaixo dos demais modelos.

OS PROBLEMAS COM LLMS

O corte cego quebra o contexto

1 — Segmentação de tamanho fixo

O corte seguia um limite de **9.000 caracteres** e ignorava as delimitações de cada vulnerabilidade.

2 — Retry com corte cego

Ao falhar, o bloco era dividido ao meio na quebra de linha, cortando de novo sem respeitar a vulnerabilidade.

! — Resultado

Contexto incompleto leva o modelo a **omitir campos ou alucinar**.

LIMITAÇÕES · 01

Validado com um só scanner

A proposta é unificar relatórios PDF de **muitos scanners diferentes**, que existem aos montes no mercado.

Mas a validação cobriu apenas um deles, o **OpenVAS**, em um único relatório pequeno. É pouco para garantir que o método se sustenta com outros scanners ou relatórios mais densos.

representa	OpenVAS	Tenable WAS
nome	NVT	Vulnerability
descrição	Summary	Description
referências	References	See Also
gravidade (CVSS)	Severity	Risk Information
SEM EQUIVALENTE DIRETO		
exclusivos	Impact Insight Detection Method	Plugin ID Plugin Details

LIMITAÇÕES · 02

Uma só métrica não basta

A avaliação usa apenas **ROUGE-L**, que mede similaridade lexical.

Isso atende os campos de texto, mas não os campos que têm uma resposta certa.

Campos descritivos `descrição · impacto · solução`

ROUGE-L faz sentido: importa o quanto o texto se aproxima do esperado.

Campos determinísticos `severidade · cvss · porta`

Têm resposta única: ou acerta, ou erra. Aqui, **precision, recall e F1-score** seriam mais adequados.

LIMITAÇÕES · 03

Só modelos em nuvem

Toda a avaliação foi feita com LLMs em nuvem (GPT-4.1, GPT-4, DeepSeek, LLAMA). Modelos locais não foram avaliados, o que deixa três questões em aberto:

A

Custo

Cada execução depende de APIs pagas por volume de dados.

Avaliar muitos relatórios fica caro em escala.

B

Privacidade

Dados de segurança sensíveis saem para servidores de terceiros.

Crítico para relatórios de vulnerabilidades.

C

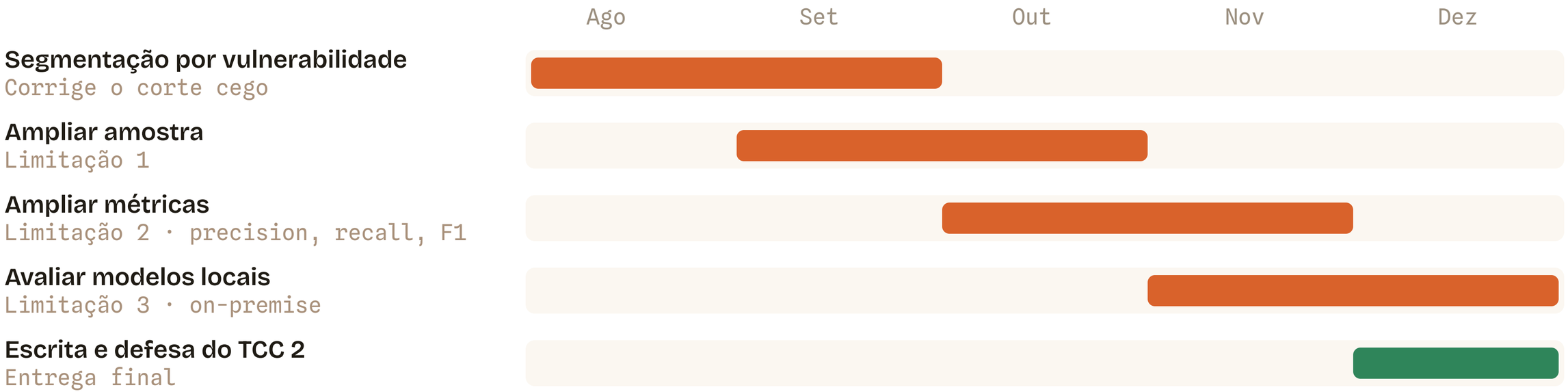
Reprodutibilidade

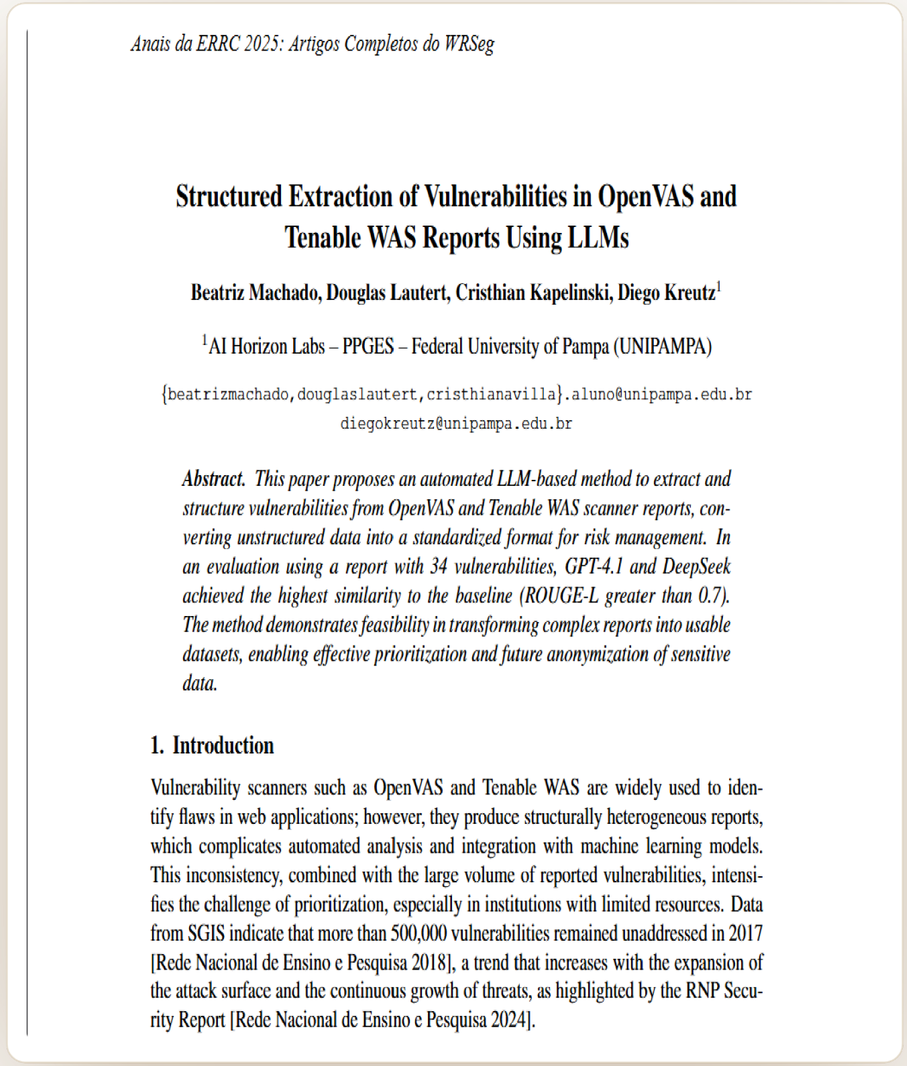
Modelos hospedados mudam de versão, e alguns são descontinuados, ao longo do tempo.

Os resultados podem não se repetir.

PRÓXIMOS PASSOS · TCC 2

Cronograma previsto





Anais da ERRC 2025: Artigos Completos do WRSeg

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

Beatriz Machado, Douglas Lautert, Cristhian Kapelinski, Diego Kreutz[1]

[1]AI Horizon Labs – PPGES – Federal University of Pampa (UNIPAMPA)

{beatrizmachado,douglaslautert,cristhianavilla}.aluno@unipampa.edu.br
diegokreutz@unipampa.edu.br

Abstract. *This paper proposes an automated LLM-based method to extract and structure vulnerabilities from OpenVAS and Tenable WAS scanner reports, converting unstructured data into a standardized format for risk management. In an evaluation using a report with 34 vulnerabilities, GPT-4.1 and DeepSeek achieved the highest similarity to the baseline (ROUGE-L greater than 0.7). The method demonstrates feasibility in transforming complex reports into usable datasets, enabling effective prioritization and future anonymization of sensitive data.*

1. Introduction

Vulnerability scanners such as OpenVAS and Tenable WAS are widely used to identify flaws in web applications; however, they produce structurally heterogeneous reports, which complicates automated analysis and integration with machine learning models. This inconsistency, combined with the large volume of reported vulnerabilities, intensifies the challenge of prioritization, especially in institutions with limited resources. Data from SGIS indicate that more than 500,000 vulnerabilities remained unaddressed in 2017 [Rede Nacional de Ensino e Pesquisa 2018], a trend that increases with the expansion of the attack surface and the continuous growth of threats, as highlighted by the RNP Security Report [Rede Nacional de Ensino e Pesquisa 2024].

PUBLICAÇÃO

2º Melhor Artigo · ERRC 2025

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

Beatriz Machado, Douglas Lautert, Cristhian Kapelinski, Diego Kreutz

ERRC 2025 · WRSeg · Sociedade Brasileira de Computação (SBC)

artigo sol.sbc.org.br/errc/article/39195

ferramenta mulitaminer.github.io

MULITAMINER

Obrigada!

Beatriz Machado · Orientador: Diego Kreutz
Ciência da Computação · UNIPAMPA