Hiago Rafael de Souza

Sistema de Busca e Enriquecimento Automático de Write-ups de Capture The Flag

São Carlos - SP
2025

Hiago Rafael de Souza

Sistema de Busca e Enriquecimento Automático de Write-ups de Capture The Flag

Trabalho de Conclusão de Curso apresentado ao Departamento de Computação da Universidade Federal de São Carlos, como parte dos requisitos para a conclusão da graduação em Engenharia de Computação.

Orientação: Prof. Dr. Paulo Matias

Universidade Federal de São Carlos – UFSCar
Centro de Ciências Exatas e de Tecnologia – CCET
Departamento de Computação – DC
Trabalho de Conclusão de Curso – TCC

São Carlos - SP
2025

Dedico este trabalho à minha família,
formadores do meu caráter e encorajadores do meu sucesso.

Agradecimentos

Agradeço à minha família, amigos e todos que fizeram parte da minha jornada.

"A força de um povo está na coragem com que defende seus princípios."

Resumo

A segurança da informação tem se consolidado como uma das áreas mais relevantes da computação, impulsionada pelo aumento constante de ataques cibernéticos e pela necessidade de medidas preventivas e corretivas. Nesse contexto, as competições *Capture The Flag* (CTF) destacam-se como ambientes de aprendizagem prática, permitindo que participantes desenvolvam competências técnicas ao explorar vulnerabilidades em sistemas e aplicar técnicas de criptografia, engenharia reversa e análise forense. Um dos principais produtos dessas competições são os *write-ups*, relatórios que descrevem o processo de resolução dos desafios e que funcionam como valioso material de estudo e disseminação de conhecimento.

Contudo, tais documentos encontram-se dispersos na internet, frequentemente incompletos, despadronizados ou hospedados em domínios temporários, o que dificulta seu aproveitamento acadêmico e profissional. Diante desse cenário, este trabalho propõe a criação de uma plataforma de automação para busca, centralização e apresentação de *write-ups* de CTF, integrando técnicas de *scraping*, uso de APIs oficiais, enriquecimento automático de dados e rotulação com inteligência artificial, além de armazenamento estruturado em banco de dados. Espera-se, com isso, oferecer uma solução que facilite o acesso e a consulta a esse tipo de conteúdo, contribuindo tanto para a formação de estudantes quanto para a prática profissional na área de cibersegurança.

Palavras-chave: Segurança da Informação. *Capture The Flag*. *Write-ups*. Inteligência Artificial. Automação.

Abstract

Information security has become one of the most relevant areas of computing, driven by the constant increase in cyberattacks and the need for both preventive and corrective measures. In this context, Capture The Flag (CTF) competitions stand out as practical learning environments, enabling participants to develop technical skills by exploring system vulnerabilities and applying techniques such as cryptography, reverse engineering, and forensic analysis. One of the main outcomes of these competitions are the write-ups, reports that describe the process of solving challenges and serve as valuable study material and knowledge dissemination tools.

However, these documents are scattered across the internet, often incomplete, unstandardized, or hosted on temporary domains, which hinders their academic and professional use. To address this issue, this work proposes the creation of an automated platform for the collection, centralization, and presentation of CTF write-ups, integrating scraping techniques, official APIs, automatic data enrichment, and labeling with artificial intelligence, as well as structured database storage. The goal is to provide a solution that facilitates access and consultation of this type of content, contributing both to student training and to professional practice in the cybersecurity field.

Keywords: Information Security. Capture The Flag. Write-ups. Artificial Intelligence. Automation.

Lista de abreviaturas e siglas

API	Application Programming Interface
AWS	Amazon Web Services
CTF	Capture The Flag
DOM	Document Object Model
ETL	Extract, Transform and Load
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IA	Inteligência Artificial
JSON	JavaScript Object Notation
NLP	Natural Language Processing
S3	Amazon Simple Storage Service
SQL	Structured Query Language
TLS	Transport Layer Security
URL	Uniform Resource Locator
VPS	Virtual Private Server
XML	Extensible Markup Language

Sumário

1 Introdução

Com a constante evolução da computação e o surgimento de novas ferramentas capazes de auxiliar o trabalho contemporâneo, novas tecnologias são desenvolvidas para suprir as necessidades dos mais diversos mercados. Nesse cenário, a segurança da informação tem se consolidado como uma das áreas mais importantes da computação, tanto de forma preventiva quanto corretiva, devido ao aumento constante de ataques cibernéticos e à crescente complexidade das ameaças digitais (1, 2).

Dentro desse contexto, principalmente acadêmico, as competições *Capture The Flag* (CTF) surgem como ambientes que simulam cenários reais de ataque e defesa. Elas permitem que os participantes aprimorem suas competências técnicas em segurança da informação, explorando vulnerabilidades em sistemas e aplicando técnicas de criptografia, engenharia reversa e análise forense (3, 4).

Diversas iniciativas globais utilizam competições CTF como ferramenta pedagógica estruturada. Um dos exemplos mais influentes é o *picoCTF*, uma competição educacional desenvolvida pela Carnegie Mellon University, amplamente reconhecida por seu impacto no ensino de segurança computacional e pela produção de *write-ups* acessíveis a iniciantes (5). O sucesso desse modelo reforça o valor dos CTFs enquanto ambientes de aprendizagem ativa e motivadora, capazes de promover retenção e aprofundamento conceitual (3).

Para fins de estudo e disseminação de conhecimento, uma das principais fontes de aprendizado sobre esses desafios são os *write-ups*, relatórios produzidos por participantes após a resolução das tarefas. Esses documentos descrevem as técnicas aplicadas, o raciocínio empregado e a exploração realizada, tornando-se valioso material didático tanto para estudantes quanto para profissionais da área (6, 5).

Entretanto, muitos *write-ups* encontram-se dispersos pela internet, muitas vezes incompletos, despadronizados ou hospedados em plataformas temporárias, o que dificulta o acesso e compromete seu uso acadêmico e profissional. A ausência de organização, de metadados estruturados e de mecanismos de rastreabilidade também limita pesquisas, análises comparativas e a preservação histórica de conteúdo técnico.

Trabalhos anteriores já destacaram dificuldades semelhantes, inclusive no contexto de infraestrutura para competições de segurança (7, 6), reforçando a necessidade de soluções que centralizem e gerenciem dados de forma automatizada.

Diante desse cenário, este trabalho propõe a criação de uma plataforma de automação para busca, centralização e apresentação de *write-ups* de CTFs. A solução combina técnicas de *scraping*, uso de APIs oficiais, enriquecimento automático de conteúdo, rotulação via inteligência artificial e armazenamento estruturado em banco de dados, permitindo consultas avançadas, análise semântica e acesso unificado a materiais antes dispersos.

Assim, espera-se não apenas facilitar o acesso a esses materiais, mas também contribuir

para a consolidação de uma ferramenta útil à comunidade acadêmica e profissional da segurança da informação, ampliando a disponibilidade, preservação e reutilização de conteúdo técnico relevante.

1.1 Uso de Inteligência Artificial

A elaboração deste trabalho contou com o auxílio de ferramentas de inteligência artificial generativa como suporte instrumental. Utilizaram-se o modelo Google Gemini 3 Pro e a plataforma de agente Jules, especificamente para: (1) auxílio no levantamento bibliográfico e (2) revisão ortográfica, gramatical e de estilo.

É imperativo ressaltar que todo o conteúdo gerado por essas ferramentas foi submetido a uma rigorosa validação humana, assegurando a precisão factual e a aderência à literatura especializada. A concepção intelectual, a análise crítica, a síntese dos dados e a redação final permanecem de inteira responsabilidade do autor, que validou integralmente o material produzido.

Ressalta-se que o uso dessas ferramentas de apoio à redação difere da aplicação de inteligência artificial como componente técnico da plataforma desenvolvida neste trabalho. Os modelos e técnicas integrados à solução (CTF Bridge) são descritos detalhadamente no Capítulo 4 (Desenvolvimento).

2 Objetivos

2.1 Objetivo Geral

Desenvolver uma plataforma de automação para coleta, centralização e rotulação de *write-ups* de competições *Capture The Flag* (CTF), utilizando técnicas de *scraping*, APIs oficiais, enriquecimento de dados com inteligência artificial e armazenamento estruturado em banco de dados, de modo a disponibilizar esse material de forma organizada, acessível e útil para a comunidade acadêmica e profissional da área de segurança da informação.

2.2 Objetivos Específicos

- Implementar mecanismos de coleta de dados por *scraping* e APIs oficiais;
- Realizar enriquecimento automático de *write-ups* com metadados e classificação por inteligência artificial;
- Estruturar o armazenamento em banco de dados de forma a permitir consultas complexas;
- Criar formas de visualização e pesquisa que favoreçam o uso acadêmico e profissional.

2.3 Resultados Esperados

- Disponibilizar uma plataforma funcional capaz de coletar automaticamente *write-ups* de CTF por meio de *scraping* e uso de APIs oficiais;
- Garantir a centralização dos documentos em um banco de dados estruturado, permitindo consultas complexas e filtragem por metadados;
- Realizar o enriquecimento automático de conteúdo com apoio de inteligência artificial, incluindo resumo, classificação por dificuldade e categorização temática;
- Fornecer uma interface de pesquisa que facilite o acesso a estudantes e profissionais, com recursos de busca e organização de resultados;
- Contribuir para a disseminação de conhecimento em segurança da informação, fortalecendo o uso acadêmico e prático de *write-ups* como material de estudo;
- Criar uma base sólida para futuras extensões, como integração com ferramentas de busca avançada (por exemplo, OpenSearch) e ampliação para repositórios adicionais.

3 Revisão Bibliográfica

A segurança da informação consolidou-se como um pilar fundamental da infraestrutura tecnológica moderna, exigindo profissionais capazes de projetar sistemas resilientes e responder a incidentes complexos (1). Nesse cenário, a evolução constante das ameaças cibernéticas demanda métodos de ensino que transcendam a teoria, preparando estudantes para a realidade do cibercrime organizado e da defesa ativa (2).

3.1 Competições de Segurança e o Papel Educacional dos CTFs

A literatura especializada converge ao apontar as competições *Capture The Flag* (CTF) como instrumentos essenciais para a formação prática em segurança da informação. Leune e Petrilli Jr. (3) argumentam que, diferentemente do ensino tradicional focado em conceitos abstratos, os CTFs proporcionam um ambiente de aprendizagem ativa, onde participantes aplicam conhecimentos de criptografia, engenharia reversa e exploração web para resolver problemas concretos. Chapman, Burket e Brumley (5) e Chothia e Novakovic (4) corroboram essa visão, apresentando evidências empíricas de que o uso de desafios práticos aumenta o engajamento e permite avaliar competências complexas.

Este trabalho compartilha da premissa de que os CTFs são vitais para a educação em segurança. No entanto, enquanto os trabalhos citados focam na *criação* e *aplicação* de competições como método de ensino, a presente proposta atua em uma camada posterior: a preservação do conhecimento gerado *após* a competição. Diferente de Chothia e Novakovic (4), que propõem uma máquina virtual offline para ensino, este TCC busca integrar o ecossistema de competições online já existentes, centralizando o aprendizado disperso em uma plataforma acessível.

No cenário nacional, Kondo et al. (7) traçam um panorama histórico das competições de segurança no Brasil, utilizando a trajetória do *Pwn2Win* como estudo de caso. O trabalho, intitulado “Casa de ferreiro, o espeto não é de pau”, documenta como uma competição organizada por brasileiros alcançou reconhecimento internacional, evidenciando a maturidade técnica da comunidade local. A plataforma proposta neste TCC dialoga diretamente com esse contexto, oferecendo uma ferramenta que valoriza e preserva a produção intelectual de competidores brasileiros e internacionais, mitigando o problema da efemeridade das plataformas de competição apontado indiretamente por Kondo.

3.2 O Valor dos Write-ups e o Problema da Dispersão

Um subproduto fundamental dessas competições são os *write-ups* — relatórios técnicos elaborados pelos participantes descrevendo o raciocínio e as técnicas utilizadas para solucionar cada desafio. Esses documentos constituem uma base de conhecimento dinâmica, contendo detalhes sobre vulnerabilidades emergentes que muitas vezes antecipam a literatura formal. Contudo, Werther et al. (6) observaram que grande parte do conhecimento prático gerado durante os eventos se perde ou fica inacessível devido à falta de padronização e à dispersão em blogs pessoais e repositórios temporários.

Além da barreira de entrada, a transitoriedade das plataformas de competição é um desafio crítico. Como observado por Kondo et al. (7) ao discutirem a infraestrutura de eventos, a manutenção de ambientes seguros e persistentes é complexa e exige evolução constante. Consequentemente, grande parte dos enunciados originais e arquivos anexos torna-se inacessível, e o conhecimento prático gerado pelos participantes (os *write-ups*) acaba disperso em blogs pessoais e repositórios temporários, sem padronização.

A lacuna identificada na literatura é a ausência de sistemas capazes de agregar e organizar esses relatórios de forma automática. Enquanto Werther et al. (6) sugerem a importância de capturar esse conhecimento, a maioria das soluções existentes depende de curadoria manual. Este trabalho avança em relação ao estado da arte ao propor uma automação completa do ciclo de coleta e catalogação, eliminando a necessidade de intervenção humana constante e garantindo a escalabilidade do acervo, algo não endereçado pelas iniciativas anteriores de preservação manual.

3.3 Uso de LLMs para Enriquecimento e Normalização de Dados

Para enfrentar o desafio da dispersão e da falta de estrutura dos dados na web, técnicas recentes de inteligência artificial têm se mostrado promissoras. Zhao et al. (8) destacam que os Grandes Modelos de Linguagem (LLMs) demonstram capacidades emergentes de compreensão e geração de texto que permitem novas aplicações em processamento de linguagem natural. Ayuso, Brogya e Ahlawat (9) revisam como essa tecnologia está redefinindo o *web scraping*, superando a rigidez dos *scrapers* tradicionais.

Uma vez coletados, os dados não estruturados necessitam de processamento. Bodor et al. (10) e Cirillo et al. (11) demonstram frameworks onde LLMs não apenas coletam, mas enriquecem e completam dados anonimizados ou não estruturados. A proposta deste TCC se alinha metodologicamente ao framework de Bodor et al. (10), mas inova ao adaptar técnicas de *Extract-Define-Canonicalize* propostas por Zhang e Soh (12) para criar taxonomias automáticas de vulnerabilidades em *write-ups* de segurança (ex: categorizar um texto livre como *Buffer Overflow* ou *SQL Injection*).

Dessa forma, o trabalho preenche uma lacuna específica: a aplicação de IA generativa

moderna para transformar o "caos" de informações técnicas de CTFs em uma base de conhecimento estruturada e semanticamente rica, algo que a literatura de educação em segurança (3) demandava, mas que a tecnologia da época não permitia realizar com a eficiência dos LLMs atuais.

4 Desenvolvimento

4.1 Arquitetura Geral da Solução

Como a plataforma foi desenvolvida com o objetivo de automatizar todas as etapas envolvidas na coleta, tratamento, enriquecimento e disponibilização de *write-ups* de competições de CTF, mostrou-se adequado adotar uma arquitetura modular e escalável, composta por diferentes serviços que operam de maneira integrada, mas com baixo acoplamento. Assim, é possível que cada etapa funcione de forma independente, possibilitando escalabilidade e expansão da coleção de ferramentas, além de facilidade de manutenção e monitoramento.

A solução combina técnicas de *scraping*, uso de APIs oficiais, inteligência artificial para enriquecimento semântico, armazenamento estruturado e um *frontend* dedicado para os usuários finais da aplicação. Toda a infraestrutura foi construída sobre serviços *serverless* da AWS (escolhida tanto devido à afinidade do autor como por vantagens técnicas que serão detalhadas mais adiante), o que garante escalabilidade, baixo custo e execução confiável.

4.1.1 Visão Macro da Plataforma

A arquitetura geral do sistema é formada pelos seguintes módulos e serviços:

- **Módulo de *scraping* (CTFtime):** encarregado de identificar e extrair *write-ups* diretamente do site CTFtime. Esse componente navega pelas páginas de competições, captura informações relevantes e identifica links externos contendo *write-ups* completos;
- **Módulo de enriquecimento com IA (Gemini):** responsável por analisar o conteúdo coletado e gerar metadados estruturados, como resumo, classificação de dificuldade, vulnerabilidades exploradas, técnicas aplicadas e *tags* temáticas. Esse módulo também identifica o idioma do *write-up* e determina se o conteúdo é completo, parcial ou apenas um link;
- ***Workflow* de tradução e rotulação automática:** quando o conteúdo está em idioma diferente de inglês, o sistema ativa um *pipeline* adicional que traduz os principais campos e uniformiza a estrutura semântica do *write-up*. Também identifica automaticamente categorias técnicas e padrões frequentes;

- **Banco de dados DynamoDB:** armazena de maneira estruturada os *write-ups* enriquecidos, permitindo consultas rápidas e flexíveis com base em diferentes dimensões, como dificuldade, categoria, evento, data ou time;
- **Armazenamento bruto (S3):** mantém cópias integrais dos conteúdos coletados, incluindo HTML original, conteúdo recursivo extraído de páginas externas e os JSONs gerados durante o processo de enriquecimento. Isso preserva o histórico e permite reprocessamento quando necessário, além de proporcionar um sistema íntegro para todo o processo, sem ter que acessar banco de dados com conteúdos não completos ou não tratados;
- **API Gateway:** expõe *endpoints* que possibilitam ao *frontend* consultar *write-ups*, metadados e estatísticas, servindo como camada intermediária entre a aplicação web e o banco de dados;
- ***Frontend* em React (CTF Bridge):** interface web desenvolvida em React para apresentar os *write-ups* de forma organizada. Foi desenvolvida pensando no usuário final. Permite navegação, filtragem, visualização de resumos, categorias técnicas, eventos, times participantes e links para o conteúdo original. Tudo de forma automática e integrada a todo processo, com apenas poucos segundos de atraso entre o início da integração e seu acesso final.

O fluxo de dados entre esses componentes segue uma sequência lógica e totalmente automatizada:

Busca → Normalização → Enriquecimento → Tradução (quando necessário) → Persistência → Disponibilização ao usuário final.

A Figura 1 ilustra a arquitetura da solução, destacando os componentes e o fluxo de dados entre eles.

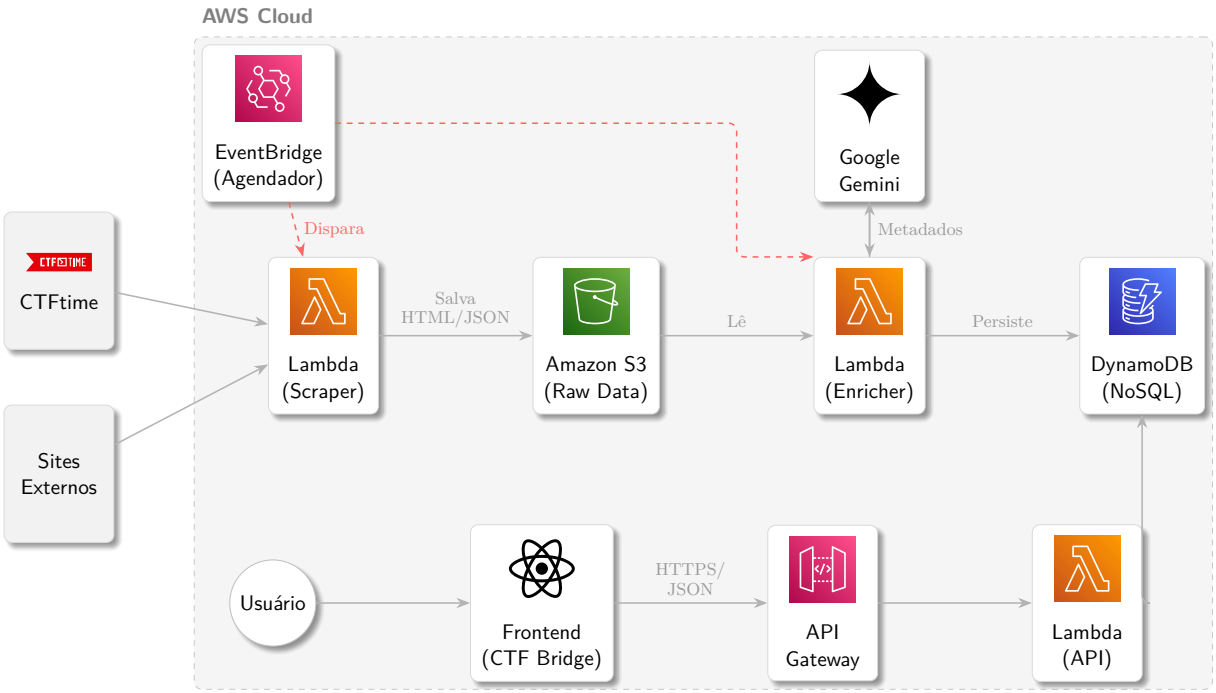


Figura 1 – Arquitetura da Solução

Esse *pipeline* garante que cada *write-up* seja coletado, organizado, rotulado e disponibilizado de forma consistente, mesmo quando o conteúdo original está disperso, incompleto ou hospedado em domínios instáveis.

4.1.2 Princípios Adotados na Arquitetura

A arquitetura proposta segue alguns princípios fundamentais:

- ***Serverless* (AWS Lambda):** reduz custos, elimina a necessidade de servidores dedicados e possibilita escalabilidade automática conforme a demanda de processamento. É fácil de manter e modificar. Possibilita times trabalhando de forma conjunta e possui baixo tempo de latência e disponibilidade após alteração;
- **Baixo acoplamento entre módulos:** cada componente funciona de forma independente, facilitando manutenções, atualizações e substituições sem impacto no restante da plataforma;
- **Custo reduzido:** a combinação de serviços *serverless*, S3 buckets e DynamoDB permite operar a plataforma com alta eficiência e baixo custo, mesmo com grandes volumes de dados;
- **Processo automatizado e recorrente:** o *pipeline* é executado periodicamente, garantindo atualização contínua dos *write-ups* e eliminando necessidade de intervenção manual.

4.2 Coleta de Dados

A etapa de coleta de dados é fundamental para a construção da plataforma, pois representa o ponto de entrada de todas as informações que posteriormente serão processadas, enriquecidas e disponibilizadas ao usuário. Como a maior parte da documentação produzida durante competições CTF é distribuída de maneira descentralizada na internet, foi necessário projetar um método robusto para localizar e capturar esses *write-ups*, garantindo consistência e recuperabilidade.

A solução adotada combina técnicas de scraping, heurísticas de identificação de conteúdo útil e, quando possível, uso de APIs oficiais. O objetivo dessa fase é extrair o máximo de informação bruta e preservá-la antes de qualquer transformação, mantendo também cópias históricas para auditoria e reprocessamento.

4.2.1 Coleta via Scraping – CTFtime

O CTFtime foi escolhido como fonte primária devido à sua relevância no cenário global de CTFs. É o site mais utilizado e com maior número de *write-ups* disponíveis. A plataforma concentra registros de competições, equipes e *write-ups* publicados por participantes, funcionando como repositório centralizado para milhares de desafios resolvidos ao longo dos anos. Essa centralização permite que o processo de coleta seja mais eficiente e consistente. Entretanto, a própria plataforma afirma utilizar grande parte de revisão manual e incompleta, deixando que usuários apenas informem que o verdadeiro conteúdo está hospedado em fontes externas.

Para realizar o *scraping*, foram adotadas técnicas específicas que garantem estabilidade e minimizam bloqueios:

- **Requisições HTTP com *User-Agent* customizado**: utilizadas para simular um agente legítimo e evitar recusas automáticas por parte do servidor do CTFtime;
- **Paginação de eventos e *write-ups***: como o CTFtime organiza seus *write-ups* de forma paginada, o *scraper* percorre sistematicamente as páginas disponíveis;
- **Limites de leitura**: parâmetros ajustáveis que evitam excesso de requisições e permitem controle preciso sobre o volume de dados coletados por execução;
- **Pausas entre requisições**: pequenas esperas são introduzidas para reduzir a chance de *rate limiting* e respeitar boas práticas de *scraping*;
- **Tratamento de erros HTTP**: respostas como 403, 404 ou *timeouts* são capturadas e registradas, permitindo que a execução continue sem interromper o processo geral.

A execução do processo de coleta é orquestrada por meio do **agendador da AWS (EventBridge)**, configurado para disparar automaticamente a função de *scraping* em

intervalos semanais. Esse período foi considerado adequado, pois a quantidade de novas competições e *write-ups* publicados simultaneamente é relativamente baixa, não havendo necessidade de atualizações em intervalos menores. Dessa forma, evita-se consumo desnecessário de recursos computacionais, mantendo o sistema eficiente e economicamente viável.

Além disso, cada *write-up* disponibilizado pelo CTFtime possui um identificador único. Esse identificador é utilizado como referência de controle no processo de coleta, permitindo que o sistema identifique recursivamente o último *write-up* já processado. A partir desse ponto, novas execuções retomam a coleta somente dos registros posteriores, evitando duplicação de dados e garantindo que não ocorra dupla indexação do mesmo conteúdo.

Após a coleta, todo conteúdo bruto é salvo diretamente em um S3 bucket, preservando a forma de que o conteúdo é obtido, seja o HTML exato retornado pelo site ou alguma outra API. Isso garante que o sistema tenha acesso à fonte original para auditoria ou revisões posteriores, mesmo que o site altere seu conteúdo ou fique offline.

4.2.2 Heurísticas para Identificação de Conteúdo Útil

Como os *write-ups* podem estar em diferentes formatos ou níveis de completude, o sistema utiliza heurísticas para identificar corretamente a natureza do conteúdo coletado. Essas heurísticas permitem diferenciar entre:

- **Links externos**: quando o CTFtime apenas referencia um *write-up* hospedado em outro domínio;
- **Texto completo colado na própria página**: quando o participante copia diretamente o conteúdo do *write-up* para dentro da plataforma;
- **Mini resumos ou descrições superficiais**: situações em que o *write-up* não contém detalhes técnicos suficientes, apenas uma breve descrição;
- ***Write-ups* extremamente curtos**: detectados quando o conteúdo possui poucos caracteres ou indica ausência de dados relevantes.

Essas classificações são importantes porque determinam como o conteúdo será processado nas etapas seguintes, especialmente na recursão externa e no enriquecimento via IA.

4.2.3 Resolução Recursiva de *Write-ups* Externos

Muitos *write-ups* estão hospedados fora do CTFtime, em *blogs*, repositórios *online*, como GitHub ou domínios pessoais. Para capturar esses conteúdos, foi implementado um mecanismo de resolução recursiva, que segue o link original e tenta extrair o texto completo do *write-up*.

O algoritmo segue os seguintes passos:

1. Identifica a URL original no *write-up* listado no CTFtime.
2. Visita o domínio externo utilizando uma nova requisição HTTP.
3. Extrai o HTML bruto retornado pela página.
4. Filtra elementos irrelevantes, como arquivos CSS, blocos de *script*, SVGs decorativos e componentes visualmente ignoráveis.
5. Busca seções de conteúdo útil, priorizando textos extensos, blocos de código, seções técnicas e explicações do desafio.

No entanto, esse processo apresenta desafios significativos, como:

- **Domínios *offline***: *write-ups* hospedados em *blogs* pessoais ou repositórios antigos podem ter sido removidos. Por isso, o *scraping* é feito por datas mais recentes, fazendo com que a plataforma seja usada para centralizar e estar sempre disponível ao usuário final.
- **Páginas dinâmicas (React, Next.js)**: o HTML inicial contém apenas *divs* vazias, dificultando a extração sem um navegador *headless*.
- **Soluções de proteção como Cloudflare**: certas páginas bloqueiam requisições automatizadas, impedindo coleta direta sem técnicas mais avançadas.

Apesar das limitações, a recursão é uma das partes mais importantes do *pipeline*, pois recupera grande parte dos *write-ups* que não estão disponíveis diretamente no CTFtime.

4.2.4 Coleta via APIs Oficiais (se aplicável)

Quando disponível, o sistema pode complementar metadados utilizando *endpoints* públicos do CTFtime, como informações sobre eventos, equipes participantes e datas. Embora nem sempre seja necessário, essa etapa contribui para enriquecer o banco de dados sem depender exclusivamente do texto presente no *write-up*.

Às vezes, o *write-up* original para o qual um registro do CTFtime aponta é um site a partir do qual é possível coletar dados a partir de uma API. Um exemplo é o do site `http://learn-cyber.net/`. Esse domínio contém vários *write-ups*, especialmente de certas equipes que aparecem constantemente nas competições. Apesar de não disponibilizar uma API oficial documentada, foi possível identificar APIs internas analisando as requisições realizadas ao acessar a página pelo navegador, implementá-las no fluxo principal e trazer o conteúdo em formato JSON, o que facilita as etapas seguintes do processamento de dados.

4.3 Normalização e Pré-processamento

A etapa de normalização e pré-processamento é responsável por transformar os dados coletados em um formato padronizado e adequado para posterior análise com inteligência artificial. Como os *write-ups* podem vir de diferentes fontes, idiomas e estruturas HTML, é essencial aplicar um conjunto de transformações que assegure consistência, qualidade e relevância do conteúdo processado. Isso garante acesso rápido, padronização e que todos os desafios sigam o mesmo formato e protocolo.

Esse processo envolve a limpeza do HTML, a limitação do volume de texto enviado para análise, a detecção automática de idioma e outras operações que tornam o conteúdo pronto para ser compreendido e enriquecido pelo modelo de IA.

4.3.1 Limpeza do HTML

A maior parte dos *write-ups* coletados contém elementos HTML que não são úteis para análise textual. Antes de qualquer processamento semântico, o sistema realiza uma limpeza profunda para extrair apenas o conteúdo relevante.

Para isso, são aplicadas técnicas utilizando `HTMLParser`, expressões regulares e filtros customizados. Entre os elementos removidos estão:

- *inline scripts*, como trechos de JavaScript embutidos;
- *tags* decorativas, incluindo estruturas de *layout* ou estilização;
- SVGs isolados, que costumam aparecer em cabeçalhos, logos e ícones.

Após a filtragem, o HTML processado é transformado em texto legível, contendo apenas parágrafos, blocos de código e partes essenciais da explicação técnica. Esse texto é o que será utilizado nas análises posteriores, facilitando a análise e economizando custos no processamento.

4.3.2 Análise de Tokens

Modelos de inteligência artificial possuem um limite sobre a quantidade de texto que conseguem analisar de uma só vez. Quanto maior o volume de conteúdo enviado, maior também é o tempo de processamento e o custo associado, obviamente. Por isso, antes de encaminhar o *write-up* para a IA, o sistema realiza uma verificação do tamanho do texto e ajusta o conteúdo conforme necessário.

Quando o *write-up* é muito extenso, o sistema reduz o material analisado, preservando as partes mais relevantes, como explicações técnicas, descrição da exploração, raciocínio utilizado e blocos de código. Esse processo evita o envio de trechos redundantes, como longos blocos decorativos, comentários irrelevantes ou conteúdo repetitivo. Dessa forma,

criamos um equilíbrio para a análise: não enviamos nada que seja irrelevante para o contexto, mas também focamos em preservar e melhorar a qualidade da análise.

Essa etapa garante que o texto esteja dentro dos limites aceitáveis para análise e mantém a qualidade das informações essenciais para que a inteligência artificial possa gerar resumos, identificar categorias técnicas e extrair os dados com precisão.

4.3.3 Identificação de Idioma

Como *write-ups* podem ser publicados em diferentes idiomas, o sistema realiza uma detecção automática de idioma antes de gerar qualquer análise.

A identificação é feita diretamente pelo módulo de IA, indicando o idioma original do texto.

Entretanto, não é perdido o conteúdo original. Ele é salvo e é até mesmo disponibilizado para o usuário final. Manter essa informação registrada é importante pelos seguintes motivos:

- fidelidade histórica: preserva o idioma em que o autor escreveu o *write-up*;
- processamento adequado: determina se a etapa de tradução será necessária;
- filtros futuros: possibilita pesquisas por idioma no *frontend* ou API.

Como se trata de um conteúdo extremamente técnico e de uma tradução feita sem revisão humana, alguns termos podem perder o sentido e dificultar a leitura do usuário. Se o usuário desejar, ele pode verificar o texto original para uma checagem dupla.

Quando o idioma identificado não é inglês, o texto é encaminhado para um *pipeline* separado de tradução, garantindo padronização e consistência no banco de dados.

4.4 Enriquecimento com Inteligência Artificial

Após a coleta e normalização dos dados, o próximo passo do sistema consiste em transformar o *write-up* bruto em um conteúdo estruturado, padronizado e enriquecido. Para isso, foi utilizado um modelo de inteligência artificial capaz de analisar o texto, interpretar seu conteúdo técnico e gerar metadados relevantes para consulta posterior. Esse processo permite que cada *write-up* seja classificado, resumido e organizado de forma consistente, independentemente de sua origem ou formato inicial.

O uso da IA contribui diretamente para a qualidade da plataforma, uma vez que automatiza tarefas que seriam inviáveis de realizar manualmente em grande escala, como categorização temática, identificação de vulnerabilidades e elaboração de resumos informativos. Isso economiza tempo, reduz custos e permite uma padronização aceitável mesmo quando não feita por humanos. Um exemplo é o sistema de *tags*: embora o próprio CTFtime

disponibilize um sistema de *tags* para os usuários, muitos optam por não utilizá-lo. Com o sistema de análise automatizado, sempre há essa disponibilidade.

4.4.1 Modelo Escolhido (Gemini 2.5 Flash)

O modelo adotado para o enriquecimento foi o Gemini 2.5 Flash, selecionado por combinar boa capacidade de análise com baixo custo e rapidez. O Flash possui arquitetura otimizada para throughput, oferecendo respostas até 3–6 vezes mais rápidas que modelos GPT de classe superior. Esse modelo é capaz de interpretar grandes trechos de texto, identificar conceitos de segurança da informação, técnicas comuns em desafios de CTF e relações entre etapas da solução. Ele também opera em valores de 10 e 40 vezes menores por token de entrada e saída comparado com outros modelos de outro nível, operando com tarifas extremamente reduzidas. Exemplificando com 1000 Input tokens, esse modelo custaria US$ 0.00035, enquanto o GPT 4.0 US$ 0.01.

Entre as principais vantagens observadas estão:

- capacidade de gerar resumos consistentes e claros;
- identificação automática de termos técnicos, ferramentas e vulnerabilidades;
- boa performance mesmo com textos extensos.

Essas características tornam o modelo adequado para lidar com a grande variedade de *write-ups* disponíveis na internet. Além disso, o sistema possui um ótimo custo-benefício. Como o projeto é de disponibilidade gratuita, ferramentas com baixo custo se tornam cruciais para sua sustentabilidade a longo prazo.

4.4.2 Sistema de Prompts

Para orientar o modelo de IA, foi elaborado um sistema de *prompts* que define exatamente quais informações o modelo deve extrair e como deve retornar os resultados. Esse *prompt* principal organiza a saída em um formato estruturado, facilitando o armazenamento e a consulta no banco de dados.

Além disso, o sistema conta com um *pipeline* complementar de tradução, acionado quando o *write-up* está em outro idioma que não o inglês. Nesse caso, a IA produz também uma versão traduzida dos principais trechos, garantindo uma experiência padronizada para consulta futura.

O conjunto final de dados enriquecidos inclui elementos como:

- resumo do *write-up*;
- dificuldade estimada;

- ferramentas utilizadas;
- vulnerabilidades exploradas;
- categorias temáticas;
- idioma original;
- versão traduzida (quando necessário).

4.4.3 Detecção Automática de Resumos

Alguns *write-ups* encontrados no CTFtime não apresentam conteúdo completo, sendo apenas pequenos trechos descritivos ou links externos. O modelo de IA é utilizado para identificar esses casos e classificá-los adequadamente.

A IA distingue entre:

- *write-up* completo;
- *write-up* resumido;
- *write-up* que contém apenas um link sem informações adicionais.

Essa classificação é importante para orientar o processo de recursão externa e determinar como o conteúdo deve ser exibido no *frontend*.

4.4.4 Tradução Automática

Como parte dos *write-ups* é publicada em diferentes idiomas, especialmente em comunidades da Rússia, China e países da Europa, o sistema realiza tradução automática sempre que o idioma original não é inglês.

O processo funciona da seguinte forma:

1. O idioma é detectado automaticamente durante o enriquecimento;
2. Quando necessário, a IA gera uma tradução dos trechos essenciais;
3. Ambos conteúdos são preservados.

Essa etapa garante que o material seja acessível a um público mais amplo e melhora a padronização das consultas.

4.4.5 Extração de Metadados Avançados

Além de resumos e traduções, a inteligência artificial também é responsável por gerar uma série de metadados técnicos que enriquecem o *write-up* e permitem consultas mais avançadas na plataforma.

O modelo identifica automaticamente:

- Categorias temáticas (como *reversing*, *crypto*, *web exploitation* ou *forensics*);
- Ferramentas mencionadas, incluindo IDA Pro, GDB, *pwntools*, Burp Suite etc.;
- Vulnerabilidades exploradas, como *buffer overflow*, *SQL injection*, XXE, RCE, entre outras;
- Nível de dificuldade provável.

Esses metadados permitem que o usuário filtre *write-ups* por tema, técnica, complexidade ou tipo de vulnerabilidade, tornando a plataforma muito mais útil tanto para estudantes quanto para profissionais de segurança.

4.4.6 Recurso de Explicações Contextuais ("Help Me Understand")

Além do enriquecimento principal, que extrai resumo, *tags*, vulnerabilidades, ferramentas e dificuldade, a plataforma conta com um segundo estágio de processamento opcional: a geração de uma explicação didática do *write-up*, utilizada na seção apresentada ao usuário como "Help me understand". Diferente de uma simples tradução ou resumo, essa explicação tem como foco tornar o conteúdo mais acessível para quem já possui uma base em programação, mas ainda não domina completamente as técnicas de exploração usadas na solução do problema em específico (ou simplesmente só gostaria de um material complementar).

No *backend*, esse recurso é implementado como uma chamada adicional ao modelo de inteligência artificial, acionada após o texto principal ter sido coletado, normalizado, eventualmente expandido e, quando necessário, traduzido para o inglês. A explicação é construída a partir do *write-up* completo, combinado com os metadados já extraídos. Esses metadados são enviados ao modelo em formato estruturado, servindo como contexto para orientar a explicação.

O *prompt* utilizado nessa etapa é diferente do *prompt* de enriquecimento. Enquanto o primeiro foca em produzir um JSON com campos bem definidos, o *prompt* do *explainer* orienta o modelo a escrever um texto em inglês, organizado em seções como visão geral, fluxo de ataque em alto nível e principais conceitos envolvidos. Ao mesmo tempo, são impostas restrições explícitas para que a explicação não forneça instruções passo a passo, comandos copiáveis ou *payloads* prontos para uso, mantendo o foco em entendimento conceitual e não em reprodução direta do ataque.

Caso a geração da explicação seja bem-sucedida, o texto resultante é armazenado em um campo separado no banco de dados, junto com os demais dados do *write-up*. Essa explicação passa então a compor uma seção própria na interface, permitindo que o usuário leia o *write-up* original e, em paralelo, tenha uma versão mais didática que resume o que aconteceu, como a vulnerabilidade funciona e por que o ataque é possível. Quando a geração falha ou o modelo retorna um conteúdo vazio, o sistema simplesmente não armazena a explicação, garantindo que o *write-up* permaneça disponível normalmente, mas sem a seção extra de auxílio.

Com isso, o recurso de "Help me understand" transforma o *pipeline* de enriquecimento em algo mais do que apenas categorização automática: ele aproxima a plataforma de uma ferramenta educacional, oferecendo interpretações em linguagem técnica, mas acessível, sobre resoluções de desafios que, muitas vezes, são densas ou difíceis de acompanhar na forma original.

4.5 Estrutura de Armazenamento

A etapa de armazenamento é essencial para garantir que todos os *write-ups* coletados, seus conteúdos brutos e as informações enriquecidas estejam organizados e acessíveis de forma eficiente. Para isso, foram adotados dois componentes principais da AWS: o Amazon S3, destinado ao armazenamento bruto e histórico dos arquivos, e o Amazon DynamoDB, utilizado para armazenar os dados estruturados e facilitar consultas rápidas.

Essa separação entre armazenamento bruto e estruturado permite ao sistema preservar o conteúdo original, manter rastreabilidade completa do processo e, ao mesmo tempo, oferecer alto desempenho nas consultas realizadas pelo *frontend* e pela API.

4.5.1 Modelo no S3

O Amazon S3 é utilizado como repositório de todos os arquivos coletados ao longo do processo, funcionando como uma camada de preservação histórica. Em cada etapa, são armazenados diferentes tipos de conteúdo, incluindo:

- HTML bruto obtido diretamente do CTFtime;
- HTML extraído de páginas externas por meio do mecanismo de recursão;
- arquivos JSON contendo resultados intermediários ou versões finais enriquecidas.

Manter esses arquivos no S3 garante:

- possibilidade de reprocessar *write-ups* futuramente caso novas técnicas de análise sejam adicionadas;

- auditoria completa de cada etapa do *pipeline*;
- proteção contra perda de dados, mesmo que as páginas originais de *write-ups* saiam do ar.

O uso do S3 reduz custos e simplifica o fluxo de trabalho, funcionando como uma base de referência para qualquer etapa da aplicação.

4.5.2 Modelo no DynamoDB

Enquanto o S3 funciona como arquivo bruto, o DynamoDB é responsável por armazenar os dados estruturados e enriquecidos, permitindo consultas rápidas e organizadas. Cada *write-up* é salvo como um registro contendo os principais campos extraídos ao longo do *pipeline*, incluindo:

- **writeup_id**: identificador único para cada *write-up*;
- **source_url** e **source_host**: endereço original e domínio de origem;
- **content_text** e **content_text_len**: conteúdo textual já limpo;
- **summary**: resumo gerado pela IA;
- **tags** e categorias técnicas;
- **difficulty**: dificuldade estimada;
- **vulnerabilities**: vulnerabilidades mencionadas ou exploradas;
- **date**: data do evento;
- **team**: equipe responsável pelo *write-up*;
- **event**: competição onde o desafio foi apresentado.

O DynamoDB foi escolhido por ser um banco NoSQL, o que traz vantagens importantes:

- **flexibilidade estrutural**: *write-ups* podem variar bastante em formato e tamanho;
- **alta performance**: consultas rápidas mesmo com grande volume de dados;
- **escalabilidade automática**: o banco se ajusta conforme a plataforma cresce.

Essa arquitetura facilita consultas dinâmicas e permite que o *frontend* ofereça filtros avançados sem comprometer o desempenho. Além disso, como os *write-ups* podem ter estruturas diferentes entre si — por exemplo, alguns contam com tradução e outros não, ou possuem quantidades distintas de *tags*, vulnerabilidades e técnicas — o uso de um banco

NoSQL como o DynamoDB é vantajoso. Esse modelo lida bem com campos opcionais e registros heterogêneos sem exigir alterações de esquema ou múltiplas tabelas de relacionamento, o que simplifica a evolução da plataforma e contribui para sua escalabilidade e estabilidade.

4.5.3 Indexação

Para otimizar as consultas realizadas pela API e pelo *frontend*, o armazenamento no DynamoDB foi estruturado para permitir buscas eficientes por diferentes dimensões. Entre as principais formas de consulta estão:

- por data do evento, permitindo análise histórica;
- por categoria técnica, como *crypto*, *web*, *pwn*, *reversing*, *forensics* etc.;
- por dificuldade, para filtrar desafios fáceis, médios ou avançados;
- por competição, agrupando *write-ups* por edição ou evento;
- por time, permitindo visualizar conteúdos produzidos por equipes específicas.

Essa flexibilidade torna o banco adequado tanto para navegação exploratória quanto para estudos direcionados.

4.5.4 Prompts

a) Prompt de extração de conteúdo principal (HTML → texto/Markdown)

Utilizado na etapa recursiva para limpar páginas externas em HTML e manter apenas o conteúdo do write-up:

```
You will receive the full HTML of a page that contains a CTF challenge
write-up or problem statement.
Your task is to extract ONLY the main technical content related to the
challenge: the description, explanation, methodology, reasoning, commands,
and code snippets.
Ignore menus, headers, footers, sidebars, ads, navigation elements,
comments, and unrelated sections.
Preserve the logical reading order.
Output the result in clean Markdown or plain text:
Use #, ##, ### for headings when appropriate.
Use lists when useful.
Use fenced code blocks ``` for code or commands.
```

```
DO NOT summarize, DO NOT comment on your process, DO NOT add
explanations.
Return ONLY the cleaned text.
Page URL: <page_url>
Full HTML:
```

b) Prompt de análise principal (JSON de resumo/tags/vulnerabilidades)

Utilizado para enriquecer o write-up com metadados estruturados:

```
You will receive a CTF write-up or long challenge description.
Respond ONLY with a VALID JSON object with the keys:
summary (2-4 sentences in ENGLISH, concise),
predicted_tags (up to 5; high-level categories like
'web','pwn','crypto','forensics','reverse','stego','osint','mobile','misc'),
vulnerabilities (1-3 main vulnerability categories, NEVER empty; prefer
values from this list when possible: ['SQL
injection','XSS','CSRF','IDOR','SSRF','RCE','Path Traversal','Open
Redirect','Privilege Escalation','Auth Logic Flaw','Insecure
Deserialization','Crypto weakness','Forensics / Artifact
analysis','Reverse Engineering / Binary','Pwn / Memory
corruption','Misconfiguration','Other / Misc']),
tools (up to 5; libraries, frameworks, or tools used in the solution),
difficulty (one of 'beginner','intermediate','advanced' - classify the
overall difficulty of the CHALLENGE, not the length of the write-up; use
'intermediate' or 'advanced' when the exploit involves multiple steps,
kernel or low-level exploitation, complex cryptography, or deep reverse
engineering),
team (string; if there is no clear team name in the text, leave it
empty),
source_lang (ISO 639-1 language code of the INPUT text, e.g.
'en','pt','es','ru').
If there is no clear vulnerability type, use ['Other / Misc'] instead of
an empty list.
Respond ONLY with the JSON object.
```

c) Prompt de tradução

Usado quando o texto original não está em inglês, preservando blocos de código:

```
Translate the following text into ENGLISH while preserving code blocks
inside ``` and inline code surrounded by backticks.
```

```
DO NOT summarize the text; keep the structure, headings, lists, and
formatting intact.
At the end of the translation, append this line in italic:
Translated from {lang}.
Text:
```

d) Prompt de explicação didática (explainer técnico)

Usado para gerar uma explicação em nível conceitual, sem passo a passo de exploração:

```
You will receive a CTF write-up (already normalized to plain text or
Markdown) plus some metadata (difficulty, tags, vulnerabilities, tools).
Your task is to produce a CLEAR, TECHNICAL and NEUTRAL explanation that
helps someone with basic Linux/programming knowledge understand the
vulnerability and the high-level solution idea.

IMPORTANT SAFETY RULES:
Explain concepts and attack flow ONLY at a high level.
DO NOT provide step-by-step exploit instructions.
DO NOT include exact payloads, full commands, or one-liner exploits.
DO NOT describe how to reproduce the attack in a copy-pasteable way.
Focus on what was wrong in the design or implementation and how the
write-up author fixed or abused it conceptually.

Language:
Write everything in ENGLISH.
Do NOT mix Portuguese or other languages in the explanation.

Style rules:
Do NOT start with any greeting or motivational phrase.
Do NOT address the reader with things like 'hey there' or 'future
cybersecurity pro'.
Do NOT talk about yourself or about being an AI.
Use an objective tone, like a concise textbook or technical write-up.
Avoid jokes, hype, or storytelling tone. Focus on the technique.

Structure the answer as:
A 1-2 sentence overview starting directly with the core idea, like:
'This challenge abuses a custom authentication scheme to read /flag from
the server.'
```

```
A 'High-level attack flow' section with a numbered list describing the
main phases of the attack, but WITHOUT copy-pasteable payloads or exact
commands.
A 'Key concepts' section listing the main vulnerabilities and techniques
involved, each in a short bullet point.
Keep the explanation focused on HOW the vulnerability works and WHY the
attack is possible, not on giving the reader a ready-to-use exploit.
```

4.6 *Backend Serverless*

O *backend* da plataforma foi construído seguindo o paradigma *serverless*, que elimina a necessidade de servidores dedicados e permite que todas as operações sejam executadas sob demanda, com escalabilidade automática e custo reduzido. Essa abordagem proporciona uma infraestrutura simples de manter, além de compatível com um *workflow* altamente automatizado, que precisa processar dados de forma recorrente sem intervenção humana.

A arquitetura *serverless* adotada combina diferentes funções AWS Lambda, cada uma responsável por uma parte específica do *pipeline*, além de serviços auxiliares para agendamento e monitoramento.

4.6.1 *Pipelines* em AWS Lambda

O *backend* é composto por três funções principais, cada uma com responsabilidades distintas, permitindo que o fluxo de dados seja modular, independente e facilmente escalável.

Scraper Lambda

Responsável pela coleta inicial dos *write-ups*. Essa função realiza:

- leitura das páginas do CTFtime;
- extração de links, conteúdos e metadados básicos;
- coleta recursiva de *write-ups* hospedados em domínios externos;
- armazenamento do HTML bruto no S3 para preservação histórica.

A função é projetada para ser independente, evitando duplicações e tratando erros de rede de forma controlada.

Em alto nível:

1. Ler variáveis de ambiente e parâmetros do evento.

2. Se o evento contiver `writeup_id` ou `writeup_url`, ativar modo *single* com apenas esse ID.
3. Caso contrário (modo paginado):
   a) Para `page` de 1 até `MAX_PAGES`:
      i. Montar URL da lista (`/writeups` ou `/writeups/<page>`).
      ii. Fazer requisição HTTP e obter HTML.
      iii. Extrair todos os IDs de *write-up* via regex `href="/writeup/<id>"`.
      iv. Acumular IDs, evitando duplicados.
   b) Aplicar limite `MAX_WRITEUPS`, se definido.
   c) Aplicar *offset* e *limit* para selecionar o subconjunto de IDs a processar.
4. Para cada `writeup_id` selecionado:
   a) Montar URL `https://ctftime.org/writeup/<id>`.
   b) Fazer requisição HTTP e obter HTML da página do *write-up*.
   c) Extrair metadados principais:
      i. Evento e desafio: a partir da tag `<title>`.
      ii. Nome da equipe: buscando links ou labels com "Team".
      iii. *Tags* do CTFtime: `spans` com classe `label label-info`.
   d) Extrair o texto da descrição principal (`original_content`) usando:
      i. Se existir: *string* de *markdown* de `MDConvert.render('...')`.
      ii. Caso contrário: o conteúdo do `<div id="id_description">`.
5. Localizar a URL do *write-up* original (`original_url`):
   a) Procurar *anchor* com texto "Original write-up" ou similar.
   b) Procurar padrão `Original writeup (https://...)` no HTML.
   c) Caso ainda não encontre, buscar qualquer URL na descrição, priorizando domínios fora de `ctftime.org`.
6. Decidir se o texto do CTFtime é apenas resumo:
   a) Compactar `original_content` removendo espaços extras.
   b) Se o comprimento for menor ou igual a `SUMMARY_MAX_CHARS`, marcar como resumo/*link-only*.
7. Coleta recursiva de conteúdo externo (`recursive_content`):

a) Se existir `original_url` e (texto for resumo ou `ALWAYS_RECURSIVE_IF_ORIG = true`):

i. Normalizar a URL (ex.: converter `github.com/.../blob/...` para `raw.githubusercontent.com/...`).

ii. Se o domínio for `learn-cyber.net`, chamar a API específica e obter o campo `content`.

iii. Caso a URL aparente ser arquivo de texto (`.md`, `.txt`, *raw* GitHub), usar o conteúdo diretamente.

iv. Caso seja HTML genérico:

A. Baixar o HTML completo.

B. Remover blocos `<style>` e `<script>`.

C. Enviar o HTML limpo para o modelo Gemini com um *prompt* para extrair apenas o conteúdo principal do *write-up*.

D. Usar a resposta do modelo como `recursive_content`.

b) Se a coleta recursiva for bem-sucedida, marcar `did_recursive_fetch = true`.

8. Construir o `content_preview`:

a) Escolher como base `recursive_content` (se existir) ou `original_content`.

b) Normalizar espaços e cortar o texto em um tamanho máximo fixo (ex.: ~400 caracteres).

9. Montar o documento final do *write-up*:

a) Incluir:

- `ctftime_writeup_id`, URL no CTFtime, `original_url`, `original_host`;
- `event`, `challenge`, `team`;
- `original_content`, `recursive_content`, `content_preview`;
- `did_recursive_fetch`;
- `ingested_at` (*timestamp*);
- `ctftime_tags`.

b) Adicionar o documento à lista `docs`.

10. Ao final do loop:

a) Converter `docs` em NDJSON (um JSON por linha).

b) Gerar um nome de arquivo baseado no *timestamp* (`writeups_<timestamp>.ndjson`).

c) Gravar o arquivo no bucket S3 configurado em `RAW_BUCKET`.

d) Retornar um resumo da execução (contagem de *write-ups*, tempo, caminho do arquivo e número de chamadas ao Gemini).

Enricher Lambda

Responsável por transformar os dados brutos em conteúdo estruturado e enriquecido. Suas principais tarefas são:

- limpeza e normalização do texto;
- envio do conteúdo para análise pela IA;
- geração de resumos, categorias técnicas e metadados;
- detecção de idioma e tradução quando necessário;
- gravação dos dados processados no DynamoDB.

Essa função automatiza um processo que seria extremamente trabalhoso de realizar manualmente, garantindo consistência entre *write-ups* diversos.

Em alto nível:

1. Ler evento de entrada:
   a) Determinar bucket e chave do S3.
   b) Carregar o arquivo bruto (NDJSON/JSON).
2. Para cada documento no arquivo:
   a) Normalizar entrada:
      - Extrair título, evento, challenge, `source_url`, *tags* do CTFtime.
      - Gerar `writeup_id` único (por ID do CTFtime ou hash da URL).
   b) Verificar duplicidade:
      - Se *write-up* já existe no DynamoDB e possui `content_text`:
      - se não houver `FORCE_REEXTRACT` → pular.
   c) Obter texto base:
      SE `original_content`/`recursive_content` vierem do S3:
      - Escolher conteúdo baseado em heurísticas:
        – usar `recursive_content` se for muito maior;
        – ou combinar ambos;
        – senão usar apenas `original_content`.

      SENÃO:
      - Baixar conteúdo via HTTP (CTFtime → URL original → *fallback*).
      - Remover HTML usando *parser*.

d) Aplicar expansão (`learn-cyber`/GitHub) se conteúdo < `THIN_MIN_LEN`.

e) Enriquecimento com Gemini (Prompt 4.5.4-a):

- Dividir texto em *chunks* (*overlap*).
- Para cada *chunk*: `resultado_i ← Gemini(JSON, usando Prompt A)`.

f) Unificar resultados:

- `summary` final;
- `predicted_tags` (deduplicadas);
- `vulnerabilities`;
- `tools`;
- `difficulty` (maior nível detectado);
- `team`;
- `source_lang`.

g) Ajuste de idioma:

- Comparar `source_lang` do modelo com heurística local.
- Escolher o mais confiável.

h) Tradução (Prompt 4.5.4-c):

SE idioma $\neq$ `"en"`:

- `final_text ← Gemini(translate, Prompt C, preservando código)`.

SENÃO:

- `final_text ← texto original`.

i) Heurísticas adicionais:

- Extração opcional de sinais (*regex*) para:
  - *tags*;
  - *vulnerabilities*;
  - *tools*.
- Garantir listas não vazias.
- Fundir *tags* do CTFtime com as *tags* derivadas.
- Ajustar dificuldade baseado em termos técnicos (ROP, kernel, BPF, cryptography, etc.).

j) Explicação didática (Prompt 4.5.4-d):

SE `EXPLAINER_ENABLED`:

- `explainer_text ← Gemini(Explainer, Prompt D, usando final_text + metadados)`.

k) Salvar resultado no DynamoDB:

- `writeup_id`;
- metadados unificados;
- `summary`;
- `difficulty`;
- `tags` / `vulnerabilities` / `tools`;
- `source_lang`;
- `preview` (1000 chars);
- `content_text` (limitado);
- `explainer_text` (se existir);
- `original`/`recursive_content` (se existirem).

3. Registrar logs e retornar: `{ status:"ok", processed, errors }`

API Handler Lambda

Responsável por fornecer os dados ao *frontend* e a consumidores externos:

- leitura do DynamoDB;
- aplicação de filtros como categoria, dificuldade, evento e data;
- formatação da resposta para consumo via API Gateway.

Sua função é garantir que a plataforma CTF Bridge consiga acessar *write-ups* e metadados rapidamente, com respostas leves e otimizadas.

Em alto nível:

1. Inicialização:
   a) Ler variáveis de ambiente:
      - `WRITEUPS_TABLE` (DynamoDB);
      - `FRONT_ORIGIN` (CORS).
   b) Criar cliente DynamoDB (`TABLE`).
   c) Definir *helpers*:
      - `_cors_headers()` → monta cabeçalhos CORS padrão;
      - `_to_jsonable(x)` → converte `Decimal` → `int/float` (recursivo);
      - `_qs(event)` → extrai filtros (*tag*, *vuln*, *difficulty*, *limit*);
      - `_scan_paginated(limit)` → faz SCAN paginado na tabela;
      - `_all_tags_from_item()` → combina *tags*, `ctftime_tags`, `raw_tags`;
      - `_build_related_for_item()` → calcula *write-ups* relacionados por *tags*.

2. Entrada principal: `lambda_handler(event, context)`:

   a) Ler `httpMethod` e `path`.

   b) Se `httpMethod == OPTIONS`:

      - Responder 204 com cabeçalhos CORS (*preflight*).

   c) Se `httpMethod == GET`:

      i. Se `path` é `/roadmap`: chamar `build_roadmap(event)`.

      ii. Se `path` é `/writeups`: chamar `list_writeups(event)`.

      iii. Se `path` contém `/writeups/{id}`: chamar `get_writeup(event)`.

      iv. Caso *fallback*:

         - Se existir `pathParameters.id` ou `pathParameters.writeup_id`: chamar `get_writeup(event)`.
         - Senão: chamar `list_writeups(event)`.

   d) Caso método não suportado:

      - Responder 404 JSON `{"error": "not_found"}` com CORS.

4.6.2 Sequenciamento Diário

Para que a plataforma permaneça atualizada, o fluxo é executado automaticamente todos os dias.

Esse agendamento é feito por meio do Amazon EventBridge, que dispara cada etapa do *pipeline* de forma ordenada:

- **Scraping**: coleta os *write-ups* mais recentes adicionados ao CTFtime;
- **Enriquecimento**: processa todos os novos conteúdos e atualiza os registros existentes, se necessário;
- **API update**: garante que o *frontend* tenha acesso aos dados mais recentes e que o *cache* da API seja atualizado.

Esse processo automatizado elimina a necessidade de execução manual e garante que a plataforma acompanhe o ritmo das competições de forma contínua.

4.6.3 Tolerância a Falhas

Como cada etapa do *pipeline* opera de forma independente, o sistema é naturalmente resistente a falhas. Alguns mecanismos adicionais reforçam essa robustez:

- *retries* automáticos, garantindo nova tentativa quando há falhas temporárias de rede;

- *logs* centralizados no CloudWatch, permitindo diagnóstico rápido de problemas;
- proteção contra conteúdo duplicado, evitando que o mesmo *write-up* seja processado diversas vezes;
- checagem de reprocessamento (como a lógica de *force re-extract*) para *write-ups* que mudaram ou foram corrigidos após a coleta inicial.

Essa combinação garante um *backend* estável, resiliente e capaz de lidar com irregularidades comuns na internet, como sites fora do ar, páginas inconsistentes ou mudanças repentinas no formato do CTFtime.

4.7 Frontend (CTF Bridge)

O *frontend* da plataforma, aqui denominado como CTF Bridge, foi desenvolvido para fornecer uma interface clara, moderna e intuitiva que facilite a navegação entre os *write-ups* coletados e enriquecidos pelo sistema. Por meio dele, estudantes, pesquisadores e profissionais de segurança da informação podem consultar conteúdos técnicos, identificar categorias, visualizar resumos e acessar os *write-ups* originais de forma simples e organizada.

A interface foi construída com foco na usabilidade e na eficiência, permitindo que o usuário encontre rapidamente os materiais de interesse. Prints da interface estão incluídos ao longo desta seção para ilustrar o funcionamento e a organização visual do sistema.

4.7.1 Estrutura do Aplicativo

O aplicativo foi desenvolvido utilizando React em conjunto com Vite, o que proporciona carregamento rápido e uma experiência fluida. Sua estrutura é organizada em componentes simples e funcionais, como:

- **Navbar Global**: localizada no topo, apresenta o nome da plataforma e facilita o retorno à página inicial, além de links úteis.
- **Página inicial com lista de *write-ups***: exibe todos os *write-ups* disponíveis, acompanhados de informações resumidas como título, evento, resumo e *tags* técnicas. Também conta com o componente de filtro para buscar *write-ups* específicos.

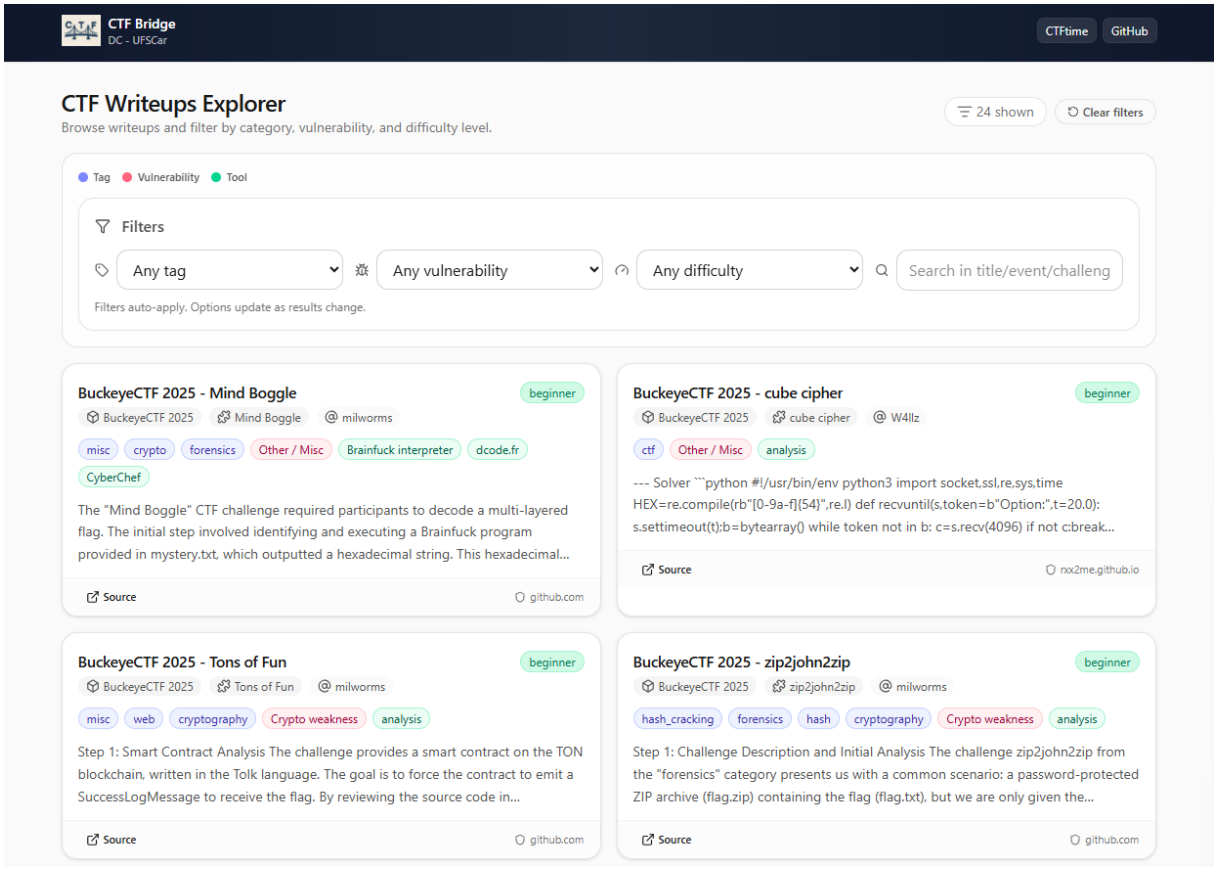


Figura 2 – CTF Bridge Homepage

Página de detalhes do *write-up*: apresenta o conteúdo completo enriquecido pela IA, incluindo resumo, metadados técnicos, categorias, dificuldades e link para o *write-up* original.

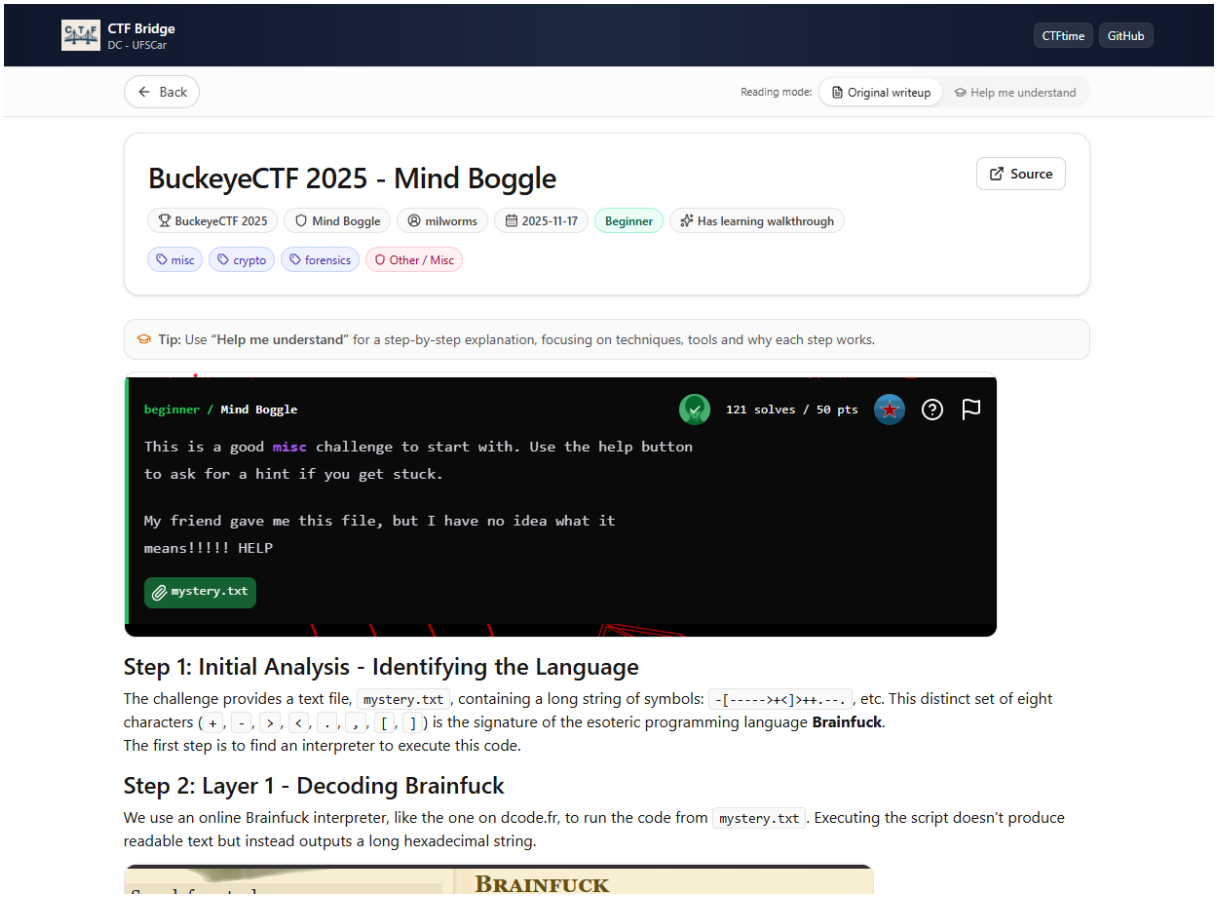


Figura 3 – Página de *write-up* na CTF Bridge

Essa estrutura modular torna a interface fácil de navegar e possibilita futuras expansões sem comprometer o *design* geral.

4.7.2 Forma de Exibição

A exibição dos *write-ups* no *frontend* foi projetada para apresentar as informações de forma clara e acessível, destacando os elementos mais relevantes para o usuário sem gerar sobrecarga visual. Cada registro é mostrado inicialmente em uma lista na página principal, onde o usuário tem uma visão geral do conteúdo disponível. Nessa listagem, aparecem o título do *write-up*, um breve resumo gerado pela inteligência artificial e alguns metadados essenciais, como a competição em que o desafio foi apresentado e a categoria técnica à qual pertence. Essa visão rápida permite que o usuário identifique rapidamente quais *write-ups* são mais relevantes para sua pesquisa ou estudo.

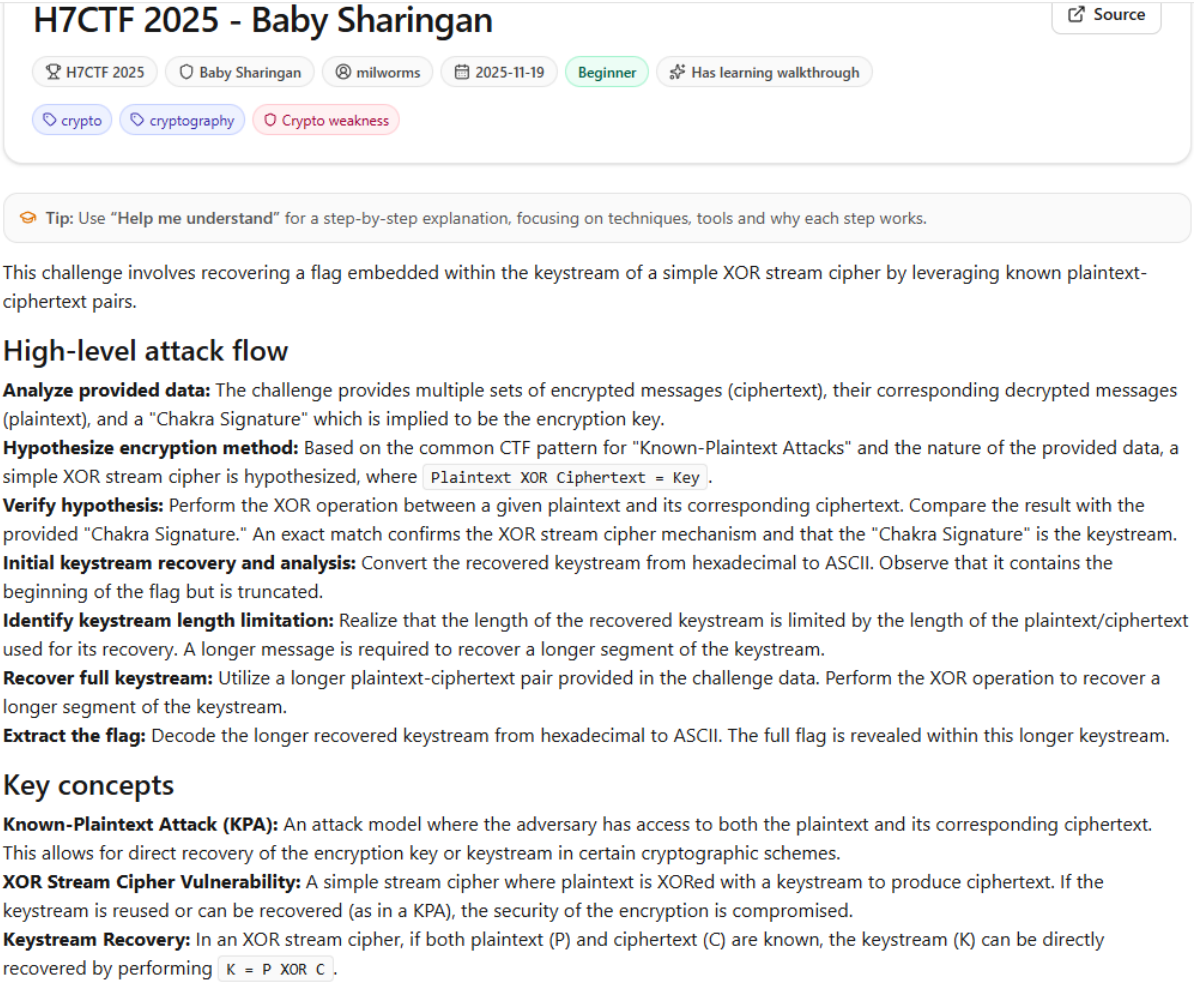
Ao selecionar um item da lista, o usuário é direcionado para a página de detalhes, onde o *write-up* é exibido de maneira mais completa. Nessa visualização ampliada, o conteúdo enriquecido pela IA é apresentado de forma organizada, incluindo um resumo conciso, as técnicas utilizadas, as vulnerabilidades identificadas, possíveis ferramentas mencionadas e demais informações técnicas extraídas durante o processo de enriquecimento. O conteúdo

original, coletado do CTFtime ou de fontes externas, também é exibido quando disponível, permitindo uma leitura fiel ao material produzido pelo autor do *write-up*.

Além disso, a interface destaca o link para o *write-up* original, quando esse existe, facilitando o acesso direto à fonte e mantendo a transparência sobre a origem dos dados. A disposição dos elementos na página foi pensada para facilitar a leitura contínua, sem interrupções ou excesso de informações na tela. Para complementar a explicação, *prints* da interface podem ser incluídos ao longo desta seção, ilustrando a organização visual e a forma como os dados são apresentados ao usuário.

Para reforçar a compreensão da interface e das funcionalidades oferecidas ao usuário, foram incluídas telas adicionais do sistema que evidenciam recursos importantes do processo de enriquecimento e navegação dos *write-ups*.

Um dos principais diferenciais da plataforma é o resumo gerado pela inteligência artificial, apresentado de forma didática e em alto nível. Esse resumo tem como objetivo auxiliar usuários com menor familiaridade técnica a compreenderem rapidamente o contexto do desafio, a ideia geral da solução e os conceitos envolvidos, servindo como um ponto de partida para o estudo aprofundado do *write-up* original.



H7CTF 2025 - Baby Sharingan

Source

H7CTF 2025 | Baby Sharingan | milworms | 2025-11-19 | Beginner | Has learning walkthrough

crypto | cryptography | Crypto weakness

Tip: Use **"Help me understand"** for a step-by-step explanation, focusing on techniques, tools and why each step works.

This challenge involves recovering a flag embedded within the keystream of a simple XOR stream cipher by leveraging known plaintext-ciphertext pairs.

High-level attack flow

Analyze provided data: The challenge provides multiple sets of encrypted messages (ciphertext), their corresponding decrypted messages (plaintext), and a "Chakra Signature" which is implied to be the encryption key.
Hypothesize encryption method: Based on the common CTF pattern for "Known-Plaintext Attacks" and the nature of the provided data, a simple XOR stream cipher is hypothesized, where `Plaintext XOR Ciphertext = Key`.
Verify hypothesis: Perform the XOR operation between a given plaintext and its corresponding ciphertext. Compare the result with the provided "Chakra Signature." An exact match confirms the XOR stream cipher mechanism and that the "Chakra Signature" is the keystream.
Initial keystream recovery and analysis: Convert the recovered keystream from hexadecimal to ASCII. Observe that it contains the beginning of the flag but is truncated.
Identify keystream length limitation: Realize that the length of the recovered keystream is limited by the length of the plaintext/ciphertext used for its recovery. A longer message is required to recover a longer segment of the keystream.
Recover full keystream: Utilize a longer plaintext-ciphertext pair provided in the challenge data. Perform the XOR operation to recover a longer segment of the keystream.
Extract the flag: Decode the longer recovered keystream from hexadecimal to ASCII. The full flag is revealed within this longer keystream.

Key concepts

Known-Plaintext Attack (KPA): An attack model where the adversary has access to both the plaintext and its corresponding ciphertext. This allows for direct recovery of the encryption key or keystream in certain cryptographic schemes.
XOR Stream Cipher Vulnerability: A simple stream cipher where plaintext is XORed with a keystream to produce ciphertext. If the keystream is reused or can be recovered (as in a KPA), the security of the encryption is compromised.
Keystream Recovery: In an XOR stream cipher, if both plaintext (P) and ciphertext (C) are known, the keystream (K) can be directly recovered by performing `K = P XOR C`.

Figura 4 – Resumo didático gerado por IA para entendimento de alto nível

Outro elemento relevante da interface é o sistema de filtros, que permite ao usuário refinar a listagem de *write-ups* de acordo com critérios técnicos específicos. Entre os filtros disponíveis estão categorias, *tags*, nível de dificuldade e outros metadados extraídos durante o processo de enriquecimento. Esse mecanismo facilita a navegação em grandes volumes de conteúdo e possibilita uma pesquisa mais direcionada, atendendo tanto usuários iniciantes quanto mais experientes.

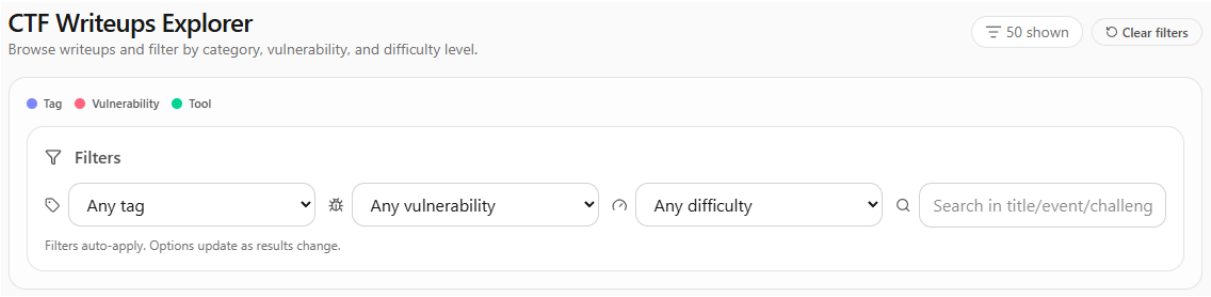


Figura 5 – Interface de filtros para seleção de *write-ups* por critérios técnicos

Além disso, a plataforma informa explicitamente quando um *write-up* foi traduzido automaticamente. Essa indicação visual no *frontend* garante transparência ao usuário, permitindo que ele saiba que o conteúdo exibido passou por um processo de tradução, ao mesmo tempo em que oferece acesso à linguagem original sempre que disponível. Esse recurso amplia o acesso a conteúdos produzidos em outros idiomas, sem comprometer a confiabilidade das informações apresentadas.

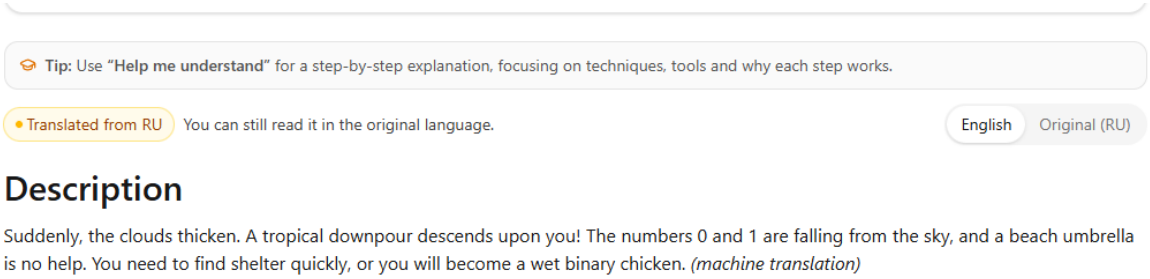


Figura 6 – Indicação visual de conteúdo traduzido e acesso ao idioma original

Esses elementos complementam a experiência do usuário ao tornar a navegação mais intuitiva, o conteúdo mais acessível e o processo de estudo mais eficiente, alinhando a interface do sistema aos objetivos educacionais propostos pelo projeto.

4.8 Desafios Encontrados

Durante o desenvolvimento da plataforma, alguns desafios se destacaram. Um deles foi a heterogeneidade das fontes: *write-ups* hospedados em *blogs* pessoais, páginas estáticas, aplicações modernas em React/Next.js e repositórios no GitHub geram HTMLs muito diferentes entre si. Em alguns casos, o conteúdo principal não aparece diretamente no HTML inicial, o que limita a extração apenas com requisições simples, sem uso de navegadores *headless*.

Outro ponto relevante foi a instabilidade das fontes externas. Diversos links apontavam para domínios expirados, páginas removidas ou serviços temporariamente fora do ar. Isso exige que o sistema trabalhe com *fallbacks*, registre erros de forma controlada e aceite que, em alguns casos, o *write-up* poderá ser parcialmente recuperado ou depender apenas do conteúdo presente no próprio CTFtime.

Por fim, houve um desafio específico relacionado ao uso de modelos de inteligência artificial em um contexto de segurança ofensiva. Muitos *write-ups* descrevem técnicas de exploração, construção de *payloads* e etapas de ataque que, se detalhadas em excesso, podem violar as políticas de uso seguro dos modelos de IA. Na prática, isso limita o nível de detalhamento que a IA pode fornecer, especialmente em explicações mais didáticas, e exige um cuidado especial na formulação dos *prompts*, priorizando descrições em alto nível e evitando instruções passo a passo. Como consequência, parte das saídas precisa ser orientada a conceitos e fluxos gerais, e não à reprodução exata dos ataques descritos nos *write-ups*.

5 Resultados e Discussão

Este capítulo apresenta os principais resultados obtidos com o desenvolvimento do projeto. A análise é qualitativa e quantitativa, destacando o desempenho do *pipeline*, a eficácia das heurísticas, a qualidade do enriquecimento produzido pela inteligência artificial e a experiência oferecida ao usuário final por meio do *frontend*. Também são discutidas limitações observadas durante o desenvolvimento e execução da solução.

5.1 Desempenho do Processo de Coleta

A etapa de coleta demonstrou desempenho consistente e eficiente ao processar os *write-ups* presentes no CTFtime. Durante os testes realizados ao longo do desenvolvimento, o *pipeline* foi capaz de:

- processar entre 1 e 50 *write-ups* por execução, dependendo da quantidade de páginas configuradas para *scraping*. A limitação principal deveu-se aos limites de requisição (*rate limits*) da API da IA;
- identificar corretamente *write-ups* completos, incompletos ou com apenas redirecionamento;
- preservar o conteúdo original no S3, garantindo a rastreabilidade e permitindo reprocessamento futuro.

O sistema demonstrou resiliência ao lidar com links inválidos, domínios fora do ar e páginas inconsistentes, registrando erros de forma controlada sem interromper o fluxo geral. A coleta recursiva foi particularmente importante para recuperar *write-ups* hospedados em sites externos, melhorando significativamente a completude dos dados e reduzindo a dependência de uma única plataforma.

A função Lambda responsável pelo *scraping* manteve tempo de execução baixo e custo reduzido, reforçando a viabilidade do uso de uma arquitetura totalmente *serverless* e com alto potencial de escalabidade.

5.2 Qualidade do Conteúdo Recursivo Extraído

A resolução recursiva apresentou resultados expressivos ao recuperar conteúdos relevantes presentes fora do CTFtime. No atual cenário, de 50 *write-ups* buscados, 49 retornaram seu conteúdo integralmente, representando uma taxa de sucesso de 98%. O único caso de falha deveu-se a uma extensão de arquivo não suportada. Em casos de páginas estáticas

ou *markdown* armazenados em repositórios GitHub, a extração foi integral para a página de entrada, preservando formato e estrutura, apresentando limitações apenas quando o conteúdo se encontrava segmentado em múltiplos arquivos pela estrutura do repositório.

Para páginas com HTML complexo ou carregamento dinâmico, a utilização do modelo de IA para extrair apenas o conteúdo principal aumentou consideravelmente a qualidade do texto final. Em *write-ups* hospedados em sites como *learn-cyber.net*, a automatização da extração via API interna descoberta mostrou-se eficaz, reduzindo custo e melhorando consistência.

Apesar disso, certos cenários permaneceram desafiadores, como páginas altamente dependentes de JavaScript para renderização. Nesses casos, parte do conteúdo pode não ser recuperável sem técnicas adicionais, como o uso de navegadores *headless*.

5.3 Resultados do Enriquecimento com IA

A etapa de enriquecimento com IA demonstrou ser um dos principais diferenciais de todas as outras plataformas existentes. Os resultados obtidos mostram que o modelo de IA foi capaz de produzir dados coerentes, consistentes e úteis para consultas futuras.

Entre os resultados obtidos, destacam-se:

- resumos curtos, objetivos e fiéis ao conteúdo técnico;
- categorização automática em *tags* temáticas para melhor busca;
- em uma análise manual de 10 *write-ups* aleatórios, a IA identificou corretamente as vulnerabilidades em 100% dos casos;
- extração precisa de ferramentas citadas;
- classificação de dificuldade alinhada ao nível técnico do desafio de forma consistente e precisa.

A tradução automática produziu textos de boa qualidade, preservando blocos de código e estrutura original. Mesmo assim, em *write-ups* longos e altamente técnicos, pequenas perdas de fidelidade semântica são esperadas, especialmente em idiomas que não se utilizam de alfabeto latino.

A geração da explicação didática reforçou a utilidade da plataforma como ferramenta educacional. O modelo conseguiu resumir conceitos avançados de forma acessível. Também focou em explicações que não infringiam os termos de uso das IAs utilizadas, já que temas de ataque são extremamente sensíveis para IAs.

5.4 Experiência do Usuário no Frontend

O *frontend* CTF Bridge apresentou navegação fluida e boa organização visual, permitindo ao usuário acessar *write-ups*, metadados e resumos com poucos cliques. Entre os resultados observados:

- carregamento rápido devido ao uso de Vite e componentes React otimizados: em checagem pelo PageSpeed Insights, o First Contentful Paint foi de apenas 0.5 s;
- visualização clara dos *write-ups*, com metadados destacados e leitura confortável;
- sistema de filtros que facilita encontrar desafios por dificuldade, categoria ou evento;
- acesso direto ao *write-up* original sempre que disponível;
- explicação didática como diferencial na compreensão do material.

Os testes realizados demonstraram que a plataforma é intuitiva e acessível mesmo para usuários que não têm familiaridade com o tema. A organização em cartões na página inicial e a estrutura limpa na página de *write-up* contribuem para uma experiência agradável.

5.5 Observações sobre Eficiência e Custo

O uso de serviços *serverless* da AWS mostrou-se eficiente e financeiramente vantajoso. Durante o desenvolvimento e execução das rotinas diárias de *scraping* e enriquecimento, observou-se:

- baixo consumo de memória, CPU e armazenamento pelas Lambdas: Em termos de armazenamento, todo o código ocupa apenas 26,6kb, enquanto de memória são configuradas em 128MB;
- execução rápida das funções: Scraping com média de 3500ms e 15000ms para enrichers complexos;
- custo total mensal irrisório devido ao modelo *pay-per-use* da AWS. Como a lambda é nosso principal mecanismo de ação e possuímos 1 milhão de chamadas gratuitas (em qualquer *tier*), os outros serviços juntos custaram US$0.05 para a execução e armazenamento.

A separação entre armazenamento bruto (S3) e dados estruturados (DynamoDB) permitiu consultas rápidas no *frontend*, mantendo alto desempenho mesmo com volume crescente de *write-ups*.

5.6 Limitações Identificadas

Apesar dos resultados positivos, algumas limitações foram identificadas:

- *write-ups* hospedados em páginas altamente dinâmicas podem ter conteúdo parcial devido à ausência de um renderizador JavaScript completo;
- a IA pode, ocasionalmente, interpretar incorretamente uma vulnerabilidade rara ou um termo muito específico, travando seu resumo para propósitos educacionais;
- traduções automáticas podem introduzir pequenas imprecisões em trechos altamente técnicos;
- links externos antigos ou expirados continuam representando um desafio inevitável, o que tentamos contornar pegando desafios diariamente e mantendo a centralização na própria plataforma.

Essas limitações não comprometem o funcionamento da plataforma, mas indicam pontos onde melhorias futuras podem expandir a precisão e alcance do sistema.

5.7 Discussão Geral

Os resultados mostram que a plataforma atinge seu objetivo central: reunir *write-ups* dispersos na internet, enriquecê-los automaticamente com metadados úteis e disponibilizá-los de forma estruturada para consulta. A combinação de *scraping*, recursão externa, normalização discursiva, enriquecimento com IA e interface intuitiva cria um *pipeline* robusto que simplifica o acesso a conteúdo técnico frequentemente difícil de localizar.

A solução se mostra especialmente relevante para estudantes e profissionais que buscam aprofundar habilidades em segurança da informação, oferecendo uma visão padronizada e acessível de desafios de CTF. O uso de IA amplia o valor agregado ao material, tornando a plataforma não apenas um repositório, mas também uma ferramenta didática.

6 Conclusão

O desenvolvimento desta plataforma para coleta, normalização, enriquecimento e disponibilização de *write-ups* de competições *Capture The Flag* permitiu consolidar uma solução completa e automatizada para um problema real enfrentado pela comunidade de segurança da informação: a dispersão, falta de padronização e dificuldade de acesso a conteúdos técnicos produzidos por participantes de competições ao redor do mundo.

A proposta inicial deste trabalho de reunir esses materiais em um único ambiente estruturado foi alcançada de forma satisfatória. O sistema construído demonstrou ser capaz de recuperar textos de diversas fontes, aplicar processamento avançado para extrair conteúdo útil, enriquecer as informações com metadados consistentes e disponibilizá-las de maneira acessível ao usuário final.

A utilização de uma arquitetura *serverless* mostrou-se particularmente vantajosa. O uso de funções AWS Lambda, Amazon S3, DynamoDB e API Gateway permitiu construir um *pipeline* modular, escalável e com baixo custo operacional, sem a necessidade de gerenciar servidores ou infraestrutura persistente. A independência entre as etapas também contribuiu para a resiliência e a simplicidade de manutenção do sistema.

O enriquecimento dos *write-ups* por meio de modelos de linguagem, especialmente o Gemini 2.5 Flash, representou um dos elementos centrais deste projeto. A capacidade do modelo de gerar resumos coerentes, identificar categorias temáticas, extrair vulnerabilidades e gerar explicações didáticas ampliou significativamente o valor informativo do material coletado. Isso permitiu transformar um conjunto de textos técnicos altamente heterogêneos em um repositório estruturado, útil tanto para fins acadêmicos quanto profissionais.

O *frontend* CTF Bridge desempenhou papel fundamental na apresentação dos resultados. A interface responsiva, clara e bem organizada possibilitou que o usuário navegue facilmente pelos *write-ups*, visualize resumos e explore metadados técnicos de forma intuitiva. A adição da seção “Help me understand” reforçou o caráter educacional da plataforma, tornando o conteúdo mais acessível mesmo para iniciantes em segurança da informação.

Ainda que o sistema tenha alcançado resultados expressivos, algumas limitações foram identificadas. A dificuldade em extrair conteúdo de páginas altamente dinâmicas, a dependência de links externos nem sempre estáveis e pequenas imprecisões ocasionais da IA em textos extremamente técnicos são aspectos que podem ser aperfeiçoados em versões futuras. Tais limitações, contudo, não comprometem a funcionalidade geral do sistema, mas apontam caminhos naturais para evolução da plataforma.

De forma geral, o trabalho demonstra que é possível unificar ferramentas modernas de *scraping*, *pipelines serverless* e modelos de inteligência artificial para resolver um problema real da comunidade de CTF. A plataforma representa um recurso valioso para estudantes, pesquisadores e profissionais de segurança, oferecendo uma forma prática, centralizada e

enriquecida de acessar materiais de alta relevância técnica.

Os objetivos definidos no início deste projeto foram atingidos, e os resultados obtidos reforçam o potencial de soluções automatizadas para organizar e transformar grandes quantidades de informação técnica dispersa. Espera-se que esta plataforma contribua não apenas como ferramenta prática, mas também como base para novos estudos e extensões, tornando o acesso a *write-ups* de CTF mais simples, eficiente e educacional.

7 Trabalhos Futuros

Embora a plataforma desenvolvida já ofereça uma solução completa para coleta, enriquecimento e disponibilização de *write-ups* de competições desse meio, há diversas oportunidades de expansão e aprimoramento que podem ampliar significativamente sua utilidade e alcance.

7.1 Coleta Direta de Desafios nos Sites das Competições

Atualmente, o sistema depende principalmente do CTFtime e dos links fornecidos pelos próprios participantes. Um avanço natural seria a capacidade de coletar não apenas *write-ups*, mas também os *desafios* diretamente das plataformas oficiais das competições. Muitas competições disponibilizam páginas próprias contendo a descrição completa dos desafios, arquivos anexos e, em alguns casos, dicas adicionais. Realizar essa coleta direta permitiria:

- registrar desafios mesmo quando nenhum *write-up* tiver sido publicado;
- comparar soluções de *write-ups* com a descrição original;
- enriquecer a base de dados com informações adicionais sobre contexto e intenção do desafio.

Com isso, a plataforma poderia se tornar um repositório completo tanto de problemas quanto de soluções, oferecendo maior valor educacional e histórico.

7.2 Servidores Temporários para Execução de Desafios

Muitos desafios de CTF dependem de serviços hospedados temporariamente pelos organizadores. Assim que a competição termina, esses serviços geralmente deixam de existir, o que inviabiliza a reprodução completa do ambiente por parte de estudantes e pesquisadores.

Uma evolução importante seria a criação de uma infraestrutura de **servidores sob demanda**, capaz de:

- levantar contêineres temporários baseados nas imagens originais dos desafios;
- subir serviços completos com o mesmo comportamento dos ambientes da competição;
- permitir que usuários reproduzam e testem os ataques descritos nos *write-ups*.

Esse recurso transformaria a plataforma em um ambiente *hands-on* completo, aproximando-se de soluções profissionais de laboratórios de segurança.

7.3 Integração com Outras Plataformas e Ecossistemas de CTF

Outra direção de evolução está na ampliação do ecossistema da plataforma. É possível integrar o sistema com:

- bancos de dados de competições além do CTFtime;
- plataformas de treinamento como picoCTF, Hack The Box, Root-Me e CyberDefenders;
- APIs oficiais de competições que disponibilizem dados estruturados;
- ferramentas acadêmicas para análise curricular ou recomendação de trilhas de estudo.

Essa integração ampliaria o alcance da plataforma e permitiria comparações entre diferentes competições, eventos e estilos de desafios.

7.4 Coleta Automática e Armazenamento dos Arquivos dos Desafios

Além da coleta de *write-ups*, a plataforma poderia incorporar mecanismos para baixar automaticamente:

- anexos dos desafios (binários, scripts, *pcaps*, imagens, *dumps* de memória);
- contêineres ou imagens Docker oferecidas pelos organizadores;
- arquivos auxiliares necessários para a resolução.

Infelizmente, é comum encontrar desafios em que esses arquivos foram completamente perdidos, por não terem sido preservados pelos organizadores do CTF após o término do evento, nem pelos autores dos *write-ups*.

O armazenamento desses arquivos, combinado com a coleta de desafios e a disponibilidade de serviços sob demanda, facilitaria a replicação completa da experiência das competições originais.

7.5 Aprimoramento da Análise Automática

A análise com IA pode ser expandida para fornecer resultados ainda mais sofisticados, incluindo:

- detecção automática de técnicas de exploração avançadas;
- classificação hierárquica mais precisa de vulnerabilidades;
- identificação de padrões recorrentes entre competições;
- geração automática de trilhas de aprendizado personalizadas com base no perfil do usuário.

Modelos mais avançados também poderiam gerar visualizações do fluxo de ataque, diagramas de arquitetura e mapas conceituais para facilitar o estudo.

7.6 Expansão do Frontend

No *frontend*, diversas melhorias podem ser exploradas, tais como:

- *dashboards* dinâmicos com estatísticas sobre categorias e vulnerabilidades mais recorrentes;
- comparações entre competições ao longo dos anos;
- recomendações personalizadas para o usuário;
- modo de estudo guiado com foco em progressão de dificuldade.

Essas adições tornariam o CTF Bridge não apenas um catálogo, mas uma plataforma educacional completa.

7.7 Aumento da Escalabilidade e Complexidade do Pipeline

Por fim, o *pipeline serverless* pode ser expandido para lidar com volumes muito maiores de dados, incorporando:

- filas SQS para desacoplamento entre etapas;
- *step functions* para orquestração do fluxo;
- lambdas paralelas para processar *write-ups* massivamente;
- *cache* inteligente na camada da API.

Essas melhorias aumentam a complexidade, mas também reforçam a escalabilidade e a robustez do sistema.

7.8 Considerações Finais

As possibilidades apresentadas demonstram que o projeto possui espaço significativo para expansão. A plataforma atual constitui uma base sólida tanto do ponto de vista técnico quanto arquitetural, permitindo a implementação progressiva de novas funcionalidades que podem transformar o CTF Bridge em uma ferramenta ainda mais completa, educacional e alinhada às necessidades da comunidade de segurança da informação.

Referências

1 ANDERSON, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*. 2. ed. Hoboken: Wiley, 2008.

2 KREBS, B. *Spam Nation: The Inside Story of Organized Cybercrime*. Naperville: Sourcebooks, 2014.

3 LEUNE, K.; JR, S. J. P. Using capture-the-flag to enhance the effectiveness of cybersecurity education. In: *Proceedings of the 18th Annual Conference on Information Technology Education (SIGITE '17)*. New York, NY, USA: ACM, 2017. p. 47–52.

4 CHOTHIA, T.; NOVAKOVIC, C. An offline capture the flag-style virtual machine and an assessment of its value for cybersecurity education. In: *Proceedings of the 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE '15)*. Washington, D.C.: USENIX Association, 2015.

5 CHAPMAN, P.; BURKET, J.; BRUMLEY, D. PicoCTF: A game-based computer security competition for high school students. In: *Proceedings of the 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE '14)*. San Diego, CA: USENIX Association, 2014.

6 WERTHER, J. et al. Experiences in cyber security education: The MIT Lincoln Laboratory capture-the-flag exercise. In: *Proceedings of the 4th Workshop on Cyber Security Experimentation and Test (CSET '11)*. San Francisco, CA: USENIX Association, 2011.

7 KONDO, L. S. d. O. D. et al. Casa de ferreiro, o espeto não é de pau: evoluindo uma plataforma segura para competições de segurança. In: *Anais do XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*. Porto Alegre: SBC, 2022. p. 1–14.

8 ZHAO, W. X. et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023. Disponível em: <https://arxiv.org/abs/2303.18223>.

9 AYUSO, E.; BROGYA, M. S. D.; AHLAWAT, V. K. From manual to machine: How AI is redefining web scraping for superior efficiency: A literature review. In: *2024 International Conference on Automation, Control, and Intelligent Systems (ACIS)*. [S.l.]: IEEE, 2024.

10 BODOR, A.; HNIDA, M.; DAOUDI, N. Integration of web scraping, fine-tuning, and data enrichment in a continuous monitoring context via large language model operations. *International Journal of Electrical and Computer Engineering (IJECE)*, v. 15, n. 1, p. 1027–1037, 2025.

11 CIRILLO, S. et al. Augmenting anonymized data with AI: Exploring the feasibility and limitations of large language models in data enrichment. *arXiv preprint arXiv:2504.03778*, 2025. Disponível em: <https://arxiv.org/abs/2504.03778>.

12 ZHANG, B.; SOH, H. Extract, define, canonicalize: An LLM-based framework for knowledge graph construction. *arXiv preprint arXiv:2404.03868*, 2024. Disponível em: <https://arxiv.org/abs/2404.03868>.