Scanners de Vulnerabilidades em Aplicações Web Modernas: Uma Comparação Baseada em Cobertura, Validação de Falsos Positivos e CVSS

Thiago Paim Escarrone[1]

[1] Universidade Federal do Pampa (UNIPAMPA)

Resumo. *Neste estudo, realizamos uma análise comparativa abrangente de scanners de vulnerabilidades para aplicações web, atendendo à demanda de três empresas parceiras que buscam avaliar a eficácia de soluções comerciais e gratuitas. Como principais contribuições, destacam-se: (1) a validação manual de todas as vulnerabilidades reportadas, eliminando falsos positivos; (2) a avaliação de dez ferramentas (incluindo OWASP ZAP, Burp Suite Pro, Nuclei e Qualys Community Edition) com métricas de cobertura, eficácia e severidade (CVSS 3.1); e (3) testes em ambientes diversificados, como o OWASP Juice Shop (benchmark padrão) e uma aplicação realista (GAUCHA) com arquitetura SPA e rotas dinâmicas em JavaScript.*

1. Introdução

A literatura sobre a eficácia de ferramentas de varredura de vulnerabilidades (*i.e.*, *scanners*) em aplicações web apresenta pelo menos três limitações importantes. Primeiro, grande parte das pesquisas concentra-se em contextos restritos ou em scanners aparentemente descontinuados, ou que carecem de manutenção e suporte contínuos, como Sqlmap, Uniscan, XSSScan, RFuzz, XSSFuzz, RegFuzzer e Atropos [Khan et al. 2023, Shahid et al. 2022, Altulaihan et al. 2023, Güler et al. 2024]. A segunda limitação é a rara utilização de aplicações alvo que podem ser consideradas benchmarks amplamente reconhecidos pela comunidade, como o OWASP Juice Shop. Essa omissão impede a generalização robusta dos resultados e dificulta a formulação de recomendações precisas sobre as melhores combinações de scanners para diferentes cenários de segurança cibernética [Appiah et al. 2018, Holík and Neradova 2017, Aydos et al. 2022, Zangana 2024, Kollepalli et al. 2024]. Por fim, a maioria dos trabalhos restringe sua análise a um número reduzido de scanners (e.g., tipicamente entre 4 e 6 majoritariamente gratuitos) [Shah 2020, Albahar et al. 2022, Idrissi et al. 2017]. Um survey recente [Aydos et al. 2022] corrobora essas limitações, apontando uma escassez de estudos focados em ferramentas e na atualização contínua de análises de segurança.

Em trabalho recente [Rosa et al. 2024], identificamos que scanners gratuitos isolados não conseguem abranger adequadamente as vulnerabilidades em aplicações Web estáticas, exigindo combinações de múltiplas ferramentas para alcançar uma cobertura satisfatória.

A avaliação comparativa entre soluções gratuitas e pagas fornece insights para decisões estratégicas de cibersegurança, equilibrando cobertura, eficiência e custos operacionais[1]. Em síntese, este estudo avança em cinco dimensões: (1) análise comparativa

[1] https://www.wiz.io/academy/devsecops-tools/

atualizada de 10 scanners de vulnerabilidades; (2) validação manual rigorosa de todas as vulnerabilidades para exclusão dos falsos positivos; (3) desenvolvimento e avaliação em uma aplicação *Single Page Application* criada especificamente para o estudo; (4) classificação da severidade das vulnerabilidades confirmadas através do Common Vulnerability Scoring System (CVSS); e (5) unificação das vulnerabilidades para cálculo de recall relativo, permitindo comparação quantitativa do desempenho dos scanners.

Esta abordagem multidimensional permite avaliar a eficácia das ferramentas frente aos desafios das arquiteturas web modernas. No estudo, avaliamos oito ferramentas gratuitas (GoLismero, Nikto, Nuclei, OpenVAS, SecretScanner, Wapiti, OWASP ZAP e Qualys Community Edition) e duas soluções comerciais (Tenable Web App Scan e Burp Suite Professional), estas últimas escolhidas por estarem disponíveis para uso ao longo do estudo. Adotamos como alvos a aplicação estática OWASP Juice Shop e o sistema dinâmico GLPI (Gestionnaire Libre de Parc Informatique)[2], uma solução open source amplamente utilizada para gestão de ativos e chamados técnicos. Para os fins deste estudo, foi utilizada uma versão adaptada do GLPI, personalizada pela Universidade Federal do Pampa (UNIPAMPA), renomeada como GAUCHA – Gestão Administrativa e Unificada de Chamados. A versão de homologação foi disponibilizada pela universidade durante a realização dos testes, representando um ambiente dinâmico mais próximo de sistemas corporativos reais.

2. Fundamentos Teóricos

Esta seção apresenta os conceitos fundamentais necessários para compreender a análise comparativa de scanners de vulnerabilidades em aplicações web modernas. Abordamos desde os princípios básicos de segurança em aplicações web até os sistemas de classificação de vulnerabilidades, estabelecendo a base teórica para a metodologia e análise dos resultados.

2.1. Single Page Applications (SPAs)

Single Page Applications (SPAs) representam um paradigma contemporâneo no desenvolvimento de aplicações web, caracterizado pela execução da lógica de apresentação e navegação inteiramente no lado cliente através de JavaScript. Esta abordagem difere fundamentalmente da arquitetura tradicional multipágina, pois opera mediante a atualização dinâmica de componentes da interface sem recarregamento completo do documento HTML. A natureza dinâmica das SPAs [Garrett 2005] implica que grande parte da estrutura de navegação e dos endpoints da aplicação são gerados programaticamente durante a execução, frequentemente mediante interações do usuário ou respostas de APIs RESTful. Esta característica introduz desafios significativos para ferramentas automatizadas [Doupé et al. 2012, Alenezi and Almomani 2021] de varredura de vulnerabilidades, uma vez que a detecção de rotas e recursos depende criticamente da capacidade de interpretação e execução de código JavaScript complexo, bem como da simulação precisa do comportamento do usuário final.

O comportamento dinâmico das rotas em SPAs manifesta-se particularmente em cenários onde a acessibilidade a determinados recursos está condicionada a estados específicos da aplicação ou a processos de autenticação e autorização. Tais rotas, frequente-

[2] https://glpi-project.org/pt-br/

mente implementadas através de mecanismos de roteamento client-side, não são explicitamente declaradas no HTML inicial, mas sim construídas durante a execução com base na lógica da aplicação. Esta particularidade arquitetural resulta em uma superfície de ataque parcialmente oculta [Stock et al. 2019] para scanners convencionais que se baseiam predominantemente em análise estática de conteúdo ou em crawlers que não reproduzem adequadamente a interação humana com a aplicação. Consequentemente, vulnerabilidades associadas a estes componentes dinâmicos podem permanecer indetectadas durante processos automatizados de avaliação de segurança, exigindo abordagens complementares que combinem análise dinâmica avançada com testes manuais especializados para uma cobertura abrangente.

Como parte deste estudo, desenvolvemos uma aplicação experimental baseada em Django e React, especificamente para avaliar essas limitações. A aplicação simula um cenário moderno com características típicas de SPAs, como a geração dinâmica de rotas API via JavaScript e a renderização condicional de componentes. Essa abordagem nos permitiu testar se os scanners avaliados são capazes de interpretar corretamente contextos dinâmicos, ampliando a análise além da simples varredura estática. Conforme apontado por [Holík and Neradova 2017], essa é uma limitação recorrente em scanners que não executam código JavaScript ou não simulam interações de usuários, comprometendo a eficácia em detectar vulnerabilidades escondidas em rotas acessadas apenas dinamicamente.

2.2. Common Vulnerability Scoring System (CVSS)

O *Common Vulnerability Scoring System* (CVSS) é um padrão aberto e amplamente adotado para avaliação quantitativa da severidade de vulnerabilidades em sistemas computacionais. Desenvolvido pelo *Forum of Incident Response and Security Teams* (FIRST), o CVSS fornece uma estrutura padronizada para classificar falhas de segurança com base em métricas objetivas, permitindo a priorização de riscos em ambientes corporativos. Além de sua ampla adoção por instituições como o NIST, por meio da *National Vulnerability Database* (NVD) [National Institute of Standards and Technology 2024], o CVSS tem sido analisado criticamente na literatura como métrica relevante para aplicações em ambientes modernos e distribuídos [Almorsy et al. 2020].

O Common Vulnerability Scoring System (CVSS) versão 3.1 estrutura sua avaliação em três grupos de métricas: (a) as métricas base, que avaliam características intrínsecas da vulnerabilidade, considerando fatores como vetor de ataque, complexidade de exploração e impacto técnico nos pilares de confidencialidade, integridade e disponibilidade; (b) as métricas temporais, que incorporam aspectos dinâmicos como a maturidade de exploits disponíveis e o nível de correções existentes; e (c) as métricas ambientais, que ajustam a pontuação conforme características específicas do ambiente onde a vulnerabilidade está inserida, como requisitos de segurança organizacional e medidas de mitigação já implementadas.

Para este estudo, optou-se pela utilização exclusiva das métricas base, decisão fundamentada na necessidade de manter uma avaliação padronizada e comparável entre as diferentes ferramentas de varredura analisadas. Como o foco recai sobre a capacidade de detecção dos scanners em ambientes controlados (Juice Shop e GAUCHA), fatores contextuais avaliados pelas métricas temporais e ambientais - que variam conforme

condições operacionais específicas - tornam-se irrelevantes para os objetivos da pesquisa. As métricas base, por sua vez, permitem uma análise objetiva da severidade técnica das vulnerabilidades, isolando características intrínsecas que são diretamente relacionadas ao desempenho das ferramentas de varredura, sem a influência de variáveis externas que poderiam enviesar os resultados comparativos.

A versão 3.1 do CVSS, utilizada neste estudo, avalia vulnerabilidades pela métrica Base de acordo com estes grupos principais:

- **Explorabilidade** (Facilidade de Exploração):
  - *Attack Vector*: Contexto de exploração (Network/Adjacent/Local/Physical)
  - *Attack Complexity*: Condições necessárias para exploração
  - *Privileges Required*: Nível de privilégios exigidos
  - *User Interaction*: Necessidade de interação do usuário
- **Impacto Técnico**:
  - Confidencialidade (*Confidentiality*): Acesso a informações sensíveis
  - Integridade (*Integrity*): Alteração não autorizada de dados
  - Disponibilidade (*Availability*): Interrupção de serviços

A pontuação final, que varia de 0.0 a 10.0, categoriza as vulnerabilidades em cinco níveis de severidade:

- **Critical (9.0–10.0)**: Falhas com alto potencial de exploração e impacto devastador (ex.: execução remota de código)
- **High (7.0–8.9)**: Vulnerabilidades que comprometem significativamente a segurança, mas exigem condições específicas para exploração
- **Medium (4.0–6.9)**: Riscos moderados, muitas vezes limitados por controles de segurança ou contextos restritos
- **Low (0.1–3.9)**: Impacto mínimo, geralmente relacionado a vazamento de informações não sensíveis
- **None/Informative (0.0)**: Vulnerabilidades sem impacto prático comprovado

A classificação por meio do CVSS 3.1 permite priorizar vulnerabilidades de forma objetiva, direcionando esforços de correção para as falhas mais críticas identificadas nos sistemas analisados. Esta abordagem padronizada não apenas facilita a comparação entre os resultados dos diferentes scanners, mas também se alinha às melhores práticas da indústria para avaliação de riscos em segurança cibernética.

3. Metodologia e Ambiente de Testes

Este estudo adota uma abordagem de cibersegurança ofensiva, centrada na análise comparativa de ferramentas de varredura para aplicações web. Utilizou-se uma estratégia de teste de caixa preta (*black-box*) [Althunayyan et al. 2022], simulando o comportamento de um atacante externo, sem acesso prévio ao código-fonte das aplicações. O objetivo foi avaliar a capacidade das ferramentas em detectar vulnerabilidades reais, com ênfase na cobertura (quantidade e variedade de falhas identificadas), precisão (validação manual dos achados para eliminar falsos positivos)[3], severidade através do CVSS [4], relevância prática (desempenho diante de características modernas, como rotas dinâmicas em SPAs) [Holík and Neradova 2017].

[3] https://owasp.org/www-project-web-security-testing-guide/

[4] https://www.first.org/cvss/v3-1/specification-document

3.1. Seleção das ferramentas

A condução do experimento seguiu uma sequência estruturada de etapas, iniciando pela seleção dos scanners a serem avaliadas. Em seguida, foi realizada a configuração de um ambiente controlado de testes, garantindo uniformidade na execução das ferramentas e na exposição às aplicações vulneráveis. As varreduras foram então executadas individualmente, com parâmetros padronizados para mitigar viés nos resultados. A seleção das ferramentas de varredura utilizadas neste estudo foi guiada por critérios previamente definidos, com base em sua relevância prática e aderência ao uso em ambientes corporativos. Foram considerados aspectos como frequência de atualizações, tipo de licença (código aberto ou comercial), grau de adoção na indústria e compatibilidade com aplicações modernas. Essa abordagem permitiu a inclusão tanto de soluções gratuitas (Nuclei, OWASP ZAP, GoLismero, Nikto, Wapiti, OpenVAS, SecretScanner e Qualys Community Edition) amplamente utilizadas quanto de ferramentas comerciais (Burp Suite Professional e o Tenable Web App Scan) utilizadas e disponibilizadas por empresas parceiras, possibilitando uma análise comparativa abrangente sobre cobertura, precisão e aplicabilidade em cenários reais. A Tabela 1 resume as características técnicas das dez ferramentas de varredura avaliadas neste estudo, abrangendo soluções open-source e comerciais. Essa diversidade permite uma análise comparativa robusta, considerando desde ferramentas acessíveis até soluções profissionais integradas a pipelines de segurança. A tabela serve como referência para as discussões sobre desempenho e aplicabilidade em cenários reais.

Tabela 1. Características das ferramentas de varredura avaliadas

Ferramenta	Licença	Interface	Tipo
Nuclei	Open Source	CLI	Scanner de templates
OWASP ZAP	Open Source	GUI/CLI	DAST Interativa
GoLismero	Open Source	CLI	Scanner híbrido
Nikto	Open Source	CLI	Scanner Web (HTTP)
Wapiti	Open Source	CLI	DAST
OpenVAS	Open Source	GUI/CLI	Vulnerability Manager
SecretScanner	Open Source	CLI	Scanner de segredos
Qualys Community Edition	Gratuita (cloud)	Web (SaaS)	DAST/VA
Tenable WAS	Comercial	Web (SaaS)	DAST Cloud
Burp Suite Pro	Comercial	GUI	Proxy/DAST manual

Nota sobre a coluna "Tipo": As ferramentas foram classificadas com base na abordagem principal de varredura. *DAST (Dynamic Application Security Testing)* refere-se a scanners que testam aplicações em execução, simulando interações reais. *Scanners de templates*, como o Nuclei, utilizam arquivos de configuração predefinidos para detectar padrões específicos. *Híbrido* indica que a ferramenta combina análise estática e dinâmica. *Vulnerability Manager* refere-se a ferramentas mais amplas, como o OpenVAS, voltadas à gestão e correlação de múltiplas fontes de vulnerabilidades.

3.2. Ambiente de teste

Como infraestrutura de testes, utilizamos três ambientes distintos para avaliação das ferramentas. As sete soluções de código aberto foram executadas em um ambiente virtualizado (Oracle VirtualBox 7.0.14/Kali Linux 2024.1) com alocação de 8GB RAM e 2

núcleos de CPU, enquanto o Burp Suite Professional operou em hardware dedicado (Windows 11, Intel Core i7-8ª geração, 32GB RAM), simulando condições reais de análise corporativa. As plataformas SaaS (Tenable Web App Scan e Qualys Community Edition) foram acessadas via interfaces web.

3.3. Aplicações alvo

Para este estudo, dois ambientes distintos foram utilizados como aplicação alvo: o OWASP Juice Shop, tradicionalmente empregado como benchmark de segurança [5], e a aplicação GAUCHA, uma adaptação do sistema GLPI feito pela Universidade Federal do Pampa, cujo uso foi disponibilizado exclusivamente durante a pesquisa. No caso do Juice Shop (versão 16.0.1), oito das dez ferramentas de varredura puderam ser executadas diretamente na nuvem, utilizando instâncias no Google Cloud Run via container Docker[6], garantindo acessibilidade remota e um ambiente controlado.

Os scanners OpenVAS e SecretScanner apresentaram limitações operacionais, funcionando apenas em redes locais. Para viabilizar sua avaliação, a aplicação alvo foi implantada em um container Docker local. No entanto, no ambiente GAUCHA - acessível publicamente na internet - essas ferramentas não puderam ser utilizadas devido à sua restrição de operação em rede local. O Tenable WAS também não foi incluído nos testes no GAUCHA devido a limitações de licenciamento.

Como parte do conjunto de testes, também foi utilizada a aplicação SPA baseada em Django/React mencionada na Seção 2.1. Essa aplicação foi integrada ao ambiente experimental com o objetivo de verificar quais scanners conseguem identificar rotas dinâmicas consumidas via JavaScript. Sua inclusão permitiu avaliar, na prática, a capacidade das ferramentas em lidar com aplicações modernas e simular interações reais em tempo de execução.

3.4. Categorização das vulnerabilidades

A análise inicial revelou significativas ambiguidades na classificação de vulnerabilidades, com fronteiras pouco definidas entre diversas categorias. O remapeamento de categorias foi realizado justamente para eliminar essas zonas cinzentas, estabelecendo delimitações claras entre os diferentes tipos de vulnerabilidades. Como mostrado abaixo, consolidamos as categorias onde as fronteiras eram mais tênues (como entre arquivos sensíveis e informações expostas) e mantivemos separadas aquelas com características técnicas bem distintas. Esta padronização permitiu classificar as vulnerabilidades de forma inequívoca, superando as ambiguidades da taxonomia original.

1. **Informações Expostas** consolida as categorias:
    - Arquivos Potencialmente Perigosos
      Descrição: Exposição de arquivos e informações técnicas que facilitam reconhecimento de vulnerabilidades
    - Arquivos Sensíveis
      Descrição: Exposição de arquivos contendo dados sensíveis
    - Informações Divulgadas
      Descrição: Divulgação não intencional de informações internas

[5] https://rafed.github.io/devra/posts/security/vulnerable-applications-for-practicing-pentesting/
[6] https://console.cloud.google.com/

2. **Informações Privadas** mantém sua categoria original:
   Descrição: Exposição de dados pessoais identificáveis (PII)
3. **Ataques de Injeção** preserva sua denominação:
   Descrição: Inserção de dados maliciosos para manipulação da lógica da aplicação
   Exemplos: SQLi, Command Injection, XSS
4. **Autenticação e Autorização** como categoria única:
   Descrição: Falhas nos mecanismos de controle de acesso
   Crítico: Principal vetor de ataques em aplicações web
5. **Configuração de Segurança** como categoria única:
   Descrição: Configurações incorretas ou ausentes que comprometem a proteção
   Recomendação: Implementar hardening seguindo benchmarks do setor
6. **Detecção de Serviços** combina:
   - Detecção de Serviços
     Descrição: Coleta de informações sobre servidores e tecnologias
   - Fingerprinting
     Descrição: Identificação de tecnologias e versões
     Atenção: Pode facilitar ataques direcionados
7. **Headers e Configurações Faltantes** como categoria única:
   Descrição: Ausência de cabeçalhos de segurança fundamentais
   Exemplo: Falta de CSP, HSTS ou X-Content-Type-Options
8. **Segurança de Protocolo** unifica:
   - Protocolos SSH
     Descrição: Falhas na configuração de protocolos como SSH, TLS/SSL
   - Segurança de Protocolo
     Descrição: Vulnerabilidades em implementações de protocolos

Essa abordagem não apenas facilitou a comparação entre as ferramentas, mas também reduziu a subjetividade na interpretação dos relatórios, reforçando a confiabilidade dos resultados obtidos.

3.4.1. Unificação de vulnerabilidades

Para garantir uma avaliação consistente e comparável entre os diferentes scanners, foi necessária a unificação das vulnerabilidades reportadas. Observou-se que uma mesma vulnerabilidade era frequentemente descrita com nomenclaturas distintas pelas diversas ferramentas, dificultando a análise direta dos resultados. Para contornar essa limitação, implementou-se um processo de normalização baseado em três etapas principais:

Primeiramente, todas as vulnerabilidades identificadas pelos scanners foram catalogadas em um banco de dados unificado. Em seguida, realizou-se uma análise técnica minuciosa para correlacionar as diferentes descrições que se referiam à mesma vulnerabilidade subjacente. Essa análise considerou fatores como: o componente afetado, o vetor de ataque, o impacto potencial e as condições necessárias para exploração. Por fim, as vulnerabilidades equivalentes foram agrupadas em entradas únicas no conjunto de referência.

3.5. Eficácia dos scanners via Recall Relativo

Para avaliar a capacidade de cada scanner em identificar vulnerabilidades existentes, adotou-se uma métrica de recall relativo, que compara as detecções individuais das ferramentas com o conjunto total de falhas confirmadas no estudo — independentemente de qual scanner as detectou. Essa abordagem é necessária devido à impossibilidade de mensurar falsos negativos (vulnerabilidades não identificadas por nenhuma ferramenta), mas ainda oferece um parâmetro válido para comparar a eficácia relativa entre as soluções. A métrica é definida como:

$$\text{Recall Relativo} = \frac{\text{Vulnerabilidades Confirmadas pelo Scanner}}{\text{Total de Vulnerabilidades Confirmadas}} \tag{1}$$

Um scanner ideal teria recall relativo igual a 1 (100%), indicando detecção completa das vulnerabilidades confirmadas. No entanto, na prática, mesmo ferramentas avançadas dificilmente atingem esse patamar devido a desafios como lógica dinâmica em SPAs ou vulnerabilidades contextuais (e.g., falhas de autenticação).

3.6. Severidade das vulnerabilidades

A classificação de severidade através do CVSS 3.1 foi adotada como metodologia central neste estudo por três razões fundamentais: (1) fornecer uma avaliação quantitativa padronizada que transcende a mera contagem de vulnerabilidades, (2) permitir a priorização inteligente de correções com base no risco real, e (3) habilitar comparações consistentes entre os scanners. Cada vulnerabilidade confirmada foi analisada segundo os vetores de métricas base - explorabilidade (Attack Vector, Attack Complexity, Privileges Required, User Interaction) e impacto técnico (Confidentiality, Integrity, Availability) - seguindo as diretrizes oficiais do CVSS. A pontuação resultante, categorizada em cinco níveis de severidade (Critical, High, Medium, Low e None), não apenas reflete o potencial dano intrínseco de cada falha, mas também considera a praticidade de exploração, criando assim um sistema de classificação multidimensional especialmente relevante para ambientes corporativos. Como demonstrado por [Janulevicius and Vasilecas 2017], esta abordagem supera as limitações de análises qualitativas ao converter características técnicas complexas em escores numéricos comparáveis, essencial para avaliar objetivamente o desempenho dos scanners em diferentes categorias de vulnerabilidade

4. Resultados

4.1. Cobertura de Categorias de Vulnerabilidades

Na Figura 1 e 2, mostramos o desempenho das ferramentas ao detectar vulnerabilidades nas oito categorias comentadas anteriormente. Observa-se que, em ambas as aplicações analisadas, nenhum dos scanners avaliados foi capaz de identificar vulnerabilidades em todas as categorias definidas.

Observamos que o Tenable WAS e o Qualys Community Edition alcançaram os melhores resultados quando executados na aplicação Juice Shop, cobrindo a maioria das categorias e identificando um bom número de vulnerabilidades. Ferramentas como o Wapiti tiveram desempenho mais limitado, enquanto o OWASP ZAP e o Nuclei apresentaram resultados intermediários, o que indica variações na eficácia dos *scanners* testados. Ao

comparar as soluções comerciais, o Tenable WAS demonstrou uma cobertura mais ampla e um volume maior de detecções, identificando vulnerabilidades em 7 das 8 categorias (87,5% de cobertura) contra 5 categorias (62,5%) do Burp Suite. A diferença foi notável no volume de vulnerabilidades confirmadas, com 61 achados válidos para o Tenable WAS e 16 para o Burp Suite Pro.

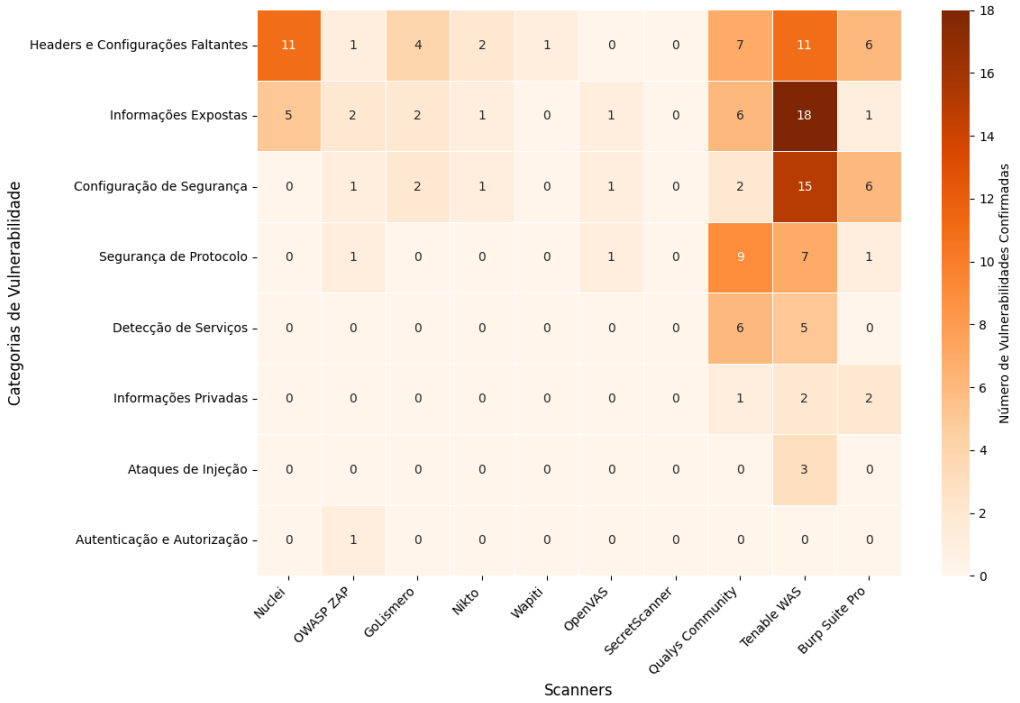

Figura 1. Número de vulnerabilidades confirmadas por categoria e scanner - Juice Shop

Os resultados obtidos na aplicação Juice Shop demonstram que o Tenable WAS se apresenta como solução ideal para grandes corporações, particularmente para provedores de serviços de segurança, devido à sua capacidade de automação em escala empresarial, ampla cobertura de vulnerabilidades e integração nativa com pipelines DevSecOps. Entretanto, sua faixa de preço superior a R$ 30 mil/ano pode ser proibitiva para pequenas e médias empresas, conforme dados do Sebrae sobre faixas de faturamento empresarial.

Como alternativa viável, o Burp Suite Professional (R$ 2,5 mil/ano) mostrou eficácia em testes pontuais, sendo particularmente adequado para empresas de médio porte. Nossos dados sugerem que sua combinação com o OWASP ZAP (open-source) forma uma solução economicamente sustentável para organizações com restrições orçamentárias, onde o ZAP complementa especialmente na detecção de vulnerabilidades de injeção - categoria em que o Burp Suite apresentou cobertura limitada.

O Qualys Community Edition emergiu como a solução gratuita mais robusta, detectando 31 vulnerabilidades e cobrindo 75% das categorias. Seu diferencial foi a identificação precisa de serviços ativos, funcionalidade crítica para a fase de reconhecimento que outras ferramentas gratuitas não performaram adequadamente. Contudo, sua incapacidade de detectar vulnerabilidades de injeção reforça a necessidade de abordagens híbridas mesmo no contexto de ferramentas gratuitas.

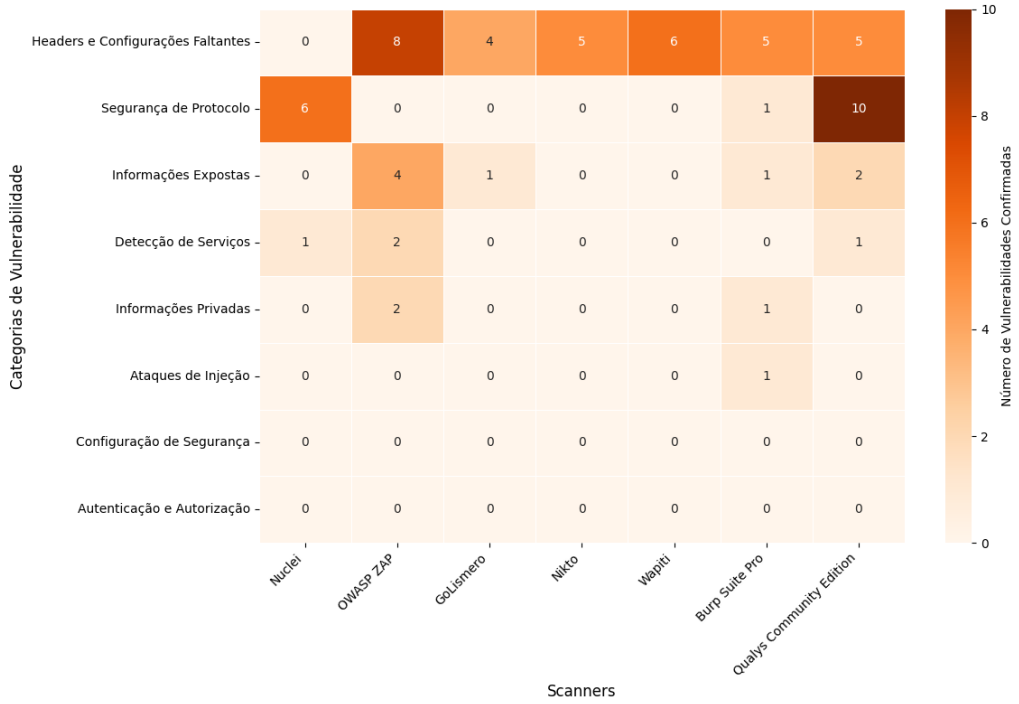

Figura 2. Número de vulnerabilidades confirmadas por categoria e scanner - GAUCHA

O scanner Qualys Community Edition, mesmo sendo uma solução gratuita, destacou-se novamente na aplicação GAUCHA, com 18 vulnerabilidades confirmadas. Seu resultado foi particularmente expressivo na categoria Segurança de Protocolo, onde identificou 10 falhas — superando até mesmo o Burp Suite Pro, ferramenta comercial consolidada no mercado. Esse desempenho reforça que soluções gratuitas, quando bem configuradas, podem oferecer resultados competitivos em ambientes corporativos reais.

O OWASP ZAP apresentou um perfil de detecção mais especializado, com 16 vulnerabilidades concentradas em apenas três categorias: Headers e Configurações Faltantes, Informações Expostas e Informações Privadas. Embora sua cobertura seja limitada, sua eficácia nessas áreas específicas o torna uma ferramenta valiosa para auditorias focadas em configurações básicas de segurança e vazamento de dados.

Um achado crítico foi a incapacidade generalizada dos scanners em detectar vulnerabilidades em Autenticação e Autorização, uma lacuna preocupante, dado que falhas nessa categoria estão entre as mais exploradas em ataques reais. Em contraste, a categoria Headers e Configurações Faltantes foi a mais coberta, com quase todas as ferramentas reportando falhas — exceto o Nuclei, que não identificou nenhuma. Essa disparidade sugere que os scanners automatizados ainda priorizam verificações superficiais, como ausência de headers de segurança, em detrimento de vulnerabilidades mais complexas e dependentes de contexto, como falhas em fluxos de autenticação.

4.2. Análise da eficácia dos scanners

A avaliação da eficácia dos scanners de vulnerabilidades é fundamental para determinar sua confiabilidade operacional em ambientes reais. Essa métrica — calculada pela razão entre vulnerabilidades confirmadas e o total de vulnerabilidades confirmadas por

todos os scanners — revela não apenas a capacidade das ferramentas de detectar falhas genuínas, mas também seu impacto na eficiência das equipes de segurança. Um scanner com uma baixa eficácia gera custos operacionais elevados, demandando tempo excessivo para triagem de falsos positivos, enquanto soluções mais eficientes permitem priorizar correções de forma ágil. Nesta seção, analisamos comparativamente a eficácia dos 10 scanners nas duas aplicações alvo.

Tabela 2. Eficácia dos Scanners nas Aplicações Juice Shop e GAUCHA

Ferramenta	Juice Shop	GAUCHA
Nuclei	0,15	0,16
OWASP ZAP	0,06	0,36
GoLismero	0,08	0,11
Wapiti	0,01	0,13
Qualys Community Edition	0,29	0,40
Nikto	0,04	0,11
Burp Suite Professional	0,15	0,20
Tenable WAS	0,58	-
SecretScanner	0,00	-
OpenVAS	0,03	-

O Tenable WAS demonstrou ser a ferramenta mais eficiente no ambiente controlado Juice Shop, com um índice de 0.58, destacando a eficácia de soluções comerciais em cenários padronizados. No entanto, sua ausência no GAUCHA — devido a restrições técnicas — limitou a avaliação em um ambiente realista, onde o Qualys Community Edition surpreendeu com resultado superior (0.40 contra 0.29 no Juice Shop). Esse salto sugere que o Qualys possui mecanismos mais adaptáveis a arquiteturas corporativas complexas. Já o OWASP ZAP exibiu variação extrema (de 0.06 para 0.36), indicando que sua eficiência depende criticamente do contexto: enquanto performou mal no Juice Shop, mostrou-se mais confiável no GAUCHA.

Ferramentas como Nuclei e Wapiti mantiveram baixa eficácia em ambos os ambientes (0.15-0.16 e 0.01-0.13, respectivamente), revelando uma incapacidade crônica de priorizar vulnerabilidades relevantes. Casos mais críticos foram o SecretScanner e o OpenVAS, com resultados próximos de zero no Juice Shop (0.00 e 0.03), o que inviabiliza seu uso prático em testes de segurança sem filtragem adicional. Esses resultados reforçam que, embora algumas ferramentas open-source sejam úteis para tarefas específicas, sua aplicação em larga escala exige complementação com soluções mais robustas ou validação humana extensiva. A disparidade de desempenho entre ambientes também ressalta a importância de testar scanners em múltiplos cenários antes de incorporar seu uso.

4.3. Validação de falsos positivos

A Figura 3 mostra os resultados da análise manual para identificar falsos positivos na aplicação Juice Shop. Observamos que o Tenable Was e o Burp Suite Pro apresentaram apenas um falso positivo cada, indicando que ambas as ferramentas, apesar de suas diferenças de abordagem (uma automatizada em nuvem e outra interativa local), podem fornecer resultados confiáveis com configuração adequada.

Por outro lado, o SecretScanner reportou oito vulnerabilidades, mas nenhuma foi confirmada, resultando em 100% de falsos positivos. Esse caso demonstra como algumas ferramentas podem produzir relatórios com muitos alertas incorretos, o que pode diminuir sua utilidade prática sem uma análise crítica adicional. Por sua vez, as ferramentas Nuclei e Wapiti se destacaram por não apresentarem falsos positivos em nossas análises.

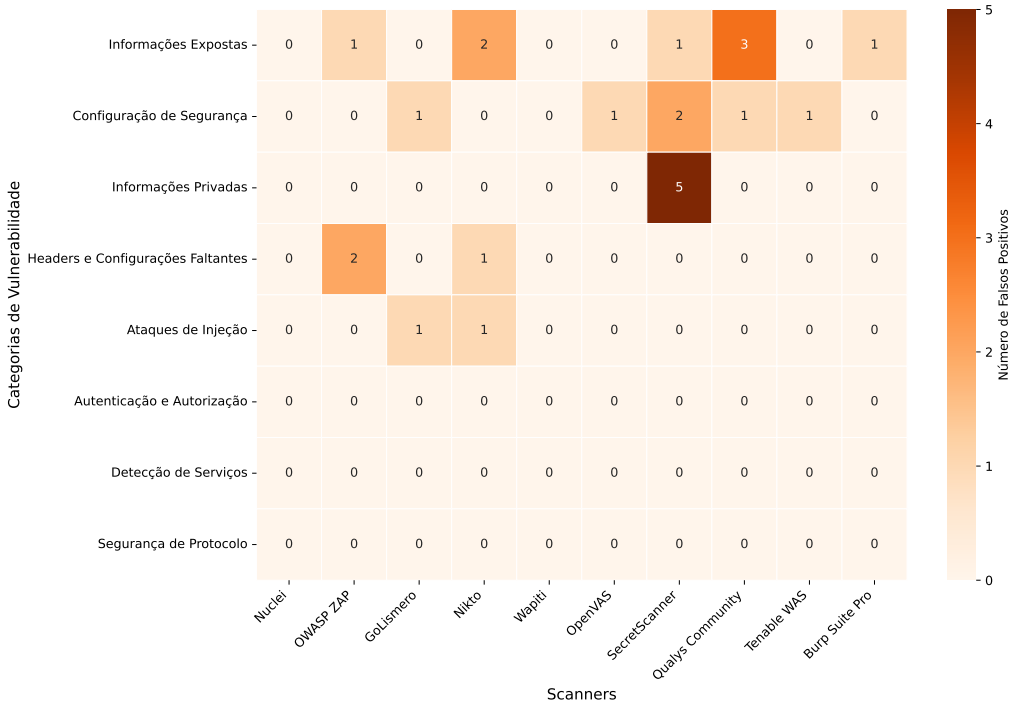

Figura 3. Número de falsos positivos por categoria e scanner- Juice Shop

Na análise da aplicação GAUCHA, os scanners OWASP ZAP e Nikto apresentaram um único falso positivo cada, demonstrando um bom nível de confiabilidade nos achados reportados. Uma possível explicação para esse baixo índice de erros pode estar relacionada à natureza das verificações realizadas por essas ferramentas, que tendem a focar em testes mais diretos e objetivos, como análise de headers e verificação de respostas HTTP padronizadas. Esse perfil reduz a chance de interpretações equivocadas sobre o comportamento da aplicação, especialmente em ambientes com respostas previsíveis como o GAUCHA, contribuindo para uma taxa de falsos positivos minimizada.

4.4. SPAs e rotas dinâmicas

Durante a avaliação no cenário com rotas geradas dinamicamente via JavaScript, apenas as ferramentas Tenable Web App Scan, OWASP ZAP e Burp Suite Professional foram capazes de identificar e interagir com a rota da API criada na nossa aplicação de testes. As demais ferramentas não conseguiram detectar a existência dessa rota, o que indica uma limitação relevante no motor de crawling ou na capacidade de interpretar e executar código JavaScript durante a análise. Uma possível causa para esse comportamento é o fato de muitos scanners realizarem apenas varreduras baseadas em HTML estático ou seguirem links diretamente visíveis no código-fonte, sem simular a interação do usuário ou renderizar a aplicação como um navegador faria. Esse tipo de limitação impacta diretamente a eficácia da análise em contextos modernos, nos quais dados e funcionalidades

são frequentemente carregados sob demanda por meio de chamadas assíncronas a APIs. A capacidade de alcançar essas rotas é fundamental para testes de segurança mais completos, especialmente em arquiteturas baseadas em SPAs ou microserviços, onde muitas vulnerabilidades críticas estão ocultas em endpoints acessados exclusivamente via JavaScript.

4.5. Classificação de Severidade por CVSS 3.1

A análise das vulnerabilidades detectadas revelou uma predominância de falhas classificadas como Medium, com distribuição heterogênea entre as ferramentas avaliadas. Essa classificação por severidade, baseada no critério CVSS 3.1 previamente detalhado na seção 2.2, permitiu identificar discrepâncias significativas na capacidade dos scanners de detectar vulnerabilidades de alto impacto, particularmente em categorias Critical ou High como autenticação e injeção de código.

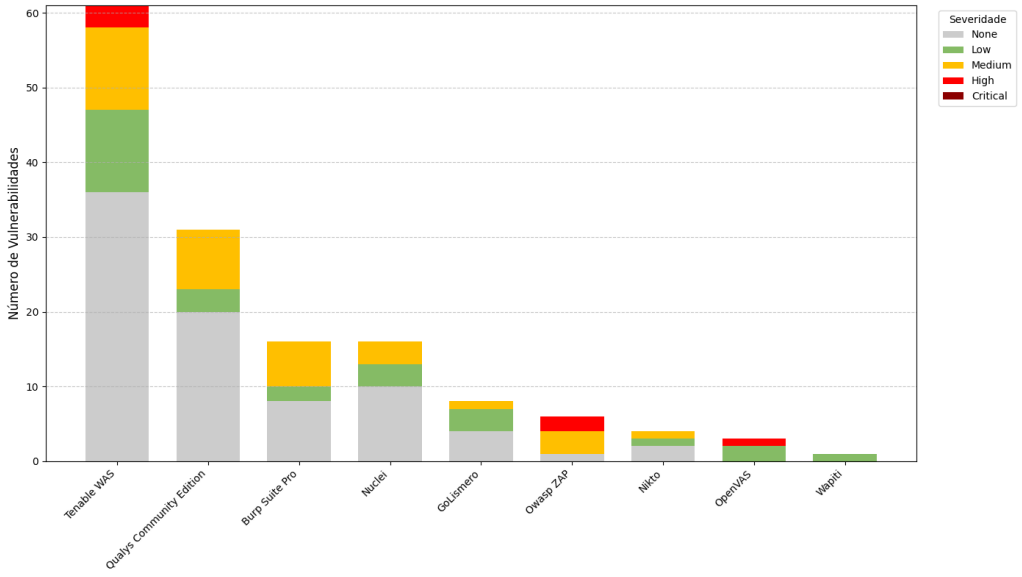

Figura 4. Severidade CVSS 3.1 em Juice Shop

Notamos que as ferramentas Tenable WAS e Qualys Community Edition apresentaram uma elevada quantidade de vulnerabilidades classificadas como None (59% e 65% de seus achados totais, respectivamente). Esse padrão indica uma estratégia de varredura focada em abrangência, que embora aumente a cobertura, demanda esforço adicional de validação para distinguir entre falsos positivos, achados informativos e vulnerabilidades efetivamente exploráveis. Tal característica impacta diretamente a eficiência operacional, sugerindo a necessidade de implementar processos de filtragem automatizada ou combinação estratégica de ferramentas.

Nenhuma das ferramentas avaliadas foi capaz de identificar vulnerabilidades classificadas como Critical no ambiente Juice Shop. Adicionalmente, apenas quatro scanners (Tenable WAS, Burp Suite, OWASP ZAP e OpenVAS) detectaram vulnerabilidades High, com um máximo de três ocorrências por ferramenta. Essa baixa efetividade na identificação de falhas de alta severidade - justamente as mais críticas em cenários reais de ataque - evidencia uma limitação metodológica significativa nas abordagens automa-

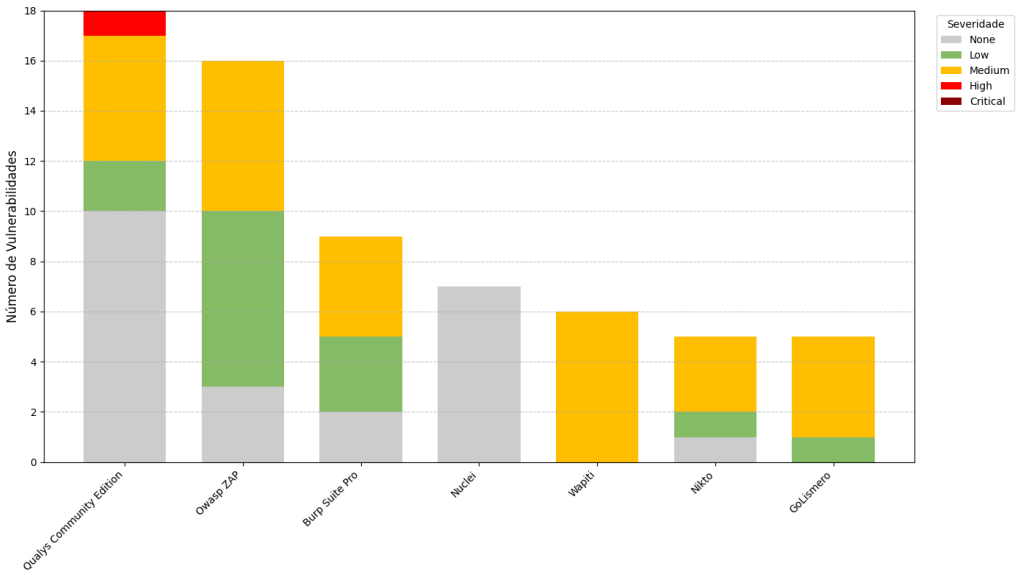

Figura 5. Severidade CVSS 3.1 em GAUCHA

tizadas atuais, particularmente para vulnerabilidades que demandam análise contextual complexa ou interação avançada com a aplicação.

O Qualys Community Edition destacou-se como a ferramenta com maior abrangência no ambiente GAUCHA, registrando 18 vulnerabilidades detectadas - incluindo a única classificação como High entre todos os scanners avaliados. Contudo, uma análise mais detalhada revela que 55% desses achados (10 de 18) foram categorizados como None, indicando um significativo volume de alertas com baixo impacto real na segurança do sistema. Esse padrão sugere que, embora o Qualys se mostre eficiente na cobertura ampla de possíveis falhas, sua utilização prática demanda um esforço considerável de triagem para distinguir entre vulnerabilidades relevantes e meros achados informativos.

Tanto o OWASP ZAP quanto o Burp Suite Pro demonstraram um equilíbrio notável entre quantidade e qualidade das detecções. O ZAP identificou 16 vulnerabilidades, com concentração nas classificações Low (7) e Medium (6), enquanto o Burp Suite Pro detectou 9 falhas, sendo 4 delas classificadas como Medium. Esse perfil de detecção indica que ambas as ferramentas, especialmente por suas características interativas, conseguem manter uma boa relação entre cobertura e precisão, minimizando a geração de ruído (alertas irrelevantes) comuns em scanners totalmente automatizados.

De modo geral, os resultados evidenciaram uma preocupante lacuna na capacidade dos scanners em identificar vulnerabilidades críticas - nenhuma ferramenta reportou falhas classificadas como Critical. Apenas o Qualys conseguiu detectar uma única vulnerabilidade High, enquanto todas as demais ignoraram completamente essa categoria de risco. Essa limitação, observada tanto no GAUCHA quanto no Juice Shop, reforça que as soluções automatizadas atuais apresentam deficiências significativas na identificação de vulnerabilidades complexas ou que demandam análise contextual mais aprofundada, destacando a importância da complementação com testes manuais especializados.

Tanto o Nuclei quanto o Wapiti apresentaram limitações significativas na avaliação

do sistema GAUCHA. O Nuclei detectou apenas 7 vulnerabilidades, todas classificadas como None, demonstrando uma baixa efetividade neste ambiente específico. De forma similar, o Wapiti identificou 6 falhas, porém concentradas exclusivamente na categoria Medium. Esse padrão divergente - onde o Nuclei reportou apenas achados irrelevantes e o Wapiti focou em vulnerabilidades de média severidade, mas sem cobrir outras categorias - sugere que ambas as ferramentas, quando utilizadas isoladamente, oferecem uma visão incompleta da postura de segurança do sistema. A ausência completa de detecções nas classificações High e Critical por ambas as ferramentas reforça a necessidade de complementá-las com outros métodos de avaliação em ambientes corporativos complexos como o GAUCHA.

5. Conclusão e Trabalhos Futuros

A análise revelou quatro limitações críticas nas ferramentas avaliadas: (1) a taxa de falsos positivos variou drasticamente, indo de 0% em ferramentas como Nuclei e Wapiti até 100% no SecretScanner, comprometendo a eficiência operacional; (2) apenas 30% dos scanners (Tenable, Burp Suite e ZAP) demonstraram capacidade efetiva de analisar Single Page Applications com rotas dinâmicas via JavaScript, deixando uma lacuna perigosa na cobertura de aplicações modernas; (3) a detecção de vulnerabilidades críticas mostrou-se deficiente, com nenhuma ferramenta identificando falhas classificadas como Critical e apenas quatro detectando algumas High; e (4) a categoria Autenticação e Autorização - fundamental para segurança corporativa - apresentou baixíssima cobertura, apesar de representar um dos vetores de ataque mais explorados na prática. Essas deficiências estruturais evidenciam que os scanners atuais ainda priorizam vulnerabilidades superficiais em detrimento de falhas complexas e contextuais, reforçando a necessidade imperativa de complementar varreduras automatizadas com testes manuais especializados, particularmente em ambientes com arquiteturas modernas e requisitos rigorosos de segurança.

Como trabalhos futuros, podemos expandir a variedade de aplicações testadas, incluindo cenários reais com diferentes arquiteturas e níveis de complexidade. Uma linha promissora é a realização de testes autenticados, que permitiriam avaliar a capacidade dos scanners em identificar falhas presentes em áreas restritas da aplicação, geralmente não acessíveis por crawlers não autenticados. Essa abordagem se aproxima mais de contextos reais, onde partes sensíveis do sistema — como painéis administrativos ou fluxos protegidos — concentram vulnerabilidades críticas que muitas vezes passam despercebidas em análises não autenticadas [Doupe et al. 2010].

Outro ponto importante a ser aprofundado é o tratamento de vulnerabilidades classificadas como *Low* pelo CVSS. Embora possam parecer de baixo risco quando analisadas isoladamente, essas falhas frequentemente servem como vetores de entrada ou pontos de apoio para ataques encadeados que resultam em compromissos graves da aplicação [Zetter 2018]. A inclusão de ferramentas baseadas em inteligência artificial, que conseguem analisar o contexto e identificar combinações de vulnerabilidades, pode contribuir significativamente para a detecção de cenários mais sofisticados e realistas [Shar and Tan 2021].

Referências

Albahar, M., Alansari, D., and Jurcut, A. (2022). An empirical comparison of pen-testing tools for detecting web app vulnerabilities. *Electronics*, 11(19).

Alenezi, M. and Almomani, A. (2021). Limitations of automated web application vulnerability scanners. *Computers & Security*, 100:102080.

Almorsy, M., Grundy, J., and Müller, I. (2020). An analysis of cvss-based vulnerability scores for cloud applications. *Journal of Systems and Software*, 170:110734.

Althunayyan, M., Saxena, N., Li, S., and Gope, P. (2022). Evaluation of black-box web application security scanners in detecting injection vulnerabilities. *Electronics*, 11(13):2049.

Altulaihan, E. A., Alismail, A., and Frikha, M. (2023). A survey on web application penetration testing. *Electronics*, 12(5).

Appiah, V., Asante, M., Nti, I. K., and Nyarko-Boateng, O. (2018). Survey of websites and web application security threats using vulnerability assessment. *Journal of Computer Science*, 15(10):1341–1354.

Aydos, M., Çiğdem Aldan, Coşkun, E., and Soydan, A. (2022). Security testing of web applications: A systematic mapping of the literature. *Journal of King Saud University - Computer and Information Sciences*, 34(9):6775–6792.

Doupé, A., Cavedon, L., Kruegel, C., and Vigna, G. (2012). Enemy of the state: A state-aware black-box web vulnerability scanner. In *USENIX Security Symposium*, pages 523–538. USENIX Association.

Doupe, A., Cova, M., and Vigna, G. (2010). Why johnny can't pentest: An analysis of black-box web vulnerability scanners. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, pages 111–131. Springer.

Garrett, J. J. (2005). Ajax: A new approach to web applications. *Adaptive Path*. Artigo seminal que introduziu o conceito de aplicações web assíncronas.

Güler, E., Schumilo, S., Schloegel, M., Bars, N., Görz, P., Xu, X., Kaygusuz, C., and Holz, T. (2024). Atropos: Effective fuzzing of web applications for {Server-Side} vulnerabilities. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 4765–4782.

Holík, F. and Neradova, S. (2017). Vulnerabilities of modern web applications. In *40th MIPRO*, pages 1256–1261. IEEE.

Idrissi, S. E., Berbiche, N., Guerouate, F., and Shibi, M. (2017). Performance evaluation of web application security scanners for prevention and protection against vulnerabilities. *International Journal of Applied Engineering Research*, 12(21):11068–11076.

Janulevicius, A. and Vasilecas, O. (2017). A comparison of vulnerability scoring systems for industrial web applications. *Computer Standards & Interfaces*, 54:50–57.

Khan, B., Bangash, J. I., Tariq, M., Gul, N., Zahir, S., and Kamal, A. (2023). A comparative model to analyze various web application penetration testing tools for different vulnerabilities. In *ICTAPP*, pages 1–6.

Kollepalli, R. P. K., Reddy, M. J. S., Sai, B. L., Natarajan, A., Mathi, S., and Ramalingam, V. (2024). An experimental study on detecting and mitigating vulnerabilities in web applications. *International Journal of Safety & Security Engineering*, 14(2).

National Institute of Standards and Technology (2024). National vulnerability database – cvss scoring. Online. https://nvd.nist.gov/vuln-metrics/cvss.

Rosa, R., Kreutz, D., Garcia, M., Pereira, S., and Mansilha, R. (2024). Análise empírica e comparativa de ferramentas de varredura de vulnerabilidades em aplicações web usando owasp bwa e juice shop. In *Anais da XXI Escola Regional de Redes de Computadores*, pages 183–188, Porto Alegre, RS, Brasil. SBC.

Shah, M. P. (2020). *Comparative analysis of the automated penetration testing tools*. PhD thesis, Dublin, National College of Ireland.

Shahid, J., Hameed, M. K., Javed, I. T., Qureshi, K. N., Ali, M., and Crespi, N. (2022). A comparative study of web application security parameters: Current trends and future directions. *Applied Sciences*, 12(8).

Shar, L. K. and Tan, H. B. K. (2021). Machine learning for security vulnerability detection: A survey. *Journal of Computer Security*, 29(3):301–351.

Stock, B., Pellegrino, G., Li, F., Backes, M., and Rossow, C. (2019). Hey, you have a problem: On the feasibility of large-scale web vulnerability notification. In *USENIX Security Symposium*, pages 2165–2182. USENIX Association.

Zangana, H. M. (2024). Exploring the landscape of website vulnerability scanners: A comprehensive review and comparative analysis. *Redefining Security With Cyber AI*, pages 111–129.

Zetter, K. (2018). Equifax's mistakes are now costing it millions in fines and a damaged reputation. https://www.wired.com/story/equifax-fine-identity-theft-settlement/. Wired Magazine.