Investigação de Vulnerabilidades em Aplicações Web de Instituição de Ensino

Diogo Leite Gomes, Carlos Alberto da Silva

[1]Faculdade de Computação – Universidade Federal de Mato Grosso do Sul (UFMS)
Cidade Universitária, Av. Costa e Silva - Pioneiros, MS, 79070-900

d.leite@ufms.br, carlos.silva@ufms.br

Abstract. *This paper presents an investigation of security vulnerabilities identified in web applications used by an educational institution. The study was conducted through reconnaissance, asset validation and vulnerability analysis techniques, using open-source tools such as Subfinder, httpx and WPScan. The objective was to identify exposed assets, technologies in use and known vulnerabilities associated with Common Vulnerabilities and Exposures (CVE). The results indicate the presence of multiple vulnerabilities in WordPress-based applications, including critical flaws with high CVSS scores. The study highlights the importance of continuous vulnerability management, software updates and proactive security practices.*

Resumo. *Este trabalho apresenta uma investigação de vulnerabilidades de segurança identificadas em aplicações web utilizadas por uma instituição de ensino. O estudo foi conduzido por meio de técnicas de reconhecimento, validação de ativos e análise de vulnerabilidades, utilizando ferramentas de código aberto como Subfinder, httpx e WPScan. O objetivo foi identificar ativos expostos, tecnologias utilizadas e vulnerabilidades conhecidas associadas a Common Vulnerabilities and Exposures (CVE). Os resultados indicam a presença de múltiplas vulnerabilidades em aplicações baseadas em WordPress, incluindo falhas críticas com altos índices classificadas pelo Common Vulnerability Scoring System (CVSS). O estudo ressalta a importância da gestão contínua de vulnerabilidades, atualização de componentes e adoção de práticas proativas de segurança.*

Palavras-chave: CVE, CVSS, *WordPress*, vulnerabilidades *web*, segurança da informação.

1. Introdução

Nos últimos anos, aplicações *web* passaram a desempenhar um papel fundamental na disponibilização de serviços digitais em instituições de ensino e organizações de diferentes setores. Sistemas acadêmicos, portais administrativos, plataformas educacionais e aplicações de gestão tornaram-se amplamente utilizados para armazenamento de informações, comunicação institucional e prestação de serviços online.

Com o aumento da exposição desses sistemas à internet, também houve um crescimento significativo na exploração de vulnerabilidades computacionais. Falhas relacionadas a componentes desatualizados, *plugins* vulneráveis e configurações inadequadas representam riscos relevantes para a segurança das aplicações *web*, podendo comprometer a confidencialidade, integridade e disponibilidade das informações.

Entre as tecnologias amplamente utilizadas no desenvolvimento de aplicações *web*, destaca-se o *WordPress*, um sistema de gerenciamento de conteúdo adotado em diversos contextos devido à sua flexibilidade e facilidade de utilização. Entretanto, sua popularidade também faz com que aplicações

baseadas nessa plataforma sejam frequentemente alvo de ataques cibernéticos, especialmente quando utilizam *plugins*, temas ou versões desatualizadas.

Nesse contexto, atividades de reconhecimento, validação de ativos e análise de vulnerabilidades tornam-se essenciais para a identificação preventiva de falhas de segurança. Técnicas utilizadas em análises de segurança permitem avaliar a superfície de ataque de aplicações *web* e identificar vulnerabilidades conhecidas associadas a bases públicas, como o *Common Vulnerabilities and Exposures* (CVE) [1].

Trabalhos anteriores desenvolvidos no âmbito da UFMS também abordaram a análise de vulnerabilidades em aplicações *web*. Um desses estudos investigou vulnerabilidades em aplicações *web* utilizadas por empresas de leilões online e instituições de Ensino a Distância (EAD), utilizando técnicas de *pentest* e ferramentas automatizadas para identificação, classificação e proposição de medidas de mitigação [2]. Outro trabalho realizou uma análise de vulnerabilidades em domínios *WordPress* de instituições de ensino, com foco na identificação de serviços *WordPress* vulneráveis por meio da ferramenta WPScan [3]. Esses trabalhos servem como referência metodológica para este estudo, que também concentra sua análise em vulnerabilidades *web* associadas a componentes *WordPress*.

Diante desse cenário, este trabalho tem como objetivo investigar vulnerabilidades presentes em aplicações *web* utilizadas por uma instituição de ensino, utilizando ferramentas de segurança para identificação de tecnologias expostas, *plugins* vulneráveis e falhas conhecidas classificadas pelo *Common Vulnerability Scoring System* (CVSS) [4].

Além disso, o trabalho busca realizar uma análise quantitativa das vulnerabilidades identificadas, observando sua recorrência entre diferentes aplicações *web* e destacando vulnerabilidades críticas que podem representar riscos relevantes para os sistemas analisados.

Por questões éticas e de confidencialidade, os nomes completos da instituição e dos domínios analisados serão parcialmente omitidos ao longo deste trabalho.

2. Fundamentação Teórica

Esta seção apresenta os conceitos fundamentais utilizados como base para a análise de vulnerabilidades *web* realizada neste estudo.

2.1. Segurança da Informação

A segurança da informação tem como objetivo proteger dados e sistemas contra acessos não autorizados, uso indevido, divulgação, modificação, destruição ou indisponibilidade. De acordo com o National Institute of Standards and Technology (NIST), essa proteção busca garantir os princípios de confidencialidade, integridade e disponibilidade [5]. A confidencialidade busca assegurar que apenas usuários autorizados tenham acesso às informações; a integridade garante que os dados não sejam modificados ou destruídos de forma indevida; e a disponibilidade assegura que os sistemas e informações estejam acessíveis quando necessário.

Em instituições de ensino, esses princípios são especialmente relevantes, uma vez que os sistemas podem armazenar dados acadêmicos, administrativos e informações pessoais de alunos, professores e servidores.

2.2. Common Vulnerabilities and Exposures

O *Common Vulnerabilities and Exposures* (CVE) é um sistema padronizado de identificação de vulnerabilidades de segurança conhecidas publicamente. Cada vulnerabilidade recebe um identificador

único, permitindo sua catalogação, consulta e compartilhamento entre pesquisadores, profissionais de segurança e organizações [1].

Neste trabalho, os identificadores CVE foram utilizados para padronizar a identificação das falhas encontradas nos sistemas analisados.

2.3. Common Vulnerability Scoring System

O *Common Vulnerability Scoring System* (CVSS) é um padrão utilizado para classificar a severidade de vulnerabilidades computacionais. A pontuação varia de 0 a 10, indicando o nível de criticidade da falha. Vulnerabilidades com pontuação mais alta tendem a representar maior risco, especialmente quando podem permitir execução remota de código, acesso indevido a dados ou comprometimento da aplicação [4].

Neste estudo, o CVSS foi utilizado para priorizar as vulnerabilidades de maior criticidade.

2.4. WordPress

O *WordPress* é um sistema de gerenciamento de conteúdo amplamente utilizado para criação e manutenção de sites. Sua popularidade está relacionada à facilidade de uso, variedade de temas e grande quantidade de *plugins* disponíveis. No entanto, a utilização de *plugins* desatualizados, temas vulneráveis e configurações inseguras pode ampliar significativamente a superfície de ataque de uma aplicação *web* [6].

2.5. Vulnerabilidades Web

Vulnerabilidades *web* são falhas de segurança que podem ser exploradas em aplicações acessíveis pela internet. Entre os tipos comuns estão a execução de *scripts* entre sites, conhecida como *Cross-Site Scripting* (XSS), a injeção de comandos SQL, conhecida como *SQL Injection*, a inclusão local de arquivos, conhecida como *Local File Inclusion* (LFI), a exposição de informações sensíveis e as falhas de controle de acesso.

Quando associadas a componentes populares, como *plugins WordPress*, essas vulnerabilidades podem afetar diversas aplicações que reutilizam as mesmas tecnologias.

3. Ferramentas Utilizadas

Nesta seção são apresentadas as ferramentas utilizadas para a condução da análise.

3.1. Oracle VM VirtualBox

O Oracle VM VirtualBox [7] foi utilizado como ferramenta de virtualização para criação e execução do ambiente de testes. Por meio dele, foi possível instalar e executar o Kali Linux em uma máquina virtual, isolando o ambiente de análise do sistema operacional principal e permitindo a utilização das ferramentas de reconhecimento e análise de vulnerabilidades de forma controlada.

3.2. Kali Linux

O Kali Linux [8] é uma distribuição Linux baseada em Debian voltada para testes de penetração, auditoria de segurança, computação forense e análise de vulnerabilidades. Neste trabalho, foi utilizado como ambiente principal para execução das ferramentas de segurança.

3.3. Subfinder

O Subfinder [9] é uma ferramenta utilizada para descoberta passiva de subdomínios. Ela permite ampliar a superfície de ataque conhecida de uma organização, identificando ativos que podem estar expostos publicamente.

Neste estudo, foi utilizado para levantar subdomínios relacionados ao domínio principal analisado.

3.4. httpx

A ferramenta httpx [10], desenvolvida pela ProjectDiscovery, foi utilizada para verificar quais subdomínios estavam ativos, identificar códigos de resposta HTTP e coletar informações sobre tecnologias utilizadas nas aplicações *web*.

3.5. WPScan

O WPScan [11] é uma ferramenta voltada para análise de segurança de aplicações *WordPress*. Ela permite identificar versões do *WordPress*, temas, *plugins* e vulnerabilidades conhecidas associadas aos componentes encontrados.

Neste trabalho, o WPScan foi utilizado nos ativos em que a presença de *WordPress* foi identificada, permitindo relacionar *plugins* e temas detectados com vulnerabilidades conhecidas em bases públicas de CVEs.

4. Metodologia

A metodologia adotada neste trabalho foi dividida em etapas sequenciais de reconhecimento, validação de ativos, identificação de tecnologias, análise de vulnerabilidades e classificação dos resultados.

4.1. Definição do Escopo

Inicialmente, foi definido o escopo da análise, concentrando-se em aplicações *web* pertencentes a uma instituição de ensino. Por questões éticas, os nomes completos dos domínios avaliados não são apresentados integralmente.

4.2. Preparação do Ambiente

Para a realização dos testes, foi configurado um ambiente virtualizado utilizando o Oracle VM VirtualBox. Nesse ambiente, foi instalada a distribuição Kali Linux, utilizada como sistema operacional principal para execução das ferramentas Subfinder, httpx e WPScan.

4.3. Reconhecimento e Coleta de Subdomínios

A primeira etapa consistiu no levantamento de subdomínios utilizando a ferramenta Subfinder. Essa etapa teve como objetivo identificar ativos expostos publicamente e ampliar a visão sobre a superfície de ataque da instituição.

4.4. Validação de Ativos Web

Após a coleta de subdomínios, foi utilizada a ferramenta httpx para verificar quais ativos estavam acessíveis via HTTP ou HTTPS. Nessa etapa, foram coletados códigos de resposta, títulos das páginas e tecnologias identificadas, como servidores *web*, linguagens, frameworks e sistemas de gerenciamento de conteúdo.

4.5. Identificação de Aplicações WordPress

A partir das tecnologias identificadas pelo httpx, os ativos que utilizavam *WordPress* foram selecionados para análise específica. Essa filtragem permitiu direcionar o uso do WPScan apenas aos sistemas compatíveis com a ferramenta.

4.6. Análise de Vulnerabilidades com WPScan

Nos ativos baseados em *WordPress*, foi executado o WPScan com uso de *token* de API, permitindo a identificação dos plugins instalados, dos temas utilizados, das versões em execução e das vulnerabilidades conhecidas associadas aos componentes detectados.

As vulnerabilidades identificadas foram relacionadas aos seus respectivos códigos CVE, permitindo a padronização da análise e a posterior classificação com base na pontuação CVSS.

4.7. Classificação e Organização dos Dados

As vulnerabilidades identificadas foram classificadas de acordo com seus respectivos identificadores CVE e pontuações CVSS. Em seguida, os resultados foram organizados em planilhas contendo o endereço IP, domínio associado, código CVE e quantidade de ocorrências.

Essa organização permitiu analisar tanto a criticidade individual das vulnerabilidades quanto sua recorrência entre as aplicações avaliadas.

5. Resultados

Durante a análise, foram identificadas vulnerabilidades associadas principalmente a aplicações *WordPress* que utilizavam plugins desatualizados. Entre os componentes vulneráveis encontrados destacaram-se *Elementor, Page Builder by SiteOrigin e Smart Slider 3*.

As vulnerabilidades identificadas incluem falhas de *Cross-Site Scripting* (XSS), *SQL Injection*, upload arbitrário de arquivos, leitura arbitrária de anexos, *Local File Inclusion* (LFI), *PHP Object Injection* e exposição de informações sensíveis.

Ao todo, foram identificadas 49 CVEs distintas. Algumas vulnerabilidades apresentaram alta criticidade, com pontuações CVSS de 9.9 e 9.8, indicando risco elevado caso sejam exploradas em condições adequadas.

Também foi observada a repetição de vulnerabilidades em mais de uma aplicação analisada. Em determinados casos, diferentes domínios estavam associados ao mesmo endereço IP, indicando possível uso de hospedagem compartilhada, *proxy* reverso ou *virtual hosts*. Mesmo nesses casos, a contagem foi realizada por aplicação analisada, pois cada domínio representa uma superfície de ataque distinta.

5.1. Vulnerabilidades de Maior Criticidade

A Tabela 1 apresenta as vulnerabilidades de maior criticidade identificadas durante a análise. Foram priorizadas as CVEs com maiores pontuações CVSS, considerando seu impacto potencial sobre as aplicações analisadas.

A partir dos resultados apresentados, observa-se que as vulnerabilidades de maior criticidade estão associadas principalmente a *plugins WordPress* desatualizados. A recorrência dessas falhas em mais de uma aplicação indica a reutilização de componentes vulneráveis em diferentes domínios analisados.

Tabela 1. Principais vulnerabilidades identificadas nas aplicações analisadas.

CVE	DESCRIÇÃO	CVSS	QTDE
CVE-2020-7055	Upload arbitrário de arquivos em versões vulneráveis do plugin Elementor.	Crítico (9.9)	1
CVE-2020-7109	Falha de Cross-Site Scripting (XSS) em versões vulneráveis do plugin Elementor.	Crítico (9.8)	2
CVE-2023-47504	Falha de autorização que pode permitir leitura arbitrária de anexos no plugin Elementor.	Crítico (9.8)	2
CVE-2020-13642	Vulnerabilidade relacionada a CSRF e XSS no plugin Page Builder by SiteOrigin.	Alto (8.8)	2
CVE-2022-3357	Falha de PHP Object Injection em versões vulneráveis do plugin Smart Slider 3.	Alto (8.8)	2
		Total de ocorrências	**9**

Embora algumas vulnerabilidades exijam autenticação para exploração, elas ainda representam risco relevante, principalmente quando combinadas com outros fatores observados durante a análise, como o uso de versões antigas de *plugins* e a presença de componentes vulneráveis em aplicações acessíveis publicamente.

5.2. Análise das Vulnerabilidades Críticas

As vulnerabilidades apresentadas na Tabela 1 representam as falhas de maior criticidade identificadas durante a análise. A seguir, são descritas as principais vulnerabilidades, suas possíveis consequências e recomendações de mitigação.

Vulnerabilidade 1 – CVE-2020-7055: Upload arbitrário de arquivos no Elementor. Esta vulnerabilidade está relacionada ao envio indevido de arquivos ao servidor por meio de versões vulneráveis do *plugin Elementor* [12]. **Consequência:** a exploração pode permitir o envio de arquivos maliciosos, possibilitando alteração indevida de conteúdo, comprometimento da aplicação ou uso do servidor para ações não autorizadas. **Mitigação:** atualizar o *plugin Elementor* para a versão corrigida, restringir permissões de *upload*, validar extensões e tipos de arquivos permitidos e monitorar diretórios de *upload*.

Vulnerabilidade 2 – CVE-2020-7109: Cross-Site Scripting (XSS) no Elementor. A CVE-2020-7109 corresponde a uma falha de *Cross-Site Scripting* em versões vulneráveis do *plugin Elementor* [13]. **Consequência:** essa falha pode permitir a execução de scripts maliciosos no navegador de usuários, possibilitando roubo de sessão, redirecionamento indevido ou manipulação do conteúdo exibido. **Mitigação:** atualizar o *plugin* para a versão corrigida, sanitizar entradas de usuários, escapar saídas HTML e aplicar políticas de segurança como *Content Security Policy* (CSP).

Vulnerabilidade 3 – CVE-2023-47504: Leitura arbitrária de anexos no Elementor. Esta vulnerabilidade envolve uma falha de autorização que pode permitir acesso indevido a anexos em versões vulneráveis do *Elementor* [14]. **Consequência:** a falha pode resultar na exposição de arquivos, documentos ou informações internas que não deveriam estar acessíveis a determinados usuários. **Mitigação:** atualizar o *plugin Elementor*, revisar permissões de acesso a arquivos e mídias, aplicar o princípio do menor privilégio e monitorar requisições suspeitas a anexos.

Vulnerabilidade 4 – CVE-2020-13642: CSRF e XSS no Page Builder by SiteOrigin. A CVE-2020-13642 está associada a falhas de *Cross-Site Request Forgery* (CSRF) e *Cross-Site Scrip-*

ting (XSS) no *plugin Page Builder by SiteOrigin* [15]. **Consequência:** a exploração pode permitir a execução de ações indevidas ou scripts maliciosos, afetando usuários autenticados e comprometendo a integridade das interações com a aplicação. **Mitigação:** atualizar o *plugin* para versão corrigida, utilizar *tokens* anti-CSRF, validar requisições e sanitizar entradas recebidas pela aplicação.

Vulnerabilidade 5 – CVE-2022-3357: PHP Object Injection no Smart Slider 3. A CVE-2022-3357 afeta versões vulneráveis do *plugin Smart Slider 3* e está relacionada à injeção de objetos PHP [16]. **Consequência:** dependendo do ambiente e dos componentes disponíveis, a falha pode permitir manipulação de objetos internos da aplicação, alteração de comportamento ou comprometimento parcial do sistema. **Mitigação:** atualizar o *plugin Smart Slider 3* para a versão corrigida, evitar desserialização de dados não confiáveis, validar entradas e manter os *plugins WordPress* continuamente atualizados.

5.3. Discussão dos Resultados

Os resultados demonstram que a presença de *plugins WordPress* desatualizados representa um fator relevante de risco. A maior parte das vulnerabilidades identificadas está associada a componentes amplamente utilizados, como *Elementor, SiteOrigin e Smart Slider 3*.

Além disso, a repetição de CVEs em mais de uma aplicação sugere a reutilização de uma mesma base tecnológica vulnerável. Esse cenário indica a necessidade de políticas de atualização contínua e controle de versões de componentes utilizados em aplicações *web* institucionais.

A análise também evidencia que vulnerabilidades de alta criticidade podem permanecer presentes em aplicações acessíveis publicamente quando não há um processo contínuo de atualização e monitoramento. Dessa forma, a gestão de componentes e a correção preventiva de falhas conhecidas tornam-se medidas essenciais para redução da superfície de ataque.

6. Conclusão

A realização deste trabalho possibilitou a análise prática de vulnerabilidades em aplicações *web* utilizadas por uma instituição de ensino. A partir do uso de ferramentas de reconhecimento e análise de vulnerabilidades, foi possível identificar ativos expostos, tecnologias utilizadas e vulnerabilidades conhecidas associadas a componentes *WordPress*.

Os resultados evidenciaram a presença de múltiplas CVEs, incluindo vulnerabilidades críticas com altos índices CVSS. A maioria das falhas identificadas está relacionada a *plugins* desatualizados, reforçando a importância de políticas contínuas de atualização, auditoria e gestão de vulnerabilidades.

Conclui-se que a utilização de ferramentas de reconhecimento e análise de vulnerabilidades, aliada à interpretação crítica dos resultados, é fundamental para a identificação preventiva de riscos e para o fortalecimento da segurança de aplicações *web* institucionais.

Referências

[1] MITRE Corporation. CVE - Common Vulnerabilities and Exposures. Disponível em: `https://www.cve.org/About/Overview`, 2026. Acesso em: 25 mai. 2026.

[2] Hatanael Lima Fernandes and Carlos Alberto da Silva. Investigação de vulnerabilidades em aplicações web utilizadas por empresas de leilões online e instituições de ensino a distância (ead). Trabalho de Conclusão de Curso, Universidade Federal de Mato Grosso do Sul. Disponível em: `https://repositorio.ufms.br/jspui/handle/123456789/12178`, 2025. Acesso em: 25 mai. 2026.

[3] Gabriel Augusto Ocampos Vitorino and Carlos Alberto da Silva. Análise de vulnerabilidades em domínios wordpress de instituições de ensino. Trabalho de Conclusão de Curso, Universidade Federal de Mato Grosso do Sul. Disponível em: https://repositorio.ufms.br/jspui/handle/123456789/13210, 2025. Acesso em: 25 mai. 2026.

[4] National Institute of Standards and Technology. CVSS Metrics - National Vulnerability Database. Disponível em: https://nvd.nist.gov/vuln-metrics/cvss, 2026. Acesso em: 25 mai. 2026.

[5] National Institute of Standards and Technology. Information Security - Glossary. Disponível em: https://csrc.nist.gov/glossary/term/information_security, 2026. Acesso em: 25 mai. 2026.

[6] WordPress.org. WordPress. Disponível em: https://wordpress.org/, 2026. Acesso em: 25 mai. 2026.

[7] Oracle. Oracle VM VirtualBox. Disponível em: https://www.virtualbox.org/, 2026. Acesso em: 25 mai. 2026.

[8] Offensive Security. Kali Linux Documentation. Disponível em: https://www.kali.org/docs/, 2026. Acesso em: 25 mai. 2026.

[9] ProjectDiscovery. Subfinder. Disponível em: https://github.com/projectdiscovery/subfinder, 2026. Acesso em: 25 mai. 2026.

[10] ProjectDiscovery. httpx. Disponível em: https://github.com/projectdiscovery/httpx, 2026. Acesso em: 25 mai. 2026.

[11] WPScan Team. WPScan: WordPress Security Scanner. Disponível em: https://wpscan.com/, 2026. Acesso em: 25 mai. 2026.

[12] MITRE Corporation. CVE-2020-7055. Disponível em: https://www.cve.org/CVERecord?id=CVE-2020-7055, 2020. Acesso em: 25 mai. 2026.

[13] MITRE Corporation. CVE-2020-7109. Disponível em: https://www.cve.org/CVERecord?id=CVE-2020-7109, 2020. Acesso em: 25 mai. 2026.

[14] MITRE Corporation. CVE-2023-47504. Disponível em: https://www.cve.org/CVERecord?id=CVE-2023-47504, 2023. Acesso em: 25 mai. 2026.

[15] MITRE Corporation. CVE-2020-13642. Disponível em: https://www.cve.org/CVERecord?id=CVE-2020-13642, 2020. Acesso em: 25 mai. 2026.

[16] MITRE Corporation. CVE-2022-3357. Disponível em: https://www.cve.org/CVERecord?id=CVE-2022-3357, 2022. Acesso em: 25 mai. 2026.