UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Giovanna Maria Alves Evangelista

Metodologia para Apoiar a Prestação de Serviços de Cibersegurança na FACOM

Uberlândia, Brasil

2025

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Giovanna Maria Alves Evangelista

Metodologia para Apoiar a Prestação de Serviços de Cibersegurança na FACOM

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informações.

Orientador: Rodrigo Sanches Miani

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Sistemas de Informações

Uberlândia, Brasil

2025

Giovanna Maria Alves Evangelista

Metodologia para Apoiar a Prestação de Serviços de Cibersegurança na FACOM

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informações.

Rodrigo Sanches Miani
Orientador

Professor Diego Nunes Molinos

Professor Ivan da Silva Sendin

Uberlândia, Brasil
2025

Aos que ousam sonhar nas alturas, pois não existem fronteiras para o que se pode alcançar quando o coração se guia pela fé.

Agradecimentos

Agradeço ao Senhor Deus, que sempre me deu forças para ficar de pé em meios as tempestades, me abençoando com a sua graça e misericórdia dia após dia.

Aos meus pais, meu irmão e a minha família, que me apoiaram e me deram todo o incentivo para continuar a perseguir meus sonhos e objetivos.

Aos meus amigos Gabriel e Guilherme, que estiveram comigo durante todo o percurso de estudos. Também, aos meus companheiros de trabalho e líderes, que fazem parte da minha jornada acadêmica e profissional. Vocês me ajudam a crescer dia após dia.

E por fim, ao professor Rodrigo, que se dispôs a me orientar, ensinar e apoiar durante a minha tragetória no mundo de Segurança da Informação.

Resumo

Este trabalho propõe o desenvolvimento e aplicação de uma metodologia de testes de intrusão (*pentest*) para prestação de serviços de cibersegurança por parte da Faculdade de Computação da Universidade Federal de Uberlândia, com foco em sistemas web. O objetivo principal foi criar uma abordagem eficiente, baseada nas melhores práticas dos *frameworks* PTES, OSSTMM e OWASP WSTG, e aplicá-la no contexto de sistemas web, respeitando as limitações legais e operacionais típicas do ambiente em questão. A metodologia foi estruturada em fases claras e objetivas, abordando desde o planejamento e coleta de informações até a exploração de vulnerabilidades e a documentação dos resultados. Assim, o método envolveu a integração das melhores práticas dos *frameworks* mencionados, adaptando-as ao contexto acadêmico, com ênfase na execução de testes de intrusão de forma ética e legalmente responsável.

Ademais, o processo incluiu a criação de um *checklist* detalhado e a automação de tarefas repetitivas por meio de scripts, o que aumentou a eficiência e consistência da metodologia. Além disso, templates de relatórios técnicos e executivos foram desenvolvidos para garantir uma documentação clara e acessível aos gestores e profissionais de segurança. Os resultados obtidos confirmaram a eficácia da metodologia, que demonstrou ser adaptável a diferentes contextos, permitindo a identificação de vulnerabilidades críticas e proporcionando uma base sólida para a melhoria contínua das defesas dos sistemas.

Palavras-chave: Metodologia de Pentest, Segurança Web, Testes de intrusão, Vulnerabilidades.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PTES	*Penetration Testing Execution Standard*
OSSTMM	*Open Source Security Testing Methodology Manual*
OWASP	Open Web Application Security Project
WSTG	Web Security Testing Guide
NIST	National Institute of Standards and Technology
SQLi	SQL Injection
XSS	*Cross-Site Scripting*
CSRF	Cross-Site Request Forgery
CVSS	Common Vulnerability Scoring System
CVE	Common Vulnerabilities and Exposures
IDS	Intrusion Detection System
WAF	Web Application Firewall
JWT	JSON Web Token
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
DNS	Domain Name System
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
API	Application Programming Interface
CVSS	Common Vulnerability Scoring System
SQL	Structured Query Language
XML	eXtensible Markup Language
CIA	Confidentiality, Integrity, and Availability
ROE	Regras de Engajamento

Sumário

1 Introdução

A crescente complexidade e interconectividade dos sistemas web tem tornado as instituições acadêmicas alvos cada vez mais frequentes de ataques cibernéticos (Verizon, 2024; European Union Agency for Cybersecurity (ENISA), 2024). A segurança da informação se tornou um elemento essencial para proteger os dados e garantir a continuidade das operações desses sistemas (NIST Computer Security Resource Center, 2025; INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2022). Frequentemente, essas plataformas armazenam informações sensíveis de alunos, professores e servidores administrativos, o que aumenta a criticidade da proteção (BRASIL, 2018). Nesse cenário, os testes de intrusão, também conhecidos como *penetration tests* (*pentests*), são fundamentais, permitindo avaliar a resistência das defesas, identificar vulnerabilidades e simular ataques reais para compreender o nível de exposição dos sistemas (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025; SHAH, 2020).

Este trabalho propõe a criação de uma metodologia de testes de intrusão (*pentest*) voltada para a prestação de serviços de cibersegurança pela Faculdade de Computação da Universidade Federal de Uberlândia (FACOM/UFU), com foco em sistemas web, aplicável tanto aos próprios sistemas institucionais quanto a órgãos públicos, organizações do terceiro setor e empresas parceiras. A pertinência desse escopo decorre do caráter sensível dos dados tratados em plataformas acadêmicas e administrativas (por exemplo, prontuários estudantis, informações pessoais e registros financeiros), além do aumento consistente de incidentes em serviços on-line no setor educacional e além dele, conforme apontam levantamentos recentes sobre tendência de ameaças e vetores de ataque (Verizon, 2024; European Union Agency for Cybersecurity (ENISA), 2024). A metodologia integra as melhores práticas dos *frameworks* reconhecidos *Penetration Testing Execution Standard* (PTES), *Open Source Security Testing Methodology Manual* (OSSTMM) e *OWASP Web Security Testing Guide* (WSTG), de modo a oferecer um procedimento estruturado, eficiente e alinhado a exigências legais e éticas típicas de ambientes institucionais, garantindo que os *pentests* sejam executados com segurança, reprodutibilidade e rigor técnico (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025).

A metodologia desenvolvida cobre todas as fases do processo de *pentest*, desde o planejamento e a coleta de informações até a exploração de vulnerabilidades e a documentação dos resultados. Para garantir uma avaliação realista da exposição dos sistemas, foi adotado o modelo *Black Box*, no qual o avaliador atua como um atacante externo sem conhecimento prévio do sistema, que se dá por uma abordagem particularmente relevante

para identificar falhas na superfície pública dos sistemas acadêmicos (DOUPÉ; COVA; VIGNA, 2010).

Assim, a proposta integra as contribuições centrais dos três referenciais estudados de modo a oferecer um escopo geral para plataformas web e, ao mesmo tempo, preservar flexibilidade para adaptação em cenários específicos. Do Penetration Testing Execution Standard derivam a cadência processual e os artefatos de governança que sustentam pré-engajamento, planejamento, regras de engajamento e documentação reprodutível, o que assegura previsibilidade e controle do exercício em ambientes institucionais heterogêneos (PENETRATION TESTING EXECUTION STANDARD (PTES),). Do Open Source Security Testing Methodology Manual advém a ampliação do olhar para além do componente estritamente técnico, permitindo incorporar, quando pertinentes, canais humano, físico e de telecomunicações às hipóteses de teste, algo valioso em organizações com integrações não convencionais entre processos e sistemas (ISECOM, 2010). Do OWASP Web Security Testing Guide provém a granularidade de casos de teste orientados a aplicações web, cobrindo autenticação, gestão de sessão, validação de entrada, exposição de dados e comportamento de respostas HTTP, o que viabiliza rastreabilidade requisito–evidência e comparabilidade entre execuções (OWASP FOUNDATION, 2025).

Embora o escopo primário seja amplo, a metodologia foi desenhada para adaptação controlada a cenários específicos, inclusive legados. Essa integração busca reduzir lacunas entre norma e operação, garantir rastreabilidade de achados e evidências e padronizar critérios de qualidade e saída por fase, preservando conformidade com boas práticas de documentação técnica e de relatório conforme o NIST SP 800-115 (SCARFONE et al., 2008). Assim, a metodologia consolida processo, métricas e casos de teste em um único arcabouço aplicável a plataformas web heterogêneas, mantendo a possibilidade de ajustes finos para cenários específicos sem perda de coerência metodológica (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025; SCARFONE et al., 2008).

Para materializar a metodologia, foram criados três instrumentos principais: um *checklist* detalhado, *scripts* automatizados e templates de relatórios. O *checklist* assegura a padronização das atividades, trazendo instruções, ferramentas e critérios para cada fase do *pentest*. Os scripts automatizados reunem o processo de comandos utilizados e aumentam a eficiência e consistência, agilizando tarefas repetitivas como varreduras de portas, brute force de diretórios, etc. Já os templates de relatórios permitem documentar os achados de forma clara, com versões técnicas e executivas. Dessa forma, gestores e profissionais de segurança conseguem compreender os riscos e recomendações de maneira objetiva.

1.1 Objetivo Geral

O objetivo principal deste trabalho é desenvolver uma metodologia estruturada e adaptada para a realização de testes de intrusão em ambientes acadêmicos. A proposta visa integrar as melhores práticas de *frameworks* de segurança ofensiva, como o PTES, OSSTMM e OWASP WSTG, para garantir uma abordagem eficiente e eficaz na identificação de vulnerabilidades em sistemas web utilizados pelas universidades (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025).

A metodologia criada será capaz de atender às limitações operacionais e legais típicas desses ambientes, respeitando a ética exigida para o contexto acadêmico (SHANLEY; JOHNSTONE, 2015). Ao fornecer uma abordagem clara e estruturada, este trabalho busca contribuir para a segurança de sistemas acadêmicos, possibilitando a realização de testes de intrusão com segurança e consistência, alinhados às melhores práticas de segurança ofensiva (SHAH, 2020).

1.2 Objetivos Específicos

O primeiro objetivo específico deste trabalho consiste em realizar um levantamento detalhado das metodologias de *pentest* mais amplamente utilizadas, como PTES, OSSTMM e OWASP WSTG, com o intuito de compreender suas abordagens e selecionar a mais apropriada para o contexto acadêmico (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025). As metodologias de *pentest* mais utilizadas foram comparadas a partir de quatro critérios objetivos: cobertura integral do ciclo (do planejamento ao relatório), clareza das orientações técnicas, aplicabilidade a aplicações web e capacidade de adaptação ao contexto institucional. Nesse recorte, o *Penetration Testing Execution Standard* destaca-se pela organização do processo e pelo rigor na definição de escopo e *Rules of Engagement* (PENETRATION TESTING EXECUTION STANDARD (PTES),); o *Open Source Security Testing Methodology Manual* acrescenta visão operacional e métricas que auxiliam a equilibrar risco e disponibilidade (ISECOM, 2010); e o *Web Security Testing Guide* oferece casos de teste objetivos e diretamente aplicáveis a aplicações web (OWASP FOUNDATION, 2025). A literatura recomenda, ademais, o alinhamento entre referência metodológica e prática operacional, reforçando essa seleção comparada (SHANLEY; JOHNSTONE, 2015).

Quanto ao modelo de avaliação, a adoção do *Black Box* mostra-se adequada por refletir a perspectiva de um atacante externo, favorecer a análise da superfície pública, reduzir a dependência de acessos privilegiados e facilitar o cumprimento de limites legais e éticos; além disso, integra-se naturalmente aos testes previstos no WSTG (DOUPÉ; COVA; VIGNA, 2010; OWASP FOUNDATION, 2025).

Ademais, destaca-se a criação de uma metodologia híbrida de *pentest*, que integre as práticas dos *frameworks* selecionados de maneira coesa e eficaz. A metodologia proposta será estruturada para abranger todas as fases do *pentest*, desde o planejamento e coleta de informações até a exploração de vulnerabilidades e a documentação dos resultados (SHANLEY; JOHNSTONE, 2015). Para otimizar a execução e garantir a reprodutibilidade dos testes, serão desenvolvidas ferramentas de apoio, como um *checklist* detalhado, *scripts* automatizados e templates de relatórios, que assegurarão a padronização, eficiência e consistência na aplicação da metodologia, atendendo tanto às necessidades dos profissionais de segurança quanto dos gestores envolvidos (SHAH, 2020; Verizon, 2024).

1.3 Organização do trabalho

Esta monografia está organizada da seguinte forma: o Capítulo 2 apresenta a fundamentação teórica sobre segurança da informação, *frameworks* de pentest e modelos de ataque; o Capítulo 3 descreve o método adotado, detalhando a construção da metodologia e o uso das ferramentas de apoio; o Capítulo 4 apresenta o desenvolvimento, incluindo aplicação prática, análise de vulnerabilidades, *scripts* automatizados e templates de relatório; e o Capítulo 5 traz as conclusões, resultados finais, discussões sobre limitações e sugestões para trabalhos futuros.

2 Fundamentação Teórica

Este capítulo visa apresentar os conceitos e as bases teóricas que sustentam a metodologia de teste de intrusão (*pentests*) desenvolvida neste trabalho. A fundamentação abrange aspectos essenciais da segurança da informação, metodologias amplamente reconhecidas no setor, como o *PTES*, *OSSTMM* e *OWASP WSTG*, além de princípios legais e éticos que orientam a prática desses *pentests*. A segurança da informação, com foco nos princípios de confidencialidade, integridade e disponibilidade, é abordada como o pilar fundamental para a elaboração de testes eficazes. Além disso, são explorados os conceitos de vulnerabilidades, ameaças e riscos, com ênfase nas abordagens técnicas e operacionais para sua identificação e mitigação. Por fim, o capítulo integra as metodologias de segurança ofensiva, as ferramentas utilizadas e as práticas legais, fornecendo uma base sólida para a compreensão e aplicação da metodologia criada.

2.1 Segurança da informação: princípios, ameaças e vulnerabilidades

A segurança da informação é comumente explicada pelos três pilares da confidencialidade, integridade e disponibilidade (CIA), que orientam políticas, processos e a seleção de controles técnicos e organizacionais (NIST Computer Security Resource Center, 2025; INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2022). De acordo com o NIST CSRC, trata-se de proteger informações e sistemas contra acesso, uso, divulgação, interrupção, modificação ou destruição não autorizados, com o objetivo explícito de assegurar esses três atributos fundamentais (NIST Computer Security Resource Center, 2025). Em convergência, a ISO/IEC 27002:2022 posiciona os controles como meios de tratamento de risco voltados a preservar o CIA dos ativos informacionais e detalha práticas e atributos que apoiam o desenho e a operação de um SGSI (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2022). Assim, a base conceitual e normativa se reforça mutuamente: o NIST delimita o que proteger e por quê, enquanto a ISO/IEC 27002 orienta como proteger no cotidiano da gestão.

Como o cenário de ameaças é dinâmico, a priorização de controles precisa acompanhar essa evolução. Segundo o *Threat Landscape* mais recente da European Union Agency for Cybersecurity (ENISA), campanhas de engenharia social permanecem persistentes e mais sofisticadas, com a ampliação de modelos *as a service*, como *phishing as a service* e *ransomware as a service*, além do uso de plataformas legítimas como vetores de disseminação (European Union Agency for Cybersecurity (ENISA), 2024). Corroborando

esse movimento, reportes nacionais registram o avanço do *QR-phishing*, também chamado de *quishing*, como tática de engajamento inicial, o que mostra que pequenos ajustes na engenharia social já ampliam a superfície de ataque, conforme o NCSC-FI/Traficom (NCSC-FI / Traficom, 2023). Diante desse quadro, a abordagem em camadas torna-se essencial, combinando controles técnicos, monitoramento contínuo e programas de conscientização, como enfatiza a própria ENISA (European Union Agency for Cybersecurity (ENISA), 2024).

Os dados do Data Breach Investigations Report 2024 indicam que credenciais comprometidas, *spear-phishing* e exploração de vulnerabilidades conhecidas seguem entre os vetores mais prevalentes (Verizon, 2024). À luz do DBIR, a gestão de risco ganha insumos objetivos para definir respostas táticas, por exemplo, priorizar correções de falhas já exploradas, elevar proteções de identidade e reforçar detecções para *phishing* direcionado (Verizon, 2024). Desse modo, a leitura prospectiva da ENISA se articula com a evidência de campo do DBIR, favorecendo alinhamento entre políticas, controles e ameaças que de fato se materializam (European Union Agency for Cybersecurity (ENISA), 2024; Verizon, 2024).

Quanto ao tratamento de fraquezas técnicas, o ecossistema CVE e a NVD definem vulnerabilidade como uma falha em software ou hardware cuja exploração pode degradar confidencialidade, integridade ou disponibilidade, conectando diretamente o defeito técnico aos princípios fundadores da segurança (The CVE Program, 2025; National Vulnerability Database (NVD), 2022). Para qualificar a severidade dessas falhas, o mercado utiliza o Common Vulnerability Scoring System (CVSS): Conforme o Guia do Usuário do CVSS v3.1, a pontuação descreve severidade e deve ser combinada com probabilidade de exploração e criticidade do ativo para estimar risco de maneira adequada (FIRST.Org, Inc., 2019). Em paralelo, o CVSS v4.0 introduz aperfeiçoamentos, como o grupo *Supplemental* e maior granularidade, e já conta com suporte oficial da NVD, o que indica adoção progressiva sem descontinuar o uso disseminado do v3.1 em fluxos de gestão (COMMON. . . , 2024; National Vulnerability Database (NVD), 2024). Na prática, convém usar v3.1 quando integrações e processos já estão estabilizados, ao mesmo tempo em que se avalia a migração gradual para v4.0 conforme o ecossistema e as ferramentas de varredura amadurecem.

Em síntese, as ameaças se manifestam principalmente por engenharia social, exposição de credenciais e exploração de falhas, como mostram ENISA e DBIR (European Union Agency for Cybersecurity (ENISA), 2024; Verizon, 2024). As vulnerabilidades descrevem fraquezas específicas que, quando exploradas, impactam diretamente o CIA, segundo CVE e NVD (The CVE Program, 2025; National Vulnerability Database (NVD), 2022). Diante disso, controles técnicos e organizacionais guiados pela ISO/IEC 27002, articulados ao conceito de proteção do NIST, reduzem probabilidade e impacto dentro

de um SGSI (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2022; NIST Computer Security Resource Center, 2025). Por fim, a combinação entre vigilância contínua do panorama de ameaças e o uso disciplinado de taxonomias e métricas padronizadas, como CVE, NVD e CVSS, fortalece a governança do risco cibernético e sustenta decisões de priorização mais consistentes (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2022; European Union Agency for Cybersecurity (ENISA), 2024).

2.2 Testes de intrusão

Os testes de intrusão (*pentest*), conforme o *NIST SP 800-115*, são exercícios controlados de avaliação técnica que têm como objetivo identificar e validar vulnerabilidades em sistemas, redes e aplicações (SCARFONE et al., 2008). Quando autorizado, o *pentest* também explora essas falhas para demonstrar possíveis caminhos de ataque e estimar seu impacto. O guia destaca a importância de um planejamento formal, uma autorização clara, Regras de Engajamento (ROE) e a documentação rigorosa das evidências, garantindo que os achados sejam reproduzíveis e acionáveis. Ele ainda descreve métodos, limites operacionais e a estrutura de relatórios, que devem incluir camadas técnica e executiva distintas (SCARFONE et al., 2008). As *ROE* constituem o acordo formal que antecede o teste e delimita, com precisão operacional e jurídica, o que pode e o que não pode ser feito durante o *pentest*. Em termos práticos , por sua vez, são estabelecidas antes do início do teste e determinam as atividades que podem ser realizadas, com restrições claras de escopo, horário e precauções para garantir a segurança do ambiente de teste (NIST Computer Security Resource Center, 2024).

É importante entender as diferenças entre *pentest*, varredura de vulnerabilidades e auditoria. As varreduras são, em sua maioria, automatizadas e baseadas em assinaturas e bancos de falhas conhecidas, oferecendo uma ampla cobertura e rapidez na detecção inicial de fraquezas. O *pentest*, por outro lado, é conduzido por profissionais, que utilizam tanto ferramentas automatizadas quanto técnicas manuais para encadear condições, explorar falhas e avaliar o impacto real de cenários de ataque. Guias e materiais recentes do mercado deixam claro essa distinção ao comparar os objetivos, escopo, métodos e entregáveis de cada abordagem (FireMon, 2025; Compyl, 2025; TechTarget SearchSecurity, 2025).

A auditoria, por sua vez, é uma atividade sistemática e independente que verifica se requisitos predefinidos estão sendo atendidos. De acordo com a *ISO 19011:2018*, a auditoria foca na conformidade e eficácia dos processos de sistemas de gestão, sem a intenção de simular um ataque. Embora as auditorias de um *SGSI* possam incluir a verificação de controles técnicos, seu objetivo principal é avaliar a aderência a critérios, políticas e procedimentos. Em resumo, a varredura e o *pentest* fazem parte da avaliação técnica, enquanto a auditoria tem como função verificar a conformidade e a eficácia do

sistema de gestão (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), 2018; ASQ, 2025).

No ciclo de um *pentest*, o *NIST* organiza o processo em várias etapas: (i) preparação e autorização, que envolvem o termo de autorização, as *ROE* e os critérios de parada; (ii) definição do escopo e dos objetivos, que incluem ativos, janelas de teste e riscos aceitáveis; (iii) técnicas de coleta, teste e análise, como a descoberta de falhas, validação, exploração controlada e manejo de dados; (iv) comunicação com as partes interessadas; e (v) elaboração do relato e das recomendações, separando os achados técnicos das implicações executivas, com cadeia de custódia e evidências rastreáveis (SCARFONE et al., 2008). Além disso, práticas regulatórias como o guia do *FedRAMP* reforçam que as *ROE* devem especificar métodos, vetores e limites de exploração, e que qualquer desvio do plano aprovado exige autorização explícita (PENETRATION..., 2022).

Essa estrutura permite que os *pentests* sejam conduzidos de forma eficiente e controlada, garantindo que as vulnerabilidades sejam identificadas, exploradas e documentadas adequadamente, sempre dentro dos parâmetros estabelecidos pelas partes envolvidas.

2.3 Modelos de teste de intrusão

Os modelos de "caixa" descrevem o nível de informação prévia concedida ao time de testes sobre o alvo e, por extensão, a perspectiva e a profundidade da avaliação (SCARFONE et al., 2008). Em caixa preta, o avaliador atua sem conhecimento interno, reproduzindo a visão de um atacante externo, útil para avaliar exposição pública, processos de descoberta e tempo até o compromisso inicial (SCARFONE et al., 2008). Em caixa branca, há acesso a documentação, credenciais, código e arquitetura, habilitando avaliação exaustiva de fluxos lógicos, controles internos e superfícies não expostas publicamente (SCARFONE et al., 2008). Em caixa cinza, fornecem-se credenciais limitadas ou documentação parcial para acelerar a cobertura sem perder aderência a cenários realistas, combinando técnicas de caixa preta e branca, como reconhecido explicitamente pelo *NIST SP 800-115* ("a combinação é conhecida como *grey box testing*") (SCARFONE et al., 2008). De modo convergente, a *OWASP WSTG* orienta que atividades de teste podem envolver caixa branca (p.ex., análise de código), caixa preta (p.ex., *pentest*) e caixa cinza (conhecimento parcial do sistema), adotando essa taxonomia para organizar técnicas e objetivos de avaliação (OWASP FOUNDATION, 2025).

A escolha do modelo deve considerar metas, restrições e maturidade do ambiente (SCARFONE et al., 2008). Em domínios regulados e com forte exigência de garantia, a abordagem de caixa branca reduz incerteza técnica e cobre caminhos de execução não triviais (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), 2018). Para validar postura externa e capacidades de detecção e resposta (p.ex., *alerting* e *triage*),

a abordagem de caixa preta é valiosa ao aferir resiliência de perímetro e processos operacionais sob um ponto de vista externo (CICHONSKI et al., 2012; SCARFONE et al., 2008). Em iniciativas com prazos restritos ou quando se busca otimizar custo-benefício, a abordagem de caixa cinza costuma equilibrar profundidade e tempo, pois o compartilhamento mínimo de informações reduz a fase de descoberta e concentra esforços em exploração e comprovação de impacto (SCARFONE et al., 2008; OWASP FOUNDATION, 2025). Essa decisão deve ocorrer ainda na fase de planejamento (escopo, objetivos, janelas de teste, critérios de parada e *rules of engagement*), conforme estruturado nas diretrizes do *NIST SP 800-115* e compatível com a forma como a *OWASP WSTG* organiza suas atividades por objetivo e nível de conhecimento (SCARFONE et al., 2008; OWASP FOUNDATION, 2025; NIST Computer Security Resource Center, 2024).

2.4 Frameworks e metodologias

A prática de testes de intrusão tem consolidado *frameworks* amplamente reconhecidos no setor de segurança. Dessa maneira, cada um desses *frameworks* oferece uma abordagem única e complementa a avaliação de segurança de sistemas de maneira distinta. O *Penetration Testing Execution Standard* (*PTES*) fornece uma estrutura sequencial e controlada para a execução dos testes, dividindo o processo em fases claramente definidas, como pré-engajamento, coleta de informações, modelagem de ameaças, exploração e pós-exploração (PENETRATION TESTING EXECUTION STANDARD (PTES),).

O *Open Source Security Testing Methodology Manual* (*OSSTMM*) amplia essa abordagem ao incluir não apenas a segurança cibernética, mas também a avaliação de riscos operacionais e de infraestruturas físicas e humanas. Sua abordagem permite uma visão abrangente da segurança, considerando múltiplos aspectos de uma organização (ISECOM, 2010).

Por outro lado, o *Web Security Testing Guide* (*OWASP WSTG*) foca exclusivamente na segurança de aplicações web, cobrindo desde a configuração até a validação de entradas e gestão de sessões. O *WSTG* é estruturado em categorias e casos de teste que possibilitam uma análise detalhada de vulnerabilidades comuns em sistemas web, o que é crucial para ambientes acadêmicos, onde sistemas de gestão de dados e plataformas de ensino estão frequentemente expostos a ataques (OWASP FOUNDATION, 2025).

A Tabela 1 apresenta as principais diferenças entre as metodologias *PTES*, *OSSTMM* e *OWASP WSTG*, destacando suas abordagens e as ferramentas recomendadas para a execução dos testes de segurança. Essas metodologias possuem características específicas que se adequam a diferentes contextos operacionais e tipos de sistemas, permitindo que os profissionais de segurança escolham a abordagem mais apropriada de acordo com as necessidades da organização.

Aspecto	**PTES**	**OSSTMM**	**OWASP**
Objetivo	Fornecer um padrão completo de *pentest*	Avaliar segurança de redes e infraestrutura	Testar e avaliar segurança de aplicações web
Abrangência	Abrange todas as fases de um *pentest*	Foca em redes, processos e pessoas	Foca em aplicações web e vulnerabilidades específicas
Foco Principal	Execução detalhada de testes	Avaliação de riscos e controle de segurança de infraestruturas	Identificação de falhas em sistemas web
Métricas de Risco	Menos focado em métricas específicas	Foco em métricas de risco e eficácia da defesa	Não foca tanto em métricas, mas na identificação de falhas específicas
Tipo de Organização	Recomendado para testes em ambientes gerais	Ideal para grandes organizações com infraestruturas complexas	Focado em aplicações web, ideal para desenvolvedores e testadores de segurança web
Exemplos de Ferramentas	Burp Suite, SQLmap, Nmap	Nikto, Wireshark, OSSEC	Burp Suite, OWASP ZAP, SQLmap

Tabela 1 – Principais Diferenças entre as Metodologias de *pentest* PTES, OSSTMM e OWASP. A tabela compila as características essenciais dessas abordagens, incluindo seus objetivos, abrangência, foco e ferramentas recomendadas. Fonte: Adaptado de (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025).

2.5 Ferramentas por fase do teste

O uso de ferramentas deve sempre estar alinhado aos objetivos de cada fase do teste e às limitações éticas e legais do engajamento. O *NIST SP 800-115* sugere que o planejamento, a autorização e as regras de engajamento sejam claramente definidos para controlar a intensidade e o impacto das atividades, além de garantir que a documentação das evidências permita a reprodutibilidade dos testes (SCARFONE et al., 2008).

Na fase de coleta de informações, tanto passiva quanto ativa, ferramentas como o *theHarvester* são úteis para automatizar a coleta de e-mails, subdomínios, *hosts* e metadados a partir de fontes públicas. Isso ajuda a criar um mapa inicial da superfície de ataque sem interagir diretamente com o alvo, mantendo o processo não intrusivo (LARAMEE; CONTRIBUTORS, 2025). Já para o *fingerprinting* de tecnologias e versões, o *WhatWeb* é uma ferramenta eficaz, identificando *frameworks*, servidores e *plugins*, o que ajuda a formar hipóteses sobre possíveis vulnerabilidades e reduz a incerteza para a próxima etapa (URBANADVENTURER; CONTRIBUTORS, 2025).

Na enumeração de serviços e ativos, o *Nmap* continua sendo a ferramenta de referência para descobrir *hosts*, portas e versões de serviços, além de oferecer suporte ao *NSE* (*Nmap Scripting Engine*), que expande suas capacidades de detecção e validação de cenários específicos (PROJECT, 2025a; PROJECT, 2025b). Para a descoberta de diretórios e arquivos ocultos em aplicações web, ferramentas como *Gobuster* e *DirBuster* são amplamente utilizadas, enquanto o *OWASP ZAP* também disponibiliza o recurso *Forced*

Browse, que é útil para identificar *endpoints* e *consoles* administrativos que podem ter sido deixados inadvertidamente expostos (REEVES; CONTRIBUTORS, 2025; OWASP Foundation, 2025b; TEAM, 2025a). Vale ressaltar que todas essas atividades devem respeitar os limites acordados, pois elas aumentam o volume de requisições e podem impactar a disponibilidade do sistema alvo (SCARFONE et al., 2008).

Na exploração e validação de impacto, suítes de *proxy* e testes, como o *Burp Suite* e o *OWASP ZAP*, permitem interceptação, manipulação de requisições e varreduras ativas e passivas. Esses recursos são essenciais para validar as hipóteses geradas nas fases de modelagem e enumeração (PORTSWIGGER, 2025; TEAM, 2025b). Para a detecção e exploração de injeções SQL, o *sqlmap* automatiza esse processo, oferecendo suporte a diversos *SGBDs* e até rotinas de exfiltração que agilizam a comprovação de impacto quando a vulnerabilidade é confirmada (G.; STAMPAR, 2025).

Na pós-exploração e análise de impacto, as ferramentas de *proxy* e automação continuam a ser valiosas, ajudando a mapear a extensão do acesso em componentes da web. Além disso, elas geram evidências reprodutíveis que são essenciais para documentar e comprovar o impacto. Logo, as documentações oficiais dessas ferramentas descrevem os modos de uso seguro e responsável, que devem ser seguidos para evitar quaisquer efeitos colaterais em ambientes de produção (PORTSWIGGER, 2025; TEAM, 2025b). A Tabela 2 compila as fases de *pentest* e suas respectivas ferramentas, abordando seus principais objetivos.

2.6 Aspectos éticos e legais

A execução ética e legal de testes de intrusão requer uma autorização formal do detentor do ativo, uma definição clara do escopo, regras de engajamento, limites técnicos e planos de contingência. O *NIST SP 800-115* destaca que o planejamento, a autorização e a comunicação com as partes interessadas são etapas essenciais para reduzir riscos operacionais e garantir a legitimidade do exercício. De acordo com o *NIST*, é necessário definir alvos, janelas de teste, métodos permitidos, critérios de parada, contatos de emergência e procedimentos de reversão, e as (*ROE*) complementam essa estrutura ao explicitar os limites e as salvaguardas antes do início das atividades (SCARFONE et al., 2008; NIST Computer Security Resource Center, 2024).

No Brasil, projetos que envolvem dados pessoais devem seguir as diretrizes da *Lei Geral de Proteção de Dados* (*LGPD*). A *Lei nº 13.709/2018* regula o tratamento desses dados por pessoas naturais ou jurídicas, incluindo processos digitais, e define os princípios, as bases legais e os direitos dos titulares. No contexto de testes de intrusão, a coleta e o processamento de informações, como *logs*, traços de sessão e tráfego de dados, podem envolver dados pessoais, o que exige uma base legal adequada, minimização de dados,

Fase	Ferramentas	Objetivo
Coleta de Informações	TheHarvester, WhatWeb	Coletar dados abertos de e-mails, subdomínios e metadados. Identificar tecnologias e versões.
Enumeração de Serviços	Nmap, Gobuster, DirBuster, OWASP ZAP	Descobrir hosts, portas e diretórios não listados, respeitando limites de disponibilidade.
Exploração	Burp Suite, OWASP ZAP, SQLmap, Nikto	Validar hipóteses de vulnerabilidades, como injeção SQL e arquivos expostos.
Pós-exploração	Burp Suite, OWASP ZAP	Mapear o impacto, movimento lateral e gerar evidências reprodutíveis.
Seleção de Ferramentas	OWASP WSTG	Organizar e verificar casos de teste para rastreabilidade e cobertura.

Tabela 2 – Levantamento de ferramentas e objetivos de acordo com cada uma das fases de *pentest*. Fonte: Adaptado de (LARAMEE; CONTRIBUTORS, 2025; URBANADVENTURER; CONTRIBUTORS, 2025; PROJECT, 2025a; REEVES; CONTRIBUTORS, 2025; OWASP Foundation, 2025b; TEAM, 2025b; PORTSWIGGER, 2025; G.; STAMPAR, 2025; CIRT, 2025; OWASP FOUNDATION, 2025).

medidas de segurança proporcionais e o registro das operações realizadas. Guias públicos, como o *Guia de Segurança da Informação para Agentes de Tratamento de Pequeno Porte* da *ANPD* e o *Guia de Boas Práticas – LGPD* do *Governo Digital*, ajudam a traduzir esses requisitos para a prática em *pentests* (BRASIL, 2018; AUTORIDADE NACIONAL DE PROTEçãO DE DADOS (ANPD), 2021; SECRETARIA DE GOVERNO DIGITAL — PORTAL GOV.BR, 2020).

Além disso, o *Marco Civil da Internet*, definido pela *Lei nº 12.965/2014*, estabelece princípios e garantias para o uso da internet no Brasil, incluindo balizas para o tratamento e a guarda de registros por provedores e o acesso a esses registros mediante ordem judicial. Esse marco orienta as obrigações e as expectativas para o tratamento de tráfego e registros de conexão ou de aplicação durante avaliações, impondo limites que devem ser respeitados por equipes de segurança e por terceiros autorizados (BRASIL, 2014).

A respeito da *cadeia de custódia*, a *Lei nº 13.964/2019*, também conhecida como *Pacote Anticrime*, introduziu os artigos 158-A a 158-F no *Código de Processo Penal*, definindo a *cadeia de custódia* como o conjunto de procedimentos que garantem a integridade

e a rastreabilidade de um vestígio. Embora o *pentest* não seja equivalente a uma perícia criminal, é uma boa prática tratar as evidências técnicas, como registros de exploração e provas do impacto, com o mesmo rigor, pois esses materiais podem embasar ações corretivas, auditorias ou medidas administrativas e jurídicas. Em consonância com isso, materiais públicos, como os manuais da *Polícia Científica do Estado do Espírito Santo* e orientações do *Ministério da Justiça e Segurança Pública*, fornecem diretrizes sobre documentação, armazenamento seguro e controle de acesso das evidências (BRASIL, 2019; MJSP, 2023; POLíCIA CIENTíFICA DO ESPíRITO SANTO (PCIES), 2024).

As diretrizes de *forense digital* recomendam que técnicas forenses sejam integradas às rotinas de resposta a incidentes para garantir que as evidências sejam preservadas e possam ser usadas de forma probatória. No contexto de *pentests* autorizados, princípios análogos devem ser seguidos para garantir que a qualidade das evidências de impacto seja elevada, evitando controvérsias sobre a integridade e a origem dos dados capturados. O *NIST SP 800-86* detalha a integração de *forense* nas respostas a incidentes, enquanto o *NIST SP 800-61r2* faz referência explícita a essas práticas (KENT et al., 2006; CICHONSKI et al., 2012).

Por fim, a ética profissional exige transparência, consentimento informado do contratante, respeito à privacidade de terceiros e a minimização do impacto sobre a operação. Os termos contratuais devem estabelecer claramente as responsabilidades, a confidencialidade, a propriedade intelectual sobre relatórios e os requisitos para o descarte seguro dos dados ao final do projeto. Esse conjunto de salvaguardas fortalece a confiança entre as partes e assegura a legitimidade da prática (SCARFONE et al., 2008; NIST Computer Security Resource Center, 2024).

2.7 Trabalhos relacionados

2.7.1 A diferença entre *frameworks* e prática operacional em ambientes institucionais

O estudo de (SHANLEY; JOHNSTONE, 2015), intitulado *"Bridging the gap between theory and practice in penetration testing frameworks"*, realiza uma análise detalhada das lacunas entre a teoria e a prática na aplicação de seis metodologias de testes de intrusão, destacando como as prescrições normativas nem sempre se alinham com as realidades enfrentadas por equipes que operam sob restrições de tempo, recursos e *compliance*. Os autores identificam que, embora *frameworks* como (PENETRATION TESTING EXECUTION STANDARD (PTES),), (ISECOM, 2010), (SCARFONE et al., 2008) e (OWASP FOUNDATION, 2025) forneçam diretrizes robustas, muitas vezes essas orientações precisam ser adaptadas para atender às necessidades de ambientes institucionais, onde as condições operacionais limitam a flexibilidade na implementação. O estudo propõe, en-

tão, a criação de critérios de qualidade para comparar diferentes metodologias e destaca a importância de personalizar esses *frameworks* para o contexto de cada organização.

Para este trabalho, esses achados são fundamentais para a construção de um método combinado, que integra as fases e artefatos do *PTES*, a disciplina processual do *NIST*, a abordagem multicanal do *OSSTMM* e os casos de teste do *WSTG*, ajustados conforme o escopo, os riscos e os recursos disponíveis dentro das instituições (SHANLEY; JOHNSTONE, 2015; PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; SCARFONE et al., 2008; OWASP FOUNDATION, 2025).

2.7.2 Revisões sistemáticas para definir escopo e profundidade em redes

A revisão sistemática *"Systematic review of penetration testing practices for network security"*, de (ALHAMED; RAHMAN, 2023) sintetiza as principais práticas e taxonomias dominantes no campo do *pentesting* de redes, destacando a caracterização dos modelos *black-box*, *white-box* e *grey-box*. Esses modelos referem-se ao nível de conhecimento que o testador tem sobre o sistema alvo antes de realizar os testes.

O trabalho elaborado também oferece *insights* sobre a eficácia das estratégias de *pentest* em diferentes camadas de rede, como o perímetro de segurança, a segmentação e os serviços críticos. Para ambientes acadêmicos e órgãos públicos, a revisão propõe uma abordagem estruturada para definir o escopo do *pentest*, orientando a escolha das camadas de rede a serem testadas e estabelecendo critérios claros para definir quando os testes devem ser interrompidos. Além disso, a revisão sugere que, para garantir uma cobertura eficaz e controlada, as técnicas de enumeração e os testes devem ser priorizados com base nos riscos identificados, ajudando a minimizar impactos enquanto maximiza a segurança da infraestrutura. Esses conceitos orientam a definição de escopo no contexto deste trabalho, especialmente para instituições acadêmicas e públicas, onde é essencial garantir a conformidade e a proteção dos ativos mais críticos da rede (ALHAMED; RAHMAN, 2023).

2.7.3 Combinação de ferramentas e matriz de seleção por fase e tecnologia

No trabalho de (SHAH, 2020), intitulado *"A comparative analysis of automated security testing tools for web applications using the OWASP Benchmark"*, é feita uma comparação detalhada de ferramentas automatizadas para testes de segurança de aplicações web, usando o *OWASP Benchmark*, mostra como a combinação de diferentes *scanners* pode melhorar a confiabilidade dos resultados. O estudo destaca que as ferramentas automatizadas variam na eficácia para detectar diferentes tipos de vulnerabilidades, e nenhuma delas é completamente eficaz para todos os tipos de falha.

O autor propõe o uso de matrizes de avaliação para a seleção das ferramentas de

pentest mais adequadas a cada fase do processo (coleta, enumeração, exploração e pós-exploração) e para diferentes tecnologias (como web, *APIs* e serviços de rede). Este estudo reforça a ideia de que, ao combinar *scanners* automatizados com validações manuais e utilitários de exploração, é possível melhorar a cobertura dos testes e aumentar a precisão dos achados. Para este trabalho, essa abordagem foi adaptada para criar um portfólio de ferramentas por fase e tecnologia, garantindo que cada etapa do *pentest* seja realizada de forma eficiente e que os resultados sejam documentados de maneira rastreável nos *templates* de relatório (SHAH, 2020; OWASP Foundation, 2025a).

2.7.4 Limitações de *scanners black-box* e papel da verificação manual guiada por casos de teste

O estudo de *Doupe*, *Comparative evaluation of black-box scanners in penetration testing"*, avalia onze *black-box scanners*, que são ferramentas automatizadas projetadas para realizar testes de intrusão sem acesso ao código-fonte ou à arquitetura interna do sistema (DOUPÉ; COVA; VIGNA, 2010). Os autores identificam várias limitações dessas ferramentas, como a dificuldade em realizar um "*crawling*"profundo, a incapacidade de lidar adequadamente com o estado de sessão em aplicações dinâmicas e a falta de detecção de falhas lógicas complexas.

O estudo sugere que o custo de uma ferramenta não está necessariamente relacionado à sua eficácia, pois ferramentas caras nem sempre têm um desempenho melhor do que alternativas mais acessíveis. Como resultado, *Doupe*, *Cova* e *Vigna* defendem que, embora as ferramentas automatizadas sejam úteis, elas devem ser complementadas por uma verificação manual para identificar falhas que as ferramentas podem deixar passar, especialmente falhas lógicas. A verificação manual deve ser orientada por hipóteses de teste, muitas vezes baseadas nos casos de teste do *OWASP WSTG*, que fornecem um conjunto claro de verificações para as falhas mais críticas em aplicações web. Esses achados reforçam a importância de combinar a automação com a validação manual, garantindo uma maior cobertura e precisão nos testes de segurança (DOUPÉ; COVA; VIGNA, 2010; OWASP FOUNDATION, 2025).

3 Metodologia

A metodologia desenvolvida neste trabalho tem como objetivo propor uma abordagem estruturada para a realização de testes de intrusão (*pentests*) em ambientes acadêmicos. A proposta busca integrar as melhores práticas dos principais *frameworks* existentes no mercado, como o *PTES*, o *OSSTMM* e o *OWASP WSTG*, adaptando-os para um contexto institucional, com foco na prestação de serviços de segurança ofensiva pela universidade, garantindo, ao mesmo tempo, legitimidade legal e conformidade ética. Como apontado pelo estudo de *Shanley* e *Johnstone* (2015), a implementação de metodologias de *pentest* deve ser adaptada para as realidades operacionais das organizações, já que as restrições de tempo e governança podem impactar diretamente a execução dos testes (SHANLEY; JOHNSTONE, 2015).

A Figura 1 ilustra o fluxograma que sintetiza o passo a passo da metodologia aplicada neste trabalho, detalhando as etapas que serão desenvolvidas ao longo do trabalho.

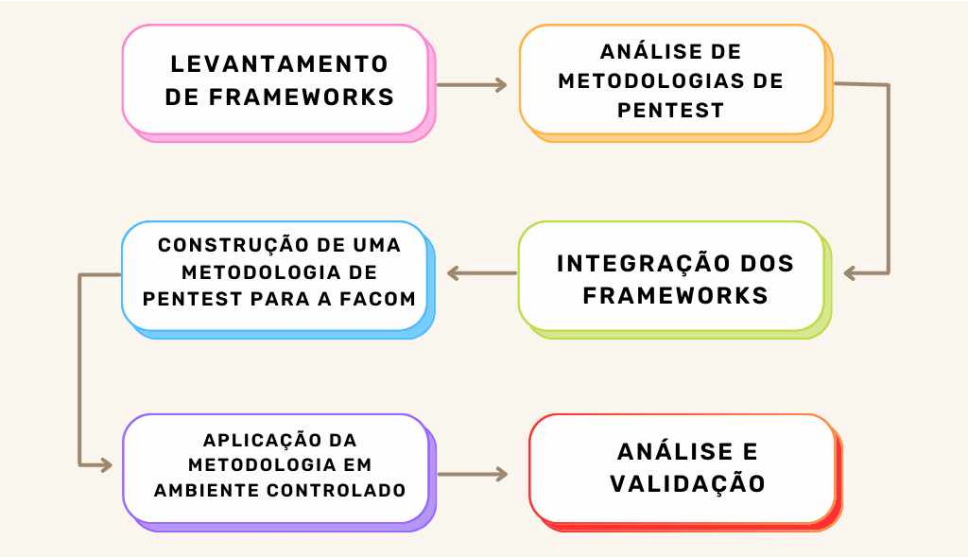

Figura 1 – Fluxograma da metodologia de *pentest* aplicada, destacando as fases de planejamento, levantamento de informações, exploração, pós-exploração e documentação dos resultados. Fonte: Da Autora

3.1 Levantamento de *frameworks* e metodologias de *pentest*

A primeira fase do desenvolvimento da metodologia foi dedicada ao levantamento das principais metodologias de *pentest* e dos *frameworks* mais utilizados. Nesse processo, foram estudadas as diferentes abordagens utilizadas em testes de intrusão, com destaque

para os modelos *Black Box*, *White Box* e *Grey Box*, que possuem características distintas e influenciam diretamente o escopo e a execução do *pentest*.

Também, os *frameworks PTES*, *OSSTMM* e *OWASP WSTG* foram criteriosamente estudados e analisados, considerando suas estruturas, abordagens e aplicações práticas em ambientes de segurança. O *Penetration Testing Execution Standard* (*PTES*) foi examinado por sua sequência didática de fases, que organiza o processo de pentest desde o pré-engajamento até a pós-exploração, garantindo clareza e rastreabilidade das atividades (PENETRATION TESTING EXECUTION STANDARD (PTES),). O *Open Source Security Testing Methodology Manual* (*OSSTMM*) foi avaliado por sua abordagem que abrange não apenas aspectos técnicos de redes e sistemas, mas também riscos operacionais, físicos e humanos (ISECOM, 2010). Já o *Web Security Testing Guide* (*OWASP WSTG*) foi analisado por sua especialização em segurança de aplicações web, fornecendo diretrizes detalhadas para identificação de vulnerabilidades comuns em sistemas de gestão e plataformas de ensino (OWASP FOUNDATION, 2025).

3.2 Integração dos *frameworks* para construção de uma metodologia de *pentest*

Com base nas análises efetuadas, a metodologia institucional foi construída por meio da integração dos três modelos de *pentest*. O objetivo foi combinar as melhores práticas de cada um deles para criar um modelo único que fosse aplicável ao contexto de universidades e instituições acadêmicas, permitindo que a metodologia fosse tanto eficaz quanto flexível para as necessidades de segurança da instituição (PENETRATION TESTING EXECUTION STANDARD (PTES), ; OWASP Foundation, 2025a; ISECOM, 2010).

A metodologia foi organizada em fases bem definidas de planejamento, coleta de informações, enumeração, exploração e documentação. Na fase de planejamento e escopo, são definidos os objetivos, o escopo do teste, a autorização e as *Rules of Engagement*. Nessa fase, o *PTES* se destaca por fornecer uma abordagem clara e didática, ajudando a estruturar todos os detalhes necessários para a execução do *pentest*, como já discutido por *Shanley* e *Johnstone* (2015), que ressaltam a importância do planejamento na execução de testes em ambientes com limitações (SHANLEY; JOHNSTONE, 2015). A coleta de informações, por sua vez, pode ser dividida em coleta passiva e ativa. A coleta passiva envolve a obtenção de dados públicos sem interagir diretamente com o sistema-alvo, utilizando ferramentas como *theHarvester* e *WhatWeb*, que buscam informações expostas em fontes como motores de busca, redes sociais e registros WHOIS. Esses dados incluem e-mails, subdomínios e tecnologias utilizadas pelo sistema, permitindo ao avaliador mapear a superfície de ataque sem gerar tráfego que possa ser detectado pela organização (LA-

RAMEE; CONTRIBUTORS, 2025; URBANADVENTURER; CONTRIBUTORS, 2025). Já a coleta ativa, por sua vez, envolve a identificação direta de serviços expostos e vulnerabilidades no sistema, sendo realizada por ferramentas como *Nmap* e *Gobuster*, que interagem com o sistema e tentam mapear portas, diretórios e arquivos sensíveis (ISECOM, 2010). O *OSSTMM* contribui para essa fase com uma abordagem mais ampla de segurança operacional, considerando não apenas a segurança dos sistemas, mas também a segurança física e a interação entre pessoas e sistemas, o que é essencial em ambientes institucionais (ISECOM, 2010).

Na fase de enumeração de serviços e ativos, a metodologia utiliza ferramentas como *Gobuster*, *DirBuster* e *OWASP ZAP* para identificar diretórios ocultos, serviços expostos e versões de software vulneráveis. O *PTES* se destaca nesta fase, pois descreve claramente os métodos de exploração e validação de vulnerabilidades. Durante a exploração, as vulnerabilidades identificadas são validadas por meio de ataques controlados, onde ferramentas como *Burp Suite*, *SQLMap* e *Nikto* são utilizadas para explorar falhas críticas, como *XSS* e *CSRF*. No entanto, a exploração deve ser feita com cautela, sempre respeitando os limites de risco definidos no planejamento, a fim de evitar danos ao sistema alvo (PORTSWIGGER, 2025; TEAM, 2025b; G.; STAMPAR, 2025; CIRT, 2025).

Por fim, a fase de relatório é dedicada à documentação dos resultados encontrados, dividida em duas partes: o relatório técnico, que descreve detalhadamente as falhas encontradas, os métodos utilizados e as evidências coletadas, e o relatório executivo, voltado para gestores e partes interessadas, fornecendo uma visão mais resumida dos riscos identificados e das recomendações de mitigação. A documentação final deve ser estruturada de forma a atender tanto aos profissionais de segurança quanto aos gestores, garantindo que as informações sejam claras e acessíveis (PORTSWIGGER, 2025; TEAM, 2025b).

3.3 Aplicação da Metodologia: execução em um domínio real

Para validar a metodologia proposta, foi selecionado um domínio com acesso autorizado e em ambiente controlado para a realização do *pentest*. O site utilizado foi o *testphp.vulnweb.com*, um ambiente específico projetado para testes de intrusão e desenvolvido pela empresa *Acunetix*, amplamente utilizado por profissionais de segurança e pesquisadores para fins educacionais e de treinamento (ACUNETIX, 2025). Este site foi escolhido devido à sua disponibilidade de informações públicas e à segurança do ambiente, permitindo que o *pentest* fosse realizado dentro dos parâmetros definidos, sem risco para sistemas reais ou dados sensíveis.

A utilização do *testphp.vulnweb.com* está em conformidade com as diretrizes de ética e segurança da informação, conforme recomendado por *Shanley* e *Johnstone* (2015) (SHANLEY; JOHNSTONE, 2015). A execução do *pentest* seguiu rigorosamente as fases

descritas na metodologia, com o uso de ferramentas específicas para cada etapa. Durante a aplicação, foi dada especial atenção à avaliação do impacto das vulnerabilidades encontradas e à comprovação dos achados, garantindo que todos os procedimentos estivessem em conformidade com as normas éticas e legais. O relatório final foi elaborado com base nas evidências obtidas e dividido em duas partes: uma técnica, voltada para profissionais de segurança, e uma executiva, destinada aos gestores e partes interessadas, assegurando que todas as informações fossem apresentadas de maneira clara e objetiva para os diferentes públicos envolvidos.

3.4 Análise e validação

A metodologia foi avaliada quanto à sua aderência aos *frameworks* e sua aplicabilidade prática em um sistema web, utilizando o laboratório Acunetix. A análise envolveu a comparação com outras abordagens de *pentest* e a identificação de pontos fortes e áreas que necessitam de melhorias contínuas. As conclusões principais destacaram a eficácia da metodologia no contexto acadêmico, a necessidade de ajustes finos em algumas ferramentas e a importância de manter um ciclo contínuo de feedback para aprimorar o método. A metodologia proposta demonstrou ser eficaz para realizar *pentests* em ambientes acadêmicos, permitindo a realização de testes de segurança com base nas melhores práticas do mercado, mas sempre respeitando as limitações e necessidades da instituição (SHANLEY; JOHNSTONE, 2015).

4 Desenvolvimento

A aplicação da metodologia de *pentest* para ambientes acadêmicos foi realizada com base nos *frameworks* análisados. O objetivo da aplicação foi garantir que o modelo desenvolvido fosse eficaz na identificação de vulnerabilidades em sistemas web, ao mesmo tempo em que respeitasse as limitações legais e operacionais características desse ambiente.

4.1 Levantamento das diferenças entre os modelos de *pentest*

No desenvolvimento da metodologia proposta para ambientes acadêmicos, a primeira etapa foi o levantamento das metodologias e *frameworks* de *pentest* mais relevantes para o contexto institucional. Para isso, foram analisados os diferentes tipos de *pentests*, com um destaque especial para os modelos *Black Box*, *White Box* e *Grey Box*. Cada uma dessas abordagens oferece vantagens e limitações que influenciam diretamente o escopo e a execução do *pentest*, especialmente no ambiente acadêmico, onde as condições operacionais e legais podem ser mais restritas.

O modelo *Black Box* foi escolhido como a abordagem principal para a metodologia, o qual de acordo com a análise de Doupe et al. (2010), permite testar a exposição pública do sistema e a capacidade de detectar e responder a ameaças externas, sem a necessidade de acesso direto ao código ou à arquitetura interna da aplicação (DOUPÉ; COVA; VIGNA, 2010). Essa característica se alinha à necessidade de testar o nível de segurança das aplicações contra ameaças externas, sem permitir que o avaliador tenha qualquer conhecimento prévio, como seria o caso nos modelos *White Box* e *Grey Box*.

Além da análise dos modelos de *pentest*, foi essencial a revisão dos *frameworks* utilizados no mercado de segurança ofensiva. A integração desses foi fundamental para a construção da metodologia, pois permitiu uma abordagem combinada que leva em consideração os melhores aspectos de cada um: o planejamento sequencial do *PTES*, a visão abrangente de segurança do *OSSTMM*, e a especialização no teste de aplicações web do *OWASP WSTG*. Dessa forma, a combinação assegura que o *pentest* seja abrangente, cobrindo desde a exploração de falhas externas, até a validação de controles internos, de maneira alinhada ao ambiente acadêmico e às restrições de segurança típicas de uma universidade (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025).

4.2 Criação de uma metodologia de pentest para prestação de serviços pela FACOM

Nessa etapa, o objetivo foi criar um modelo que seja aplicável a uma grande variedade de sistemas web. O *PTES* foi fundamental na definição da fase de planejamento e escopo, pois fornece um guia claro sobre como organizar e estruturar todos os detalhes do *pentest*, desde o consentimento formal até a definição dos parâmetros do teste (PENETRATION TESTING EXECUTION STANDARD (PTES),).

A metodologia foi organizada em fases bem definidas, com ênfase no planejamento, coleta de informações, exploração e documentação. Na fase de planejamento e escopo, são definidos os objetivos do *pentest*, o escopo do teste, a autorização formal e as regras de engajamento. O *PTES* foi seguido de perto, garantindo que todas as partes envolvidas fossem informadas sobre os objetivos e as restrições do teste (PENETRATION TESTING EXECUTION STANDARD (PTES),). A fase de coleta de informações foi dividida em passiva e ativa. Na coleta passiva, utilizamos ferramentas como *theHarvester* e *WhatWeb* para identificar *subdomínios*, *tecnologias* e outras informações expostas publicamente. A coleta ativa, com ferramentas como *Nmap* e *Gobuster*, permitiu mapear os serviços e identificar vulnerabilidades (LARAMEE; CONTRIBUTORS, 2025; URBANADVENTURER; CONTRIBUTORS, 2025; PROJECT, 2025a; REEVES; CONTRIBUTORS, 2025).

Na fase de enumeração de serviços e ativos, o objetivo foi identificar diretórios ocultos e arquivos sensíveis, utilizando ferramentas como *Gobuster*, *DirBuster* e *OWASP ZAP* (REEVES; CONTRIBUTORS, 2025; OWASP Foundation, 2025b; TEAM, 2025a). O *PTES* orientou essa fase ao fornecer uma estrutura clara para a exploração e validação de vulnerabilidades, com uma abordagem meticulosa para garantir que todas as falhas encontradas fossem devidamente documentadas (PENETRATION TESTING EXECUTION STANDARD (PTES),). Durante a exploração, as vulnerabilidades identificadas foram validadas por meio de ataques controlados, com o uso de ferramentas como *Burp Suite*, *SQLMap* e *Nikto*, que permitem explorar falhas críticas como *SQL Injection* (*SQLi*), *Cross-Site Scripting* (*XSS*) e *Cross-Site Request Forgery* (*CSRF*) (PORTSWIGGER, 2025; G.; STAMPAR, 2025; CIRT, 2025).

A fase de pós-exploração, focada na avaliação do impacto das falhas exploradas, foi fundamental para entender as consequências de um possível ataque *real*. As ferramentas *Burp Suite* e *OWASP ZAP* continuaram sendo úteis para validar o acesso e documentar as evidências de exploração, como a exfiltração de dados sensíveis e a movimentação lateral (PORTSWIGGER, 2025; TEAM, 2025b). A fase final envolveu a documentação dos resultados, com a elaboração de dois tipos de relatório: um técnico, que detalha as falhas e as evidências, e um executivo, com foco nas recomendações de mitigação para os gestores (PORTSWIGGER, 2025; TEAM, 2025b).

4.3 Criação de *checklist*

A criação do *checklist* representa a etapa em que a metodologia híbrida de *pentest* se transforma em um instrumento operacional concreto. O objetivo principal é traduzir as melhores práticas do *PTES*, do *OSSTMM* e do *OWASP WSTG* em itens verificáveis, com entradas, atividades, evidências e critérios de saída claramente definidos para cada fase do teste (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010; OWASP FOUNDATION, 2025). Em ambientes acadêmicos, diferentes tipos de *pentest* exigem graus variados de profundidade e rigor, o que justifica que o *checklist* seja estruturado como um documento vivo, versionado e rastreável, permitindo adaptações sem comprometer a coerência metodológica e a aderência aos *frameworks* (SHANLEY; JOHNSTONE, 2015).

Assim, a construção do *checklist* seguiu três princípios fundamentais. Primeiro, rastreabilidade entre *frameworks*: cada item do *checklist* está associado explicitamente a um requisito ou prática de origem no *PTES*, *OSSTMM* ou *WSTG*, facilitando auditoria, avaliação de qualidade e evolução informada da metodologia. Segundo, foco em evidências reprodutíveis: cada verificação indica o tipo mínimo de artefato necessário para comprovar a execução correta de uma atividade e o impacto associado. Por fim, critérios de saída objetivos por fase: cada fase possui *gates* de qualidade que permitem avançar somente quando todas as condições necessárias forem satisfeitas, assegurando consistência e confiabilidade dos resultados (PENETRATION TESTING EXECUTION STANDARD (PTES), ; OWASP FOUNDATION, 2020).

O *checklist* foi organizado em formato PDF, dividido em blocos alinhados às fases definidas na metodologia. No bloco de *Coleta de Informações*, a coleta passiva envolve itens relacionados a *subdomínios*, *tecnologias*, vazamentos e reputação do domínio. Ferramentas como *TheHarvester* e *WhatWeb* são referenciadas para realizar essa coleta de forma segura e documentada. As evidências mínimas incluem listas de artefatos coletados, *prints* ou exportações de resultados e anotações de data e hora, garantindo rastreabilidade (LARAMEE; CONTRIBUTORS, 2025; URBANADVENTURER; CONTRIBUTORS, 2025). A coleta ativa, por sua vez, contempla varreduras de portas e serviços com *Nmap* e mapeamento inicial de diretórios com *Gobuster*, com parâmetros registrados e limites de taxa documentados para respeitar as *Rules of Engagement* definidas na fase de planejamento (PROJECT, 2025a; REEVES; CONTRIBUTORS, 2025).

Na seção de *Enumeração*, os itens concentram-se na descoberta de *endpoints*, diretórios e arquivos sensíveis, utilizando ferramentas como *Gobuster* e *DirBuster*, além da análise de cabeçalhos de segurança e políticas de resposta por meio de ferramentas intermediárias. Cada verificação exige evidências, como listas de *endpoints* encontrados, amostras de respostas *HTTP* e observações sobre autenticação e controle de acesso. Assim, o critério de saída para essa fase define o conjunto mínimo de superfícies de ataque

necessárias para avançar à fase de exploração, evitando transições precoces (REEVES; CONTRIBUTORS, 2025; OWASP Foundation, 2025b; TEAM, 2025a).

O bloco de *Exploração* concentra-se na validação controlada de vulnerabilidades típicas, como *SQL Injection* (*SQLi*), *Cross-Site Scripting* (*XSS*), *Cross-Site Request Forgery* (*CSRF*) e falhas de autenticação e sessão. Cada item do *checklist* especifica pré-condições, como ambiente isolado e dados sintéticos, técnicas de exploração com ferramentas como *Burp Suite*, *SQLMap* e *Nikto*, além dos requisitos de evidência, incluindo prova do impacto, passos reproduzíveis e indicadores de risco. O critério de saída exige a classificação de severidade e a correlação com *CWEs* e *CVSS*, garantindo que as conclusões sejam mensuráveis e confiáveis (PORTSWIGGER, 2025; G.; STAMPAR, 2025; CIRT, 2025).

Na *Pós-Exploração*, o foco é no impacto e contenção, respeitando os limites do ambiente acadêmico. Os itens do *checklist* validam se a exploração resultou em acesso a dados sensíveis, elevação de privilégios ou movimento lateral. As evidências obrigatórias comprovam o alcance e a reversibilidade das ações realizadas, enquanto o critério de saída consolida as lições aprendidas e confirma que o ambiente retornou ao estado original (PORTSWIGGER, 2025; TEAM, 2025b).

O bloco de *Relatórios* define os itens necessários para a produção do relatório técnico e executivo. O relatório técnico detalha cada vulnerabilidade, os passos de reprodução, as evidências

Para facilitar a adoção e reduzir a variabilidade na execução, o *checklist* foi complementado com anexos operacionais, incluindo arquivos de automação para tarefas repetitivas, como perfis de Nmap e dicionários validados para brute force. Estes recursos incluem instruções de rate limiting e registro de parâmetros, garantindo que todas as atividades sejam auditáveis e realizadas dentro das regras de engajamento (PENETRATION TESTING EXECUTION STANDARD (PTES), ; OWASP FOUNDATION, 2020).

4.4 Execução do *checklist* em ambiente simulado

A aplicação do *checklist* em ambiente simulado e controlado utilizou o domínio de laboratório da Acunetix como alvo didático, seguindo o procedimento que lista as diretrizes metodológicas estabelecida na metodologia híbrida criada. Assim, o objetivo foi mapear a *superfície de ataque* de forma progressiva, partindo de coleta passiva até consultas ativas.

4.4.1 Coleta de evidências

Na coleta passiva, a identificação de *ativos* e *metadados* públicos foi realizada com o *TheHarvester*, o que permitiu confirmar *subdomínios* associados ao alvo e consolidar a visão inicial de escopo. A Figura 2 traz o comando executado juntamente com os seus resultados.

Figura 2 – Resultado inicial após a execução do *TheHarvester*. Fonte: Da Autora

Em seguida, realizou-se o processo de identificar e rastrear dispositivos ou usuários com base em suas características únicas e coletáveis (*fingerprinting* tecnológico), que foi conduzido com o *WhatWeb* para inferir *servidor web*, versão de linguagem e pistas adicionais presentes em *headers* e no conteúdo *HTML*. Como complemento, foram empregados *Google Dorks* focados em listagens de diretórios e artefatos possivelmente sensíveis. Conforme demonstrado pelas Figuras 3 e 4, a consulta que utiliza o comando *filetype* não retornou nenhum resultado, enquanto o comando *"intitle"* retornou um índice acessível com referência a um *script SQL*, *Index of /admin* com um arquivo: *create.sql*, acompanhado de outras informações do *PHP* e suporte a *FTP* no *servidor*, que serão fundamentais para a execução da etapa de enumeração.

Para fechar a análise passiva da superfície, executou-se uma verificação de portas com a ferramenta Nmap no intervalo completo, confirmando exposição de HTTP em 80/tcp, conforme apresentado na Figura 5 e na Figura 6. Todas as atividades e achados dessa etapa correspondem aos itens do *checklist* relativos a mapeamento de subdomínios, identificação de tecnologias, busca por dados expostos e detecção inicial de serviços. Dessa forma, obtem-se uma visão sobre possíveis acessos que poderão ser explorados na próxima

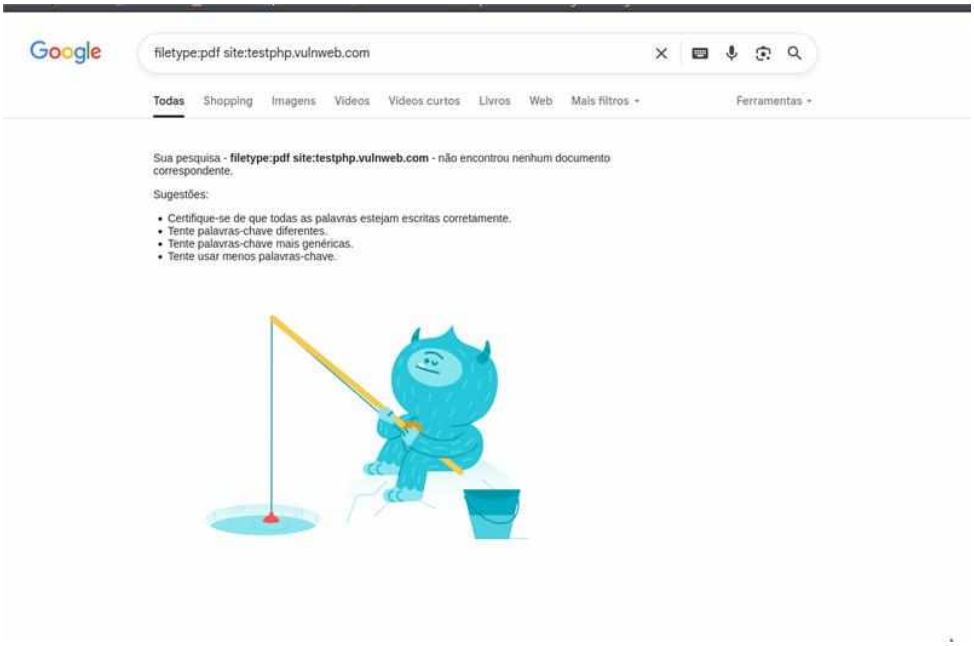

Figura 3 – Execução do comando Google Dorks file type. Fonte: Da Autora

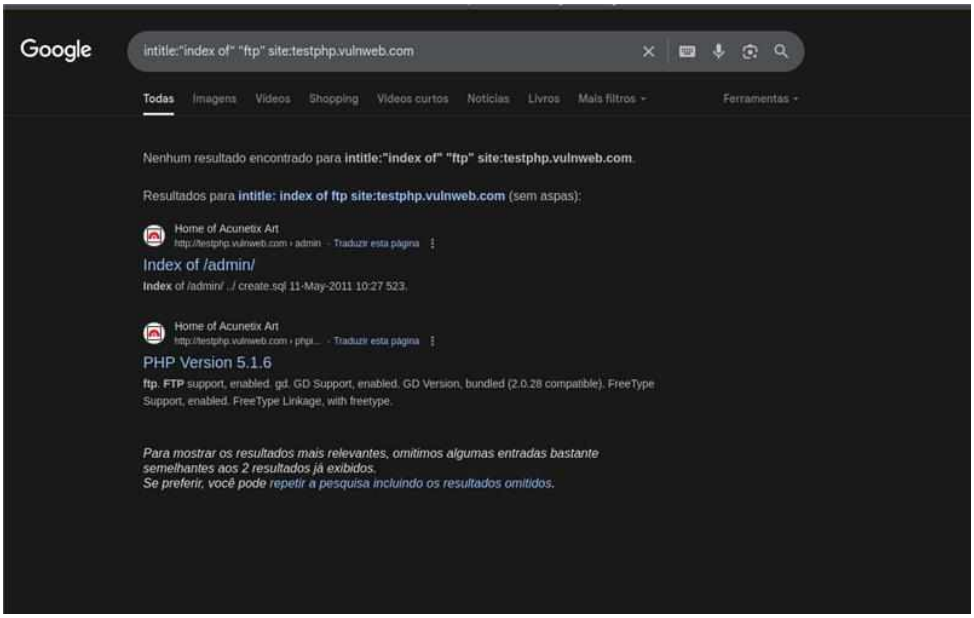

Figura 4 – Execução do comando Google Dorks intitle. Fonte: Da Autora

etapa do teste de intrusão. (LARAMEE; CONTRIBUTORS, 2025; URBANADVENTURER; CONTRIBUTORS, 2025; OWASP FOUNDATION, 2025; PROJECT, 2025a).

Na coleta ativa, avançou-se para consultas controladas que exigem interação direta com a infraestrutura. Assim, a Figura 7 mostra que tentativa de transferência de zona DNS via AXFR foi recusada pelos servidores autoritativos, resultado coerente com uma configuração mais restritiva.

Na sequência, a consulta *Whois* trouxe dados úteis para reforçar a rastreabi-

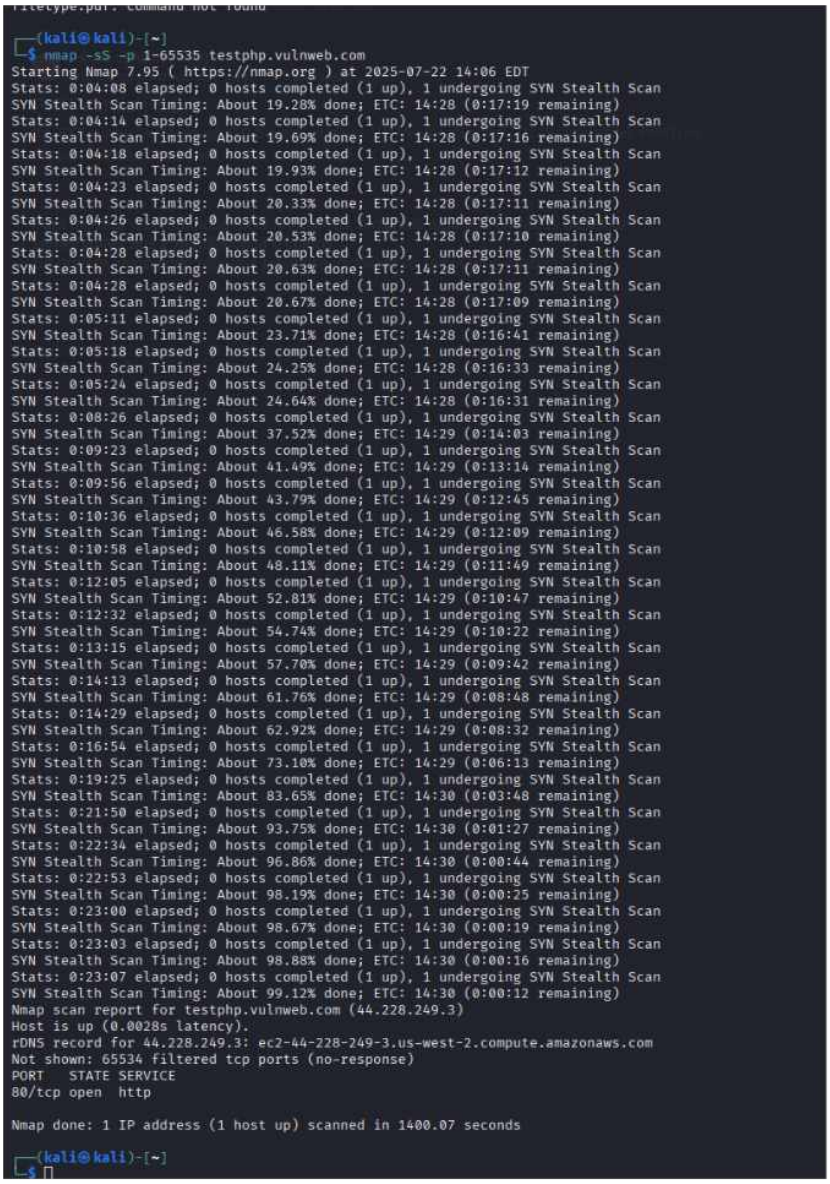

Figura 5 – Identificação de Portas com Nmap. Fonte: Da Autora

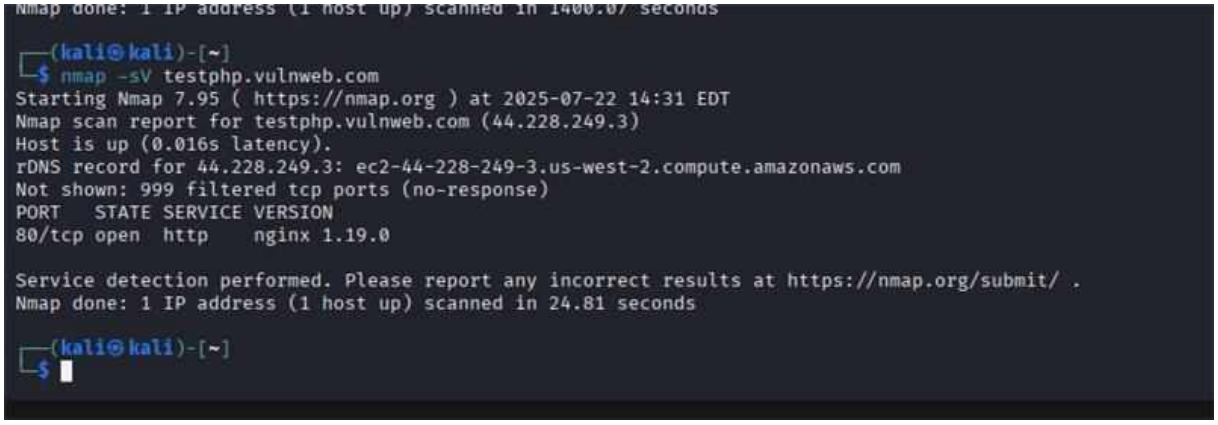

Figura 6 – Resultado de informações e portas expostas com Nmap. Fonte: Da Autora

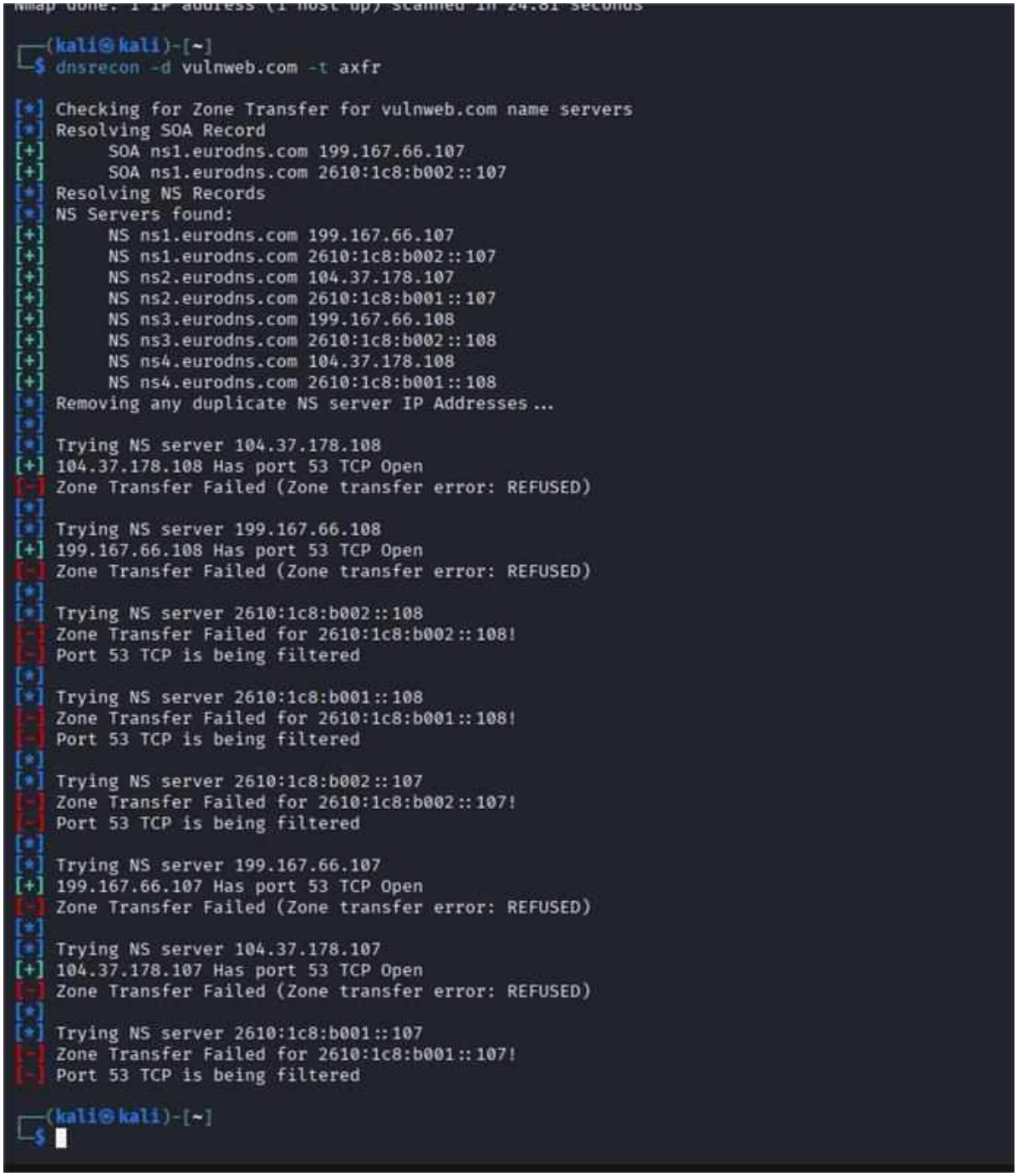

Figura 7 – Execução de DNS Zone Transfer (dnsrecon). Fonte: Da Autora

lidade do alvo de titularidade e infraestrutura de DNS, conforme as Figuras 8 e 9, sendo eles Antveski Gjorgji (registrante), Acunetix Limited (empresa) e DNS: NS1, NS2, NS3.eurodns.com. Por fim, uma varredura com Nmap (Figura 10) com detecção de versão e execução de *scripts* padrão confirmou o serviço HTTP, o banner do servidor e o título da página. Os parâmetros utilizados foram registrados para garantir reprodutibilidade e alinhamento às regras de engajamento definidas na etapa de planejamento (SCARFONE et al., 2008; PROJECT, 2025a; PROJECT, 2025b).

Figura 8 – Análise Whois - Execução do comando para realização da varredura. Fonte: Da Autora

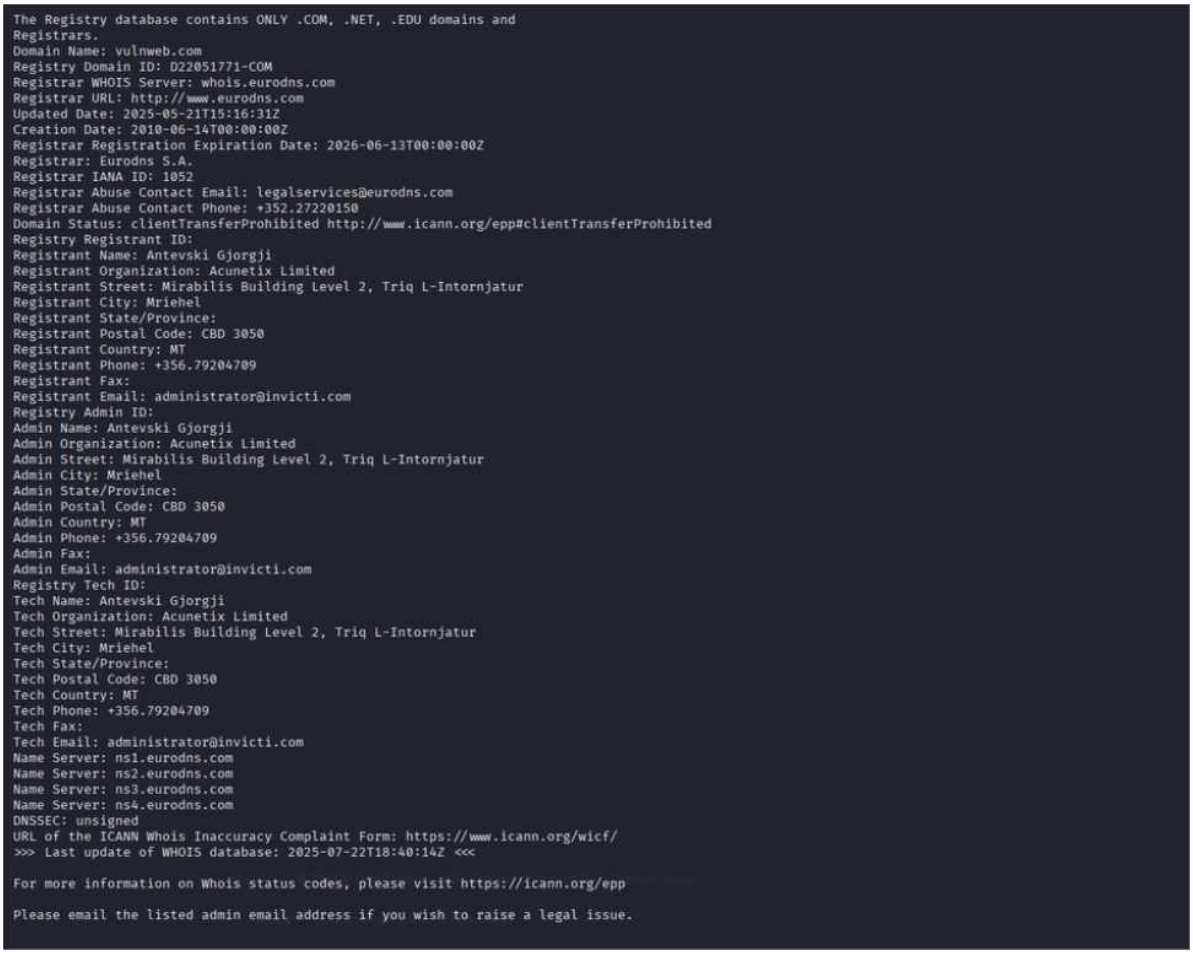

Figura 9 – Análise Whois - Resultado com informações do alvo. Fonte: Da Autora

Os critérios de saída da fase de Coleta de Informações foram atendidos, trazendo uma gama de informações úteis para exploração mais profunda nas próximas etapas do teste de intrusão. Dessa forma, as evidências mínimas de cada item foram produzidas, conforme ilustrado na Tabela 3 incluindo entre elas: a lista de subdomínios coletada, fingerprint de tecnologias, retorno dos dorks com indicação de diretório indexado e refe-

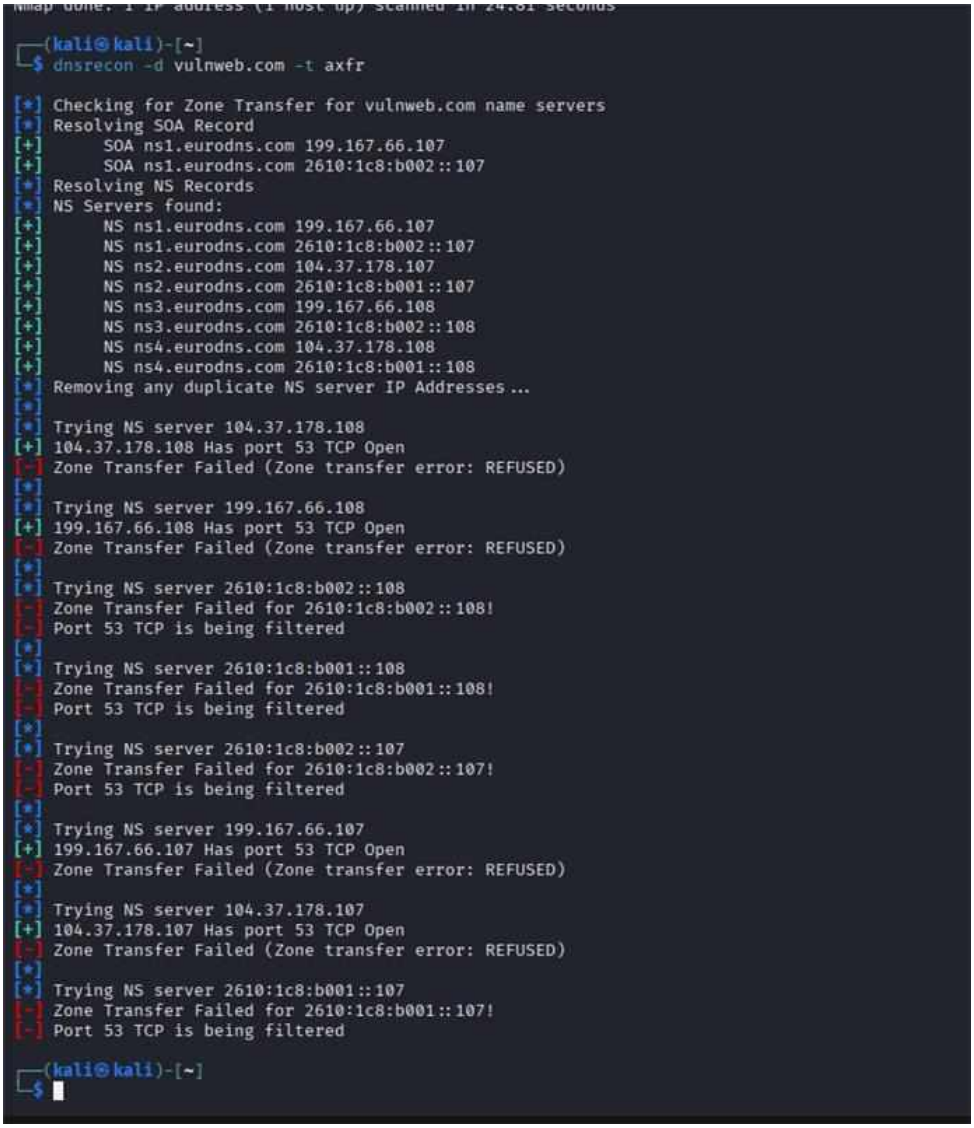

Figura 10 – Varredura com Nmap - detecção de versão e script. Fonte: Da Autora

rência a artefato SQL, negativa documentada de transferência de zona e confirmação de serviços e versões. Esse conjunto estabelece a linha de base necessária para a fase de Enumeração, que será direcionada para descoberta estruturada de endpoints e diretórios, com prioridade para verificação de exposições relacionadas ao artefato SQL identificado. (REEVES; CONTRIBUTORS, 2025; OWASP Foundation, 2025b; TEAM, 2025a; OWASP FOUNDATION, 2025).

4.4.2 Enumeração

Dando seguimento a execução do *checklist*, a fase de Enumeração teve como objetivo aprofundar o mapeamento dos recursos acessíveis no servidor, identificando diretórios e arquivos expostos que poderiam ser explorados nas fases subsequentes de uma maneira mais aprofundada. Para realizar essa tarefa, utilizou-se a ferramenta Gobuster (REEVES;

Categoria	**Ferramenta**	**Resultado Obtido**
Subdomínios/URLs	TheHarvester	5 subdomínios
Tecnologias	WhatWeb	PHP 5.6, nginx, JS, ActiveX
Diretórios Expostos	Google Dork	/admin/create.sql
Portas Abertas	Nmap	80/tcp
Versões de Serviços	Nmap -sV	nginx/1.19.0
Zone Transfer	DNSRecon	Recusado
Registro do Domínio	Whois	Acunetix Limited

Tabela 3 – Resumo da fase de coleta de evidências e resultados obtidos. Fonte: Da Autora.

CONTRIBUTORS, 2025), empregando a *wordlist* common.txt para realizar uma busca por diretórios e arquivos comumente encontrados em ambientes vulneráveis, abrangendo extensões como .bak, .zip, .sql e .old que geralmente podem expor uma vulnerabilidade moderada ou grave. A Figura 11 e a Figura 12 evidênciam esse processo.

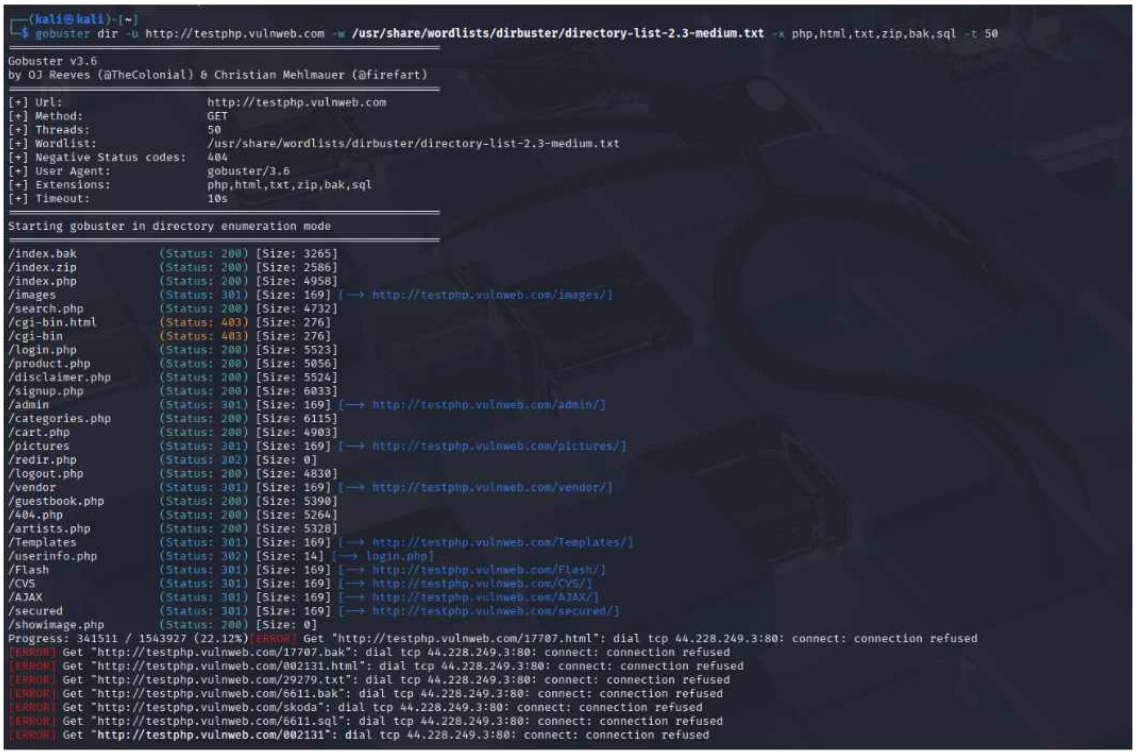

Figura 11 – Utilização do Gobuster para realizar brute force de diretórios e arquivos ocultos no alvo. Fonte: Da Autora

A análise das respostas *HTTP* revelou comportamentos significativos. O *Status 200 (OK)* confirmou que vários diretórios e arquivos estavam acessíveis, enquanto os *Status 301/302 (Redirect)* indicaram *redirecionamentos* legítimos, como observado no diretório `/admin/`, que redireciona o usuário para um destino específico. Já o *Status 403 (Forbidden)* mostrou que, embora o acesso a determinados diretórios tenha sido restrito, sua existência foi confirmada, o que pode ser relevante para futuras tentativas de reconhecimento e

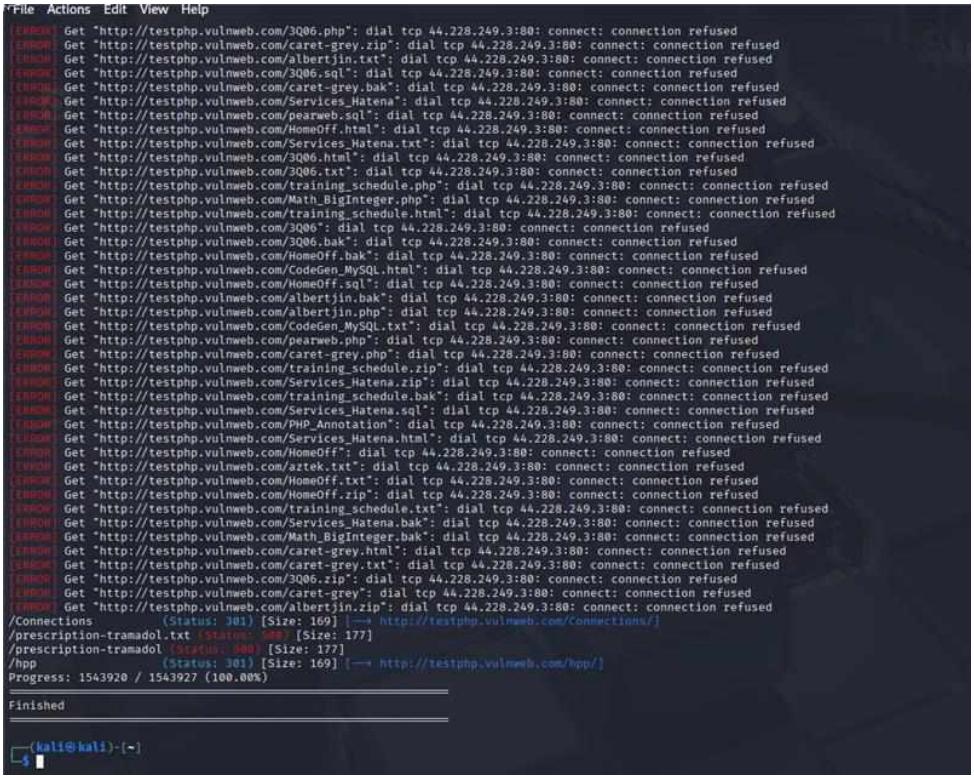

Figura 12 – Resultado do bruteforce. Fonte: Da Autora

ataque. O *Status 404 (Not Found)* foi registrado nas tentativas de acessar diretórios como `/cart/`, que não estavam presentes no *servidor*, evidenciando a necessidade de ajustes finos nas listas de diretórios e arquivos a serem testados (Compyl, 2025; TechTarget SearchSecurity, 2025). Durante a execução, erros de *conexão* foram observados em várias tentativas de acesso (Figura 12), como no *subdomínio* `/Math_BigInteger.php` e também no intitulado `/services_Hatena.bak`, que resultaram em mensagens de "*connection refused*". Esses erros indicam que os *services* podem não estar em funcionamento no momento da execução ou podem estar protegidos por medidas de *security*, como *firewalls* ou sistemas de *mitigation*, o que pode exigir novas abordagens ou a configuração de parâmetros diferentes para exploração (National Vulnerability Database (NVD), 2024).

Também, utilizando outro comando do *Gobuster*, foi possível extrair uma lista de diretórios e arquivos de interesse, expostos de maneira que seja possível acessá-los facilmente de acordo com a Figura 13. O diretório `/admin/`, por exemplo, redireciona para si mesmo, retornando o *Status 301 (Redirect)*. Dessa maneira, pode-se observar que o *status* apresentado indica que o diretório existe e está configurado para um *URL canônico*, uma descoberta importante que pode vir a ser útil em tentativas futuras de exploração, caso haja falhas na autenticação ou configuração do *server*. Ademais, pode-se citar também o diretório `/cgi-bin/`, que embora tenha retornado o *Status 403 (Forbidden)*, confirmou sua existência e restrição de acesso, o que pode resultar em uma forte chance de sucesso em tentativas de *privilege escalation* ou outras formas de exploração que envolvem per-

missões inadequadas dentro do *system* que está sendo testado (European Union Agency for Cybersecurity (ENISA), 2024).

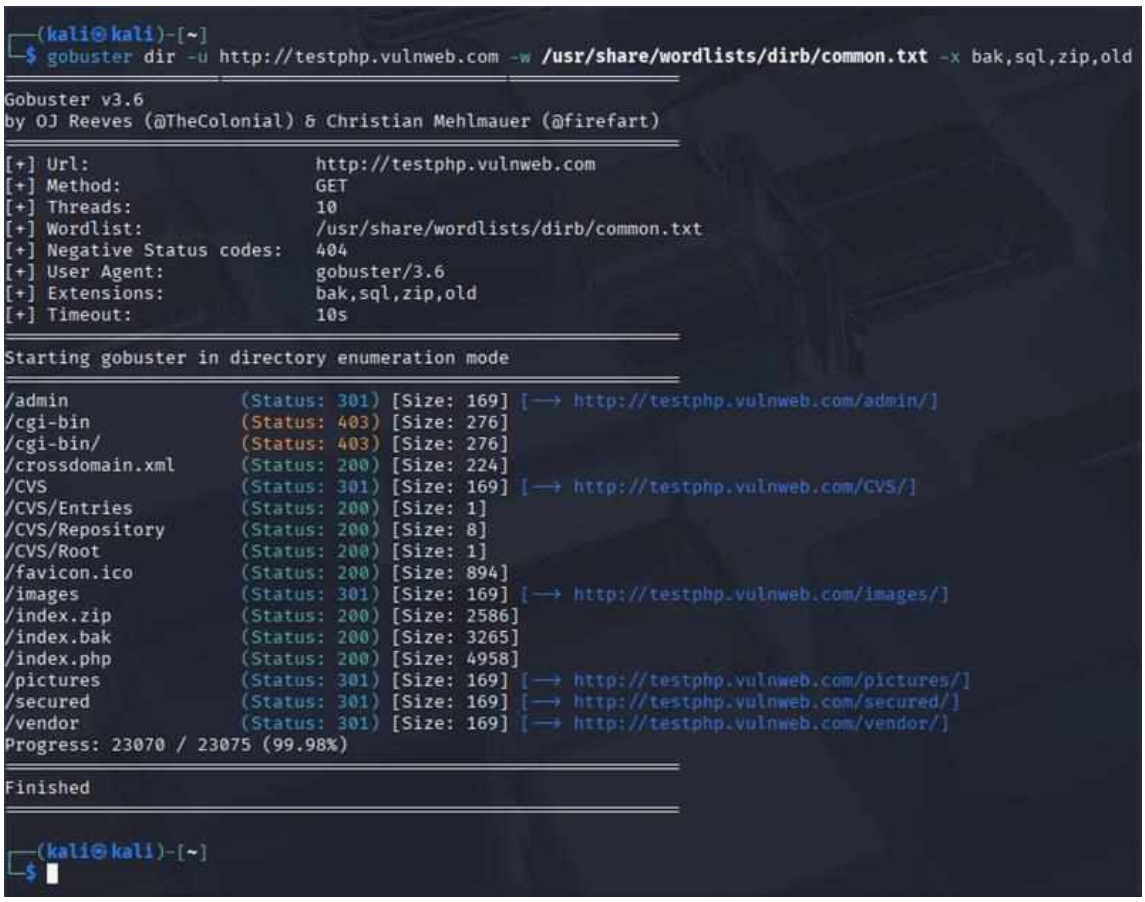

Figura 13 – Diretórios e arquivos expostos extraídos pelo Gobuster. Fonte: Da Autora

Além disso, a exposição de arquivos como `/index.bak` e `/index.zip` (Figura 13) revelou a presença de *backups* não removidos adequadamente. Esse é um erro comum em ambientes de desenvolvimento, onde arquivos sensíveis, como *source code* ou até *hardcoded credentials*, podem estar presentes. Desta maneira, a presença de tais arquivos indica uma falha de *security* grave, que pode ser explorada por um *attacker* para obter acesso não autorizado a informações cruciais. O arquivo `/prescription-tramadol.txt` foi identificado como possivelmente sensível, podendo ser uma *honeypot*, uma tática que visa enganar *attackers*, levando-os a interagir com arquivos monitorados. Além disso, o diretório `/hpp/` foi encontrado acessível, o que pode indicar a presença de *internal scripts* ou funções que exigem maior atenção, já que eles podem estar mal configurados ou expostos indevidamente (FireMon, 2025).

Por fim, a análise dos *HTTP headers*, realizada conforme a Figura 14, revelou que o *server* utiliza *nginx/1.19.0*, enquanto o *PHP* está na versão 5.6.40, uma versão antiga e desatualizada. A utilização de versões desatualizadas do *PHP* é um ponto crítico, pois essa versão é vulnerável a diversas falhas conhecidas, como *remote code execution (RCE)* e

falhas de *local/remote file inclusion (LFI/RFI)*, além de possibilitar ataques como *session hijacking* e *authentication bypass*. Isso exige uma atenção especial à *server security*, com a implementação de correções e atualizações constantes para mitigar esses riscos, como discutido no estudo de Cichonski et al. (2012), que reforça a importância de manter os sistemas sempre atualizados para evitar falhas de *security* (CICHONSKI et al., 2012).

Figura 14 – Resposta de análise HTTP utilizando cURL. Fonte: Da Autora

Assim, chega-se ao fim da etapa de enumeração com os pontos identificados e levantados, apresentados na Tabela 4.

Elemento Identificado	**Severidade**	**Observação**
Arquivos .bak, .zip, .sql	Alta	Backups expostos e arquivos sensíveis recuperáveis
Diretório /prescription-tramadol.txt	Alta	Possivelmente dado sensível ou armadilha
Diversos 403 Forbidden	Média	Diretórios protegidos mas identificados
Diversos 301/302	Baixa	Informações úteis para mapear estrutura
Várias connection refused	Informativo	Provável firewall/WAF ou erro na wordlist

Tabela 4 – Resultados obtidos na etapa de enumeração. Fonte: Da Autora.

4.4.3 Exploração

A fase de Exploração teve como principal objetivo validar, na prática, o impacto das vulnerabilidades previamente mapeadas durante a fase de Enumeração. Durante esta etapa, foram realizadas tentativas controladas de exploração das falhas encontradas, com o foco em demonstrar o risco real e as consequências das vulnerabilidades para a *security* da aplicação. O processo foi cuidadosamente realizado para não causar danos permanentes, respeitando os limites definidos no escopo do teste, como enfatizado pelo PTES e OSSTMM, que definem claramente as *Rules of Engagement* para evitar danos em sis-

temas durante a *exploration* (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010).

4.4.3.1 Exposição de arquivos sensíveis

A primeira falha crítica identificada foi a exposição de *sensitive files*. Durante os testes, foi possível acessar diretamente por meio da barra de pesquisa do *browser* arquivos como `/admin/create.sql`, `/index.bak` e `/index.zip` (Figura 15) sem qualquer proteção. O arquivo `create.sql`, obtido conforme a Figura 16, continha comandos *SQL* estruturais que revelavam a estrutura do *database*, enquanto os arquivos de *backup* poderiam conter *source code*, *hardcoded credentials* ou outras informações sensíveis.

Também foi encontrado o arquivo `/prescription-tramadol.txt`, que pode conter dados *confidential*. Para mitigar esse risco, é recomendável remover arquivos desnecessários do ambiente de produção e restringir o acesso a diretórios públicos por meio de diretivas de negação, como *deny all* no *NGINX* ou regras de `.htaccess`. Com isso em mente, a exposição de arquivos sensíveis é um erro comum que pode trazer riscos graves de segurança, como discutido no estudo da Verizon (2024), que revela como práticas inadequadas de gerenciamento de *backups* podem ser exploradas por *attackers* (Verizon, 2024).

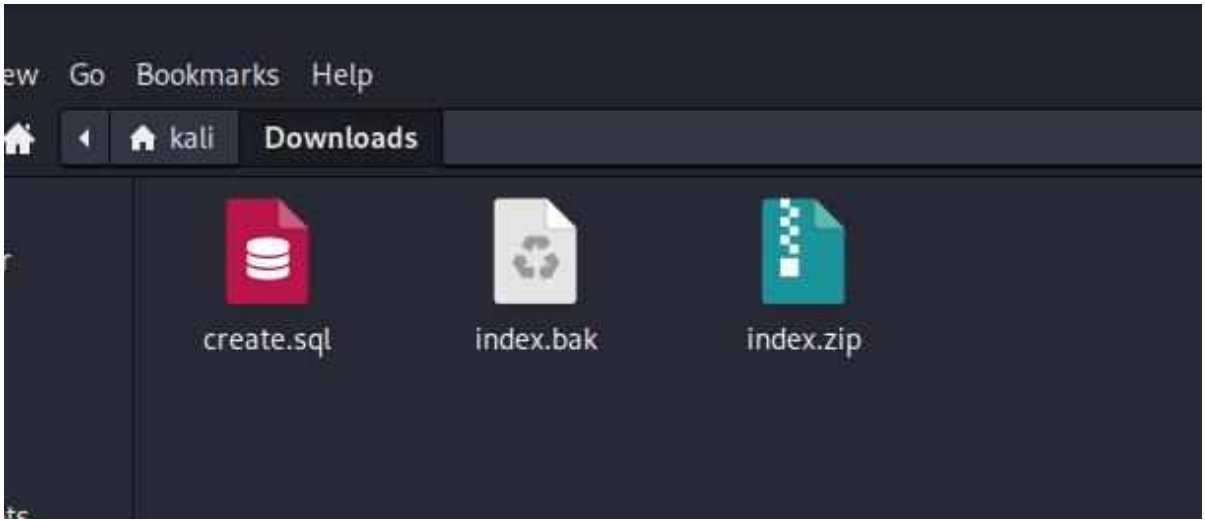

Figura 15 – Arquivos sensíveis obtidos pelo acesso aos respectivos diretórios no navegador. Fonte: Da Autora.

4.4.3.2 SQL Injection

A segunda vulnerabilidade explorada durante essa fase foi a *SQL Injection*. Como ilustrado na Figura 17, o ponto de partida para a execução desse procedimento foi a realização da injeção do comando ' OR 1=1– no campo de login da aplicação (/login.php),

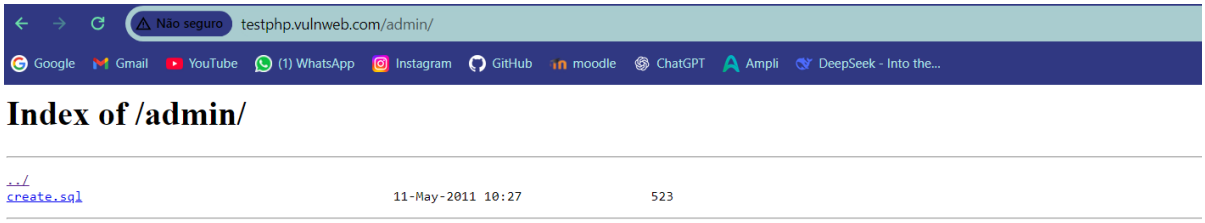

Figura 16 – Acesso ao diretório /admin pelo navegador. Fonte: Da Autora.

o que permitiu o sucesso do processo de autenticação e o acesso às páginas internas, mostrado na Figura 18.

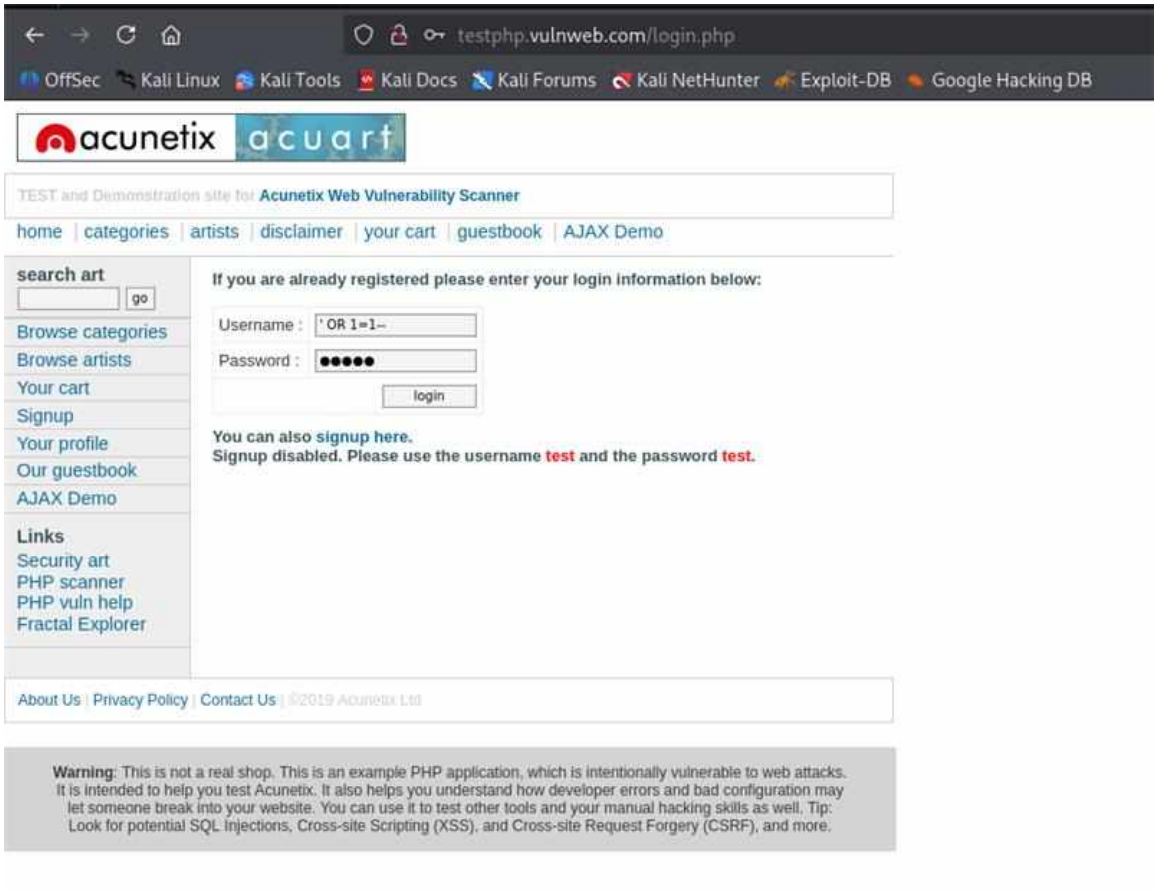

Figura 17 – SQL Injection - injeção do comando ' OR 1=1– para tentativa de login. Fonte: Da Autora.

Com a *injection* confirmada, o processo de *exploration* seguiu com a análise de *input parameters* da *application*, utilizando a ferramenta *SQLMap*. O objetivo inicial foi testar a vulnerabilidade de *SQL Injection* nos *parameters* fornecidos pela *application*. A ferramenta começou a examinar os *input fields*, especificamente o *parameter* "User-Agent", e gerou uma série de mensagens indicando que a *SQL injection* não parecia ser

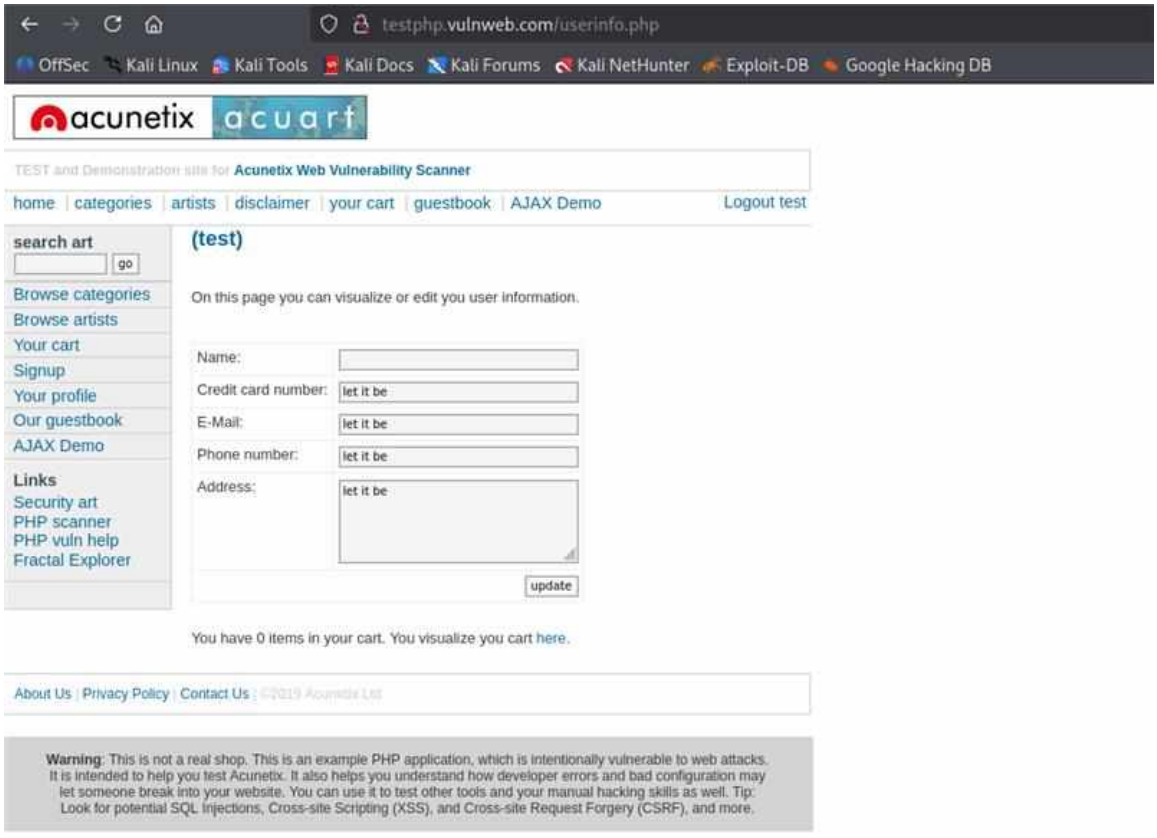

Figura 18 – SQL Injection - login por comando confirmado. Fonte: Da Autora

possível nesse campo. Em seguida, a ferramenta se concentrou no *parameter* "comment", realizando mais testes para verificar a possibilidade de *injection*. Também, a ferramenta fez uma sugestão de ajustes adicionais e informou que talvez fosse necessário diminuir o número de *requests* devido à detecção de outros mecanismos de *protection* que pudessem estar barrando a execução e o sucesso na *injection*. Dessa maneira, conforme demonstra a Figura 19 contendo os resultados da sequência de *commands*, a recomendação feita foi para usar a opção --technique=E"e o *command* --time-sec", assim visando ajustar a *attack strategy*, uma vez que o teste inicial não conseguiu identificar falhas críticas pontuais.

Os testes de *SQL Injection* não foram totalmente bem-sucedidos, o que pode indicar que a *application* pode estar implementando *defenses* contra ataques desse tipo, como validação de *parameters* e limitações de *request rate*. Isso também sugere que, embora as *direct SQL injections* não tenham sido eficazes, outras técnicas ou ajustes podem ser necessários para uma *deeper exploitation*. A presença de medidas de *mitigation*, como *firewalls* ou *WAFs*, também foi um fator a ser considerado nos testes, dificultando a execução dos *attacks*, como detalhado em diversos estudos sobre *web application security* (National Vulnerability Database (NVD), 2024; TechTarget SearchSecurity, 2025).

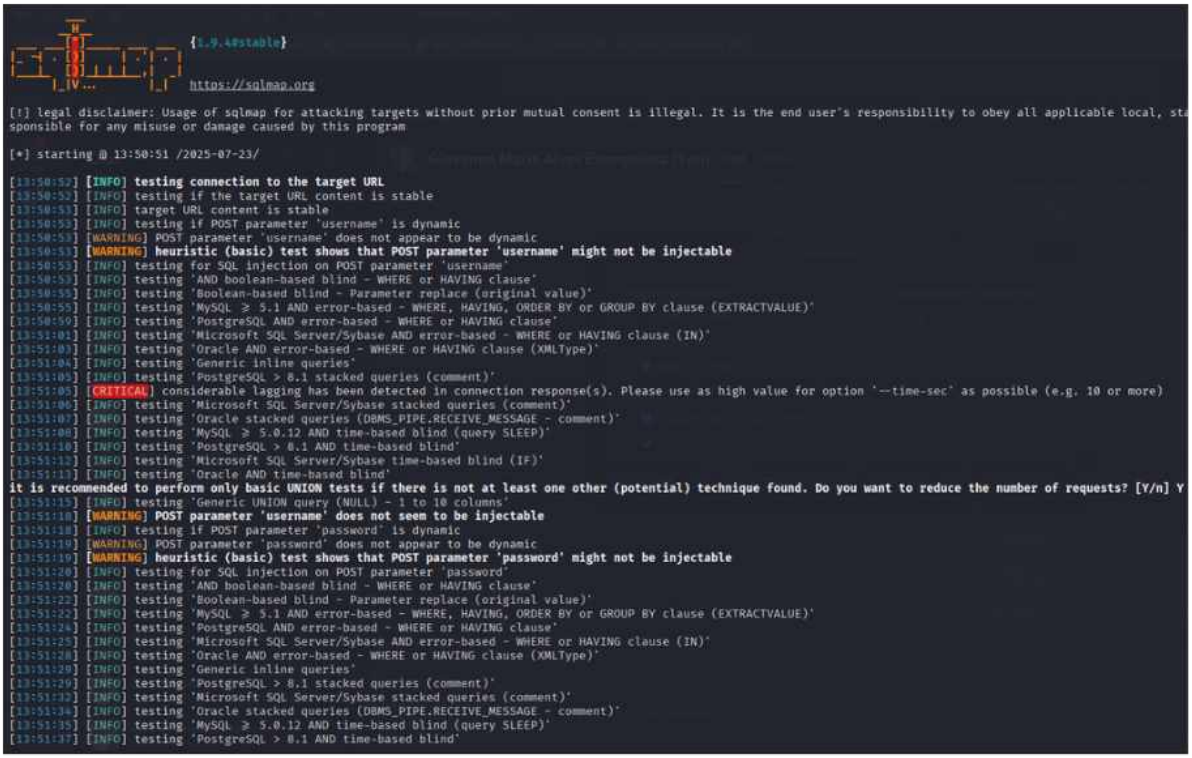

Figura 19 – SQL Injection - Execução com SQL Map. Fonte: Da Autora.

4.4.3.3 Autenticação baseada em Cookie Estático

Outra falha crítica explorada durante o processo foi a autenticação baseada em *static cookie*. Após o *login* realizado, também utilizando o *command* ' OR 1=1– no *login field* da *application* (/login.php), foi identificado um *cookie* chamado "login"com o valor bm9yYTpwYXNz, mostrado na Figura 20, que representa nora:pass quando descriptografado em *Base64*.

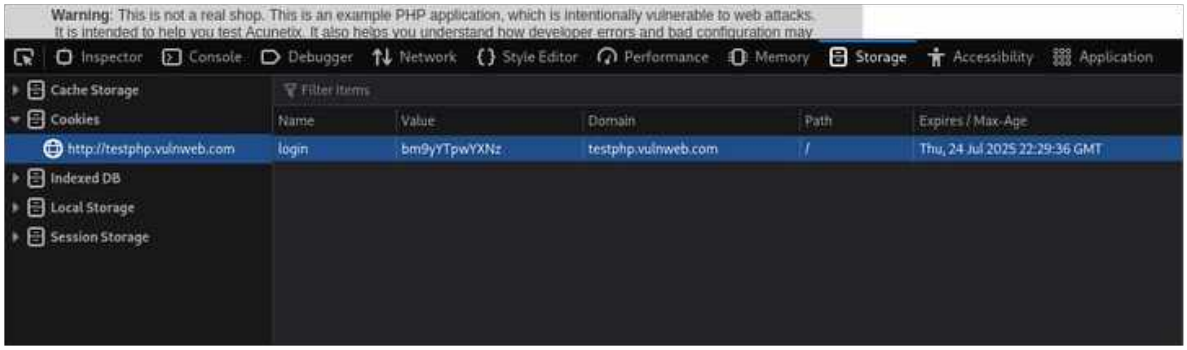

Figura 20 – Cookie Statico em Base64 (nora:pass) encontrado no login. Fonte: Da Autora.

Esse valor pôde ser reutilizado manualmente em outro *browser*, o que possibilita o *session hijacking* ou a *session forgery* com a simples cópia do *cookie*, teste que foi realizado com sucesso. Além disso, o sistema não invalidava o *cookie* no *logout*, demonstrando a falta

de tratativa de *authentication* e *authorization*. Deste modo, para resolver esse problema, é importante migrar para um modelo de *session management* com *random identifiers* e ativar as *security flags* dos *cookies*. A documentação do *MDN Web Docs* sobre segurança de *cookies* recomenda expirar o *cookie* no *logout* e definir um *lifetime* adequado, destacando as práticas essenciais para proteger a *communication* e a *user authenticity* (MDN Web Docs, 2025c).

4.4.3.4 Cross-Site Scripting (XSS)

Em seguida, foi identificada a vulnerabilidade de *Cross-Site Scripting* (*XSS*). Ao inserir o *command* <script>alert('XSS')</script> no *search field*, demonstrado na Figura 21, o *JavaScript code* foi executado no *browser*, comprovando a ocorrência de *reflected XSS*, como mostra a Figura 22, falha que permite o *cookie theft*, *user redirection* ou execução de *malicious scripts*.

Para lidar com cenários em que esse tipo de vulnerabilidade está presente, o *PortSwigger* recomenda uma abordagem em camadas para prevenir *XSS*, que inclui *input filtering*, *output encoding*, usar *response headers* adequados, como *Content-Type* e *X-Content-Type-Options*, e implementar uma *Content Security Policy* (*CSP*). Já a *CSP* instrui o *browser* a restringir quais *resources* podem ser carregados, servindo como última linha de defesa contra *XSS*. Portanto, o uso adequado de *CSP* é uma recomendação essencial para proteger *web applications* contra esse tipo de falha, como detalhado no estudo de segurança do *PortSwigger* (PortSwigger Web Security Academy, 2025) e na documentação do *MDN Web Docs* (MDN Web Docs, 2025a).

Figura 21 – Execução do payload para teste de XSS. Fonte: Da Autora

4.4.3.5 Cross-Site Request Forgery (CSRF)

Por fim, foi realizado um teste de *Cross-Site Request Forgery* (*CSRF*). Foi criado e executado, demonstrado na Figura 23, um *malicious form* (*CSRF*) que enviava uma *POST request* automática para `/update_email.php`; embora o teste não tenha apresentado um resultado visível ao ser aplicado no *site* (Figura 24), a falha potencial continua.

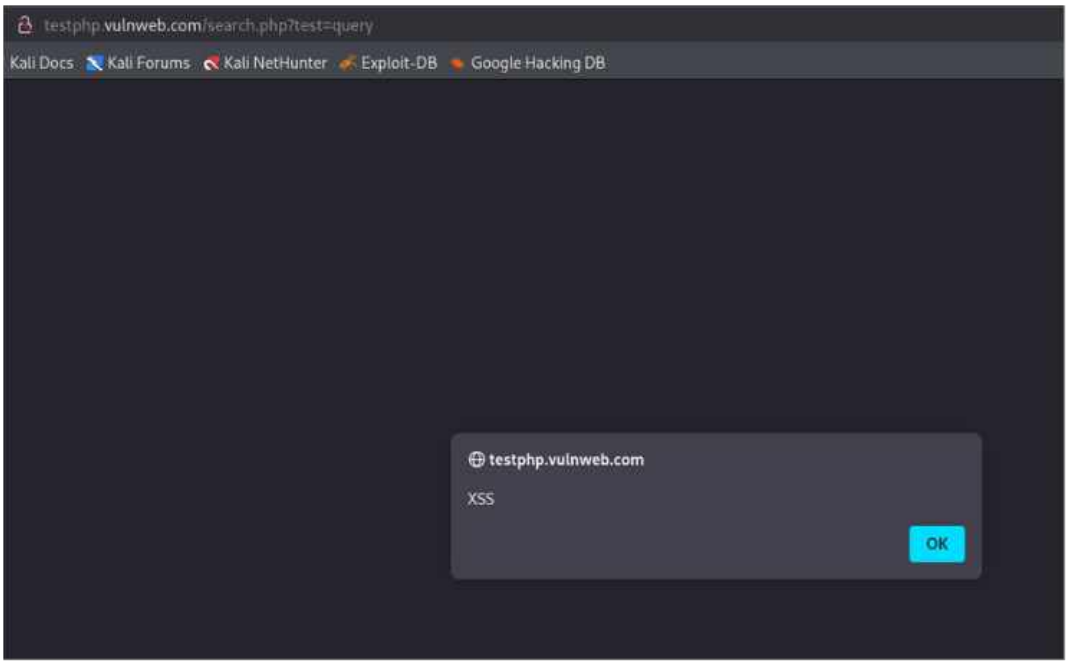

Figura 22 – Retorno do comando para teste de XSS com sucesso. Fonte: Da Autora

Assim, o *MDN Web Docs* observa que a defesa mais comum contra *CSRF* é utilizar *unpredictable tokens* embutidos nas páginas, que devem ser enviados de volta nas *state-changing requests*. O *server* verifica o valor do *token* e a ação só é executada se o *token* coincidir. Além disso, *request metadata headers*, como *Sec-Fetch-Site*, podem ser usados para verificar se a requisição é *cross-site* e rejeitá-la, caso seja o caso. Essas práticas ajudam a proteger contra ataques *CSRF*, como discutido nas diretrizes do *MDN Web Docs* (MDN Web Docs, 2025b).

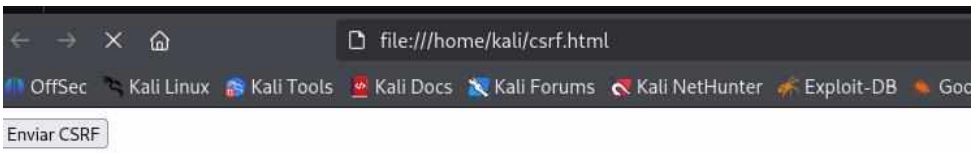

Figura 23 – Teste do CSRF pelo navegador utilizando o formulário malicioso HTML. Fonte: Da Autora.

4.4.4 Pós-Exploração

A fase de Pós-exploração foi conduzida com o objetivo de avaliar o impacto real das vulnerabilidades exploradas, especialmente no que se refere à possibilidade de exfiltração de dados e à análise das consequências práticas de cada falha identificada. Durante essa

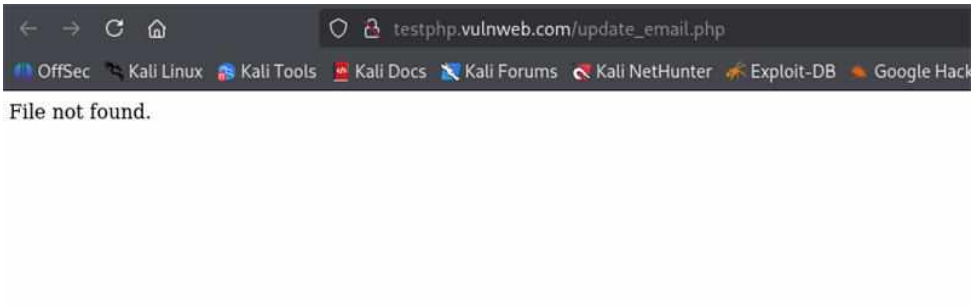

Figura 24 – Resultado do teste sem sucesso. Fonte: Da Autora.

fase, foram realizados levantamentos para validar as condições sob as quais os dados sensíveis poderiam ser extraídos e os riscos associados a cada vulnerabilidade mapeada na fase anterior. O foco estava nas falhas confirmadas, analisando como essas vulnerabilidades poderiam ser exploradas para afetar a operação e a segurança da aplicação em um cenário real. O *NIST SP 800-115* e o *OSSTMM* destacam a importância de realizar essa avaliação com rigor, garantindo que a análise de impacto seja fundamentada em dados concretos de exploração (SCARFONE et al., 2008; ISECOM, 2010).

Os testes realizados na pós-exploração foram direcionados a examinar o comportamento da aplicação após a exploração das vulnerabilidades críticas encontradas. Com base nas falhas de *SQL Injection*, exposição de arquivos sensíveis, autenticação via *cookie* fixo, *XSS* refletido e potencial *CSRF*, a análise teve como foco o acesso a dados sensíveis e a persistência do atacante dentro do sistema. A investigação também procurou identificar cenários em que a exploração dessas falhas poderia resultar em um comprometimento completo da aplicação, como o roubo de dados, a movimentação lateral para outros sistemas ou a alteração de informações sem a autorização adequada. Segundo a análise de Alhamed e Rahman (2023), a exploração dessas falhas pode criar um caminho para comprometer a integridade e a confidencialidade dos dados da organização (ALHAMED; RAHMAN, 2023).

A análise de *Data Exfiltration* foi um dos pontos-chave dessa fase de pós-exploração. Durante a execução dos testes, foi verificado que falhas como *SQL Injection* e a exposição de arquivos sensíveis permitiam a extração de informações confidenciais, como credenciais, estruturas de banco de dados e código-fonte. O acesso a arquivos como create.sql, .bak, .zip e prescription-tramadol.txt foi identificado como um risco significativo, pois esses arquivos poderiam conter dados cruciais para a continuidade do ataque e o comprometimento da segurança. A exposição desses arquivos segue as recomendações do *OWASP* e do ISO/IEC 27002, que enfatizam a importância de remover dados sensíveis de ambientes de produção e restringir o acesso público a diretórios (OWASP Foundation, 2025c; INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2022). Além disso, a autenticação via *cookie* fixo permitiu a persistência da sessão, assim, possibilitando que

um atacante reutilizasse o *cookie* para manter o acesso à aplicação sem a necessidade de reautenticação. Esse comportamento representa um risco crítico de segurança, como discutido por *MDN Web Docs*, que recomenda o uso de *secure cookies*, com atributos como *HttpOnly* e *SameSite*, para mitigar esse tipo de falha (MDN Web Docs, 2025c).

A tabela 5 compila os principais resultados obtidos e exfiltrados, levando em consideração a vulnerabilidade, seu potencial de exfiltração, o nível de risco apresentado e a recomendação de procedimento a ser seguido para lidar com ele.

Vulnerabilidade	**Potencial de Exfiltração / Observação**	**Severidade**	**Recomendação**
SQL Injection (login bypass)	Permite login sem credenciais, acesso não autorizado ao sistema e leitura de estrutura de banco de dados (`/admin/create.sql`).	Crítica	Utilizar consultas parametrizadas e validação de entrada.
Exposição de arquivos sensíveis	Arquivos `create.sql`, `.bak` e `.zip` expõem estrutura do banco e podem conter código-fonte ou credenciais.	Alta	Remover arquivos desnecessários e restringir acesso a diretórios sensíveis.
Autenticação via cookie estático	Cookie `login` em Base64 (`nora:pass`) permite clonagem de sessão e acesso contínuo sem credenciais.	Crítica	Implementar sessões seguras com cookies `HttpOnly`, `Secure` e expiração adequada.
XSS (*Cross-Site Scripting*) refletido	Injeção de `<script>alert('XSS')</script>` em campo de busca, possibilitando roubo de cookies e execução de scripts no navegador da vítima.	Alta	Escapar entradas/saídas, implementar CSP e validação no backend.
Potencial CSRF (Cross-Site Request Forgery)	Formulário `POST` para `/update_email.php` indica ausência de proteção anti-CSRF.	Indefinida	Implementar tokens CSRF e validar cabeçalhos `Origin` e `Referer`.

Tabela 5 – Levantamento consolidado de vulnerabilidades com potencial de exfiltração e recomendações. Fonte: Da Autora.

Dessa maneira, a avaliação do impacto foi realizada com base nas vulnerabilidades que permitiam o acesso não autorizado, a exfiltração de informações e a persistência do atacante no sistema. A Tabela 6, apresentada abaixo, reune os principais aspectos do impacto identificado com os testes, que considerou o risco de comprometimento da integridade da aplicação e da confidencialidade dos dados armazenados para avaliar os principais pontos encontrados.

4.5 Análise da metodologia

Após a conclusão da fase de Pós-exploração, toda a metodologia de *pentest* executada foi minuciosamente avaliada e revisada para validação. Durante esse processo de

Impacto Identificado	**Descrição**
Acesso não autorizado	SQL Injection permitiu login sem credenciais, comprometendo o controle de autenticação.
Exfiltração de informações	Arquivos sensíveis e XSS refletido indicam risco de roubo de dados e credenciais.
Persistência do atacante	Autenticação via cookie fixo permite sessões clonadas sem expiração.
Escalonamento potencial	Arquivos de backup podem conter senhas ou tokens que levem a outros sistemas.

Tabela 6 – Avaliação de impacto das vulnerabilidades exploradas. Fonte: Da Autora.

revisão, foram identificados pontos de melhoria e ajustes necessários para otimizar a execução das fases subsequentes, além de aumentar a eficiência do trabalho realizado. O principal objetivo da avaliação foi aprimorar os procedimentos seguidos, garantir que as ferramentas e comandos utilizados estivessem bem definidos, e desenvolver soluções para automatizar as etapas repetitivas do *pentest*, o que, conforme discutido por Shanley e Johnstone (2015), é essencial para manter a consistência e a reprodutibilidade do processo em diferentes cenários (SHANLEY; JOHNSTONE, 2015).

A primeira ação foi a melhoria do *checklist*, incorporando a indicação de comandos específicos e ferramentas a serem utilizadas em cada fase do teste. Essa atualização visou aumentar a clareza e objetividade da metodologia, tornando as diretrizes mais precisas e acessíveis para a execução das atividades em cada fase, como coleta de informações, enumeração, exploração e pós-exploração. Portanto, o *checklist* foi reestruturado para garantir que cada ação fosse bem documentada, com os comandos detalhados e as ferramentas recomendadas, permitindo que o processo fosse reproduzido de maneira consistente em testes futuros. A estruturação detalhada do *checklist* também facilita tanto a auditoria do processo quanto a avaliação da qualidade do trabalho realizado, como sugerido pelo PTES e pelo OSSTMM, que enfatizam a importância de documentação clara para garantir a conformidade e eficácia do *pentest* (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010).

Além disso, foi estabelecida a criação de *scripts* automatizados para a execução dos passos definidos no *checklist*, com base nas ferramentas e parâmetros recomendados para cada fase. A automação dessas tarefas visou aumentar a eficiência do processo de *pentest*, e também assegurar que os testes fossem realizados de maneira mais rápida e com uma margem reduzida de erro. Assim, com a implementação de scripts, a consistência nas fases de execução foi significativamente melhorada, como discutido por Shah (2020), que destaca a importância da automação na melhoria da precisão e velocidade durante os *pentests* (SHAH, 2020).

Consequentemente, essas melhorias fizeram com que a metodologia híbrida de

pentest ficasse mais robusta, eficiente e automatizada, o que aumentou a precisão dos testes realizados e garantiu que todas as etapas fossem bem documentadas e rastreáveis. Com a atualização do *checklist* e a implementação de *scripts*, o processo revisado se tornou mais ágil e adaptável, facilitando futuras atualizações e a adaptação da metodologia para novos cenários de segurança, como recomendado pelo NIST SP 800-115, que destaca a importância de revisões contínuas para manter a relevância de uma metodologia de *pentest* (SCARFONE et al., 2008).

4.6 Criação dos scripts automatizados

Os *scripts* automatizados foram criados com o objetivo de tornar as fases do *pentest* mais eficientes e consistentes, permitindo que a execução dos comandos utilizados nas etapas da metodologia fosse realizada de forma rápida e precisa. Utilizando a linguagem *Shell Script*, cada *script* foi desenvolvido para abordar diferentes fases do *pentest*, como Coleta de Informações, Enumeração e Exploração, sendo estruturados para utilizar as melhores ferramentas disponíveis. Os comandos foram previamente definidos para garantir a reprodutibilidade e a precisão dos resultados, uma prática recomendada em *frameworks* como PTES e OSSTMM (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010). Assim, a automação desses processos foi fundamental para melhorar a eficiência do *pentest*, possibilitando a execução das etapas de forma padronizada, com maior rapidez e menos margem de erro, como observado por Shah (2020) em sua análise sobre a importância da automação nos testes de segurança (SHAH, 2020).

4.6.1 Script de Coleta de Informações

O *script* de Coleta de Informações foi projetado para realizar tanto a coleta passiva quanto ativa de dados do alvo, integrando diversas ferramentas essenciais. Ele abrange um espectro amplo de informações que podem ser úteis em fases subsequentes do *pentest*, como mapeamento de subdomínios, identificação de tecnologias, e varredura de serviços expostos. Dessa maneira, ele se divide da seguinte forma:

Coleta Passiva: Com o uso de TheHarvester, o *script* coleta informações sobre subdomínios e e-mails, realizando consultas em motores de busca como Google e Bing. Ferramentas como WhatWeb são usadas para identificar as tecnologias empregadas no sistema alvo, criando um mapeamento das ferramentas e *frameworks* utilizados pela aplicação. A coleta passiva é importante para garantir que as informações sensíveis sejam identificadas sem interação direta com o sistema, uma abordagem defensiva recomendada por Shanley e Johnstone (2015), que destacam a necessidade de respeitar o escopo acordado em *pentests* (SHANLEY; JOHNSTONE, 2015).

Coleta Ativa: O script também realiza coleta ativa, utilizando ferramentas como

DNSRecon para realizar tentativas de transferência de zona DNS (AXFR) e Whois para coletar dados de registro de domínio. A ferramenta Nmap é utilizada para realizar uma varredura mais detalhada dos serviços em execução no servidor, o que é fundamental para mapear a superfície de ataque de forma aprofundada. A combinação de coleta passiva e ativa é um princípio básico de segurança ofensiva, alinhado às práticas de OSSTMM e PTES, que recomendam a coleta abrangente de dados antes de avançar para fases mais intrusivas do *pentest* (PROJECT, 2025a; REEVES; CONTRIBUTORS, 2025).

4.6.2 Script de Enumeração

O script de Enumeração tem como foco a descoberta de diretórios e arquivos ocultos, além da identificação de arquivos sensíveis e a verificação de cabeçalhos HTTP. Essa fase é crucial, pois ajuda a mapear pontos de vulnerabilidade no sistema e fornecer informações detalhadas para as fases seguintes do *pentest*.

Descoberta de Diretórios e Arquivos Ocultos: O Gobuster é utilizado para realizar um brute force em diretórios e arquivos, testando extensões como .php, .html, .bak, .zip, .sql, entre outras. Caso o Gobuster falhe, o DirBuster é utilizado como alternativa. Esses testes ajudam a identificar recursos expostos que não estão visíveis de imediato, mas que podem representar riscos de segurança significativos, como os apontados por PortSwigger em suas recomendações para testes de *Cross-Site Scripting* (XSS) e SQL Injection (SQLi) (PortSwigger Web Security Academy, 2025).

Verificação de Arquivos Sensíveis: O script realiza a verificação da existência de arquivos sensíveis, como `/admin/create.sql`, `/index.bak` e `/index.zip`, que podem estar expostos e representar riscos graves, além de identificar outros arquivos críticos que possam estar disponíveis para exploração. A exposição de arquivos sensíveis, como discutido no *Verizon DBIR 2024*, é um ponto de ataque comum, principalmente em sistemas mal configurados (Verizon, 2024).

Verificação de Cabeçalhos HTTP: Ferramentas como Curl são utilizadas para coletar os cabeçalhos HTTP, permitindo uma análise da configuração de segurança do servidor. Isso inclui a identificação de versões de software e a verificação de cabeçalhos de segurança, como *Strict-Transport-Security* e *X-Content-Type-Options*, que ajudam a proteger contra ataques Man-in-the-Middle (MITM) (Compyl, 2025).

4.6.3 Script de Exploração

Por último, *script* de Exploração foi desenvolvido para validar as vulnerabilidades encontradas durante as fases anteriores, como SQL Injection, XSS, CSRF e vulnerabilidades de autenticação. Ele aplica testes controlados para validar o impacto real dessas falhas e verificar sua exploração prática. Portanto, o *script* considerou testes das vulnerabilidades

anteriormente testadas em ambiente real, listadas abaixo.

SQL Injection: O SQLMap foi utilizado para explorar a vulnerabilidade de SQL Injection no campo de login da aplicação, utilizando o comando –dump para tentar extrair dados do banco de dados. Esse ataque valida a falha e comprova a exploração bem-sucedida da vulnerabilidade. A recomendação do OWASP para mitigar SQL Injection envolve o uso de consultas parametrizadas ou ORMs que separam dados de código (OWASP Foundation, 2025c).

XSS Refletido: O *script* realiza um teste simples de XSS refletido, inserindo um payload JavaScript no campo de busca, verificando se o código é executado na resposta do servidor. Isso valida se a aplicação é vulnerável a XSS, como discutido por PortSwigger, que recomenda diversas camadas de defesa contra XSS, como codificação de entrada e implementação de Política de Segurança de Conteúdo (CSP) (PortSwigger Web Security Academy, 2025).

CSRF: O *script* gera um formulário CSRF que envia uma requisição *POST* maliciosa para `/update_email.php`.

Verificação de Arquivos Sensíveis e Sessão: O *script* verifica novamente a existência de arquivos sensíveis e realiza um teste para avaliar a persistência da sessão, verificando se o sistema é vulnerável a cookie fixation, uma falha crítica que compromete a segurança de autenticação (MDN Web Docs, 2025c).

4.7 Criação dos templates de relatórios

A criação dos templates dos relatórios é uma etapa fundamental para garantir que a documentação do *pentest* seja organizada, clara e eficaz, atendendo tanto aos requisitos técnicos quanto executivos. No desenvolvimento da metodologia híbrida, foram projetados dois tipos de relatórios: o relatório técnico e o relatório executivo, cada um com seu público-alvo específico e objetivos distintos. A estrutura e o formato dos relatórios foram baseados nas melhores práticas de segurança ofensiva, conforme descrito em *frameworks* como PTES, OWASP e NIST SP 800-115 (PENETRATION TESTING EXECUTION STANDARD (PTES), ; OWASP FOUNDATION, 2025; SCARFONE et al., 2008).

A estrutura dos relatórios foi cuidadosamente planejada para garantir que fossem completos e adequados a ambos os públicos (técnicos e executivos). Assim, utilizou-se o processo e os resultados do estudo de caso realizado na seção 4.4 para a modelação dos *layouts* e criação dos exemplos. Ademais, o LaTeX foi escolhido como ferramenta para formatar os documentos, pois oferece flexibilidade e controle sobre o layout, garantindo que os relatórios tenham uma aparência profissional e sejam facilmente modelados. Por fim, os relatórios foram convertidos para o formato .docx, visando uma edição de maneira

mais rápida e facilitada para aqueles que usarão para fins reais, permitindo a adição ou remoção de seções conforme necessário, sem comprometer a integridade do conteúdo.

4.7.1 Relatório Técnico

O relatório técnico destina-se a profissionais de segurança da informação, como administradores de sistema, analistas de segurança e engenheiros de redes, que necessitam de um nível de detalhe profundo sobre as vulnerabilidades encontradas, as ferramentas utilizadas e as evidências coletadas. A principal função deste relatório é fornecer uma documentação exaustiva do *pentest*, permitindo que as falhas de segurança sejam compreendidas e corrigidas com precisão.

A criação do template para o relatório técnico seguiu as recomendações do PTES, que sugere que todas as etapas do *pentest* sejam claramente documentadas, incluindo os comandos executados, as ferramentas utilizadas e as evidências coletadas (PENETRATION TESTING EXECUTION STANDARD (PTES),). Cada vulnerabilidade identificada é descrita de forma detalhada, incluindo:

- **Resultado dos testes:** O tipo de falha encontrada, com uma explicação técnica sobre como ela afeta o sistema.
- **Fase de pós-exploração:** Faz um levantamento de todo o processo, realizando a exfiltração de dados e a análise do impacto.
- **Evidências:** A seção de apêndice é reservada as capturas de tela, logs e outros artefatos que comprovam a existência da vulnerabilidade.

A formatação foi feita de forma a permitir que o relatório técnico seja acessível, organizado e fácil de entender para técnicos, mas também oferece detalhes suficientes para permitir a correção e remediação completa das falhas. A separação das fases, como coleta de informações, enumeração, exploração e pós-exploração, foi mantida, garantindo que cada etapa do *pentest* fosse documentada com precisão.

4.7.2 Relatório executivo

O relatório executivo foi criado para ser compreensível por gestores e *stakeholders* não técnicos. Dessa forma, esse relatório resume as descobertas do *pentest*, focando nas vulnerabilidades críticas e nos riscos associados, com uma linguagem acessível para aqueles que não possuem conhecimentos técnicos aprofundados.

A criação do *template* para o relatório executivo seguiu as orientações de ISO/IEC 19011 sobre a documentação de auditorias e relatórios técnicos (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO), 2018), garantindo que os pontos mais

importantes fossem destacados de forma clara e objetiva. As seções principais desse relatório incluem:

- **Vulnerabilidades Encontradas:** Uma lista resumida das falhas com uma explicação sobre os riscos que elas representam para a segurança e operação da aplicação.
- **Riscos Identificados:** Seção destinada a descrever como as vulnerabilidades encontradas podem acerretar em riscos de diversos graus de severidade.
- **Recomendações de Ação:** Soluções sugeridas para mitigar os riscos identificados, apresentadas de forma direta e prática, sem detalhes excessivamente técnicos.
- **Acompanhamento e próximos passos:** Apresenta uma estrátegia de monitoramento e ações de governança a serem efetuadas.

Este relatório serve como um documento de decisão, fornecendo uma visão clara das ações que precisam ser tomadas para proteger a infraestrutura da organização, com foco em reduzir a exposição a riscos cibernéticos.

4.8 Estruturação do repositório no Github

A estruturação do repositório no *GitHub* foi uma etapa pensada para organizar e centralizar todos os materiais relacionados à metodologia de *pentest* desenvolvida. Esse repositório funciona como uma plataforma colaborativa, permitindo que materiais como o *checklist* detalhado, os *scripts* automatizados e os *templates* de relatório sejam facilmente acessados, adaptados e reutilizados em futuros testes de segurança. A centralização da metodologia em um repositório segue as melhores práticas de versionamento e controle de qualidade, conforme recomendado pelos frameworks PTES e OSSTMM, que enfatizam a importância de uma documentação clara e rastreável para garantir a eficiência e consistência do processo de *pentest* (PENETRATION TESTING EXECUTION STANDARD (PTES), ; ISECOM, 2010).

O repositório foi configurado utilizando o *Git*, garantindo o versionamento adequado de todos os materiais criados, permitindo o controle das versões e facilitando a atualização contínua do conteúdo. O uso do *GitHub* torna o processo colaborativo, possibilitando a visualização e acompanhamento das modificações realizadas, além de garantir que a versão mais atualizada da metodologia esteja sempre acessível. Como destacado por Shanley e Johnstone (2015), o versionamento não só facilita a evolução controlada do processo de *pentest*, mas também a auditoria contínua da metodologia (SHANLEY; JOHNSTONE, 2015).

Além disso, o repositório também inclui um *README* detalhado que orienta os usuários sobre como utilizar os scripts automatizados e executar as fases de coleta de informações, enumeração e exploração. Também, apresenta de maneira estruturada a organização do repositório, os requisitos do ambiente (como ferramentas necessárias), e as instruções para a execução dos scripts em sistemas Linux (Kali/Ubuntu/Debian).

O acesso ao repositório pode ser feito pelo link: <https://github.com/giovannamevan/metodologia-pentest>, onde toda a metodologia, incluindo o *checklist* e os scripts, pode ser acessada e utilizada.

5 Conclusão

Em síntese, este trabalho ressalta a importância de se criar uma metodologia de *pentest* adaptada ao contexto de sistemas web, que seja capaz de identificar vulnerabilidades de forma eficaz, enquanto respeita as limitações legais e operacionais típicas desse ambiente. O objetivo principal foi desenvolver uma abordagem estruturada que integrasse as melhores práticas de *frameworks* de segurança ofensiva amplamente reconhecidos, como o PTES, OSSTMM e OWASP WSTG, e que fosse aplicável a sistemas web usados por universidades. A metodologia proposta demonstrou ser eficaz, permitindo a realização de testes de intrusão com segurança e consistência, alinhados às normas éticas e legais exigidas para o ambiente acadêmico.

Durante o desenvolvimento da metodologia, foi feita a análise de diferentes abordagens de *pentest*, com ênfase no modelo *Black Box* e na integração dos *frameworks* PTES, OSSTMM e OWASP WSTG. Além disso, a criação de um *checklist* detalhado, *checklist* automatizados e templates de relatórios garantiu a padronização e a eficiência na execução dos testes, como recomendado por Shah (2020) (SHAH, 2020) e Shanley e Johnstone (2015) (SHANLEY; JOHNSTONE, 2015).

A validação da metodologia foi realizada em um ambiente controlado e autorizado, o domínio testphp.vulnweb.com, um site amplamente utilizado para a prática de *pentests*, garantindo que os testes fossem executados de forma ética e legal, conforme sugerido por Acunetix (ACUNETIX, 2025). Dessa maneira, os resultados obtidos confirmaram a eficácia da metodologia, que demonstrou ser adaptável a diferentes contextos acadêmicos, permitindo a identificação de vulnerabilidades críticas e proporcionando uma base sólida para a melhoria contínua das defesas dos sistemas.

Como trabalhos futuros, a metodologia pode ser expandida para incluir novas abordagens de segurança em diferentes ambientes, como a avaliação de sistemas *mobile* ou em redes privadas, que têm se tornado cada vez mais comuns em ambientes acadêmicos. Igualmente, a criação de novos *scripts* automatizados, juntamente com a adaptação da metodologia às novas ameaças cibernéticas, será fundamental para manter a eficácia do processo de *pentest*.

Referências

ACUNETIX. **testphp.vulnweb.com - Vulnerability Web Application**. 2025. Acesso em: 02 set. 2025. Disponível em: <https://www.acunetix.com/vulnerable-web-applications/>. Citado 2 vezes nas páginas 28 e 59.

ALHAMED, M.; RAHMAN, M. M. H. A systematic literature review on penetration testing in networks: Future research directions. **Applied Sciences**, v. 13, n. 12, p. 6986, 2023. Disponível em: <https://www.mdpi.com/2076-3417/13/12/6986>. Citado 2 vezes nas páginas 24 e 50.

ASQ. **ISO 19011 — Guidelines for Auditing Management Systems**. 2025. <https://asq.org/quality-resources/iso-19011>. Resumo e escopo do padrão. Acesso em: 12 ago. 2025. Citado na página 18.

AUTORIDADE NACIONAL DE PROTEçãO DE DADOS (ANPD). **Guia de Segurança da Informação para Agentes de Tratamento de Pequeno Porte**. 2021. Acesso em: 17 ago. 2025. Disponível em: <https://www.gov.br/anpd/pt-br/centrais-de-conteudo/materiais-educativos-e-publicacoes/guia-vf.pdf>. Citado na página 22.

BRASIL. **Lei nº 12.965, de 23 de abril de 2014 (Marco Civil da Internet)**. 2014. Acesso em: 17 ago. 2025. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2011-2014/2014/lei/l12965.htm>. Citado na página 22.

_____. **Lei nº 13.709, de 14 de agosto de 2018 (Lei Geral de Proteção de Dados Pessoais)**. 2018. Acesso em: 16 ago. 2025. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>. Citado 2 vezes nas páginas 11 e 22.

_____. **Lei nº 13.964, de 24 de dezembro de 2019 (altera o CPP e introduz os arts. 158-A a 158-F)**. 2019. Acesso em: 17 ago. 2023. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2019-2022/2019/lei/l13964.htm>. Citado na página 23.

CICHONSKI, P.; MILLAR, T.; GRANCE, T.; SCARFONE, K. **Computer Security Incident Handling Guide**. Gaithersburg, MD, 2012. Acesso em: 16 ago. 2025. Disponível em: <https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-61r2.pdf>. Citado 3 vezes nas páginas 19, 23 e 43.

CIRT. **Nikto Web Server Scanner**. 2025. <https://cirt.net/Nikto2>. Scanner para arquivos perigosos e más configurações em servidores web. Acesso em: 16 ago. 2025. Citado 5 vezes nas páginas 7, 22, 28, 31 e 33.

COMMON Vulnerability Scoring System (CVSS) v4.0 — Specification Document. [S.l.], 2024. Lançado originalmente em 1 nov. 2023; versão consultada de 18 jun. 2024. Acesso em: 10 ago. 2025. Disponível em: <https://www.first.org/cvss/v4-0/cvss-v40-specification.pdf>. Citado na página 16.

Compyl. **Penetration Testing vs. Vulnerability Scanning**. 2025. Acesso em: 12 ago. 2025. Disponível em: <https://compyl.com/blog/penetration-testing-vs-vulnerability-scanning/>. Citado 3 vezes nas páginas 17, 41 e 54.

DOUPÉ, A.; COVA, M.; VIGNA, G. Why johnny can't pentest: An analysis of black-box web vulnerability scanners. In: **Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2010)**. Springer, 2010. (Lecture Notes in Computer Science), p. 111–131. Disponível em: <https://sites.cs.ucsb.edu/~vigna/publications/2010_doupe_cova_vigna_dimva10.pdf>. Citado 4 vezes nas páginas 12, 13, 25 e 30.

European Union Agency for Cybersecurity (ENISA). **ENISA Threat Landscape 2024**. Atenas, 2024. Acesso em: 10 ago. 2025. Disponível em: <https://www.enisa.europa.eu/publications/enisa-threat-landscape-2024>. Citado 5 vezes nas páginas 11, 15, 16, 17 e 42.

FireMon. **Pen Test vs Vulnerability Scan: Key Differences**. 2025. Acesso em: 12 ago. 2025. Disponível em: <https://www.firemon.com/blog/pen-test-vs-vulnerability-scan/>. Citado 2 vezes nas páginas 17 e 42.

FIRST.Org, Inc. **Common Vulnerability Scoring System (CVSS) v3.1 — User Guide**. 2019. <https://www.first.org/cvss/v3-1/user-guide>. Acesso em: 10 ago. 2025. Citado na página 16.

G., B. D. A.; STAMPAR, M. **sqlmap project**. 2025. Ferramenta de detecção e exploração de SQL Injection. Acesso em: 16 ago. 2025. Disponível em: <https://sqlmap.org/>. Citado 6 vezes nas páginas 7, 21, 22, 28, 31 e 33.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO/IEC 27002:2022 — Information security, cybersecurity and privacy protection — Information security controls**. Genebra, 2022. Acesso em: 08 ago. 2025. Disponível em: <https://www.iso.org/standard/75652.html>. Citado 4 vezes nas páginas 11, 15, 17 e 50.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO). **ISO 19011:2018 — Guidelines for auditing management systems**. Genebra, 2018. Acesso em: 12 ago. 2025. Disponível em: <https://www.iso.org/standard/70017.html>. Citado 2 vezes nas páginas 18 e 56.

ISECOM. **Open Source Security Testing Methodology Manual**. 3.0. ed. [S.l.], 2010. Acesso em 27 ago 2025. Citado na página 27.

ISECOM. **The Open Source Security Testing Methodology Manual (OSSTMM), Version 3**. 2010. <https://www.isecom.org/OSSTMM.3.pdf>. Acesso em: 14 ago. 2025. Citado 17 vezes nas páginas 7, 11, 12, 13, 19, 20, 23, 24, 27, 28, 30, 32, 44, 50, 52, 53 e 57.

KENT, K.; CHEVALIER, S.; GRANCE, T.; DANG, H. **Guide to Integrating Forensic Techniques into Incident Response**. Gaithersburg, MD, 2006. Acesso em: 16 ago. 2025. Disponível em: <https://csrc.nist.gov/pubs/sp/800/86/final>. Citado na página 23.

LARAMEE, C.; CONTRIBUTORS. **theHarvester**. 2025. Coleta de fontes abertas: e-mails, domínios e metadados. Acesso em: 14 ago. 2025. Disponível em: <https://github.com/laramies/theHarvester>. Citado 7 vezes nas páginas 7, 20, 22, 28, 31, 32 e 35.

MDN Web Docs. **Content Security Policy (CSP)**. 2025. Última modificação em 4 de julho de 2025. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CSP>. Citado na página 48.

_____. **Cross-site request forgery (CSRF)**. 2025. Última modificação em 7 de junho de 2025. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/Security/Attacks/CSRF>. Citado na página 49.

_____. **Set-Cookie header**. 2025. Última modificação em 16 de julho de 2025. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/Set-Cookie>. Citado 3 vezes nas páginas 48, 51 e 55.

MJSP. **Cadeia de Custódia — Ministério da Justiça e Segurança Pública**. 2023. Acesso em: 17 ago. 2025. Disponível em: <https://www.gov.br/mj/pt-br/assuntos/sua-seguranca/seguranca-publica/cadeia-de-custodia>. Citado na página 23.

National Vulnerability Database (NVD). **Vulnerability Metrics — CVSS**. 2022. <https://nvd.nist.gov/vuln-metrics/cvss>. Página atualizada periodicamente. Acesso em: 10 ago. 2025. Citado na página 16.

_____. **CVSS v4.0 Official Support at NVD**. 2024. <https://nvd.nist.gov/general/news/cvss-v4-0-official-support>. Acesso em: 10 ago. 2025. Citado 3 vezes nas páginas 16, 41 e 46.

NCSC-FI / Traficom. **The National Cyber Security Centre Finland's weekly review – 40/2023**. 2023. <https://www.kyberturvallisuuskeskus.fi/en/news/national-cyber-security-centre-finlands-weekly-review-402023>. Registra aumento de QR-phishing. Acesso em: 12 ago. 2025. Citado na página 16.

NIST Computer Security Resource Center. **Rules of Engagement (ROE) — CSRC Glossary**. 2024. <https://csrc.nist.gov/glossary/term/rules_of_engagement>. Baseado no NIST SP 800-115. Acesso em: 12 ago. 2025. Citado 4 vezes nas páginas 17, 19, 21 e 23.

_____. **INFOSEC — Information Security (CSRC Glossary)**. 2025. <https://csrc.nist.gov/glossary/term/INFOSEC>. Acesso em: 08 ago. 2025. Citado 3 vezes nas páginas 11, 15 e 17.

OWASP FOUNDATION. **OWASP Web Security Testing Guide**. [S.l.], 2020. Acesso em 27 ago 2025. Citado 2 vezes nas páginas 32 e 33.

OWASP Foundation. **OWASP Benchmark Project**. 2025. <https://owasp.org/www-project-benchmark/>. Suíte de testes para avaliação de ferramentas SAST/DAST/IAST. Acesso em: 20 ago. 2025. Citado 2 vezes nas páginas 25 e 27.

_____. **OWASP DirBuster Project**. 2025. <https://owasp.org/www-project-dirbuster/>. Projeto legado para descoberta de diretórios. Acesso em: 14 ago. 2025. Citado 6 vezes nas páginas 7, 21, 22, 31, 33 e 39.

_____. **SQL Injection Prevention Cheat Sheet**. 2025. Disponível em: <https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html>. Citado 2 vezes nas páginas 50 e 55.

OWASP FOUNDATION. **WSTG — Latest: Introduction**. 2025. Taxonomia de white/black/gray-box e enquadramento de atividades. Acesso em: 13 ago. 2025. Disponível em: <https://owasp.org/www-project-web-security-testing-guide/latest/2-Introduction/>. Citado 17 vezes nas páginas 7, 11, 12, 13, 18, 19, 20, 22, 23, 24, 25, 27, 30, 32, 35, 39 e 55.

PENETRATION Test Guidance. [S.l.], 2022. Exigências sobre ROE, vetores e autorização. Acesso em: 13 ago. 2025. Disponível em: <https://www.fedramp.gov/assets/resources/documents/CSP_Penetration_Test_Guidance.pdf>. Citado na página 18.

PENETRATION TESTING EXECUTION STANDARD (PTES). **PTES Technical Guidelines**. Acesso em: 13 ago. 2025. Disponível em: <https://www.pentest-standard.org/index.php/PTES_Technical_Guidelines>. Citado 19 vezes nas páginas 7, 11, 12, 13, 19, 20, 23, 24, 27, 30, 31, 32, 33, 44, 52, 53, 55, 56 e 57.

POLíCIA CIENTíFICA DO ESPíRITO SANTO (PCIES). **Manual da Cadeia de Custódia**. 2024. Acesso em: 17 ago. 2025. Disponível em: <https://pci.es.gov.br/Media/PCIES/Manuais/Manual%20de%20Cadeia%20de%20Cust%C3%B3dia%202024%2005.07.pdf>. Citado na página 23.

PORTSWIGGER. **Burp Suite Documentation**. 2025. Interceptação, manipulação e automação de testes web. Acesso em: 14 ago. 2025. Disponível em: <https://portswigger.net/burp/documentation>. Citado 6 vezes nas páginas 7, 21, 22, 28, 31 e 33.

PortSwigger Web Security Academy. **What is cross-site scripting (XSS) and how to prevent it?** 2025. Disponível em: <https://portswigger.net/web-security/cross-site-scripting>. Citado 3 vezes nas páginas 48, 54 e 55.

PROJECT, N. **Nmap Reference Guide**. 2025. Guia de referência oficial do Nmap. Acesso em: 14 ago. 2025. Disponível em: <https://nmap.org/book/man.html>. Citado 8 vezes nas páginas 7, 20, 22, 31, 32, 35, 37 e 54.

_____. **Nmap Scripting Engine (NSE) Guide**. 2025. Documentação oficial do NSE. Acesso em: 14 ago. 2025. Disponível em: <https://nmap.org/book/nse.html>. Citado 2 vezes nas páginas 20 e 37.

REEVES, O.; CONTRIBUTORS. **Gobuster**. 2025. Força bruta de diretórios, arquivos e DNS. Acesso em: 14 ago. 2025. Disponível em: <https://github.com/OJ/gobuster>. Citado 9 vezes nas páginas 7, 21, 22, 31, 32, 33, 39, 40 e 54.

SCARFONE, K.; SOUPPAYA, M.; CODY, A.; OREBAUGH, A. **Technical Guide to Information Security Testing and Assessment**. Gaithersburg, MD, 2008. Acesso em: 12 ago. 2025. Disponível em: <https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-115.pdf>. Citado 12 vezes nas páginas 12, 17, 18, 19, 20, 21, 23, 24, 37, 50, 53 e 55.

SECRETARIA DE GOVERNO DIGITAL — PORTAL GOV.BR. **Guia de Boas Práticas — Lei Geral de Proteção de Dados (LGPD)**. 2020. Acesso em: 17 ago. 2025. Disponível em: <https://www.gov.br/governodigital/pt-br/privacidade-e-seguranca/guias/guia_lgpd.pdf>. Citado na página 22.

SHAH, M. P. **Comparative Analysis of the Automated Penetration Testing Tools**. Dissertação (Mestrado) — National College of Ireland, Dublin, 2020. Benchmark com OWASP Benchmark; recomenda combinações de scanners. Disponível em: <https://norma.ncirl.ie/4165/1/mandarprashantshah.pdf>. Citado 8 vezes nas páginas 11, 13, 14, 24, 25, 52, 53 e 59.

SHANLEY, A.; JOHNSTONE, M. N. Selection of penetration testing methodologies: A comparison and evaluation. In: **Proceedings of the 13th Australian Information Security Management Conference**. Perth, Western Australia: Security Research Institute, Edith Cowan University, 2015. p. 65–72. Disponível em: <https://ro.ecu.edu.au/ism/182/>. Citado 13 vezes nas páginas 13, 14, 23, 24, 26, 27, 28, 29, 32, 52, 53, 57 e 59.

TEAM, O. Z. **Forced Browse**. 2025. Descoberta de diretórios e arquivos com OWASP ZAP. Acesso em: 14 ago. 2025. Disponível em: <https://www.zaproxy.org/docs/desktop/addons/forced-browse/>. Citado 4 vezes nas páginas 21, 31, 33 e 39.

_____. **OWASP ZAP Documentation**. 2025. Documentação geral do ZAP. Acesso em: 16 ago. 2025. Disponível em: <https://www.zaproxy.org/docs/>. Citado 6 vezes nas páginas 7, 21, 22, 28, 31 e 33.

TechTarget SearchSecurity. **The differences between pen tests vs. vulnerability scanning**. 2025. Acesso em: 13 ago. 2025. Disponível em: <https://www.techtarget.com/searchsecurity/tip/The-differences-between-pen-tests-vs-vulnerability-scanning>. Citado 3 vezes nas páginas 17, 41 e 46.

The CVE Program. **Overview — About the CVE® Program**. 2025. <https://www.cve.org/about/overview>. Acesso em: 10 ago. 2025. Citado na página 16.

URBANADVENTURER; CONTRIBUTORS. **WhatWeb**. 2025. Fingerprinting de tecnologias e versões. Acesso em: 14 ago. 2025. Disponível em: <https://github.com/urbanadventurer/WhatWeb>. Citado 7 vezes nas páginas 7, 20, 22, 28, 31, 32 e 35.

Verizon. **Data Breach Investigations Report (DBIR)**. Basking Ridge, NJ, 2024. Acesso em: 10 ago. 2025. Disponível em: <https://www.verizon.com/business/resources/reports/2024-dbir-data-breach-investigations-report.pdf>. Citado 5 vezes nas páginas 11, 14, 16, 44 e 54.