

Abordagem Prática de Análise de Vulnerabilidades Web: Black Box e Grey Box com OWASP Top 10

Leonardo Osvald de Souza¹, Vitalino Pitt¹

¹Ciência da Computação – Atitus Educação
Passo Fundo – RS – Brasil

1121661@atitus.edu.br, vitalino.pitt@atitus.edu.br

Abstract. *The increasing reliance on connected systems has significantly expanded organizations' exposure to cyber threats, particularly in web applications and APIs. Vulnerabilities classified in the OWASP Top 10 remain among the most exploited by malicious actors, emphasizing the need for secure development practices and a solid understanding of their impact. This study aims to analyze the vulnerabilities listed in the OWASP Top 10 (2021) through experiments conducted in intentionally vulnerable environments. Two widely known platforms were selected for this purpose: Metasploitable 3, used for Black Box testing, and OWASP Juice Shop, used in a Grey Box approach with access to the source code. The methodology includes reconnaissance, scanning, exploitation, and validation phases using tools such as Nmap, OWASP ZAP, SQLMap, and Hydra, in addition to manual code analysis. The expected results involve demonstrating the exploitation of the main identified vulnerabilities, assessing their severity, and presenting mitigation strategies aligned with security best practices. This research aims to support developers and security analysts by highlighting the importance of continuous assessments and structured pentesting methodologies for building secure and resilient applications. Keywords: Information Security; OWASP Top 10; Penetration Testing; Vulnerabilities; Web Applications.*

Resumo. *A crescente dependência de sistemas conectados elevou significativamente a exposição de organizações a ameaças cibernéticas, especialmente em aplicações web e APIs. Vulnerabilidades classificadas pelo OWASP Top 10 continuam entre as mais exploradas por agentes mal-intencionados, tornando essencial a compreensão de seus impactos e a adoção de práticas de desenvolvimento seguro. Este trabalho tem como objetivo analisar as vulnerabilidades listadas no OWASP Top 10 (2021) por meio de experimentos conduzidos em ambientes deliberadamente vulneráveis. Para isso, foram selecionadas duas plataformas amplamente utilizadas em treinamentos de segurança: Metasploitable 3, para testes do tipo Black Box, e OWASP Juice Shop, para a abordagem Grey Box com acesso ao código-fonte. A metodologia inclui etapas de reconhecimento, varredura, exploração e validação utilizando ferramentas como Nmap, OWASP ZAP, SQLMap e Hydra, além de análise manual de trechos de código. Os resultados esperados envolvem demonstrar a exploração das principais vulnerabilidades identificadas, compreender sua criticidade e apresentar recomendações de mitigação alinhadas às boas práticas de segurança. A pesquisa busca*

contribuir para a formação de desenvolvedores e analistas de segurança, reforçando a importância de avaliações contínuas e metodologias estruturadas de pentest na construção de aplicações robustas.

Palavras-chave: Segurança da Informação; OWASP Top 10; Pentest; Vulnerabilidades; Aplicações Web.

1. Introdução

A crescente digitalização de serviços nas áreas corporativa, governamental e pessoal ampliou significativamente a dependência de sistemas conectados. Esse cenário aumentou também a superfície de ataque disponível para agentes maliciosos, que exploram vulnerabilidades para obter acessos indevidos, manipular dados ou comprometer a disponibilidade dos serviços. Dessa forma, a segurança da informação tornou-se um dos pilares essenciais para garantir a integridade e a confiabilidade dos sistemas modernos.

Uma vulnerabilidade consiste em uma fraqueza capaz de comprometer a confidencialidade, integridade ou disponibilidade das informações, Almeida (2021). Em aplicações web e APIs, tais falhas são frequentes devido à alta complexidade arquitetural, à crescente demanda por integrações e à pressão por ciclos de desenvolvimento cada vez mais rápidos. Como consequência, falhas de segurança podem resultar em vazamento de dados sensíveis, perdas financeiras, danos reputacionais e interrupção de serviços críticos.

Nesse contexto, iniciativas como o OWASP Top 10 desempenham papel fundamental ao catalogar e classificar as vulnerabilidades mais comuns observadas em sistemas ao redor do mundo. Esse documento, amplamente adotado pela comunidade de segurança, orienta desenvolvedores e equipes técnicas na identificação e mitigação de riscos prioritários.

Diante da relevância do tema, o objetivo geral deste trabalho é identificar, explorar e analisar vulnerabilidades presentes nas listas do OWASP Top 10 por meio de testes de intrusão em ambientes vulneráveis.

Essa abordagem busca não apenas demonstrar a presença e o impacto das vulnerabilidades, mas também apresentar técnicas de detecção e medidas de mitigação, contribuindo para a formação de boas práticas de segurança no desenvolvimento e manutenção de aplicações.

2. Referencial Teórico

Esta seção apresenta os fundamentos teóricos que embasam a pesquisa, abordando os principais conceitos, normas e vulnerabilidades relacionados à segurança de aplicações web. O entendimento desses elementos é essencial para a correta identificação, análise e mitigação de riscos de segurança, bem como para a execução dos testes de invasão propostos neste trabalho.

Inicialmente, será detalhado o OWASP Top 10, documento de referência global que cataloga as ameaças mais críticas e recorrentes em aplicações web, servindo como guia prioritário para esforços de segurança. Em seguida, o referencial avança para a discussão sobre Segurança de APIs e Aplicações Web, contextualizando a evolução das arquiteturas de software e os novos vetores de ataque decorrentes do uso massivo de interfaces de programação.

Serão também descritas as Vulnerabilidades mais Comuns em Aplicações Web, com foco naquelas constantes no OWASP Top 10 2021 – incluindo Quebra de Controle de Acesso, Falhas Criptográficas, Injeção e Quebra de Autenticação –, explicando seus mecanismos, impactos e formas de prevenção.

Por fim, aborda-se o conceito de Pentest (teste de invasão), apresentando suas metodologias, finalidades e a crescente incorporação de técnicas de inteligência artificial para automação de atividades de reconhecimento e exploração. Esse embasamento permitirá uma compreensão sólida dos experimentos realizados e das estratégias de mitigação propostas.

2.1. OWASP Top 10

A Open Web Application Security Project (OWASP) é uma comunidade global sem fins lucrativos dedicada a melhorar a segurança de software. Uma de suas iniciativas mais reconhecidas é o OWASP Top 10, um documento que lista e descreve os dez riscos de segurança mais críticos para aplicações web. Este ranking é atualizado a cada 4 anos com base em dados de vulnerabilidades, coletados de milhares de aplicações, compartilhados por profissionais e especialistas, servindo como um guia essencial para desenvolvedores e profissionais de segurança (OWASP, 2021).

O OWASP Top 10 não é uma lista de todas as vulnerabilidades existentes, mas sim um consenso sobre os riscos mais notáveis e impactantes que as empresas devem priorizar em seus esforços de segurança. Ele visa aumentar a conscientização sobre os riscos de segurança de aplicações e dar orientações para uma redução de possíveis falhas em sistemas. O entendimento e aplicação dos conceitos do OWASP Top 10 são cruciais para o desenvolvimento de aplicações robustas e seguras.

A lista do TOP 10 é atualizada a cada 4 anos e geralmente ocorre no final do ano do ciclo. O presente trabalho será baseado na lista de 2021. Abaixo é possível identificar o que mudou em comparação a lista de 2017, após reunir os dados fornecidos durante os 4 anos.

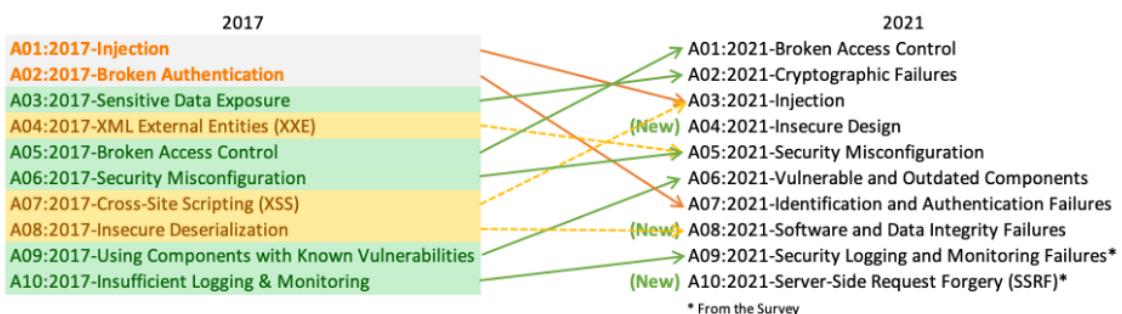


Figura 1. OWASP Top 10 2021

A proxima atualização (release) acontecerá no final de 2025 e possui uma pré release disponivel. Como é possível identificar na imagem, a lista em sua maioria manteve grande parte das vulnerabilidades em comparação a 2021, com alterações no nome para englobar mais vulnerabilidades de mesma categoria no mesmo item. Falhas criptográficas (Cryptographic Failures), Injeção (Injection) e design inseguro (Insecure Design) perdem

2 posições para Configurações Incorretas de segurança (Security Misconfiguration) e Falhas na Cadeia de Suprimentos de Software (Software Supply Chain Failures), que era previamente o "Vulnerable and outdated components";

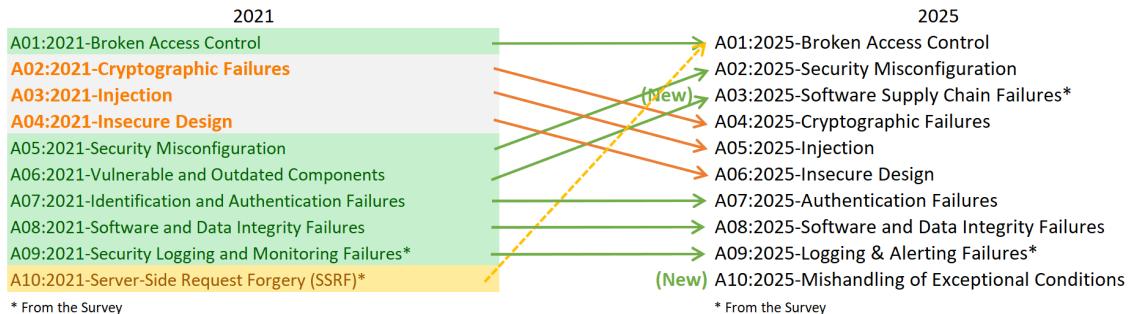


Figura 2. OWASP Top 10 2025 pré-release

2.2. Segurança de APIs e aplicações Web

Com a arquitetura de microsserviços e a crescente adoção de APIs (Application Programming Interfaces) para comunicação entre sistemas, a segurança dessas interfaces tornou-se uma preocupação crítica. As APIs são frequentemente a porta de entrada para dados e funcionalidades de backend, tornando-as alvos atraentes para atacantes . O OWASP API Security Top 10 é uma iniciativa específica que foca nos riscos de segurança mais críticos para APIs, complementando o OWASP Top 10 tradicional .

Segundo Leite (2022) "Uma aplicação web é um tipo de software projetado para operar mediante o acesso e a interação por meio de um navegador web" as quais, caso desenvolvidas negligenciando boas práticas de segurança, tornam-se alvos em potencial para invasores, tendo em vista que toda grande corporação possui algum tipo de sistema que está conectado a rede mundial de computadores. As vulnerabilidades em APIs e sistemas web podem incluir quebras de autenticação e autorização, exposição excessiva de dados, injeção, e configurações de segurança inadequadas. A análise de maneiras seguras no desenvolvimento de APIs, de acordo com as diretrizes do OWASP API Security Top 10, é fundamental para proteger os sistemas modernos . Isso envolve a implementação de autenticação e autorização robustas, validação de entrada e saída, gerenciamento de erros seguro e monitoramento contínuo.

2.3. Vulnerabilidades mais Comuns em Aplicações Web

As vulnerabilidades listadas no OWASP Top 10 abrangem uma infinidade de falhas de segurança que podem ser exploradas por atacantes. Dentre as mais persistentes e perigosas, segundo OWASP (2021) destacam-se:

2.3.1 Quebra de Controle de Acesso (Broken Access Control)

As falhas de controle de acesso ocorrem quando as restrições sobre o que usuários autenticados podem fazer não são aplicadas corretamente. Atacantes podem explorar essas falhas para acessar funcionalidades não autorizadas, visualizar ou modificar dados de outros usuários, ou alterar privilégios , (Bach-Nutman, 2020). A implementação de um controle de acesso robusto e a verificação de permissões em cada requisição são cruciais para prevenir essas vulnerabilidades .

2.3.2 Exposição de dados sensíveis (Cryptographic Failures)

As exposições de dados sensíveis (Cryptographic Failures) acontecem no momento em que, segundo Joseph (2024), dados são enviados/recebidos sem serem criptografados assim levando dados sensíveis a serem expostos. A utilização do protocolo HTTPS, por exemplo, é imprescindível para que dados possam transitar na web de forma segura, permitindo que dados sejam criptografados e que apenas quem de fato deveria ter acesso aos dados, possa acessá-los.

2.3.3 Injeção (Injection)

As falhas de injeção, como SQL Injection, ocorrem quando dados não confiáveis são enviados a um interpretador como parte de um comando ou consulta. Os dados maliciosos do atacante podem enganar o interpretador para executar comandos não intencionais ou acessar dados sem autorização . Este tipo de ataque pode levar à divulgação completa de dados, modificação ou exclusão de informações, e até mesmo ao controle total do servidor . No passado, o SQL injection costumava ser o tipo de falha de injeção mais comum e segundo (OWASP, 2021) continua dentro o grupo falhas de injeção mais comuns. Segundo Souza e Pereira (2021) ataques de injeção do tipo XSS e CSRF são os tipos mais populares de ataques de injeção de código script atualmente, dado a popularidade destas tecnologias em aplicações web modernas. A prevenção envolve o uso de consultas parametrizadas, procedimentos armazenados e validação rigorosa de entrada através de técnicas de sanitização de dados.

2.3.4 Quebra de Autenticação e Gerenciamento de Sessão (Broken Authentication)

Broken Authentication é o termo utilizado para relatar o problema de quebra de credenciais de usuário. O problema pode acontecer de diversas maneiras, desde senhas fracas do usuário que facilitam métodos de força bruta, senhas armazenadas no banco de dados de maneira incorreta, sem técnicas de hashing por exemplo, (Bach-Nutman, 2020).

2.3.5 Configuração Incorreta de Segurança (Security Misconfiguration)

Configurações incorretas de segurança, acontecem quando alguma etapa na publicação de uma aplicação web é negligenciada. São diversas as possibilidades de má configuração: portas desnecessariamente abertas, frameworks desatualizados, falta de autenticação/autorização, entre muitos outros. Maneiras de prevenção incluem rotina rigorosa de testes com configurações idênticas em todos ambientes, com exceção das credenciais correspondentes a cada ambiente.

2.3.6 Componentes Vulneráveis e Desatualizados

Componentes vulneráveis e desatualizados representam uma das causas mais recorrentes de ataques em aplicações modernas. Segundo OWASP (2021), essas falhas ocorrem quando bibliotecas, frameworks, servidores de aplicações ou módulos de terceiros utilizados pelo sistema estão em versões antigas, com vulnerabilidades já conhecidas e catalogadas. É comum que equipes de desenvolvimento foquem apenas na lógica da aplicação e deixem de atualizar dependências externas, o que abre espaço para exploração por parte de atacantes que buscam precisamente componentes com CVEs conhecidos. A prevenção envolve manter inventários atualizados, aplicar ferramentas de análise de composição de software (SCA), como o OWASP Dependency-Check, e garantir que todos os

artefatos sejam obtidos de repositórios confiáveis e com verificação de integridade digital.

2.3.7 Falhas de Identificação e Autenticação

As falhas de identificação e autenticação (Identification and Authentication Failures) englobam problemas relacionados ao processo de autenticar usuários e gerenciar sessões de forma segura. Segundo OWASP (2021), tais vulnerabilidades incluem senhas fracas, ausência de autenticação multifator, IDs de sessão previsíveis, ausência de limitação de tentativas de login e recuperação de senha insegura. Estudos brasileiros, como o trabalho de Silva e Borges (2022), demonstram que a maior parte dos incidentes relacionados a este tipo de falha ocorre devido a implementações incorretas de mecanismos básicos de autenticação. Para mitigação, recomenda-se a adoção de multifator, hashes seguros para senhas, renovação de sessão após login e registro detalhado de tentativas mal-sucedidas de acesso.

2.3.8 Falhas de Integridade de Software e Dados

Falhas de integridade de software e dados (Software and Data Integrity Failures) surgem quando aplicações não verificam adequadamente a autenticidade ou confiabilidade de dados, atualizações ou componentes carregados dinamicamente. Conforme explica OWASP (2021), uma classe comum desse tipo de falha são processos de CI/CD vulneráveis, que podem permitir a inserção de código malicioso durante o pipeline de deploy. Outro exemplo amplamente documentado é a desserialização insegura, onde objetos recebidos de fontes externas podem ser manipulados para execução arbitrária de código. Pesquisas nacionais como as de Ferreira e Ramos (2024) apontam que grande parte das organizações negligencia controles de integridade nas etapas de automação, tornando seus sistemas suscetíveis a ataques supply chain. Entre as medidas preventivas incluem-se a verificação de assinatura digital de artefatos, controle rigoroso de acesso ao pipeline e limitação do carregamento de objetos serializados.

2.3.9 Falhas de Log e Monitoramento de Segurança

Falhas de log e monitoramento (Security Logging and Monitoring Failures) se referem à incapacidade de registrar, detectar ou responder adequadamente a eventos suspeitos ou maliciosos. Conforme destacado pela OWASP (2021), sistemas que não registram informações essenciais como falhas de autenticação, alterações de privilégio ou eventos de acesso a recursos sensíveis tornam-se extremamente difíceis de auditar após incidentes, além de não conseguirem detectar ataques em andamento. Trabalhos como o de Ferreira e Ramos (2024) reforçam que, sem monitoramento estruturado, é praticamente impossível realizar análises forenses ou identificar padrões de ataque em tempo hábil. Boas práticas incluem geração de logs estruturados, uso de plataformas de correlação como SIEM, retenção segura de registros e implementação de alertas automáticos para eventos críticos.

2.3.10 Falsificação de Requisição no Lado do Servidor (SSRF)

A vulnerabilidade de Server-Side Request Forgery (SSRF) ocorre quando um atacante consegue induzir o servidor a enviar requisições HTTP para destinos internos ou externos não autorizados. Segundo OWASP (2021), esse tipo de ataque é especialmente crítico em ambientes de nuvem, nos quais metadados de instâncias podem ser expostos se o servidor for induzido a acessá-los inadvertidamente. Aplicações que permitem ao usu-

ário fornecer URLs arbitrárias, como webhooks, integrações e serviços de importação de dados, tornam-se particularmente suscetíveis a SSRF. Para mitigação, recomenda-se a validação rigorosa de URLs fornecidas pelo usuário, uso de listas de permissão (whitelists), restrição de tráfego de saída do servidor por meio de firewalls e proibição de protocolos perigosos, como file:// e access a IPs internos.

2.4. Pentest

O teste de intrusão (pentest) é uma prática fundamental na segurança da informação, pois permite identificar vulnerabilidades exploráveis antes que agentes maliciosos o façam. Estudos recentes mostram que, apesar da evolução das metodologias, ainda existem desafios significativos na padronização, automação e avaliação dos resultados de pentests (Bertoglio; Zorzo, 2017). Pesquisas também têm avançado no uso de inteligência artificial, especialmente reinforcement learning (RL), para automatizar etapas do processo de ataque, simulando ambientes como Capture-the-Flag (CTF) e aplicações web vulneráveis (Zennaro; Erdődi, 2023). Frameworks de automação, como o HARMer, demonstram que a modelagem hierárquica de ataques pode aumentar a eficácia e permitir avaliações mais precisas de risco (Enoch *et al.*, 2020). Essas abordagens mostram que o campo de pentesting está evoluindo rapidamente, integrando técnicas tradicionais com ferramentas baseadas em IA, o que amplia tanto o potencial ofensivo quanto a complexidade das defesas necessárias.

3. Trabalhos Relacionados

Neste item serão apresentados os principais trabalhos que possuem uma relação com o tema definido neste estudo. Os 5 trabalhos selecionados abrangem a maioria dos assuntos chave para a evolução da pesquisa. Os estudos abaixo citam boas práticas, preocupações, e métodos adequados para prevenir possíveis vulnerabilidades em aplicações Web, assim como explicações sobre as vulnerabilidades mais comuns.

- **Vulnerabilidade em Aplicações Web (Almeida, 2021)**

Com o avanço da internet e o aumento das transações online, a segurança da informação tornou-se um tema essencial. As aplicações web, por estarem sempre conectadas, tornam-se alvos frequentes de invasões que exploram falhas como autenticação quebrada, injeções de código e erros de configuração.

O estudo de Heitor Almeida teve como objetivo compreender o funcionamento dessas vulnerabilidades e propor formas práticas de mitigá-las. O autor buscou base teórica no OWASP Top 10, analisando as principais falhas que afetam servidores web e os riscos que representam tanto para usuários quanto para empresas. No desenvolvimento, o trabalho apresentou exemplos práticos e didáticos de códigos que exploram e corrigem vulnerabilidades, como SQL Injection, Broken Access Control e Falhas Criptográficas. Foram demonstradas técnicas de ataque e, em seguida, formas de defesa, como o uso de consultas parametrizadas e criptografia adequada para senhas.

Como conclusão, o estudo reforçou a importância da adoção de boas práticas de desenvolvimento seguro e o uso contínuo das diretrizes do OWASP para prevenir ataques. O autor destaca que entender o funcionamento das falhas é essencial para desenvolver sistemas realmente protegidos e garantir a segurança dos dados de usuários e organizações.

- **Análise de Vulnerabilidades em Aplicações Web: Um Estudo de Caso Utilizando o OWASP ZAP (Sampaio, 2021)**

A segurança de aplicações web é um dos maiores desafios da atualidade, especialmente com o crescimento de ataques cibernéticos que exploram falhas em sistemas mal configurados. Diante desse cenário, o trabalho de Fernando Sampaio propõe analisar vulnerabilidades em aplicações web reais utilizando ferramentas de código aberto e metodologias consagradas.

O estudo teve como objetivo identificar e classificar vulnerabilidades presentes em um sistema web por meio da ferramenta OWASP ZAP, adotando como referência o OWASP Top 10. A pesquisa buscou demonstrar como falhas simples de configuração ou validação de entrada podem expor dados sensíveis e comprometer todo o sistema.

Durante o desenvolvimento, o autor realizou testes práticos em uma aplicação web, aplicando técnicas de varredura automatizada e inspeção manual. As vulnerabilidades encontradas foram analisadas quanto à gravidade e ao impacto, e foram sugeridas medidas corretivas como sanitização de entradas, uso de HTTPS, e gerenciamento adequado de sessões.

O trabalho conclui que o uso de ferramentas como o OWASP ZAP é fundamental para a detecção precoce de falhas, permitindo que desenvolvedores corrijam problemas antes que sejam explorados por atacantes. Sampaio reforça a necessidade de incorporar práticas de teste de segurança contínuo no ciclo de desenvolvimento de software.

- **Análise de Vulnerabilidades sobre a Aplicação Web do CINTE-RN: Um Estudo de Caso (Duarte, 2020)**

Com o aumento constante de ataques cibernéticos, a segurança de aplicações web se tornou uma preocupação crítica, especialmente porque muitas falhas surgem quando a segurança não é uma prioridade durante o desenvolvimento do software. Este estudo se propôs a investigar, na prática, as vulnerabilidades presentes no site do Centro de Inovações Tecnológicas do Rio Grande do Norte (CINTE-RN). O objetivo era não apenas apontar falhas, mas também refletir sobre a importância de se adotar uma cultura de segurança desde a codificação até a hospedagem de uma aplicação.

Para isso, os pesquisadores simularam a abordagem de um invasor externo, utilizando ferramentas especializadas como o NMAP e o Nessus para analisar o site e seu servidor sem ter nenhum conhecimento prévio da infraestrutura, uma metodologia conhecida como teste de "caixa preta".

Os resultados revelaram 216 possíveis vulnerabilidades. A boa notícia é que a grande maioria das falhas mais sérias poderia ter sido evitada com medidas relativamente simples, como manter os sistemas atualizados, aplicar correções de segurança e configurar o servidor para não divulgar informações que facilitem um ataque. O trabalho conclui que a segurança não é uma etapa complexa e inacessível, mas sim uma soma de boas práticas e atenção contínua que podem prevenir a maioria dos problemas.

- **Understanding The Top 10 OWASP Vulnerabilities (Bach-Nutman, 2020)**

A segurança de aplicações web é um tema urgente, já que falhas durante o desenvolvimento podem expor dados sensíveis, interromper serviços e causar sérios

prejuízos financeiros e reputacionais. Com o aumento de ataques cibernéticos, é fundamental que empresas e desenvolvedores compreendam as vulnerabilidades mais comuns para se protegerem de forma eficaz.

Este trabalho buscou explicar e detalhar as 10 principais vulnerabilidades listadas pelo OWASP (Open Web Application Security Project), mostrando como elas podem ser exploradas e, principalmente, como podem ser mitigadas. O objetivo era oferecer uma visão clara e prática para ajudar desenvolvedores e organizações a protegerem suas aplicações.

O estudo foi baseado em uma revisão aprofundada de cada uma das vulnerabilidades do Top 10 do OWASP, como injeção de SQL, quebra de autenticação, exposição de dados sensíveis e configurações inseguras. Para cada item, foram apresentados exemplos de como os ataques ocorrem e medidas práticas de prevenção, como o uso de prepared statements, validação de entradas e a implementação de logging e monitoramento.

A pesquisa concluiu que a maioria das vulnerabilidades pode ser evitada com práticas de desenvolvimento seguro, atualizações regulares e configurações adequadas. Além disso, destacou que ataques bem-sucedidos não só causam perdas financeiras diretas, mas também abalam a confiança dos clientes. Investir em segurança desde o início do desenvolvimento é essencial para proteger tanto os negócios quanto os usuários finais.

- **Segurança Em Aplicações WEB: Uma abordagem prática para identificar vulnerabilidades (Santos, 2023)**

A segurança em aplicações web é um tema de extrema importância na atualidade, mas ainda é negligenciada em muitos ambientes acadêmicos e profissionais, especialmente em empresas de pequeno e médio porte. A falta de conhecimento técnico e a priorização de funcionalidades em detrimento da segurança tornam os sistemas vulneráveis a ataques, o que pode resultar em prejuízos financeiros, danos à reputação e problemas legais.

Este trabalho teve como objetivo principal demonstrar, de forma prática e acessível, o processo de análise de segurança em uma aplicação web. Para isso, foi desenvolvido e analisado o sistema RentX – uma plataforma de aluguel de carros – com o intuito de identificar vulnerabilidades e propor melhorias que garantam um nível mínimo de segurança, servindo como exemplo para desenvolvedores e empresas.

Foi desenvolvida uma aplicação web completa, o RentX, utilizando tecnologias modernas como React, Node.js, Express e PostgreSQL. Em seguida, realizou-se um teste de penetração utilizando a ferramenta Zed Attack Proxy (ZAP), que permitiu uma análise dinâmica da aplicação em busca de vulnerabilidades com base no OWASP Top 10.

A análise revelou a existência de vulnerabilidades de baixo e médio risco, como a ausência de cabeçalhos de segurança (CSP, X-Content-Type-Options), bibliotecas desatualizadas e configurações inadequadas. Concluiu-se que a maioria das falhas poderia ser corrigida com ajustes simples de configuração e atualizações, reforçando que a segurança deve ser integrada desde as fases iniciais do desenvolvimento. O trabalho mostrou que mesmo aplicações bem estruturadas podem conter brechas significativas, destacando a necessidade de testes de segurança contínuos.

A escolha dos 5 trabalhos acima se dá pela a correlação entre as pesquisas e o tema abordado neste artigo, onde cada um dos tópicos cita pontos importantes para a presente pesquisa, como segurança, pentest, autenticação, tipos de vulnerabilidades e como as prevenir, etc...

4. Materiais e Métodos

Nesta seção serão estabelecidos os métodos adotados para a validação científica desta pesquisa, bem como as ferramentas utilizadas para identificação, exploração e mitigação das vulnerabilidades encontradas, tendo como principal referencial o *OWASP Top 10*.

4.1. Máquinas virtuais disponíveis na WEB

Para a validação e análise das vulnerabilidades estudadas, serão utilizadas máquinas virtuais e aplicações deliberadamente vulneráveis, obtidas em repositórios públicos (GitHub) para fins educacionais.

Visando simular um ambiente próximo ao encontrado em cenários reais, será empregada uma rede local composta por duas aplicações vulneráveis executadas em um único equipamento (servidor/notebook) e atacadas a partir de outro dispositivo na mesma rede. A abordagem do pentest adotada combina dois modos: *gray-box* para uma das aplicações e *black-box* para a outra, permitindo avaliar diferenças na detecção e exploração de falhas sob distintos níveis de informação.

4.2. Cenário Black Box: Metasploitable 3

Para simular um teste de penetração do tipo **Black Box**, onde o analista não possui conhecimento prévio da infraestrutura interna, foi utilizado o Metasploitable 3. Este ambiente é uma máquina virtual intencionalmente vulnerável, baseada em Windows Server, que expõe uma variedade de serviços e aplicações, incluindo tecnologias do ecossistema Microsoft.

- **Justificativa da Escolha:** A escolha do Metasploitable 3 se deu por sua natureza como um "alvo desconhecido". Ele simula um servidor corporativo real, permitindo que a análise siga as fases de um pentest ético (reconhecimento, varredura, exploração) sem qualquer informação privilegiada.
- **Abordagem (Black Box):** O teste foi iniciado conhecendo apenas o endereço IP da máquina. A partir daí, todas as informações sobre serviços, tecnologias e vulnerabilidades foram obtidas através das ferramentas de análise, replicando a perspectiva de um atacante externo.

4.3. Cenário Grey Box: OWASP Juice Shop

Para a simulação de um teste do tipo Grey Box, foi escolhido o OWASP Juice Shop (OWASP, 2024). Trata-se de uma aplicação web moderna e complexa (escrita em Node.js e Angular), projetada pela OWASP para ser um campo de treinamento para segurança de aplicações.

- **Justificativa da Escolha:** O Juice Shop é ideal para uma análise Grey Box, pois seu código-fonte é aberto e acessível. Isso permite que o analista tenha acesso a informações parciais, como a lógica de negócios e a estrutura do código, para guiar os testes de segurança, um cenário comum quando se audita uma aplicação da própria empresa.

- **Abordagem (Grey Box):** A análise foi conduzida com acesso total ao repositório do código-fonte no GitHub. O código foi consultado para entender como certas funcionalidades foram implementadas, identificar potenciais falhas lógicas e construir ataques mais direcionados. Por exemplo, ao analisar o código de autenticação, foi possível formular hipóteses sobre como explorar a vulnerabilidade de "Quebra de Autenticação".

4.4. Ferramentas e Instrumentos de Análise

A seleção de ferramentas buscou cobrir as diferentes fases de um teste de invasão, sendo aplicadas em ambos os cenários conforme a necessidade, assim como ferramentas para o setup de infraestrutura requerida para a exposição das aplicações:

- **Nmap (Nmap, 2025):** Utilizado na fase de reconhecimento mapeamento de portas e serviços.
- **OWASP ZAP (OWASP, 2025):** Empregado como proxy de interceptação para analisar o tráfego HTTP/S em ambos os ambientes, permitindo a manipulação de requisições para testar falhas de injeção, controle de acesso e outras.
- **Ferramentas Específicas:** Conforme a vulnerabilidade, ferramentas como SQL Map (SQL..., 2025) (para SQL Injection) e Hydra (HYDRA..., 2025) (para ataques de força bruta) foram utilizadas para automatizar e validar a exploração.
- **Análise de Código-Fonte:** O Visual Studio (VISUAL..., 2025) Code foi utilizado para navegar e inspecionar o código-fonte do Juice Shop, sendo uma ferramenta essencial para a abordagem Grey Box.
- **Docker (Docker, 2025):** Para a configuração das VMs e exposição das aplicações, serão utilizados containeres Docker para facilitar o processo de *deploy*, onde toda a aplicação será *containerizada*.

Com base nos Materiais descritos previamente nesta seção, a análise será executada visando ter duas perspectivas de análise das vulnerabilidades, uma onde o pentester não tem informações prévias sobre o alvo (Black box) e outra onde as informações serão limitadas (Grey box). As escolhas de infraestrutura foram realizadas com o objetivo de simular cenários semelhantes aos encontrados em ambientes produtivos. Essa configuração permite reproduzir de forma realista o comportamento das aplicações, enquanto as ferramentas selecionadas oferecem suporte às etapas de análise, exploração e validação das vulnerabilidades.

5. Resultados e Discussão

Essa seção deverá ser escrita na segunda parte do trabalho, conhecida como TCC2, e deverá conter os resultados dos experimentos realizados, discussão comparando os resultados obtidos com outros encontrados em trabalhos similares, além de um parágrafo apontando as limitações da metodologia adotada.

6. Considerações Finais

Essa seção deverá ser escrita na segunda parte do trabalho, conhecida como TCC2.

Referências

- ALMEIDA, H. S. de. **Vulnerabilidade em Aplicações Web**. 2021. Faculdade Pitágoras de Sobral. Disponível em: <https://repositorio.pgsscogna.com.br/bitstream/123456789/40152/1/HEITOR%5C%20SOUSA%5C%20DE%5C%20ALMEIDA.PDF>.
- BACH-NUTMAN, M. Understanding The Top 10 OWASP Vulnerabilities. **arXiv preprint**, arXiv:2012.09960, 2020. Disponível em: <https://arxiv.org/abs/2012.09960>.
- BERTOGLIO, D. D.; ZORZO, A. F. Overview and open issues on penetration test. **Journal of the Brazilian Computer Society**, v. 23, n. 1, 2017. Disponível em: <https://journal-bcs.springeropen.com/articles/10.1186/s13173-017-0051-1>.
- DOCKER. **Docker**. Acesso em: 01 nov. 2025. 2025. Disponível em: <https://www.docker.com/>.
- DUARTE, G. C. **Análise de Vulnerabilidades sobre a Aplicação Web do CINTE-RN: Um Estudo de Caso**. 2020. Instituto Federal do Rio Grande do Norte. Disponível em: <https://memoria.ifrn.edu.br/bitstream/handle/1044/2291/TCC%5C%20Final-%5C%20GILENO%5C%20CORDEIRO%5C%20DUARTE.pdf>.
- ENOCH, S. Y. *et al.* HARMer: Cyber-attacks Automation and Evaluation. **IEEE Access**, v. 8, p. 113773–113787, 2020. Disponível em: <https://arxiv.org/abs/2006.14352>.
- FERREIRA, C.; RAMOS, D. Avaliação de Vulnerabilidades de Integridade e Monitoramento de Segurança. **Revista da Universidade de Brasília de Segurança Cibernética**, 2024. Disponível em: https://bdm.unb.br/bitstream/10483/40115/1/2024_DafneMoreira_LucasAndrade_tcc.pdf.
- HYDRA. [S. l.: s. n.], 2025. Acesso em: 01 nov. 2025. Disponível em: <https://www.kali.org/tools/hydra/>.
- JOSEPH, J. **OWASP Top 10 Vulnerabilities in .NET Core 7+ (and How to Beat Them)**. 2024. Disponível em: <https://james-joseph.medium.com/owasp-top-10-vulnerabilities-in-net-core-7-and-how-to-beat-them-2f27c38fb60b>.
- LEITE, M. V. C. **Segurança em Aplicações Web: Análise de Vulnerabilidades e Testes de Penetração**. 2022. Universidade Federal do Maranhão. Disponível em: https://rosario.ufma.br/jspui/bitstream/123456789/7552/1/Marcos_Vinicius_Correia_Leite.pdf.
- NMAP. **Nmap**. [S. l.: s. n.], 2025. Acesso em: 01 nov. 2025. Disponível em: <https://nmap.org/>.
- OWASP. **OWASP Juice Shop**. [S. l.: s. n.], 2024. Acesso em: 01 nov. 2025. Disponível em: <https://owasp.org/www-project-juice-shop/>.

OWASP. **OWASP TOP 10**. [S. l.: s. n.], 2021. Acesso em: 01 nov. 2025. Disponível em: <https://owasp.org/Top10>.

OWASP. **OWASP ZAP**. [S. l.: s. n.], 2025. Acesso em: 01 nov. 2025. Disponível em: <https://owasp.org/www-chapter-dorset/assets/presentations/2020-01/20200120-OWASPDorset-ZAP-DanielW.pdf>.

SAMPAIO, F. F. Uma Análise Prática das Principais Vulnerabilidades em Aplicações Web Baseado no Top 10. 2021. Universidade Federal do Ceará. Disponível em: https://repositorio.ufc.br/bitstream/riufc/62466/1/2021_tcc_ffsampaio.pdf.

SANTOS, M. G. dos. Segurança em Aplicações Web: Uma Abordagem Prática para Identificar Vulnerabilidades. João Pessoa, PB, 2023. Trabalho de Conclusão de Curso. Disponível em: <https://repositorio.ufpb.br/jspui/handle/123456789/31664>.

SILVA, M.; BORGES, A. Falhas de Autenticação em Aplicações Web: Análise de Vulnerabilidades Comuns. **Revista Científica do IF Goiano**, 2022. Disponível em: https://repositorio.ifgoiano.edu.br/bitstream/prefix/3850/3/tcc_Livia%20Ester%20Felipusso%20Timoteo.pdf.

SOUZA, A.; PEREIRA, M. A [in]Segurança dos Sistemas Governamentais Brasileiros: Um Estudo de Caso em Sistemas Web e Redes Abertas. **arXiv preprint**, arXiv:2109.06068, 2021. Disponível em: <https://arxiv.org/pdf/2109.06068>.

SQL Map. [S. l.: s. n.], 2025. Acesso em: 01 nov. 2025. Disponível em: <https://sqlmap.org/>.

VISUAL Studio Code. [S. l.: s. n.], 2025. Acesso em: 01 nov. 2025. Disponível em: <https://code.visualstudio.com/>.

ZENNARO, F. M.; ERDÓDI, L. Modelling Penetration Testing with Reinforcement Learning Using Capture-the-Flag Challenges: Trade-offs Between Model-free Learning and A Priori Knowledge. **IET Information Security**, 2023. Disponível em: <https://arxiv.org/abs/2005.12632>.