FACULDADES INTEGRADAS DE BAURU

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MURILO STORTI IRANI
PEDRO AUGUSTO DE OLIVEIRA

SCANNER PARA TESTE DE PENETRAÇÃO OWASP INTEGRANDO CHAOS ENGINEERING: FORTALECENDO A SEGURANÇA E RESILIÊNCIA DE APLICAÇÕES WEB

BAURU - SP

2024

MURILO STORTI IRANI

PEDRO AUGUSTO DE OLIVEIRA

SCANNER PARA TESTE DE PENETRAÇÃO OWASP INTEGRANDO CHAOS ENGINEERING: FORTALECENDO A SEGURANÇA E RESILIÊNCIA DE APLICAÇÕES WEB

Trabalho de Conclusão de Curso apresentado para obtenção do título de graduação em Ciência da Computação das Faculdades Integradas de Bauru – FIB.

Orientador: Prof. Me. Ronaldo César Dametto

Co-orientador: Prof. Me. Leandro Luis Pauro

BAURU - SP

2024

MURILO STORTI IRANI

PEDRO AUGUSTO DE OLIVEIRA

SCANNER PARA TESTE DE PENETRAÇÃO OWASP INTEGRANDO CHAOS ENGINEERING: FORTALECENDO A SEGURANÇA E RESILIÊNCIA DE APLICAÇÕES WEB

Trabalho de Conclusão de Curso apresentado para obtenção do título de graduação em Ciência da Computação das Faculdades Integradas de Bauru – FIB.

Orientador: Prof. Me. Ronaldo César Dametto

Co-orientador: Prof. Me. Leandro Luís Pauro

BANCA EXAMINADORA:

Presidente/Orientador: Prof. Me. Ronaldo César Dametto

Instituição: Faculdades Integradas de Bauru

Prof.1:

Instituição: Faculdades Integradas de Bauru

Prof.2:

Instituição: Faculdades Integradas de Bauru

IRANI, Murilo Storti; OLIVEIRA, Pedro Augusto. **Scanner Para Teste De Penetração Owasp Integrando Chaos Engineering: Fortalecendo A Segurança E Resiliência De Aplicações Web**. 2024. 50f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação). FIB. Bauru, 2024.

RESUMO

No cenário atual, onde a digitalização é ubíqua e a conectividade é a espinha dorsal de nossa sociedade globalizada, a segurança digital é uma preocupação fundamental. Com a proliferação de aplicativos web e a crescente complexidade das infraestruturas de TI, as vulnerabilidades tornaram-se uma porta de entrada para ameaças cibernéticas. Este trabalho propõe o desenvolvimento de um Scanner de Segurança para Aplicações Web, com foco na detecção de vulnerabilidades conforme o modelo OWASP. A Organização de Segurança de Aplicações Web (OWASP) estabeleceu padrões amplamente aceitos para identificar e mitigar vulnerabilidades em sistemas online. Utilizando esses padrões como base, nosso estudo integra conceitos de *chaos engineering* nos testes de intrusão, permitindo simular cenários de falhas e instabilidade em ambientes controlados. Diferente de ferramentas voltadas exclusivamente para grandes organizações, nosso scanner também visa democratizar o acesso à segurança digital, possibilitando que pequenas empresas e usuários finais acessem o sistema de forma online e simplificada. Ao destacar vulnerabilidades comuns e fornecer recomendações para mitigá-las, pretendemos fortalecer a postura de segurança de qualquer usuário, independentemente de seu porte ou conhecimento técnico, na era digital. Ao final, buscamos não apenas apresentar uma solução técnica para a detecção de vulnerabilidades, mas também promover uma compreensão mais profunda das ameaças enfrentadas pelas organizações e indivíduos. Ao abordar a importância da segurança cibernética em um mundo cada vez mais interconectado, esperamos contribuir para a criação de ambientes online mais seguros e resilientes.

Palavras-chave: *Segurança*, *Pentest*, *OWASP, Chaos Engineering, Acessibilidade*

IRANI, Murilo Storti; OLIVEIRA, Pedro Augusto. **Penetration Testing Scanner OWASP Integrating Chaos Engineering: Strengthening Web Application Security and Resilience.** 2024. 50p. Bachelor's Thesis (Computer Science). FIB. Bauru, 2024.

ABSTRACT

In the current scenario, where digitization is ubiquitous and connectivity is the backbone of our globalized society, digital security has become a fundamental concern. With the proliferation of web applications and the increasing complexity of IT infrastructures, vulnerabilities have become a gateway for cyber threats. This work proposes the development of a Web Application Security Scanner, focusing on vulnerability detection according to the OWASP model, in simulated environments. The Open Web Application Security Project (OWASP) has established widely accepted standards for identifying and mitigating vulnerabilities in online systems. Using these standards as a basis, our study integrates chaos engineering concepts into penetration testing, allowing for the simulation of failure and instability scenarios in controlled environments. Unlike tools aimed exclusively at large organizations, our scanner also aims to democratize access to digital security, enabling small businesses and end users to access the system online in a simplified way. By highlighting common vulnerabilities and providing recommendations to mitigate them, we aim to strengthen the security posture of any user, regardless of their size or technical knowledge, in the digital age. In conclusion, we seek not only to present a technical solution for vulnerability detection but also to promote a deeper understanding of the threats faced by organizations and individuals. By addressing the importance of cybersecurity in an increasingly interconnected world, we hope to contribute to the creation of safer and more resilient online environments.

Keywords: *Cybersecurity*, *Pentest*, OWASP*, Chaos Engineerin, Accessibility*

LISTA DE ABREVIATURAS E SIGLAS

API.....Application Programming Interface

CISO.....Chief Information Security Officer

CVE.....Common Vulnerabilities and Exposures

DDoS.....Distributed Denial of Service

DNS.....Domain Name System

ICS.....Industrial Control System

IP.....Internet Protocol

LGPD.....Lei Geral de Proteção de Dados

OT.....Operational Technology

OWASP.....Open Web Application Security Project

Pentest.....Penetration Testing

SSL.....Secure Sockets Layer

TIC.....Tecnologias de Informação e Comunicação

TLS.....Transport Layer Security

TLD.....Top-Level Domain

ToS.....Terms of Service

WSTG.....Web Security Testing Guide

LISTA DE IMAGENS

Lista de Tabelas

SUMÁRIO

1. Introdução

A era da globalização trouxe consigo uma transformação radical no panorama tecnológico, impulsionando o desenvolvimento exponencial de redes e sistemas de comunicação. Essa revolução digital trouxe inúmeras vantagens, facilitando a interconexão entre pessoas e empresas em todo o mundo, impulsionando o surgimento de novos modelos de negócios e inovações tecnológicas. No entanto, junto com esses avanços, emergiram desafios significativos relacionados à segurança cibernética.

Em uma das declarações apresentadas no documentário *A globalização vista do lado de cá*, Milton Santos (2006) manifestou sua insatisfação com a direção que a ciência e a tecnologia pareciam ter seguido: *"Nunca na história da humanidade houve condições técnicas e científicas tão adequadas a construir um mundo da dignidade humana, apenas essas condições foram expropriadas por um punhado de empresas que decidiram construir um mundo perverso. Cabe a nós fazer dessas condições materiais, a condição material da produção de uma outra política",*

Essa reflexão ressalta como a tecnologia, embora ofereça oportunidades sem precedentes, pode ser manipulada por interesses comerciais para criar um ambiente digitalmente perigoso e hostil.

Nesse contexto, a engenharia do caos emerge como uma abordagem inovadora para fortalecer a resiliência dos sistemas digitais. Originada no Vale do Silício (conceito criado por engenheiros da Netflix), essa prática busca proativamente identificar e corrigir fragilidades nos sistemas por meio da introdução controlada de falhas (Basiri *et al*., 2019).

Através dessa abordagem, os engenheiros do caos podem simular cenários de estresse e avaliar como os sistemas respondem a essas condições adversas. Essa metodologia, combinada com os princípios do *Pentest* (Teste de Penetração), permite uma avaliação mais abrangente da segurança cibernética, identificando e corrigindo vulnerabilidades antes que sejam exploradas por agentes maliciosos.

O advento das Tecnologias da Informação e Comunicação (TICs) permitiu o surgimento de protocolos de rede, como o TCP/IP, que servem como espinha dorsal da comunicação digital global. Esses protocolos, embora essenciais para a conectividade mundial, também introduziram novas vulnerabilidades que podem ser

exploradas por indivíduos mal-intencionados. A proliferação de ameaças cibernéticas, como *malware*, *phishing* e ataques de negação de serviço (DDoS), evidencia a necessidade premente de medidas eficazes de segurança cibernética.

A mescla de múltiplas abordagens analíticas pode substancialmente elevar a acurácia na detecção de ataques, gerando métodos mais eficientes para identificar irregularidades com maior precisão (Liang, 2021).

A proteção de dados tornou-se uma preocupação central em um mundo onde a informação é um ativo valioso. Entre janeiro de 2023 e junho de 2024, foram registrados 108 bilhões de ataques a APIs. Esses ataques persistentes podem levar ao roubo de dados, danos à reputação da marca e multas regulatórias significativas. O abuso dessas APIs é uma preocupação crescente para empresas que dependem delas para disponibilizar seus dados e serviços.

Nesse contexto desafiador, este trabalho propõe o desenvolvimento de um Scanner de Segurança para Aplicações Web focado na detecção de vulnerabilidades conforme o modelo OWASP, fundamentado na engenharia do caos. Nossa ferramenta visa fortalecer a postura de segurança das organizações ao identificar e mitigar vulnerabilidades antes que sejam exploradas por agentes maliciosos. Além disso, busca-se democratizar o acesso à segurança digital, permitindo que pequenas empresas e pessoas físicas utilizem a ferramenta online e simplificada.

1.1 Tema

Segurança da Informação em redes de computadores.

1.2 Delimitação do Tema

Um número considerável de indivíduos dedica seus esforços à busca por maneiras de driblar as defesas dos mais variados sistemas: desde instituições bancárias até órgãos governamentais.

- **Cibercriminosos** buscam lucro financeiro utilizando métodos como *ransomware*, *phishing* e exploração de vulnerabilidades.
- **Hacktivistas**, motivados por ideologias políticas ou sociais, realizam ataques para promover causas.

- **Atores estatais**, patrocinados por governos, realizam espionagem ou sabotagem.
- **Ciberterroristas** visam causar caos atacando infraestruturas críticas.
- ***Insiders* maliciosos**, como funcionários descontentes ou negligentes, podem vazar dados internos.
- ***Script kiddies***, com pouca experiência técnica, utilizam ferramentas prontas para realizar ataques menos sofisticados.

Cada grupo representa uma ameaça distinta no cenário da segurança cibernética (Faronics, 2024). Um exemplo recente foi o ataque do *ransomware WannaCry* em 2017. Embora contido rapidamente por Marcus Hutchins (Agência Brasil, 2017), esse ataque destacou a vulnerabilidade das infraestruturas digitais globais.

Nosso trabalho foca na proteção da confidencialidade, integridade e disponibilidade das redes computacionais — elementos essenciais para o funcionamento eficaz das organizações.

1.3 Problematização do Tema

Conforme relatório do dfndr lab3 (PSafe), foram registrados 120,7 milhões de ataques cibernéticos no Brasil no primeiro semestre de 2018 — quase o dobro do ano anterior. Diante desses números preocupantes e considerando o avanço tecnológico até 2024, é plausível inferir que os ataques tenham aumentado significativamente.

Uma pesquisa conduzida pela Abrasca em parceria com o The Security Design Lab (SDL) em 2023 revelou lacunas significativas nas práticas de segurança digital adotadas pelas empresas brasileiras. Embora muitas organizações tenham mecanismos básicos para detectar ataques cibernéticos (93%), ainda há deficiências graves: 42% não possuem planos formais para resposta a incidentes e 65% não treinam suas equipes adequadamente para lidar com situações críticas.

Esse cenário reforça a necessidade urgente por soluções acessíveis que possam ser adotadas por pequenas empresas com recursos limitados.

1.4 Objetivo Geral

Desenvolver uma ferramenta abrangente para identificar e mitigar vulnerabilidades em aplicações web simuladas. Integrando conceitos da engenharia do caos aos testes de segurança cibernética baseados nas diretrizes OWASP, pretende-se fortalecer as defesas digitais das organizações contra ameaças conhecidas e desconhecidas.

1.5 Objetivos Específicos

I. Desenvolver um Scanner de Segurança para Aplicações Web que incorpore os princípios da engenharia do caos aos testes de penetração.

II. Integrar funcionalidades baseadas no modelo OWASP para detecção abrangente das principais ameaças cibernéticas.

III. Democratizar o acesso à segurança digital ao fornecer uma ferramenta acessível para pequenas empresas e usuários individuais com menos recursos técnicos ou financeiros.

IV. Comparar a eficácia dos testes baseados na engenharia do caos com métodos tradicionais em termos de precisão na detecção e mitigação de vulnerabilidades.

2. Fundamentação Teórica

Como anteriormente destacado, os testes de vulnerabilidade representam uma série complexa de procedimentos elaborados para identificar possíveis pontos fracos em redes ou sistemas. Por meio da exposição dessas fragilidades, é possível traçar um perfil detalhado dos pontos críticos e das potenciais vulnerabilidades às quais o sistema em análise pode estar exposto. Com base nesses resultados, é viável conceber uma estratégia abrangente de mitigação de vulnerabilidades, com o intuito

de fortalecer de forma ágil as defesas do sistema e garantir sua integridade e segurança.

Ao longo deste capítulo, serão explorados de maneira aprofundada aspectos técnicos e conceituais, bem como a estrutura detalhada dos principais tipos de ataques que serão empregados no contexto deste projeto.

2.1 Segurança da Informação

A segurança da informação representa um pilar essencial no panorama empresarial contemporâneo, onde a preservação dos ativos digitais é crucial para garantir a continuidade e a confiança nos negócios. Buscando proteger esses elementos fundamentais em todas as etapas do ciclo de vida da informação, as organizações adotam uma gama diversificada de estratégias e ferramentas. Essas medidas não se limitam apenas à implementação de políticas de segurança rigorosas e à adoção de tecnologias avançadas, mas também abrangem a criação de uma cultura organizacional focada na conscientização e no treinamento contínuo dos colaboradores sobre práticas de segurança (Whitman; Mattord, 2011).

De acordo com as reflexões de Lyra (2008), a segurança da informação repousa sobre três pilares basilares, os quais constituem a base para uma proteção eficaz dos dados. Ao abordar esse conceito, é imprescindível considerar não apenas a confidencialidade, integridade e disponibilidade das informações, mas também outros aspectos relacionados à segurança da informação, ajustados de acordo com as demandas específicas de cada organização.

Esses pilares são definidos detalhadamente como segue:

- Confidencialidade: Refere-se à garantia de que somente indivíduos autorizados tenham acesso às informações, protegendo assim a sensibilidade dos dados contra divulgação não autorizada.
- Integridade: Assegura que os dados permaneçam íntegros e inalterados ao longo do tempo, protegendo-os contra modificações indevidas que possam comprometer sua precisão e confiabilidade.
- Disponibilidade: Certifica-se de que as informações estejam sempre acessíveis para os usuários autorizados, sem interrupções ou falhas que possam prejudicar a continuidade das operações empresariais.

Diante do cenário cada vez mais fatigante das ameaças cibernéticas, as organizações estão intensificando seus esforços para fortalecer suas defesas e proteger seus ativos digitais contra-ataques maliciosos. Esse compromisso envolve não apenas a implementação de tecnologias de segurança avançadas, mas também a adoção de abordagens proativas, como testes de intrusão e monitoramento contínuo da rede. Ao priorizar a segurança da informação, as empresas podem mitigar os riscos de violações de dados e reforçar a confiança de seus clientes e parceiros comerciais.

2.2 Chaos Enginnering

Ao lançar uma aplicação em ambiente de produção, ela se depara com um emaranhado de incertezas: desde a imprevisibilidade da rede até a eventual indisponibilidade de serviços de terceiros, sem mencionar o desequilíbrio quanto ao fluxo de tráfego. É imprescindível que um aplicativo atual seja capaz de enfrentar esses desafios inesperados sem comprometer a experiência do usuário. Nesse contexto, as ferramentas de engenharia do caos emergem como um recurso importante, introduzindo de forma deliberada falhas nos sistemas de produção e monitorando minuciosamente suas reações diante dessas perturbações. À medida que o sistema responde conforme o previsto, tais experimentações sólidas e controladas solidificam a confiança dos desenvolvedores na robustez e resiliência do sistema (Simonsson *et al*., 2019).

Ao submeter os sistemas a cenários de estresse controlados, como falhas de rede ou sobrecarga de tráfego, a Engenharia do Caos oferece insights cruciais sobre seu desempenho em situações adversas. Essa análise em tempo real permite às equipes melhorarem proativamente a estabilidade e a confiabilidade dos sistemas, reduzindo os riscos de falhas não previstas (Basiri; Hochstein; Jones; Tucker, 2019).

Uma proeminente defensora e pioneira dessa prática é a empresa Netflix, que se destaca por desenvolver algumas ferramentas de código aberto conhecidas como Simian Army (Basiri; Hochstein; Jones; Tucker, 2019), (Gremlin, 2023) voltadas para a introdução de falhas controladas nos ambientes da AWS. A Netflix lançou a primeira ferramenta, chamada de *Chaos Monkey*, em 2010, com o objetivo de facilitar a migração de seus serviços de uma infraestrutura física para a nuvem (Agarwal, Rao e Mahendra, 2020). Ao longo do tempo, a equipe de engenharia da Netflix ampliou ainda

mais essas ferramentas, lançando também o *Chaos Gorilla* e o *Chaos Kong*, ambos destinados a introduzir falhas de diferentes maneiras na AWS. Essas duas ferramentas foram integradas ao *Chaos Monkey*, formando o Simian Army (Gremlin, 2023).

De acordo com Basiri *et al.* 2019, há quatro princípios fundamentais para a aplicação de experimentos utilizando a abordagem de Engenharia do Caos:

2.2.1 Construção de hipóteses em torno do estado estável

Este princípio envolve a formulação de hipóteses sobre o impacto de uma recuperação de falhas no estado estável da aplicação. A suposição inicial é que a aplicação, mesmo diante de uma falha, manterá sua estabilidade operacional. Por exemplo, em uma arquitetura de *software* distribuída por várias regiões geográficas, uma hipótese comum é de que, ao ocorrer uma falha em uma dessas regiões, o sistema redirecionará o tráfego automaticamente para outra região sem afetar significativamente a experiência do usuário. A hipótese, nesse caso, seria que o *failover* – ou seja, o redirecionamento automático de tráfego – teria um impacto mínimo na latência e na performance do sistema, preservando a experiência do usuário final de forma transparente (Basiri, A. *et al., 2019*).

2.2.2 Alternância entre eventos do mundo real

Cenários de teste tradicionais frequentemente falham em refletir as complexidades de um ambiente de produção real. No uso cotidiano de uma aplicação, eventos inesperados, como preenchimento incorreto de formulários, saturação de espaço em disco ou latências temporárias de rede, são comuns e podem afetar o desempenho do sistema. Assim, na Engenharia do Caos, é essencial criar experimentos que simulem esses eventos, permitindo a observação dos impactos causados por erros e situações atípicas. A abordagem visa preparar a aplicação para reagir adequadamente a eventos imprevisíveis que surgem apenas em produção, mitigando os riscos de falhas catastróficas (Basiri, A. *et al.*, 2019).

2.2.3 Execução de experimentos em ambiente de produção

Em muitos casos, é impraticável replicar com exatidão o ambiente de produção, devido à complexidade das integrações de serviço e às variáveis de uso real. Além disso, algumas regras de negócio específicas da empresa – como limitações de acesso baseadas em geolocalização, processos automatizados de segurança e fluxos de autorização complexos – muitas vezes não são ativadas em ambientes de homologação. Isso torna o ambiente de produção o único espaço capaz de validar o comportamento real da aplicação, em função das integrações e das regras que só são acionadas no contexto de produção. Assim, realizar experimentos no ambiente de produção permite verificar como o sistema responde em condições reais, identificando falhas ou áreas para aprimoramento (Basiri, A. *et al.*, 2019).

2.2.4 Automação contínua dos experimentos

A atualização constante de aplicações e sistemas demanda um processo de automação dos experimentos de caos, garantindo que novos lançamentos e alterações sejam testados continuamente quanto à resiliência. A automação é essencial para detectar vulnerabilidades recém-introduzidas, permitindo que a equipe identifique e corrija problemas antes que eles afetem os usuários finais. Com um pipeline de testes automatizado, as organizações podem monitorar a estabilidade da aplicação de forma proativa, aumentando a segurança e a confiabilidade do sistema (Basiri, A. *et al.*, 2019).

2.3 PENTEST: Teste de Penetração

Conforme Longatto (2017), a segurança da informação é uma disciplina abrangente, subdividida em várias áreas, incluindo: Segurança Defensiva, Ofensiva, Pesquisa, Gestão em Segurança, Investigação e Monitoramento. A segurança defensiva concentra-se nos mecanismos de proteção dos ativos, ou seja, nos meios para resguardar as informações contra ameaças potenciais, como o *hardening* de servidores, que reforça a segurança deles, e o desenvolvimento de *softwares* seguros. A segurança ofensiva, por sua vez, se caracteriza pela realização de testes que

simulam ataques reais, com o objetivo de avaliar a resiliência das defesas, sendo os testes de invasão (*Pentest*) um exemplo clássico dessa prática. A gestão em segurança envolve a criação e administração das políticas de segurança nas organizações, enquanto a investigação se dedica à perícia em ambientes computacionais, e o monitoramento se foca na análise de tráfego de redes.

O teste de penetração, comumente conhecido como *pentest*, é uma abordagem metodológica essencial para avaliar a segurança de sistemas operacionais, redes de computadores, *websites*, redes sem fio, bancos de dados, aplicativos e programas. Este processo visa descobrir, mapear e expor vulnerabilidades potenciais, permitindo que as organizações criem estratégias de defesa adequadas (Weidman, 2014; Moreno, 2015). O objetivo primordial do *pentest* é identificar falhas de segurança para que possam ser corrigidas, sem, no entanto, buscar obter acesso não autorizado a sistemas ou servidores (Moreno, 2015).

Ao final de cada conjunto de testes, é crucial a elaboração de um relatório detalhado que apresente as vulnerabilidades identificadas e as correções recomendadas. Esse relatório é então entregue ao cliente que solicitou o *pentest*, servindo como uma base para a implementação de melhorias na segurança (Moreno, 2015).

Existem diversas metodologias aplicáveis à execução de um *pentest*, com destaque para a SSTMM *(Security Testing Methodology Manual)* e a OWASP (*Open Web Application Security Project*), que são amplamente reconhecidas e utilizadas (Moreno, 2015).

2.3.1 Categorias de Pentest

Conforme Moreno (2015) e Giavaroto e Santos (2013), o *Pentest* pode ser classificado em vários tipos: *blind, double blind, blackbox, graybox, whitebox, tandem* e *reversal* (Figura 01).

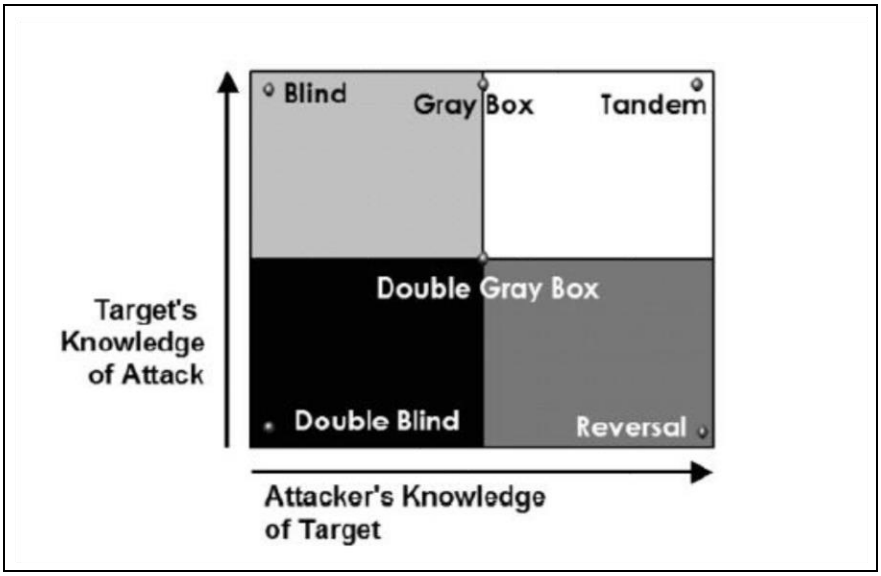
No *Pentest blind*, o *Pentester* não possui informações prévias sobre o sistema alvo, enquanto a equipe de TI da empresa sabe que um ataque ocorrerá e quais sistemas serão testados, o que possibilita a implementação de contramedidas para dificultar o ataque e proteger os sistemas. Já no *double blind*, tanto o *Pentester* quanto

a equipe de TI desconhecem que haverá um ataque, tornando o processo ainda mais desafiador.

O teste *blackbox*, ou caixa preta, é aquele em que o auditor não possui conhecimento prévio sobre o alvo ou sua infraestrutura de rede, simulando assim um ataque típico de um *hacker* malicioso, que geralmente desconhece a estrutura interna das empresas. No teste *graybox*, o *Pentester* tem informações parciais sobre o alvo e o ambiente a ser testado, permitindo uma avaliação mais direcionada. O teste *whitebox*, ou caixa branca, é o oposto do *blackbox*, onde o *Pentester* tem total conhecimento sobre o alvo, incluindo infraestrutura de rede, aplicações, endereços IP, *firewalls* e até código-fonte, sendo frequentemente utilizado para simular ataques internos, como os que poderiam ser perpetrados por funcionários descontentes.

No teste *tandem*, o alvo é informado sobre o ataque e os testes que serão realizados, enquanto o *Pentester* possui conhecimento completo sobre o sistema. O teste *reversal*, por outro lado, é onde o *Pentester* conhece o alvo, mas não é informado sobre o ataque ou os testes que serão aplicados. Independentemente do tipo, o objetivo principal do *Pentest* é sempre explorar e testar a eficácia das defesas dos sistemas, proporcionando uma avaliação realista e detalhada da segurança.

Figura 01: Matriz de correlação de conhecimento.



Fonte: HERZOG, 2010, p. 36.

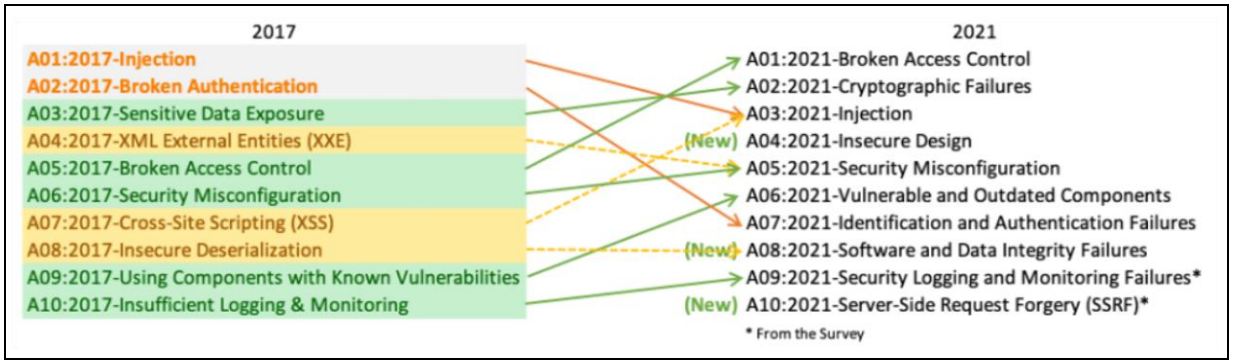
2.4 OWASP

A proteção de sistemas web contra ataques cibernéticos é uma prioridade essencial em Segurança da Informação. Entre os principais referenciais da área está o *Open Web Application Security Project* (OWASP), uma organização global sem fins lucrativos fundada em 2001, que visa aprimorar a segurança de aplicações *web*. O OWASP congrega profissionais de segurança, desenvolvedores e organizações em torno de boas práticas que orientam o desenvolvimento de software seguro (OWASP Foundation). As diretrizes da organização são amplamente reconhecidas, com destaque para o OWASP Top 10, uma lista atualizada dos principais riscos de segurança em aplicativos web. Este guia é um recurso fundamental para prevenir e mitigar vulnerabilidades e ameaças emergentes no contexto de desenvolvimento seguro.

2.4.1 OWASP Top Ten

OWASP Top Ten é uma lista utilizada amplamente na área de segurança (Figura 02), atualizada periodicamente, identifica as dez principais categorias de riscos de segurança em aplicações *web* mais exploradas no período, incluindo vulnerabilidades como Injeção de Código, Autenticação Quebrada e Exposição de Dados Sensíveis (OWASP Foundation). A ferramenta de scanner desenvolvida neste projeto é projetada para abordar e verificar a presença desses riscos nas aplicações, sendo embasada na lista publicada no ano de 2021.

Figura 02: Atualização OWASP Top 10 em 2021.



Fonte: OWASP, 2021.

2.4.2 OWASP WSTG

Outro recurso essencial oferecido pelo OWASP é o *Web Security Testing Guide* (WSTG), um guia completo para a realização de testes de segurança em aplicações web. O WSTG descreve metodologias de testes de segurança que cobrem diversos aspectos, desde o mapeamento de arquitetura da aplicação até testes detalhados de segurança para identificar vulnerabilidades comuns, como as listadas no OWASP Top 10. O guia é estruturado para auxiliar tanto desenvolvedores quanto analistas de segurança a executar testes sistemáticos, garantindo que as vulnerabilidades mais críticas sejam abordadas (OWASP Foundation).

2.5 CVE

Além das recomendações do *Open Web Application Security Project* (OWASP), outra fonte crucial para a segurança da informação é o CVE (*Common Vulnerabilities and Exposures*), um sistema que fornece identificadores exclusivos para vulnerabilidades de segurança conhecidas. O CVE é mantido pela organização MITRE Corporation e é amplamente utilizado na comunidade de segurança cibernética para identificar e referenciar vulnerabilidades específicas. Integrar a gestão de vulnerabilidades usando o CVE é essencial para mitigar riscos e manter a segurança dos sistemas web. Essa prática é amplamente recomendada por especialistas em segurança da informação (MITRE Corporation).

2.6 Esteios da comunicação: Protocolos de rede

Os protocolos atuam como o idioma dos computadores, permitindo que sistemas distintos se comuniquem de maneira eficiente e coordenada. São como as regras de etiqueta em uma conversa, garantindo que as informações sejam trocadas de forma compreensível e precisa entre os dispositivos envolvidos. Um exemplo proeminente desse "idioma digital" é o TCP/IP, ou Protocolo de Controle de Transmissão/Protocolo Internet, que serve como um padrão universal para a transmissão de dados na internet e redes locais (PROTOCOLOS, 2015).

Essencialmente, esses protocolos são os alicerces que sustentam a teia de comunicação que conecta o mundo digital.

2.6.1 Protocolo IP

O Protocolo de Internet (IP), sendo o componente primordial no âmbito das comunicações da Internet, assume a responsabilidade de atribuir endereços e direcionar os pacotes que fluem através da vasta teia de computadores interligados globalmente. Na arquitetura de redes IP, os dados são organizados em unidades discretas conhecidas como pacotes. Cada pacote constitui uma entidade composta por duas partes distintas: o cabeçalho, que se assemelha a um invólucro contendo informações de endereçamento essenciais, e os dados propriamente ditos, que representam a mensagem a ser transmitida. No processo de encapsulamento, cada pacote é dotado de um cabeçalho IP composto por uma série de campos de controle cruciais, dentre os quais se destacam os endereços IP de origem e destino. À medida que cada pacote percorre sua rota através dos diversos roteadores interligados, o endereço de destino é minuciosamente verificado em cada etapa, e o pacote é então encaminhado para o próximo salto na sequência de destinos, conforme descrito por PISA (2014).

2.6.2 Protocolo TCP

Tanenbaum (2011, p. 347) destaca que os pacotes IP não garantem a entrega nem a velocidade de transmissão para o destinatário. Nesse contexto, o Protocolo de Controle de Transmissão (TCP) assume o papel de garantir a entrega confiável dos datagramas, ajustando a velocidade de transmissão para utilizar a capacidade disponível sem causar congestionamentos na rede. A camada de transporte, juntamente com a camada de rede, constitui o cerne da hierarquia de protocolos que sustenta a comunicação em redes de computadores. Enquanto a camada de rede se encarrega da remessa de pacotes fim a fim através de datagramas ou circuitos virtuais, a camada de transporte complementa esse processo ao garantir o transporte de dados de forma confiável e independente das infraestruturas físicas subjacentes (Tanenbaum, 2011, p. 310).

Enquanto o IP se concentra na entrega eficiente, o TCP é responsável pela transferência fim a fim dos dados encapsulados pelas camadas superiores, proporcionando confiabilidade na comunicação.

Além de assegurar a confiabilidade na transmissão de dados, o TCP desempenha diversas funções essenciais:

- Permite a comunicação simultânea de múltiplas aplicações na rede dentro de um único dispositivo.
- Garante que todos os dados sejam recebidos pela aplicação correta, de forma confiável e na ordem correta, quando necessário.
- Implementa mecanismos robustos de tratamento de erros para lidar com eventuais falhas durante a transmissão de dados.

2.6.3 HTTP

O *Hypertext Transfer Protocol* (HTTP) é o protocolo de comunicação fundamental para a transmissão de dados na *web*, servindo como base para a interação entre clientes (como navegadores) e servidores web. Criado originalmente nos anos 1990, o HTTP é um protocolo baseado em requisições e respostas, onde o cliente envia uma solicitação ao servidor, que responde com o recurso solicitado, seja ele uma página HTML, uma imagem ou um arquivo JSON, por exemplo (Fielding e Reschke, 2014). As comunicações HTTP são especialmente importantes no contexto da segurança da informação, pois vulnerabilidades e riscos associados ao HTTP podem comprometer a integridade e a privacidade dos dados transferidos, principalmente em aplicativos web modernos.

Como protocolo "sem estado", o HTTP não retém informações entre as conexões, o que significa que cada solicitação é independente. No entanto, essa característica de “sem estado” exige o uso de mecanismos adicionais, como cookies e tokens, para persistir dados e autenticar usuários em sessões, aspectos que frequentemente apresentam riscos de segurança, como o roubo de cookies ou sessões comprometidas (Stallings e Brown, 2018). A compreensão dos componentes do HTTP e o uso correto de suas funcionalidades são, portanto, aspectos essenciais na criação de uma ferramenta de scanner de vulnerabilidades.

2.6.3.1 HTTP Status Code

Os Status Codes do HTTP são códigos de três dígitos enviados pelo servidor em resposta a uma solicitação do cliente, servindo como indicadores sobre o resultado da solicitação e o estado da comunicação. Eles são divididos em cinco classes principais, cada uma das quais oferece informações específicas sobre o comportamento do servidor e possíveis problemas na aplicação web:

1. 1xx (Informativo): Indica que a solicitação foi recebida e o processo continua.
2. 2xx (Sucesso): Indica que a solicitação foi recebida, compreendida e processada com sucesso. O código 200 OK, por exemplo, é uma resposta padrão para solicitações HTTP bem-sucedidas (Fielding e Reschke, 2014).
3. 3xx (Redirecionamento): Indica que a solicitação necessita de uma ação adicional para ser completada, como o código 301 *Moved Permanently*, usado para redirecionar permanentemente a URL solicitada.
4. 4xx (Erro do Cliente): Indica que houve um erro na solicitação do cliente. Por exemplo, o código 404 *Not Found* é retornado quando o recurso solicitado não existe, enquanto o código 403 *Forbidden* indica que o acesso ao recurso é negado.
5. 5xx (Erro do Servidor): Indica que houve uma falha no servidor ao processar a solicitação. O código 500 *Internal Server Error* é um dos mais comuns e sinaliza uma falha interna no servidor que pode ter origem em erros de código ou problemas de configuração (Stallings e Brown, 2018).

Esses códigos não apenas facilitam o diagnóstico de problemas de comunicação entre o cliente e o servidor, mas também têm implicações de segurança. Por exemplo, erros da série 4xx e 5xx podem expor informações sobre a estrutura interna de um sistema, revelando diretórios sensíveis ou fornecendo indícios de vulnerabilidades. A ferramenta de scanner de vulnerabilidades desenvolvida neste projeto é configurada para detectar respostas de erro, como 403 *Forbidden* e 500 *Internal Server Error*, que podem indicar configurações incorretas ou falhas de segurança no servidor (Kurose e Ross, 2021).

2.6.3.2 HTTPS

O *Hypertext Transfer Protocol Secure* (HTTPS) é a versão segura do HTTP, que utiliza criptografia para proteger a transmissão de dados entre o cliente e o servidor. Ele é essencial para garantir a confidencialidade e integridade das informações transferidas na *web*, protegendo dados sensíveis, como informações de login, transações financeiras e dados pessoais, contra interceptações e ataques de terceiros, como ataques *man-in-the-middle* (MITM) (Stallings e Brown, 2018).

O HTTPS é implementado usando o *Transport Layer Security* (TLS), protocolo criptográfico que substituiu o SSL (*Secure Sockets Layer*) e se tornou o padrão atual para conexões seguras na internet. O TLS autentica o servidor para o cliente por meio de certificados digitais, que são emitidos por Autoridades Certificadoras (CAs) confiáveis. Esses certificados garantem que o site é legítimo, evitando que o usuário seja enganado por sites falsos (Kurose e Ross, 2021).

2.7 DNS (Domain Name System)

Quando um usuário insere uma URL no navegador, seu computador inicia uma consulta progressiva aos servidores DNS para encontrar as informações e registros de recursos necessários. Esse processo continua até que o DNS localize a resposta correta no servidor DNS autoritativo associado ao domínio em questão (IBM, 2024). O *Domain Name System* (DNS) é um componente essencial para a funcionalidade da internet, permitindo a comunicação entre diversos dispositivos, desde smartphones e laptops até servidores de grandes websites comerciais. Cada dispositivo na internet possui um número único, conhecido como endereço IP, que é utilizado para a comunicação. No entanto, memorizar esses números seria impraticável para os usuários, que preferem utilizar nomes de domínio amigáveis, como exemplo.com. É aqui que entra o papel crucial do DNS, que traduz esses nomes em endereços IP, facilitando a navegação na web (AWS Route 53, 2024).

Desde sua concepção, os desenvolvedores do *Domain Name System* (DNS) estruturaram-no como um banco de dados hierárquico e distribuído, permitindo uma abordagem dinâmica na resolução de nomes de domínio. Essa hierarquia começa no nível raiz, representado por um ponto (.), e se ramifica em domínios de nível superior

(TLDs) como ".com", ".org", ".net", e também TLDs com códigos de país (ccTLDs) como ".uk" e ".jp", além dos domínios de segundo nível (IBM, 2024).

Quando um usuário insere uma URL no navegador, seu computador inicia uma consulta progressiva aos servidores DNS para encontrar as informações e registros de recursos necessários. Esse processo continua até que o DNS localize a resposta correta no servidor DNS autoritativo associado ao domínio em questão (IBM, 2024).

2.7.1 Subdomínio

Os subdomínios são divisões de um domínio principal, permitindo a organização e estruturação de serviços distintos sob um mesmo nome de domínio. Por exemplo, em um domínio como exemplo.com, subdomínios podem ser configurados como blog.exemplo.com ou loja.exemplo.com, onde cada subdomínio aponta para um servidor ou aplicação específica.

Os subdomínios ajudam a segmentar e organizar serviços, além de melhorar a escalabilidade e a segurança das aplicações web. No contexto de segurança, subdomínios podem ser utilizados para isolar áreas sensíveis de um site, como ambientes de desenvolvimento, teste ou zonas de acesso restrito. No entanto, subdomínios mal configurados podem representar uma vulnerabilidade, pois um subdomínio comprometido pode fornecer acesso a informações de outros subdomínios ou serviços associados ao domínio principal (Mcclure; Scambray; Kurtz, 2012). A ferramenta de scanner de vulnerabilidades pode verificar subdomínios para identificar riscos como subdomínio esquecido ou *misconfigurations*, que podem expor o sistema a ataques.

2.7.2 Diretórios

O diretório (path) de uma URL é a parte da URL que segue o domínio e que especifica a localização de um recurso específico no servidor, como /produtos ou /usuarios/login. O path é crucial para direcionar as solicitações do usuário para o local correto no servidor. No entanto, o uso inadequado ou inseguro de caminhos de URL pode abrir portas para ataques como *path traversal* ou exposição de diretórios sensíveis (OWASP Foundation).

A segurança em relação ao path envolve a implementação de boas práticas, como a validação rigorosa de entradas e a limitação de acesso a diretórios internos sensíveis, garantindo que usuários não autorizados não possam acessar áreas restritas. A ferramenta de scanner de vulnerabilidades verifica o path das URLs para detectar falhas de segurança, como diretórios acessíveis publicamente ou com permissões inadequadas, ajudando a proteger dados e a integridade do sistema.

3. Documentação

3.1 Requisitos

Segundo a ISO/IEC/IEEE (2017), um requisito de *software* representa a capacidade de um projeto em abordar de maneira eficaz um problema específico e alcançar um objetivo previamente definido. Esse objetivo pode ser formalizado por meio de um contrato, um padrão, uma metodologia ou um documento que especifique de forma detalhada as necessidades do usuário final. Documentar esses requisitos de maneira precisa e organizada é fundamental para o sucesso do desenvolvimento de software, pois permite alinhar as expectativas de todas as partes interessadas, desde desenvolvedores e engenheiros até gestores e usuários.

Conforme descrito por Sommerville (2016), os requisitos de *software* são comumente divididos em dois tipos principais: requisitos funcionais (RF) (Tabela 01) e requisitos não funcionais (RNF) (Tabela 02).

No contexto da Engenharia de Software, a documentação de requisitos se torna especialmente relevante em projetos de grande escala ou de missão crítica, onde as falhas de comunicação ou interpretação podem ter consequências significativas. Por isso, adotar metodologias e ferramentas que assegurem a clareza e a precisão dessa documentação é crucial para atender às exigências técnicas e garantir a confiabilidade do sistema em produção.

3.1.1 Requisitos Funcionais

Os requisitos funcionais (RF) descrevem as funcionalidades e serviços que o sistema deve oferecer, focando diretamente nas operações que o software deverá realizar para atender às necessidades do usuário (Sommerville, 2016).

Tabela 01: Requisitos Funcionais do Projeto

Identificação	Descrição
RF01	Implementação de uma plataforma para integração dos conceitos de Engenharia do Caos e OWASP em um ambiente de simulação de ataques.
RF02	O projeto deverá permitir que o usuário insira um domínio para a varredura de vulnerabilidades.
RF03	O projeto deve identificar e listar os subdomínios associados ao domínio inserido.
RF04	O projeto deve realizar testes de intrusão, incluindo *SQL Injection, Cross-Site Scripting (XSS)* e outros.
RF05	O projeto deve fornecer um *feedback* com métricas das ações executadas através do terminal de texto e gráficos.
RF06	O projeto deve gerar um relatório em formato .PDF, dividido entre um relatório técnico e um relatório corporativo, conforme as diretrizes da metodologia OWASP e métricas baseadas na Engenharia do Caos.

Fonte: Autoria própria, 2024.

3.1.2 Requisitos Não Funcionais

Os requisitos não funcionais (RNF) estabelecem restrições e critérios de qualidade para o sistema, abrangendo aspectos como desempenho, segurança, escalabilidade, usabilidade e manutenibilidade, que são essenciais para garantir a eficácia e a durabilidade do software no ambiente real de operação (Sommerville, 2016).

Tabela 02: Requisitos Não Funcionais do Projeto.

Identificação	Descrição
RNF01	A plataforma deve ser capaz de escalar horizontalmente, permitindo a execução de testes em múltiplos ambientes sem impactar a performance.
RNF02	O sistema deve ser otimizado para reduzir a latência das varreduras e simulações de falhas, garantindo que os testes não excedam 5 segundos de latência.
RNF03	A plataforma deve implementar criptografia de ponta a ponta para todos os dados sensíveis e garantir a conformidade com a LGPD (Lei Geral de Proteção de Dados).
RNF04	Garantir que todas as informações coletadas e relatórios gerados durante os testes sejam armazenados com integridade, utilizando armazenamento seguro e backup regular.
RNF05	O código-fonte deve seguir boas práticas de desenvolvimento de software, incluindo modularidade, documentação completa, e testes automatizados com cobertura mínima de 90%.

Fonte: Autoria própria, 2024.

3.2 Regras de Negócio

As regras de negócio (Tabela 03) constituem um elemento basilar no desenvolvimento de sistemas de informação, proporcionando uma estrutura normativa que orienta e harmoniza a execução das operações empresariais com os objetivos estratégicos da organização. Segundo Dallavalle e Cazarini (2000), as regras de negócio podem ser conceituadas como uma categoria de requisitos do sistema que encapsulam decisões sobre a condução dos processos organizacionais, representando, em essência, a manifestação de políticas internas e de diretrizes institucionais que guiam a operacionalização das atividades.

Essas regras, enquanto diretrizes expressas de maneira clara e específica, transcendem a mera implementação técnica ao ditarem práticas operacionais detalhadas e delinearem como cada processo deve ocorrer dentro do contexto organizacional. Em outras palavras, as regras de negócio articulam não apenas os

objetivos da organização, mas também a sua estrutura e comportamento, de modo que cada requisito reflete um compromisso explícito com padrões de qualidade e conformidade. Esse direcionamento normativo permite que o sistema de informação exerça uma função estruturante, estabelecendo não apenas permissões, mas também limitações que assegurem a aderência às políticas e aos padrões estipulados, resultando em operações consistentes e alinhadas ao escopo institucional.

Tabela 03: Regras de Negócio.

Identificador	Nome	Descrição
RN01	Identificação do Domínio	O projeto deve permitir que o usuário insira um domínio principal para iniciar a varredura.
RN02	Varredura de Subdomínios	O projeto deve identificar e listar todos os subdomínios associados ao domínio principal fornecido pelo usuário.
RN03	Testes de intrusão	O projeto deve realizar diversos testes de intrusão, incluindo *SQL Injection, Cross-Site Scripting (XSS)* e exploração de vulnerabilidades conhecidas.

RN04	Relato de Vulnerabilidades	O projeto deve notificar o usuário imediatamente em caso de detecção de vulnerabilidades críticas, recomendando ações corretivas.
RN05	Relatórios Detalhados	O projeto deve gerar relatórios detalhados em formatos .CSV e .PDF, contendo informações técnicas e executivas sobre as vulnerabilidades encontradas.
RN06	Configurações Personalizáveis	O projeto deve permitir ao usuário configurar quais tipos de testes de intrusão serão executados durante a varredura.

Fonte: Autoria própria, 2024.

4. Desenvolvimento

4.1 Desenvolvimento da Ferramenta

A ferramenta foi desenvolvida com base nos princípios das principais vulnerabilidades encontradas na web, conforme o OWASP Top Ten, e utilizou a metodologia de testes de segurança WSTG (OWASP), além dos princípios da Engenharia do Caos. A técnica de submeter a aplicação a um estado de sobrecarga controlada foi aplicada para simular condições de falha, fortalecendo assim a resiliência da aplicação (Simonsson *et al.,* 2019).

O desenvolvimento da aplicação seguiu uma arquitetura que integra um *back-end* robusto em Python, essencial para a execução de *scans* de segurança, *parsing* de *endpoints* e outras funcionalidades críticas na análise de vulnerabilidades. Utilizando o framework Flask, o back-end garante uma comunicação eficiente com o *front-end*, enquanto o React, juntamente com bibliotecas públicas, foi utilizado para criar uma interface interativa e intuitiva (Grinberg, 2018; Banks, 2020). No *front-end*, a aplicação também gera um relatório em PDF, que consolida e organiza os dados obtidos pelos *scans* e testes, facilitando a extração e análise detalhada dos resultados. Dessa forma, a arquitetura está alinhada com as práticas recomendadas de segurança pelo OWASP e com os princípios da Engenharia do Caos, simulando cenários de falha para assegurar a robustez e a integridade da aplicação (OWASP, 2021; Simonsson *et al*., 2019).

4.2 Dashboard

O *Dashboard* (Figura 03) é o módulo inicial da ferramenta, projetado para fornecer uma visão geral das atividades e métricas principais do sistema. Ele é responsável por apresentar os dados de forma visual e intuitiva, facilitando o monitoramento e análise dos resultados de segurança. Neste módulo, os dados são exibidos por meio de gráficos interativos, desenvolvidos com a biblioteca *Recharts*, uma biblioteca popular e eficiente para a criação de gráficos em aplicações web baseadas em *React*.

Os gráficos no *Dashboard* são alimentados por dados obtidos de um *endpoint*, o qual a aplicação disponibiliza para fornecer as métricas de uso. Esse *endpoint* retorna informações consolidadas, como o número total de *scans* realizados e o número de usuários cadastrados e ativos. Esses dados são processados pela aplicação para serem exibidos no formato de gráficos, permitindo que o administrador ou usuário visualize, de forma clara e organizada, o volume de atividades realizadas no sistema e o engajamento dos usuários.

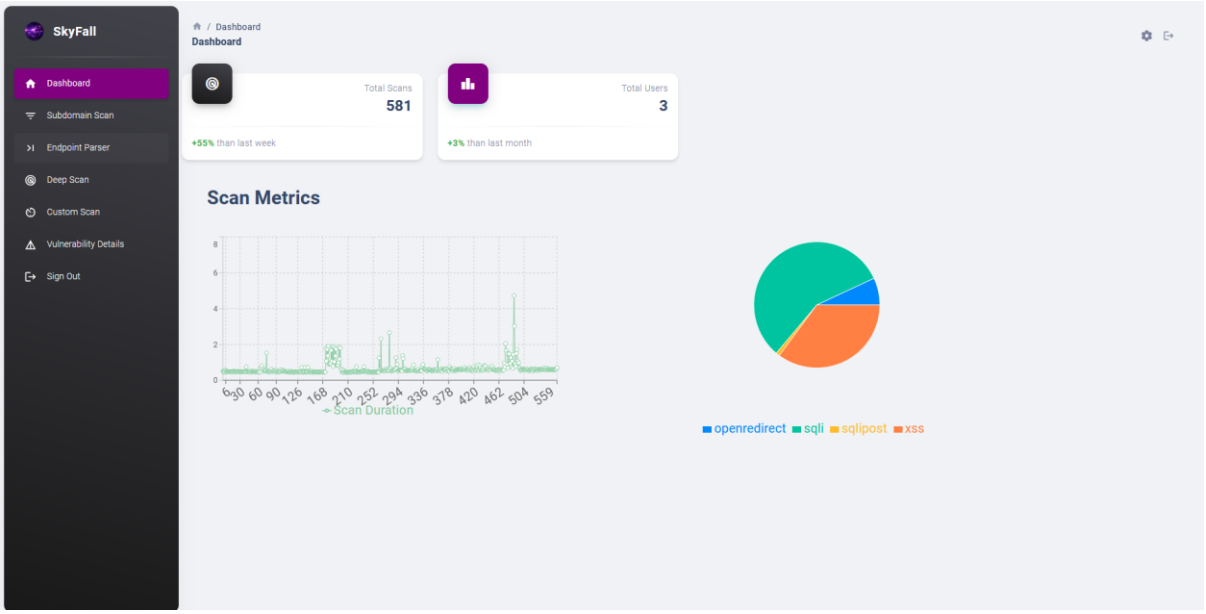
Além dos gráficos, a tela do *Dashboard* apresenta métricas em tempo real, incluindo:

- Total de *Scans* Realizados: Número acumulado de análises de vulnerabilidade executadas pelo sistema, permitindo uma visão rápida da utilização e eficácia da ferramenta.

- Número de Usuários Cadastrados: Quantidade de usuários registrados na aplicação, uma métrica que auxilia na gestão de acessos e na análise de engajamento dos usuários.

Esse módulo proporciona uma visão consolidada e prática do uso da ferramenta, sendo uma peça fundamental para o monitoramento da segurança e para a análise do desempenho da aplicação.

Figura 03: Painel *Dashboard*.



Fonte: Autoria própria, 2024.

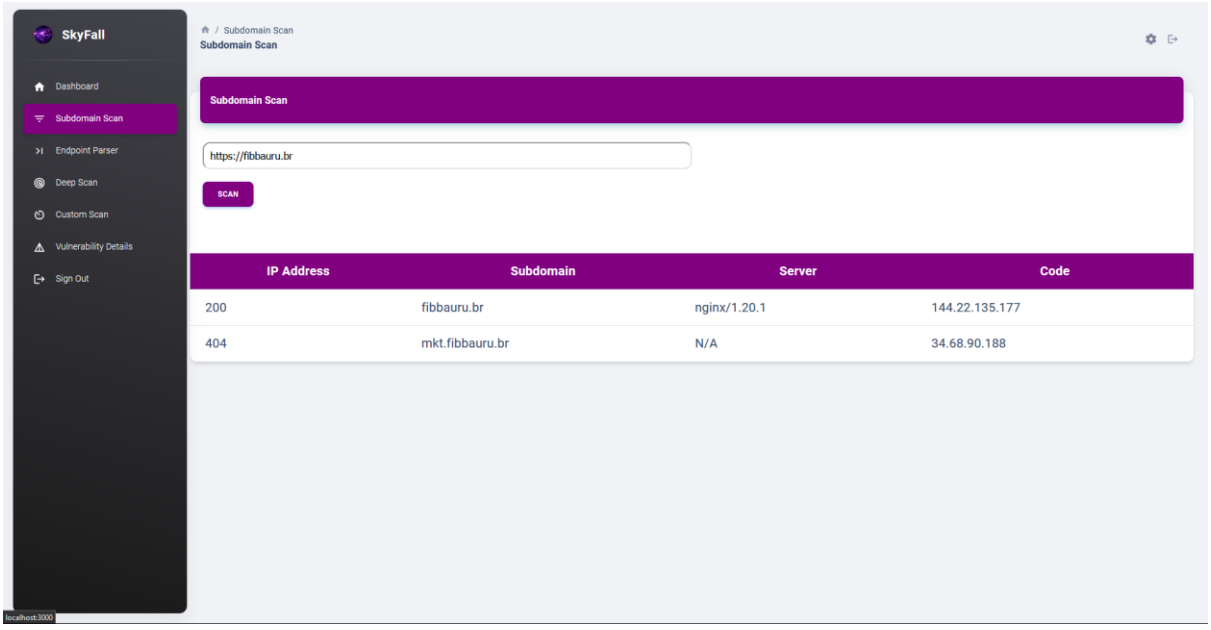
4.3 Subdomain Scanner

O módulo *Subdomain Scanner* (Figura 04) foi projetado para permitir a descoberta e mapeamento de subdomínios associados a um domínio principal, fornecendo uma visão detalhada da superfície de ataque que pode estar dispersa em subdomínios menos conhecidos e, portanto, mais suscetíveis a vulnerabilidades. Esse módulo opera a partir da inserção de uma URL pelo usuário, a qual é utilizada como base para uma série de consultas e técnicas de *scraping* aplicadas em diversos motores de busca, como Google, Bing e *DuckDuckGo*, além de uma integração com a API do *Shodan*.

A coleta de subdomínios através do *scraping* nos motores de busca consiste em automatizar requisições que retornam resultados indexados publicamente. Essas técnicas buscam variações do domínio inserido, identificando subdomínios que possam estar vinculados a ele. Os resultados são analisados e organizados pelo módulo, compondo um inventário detalhado de subdomínios que foram identificados como ativos e acessíveis. Complementando essa abordagem, a integração com a API do *Shodan* (Figura 05) permite expandir a identificação de subdomínios por meio de dados coletados em dispositivos conectados à internet, que não estão necessariamente indexados em motores de busca convencionais.

Essa integração entre técnicas de *scraping* e o uso da API do *Shodan* é fundamental para ampliar a cobertura e a precisão dos dados coletados pelo módulo. O *Subdomain Scanner*, portanto, representa uma ferramenta essencial para profissionais de segurança da informação, oferecendo um mapeamento extensivo e detalhado dos subdomínios de um domínio específico, o que contribui para uma análise de segurança mais profunda e eficaz. A detecção desses subdomínios é particularmente relevante no contexto da avaliação de superfícies de ataque, uma vez que subdomínios menos conhecidos podem apresentar vulnerabilidades significativas e servir de ponto de entrada para ataques.

Figura 04: Painel Módulo Subdomínio



Fonte: Autoria própria, 2024.

Figura 05: *Snippet code subdomain scanner.*

```
def forAPI(self):
    query = f"SELECT COUNT(*) FROM subdomains WHERE url='{self.url}'"
    cursor = self.db_conn.execute(query)
    count = cursor.fetchone()[0]
    if count == 0:
        list1=Search(self.url).domains()
        list2=Shodan(self.url).shodan_search()
        list3=Shodan(self.url).virus_total()
        subdomains=self.listOfSubdomains(list1,list2,list3)
        self.insert_subdomains(subdomains)
        subdomains=self.get_subdomains_from_db()
    else:
        subdomains = self.get_subdomains_from_db()
    return subdomains
```

Fonte: Autoria própria, 2024.

4.4 Endpoint Parser

O módulo *Endpoint Parser* (Figura 06) foi desenvolvido para realizar uma análise detalhada dos *endpoints* de um domínio específico, construindo um inventário de URLs relevantes para mapear potenciais superfícies de ataque e identificar pontos vulneráveis. Esse processo envolve a extração de URLs de diversas fontes, com foco na coleta de dados públicos arquivados e na estrutura interna do próprio domínio.

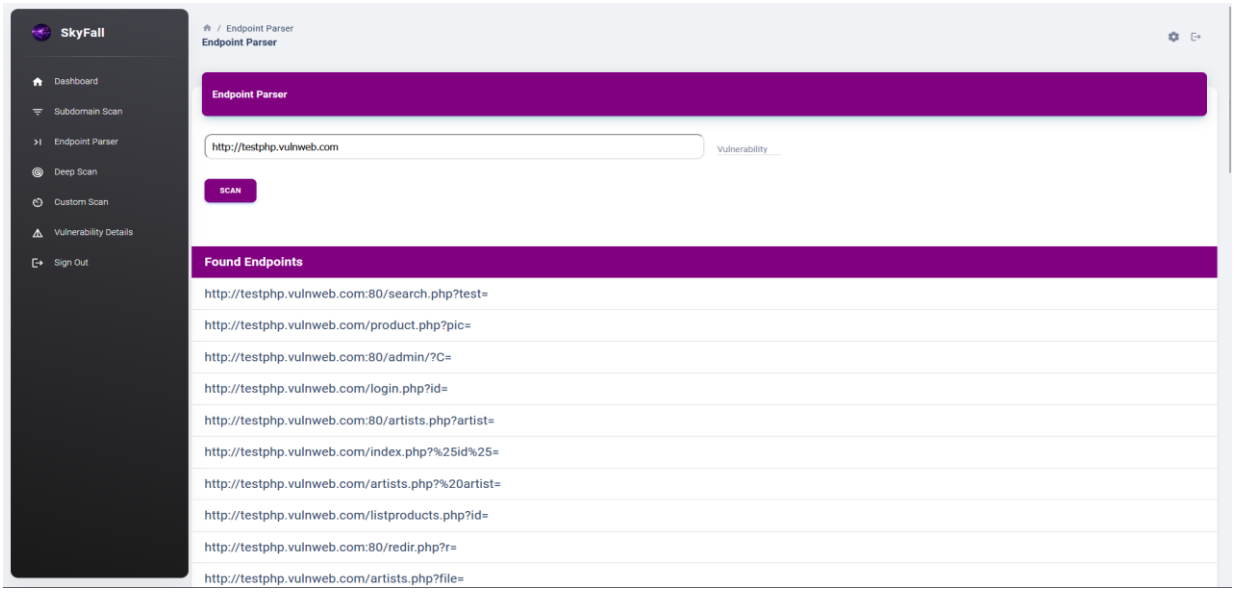
Inicialmente, o módulo realiza uma consulta ao *Web Archive*, uma biblioteca digital que armazena versões históricas de sites e páginas da internet ao longo do tempo. O *Web Archive* (Figura 07) é uma fonte valiosa para análise de segurança, pois permite acessar registros antigos de URLs e *endpoints* que podem estar inativos ou não indexados nos mecanismos de busca convencionais. Essa funcionalidade é essencial para obter uma visão histórica e abrangente de todos os *endpoints* que já estiveram ativos para um domínio específico, oferecendo uma perspectiva mais completa das superfícies de ataque potenciais. A consulta ao Web *Archive* aumenta as chances de identificar *endpoints* obsoletos que possam conter vulnerabilidades de segurança não mitigadas.

Em paralelo, o módulo executa o *scraping* na página principal do domínio por meio da função *FindLinksInPage*, que extrai links ancorados diretamente do HTML da página inicial. Com isso, o *Endpoint* Parser é capaz de capturar uma quantidade

significativa de *endpoints*, tanto a partir de registros históricos quanto de links ativos presentes na página.

Por fim, as URLs coletadas e filtradas são apresentadas ao usuário em uma tabela interativa no *front-end*, onde os dados são organizados para facilitar a navegação e análise de cada *endpoint*. Essa visualização permite uma consulta eficiente e a análise dos dados diretamente pela interface, otimizando o processo de mapeamento de possíveis superfícies de ataque para o domínio em questão.

Figura 06: Painel *endpoint parser*.



Fonte: Autoria própria, 2024.

Figura 07: *Snippet code endpoint parser.*

```python
# Prepare the URL for the Web Archive
if subs == True or subs == "True":
    url = f"https://web.archive.org/cdx/search/cdx?url=*.{domain}/*&output=txt&fl=original&collapse=urlkey&page=/"
else:
    url = f"https://web.archive.org/cdx/search/cdx?url={domain}/*&output=txt&fl=original&collapse=urlkey&page=/"

# Find anchor links in the page
alist = anchortags.FindLinksInPage(f'https://{domain}')

retry = True
response = None

# Attempt to connect to the Web Archive
while retry and retries <= int(retries):
    response, retry = requester.connector(url)
    retries += 1

    # Check if the response was successful
    if response is not None:  # If we have a valid response
        break

print(response)
# If no valid response, call the crawler
if response is None:
    print(f"Web Archive returned an error. Starting the crawler for the domain {domain}...")
    response = crawl_domain(domain)

# Blacklist for extensions to exclude
black_list = []
if exclude:
    if "," in exclude:
        black_list = exclude.split(",")
        black_list = [f".{ext.strip()}" for ext in black_list]  # Clean each extension
    else:
        black_list.append(f".{exclude.strip()}")  # Clean single extension

if exclude:
    print(f"\u001b[31m[!] URLs containing these extensions will be excluded from the results: {black_list}\u001b[0m\n")

ex = Extractor()
final_uris = ex.param_extract(response, level, black_list, placeholder)
final_uris.extend(alist)
final_uris = list(set(final_uris))  # Ensure unique URLs
```

Fonte: Autoria própria, 2024.

4.5 Scanners

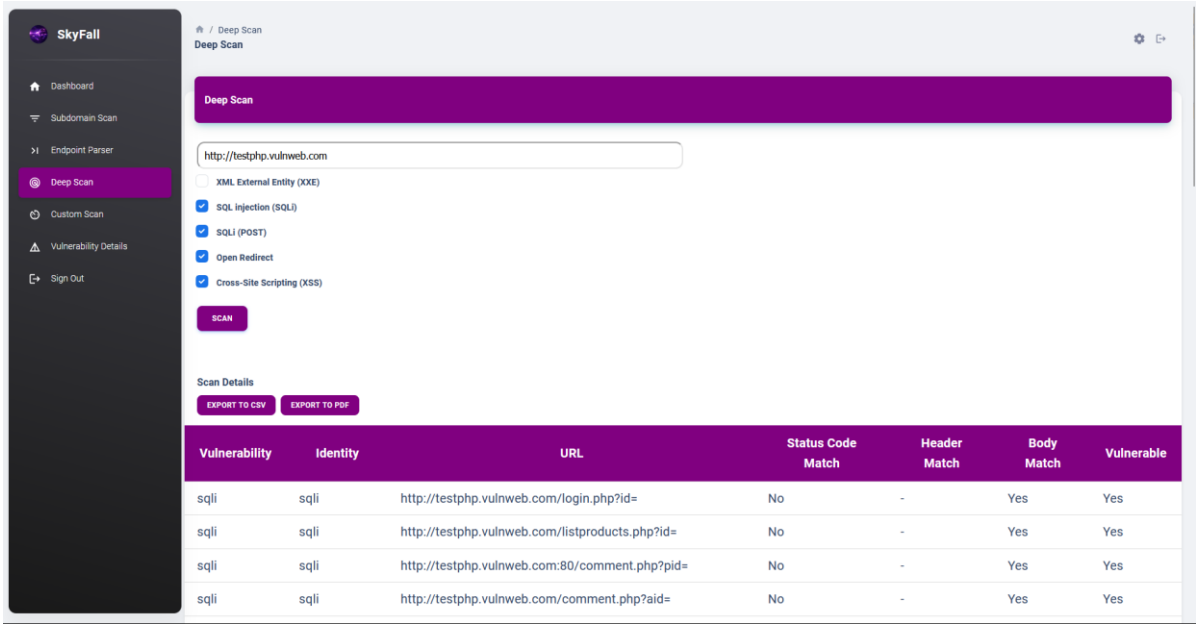
O módulo *Deep Scan* (Figura 08) foi desenvolvido para realizar uma análise aprofundada e automatizada de segurança em todos os *endpoints* de um domínio específico. Esse módulo se baseia no mapeamento inicial realizado pelo módulo *Endpoint Parser*, que identifica e organiza uma lista de *endpoints* acessíveis do domínio. A partir dessa lista, o *Deep Scan* executa uma série de testes personalizados para identificar possíveis vulnerabilidades de segurança. Para estruturar esses testes, o módulo utiliza *blueprints* de vulnerabilidades — modelos em formato JSON que descrevem potenciais falhas de segurança e comportamentos indesejados. Essas *blueprints* contêm dados específicos sobre cada vulnerabilidade, incluindo padrões de resposta e descrições de ameaças. Baseando-se nesses modelos, o *Deep Scan* realiza requisições automatizadas que incluem *payloads* maliciosos para simular ataques nos terminais detectados, coletando as respostas dos servidores para uma análise criteriosa.

A verificação de vulnerabilidades nos *endpoints* é realizada utilizando expressões regulares (*regex*), que filtram as respostas com base em padrões previamente definidos nas *blueprints*. As *regex* permitem que o módulo identifique com precisão características típicas de comportamentos vulneráveis, como falhas de segurança e exposição de dados. A comparação dos resultados obtidos com esses padrões permite ao módulo detectar de forma automatizada se um *endpoint* é suscetível a uma vulnerabilidade específica, resultando em uma análise sistemática e precisa. Esse procedimento reduz a necessidade de intervenção manual e torna o processo de identificação de falhas de segurança mais ágil e eficiente.

O módulo Custom *Scan* foi desenvolvido com o mesmo objetivo, mas oferece uma abordagem personalizada. Diferentemente do *Deep Scan*, o Custom *Scan* permite ao usuário especificar um único *endpoint* para análise, concentrando os testes de segurança em uma URL individual fornecida. Essa abordagem é ideal para situações em que o usuário deseja verificar uma potencial vulnerabilidade em um *endpoint* específico, sem a necessidade de escanear todo o domínio. O Custom *Scan* emprega o mesmo processo de criação de requisições e análise de respostas do *Deep Scan*, utilizando *payloads* e *blueprints* de vulnerabilidades. Entretanto, o foco é restrito ao *endpoint* selecionado, o que permite uma análise direcionada e mais rápida.

Portanto, ambos os módulos, *Deep Scan* e Custom *Scan*, foram desenvolvidos para automatizar o processo de teste de segurança, utilizando o mesmo mecanismo de análise de vulnerabilidades com níveis distintos de abrangência. O *Deep Scan* executa uma verificação completa de todos os terminais de um domínio, enquanto o *Custom Scan* se limita ao *endpoint* especificado pelo usuário. Essa arquitetura permite ao sistema oferecer tanto uma cobertura ampla de segurança quanto uma ferramenta específica e direcionada, otimizando a análise de vulnerabilidades de acordo com as necessidades do usuário.

Figura 08: Painel Deep Scan.



Fonte: Autoria própria, 2024.

4.6 Vulnerability Details

O módulo *Vulnerability Details* foi desenvolvido para fornecer uma descrição detalhada de cada vulnerabilidade identificada durante a análise de segurança, bem como suas classificações, potenciais impactos e métodos recomendados de mitigação. Ele abrange uma variedade de vulnerabilidades comuns em aplicações web, como *SQL Injection*, *Cross-Site Scripting (XSS)*, *Insecure Direct Object Reference (IDOR)*, entre outras, sendo organizado de maneira a explicar os aspectos técnicos e práticos de cada falha, facilitando o entendimento e a remediação por parte dos desenvolvedores e analistas de segurança.

Para cada vulnerabilidade, o módulo apresenta uma descrição abrangente, explicando seu funcionamento, o contexto em que pode ocorrer e os riscos que ela representa para a integridade e segurança da aplicação. Por exemplo, no caso de *SQL Injection (SQLi),* a explicação cobre o impacto de injeções de código SQL malicioso em uma aplicação, que permite ao atacante manipular e acessar dados no banco de dados de maneira não autorizada. Dentro dessa mesma vulnerabilidade, o módulo aprofunda-se em variantes específicas, como *Union-Based SQL Injection*, que utiliza o operador UNION para combinar consultas legítimas e maliciosas, e *Blind SQL*

Injection, onde o atacante utiliza perguntas booleanas para inferir respostas do banco de dados, mesmo quando os erros de consulta SQL não são exibidos na aplicação.

Cada vulnerabilidade listada no módulo também é classificada com base em sistemas amplamente aceitos, como o *Common Weakness Enumeration* (CWE) e o *Common Attack Pattern Enumeration and Classification* (CAPEC). Essas classificações ajudam a categorizar e identificar as vulnerabilidades em padrões conhecidos, como CWE-89 (SQL *Injection*) e CAPEC-66 (SQL *Injection*), proporcionando uma referência técnica robusta para os profissionais que utilizam o módulo. Além disso, o módulo especifica a severidade típica de cada vulnerabilidade, usando classificações como "Alta", "Média" ou "Baixa", dependendo do impacto potencial na aplicação.

4.7 Relatório PDF/CSV

Os módulos *Deep Scan* e *Custom Scan* foram desenvolvidos para oferecer não apenas uma análise aprofundada e personalizada dos *endpoints*, mas também uma maneira eficiente de exportar os resultados obtidos para futuras referências e documentação. Esses módulos contam com funcionalidades de exportação que permitem ao usuário extrair os dados gerados durante o processo de escaneamento em diferentes formatos, incluindo PDF (Figura 09) e CSV. Essa flexibilidade facilita o arquivamento, compartilhamento e integração dos resultados com outras ferramentas ou relatórios de segurança.

O formato PDF é especialmente útil para documentação formal e distribuição dos resultados de maneira estruturada. O relatório em PDF inclui detalhes sobre cada *endpoint* analisado, as vulnerabilidades identificadas e as respectivas descrições. Esse documento é formatado para fornecer uma visão geral acessível e pode ser utilizado em apresentações ou relatórios oficiais para equipes de segurança e gestão, garantindo que os resultados do *scan* sejam claros e organizados. Além disso, o PDF pode incluir tabelas, gráficos e seções dedicadas a cada vulnerabilidade detectada, proporcionando uma documentação visualmente adequada para consulta.

Por outro lado, a extração em CSV oferece uma opção prática para integrar os dados com outras plataformas de análise e ferramentas de segurança. O formato CSV permite uma visualização dos resultados em forma de tabela, que pode ser aberta e

editada em aplicativos de planilha como Microsoft Excel ou Google *Sheets*. Esse formato é especialmente útil para análises quantitativas, filtragem de dados e criação de gráficos, permitindo que a equipe de segurança manipule os resultados dos *scans* com flexibilidade. No arquivo CSV, cada linha representa um *endpoint* escaneado, e as colunas incluem informações como o tipo de vulnerabilidade encontrada, o status da requisição e o nível de severidade.

Em resumo, os módulos *Deep Scan* e *Custom Scan* oferecem recursos completos para extração dos resultados em formatos que atendem a diferentes necessidades, seja para documentação oficial com o PDF ou para análise e manipulação de dados com o CSV. Dessa forma, o sistema se torna uma ferramenta versátil e eficiente para profissionais de segurança que buscam não apenas identificar, mas também documentar e analisar vulnerabilidades com precisão.

Figura 09: Relatório em PDF.

SkyFall Vulnerability Report

Report Generated: 01/11/2024, 14:06:20

Vulnerability Type: sqli

SQL Injection (SQLi) allows attackers to manipulate SQL queries, potentially gaining unauthorized access to the database.
Mitigation: Use parameterized queries and prepared statements to prevent SQL injection attacks.

More information at OWASP

Vulnerability: sqli

Severity: high

Parameter	Value
Body Match	Yes
Identity	sqli
Info	An SQLi injection involves attacker sending SQL queries to the database server.
Severity	high
Status Code Match	Yes
Url	http://testasp.vulnweb.com/showthread.asp?id=
Vulnerability	Yes

Vulnerability: sqli

Severity: high

Parameter	Value
Body Match	Yes
Identity	sqli
Info	An SQLi injection involves attacker sending SQL queries to the database server.
Severity	high
Status Code Match	Yes
Url	http://testasp.vulnweb.com/showforum.asp?id=
Vulnerability	Yes

Fonte: Autoria própria, 2024.

5. Conclusão

O projeto alcançou de forma satisfatória seus principais objetivos, contribuindo para a segurança digital e inovando ao combinar métodos de teste tradicionais com *Chaos Engineering*. Com o desenvolvimento do *Scanner* de Segurança para Aplicações Web, foi possível integrar *Chaos Engineering* aos testes de penetração, criando uma ferramenta que simula cenários complexos de falha e avalia a resiliência dos sistemas em condições reais. Esse recurso mostrou-se eficaz para identificar falhas que poderiam passar despercebidas em testes convencionais, demonstrando o diferencial dessa abordagem na identificação e mitigação de vulnerabilidades.

A implementação de funcionalidades de detecção de vulnerabilidades, com base nas diretrizes do OWASP, foi outro objetivo alcançado, abrangendo várias categorias de ameaças definidas pela organização. Assim, usuários com diferentes níveis de experiência podem utilizar o scanner para avaliar a segurança de suas aplicações, compreendendo claramente as falhas encontradas e os riscos associados a cada uma delas.

Outro ponto relevante foi a contribuição para a democratização da segurança digital, um objetivo central deste projeto. A ferramenta foi desenvolvida para atender empresas de pequeno porte e usuários individuais, muitas vezes sem orçamento ou conhecimento técnico em segurança digital. Ao simplificar o uso e facilitar o acesso a informações sobre vulnerabilidades, o *scanner* cumpre seu papel social, promovendo conscientização e segurança entre públicos mais amplos.

Este estudo também abre caminho para aprimoramentos e novas funcionalidades no scanner de segurança. Futuros desenvolvimentos podem incorporar técnicas avançadas de inteligência artificial e aprendizado de máquina, possibilitando uma detecção ainda mais proativa e precisa de padrões de vulnerabilidade. Além disso, seria benéfico expandir o uso de *Chaos Engineering* para incluir testes específicos de estresse em diferentes tipos de arquitetura web e melhorar a interface da ferramenta para otimizar a experiência do usuário. Essas melhorias podem tornar a aplicação mais acessível e eficaz, permitindo uma análise mais profunda e personalizada dos riscos de segurança. Com o avanço constante das ameaças cibernéticas, é essencial manter o projeto atualizado e alinhado com as

novas demandas de segurança digital, beneficiando desde usuários individuais até empresas de maior porte.

Portanto, o projeto não apenas superou os desafios técnicos impostos, mas também oferece uma solução de impacto social e científico, promovendo uma segurança digital mais inclusiva e robusta.

6. Considerações Finais

Este projeto buscou não apenas resolver desafios técnicos na segurança de aplicações web, mas também gerar um impacto significativo na comunidade científica e na sociedade. Em um contexto de digitalização crescente, a segurança digital permanece, para muitos, um recurso inacessível, reservado a grandes corporações com orçamentos robustos e equipes especializadas. Assim, este trabalho propôs uma solução inovadora e tecnicamente sólida que visa democratizar a segurança digital, tornando-a acessível para pequenas empresas e usuários finais com menos conhecimento técnico e recursos financeiros.

Além do impacto direto na segurança dos usuários, este projeto possui potencial para contribuir à pesquisa científica, ao explorar a interseção entre engenharia do caos e cibersegurança. Ao aplicar conceitos de falha controlada para revelar vulnerabilidades ocultas, abre-se novos caminhos para pesquisa e experimentação na busca pela robustez digital. A metodologia empregada, que equilibra precisão técnica com acessibilidade, visa inspirar futuras investigações e o desenvolvimento de tecnologias que coloquem a segurança digital ao alcance de todos.

Com este trabalho, buscamos ecoar um propósito maior: não apenas proteger dados, mas fortalecer a dignidade digital daqueles que navegam no vasto território virtual. Ao fornecer conhecimento e ferramentas acessíveis, buscamos equipar cada usuário com a capacidade de proteger sua presença digital e, assim, reafirmar a liberdade e a autonomia em um espaço cada vez mais vulnerável.

Que este projeto inspire um legado, guiado pelo princípio eterno de que "*Non sibi sed omnibus*" – "Não para si, mas para todos". A verdadeira segurança digital é aquela que abraça a coletividade, erguendo um escudo invisível, compartilhado e duradouro, para todos os que habitam este mundo interconectado.

7. Referências

AGARWAL, A.; RAO, S. K. S.; MAHENDRA, B. M. *Comprehensive review of virtualization tools*. *International Research Journal of Engineering and Technology (IRJET)*, v. 7, n. 6, 2020. Disponível em: https://www.irjet.net/archives/V7/i6/IRJET-V7I6821.pdf. Acesso em: 11 out. 2024.

AGÊNCIA BRASIL. *Cibercrimes causaram prejuízos de bilhões de dólares no mundo em 2016*. Disponível em: https://agenciabrasil.ebc.com.br/geral/noticia/2017-05/cibercrimes-causaram-prejuizos-de-bilhoes-de-dolares-no-mundo-em-2016#:~:text=Uma%20estimativa%20da%20Cyberventures%20%E2%80%93%20consultoria,US%24%205%20bilh%C3%B5es%20em%202016. Acesso em: 10 jun. 2024.

BANKS, Alex; PORCELLI, Eve. *Learning React: modern patterns for developing React apps*. 2. ed. Sebastopol: O'Reilly Media, 2020. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=tDjrDwAAQBAJ. Acesso em: 13 mai. 2024.

BASIRI, A.; HOCHSTEIN, L.; JONES, N.; TUCKER, H. *Automating chaos experiments in production*. arXiv:1905.04648, 2019. Disponível em: http://arxiv.org/abs/1905.04648. Acesso em: 07 maio 2024.

DALLAVALLE, S. I.; CAZARINI, E. W. *Regras do negócio, um fator chave de sucesso no processo de desenvolvimento de sistemas de informação*. Anais do XX ENEGEP - Encontro Nacional de Engenharia de Produção, São Paulo, 2000. Disponível em: https://d1wqtxts1xzle7.cloudfront.net/46077734/Regras_do_negcio_fator_chave_de_sucesso_20160530-30540-z1tfze-libre.pdf. Acesso em: 10 fev. 2024.

FARONICS. *7 tipos de crimes e criminosos digitais*. 2024. Disponível em: https://www.faronics.com/pt-br/news/blog/7-tipos-de-crimes-e-criminosos-digitais. Acesso em: 10 nov. 2024.

FIELDING, R.; RESCHKE, J. *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing (RFC 7230)*. 2014. Internet Engineering Task Force (IETF). Disponível em: https://www.rfc-editor.org/rfc/rfc7230. Acesso em: 12 jun. 2024.

GIAVAROTO, S.; SANTOS, G. *Backtrack Linux: auditoria e teste de invasão em redes de computadores*. São Paulo: Moderna, 2013. Disponível em: http://site.livrariacultura.com.br/imagem/capitulo/30757884.pdf. Acesso em: 05 jun. 2024.

GREMLIN COMMUNITY. *Chaos Engineering: the history, principles and practice*. Disponível em: https://www.gremlin.com/community/tutorials/chaos-engineering-the-history-principles-and-practice. Acesso em: 07 maio 2024.

GRINBERG, M. *Flask Web Development: Developing Web Applications with Python*. 2. ed. Sebastopol: O'Reilly Media, 2018. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=cVlPDwAAQBAJ. Acesso em: 11 set. 2024.

HERZOG, P. *OSSTMM3: The Open Source Security Testing Methodology Manual*. 2010. Disponível em: https://www.isecom.org/OSSTMM.3.pdf. Acesso em: 15 jul. 2024.

IBM. *O que é um fluxograma?* 2024. Disponível em: https://www.ibm.com/br-pt/think/topics/flowchart. Acesso em: 10 nov. 2024.

INATEL. *Pesquisa inédita em cibersegurança avalia empresas de capital aberto no Brasil*. 2023. Disponível em: https://inatel.br/noticias/pesquisa-inedita-em-ciberseguranca-avalia-empresas-de-capital-aberto-no-brasil#:~:text=Foi%20divulgado%20na%20quarta%2Dfeira,pr%C3%A1ticas%20indicadas%20pelas%20principais%20ag%C3%AAncias. Acesso em: 10 nov. 2024.

ISO/IEC/IEEE. *Systems and software engineering — Vocabulary*. 2017.

KUROSE, J. F.; ROSS, K. W. *Computer Networking: A Top-Down Approach*. 8. ed. Pearson, 2021. Disponível em: https://www.cs.uga.edu/sites/default/files/CIS_CSCI_4760.pdf. Acesso em: 13 ago. 2024.

LIANG, X. *An intelligent, distributed and collaborative DDoS defense system*. 2021. Tese (Doutorado) — University of Pittsburgh. Disponível em: https://www.proquest.com/openview/80b76754f692f8e7999bcff7a131ded6/1?pq-origsite=gscholar&cbl=18750&diss=y. Acesso em: 08 abr. 2024.

LONGATTO, R. *Curso de Pentest*. Desec Security, São Paulo, 2017. Disponível em: https://www.youtube.com/watch?v=dCCKBun0KE8&t=451s. Acesso em: 15 jul. 2024.

LYRA, M. R. *Segurança e auditoria em sistemas da informação*. Rio de Janeiro: Ciência Moderna, 2008.

McCLURE, Stuart; SCAMBRAY, Joel; KURTZ, George. *Hacking Exposed 7: Network Security Secrets & Solutions*. 7. ed. Nova York: McGraw-Hill, 2012. Disponível em: https://d1wqtxts1xzle7.cloudfront.net/55428554/McGraw-Hill_-_Hacking_Exposed__3rd_Ed_-_Hacking_Exposed_Win2-libre.pdf. Acesso em: 09 nov. 2024.

MORENO, D. *Introdução ao Pentest*. São Paulo: Novatec Editora, 2015. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=FD-4DwAAQBAJ. Acesso em: 08 abr. 2024.

MORENO, D. *Pentest em Redes Sem Fio*. São Paulo: Novatec, 2016. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=tcm_CwAAQBAJ. Acesso em: 18 abr. 2024.

O que é o DNS. *Amazon AWS*. Disponível em: https://aws.amazon.com/pt/route53/what-is-dns/. Acesso em: 20 jun. 2024.

O que é o DNS?. *IBM*. Disponível em: https://www.ibm.com/br-pt/topics/dns. Acesso em: 14 jun. 2024.

OWASP Foundation. *About OWASP*. Disponível em: https://owasp.org/about/. Acesso em: 10 nov. 2024.

OWASP Foundation. *OWASP Top Ten Project*. Disponível em: https://owasp.org/www-project-top-ten/. Acesso em: 10 nov. 2024.

OWASP Foundation. *OWASP Web Security Testing Guide (WSTG)*. Disponível em: https://owasp.org/www-project-web-security-testing-guide/. Acesso em: 10 nov. 2024.

PISA, P. *O que é IP*. Disponível em: https://www.techtudo.com.br/noticias/2023/03/o-que-e-ip-e-para-que-serve-o-numero-edinfoeletro.ghtml. Acesso em: 10 maio 2024.
PROTOCOLOS. Direção: Márcio Lima. Produção: Cantinho da Informática. *YouTube*. Disponível em: https://youtu.be/031c04syZLs. Acesso em: 26 abr. 2024.

PSAFE. *Relatório da Segurança Digital no Brasil*. dfndr lab. Disponível em: https://www.psafe.com/dfndr-lab/wp-content/uploads/2018/08/dfndr-lab-Relat%C3%B3rio-da-Seguran%C3%A7a-Digital-no-Brasil-2%C2%BA-trimestre-de-2018.pdf. Acesso em: 10 jun. 2024.

SANTOS, M. *Encontro com Milton Santos ou O Mundo Global Visto do Lado de Cá*. Documentário. Direção: Sylvio Tendler. Produção: Caliban Produções Cinematográficas, 2006. 1 DVD (90 min). Disponível em: https://www.youtube.com/watch?v=ifZ7PNTazgY. Acesso em: 13 mai. 2024.

SIMONSSON, J.; et al. *Observability and chaos engineering on system calls for containerized applications in docker*. arXiv:1907.13039. Disponível em: http://arxiv.org/abs/1907.13039. Acesso em: 19 ago. 2024.

SOMMERVILLE, I. *Engenharia de Software*. 10. ed. São Paulo: Pearson Education, 2018.

STALLINGS, W.; BROWN, L. *Computer Security: Principles and Practice*. 4. ed. Pearson, 2018.

TANENBAUM, A. S. *Redes de Computadores*. 5. ed. São Paulo: Pearson Education, 2011.

WEIDMAN, G. *Testes de invasão: uma introdução prática ao hacking*. São Paulo: Novatec Editora, 2014. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=hnXVBAAAQBAJ&oi=fnd&pg=PT3&dq=Weidman,+G.+(2014).+Testes+de+Invas%C3%A3o:+Uma+Introdu%C3%A7%C3%A3o+pr%C3%A1tica+ao+hacking.+S%C3%A3o+Paulo:+Novatec+Editora.&ots=f4XLn3U3rE&sig=cK6O1tS7pnMq89ZifqBabW_f14Q#v=onepage&q&f=false. Acesso em: 02 fev. 2024.

WHITMAN, M. E.; MATTORD, H. J. *Principles of Information Security*. 4. ed. Boston: Course Technology, Cengage Learning, 2012.