INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ

CAMPUS FLORIANO

CURSO TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ARTHUR STRELOW LIMA

ANÁLISE DE SEGURANÇA NA PLATAFORMA BIBLIOTECÁRIA SARDES:

Estudo de caso com Pentest

FLORIANO - PI

2024

ARTHUR STRELOW LIMA

ANÁLISE DE SEGURANÇA NA PLATAFORMA BIBLIOTECÁRIA SARDES:
Estudo de caso com Pentest

Trabalho de Conclusão de Curso (artigo científico) apresentado como exigência parcial para obtenção do diploma do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal do Piauí, Campus Floriano .

Orientador: Prof. Me. Silvino Marques da Silva Júnior.

FLORIANO - PI
2024

ANÁLISE DE SEGURANÇA NA PLATAFORMA BIBLIOTECÁRIA SARDES: ESTUDO DE CASO COM PENTEST

Arthur Strelow Lima[1]

Silvino Marques da Silva Júnior[2]

RESUMO

Com o avanço constante das práticas de gestão de segurança da informação, torna-se fundamental a realização de testes de intrusão para identificar e mitigar possíveis vulnerabilidades em sistemas. Esses testes consistem em simulações controladas de ataques cibernéticos, reproduzindo cenários reais de invasões para avaliar a robustez dos sistemas de segurança. No contexto atual, os sites que compõem a web se destacam como alvos frequentes, principalmente devido ao interesse de indivíduos mal-intencionados em acessar e explorar dados sensíveis que ali estão armazenados. A metodologia adotada neste trabalho consistiu em uma pesquisa bibliográfica com o objetivo de fundamentar os principais conceitos de segurança cibernética. Para isso, foram utilizados artigos científicos, fontes bibliográficas, teses, dissertações e livros especializados na área. Além disso, foi realizado um estudo de caso que complementou e reforçou os achados obtidos por meio da pesquisa teórica. Os resultados obtidos, por meio da análise de vulnerabilidades realizada na plataforma alvo, identificaram falhas de segurança significativas que impactam de maneira expressiva a integridade da aplicação.

Palavras-chave: segurança da Informação; pentest; vulnerabilidades; cibersegurança.

ABSTRACT

With the constant advancement of information security management practices, it has become essential to perform intrusion tests to identify and mitigate potential vulnerabilities in systems. These tests consist of controlled simulations of cyberattacks, reproducing real intrusion scenarios to assess the robustness of security systems. In the current context, websites that make up the web stand out as frequent targets, mainly due to the interest of malicious individuals in accessing and exploiting sensitive data stored there. The methodology adopted in this work consisted of a bibliographical research with the objective of substantiating the main concepts of cybersecurity. For this, scientific articles, bibliographical sources, theses, dissertations and specialized books in the area were used. In addition, a case study was carried out to complement and reinforce the findings obtained through theoretical research. The results obtained, through the vulnerability analysis performed on the target platform, identified significant security flaws that significantly impact the integrity of the application.

Keywords: information Security; pentest; vulnerabilities; cybersecurity.

---

[1] Graduando em Tecnologia em Análise e Desenvolvimento de Sistemas. Instituto Federal do Piauí - Campus Floriano. caflo.2022114TADS15@aluno.ifpi.edu.br.

[2] Doutorando em Ensino na Universidade do Vale do Taquari – Univates. Mestre em Tecnologia e Gestão em EaD. Professor do Ensino Básico, Técnico e Tecnológico na área de Informática. Instituto Federal do Piauí - Campus Floriano. silvinomarques@ifpi.edu.br.

Data de aprovação: 08/01/2025 (data de apresentação do TCC)

1 INTRODUÇÃO

A segurança da informação emergiu como uma prioridade crucial, impulsionada pela percepção das grandes corporações de que, com o passar dos anos, dados e informações tornaram-se ativos estratégicos. Nesse contexto, a proteção adequada se torna essencial, especialmente considerando sua integração intrínseca com a Tecnologia da Informação (Galegale; Fontes, 2017). Então destaca-se a significância dos pentesters, especialistas em segurança da informação que realizam testes de penetração para identificar e corrigir possíveis vulnerabilidades.

No entanto, o aumento de aplicações web trouxe consigo o aumento do número de ataques cibernéticos, ainda considerando que com a pandemia do vírus COVID-19 apresentou uma modalidade de trabalho à distância, muitos desses ataques são destinados ao fator mais fraco dentro das medidas de seguranças de uma corporação, o usuário (Nagli, 2022).

Com o aumento dos ciberataques em diferentes aplicações, incluindo plataformas escolares, alunos, por falta de familiaridade com a legislação e compreensão dos impactos gerais, muitas vezes se envolvem em ações prejudiciais, sem plena consciência das consequências, motivados pela busca de integração social. Devido, os mesmos interpretarem a internet como um espaço sem limites, enquanto outros, carentes de educação sobre segurança da informação, desconhecem métodos para se proteger contra ameaças comuns, como o *phishing*.

É notório que a vulnerabilidade se manifesta como condições que, quando exploradas por indivíduos mal intencionados, geram um comportamento distinto do esperado na aplicação, ocasionando permissões inadequadas para usuários comuns (Santos e Soares, 2018). É possível considerar diversas razões para o comportamento atípico da aplicação como por exemplo, a ausência de atualizações do servidor e o uso de *frameworks* ou extensões que podem deixar a aplicação vulnerável. Além disso, a parte de segurança é negligenciada pelos programadores, que muitas vezes priorizam outras áreas durante o desenvolvimento.

Segundo a organização sem fins lucrativos, *Open Web Application Security* (2021), as falhas de segurança mais visadas na atualidade são: Quebra de Controle

de Acesso, Falhas Criptográficas e Injeção. Isso evidencia como os *hackers* buscam explorar todas as vulnerabilidades possíveis dentro de uma aplicação. No panorama atual, o Brasil figura como o segundo país com o maior número de ataques *hackers* entre os 193 países existentes com 357.422 ataques no segundo semestre de 2023 (Maia, 2024).

Certamente, entre as numerosas aplicações web disponíveis na internet, destaca-se o PERGAMUM, sendo especialmente relevante como sistema de gerenciamento de bibliotecas utilizado pelo Instituto Federal do Piauí (IFPI). É oferecido aos usuários a possibilidade de buscar, solicitar e gerenciar empréstimos de livros, facilitando o acesso ao acervo da instituição de forma prática e eficiente. Isso reforça a necessidade de manter a segurança da aplicação, pois ataques cibernéticos podem não apenas comprometer seu funcionamento, mas também causar problemas como impedir que o aluno renove um empréstimo, resultando em multas.

Deste modo, este artigo aborda a exploração de vulnerabilidades em uma aplicação web, focando na análise de segurança na plataforma bibliotecária PERGAMUM. O estudo busca identificar vulnerabilidades, avaliar os riscos e propor soluções para mitigação. Além disso, são apresentados aspectos gerais da segurança da informação e ferramentas utilizadas em testes de invasão, com o intuito de identificar e catalogar as falhas na aplicação web PERGAMUM. Com isso, o trabalho visa destacar a importância contínua dos profissionais de segurança no desenvolvimento e manutenção de uma aplicação, uma vez que a segurança é um estado momentâneo e não definitivo.

2 REFERENCIAL TEÓRICO

2.1 SEGURANÇA DA INFORMAÇÃO

A nova era destaca-se pela alta imersão do mundo digital na vida cotidiana das pessoas, fazendo da segurança da informação um alicerce crucial, indispensável para assegurar a confiança e a estabilidade em vários setores (Sousa, 2024). É importante conceituar a Segurança da Informação, um princípio fundamental que envolve a proteção dos dados pertencentes a pessoas físicas e jurídicas (Ramos, 2024). Ademais, com o passar dos anos, é evidente que a

informação vai atribuindo mais valor simbólico e monetário à medida que a internet se torna um 'mundo' onde as pessoas passam mais tempo do que no 'mundo real'.

Devido ao grande valor da informação, pessoas mal-intencionadas, conhecidas como *crackers* ou *black hat hackers*, indivíduos tecnicamente habilidosos que exploram sistemas de computador ou redes para encontrar falhas e explorá-las com intenções maliciosas. Essas pessoas tendem a querer controlar indivíduos, instituições públicas e privadas, já que esses órgãos possuem uma riqueza extrema em informação e, dependendo do nível e da quantidade de dados obtidos por meios ilícitos, pode haver um valor monetário gigantesco envolvido (Sanches; Antunes; Lopes, 2022).

Desse modo, reforça-se a ideia de que a informação é um ativo valioso, dependendo do valor a ela atribuído. Surge então a necessidade de adotar práticas para protegê-la contra ameaças. Um dos conjuntos de práticas possíveis é a tríade CIA, que abrange a confidencialidade, a integridade e a disponibilidade dos dados, com o objetivo de mitigar os riscos a que esses dados podem estar expostos (ISO, 2013). Em seguida, examinar-se detalhadamente cada um dos componentes da tríade CIA: confidencialidade, integridade e disponibilidade:

A confidencialidade é um elemento crucial da segurança da informação, assegura que apenas indivíduos autorizados acessem dados sensíveis, protegendo-os de acessos não autorizados. É fundamental para manter a privacidade e o sigilo das informações (Barcelos, et al., 2021). Um dos mecanismos usados para garantir essa proteção é a criptografia. As criptografias mais modernas demandam considerável processamento, tornando extremamente difícil para um invasor decifrar os dados, mesmo que consiga acessá-los.

A Integridade refere-se à autenticidade e à proteção contra adulterações por partes não autorizadas. Mecanismos como *checksums* e *hashes* são operações matemáticas usadas para detectar alterações em dados ou arquivos. Através de funções de *hashing*, é gerado um código único que permite verificar a integridade da informação; se os valores não corresponderem, indica que os dados foram modificados. Esse pilar da tríade é fundamental para garantir a precisão e consistência dos dados ao longo do tempo (Barcelos, et al., 2021).

A disponibilidade assegura que sistemas e dados estejam acessíveis para uso legítimo. Se a segurança falhar ou ataques cibernéticos tornarem os recursos inacessíveis, a operação da organização será prejudicada. Fatores como ataques DDoS (Negação de Serviço, que sobrecarrega o servidor com requisições), *ransomware* (que criptografa dados ao infectar o sistema), desastres naturais e quedas de energia podem tornar os dados inativos ou temporariamente inacessíveis, deixando servidores inoperantes. Torna-se crucial garantir a constante disponibilidade de sistemas e dados. A adoção de redundâncias, backups regulares e planos de recuperação de desastres é fundamental para assegurar essa continuidade (Barcelos, *et al*., 2021).

Considerando esses aspectos, a segurança da informação não é regida apenas pela tríade CIA, pois existem outras variáveis que fortalecem todo o setor de segurança, como a irretratabilidade e a autenticidade (Swapna, *et al*., 2022). Esses fatores aumentam a robustez do sistema, garantindo que a segurança não dependa exclusivamente de um único elemento, mas sim de um conjunto abrangente de práticas.

A principal meta da segurança da informação é proteger os recursos de dados contra uma variedade de ameaças, mitigando os riscos para as atividades comerciais. Isso ressalta a importância de investir em segurança desde o início de qualquer aplicação, projeto ou sistema, garantindo a prevenção de problemas para todas as partes envolvidas.

2.2 SEGURANÇA EM APLICAÇÕES WEB

A segurança em aplicações Web abrange a proteção do código, das bibliotecas e dos servidores Web. Esses aplicativos, acessados por navegadores e que utilizam scripts do lado do servidor e do cliente, podem ser desenvolvidos em várias linguagens e rodar em qualquer sistema operacional (Aydos, *et al.,* 2021). Dessa forma, garantir a segurança em todos esses níveis é fundamental para a integridade das aplicações Web. A incidência de ataques bem-sucedidos em aplicações web é significativa devido às vulnerabilidades presentes em vários níveis (Martins, 2018*)*.

Ao longo dos anos, em resposta a esses desafios, surgiram organizações sem fins lucrativos e fóruns dedicados a auxiliar desenvolvedores na identificação e correção de possíveis vulnerabilidades que possam surgir durante ou após o processo de desenvolvimento de aplicações. Exemplos notáveis incluem a *Open Web Application Security Project* (OWASP), a *Web Application Security Consortium* (WASC) e o *Common Weakness Enumeration* (CWE).

A *Open Web Application Security Project* (OWASP) é uma organização sem fins lucrativos reconhecida globalmente por seu compromisso em melhorar a segurança do software. Por meio de recursos como guias e ferramentas, a OWASP oferece suporte vital para profissionais de segurança e desenvolvedores na detecção e correção de vulnerabilidades em aplicações web (Kuncoro; Fayruz Rahma; Eng, 2022). Projetos emblemáticos, como o Top 10 de Segurança de Aplicações Web, estabelecem padrões na indústria, fornecendo *insights* essenciais sobre ameaças predominantes e melhores práticas para proteger sistemas contra ataques cibernéticos.

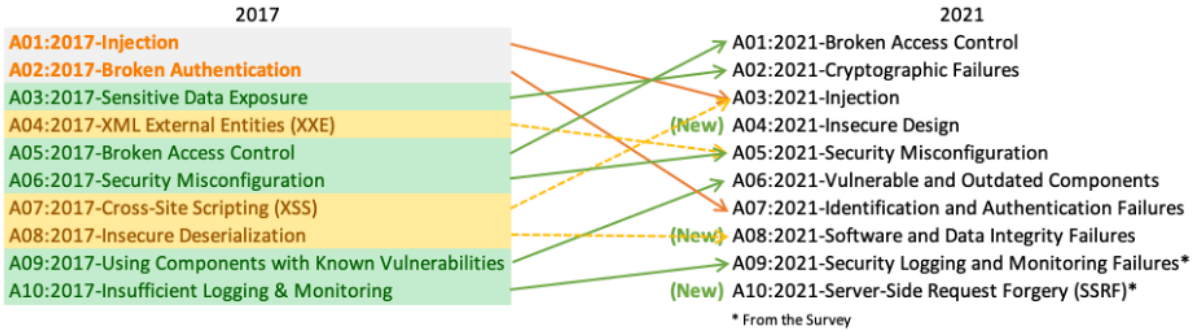
A *Web Application Security Consortium* (WASC) é uma organização global que desempenha um papel fundamental na promoção da segurança das aplicações web (Saveliev, 2020). Por meio de suas iniciativas, como o *Web Security Threat Classification* e o *Web Hacking Incident Database*, a WASC fornece recursos essenciais para profissionais de segurança e desenvolvedores identificarem e abordarem ameaças emergentes. Seus projetos são amplamente reconhecidos na indústria, oferecendo *insights* valiosos sobre vulnerabilidades e melhores práticas para mitigar riscos de segurança em aplicações web.

O *Common Weakness Enumeration* (CWE) é um projeto que fornece uma lista detalhada e classificação de vulnerabilidades de software comuns. Desenvolvido pelo MITRE Corporation, o CWE é uma referência essencial para profissionais de segurança e desenvolvedores, ajudando a identificar e mitigar riscos de segurança em sistemas de software (Wang; Qin; Chow, 2021). Ao categorizar e descrever as fraquezas comuns, o CWE ajuda a melhorar a compreensão e a abordagem das vulnerabilidades, permitindo a implementação de medidas preventivas mais eficazes.

2.2.1 Owasp Top Ten

A OWASP possui um Top 10 Riscos de Segurança em Aplicações Web (conforme a Figura 1) que destaca as principais vulnerabilidades enfrentadas por desenvolvedores e empresas na proteção de seus sistemas.

Figura 1 - Principais Vulnerabilidades em Sistemas Web



Fonte: OWASP (2021)

Com base no livro "Segurança em Aplicação Web" da Casa do Código, que oferece uma abordagem detalhada e prática sobre como mitigar esses riscos, serão apresentadas algumas das vulnerabilidades listadas no Top 10, conforme a correspondência com aquelas abordadas no livro e no ranking.

A vulnerabilidade A03:2021-*Injection* (Injeção**)** utiliza HTTP/HTTPS frente a ataques de injeção SQL é discutida, destacando como esses ataques visam a extração não autorizada de dados sensíveis e comprometem a segurança digital (Alghawazi, Alghazzawi & Alarifi, 2022). Trata-se de uma vulnerabilidade que afeta sistemas conectados a bancos de dados, especialmente aqueles que utilizam a linguagem SQL (*Structured Query Language*). O ataque, embora simples, envolve a inserção de caracteres que interrompem o comando SQL original, como um ponto e vírgula (“;”) ou uma aspa simples (“’”). Essa interrupção permite que o invasor insira e execute comandos arbitrários no banco de dados, explorando a quebra do comando inicial para realizar ações não autorizadas.

A A05:2021-*Security Misconfiguration* (Configuração incorreta de segurança), ou seja, a exposição de dados sensíveis representa uma grave ameaça à

confidencialidade dos dados organizacionais e pessoais, resultando de sistemas comprometidos, dispositivos perdidos ou dados não criptografados, tanto em armazenamento quanto em transmissão (Jyothi, et al. 2022). É evidente que muitos dos vazamentos de dados ocorrem devido a ataques cibernéticos; no entanto, uma parcela significativa resulta de erros cometidos por proprietários ou funcionários que operam o sistema.

Além disso, existem combinações de vulnerabilidades que podem ser exploradas, como uma aplicação que não possui outros usuários com permissões limitadas no banco de dados. Quando essa aplicação utiliza o usuário *root*, que possui permissões para executar qualquer operação no banco de dados, a situação se torna ainda mais crítica. Se essa configuração for combinada com uma possível injeção de SQL, um atacante pode obter controle total sobre o banco de dados, podendo executar qualquer comando ou operação, exacerbando significativamente os riscos de segurança (Zhang, et al., 2017).

Também há tipos de vulnerabilidades que, à primeira vista, podem parecer triviais. No entanto, ainda existem numerosas aplicações web que não modificam ou eliminam as configurações padrão das ferramentas que utilizam, seja do banco de dados ou do servidor.

Este tipo de vulnerabilidade pode ser explorado quando um hacker emprega ferramentas para identificar as tecnologias utilizadas na aplicação, como o *Wappalyzer*, um plugin de navegador que exibe as tecnologias empregadas ao visitar um site.

Essa situação pode resultar em uma das vulnerabilidades mais significativas, atribuída frequentemente à negligência do desenvolvedor, decorrente do excesso de plugins ou componentes obsoletos ou descontinuados na aplicação, o que pode acarretar em uma vulnerabilidade cuja gravidade varia de leve a crítica.

Semelhante à vulnerabilidade de Injeção SQL, a vulnerabilidade de categoria A07:2021-Identification and Authentication Failures (Identificação e Falhas de autenticação), pode ser representado pelo *Cross-Site Scripting* (XSS), um script feito em JavaScript, é uma falha de segurança conhecida há algum tempo, porém continua a ser extremamente eficaz. Um exemplo famoso foi o ‘Samy Worm’ no MySpace, que adicionava automaticamente o visitante como amigo do criador do

worm e se replicava nos perfis dos amigos. Um worm, em termos de cibersegurança, é um programa que se replica e se espalha por uma rede, explorando vulnerabilidades e causando danos sem a necessidade de um arquivo hospedeiro (Bhargava; Choudhary; Gupta, 2022).

A maneira como o Cross-Site Scripting (XSS) opera é notável devido à sua dependência quase exclusiva do JavaScript. Qualquer página da web que faça uso de JavaScript e não tenha medidas de segurança adequadas está potencialmente vulnerável a XSS (Ferreira, 2017). Com isso, surgem várias variantes do Cross-Site Scripting, nomeadamente o XSS Armazenado, o XSS Refletido e o XSS Baseado em DOM.

Quadro 1 - Definindo tipos de Ataque de XSS

Tipos de XSS	**Descrição**
XSS Armazenado	Também conhecido como XSS Persistente, representa uma ameaça significativa para aplicações web, pois permanece persistente nas páginas afetadas. Em uma plataforma de rede social, um ataque XSS armazenado em uma publicação pode comprometer todos os usuários que visualizaram tal conteúdo.
XSS Refletido	Nesse caso, caracteriza-se por URLs manipuladas que, ao serem acessadas pela vítima, desencadeiam o ataque diretamente no navegador. O código malicioso é refletido de volta para o usuário.
XSS Baseado em DOM	Ocorre no lado do cliente e é menos detectável pelos desenvolvedores, dado que a carga maliciosa (payload) não transita pelo servidor. Este tipo de ataque explora a manipulação do Modelo de Objeto de Documento (DOM) para executar scripts prejudiciais sem o

	conhecimento do usuário.

Fonte: Adaptado de Ferreira, 2017.

Por outro lado, a falha A08:2021-*Software and Data Integrity Failures* (Falhas de software e integridade de dados), destaca que nenhuma vulnerabilidade deve ser subestimada ou considerada irrelevante, independentemente do nível de criticidade atribuído. Nesse contexto, destaca-se a vulnerabilidade CSRF (*Cross-Site Request Forgery*), que permite a um invasor manipular o navegador de um usuário. Isso pode levar à execução de ações não autorizadas em aplicações web nas quais o usuário esteja autenticado (Agrawal, 2023).

Essas ações não intencionais podem ser executadas de diversas maneiras, entre elas, por meio do uso de cookies. Cookies são informações enviadas pela aplicação ao navegador do usuário. Quando esse tipo de ataque ocorre, há uma manipulação dos cookies em benefício do atacante.

Observa-se, portanto, a emergência de um problema: a percepção de que os cookies não constituem um meio seguro para armazenar dados sensíveis, como senhas e informações pessoais. Em resposta a essa limitação, surge o conceito de sessão (*session*), uma área de memória utilizada para o armazenamento de informações. A principal diferença entre cookies e sessões reside no local de armazenamento dos dados. Enquanto os cookies são armazenados no navegador do usuário, as informações da sessão são mantidas no servidor, conferindo um nível de segurança superior a esse recurso (Ferreira, 2017).

2.3 TESTE DE SEGURANÇA

Teste de penetração, também conhecido como *pentest*, teste de invasão legítimo ou *ethical hacking*, é uma prática autorizada por ambas as partes envolvidas, geralmente formalizada por um contrato. Esse contrato concede ao hacker ético o direito de analisar, identificar e explorar todas as vulnerabilidades presentes no sistema, com o objetivo de melhorar sua segurança.

Esses testes de segurança não se limitam a procurar vulnerabilidades diretamente na aplicação principal. Eles também investigam o ambiente ao redor,

incluindo subdomínios relacionados. Isso é crucial porque, mesmo que a aplicação principal esteja bem protegida, vulnerabilidades graves ou críticas em subdomínios podem servir como ponto de entrada, comprometendo a segurança do domínio principal.

Este fenômeno ocorre devido à negligência de algumas etapas no desenvolvimento de software. Estatísticas indicam que apenas 10 a 60% do esforço total de um projeto é dedicado aos testes (López-Martín, 2021). Diante disso, emergem duas abordagens distintas para o pentester realizar o processo de testes de software: a Caixa Branca (*White Box*) e a Caixa Preta (*Black Box*).

Cada uma dessas abordagens possui características distintas. Por exemplo, o teste de caixa preta reflete mais fielmente a realidade enfrentada pelos profissionais de cibersegurança. Nele, o profissional simula um ataque externo à organização sem acesso a informações privilegiadas, com o objetivo de obter acesso confidencial. Este método é vantajoso porque permite identificar falhas na proteção e, de forma crítica, avaliar a eficácia da equipe de resposta a incidentes (Shravan, Neha e Pawan, 2014).

Por outro lado, o teste de caixa branca é uma abordagem onde o profissional tem acesso completo às informações da aplicação a ser testada, incluindo código-fonte, infraestrutura, e credenciais de acesso, entre outros detalhes. Esta abordagem simula um cenário em que o atacante possui acesso interno privilegiado. A principal vantagem desse método é que ele permite uma análise mais profunda e detalhada do sistema, facilitando a identificação precisa de vulnerabilidades e proporcionando um *feedback* mais preciso sobre as falhas detectadas (Shravan, Neha e Pawan, 2014).

No entanto, surge uma terceira via: a Caixa Cinza (*Gray Box*), que combina os princípios da Caixa Branca e da Caixa Preta. Nessa abordagem, o pentester possui conhecimento limitado sobre o ambiente interno do sistema, como credenciais básicas ou diagramas de arquitetura. Isso permite uma análise mais aprofundada das vulnerabilidades presentes, sem comprometer totalmente o realismo do ataque (Shravan, Neha e Pawan, 2014).

Entretanto, o teste de penetração vai além da mera aplicação de ferramentas de *hacking*. Antes da execução prática, etapas preliminares cruciais definem o

sucesso do projeto. Nesse contexto, surgem metodologias estruturadas como o ZEH (*Zero Entry Hacking*), que guiam o pentester desde o planejamento até a execução do teste. Cada fase do processo ZEH avança progressivamente, permitindo a identificação precisa de vulnerabilidades e proporcionando orientação ao responsável pela atividade.

A metodologia ZEH é divida em:

A parte inicial da metodologia é a fase preparatória passiva, também chamada de Reconhecimento, na qual não há interação direta com o objeto de estudo e ativa que interage diretamente com o alvo por qualquer meio. Nessa etapa, o objetivo é analisar e coletar informações relevantes de diversas fontes disponíveis (Moreno, 2015).

A Varredura é uma etapa pré-ataque crucial, onde as informações coletadas anteriormente são analisadas em maior profundidade para identificar hosts (*servidores*) ativos, portas abertas, serviços em execução e vulnerabilidades na rede (Moreno, 2015).

A fase de Exploração concentra-se em obter acesso ao sistema. É nesse estágio que todo o conhecimento do *hacker* é aplicado. As vulnerabilidades descobertas nas fases de reconhecimento e varredura são exploradas para, de qualquer forma, ganhar acesso(Moreno, 2015). Alguns desses ataques incluem *SQL Injection* (Injeção de SQL), *Session Hijacking* (Sequestro de Sessão), CSRF (*Cross-Site Request Forgery*), *Cross-Site Scripting* (XSS) e outros.

Após obter acesso inicial, o invasor busca garantir persistência no sistema, visando explorá-lo e realizar ataques futuros. Essa etapa é conhecida como a fase de "Mantendo acesso" ou “Pós-Exploração”. Em alguns casos, os hackers reforçam os sistemas e asseguram o acesso futuro por meio da instalação de malware, frequentemente um backdoor ou rootkit.Para garantir o êxito dessa fase, o invasor também elimina todos os vestígios deixados durante o ataque (Moreno, 2015).

Todo esse trabalho é realizado pelo *pentester*, o hacker ético responsável pelo teste de penetração. Suas atividades, incluindo as ferramentas utilizadas, métodos aplicados e identificação de ameaças potenciais, são detalhadamente documentadas. Esse relatório serve como a Prova de Conceito (*Proof of Concept*),

cujo objetivo é apresentar uma visão geral do ataque, demonstrando as vulnerabilidades encontradas e seu impacto potencial.

3 METODOLOGIA

Este trabalho se configura como uma investigação qualitativa de cunho aplicado, amparada nas metodologias de pesquisa bibliográfica e estudo de caso. A busca bibliográfica se deu por meio de livros, artigos, periódicos e monografias acessíveis em repositórios digitais como Google Acadêmico, Base de Dados de Periódicos da CAPES e *IEEE Xplore*. A seleção das fontes de interesse ocorreu com base em critérios de inclusão e exclusão, sendo o critério de exclusão a data de publicação, enquanto os de inclusão consideraram pesquisas publicadas nos últimos 10 anos, trabalhos similares e abordagem do tema de interesse.

A principal vantagem da pesquisa bibliográfica está na possibilidade de o investigador abranger uma variedade muito maior de fenômenos do que seria possível pesquisar diretamente. Essa vantagem é especialmente relevante quando o problema de pesquisa exige dados amplamente dispersos geograficamente (Gil, 2022). Na pesquisa bibliográfica foram buscados artigos publicados a partir de 2014 com base nos seguintes descritores: “*Pentest*, Segurança da informação e Vulnerabilidades *Web*”. Foram obtidos 48 resultados, dos quais 22 foram selecionados para a elaboração da revisão teórica, de acordo com os critérios estabelecidos de inclusão e exclusão.

Considerando o objetivo deste estudo, que é identificar as estratégias e técnicas para comprometer a segurança da informação em uma instituição de ensino, a pesquisa é classificada como exploratória. Em estudos exploratórios, analíticos ou descritivos, uma abordagem frequentemente utilizada é o Estudo de Caso (EC), onde um caso representa um evento ou fenômeno específico em análise (Figueiredo, 2020).Para a realização do EC, selecionamos como objeto de análise de vulnerabilidade uma aplicação web amplamente utilizada, o SARDES. Além da aplicação em si, o servidor de hospedagem também foi incluído no escopo da análise.

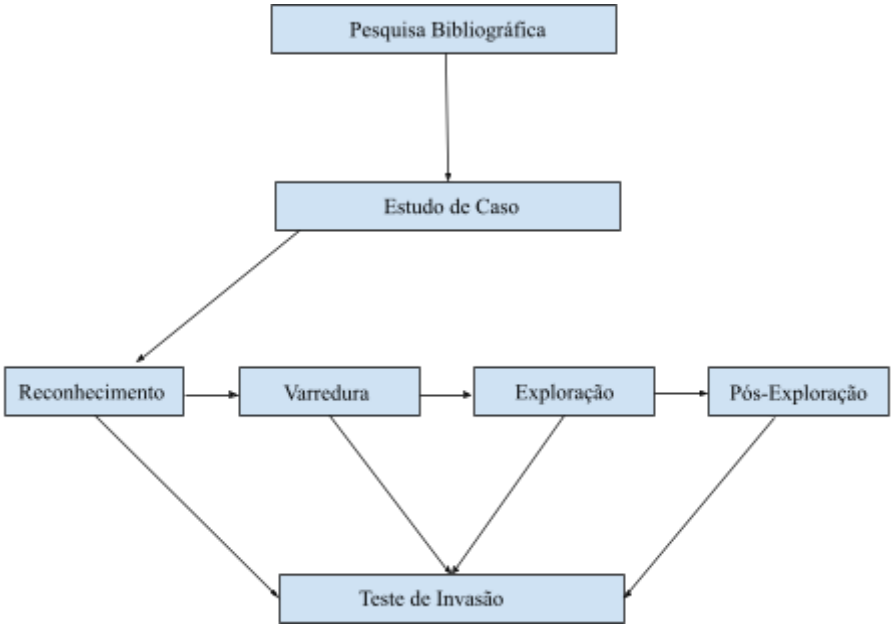
Para a realização do teste de penetração, optou-se pela abordagem do tipo “Caixa Preta” (Black Box), que consiste em analisar o sistema sem informações

prévias sobre sua estrutura interna ou código-fonte. Além disso, foram utilizadas ferramentas especializadas para avaliar as vulnerabilidades do sistema, como **Whois** e **Dig** (*DNS Lookup*) para obter informações sobre domínios e servidores, **Nmap** para mapear portas e serviços em execução, e o **Burp Suite** para realizar análises detalhadas de requisições HTTP e possíveis falhas de segurança, como injeções de código e XSS. Essa combinação de técnicas e ferramentas permitiu uma avaliação abrangente e precisa das potenciais vulnerabilidades do sistema.

No contexto deste trabalho, a abordagem "Caixa Preta" implica na análise da aplicação web SARDES unicamente com base nos resultados obtidos pelas ferramentas de varredura. Ou seja, o pesquisador não possui conhecimento prévio sobre o cenário da aplicação, incluindo tipo de sistema operacional, servidor de hospedagem, tecnologias empregadas, endereços IP e demais informações relevantes.

O projeto de pesquisa propõe o seguinte fluxo de trabalho para o desenvolvimento do trabalho até o seu final, conforme esquema apresentado na Figura 2.

Figura 2 - Fluxo de trabalho



Fonte: Elaborado pelo autor, 2024.

Conforme demonstrado na Figura 2, a metodologia segue um fluxo estruturado que inicia com a Pesquisa Bibliográfica e se desdobra em um Estudo de Caso, abordando as etapas fundamentais de um Teste de Invasão, que incluem: **Reconhecimento**, **Varredura**, **Exploração** e **Pós-Exploração**. Esse processo garante uma abordagem sistemática e completa na análise de segurança.

4 RESULTADOS

Os resultados do estudo de caso indicam que ambientes digitais podem estar suscetíveis a vulnerabilidades, dependendo de suas configurações de segurança e práticas de proteção, variando desde falhas menos impactantes até ameaças críticas que podem comprometer completamente os pilares da segurança da informação. Utilizando ferramentas amplamente acessíveis e gratuitas na web, é possível obter informações detalhadas sobre o alvo a ser analisado. Ferramentas como o "**Whois**" e o "**Registro.br**" permitem acessar dados relevantes sobre domínios, conforme descrito a seguir:

% Copyright (c) Nic.br

% The use of the data below is only permitted as described in

% full by the Use and Privacy Policy at https://registro.br/upp ,

% being prohibited its distribution, commercialization or

% reproduction, in particular, to use it for advertising or

% any similar purpose.

% 2024-09-05T19:59:49-03:00 - IP: 2804:460c:862f:2900:204:4df7:e9bc:afcc

domain: ifpi.edu.br

owner: INSTITUTO FEDERAL DE EDUC, CIENCIA E TEC DO PIAUI

ownerid: 10.806.496/0001-49

responsible: Paulo Borges da Cunha

country: BR

owner-c: PABCU20

tech-c: DERTE12

nserver: ns1.p201.dns.oraclecloud.net

nsstat: 20241122 AA

nslastaa: 20241122

nserver: ns2.p201.dns.oraclecloud.net

nsstat: 20241122 AA

nslastaa: 20241122

nserver: ns3.p201.dns.oraclecloud.net

nsstat: 20241122 AA

nslastaa: 20241122

nserver: ns4.p201.dns.oraclecloud.net

nsstat: 20241122 AA

nslastaa: 20241122

created: 20090415 #5463341

changed: 20240419

status: published

nic-hdl-br: PABCU20

person: PAULO BORGES DA CUNHA

e-mail: dti@ifpi.edu.br

country: BR

created: 20230302

changed: 20230302

nic-hdl-br: DERTE12

person: DEPARTAMENTO DE REDES E TELECOMUNICAÇÕES

e-mail: DRT.REITORIA@ifpi.edu.br

country: BR

created: 20210318

changed: 20210318

% Security and mail abuse issues should also be addressed to

% cert.br, http://www.cert.br/ , respectivelly to cert@cert.br

% and mail-abuse@cert.br

Informações relevantes para análise:

Responsável: Paulo Borges da Cunha

Servidores DNS: ns1.p201.dns.oraclecloud.net, ns2.p201.dns.oraclecloud.net, ns3.p201.dns.oraclecloud.net, ns4.p201.dns.oraclecloud.net

Email: DRT.REITORIA@ifpi.edu.br

Tais dados são bastante importantes pois a identificação daquela pessoa responsável pelo domínio proporciona o uso de engenharia social - na forma do *phishing* - para ludibriar e conseguir uma posse valiosa de informações. Os servidores DNS vinculados ao domínio podem ser investigados também à busca de eventuais vulnerabilidades de configuração, o que tornaria possível uma série de ataques, como, por exemplo, o DNS *Spoofing*. Além disso, o e-mail técnico do domínio pode ser um alvo direto a ataques de *phishing* ou ataques de força bruta que abrirão brechas comunicativas e mostrarão novas avenidas para acessos não autorizados.

Com as informações coletadas, foram iniciadas as primeiras tentativas de intrusão, começando pelo servidor DNS. O objetivo era verificar se existiam falhas ou erros de configuração que pudessem ser explorados. Ao realizar testes específicos, como a tentativa de transferência de zona DNS, ficou evidente que o servidor estava configurado para bloquear essa ação, garantindo uma camada importante de segurança nessa parte do sistema.

Quadro 2 - Resultado da análise da ferramenta “nslookup”

Dominio	Tempo de Vida (Time to Live) em segundos	Classe do Registro DNS	Tipo	Detalhes
ifpi.edu.br.	300	IN (Internet)	SOA (Start of Authority ou Servidor autoritativo)	ns1.p201.dns.oraclecloud.net. hostmaster.ifpi.edu.br. 2024032754 3600 600 604800 1800
ifpi.edu.br.	300	IN	SOA	ns1.p201.dns.oraclecloud.net. hostmaster.2024032754 3600 600 604800 1800
ifpi.edu.br.	86400	IN	NS (Name Server)	ns1.p201.dns.oraclecloud.net.
ifpi.edu.br.	86400	IN	NS	ns3.p201.dns.oraclecloud.net.

ifpi.edu.br.	86400	IN	NS	ns2.p201.dns.oraclecloud.net.
ifpi.edu.br.	86400	IN	NS	ns4.p201.dns.oraclecloud.net.
ifpi.edu.br.	3600	IN	MX (Mail Exchange)	10 alt4.aspmx.l.google.com.
ifpi.edu.br.	3600	IN	MX	5 alt2.aspmx.l.google.com.
ifpi.edu.br.	3600	IN	MX	5 alt1.aspmx.l.google.com.
ifpi.edu.br.	3600	IN	MX	1 aspmx.l.google.com.
ifpi.edu.br.	3600	IN	CAA (Certificate AUthority Authorization)	0 issue "amazonaws.com"
ifpi.edu.br.	3600	IN	CAA	0 issue "awstrust.com"

ifpi.edu.br.	3600	IN	CAA	0 issue "globalsign.com"
ifpi.edu.br.	3600	IN	CAA	0 issue "letsencrypt.org"
ifpi.edu.br.	3600	IN	CAA	0 issue "serpro.gov.br"
ifpi.edu.br.	3600	IN	CAA	0 issue "certificados.serpro.gov.br"
ifpi.edu.br.	3600	IN	CAA	"_globalsign-domain-verification=guQa23AF99a5Cnn52qGmayt6TDfGFyLd0bLHdIjSpw"
ifpi.edu.br.	3600	IN	TXT (Contém dados arbitrários de autenticação)	"v=spf1 ip4:170.245.32.130 include:_spf.google.com ~all"
ifpi.edu.br.	600	IN	A (Endereço IPv4 Associado)	168.75.76.122

Fonte: Autor, 2024.

Uma etapa crucial na análise de segurança é verificar a existência de portas abertas e identificar os serviços em execução. Essa verificação permite avaliar se os serviços ativos estão devidamente atualizados e, também, se há algum serviço rodando em portas não utilizadas, que podem representar potenciais vulnerabilidades devido a falhas de configuração ou falta de manutenção.

Starting Nmap 7.95 (https://nmap.org) at 2024-09-06 21:25 -03

Nmap scan report for sardes.ifpi.edu.br (129.148.48.168)

Host is up (0.060s latency).

Not shown: 65533 filtered tcp ports (no-response)

PORT STATE SERVICE

80/tcp open http

443/tcp open https

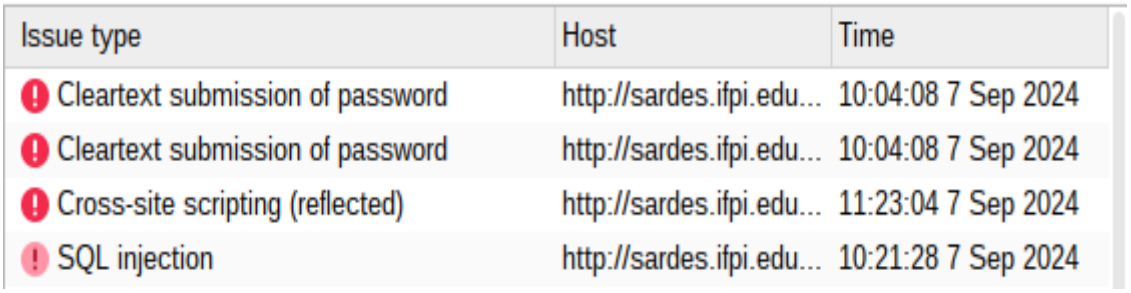
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port

OS fingerprint not ideal because: Missing a closed TCP port so results incomplete

No OS matches for host

Também foi utilizado o **Burp Suite** que é uma ferramenta de teste de segurança para aplicações web, usada para identificar vulnerabilidades como injeções, XSS e falhas de autenticação. Foram destacadas três categorias de vulnerabilidades específicas, selecionadas devido ao seu alto nível de criticidade. A partir dessas vulnerabilidades, desenvolve-se a análise que segue, abordando desde a natureza e os impactos de cada tipo até as melhores práticas para mitigação.

Figura 3 - Vulnerabilidades encontradas pelo Burp Suite



Issue type	Host	Time
Cleartext submission of password	http://sardes.ifpi.edu...	10:04:08 7 Sep 2024
Cleartext submission of password	http://sardes.ifpi.edu...	10:04:08 7 Sep 2024
Cross-site scripting (reflected)	http://sardes.ifpi.edu...	11:23:04 7 Sep 2024
SQL injection	http://sardes.ifpi.edu...	10:21:28 7 Sep 2024

Fonte: Autor, 2024.

A Figura 3 apresenta um resumo das vulnerabilidades detectadas na plataforma bibliotecária SARDES durante o teste de segurança realizado com a ferramenta *Burp Suite*. Essas vulnerabilidades incluem: envio de senha em texto claro (***Cleartext submission of password***), ***cross-site scripting*** (XSS - reflected) e **injeção SQL** (*SQL Injection*).

O ***Cleartext submission of password*** é uma falha de segurança que acontece quando senhas são enviadas sem qualquer tipo de criptografia, em texto simples (Blanchard, 2022). Isso expõe as credenciais dos usuários em redes não seguras, facilitando ataques de interceptação, como o *Man-in-the-Middle*, e possibilitando o roubo de credenciais e acessos não autorizados. Além disso, o uso de sniffers, ferramentas que monitoram e registram o tráfego de rede, permite que invasores capturem esses dados durante a requisição, potencializando o risco de vazamento de dados do usuário.

Cross-site scripting (XSS) reflected*,* ocorre quando a aplicação web reflete entradas maliciosas fornecidas pelo usuário em respostas sem a devida sanitização, permitindo que scripts sejam executados no navegador da vítima (Silva, 2024). Quando um script malicioso é injetado e executado no navegador da vítima, o invasor pode obter acesso a informações sensíveis, como *cookies*, sessões autenticadas e credenciais. Além disso, o XSS pode ser explorado para redirecionar a vítima a sites fraudulentos, expondo-a a ataques de *phishing* ou coleta de dados.

O ***SQL injection*** é uma vulnerabilidade que ocorre quando as entradas fornecidas pelo usuário não são devidamente sanitizadas, permitindo que consultas SQL maliciosas sejam executadas diretamente no banco de dados (Nasereddin, et

al., 2023). Com essa liberdade de manipulação, um atacante pode realizar diversas ações prejudiciais, como acessar informações confidenciais e executar comandos administrativos no servidor, comprometendo a integridade, confidencialidade e disponibilidade dos dados.

No Quadro 3, são apresentados os riscos associados a cada vulnerabilidade identificada na aplicação analisada, destacando os perigos que cada uma representa para o usuário e as medidas recomendadas para sua mitigação.

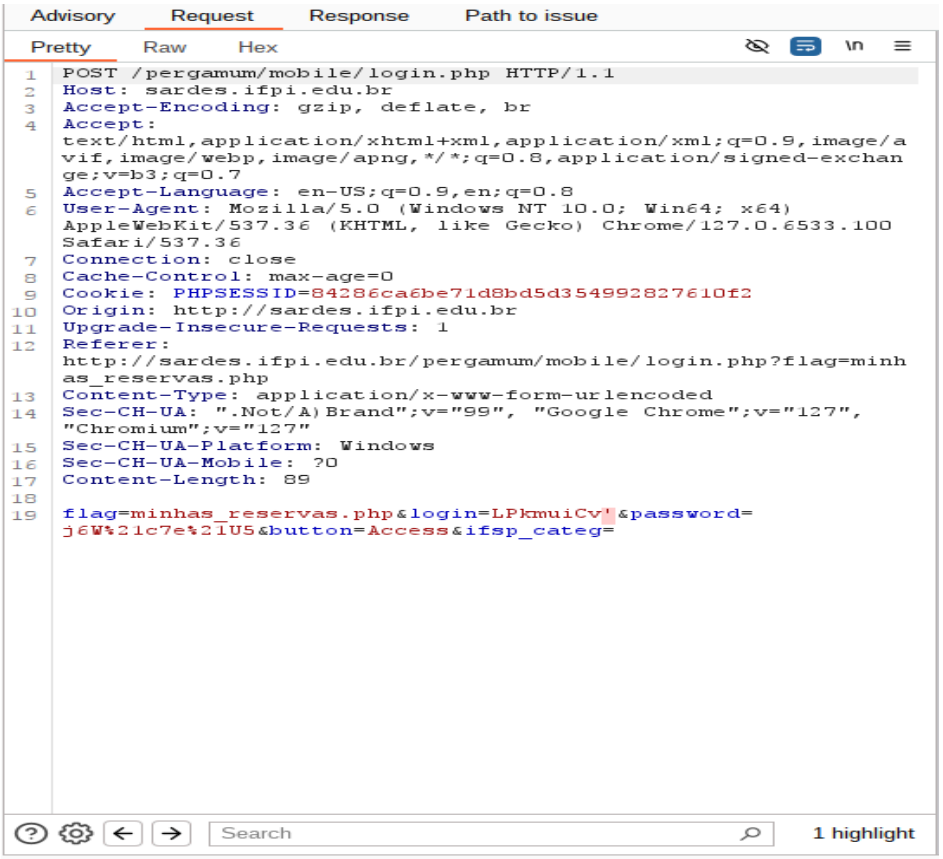
Quadro 3 - Tipos de vulnerabilidades encontradas

Vulnerabilidade	Riscos	Mitigação
Cleartext Submission of Password	Exposição das credenciais de usuários em redes não seguras, possibilitando ataques de interceptação (Man-in-the-Middle), roubo de credenciais e acesso não autorizado.	Implementar criptografia como SSL/TLS para garantir que as comunicações sejam transmitidas de forma segura. Além disso, forçar o uso de HTTPS em todas as páginas que lidam com senhas ou informações sensíveis
Cross-Site Scripting (XSS) Reflected	Execução de scripts maliciosos no navegador do usuário, podendo roubar cookies, sessões, credenciais ou redirecionar o usuário para sites maliciosos.	Validar e sanitizar todas as entradas do usuário, escapando caracteres especiais e utilizando bibliotecas seguras para impedir a execução de scripts não autorizados.

SQL Injection	Manipulação ou exclusão de dados no banco de dados, obtenção de informações confidenciais, controle total do sistema ou execução de comandos administrativos no servidor.	Utilizar consultas parametrizadas (Prepared Statements) para impedir a execução de SQL arbitrário. Além disso, aplicar mecanismos de filtragem e validação nas entradas de usuário.

Fonte: Adaptado de Ferreira (2017).

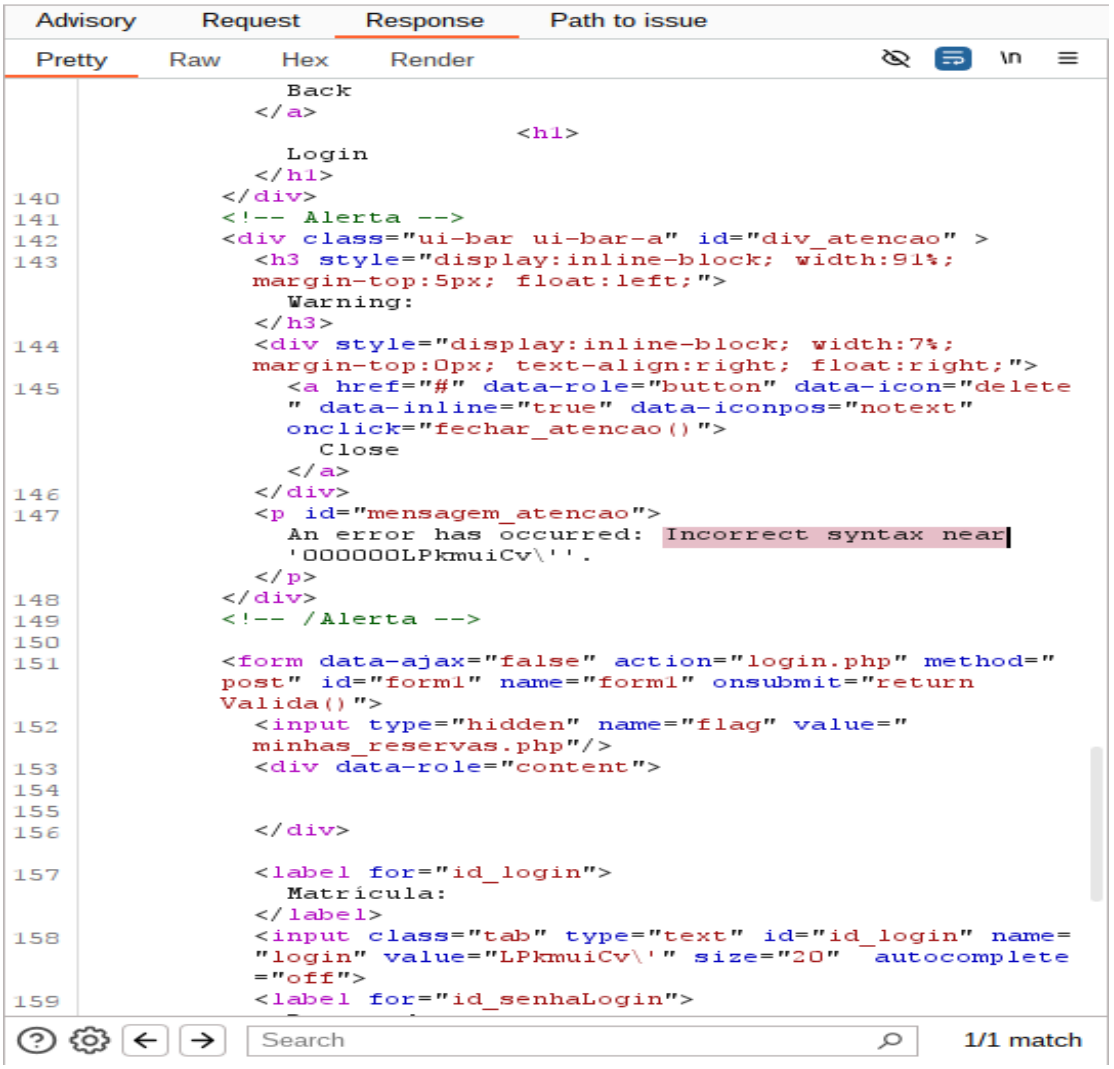
Figura 4 - Requisição (request) realizada pelo Burp Suite.



Fonte: Autor, 2024.

Conforme mostrado na Figura 4, a requisição capturada demonstra uma tentativa de acesso ao sistema por meio da URL de login, com um ataque de *SQL Injection* no parâmetro "login". O objetivo é avaliar o retorno da aplicação e verificar se essa entrada vulnerável permite a exploração da base de dados ou outros acessos não autorizados, identificando, assim, a presença de vulnerabilidades que comprometam a segurança do sistema (Nasereddin, et al., 2023).

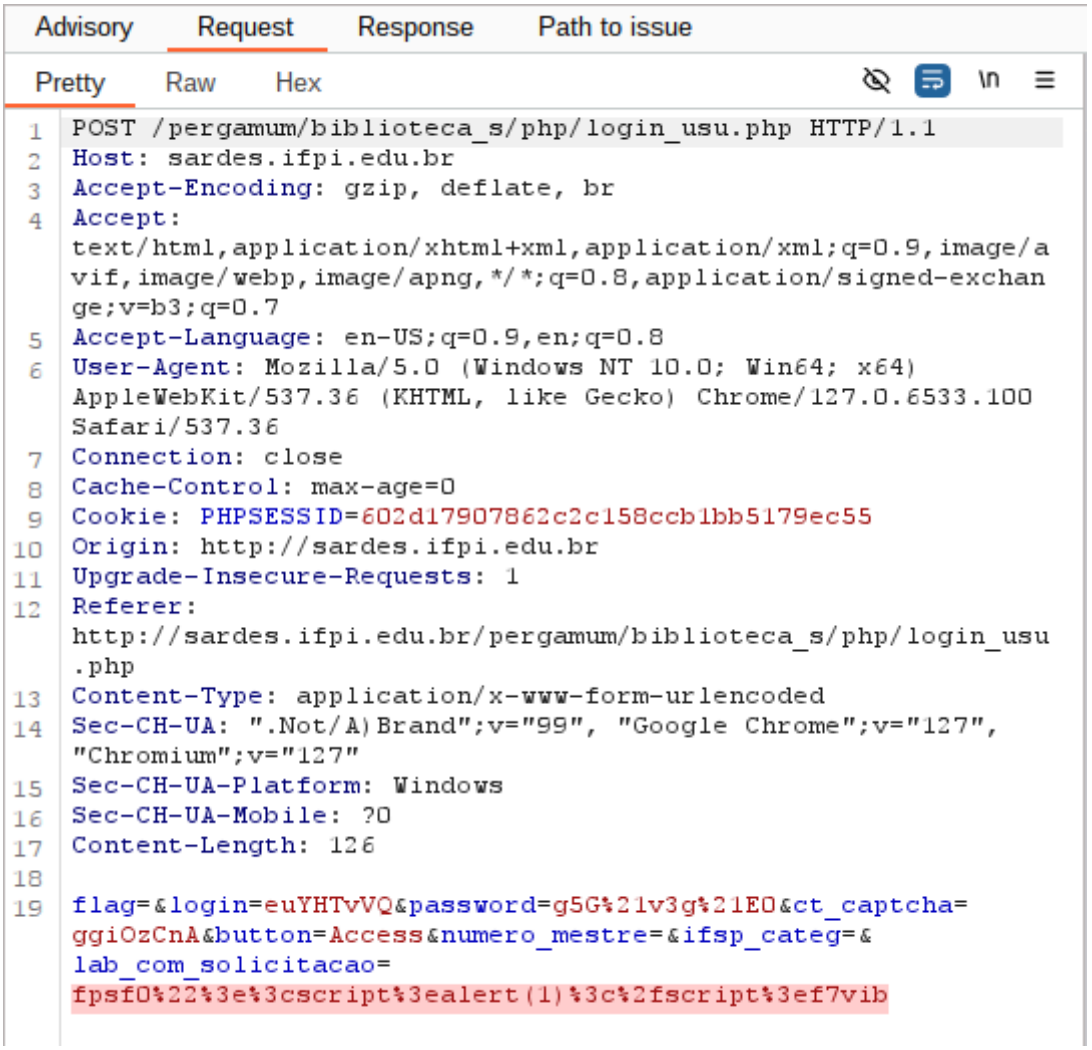
Figura 5 - Resposta (response) realizada pelo Burp Suite.



Fonte: Autor, 2024.

A Figura 5 apresenta a resposta do sistema a uma tentativa de SQL Injection, evidenciada pela mensagem de erro retornada: "Incorrect syntax near '0000D0LKPxmuiCv'". Esse retorno revela uma falha de segurança, pois a aplicação não valida adequadamente as entradas do usuário, permitindo que comandos SQL não tratados sejam inseridos na consulta ao banco de dados. Essa vulnerabilidade possibilita que um atacante manipule as consultas SQL, potencialmente acessando dados confidenciais ou comprometendo a integridade do sistema.

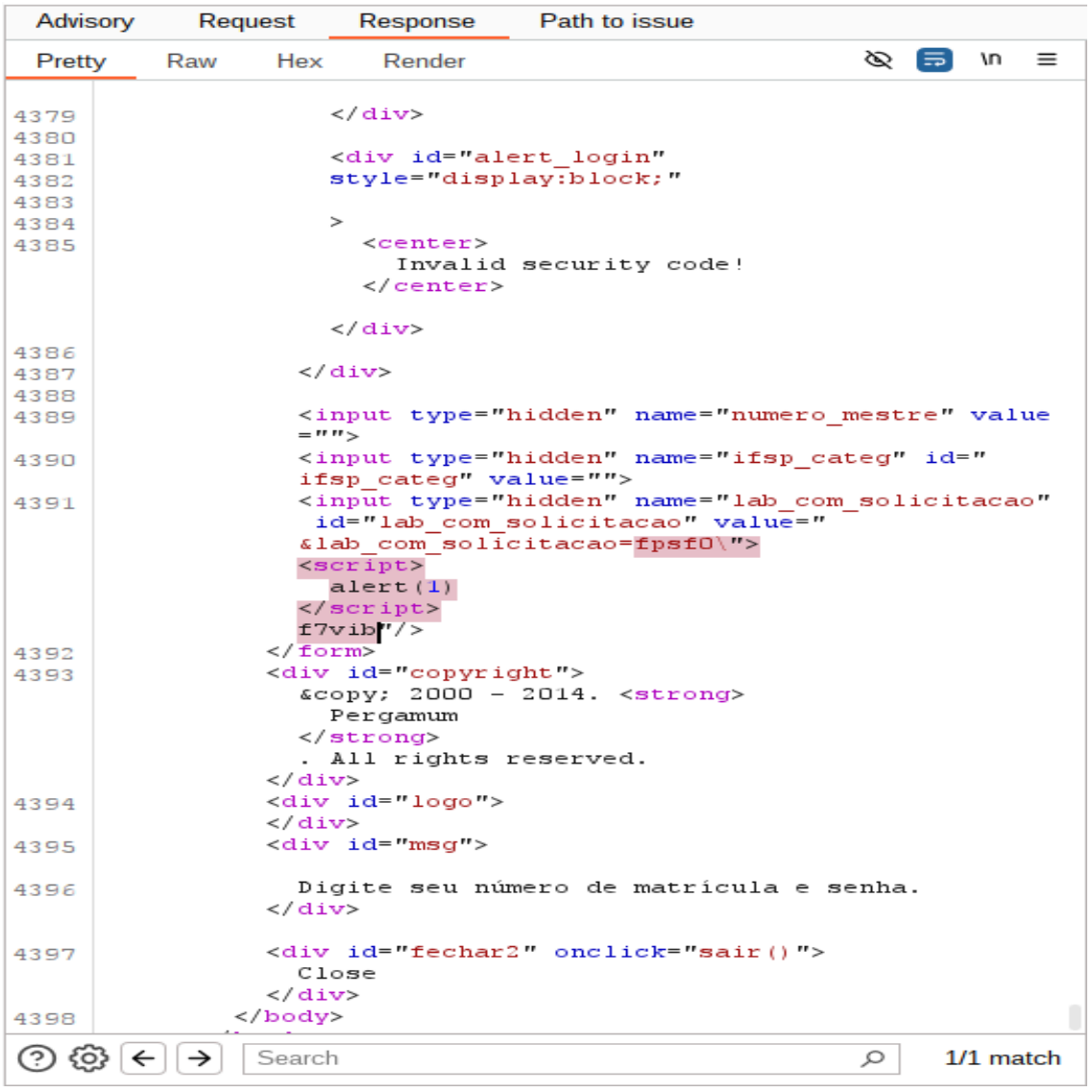
Figura 6 - Requisição (request) realizada pelo Burp Suite.



Fonte: Autor, 2024.

A Figura 6 apresenta uma requisição realizada pelo Burp Suite, que ilustra uma tentativa de ataque do tipo XSS (Cross-Site Scripting) no parâmetro “lab_com_solicitacao” da URL de login. O propósito dessa abordagem é verificar se o sistema é vulnerável a esse tipo de exploração. Para isso, foi inserido um código malicioso que, caso seja refletido na resposta sem a devida sanitização, confirmará a presença da vulnerabilidade. Esse teste é fundamental para avaliar a robustez do sistema contra ataques de injeção de scripts maliciosos.

Figura 7 - Resposta (response) realizada pelo Burp Suite.



Fonte: Autor, 2024.

A Figura 7 apresenta a resposta capturada pelo Burp Suite, demonstrando uma vulnerabilidade de XSS refletido. A ausência de sanitização adequada das entradas dos usuários permitiu que o script malicioso fosse refletido e executado no navegador da vítima. Na resposta exibida, podemos observar que o código malicioso, representado pelo script de alerta (alert(1)), foi injetado e executado com sucesso, evidenciando a falha. Esse comportamento destaca a necessidade de implementar mecanismos robustos de validação e sanitização para proteger o sistema contra ataques desse tipo.

Com base nos resultados apresentados, fica claro que a plataforma analisada possui vulnerabilidades críticas que podem ser exploradas, colocando em risco a **confiabilidade**, **integridade** e **disponibilidade** das informações. Entre as falhas identificadas, destacam-se o **envio de senhas em texto claro**, **Cross-Site Scripting (XSS)** e **SQL Injection**, que representam ameaças significativas, como o roubo de credenciais, a execução de comandos maliciosos e o comprometimento da base de dados.

5 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Diante da pesquisa realizada, foi observado que a plataforma SARDES apresenta um nível de segurança abaixo do esperado, conforme demonstrado pelos testes realizados, que identificaram vulnerabilidades comuns e críticas em aplicações web. É amplamente conhecido na área de segurança da informação que uma vulnerabilidade de XSS refletido. A ausência de sanitização adequada das entradas dos usuários permitiu que o script malicioso fosse refletido e executado no navegador da vítima. Na resposta exibida, podemos observar que o código malicioso, representado pelo script de alerta (alert(1)), foi injetado e executado com sucesso, evidenciando a falha. Esse comportamento destaca a necessidade de implementar mecanismos robustos de validação e sanitização para proteger o sistema contra ataques desse tipo.nenhum sistema é totalmente seguro; sempre haverá um ponto passível de exploração, seja por falhas tecnológicas, físicas ou até mesmo por meio de interações humanas, como no caso da engenharia social.

Os testes realizados obtiveram sucesso em sua maioria, evidenciando a presença de vulnerabilidades significativas na plataforma analisada. As falhas identificadas confirmam que os métodos aplicados foram eficazes em explorar

pontos críticos, reforçando a necessidade de melhorias na segurança para mitigar possíveis ataques e proteger os dados de forma mais robusta.

Podemos concluir que o pentest é uma etapa fundamental e de grande impacto na segurança de uma aplicação web, não podendo ser negligenciado de forma alguma. Sua realização é essencial para identificar e corrigir vulnerabilidades, garantindo a proteção dos dados e fortalecendo a segurança do sistema contra possíveis ameaças.

Para trabalhos futuros, seria interessante utilizar uma gama mais ampla de ferramentas especializadas, com destaque para o sistema operacional Kali Linux, amplamente reconhecido na área de segurança cibernética. Esse sistema oferece um conjunto robusto de ferramentas integradas voltadas para análise de vulnerabilidades e execução de testes de intrusão.

REFERÊNCIAS

ABNT ISO 27002. (2013). ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. NBR/ISO/IEC 27002:2013 tecnologia da informação – técnicas de segurança – código de prática para controles de segurança da informação.

AGRAWAL, S. Mitigating Cross-Site Request Forgery (CSRF) Attacks Using Reinforcement Learning and Predictive Analytics. **Applied Research in Artificial Intelligence and Cloud Computing**, v. 6, n. 9, p. 17-30, 2023.

ALGHAWAZI, M.; ALGHAZZAWI, D.; ALARIFI, S.. Detection of sql injection attack using machine learning techniques: a systematic literature review. **Journal of Cybersecurity and Privacy**, v. 2, n. 4, p. 764-777, 2022.

AYDOS, M. *et al*. Security testing of web applications: A systematic mapping of the literature. **Journal of King Saud University-Computer and Information Sciences**, v. 34, n. 9, p. 6775-6792, 2022.

BLANCHARD, Enka. Client-side hashing for efficient typo-tolerant password checkers. **International Journal of Systems and Software Security and Protection (IJSSSP)**, v. 13, n. 1, p. 1-24, 2022.

SANTOS, E. E.; SOARES, T. M. M. K. **Riscos, ameaças e vulnerabilidades: o impacto da segurança da informação nas organizações**. 2018.

FERREIRA, R. **Segurança em Aplicações Web**. 1ª Ed. São Paulo. Casa do Código, 2017. 156p.

FIGUEIREDO, D. A. et al. Vulnerabilidades em redes Wi-Fi de instituições de ensino superior: um estudo de múltiplos casos. **Research, Society and Development**, v. 9, n. 2, p. e178921979-e178921979, 2020.

GALEGALE, N.; FONTES, E.L.G; GALEGALE; B.P. Uma contribuição para a segurança da informação: um estudo de casos múltiplos com organizações brasileiras. **Perspectivas em Ciência da Informação**, Minas Gerais, v. 22, p. 75-97, jul.. 2017.

GIL, A. **Métodos e técnicas de pesquisa social**. São Paulo: Atlas. 7ª. Edição, 2022. 208 p.

JYOTHI, R. Naga Sravana et al. Protection and Saving of Delicate Data by using Cloud Computing. In: **2022 International Conference on Electronics and Renewable Systems (ICEARS)**. IEEE, 2022. p. 1660-1667.

MARTINS, Jonathan dos Santos. **EXPLORAÇÃO DE VULNERABILIDADES EM REDES IOT**, 2018

MORENO, Daniel. **Introdução ao Pentest**. Novatec Editora, 2015.

KUNCORO, A.W; FAYRUZ RAHMA S.T; ENG, M. Analisis Metode Open Web Application Security Project (OWASP) pada Pengujian Keamanan Website: Literature Review. **Automata**, v. 3, n. 1, 2022.

LIU, F.; *et al*. Privacy-preserving scanning of big content for sensitive data exposure with MapReduce. In: **Proceedings of the 5th ACM Conference on Data and Application Security and Privacy**. 2015. p. 195-206.

LÓPEZ-MARTÍN, Cuauhtémoc. Effort prediction for the software project construction phase. **Journal of Software: Evolution and Process**, v. 33, n. 7, p. e2365, 2021.

MAIA, E. Ataques hackers aumentam 8,8% no Brasil e país segue como 2º mais atacado do mundo. **CNN Brasil**, Brasília, 03, maio, 2024. Disponível em: <https://www.cnnbrasil.com.br/nacional/ataques-hackers-aumentam-88-no-brasil-e-pais-segue-como-2o-mais-atacado-do-mundo/>. Acesso em: 13 de maio 2024

NAGLI, L.S.D. Pandemia na pandemia: a escalada de ataques cibernéticos pós COVID-19 Pandemic in pandemic: the climbing of post COVID-19 cyber attacks. **Brazilian Journal of Development**, v. 8, n. 4, p. 28482-28493, 2022.

NASEREDDIN, Mohammed et al. A systematic review of detection and prevention techniques of SQL injection attacks. **Information Security Journal: A Global Perspective**, v. 32, n. 4, p. 252-265, 2023.

OWASP. OWASP Top 10 Vulnerabilities. Disponível em: <https://owasp.org/www-project-top-ten/>. Acesso em: 06 jun. 2024.

RAMOS, R.G.G. *et al*. Aplicação da segurança da informação e LGPD para a experiência e usabilidade dos usuários em aplicativos móveis. **Revista Sociedade Científica**, v. 7, n. 1, p. 1717-1738, 2024.

SANCHES, T. *et al*. A informação tem valor. 2022.

SAVELIEV, Denys. Definition the list of threats of security of web systems and applications at the development stage. **ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ І БЕЗПЕКА**, p. 135, 2020.

Shravan, K.; Neha, B.; Pawan, B.(2014).**Penetration testing: A Review.** Compusoft, Faridabad v. 3, n. 4, p. 752–7.

SILVA, Lidia Paula de Oliveira. Pentest baseado em imagens encase: uma análise de vulnerabilidades em servidores web. 2024.

SOUSA, J.E.J. **Análise de segurança cibernética no Tribunal de Contas do Estado do Rio Grande do Norte: aplicação dos princípios OWASP na identificação e mitigação de vulnerabilidades**. 2024. Trabalho de Conclusão de Curso. Universidade Federal do Rio Grande do Norte.

SWAPNA, P. et al. Hybrid Cryptosystem Ensuring CIA Triad. International Journal of Engineering and Advanced Technology, 12(1):50-53. doi: 10.35940/ijeat.a3841.1012122.

WANG, T.; QIN, S.; CHOW, K.P . Towards vulnerability types classification using pure self-attention: A common weakness enumeration based approach. In: 2021 IEEE 24th International Conference on Computational Science and Engineering (CSE). IEEE, 2021. p. 146-153.

ZHANG, Xiaoyu et al. Ensuring confidentiality and availability of sensitive data over a network system under cyber threats. **Reliability Engineering & System Safety**, v. 214, p. 107697, 2021.