



UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE SISTEMAS DE COMPUTAÇÃO

Laboratório de Engenharia de Segurança - SSC0901

Implementação e exploração de um site vulnerável a SQL Injection e XSS

Projeto 1

Gustavo Moura Scarenci de Carvalho Ferreira - 12547792
Matheus Henrique Dias Cirillo - 12547750

Docente responsável: Dr. Professor Rodolfo Ipolito Meneguette

São Carlos
2º semestre/2024

Sumário

1	Introdução	1
2	Configuração do Ambiente	2
3	O site	2
4	Vulnerabilidades	3
4.1	SQL Injection	3
4.2	Cross-Site Scripting (XSS)	4
5	Conclusão	5

1 Introdução

O presente projeto tem como objetivo a criação de uma aplicação web vulnerável, com foco na exploração de duas principais vulnerabilidades: SQL Injection e Cross-Site Scripting (XSS). A aplicação foi desenvolvida para rodar em uma máquina virtual configurada com Apache, PHP e MySQL, permitindo que os conhecimentos adquiridos na disciplina de Laboratório de Engenharia de Segurança sejam colocados em prática. Este relatório descreve o ambiente configurado, as vulnerabilidades exploradas, os métodos utilizados e os resultados obtidos, além de discutir possíveis formas de mitigação dessas vulnerabilidades em ambientes de produção. O trabalho foi mostrado também em um [vídeo no Youtube](https://youtu.be/SEvqmmeS5eg)¹ e os códigos estão disponíveis em um [repositório do GitHub de um dos alunos da dupla](https://github.com/GuScarenci/VulnerableSite).²

¹<https://youtu.be/SEvqmmeS5eg>

²<https://github.com/GuScarenci/VulnerableSite>

2 Configuração do Ambiente

O ambiente utilizado para esse projeto consistiu de:

- VM no Virtual Box 1.2 com o SO Ubuntu 24.04.1 LTS. A VM foi configurada para permitir comunicação entre a máquina host e a VM.
- MySQL 8.0.39
- PHP 5.6.40
- Apache 2.4.58

Ao configurar o MYSQL com o comando `sudo mysql_secure_installation`, escolheram-se sempre as opções que deixassem a configuração mais insegura.

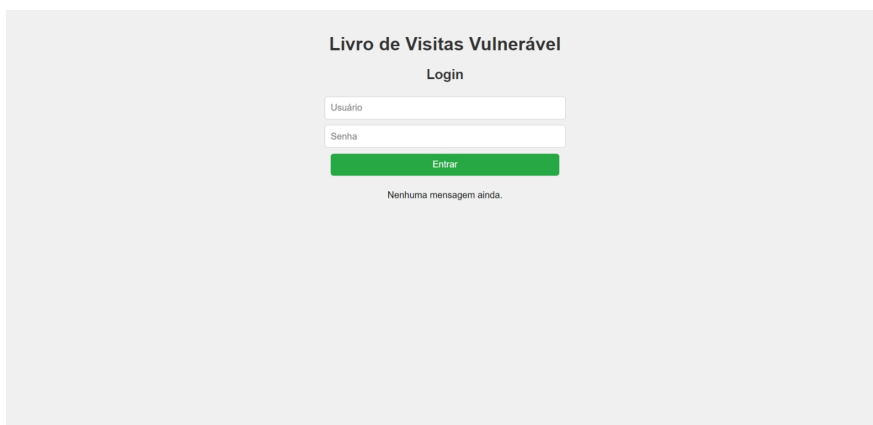
Para a execução do site como esperado, o método de autenticação padrão `bcrypt` foi substituído pelo método `mysql_native_password` suportado pela versão do PHP 5.6, requerida pelo professor e, além disso, que é mais insegura.

Em síntese, o Apache e MySQL foram configurados com permissões mínimas para simular um ambiente vulnerável. O site foi desenvolvido usando uma versão desatualizada e vulnerável do PHP para aumentar as possibilidades de exploração. As vulnerabilidades foram exploradas tanto na interface de login quanto no formulário de envio de mensagens, ambos propositalmente sem as medidas de segurança adequadas.

3 O site

A aplicação web desenvolvida consiste em um site simples de "Livro de Visitas", onde os usuários podem se registrar, fazer login e postar mensagens visíveis para todos. Abaixo, estão destacadas as principais funcionalidades implementadas:

- Login e Logout: Um sistema simples de autenticação, vulnerável a SQL Injection.



A imagem mostra a interface de login de um sistema web. No topo, o título "Livro de Visitas Vulnerável" é exibido em uma fonte preta. Abaixo dele, o texto "Login" aparece em uma fonte menor. Há dois campos de entrada de texto: o primeiro é rotulado "Usuário" e o segundo "Senha". Ambos os campos são retangulares com bordas cinzas. Abaixo dos campos, há um botão verde com o texto "Entrar" em branco. Na base da interface, uma linha de texto pequena diz "Nenhuma mensagem ainda."

Figura 1: Tela inicial de login.

- Formulário de envio de mensagens: Usuários autenticados podem postar mensagens públicas, onde há vulnerabilidade a Cross-Site Scripting (XSS).

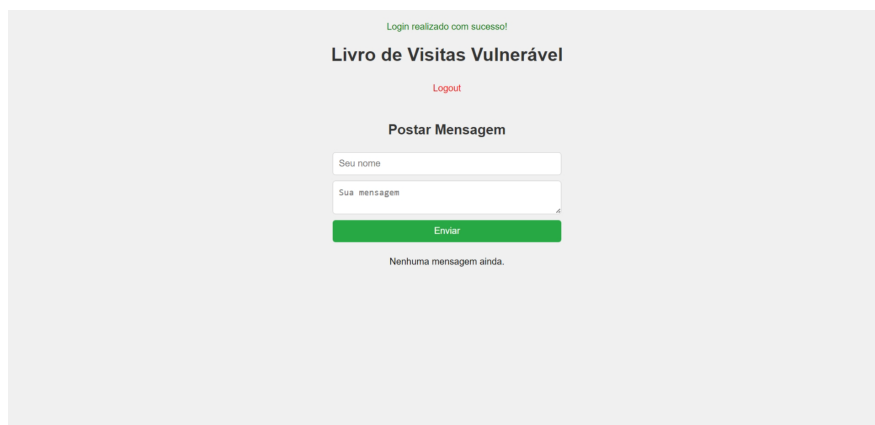


Figura 2: Tela logada.

- Listagem de mensagens: As mensagens postadas são exibidas para todos os visitantes da página, com conteúdo HTML não sanitizado, permitindo exploração XSS.
- O código da aplicação foi desenvolvido intencionalmente vulnerável para permitir a exploração de injeções de SQL e execução de scripts maliciosos (XSS).

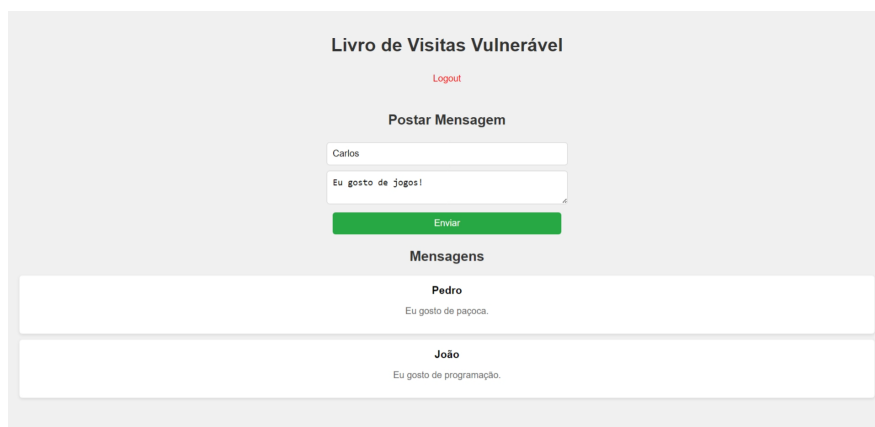


Figura 3: Tela com algumas mensagens.

O código do site está disponível no [repositório do GitHub da dupla](#) e o seu funcionamento padrão e exploração de vulnerabilidades pode ser visto no [vídeo do YouTube](#) em que um dos integrantes mostrou o trabalho.

4 Vulnerabilidades

4.1 SQL Injection

A vulnerabilidade de SQL Injection foi explorada no campo de mensagens, onde não há qualquer tipo de sanitização dos dados fornecidos pelo usuário. Isso permite que consultas SQL adicionais

sejam injetadas no banco de dados.

Por exemplo, ao tentar injetar o seguinte comando no campo de mensagens:

```
Test message'); DROP TABLE mensagens; -
```

A consulta enviada ao banco de dados se torna:

```
INSERT INTO mensagens (nome, mensagem) VALUES ('User', 'Test message'); DROP TABLE  
mensagens; -
```

Como resultado, a tabela mensagens é deletada do banco de dados, demonstrando a gravidade de uma vulnerabilidade SQL Injection. Em um cenário real, um atacante poderia usar essa técnica para obter informações confidenciais ou modificar dados críticos da aplicação.

4.2 Cross-Site Scripting (XSS)

A vulnerabilidade de Cross-Site Scripting (XSS) foi explorada no formulário de envio de mensagens. Como o conteúdo das mensagens é exibido diretamente na página sem ser filtrado, um atacante pode injetar *scripts* maliciosos que serão executados no navegador de qualquer usuário que visitar a página.

Para explorar essa vulnerabilidade, foi inserido o seguinte código no campo de mensagem:

```
<script>alert('XSS Vulnerability Exploited');</script>
```

Ao carregar a página, o script foi executado, exibindo um alerta no navegador. Em cenários mais graves, o XSS pode ser usado para roubar cookies de usuários ou redirecioná-los para sites maliciosos.

Após realizar os testes de SQL Injection e Cross-Site Scripting (XSS), observou-se que as vulnerabilidades presentes permitiram a execução de ataques bem-sucedidos.

5 Conclusão

Este projeto demonstrou, de forma prática, a exploração de vulnerabilidades comuns em aplicações web: *SQL Injection* e *Cross-Site Scripting* (XSS). Ambas as vulnerabilidades poderiam permitir que ataques críticos fossem executados, incluindo a manipulação do banco de dados e a execução de *scripts* maliciosos no navegador do usuário.

Para mitigar essas vulnerabilidades em sistemas reais, as seguintes medidas podem ser implementadas:

SQL Injection: Utilização de consultas preparadas (*prepared statements*) e parametrização de entradas.

XSS: Sanitização adequada das entradas do usuário e uso de funções como `htmlspecialchars()` ao exibir conteúdo gerado por usuários.