Segurança Informática e nas Operações

Vulnerabilidades

Projeto nº 1

2022/2023

Diogo Pires, 97889, P7
João Monteiro, 102690, P7
João Sousa, 103415, P1
Vânia Morais, 102383, P7

Índice

Introdução **2**

Vulnerabilidades **3**

CWE-256: Plaintext Storage of a Password(7.8HIGH) 3
CWE-79: Cross Site Scripting(XSS)(4.8 MEDIUM) 3
CWE-521: Weak Password Requirements(7.5HIGH) 3
CWE-89: SQL Injection(7.8HIGH) 4
CWE-20: Improper Input Validation(6.5MEDIUM) 4
CWE-200: Exposure of Sensitive Information to an Unauthorized Actor(5.3LOW) 5

Conclusão **6**

Fontes **6**

Introdução

Este projeto foi feito no contexto da cadeira de SIO, onde nos foi pedido para desenvolver uma página web simples para uma clínica de saúde.

A eHealth é uma página web onde o utilizador, após ter uma conta, pode marcar uma consulta numa das 3 especialidades apresentadas, ver o seu histórico de consultas e ainda consultar exames feitos. Pode ainda, mesmo que não tenha feito Login, conhecer a equipa de médicos e contactar a equipa.

Um website com todas estas funcionalidades guarda muitos dados delicados acerca dos seus utilizadores, como emails e passwords.

Tendo isso em conta, há duas versões da página web: uma com várias vulnerabilidades e outra, segura, onde as vulnerabilidades anteriores são resolvidas. Neste relatório está então presente quais as vulnerabilidades encontradas na primeira versão do website e como as resolvemos na segunda versão do mesmo.

Vulnerabilidades

CWE-256: Plaintext Storage of a Password(7.8HIGH)

Há certas informações que são mais delicadas que outras, como o exemplo da password. Deste modo, não deve ser guardada na base de dados como os outros dados (email e nome do utilizador), onde uma pessoa mal intencionada pode facilmente aceder.

Para resolver esta vulnerabilidade, password hashing é uma boa solução, onde a password é encriptada.

CWE-79: Cross Site Scripting(XSS)(4.8 MEDIUM)

O Cross-Site Scripting é uma vulnerabilidade, da família das *Injections*, que permite a um atacante manipular as interações de um utilizador com uma aplicação web vulnerável. Os atacantes muitas vezes realizam quaisquer ações que o utilizador alvo normalmente realizaria, incluindo a obtenção de acesso aos seus dados.

Para prevenir tudo isto, usamos a nosso favor o facto da linguagem Flask já resolver esta vulnerabilidade sem serem necessárias linhas de código adicionais.

CWE-521: Weak Password Requirements(7.5HIGH)

Para um site mais seguro para os utilizadores, é também preciso que os próprios colaborem. Uma das maneiras de o fazer, é obrigando-os a criar passwords mais fortes, com letras maiúsculas e números.

Para combater esta vulnerabilidade foi feita uma função (*password_check*) onde são definidos todos os requisitos que achamos necessários para uma password ser forte (mais de 8 caracteres, pelo menos um número e pelo menos uma letra maiúscula).

Passwords mais fortes previnem ataques do tipo *Brute Force*, onde as passwords são descobertas por tentativas.

CWE-89: SQL Injection(7.8HIGH)

SQL é uma linguagem de consulta concebida para aceder, modificar, e apagar dados armazenados em bases de dados relacionais. Numerosas aplicações web e websites utilizam bases de dados SQL como método de armazenamento de dados.

Aplicações são vulneráveis a este tipo de ataques quando os dados fornecidos pelo utilizador não são devidamente filtrados.

Um atacante pode usar a SQL Injection para manipular uma consulta SQL através dos dados de entrada do cliente para a aplicação, forçando assim o servidor SQL a executar uma operação não intencional construída usando uma entrada não confiável.

Para resolver toda esta situação, basta não permitir a inserção de caracteres especiais nos campos de preenchimento.

CWE-20: Improper Input Validation(6.5MEDIUM)

Esta vulnerabilidade assume que os dados introduzidos pelo utilizador, são logo armazenados na base de dados. Assim, o atacante pode ser capaz de manipular a informação que entra no sistema, o que pode trazer várias consequências negativas.

Para evitar tudo isto, devem ser feitas validações de modo a tornar o sistema mais seguro.

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor(5.3LOW)

A aplicação fornece informações que põem em causa a confidencialidade sobre os utilizadores. O utilizador pode tirar proveito desta vulnerabilidade para obter dados relativos à base de dados, o estado atual da aplicação e muitas outras finalidades.

Ao fazer o login, quando o email está errado aparece a mensagem de erro “Invalid username or password” ao utilizador, mas quando o utilizador erra apenas o email ou a password aparece uma mensagem de erro a dizer “Invalid password”.

Através das mensagens o atacante pode aproveitar um email válido de alguma conta e, por exemplo, fazer um ataque de força bruta para descobrir a password da conta em específico.

Para prevenir esta vulnerabilidade basta alterar no código para a mensagem de erro da password ser a mesma mesmo que só a password esteja errada (“Invalid username or password”).

Conclusão

Com este projeto, consolidamos os nossos conhecimentos sobre vulnerabilidades, quais as suas consequências e como as prevenir. Também aprendemos *Flask* e pusemos em prática os nossos conhecimentos em *Docker*.

Em suma, concluímos que um site inseguro pode trazer muitas complicações que podem ser prevenidas facilmente.

Fontes

Para a realização deste projeto, usamos maioritariamente slides e informações disponíveis no site da cadeira de SIO.
Para aprendizagem de *flask*, foi usada a documentação do mesmo (https://flask.palletsprojects.com/en/2.2.x/) com adição a outras páginas quando havia erros específicos. Isto também se aplica para o uso de *sqlalchemy.*