Introdução

Este projeto **Análise de Vulnerabilidades** apresenta um script em Python criado para demonstrar conceitos básicos de reconhecimento e detecção de vulnerabilidades web, com foco em **SQL Injection (SQLi)** e **Cross-Site Scripting (XSS)**. O objetivo não é realizar um pentest completo, mas sim automatizar verificações iniciais em ambientes de teste controlados para consolidar conhecimentos em segurança da informação.

O documento foi feito para consolidar o aprendizado em segurança da informação e contextualizar o projeto.

Objetivo

- Demonstrar técnicas de detecção de vulnerabilidades web.
- Registrar metodologia, resultados, limitações e recomendações.

Funcionamento

- O script faz uma requisição inicial ao alvo, analisa o HTML em busca de formulários e verifica a presença de tokens CSRF (indicador de proteção contra certos ataques de injeção).

- Para detecção de SQLi, o script injeta *payloads* simples em parâmetros comuns (ex.: `id`, `q`, `search`) e analisa as respostas do servidor à procura de indícios de erro de banco de dados.

- Os resultados são agrupados em um relatório simplificado.

Aviso legal e ético

O projeto foi destinado apenas para fins educacionais em um site autorizado.(**http://testphp.vulnweb.com**)

Contextualização

SQL injection - Uma injeção SQL (SQLi) é um tipo de ataque no qual cibercriminosos tentam explorar vulnerabilidades no código de um aplicativo inserindo uma consulta SQL em campos regulares de entrada ou formulário, como um nome de usuário ou senha. A instrução SQL é então passada para o banco de dados SQL subjacente do aplicativo.

fonte (https://www.fortinet.com/br/resources/cyberglossary/sql-injection)

No projeto ***Análise de vulnerabilidades*** , o conceito de SQL Injection é aplicado por meio de uma funcionalidade que testa parâmetros comuns de URLs, como id ou search, com payloads simples, como ' OR 1=1 --. Esses payloads são enviados em requisições HTTP para simular tentativas de manipulação do banco de dados de um site, explorando possíveis falhas na validação de entradas. O script analisa as respostas do servidor, procurando por mensagens de erro (como "SQL syntax error" ou "database error") que indicam uma vulnerabilidade.

Cross-site scripting (XSS) - É uma vulnerabilidade de segurança na web que permite que um invasor comprometa as interações dos usuários com um aplicativo vulnerável. Ela permite que um invasor contorne a mesma política de origem, projetada para segregar diferentes sites uns dos outros. Vulnerabilidades de cross-site scripting normalmente permitem que um invasor se faça passar por um usuário vítima, execute quaisquer ações que o usuário possa realizar e acesse quaisquer dados do usuário. Se o usuário vítima tiver acesso privilegiado ao aplicativo, o invasor poderá obter controle total sobre todas as funcionalidades e dados do aplicativo.

fonte(https://portswigger.net/web-security/cross-site-scripting)

No projeto **Análise de vulnerabilidade**, o conceito de Cross-Site Scripting (XSS) é abordado por meio de uma funcionalidade que escaneia formulários HTML em um site para verificar a presença de tokens CSRF, cuja ausência pode indicar um risco de vulnerabilidades XSS. O script utiliza a biblioteca BeautifulSoup para analisar o código HTML e identificar formulários que não implementam proteções

adequadas, permitindo, potencialmente, que um invasor injete scripts maliciosos, como , que poderiam ser executados no navegador de um usuário vítima.

para executar o projeto certifique-se de ter instalado as seguintes bibliotecas:

(atalho para abrir o terminal: CTRL + “)

pip install requests beautifulsoup4

para executar o script é necessário fazer o seguinte:
python main.py http://testphp.vulnweb.com

Tipos de abordagens em pentest:

Black-box

- Descrição: O testador não recebe nenhuma informação prévia sobre o alvo, atuando como um atacante externo desconhecido.

Grey-box

- Descrição: O testador recebe informações parciais por exemplo: documentação limitada, contas de usuário não-privilegiadas ou endereços IP internos.

White-box

- Descrição: O testador recebe todas (ou quase todas) as informações: arquiteturas, código-fonte, diagramas, credenciais e acesso interno.

.