GAC119 - Trabalho Prático: The Security Blog

Luiz Otavio Andrade Soares - 201810544
Mateus Carvalho Gonçalves - 201810245
Pedro Antônio de Souza - 201810557

Introdução

Este documento descreve a metodologia adotada durante o desenvolvimento do Trabalho Prático da disciplina Metodologia de Desenvolvimento Seguro de Aplicações. O trabalho teve como objeto de estudo o projeto de um blog web, The Security Blog.

Para o projeto do blog, foi utilizado a biblioteca React para construir as telas, Ant Design como design system. No lado do servidor, foi utilizada a biblioteca express.js com a ORM Sequelize. A aplicação roda no Node e o banco de dados escolhido foi Postgres. O back-end e o banco foram hospedados no Heroku.

Nessa aplicação, existe um sistema de autenticação com três papéis de usuário que definem as ações possíveis. Um usuário com papel Padrão pode apenas ver posts e criar comentários; o Criador de conteúdo pode criar posts e apagar os próprios conteúdos; e o Moderador pode remover posts e comentários alheios, além de gerenciar os papéis e apagar usuários. Também, é possível navegar na aplicação sem estar logado, e nesse caso o "usuário" consegue apenas visualizar os conteúdos, sem conseguir interagir.

Metodologia

O sistema de autenticação, composto por e-mail e senha, apoia na resolução de inúmeros problemas de segurança. Para criar, editar e remover conteúdo é necessário estar logado, além de que um usuário só pode remover posts e comentários dele mesmo (exceto Moderadores).

Para mitigar o problema da criação de senhas fracas e consequentemente acesso a contas de terceiros, foi definido um padrão para criação de senha que consiste em ter no mínimo 8 caracteres e incluir obrigatoriamente uma letra maiúsculas e uma minúscula, um números e um dos seguintes caracteres especiais: "!", ".", "@", "#", "$", "&", "*", "_" ou "-". Essa validação é feita por meio de uma expressão regular no back-end.

O cadastro de usuário recebe apenas nome, e-mail e senha. Como entre esses atributos apenas a senha é sensível, apenas ela é guardada criptografada no banco de dados. O nome, por exemplo, é um dado público já que é mostrado no header de posts e comentários. Para

login, a senha informada é criptografada utilizando o método bcrypt e comparada com o hash presente no banco. Para evitar ataques de força bruta no login, também é adicionado um delay de três segundos na resposta quando uma tentativa de login é inválida.

Outra parte importante que envolve segurança é a validação das entradas. Parte dessas validações é feita pelo Sequelize, uma ORM. Entre as validações que esse framework faz são a sanitização de SQL e validação de tipo. Abaixo estão listadas, por entidade e atributo, as validações desenvolvidas em código:

- Usuário
    - Nome
        - Tamanho: 1 a 128 caracteres
        - Expressão regular que aceita apenas letras, ponto final, vírgula, apóstrofo, hífen e espaço
    - E-mail
        - Expressão regular para verificar se o formato do dado enviado segue os padrões de um e-mail, isto é, uma sequência de caracteres sem espaço, seguida de arroba e um domínio válido.
- Post
    - Título
        - Tamanho: 1 a 32 caracteres
        - Sanitização de HTML
    - Conteúdo
        - Tamanho: 1 a 1024 caracteres
        - Sanitização de HTML
- Comentário
    - Conteúdo
        - Tamanho: 1 a 256 caracteres
        - Sanitização de HTML

Para lidar com problemas de disponibilidade, escolheu-se hospedar o back-end e o banco de dados no Heroku, uma plataforma de nuvem como serviço já consolidada e reconhecida. Apesar do projeto usar a versão grátis do serviço, que consequentemente é mais fraca, entende-se que num cenário real as configurações seriam mudadas para suportar o uso da aplicação.

Por fim, como descrito na introdução, existem papéis que controlam o acesso às funcionalidades, similar a ACLs. Quando um usuário se cadastra, ele automaticamente o menor papel possível, o Padrão. A única forma de mudar isso seria por meio de um serviço que apenas Moderadores têm acesso. Logo, para fazer ataques de elevação de privilégio a única forma possível é ter acesso a uma conta de usuário moderador.

Limitações

Apesar de o projeto seguro da aplicação (primeira etapa do trabalho) prever a possibilidade e tratamento de upload de arquivos, essa implementação foi reconsiderada e não foi desenvolvida.

Outro questão abordada naquele documento é a existência dos Termos de Uso, que não foram escritos devido a falta de conhecimento dos autores nesse nicho e pelo objetivo do trabalho. Porém, entende-se a necessidade desse componente em uma aplicação real disponível na internet.

É entendido pelo grupo que existem mudanças necessárias com certa urgência. A primeira é impor regras mais fortes de senha para usuários moderadores. Seria interessante, por exemplo, duplicar a quantidade mínima de caracteres nesse caso, passando para 16.

Outro problema corrente é a invalidação do token de cadastro. Esse token é criado e fica disponível para finalizar o cadastro por 24 horas. Porém se o usuário for removido por qualquer motivo e o tempo de expiração não terminou, é possível utilizá-lo novamente, mesmo que seja um token idealizado para uso único. Dessa forma, é importante implementar uma lista de tokens inválidos para armazenar todos aqueles que já forem utilizados.

Por fim, exalta-se que, por questões de tempo, não foi possível refinar as mensagens de erro no front-end, apesar do back-end retornar mensagens explicativas, ainda que diretas.