**Capítulo 8. Olá, Spring Security**

Esta seção aborda a configuração mínima para usar o Spring Security com Spring Boot. O aplicativo completo pode ser encontrado em samples/boot/helloworld.

Para sua conveniência, você pode baixar um aplicativo minimalista do Spring Boot + Spring Security clicando aqui.

**8.1. Atualizando Dependências**

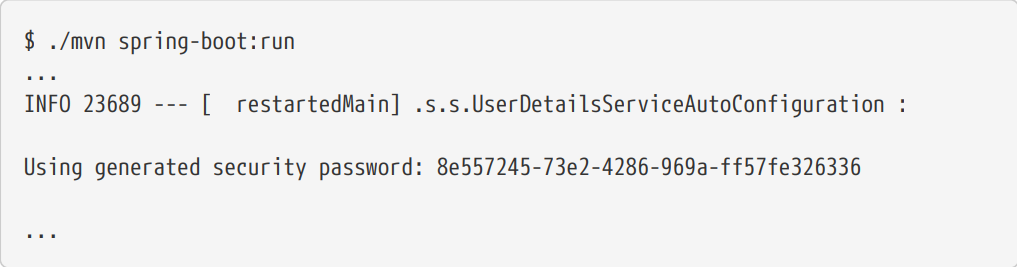
O único passo necessário é atualizar as dependências usando Maven ou Gradle.

**8.2. Iniciando o Hello Spring Security Boot**

Agora você pode executar a aplicação Spring Boot usando o objetivo run do Plugin Maven.

O exemplo a seguir mostra como fazer isso (junto com o início da saída gerada ao executar o comando):

**Exemplo 46. Executando a Aplicação Spring Boot**



**8.3. Configuração Automática do Spring Boot**

O Spring Boot automaticamente:

* **Habilita a configuração padrão do Spring Security**, que cria um filtro servlet como um bean denominado springSecurityFilterChain. Esse bean é responsável por toda a segurança (proteção das URLs da aplicação, validação do nome de usuário e senha enviados, redirecionamento para o formulário de login, entre outros) dentro da sua aplicação.
* **Cria um bean UserDetailsService** com o nome de usuário user e uma senha gerada aleatoriamente, que é registrada no console.
* **Registra o filtro** com o bean denominado springSecurityFilterChain no contêiner Servlet para cada solicitação.

Embora o Spring Boot não configure muitos aspectos, ele realiza várias ações importantes. Um resumo das funcionalidades é o seguinte:

* Exige um usuário autenticado para qualquer interação com a aplicação.
* Gera um formulário de login padrão para você.
* Permite que o usuário com o nome de usuário user e uma senha registrada no console se autentique por meio de autenticação baseada em formulário (no exemplo anterior, a senha é 8e557245-73e2-4286-969a-ff57fe326336).
* Protege o armazenamento de senhas com BCrypt.
* Permite que o usuário faça logout.
* Prevenção contra ataques CSRF.
* Proteção contra Fixação de Sessão.
* Integração de Cabeçalhos de Segurança:
  + HTTP Strict Transport Security para requisições seguras.
  + Integração de X-Content-Type-Options.
  + Controle de Cache (pode ser substituído mais tarde pela sua aplicação para permitir o cache de recursos estáticos).
  + Integração de X-XSS-Protection.
  + Integração de X-Frame-Options para ajudar a prevenir o Clickjacking.
* Integra-se com os seguintes métodos da API Servlet:
  + HttpServletRequest#getRemoteUser()
  + HttpServletRequest#html#getUserPrincipal()
  + HttpServletRequest#html#isUserInRole(java.lang.String)
  + HttpServletRequest#html#login(java.lang.String, java.lang.String)
  + HttpServletRequest#html#logout()