Spring Security

Spring Security Fundamentals

Main concepts

- authentication
  (who I am)
- authorization
  (what I can do)
- encryption

Authentication

- used by a server when it needs to know exactly who is accessing their information
- usually, authentication entails the use of a user name and password, other ways to authenticate can be through cards, voice recognition and fingerprints
- does not determine what tasks a user can do or what files he can see, it just identifies and verifies who the person is
- should be used whenever you want to know exactly who is using or viewing your site

Authorization

- defines a process by which a server determines if the client has permission to use a resource or access a file
- usually coupled with authentication so that the server has some concept of who the client is that is requesting access
- should be used whenever you want to control viewer access of certain pages
- in some cases, there is no authorization, any user can use a resource or access a file simply by asking for it

Encryption

- a process of transforming data so that it is unreadable by anyone who does not have a decryption key
- https protocol is usually used in encryption processes
- by encrypting the data exchanged between the client and server information can be sent over the Internet with less risk of being intercepted during transit
- should be used whenever people are giving out personal information to register for something or buy a product

Maven dependencies

- spring-security-web
  (groupId: org.springframework.security)
- spring-security-config
  (groupId: org.springframework.security)

Web configuration additions

- define a filter
  org.springframework.web.filter.DelegatingFilterProxy
- define a listener
  org.springframework.web.context.ContextLoaderListener
  context-param: contextConfigLocation points to security-config.xml

Minimal security configuration

```
<http auto-config="true">
    <intercept-url pattern="/**" access="ROLE_USER"/>
</http>

<authentication-manager>
    <authentication-provider>
        <user-service>
            <user name="john" password="123" authorities="ROLE_USER"/>
        </user-service>
    </authentication-provider>
</authentication-manager>
```

Database configuration

- create two tables
  users (fields: username, password, enabled)
  authorities (fields: username, authority)
- create a user and his rights
  insert some data into the tables
- change “user-service” to “jdbc-user-service” in the security-config.xml

Spring Security tags

- the library needs to be included in your jsp page:
  <%@ taglib prefix="sec"
  uri="http://www.springframework.org/security/tags" %>

- tags:
  - authentication
  - authorization

Authentication tag

- used to gain access to the authenticated user object
- has a property attribute for accessing properties of that object
  - name
  - authorities
  - credentials
  - details
  - principal
  - isAuthenticated

Authorize tag

- used to control access to parts of the page
- has such attributes:
  - url
  - method
  - var
  - access
  - ifAnyGranted (any of the listed roles must be granted)
  - ifAllGranted (all the listed roles must be granted)
  - ifNotGranted (none of the listed roles must be granted)

Password encryption

- MD5 hash
- BCrypt

MD5 hash

- one of the first hash algorithms
- <password-encoder hash="md5">
- update the database with a new password

BCrypt

- more secure than MD5
- <password-encoder hash="bcrypt"/>
- update the database with a new password

Basic authentication

- usually used for REST applications
- when you enter a url, browser will show a popup window
- enabled with <http-basic/> tag

Custom login form

- define an intercept-url with access to any user
  <intercept-url pattern="/login"
  access="IS_AUTHENTICATED_ANONYMOUSLY"/>
- add a form-login tag instead of http-basic
  <form-login login-page="/login"/>
- add a jsp page with a few key points:
  - action="j_spring_security_check"
  - input with name "j_username"
  - input with name "j_password"

Expressions

- set use-expression to the http tag
  <http use-expressions="true"/>

- simplifies boolean logic

- expressions list:
  - hasRole
  - hasAnyRole
  - permitAll
  - hasPermission