

Security

Petr Křemen, Lama Saeeda, and Martin Ledvinka

KBSS

Winter Term 2023



Contents

1 Spring Security

2 Tasks



Spring Security



Spring Security

Spring Security offers the following annotations:

- `@PreFilter` for filtering input iterable argument based on security constraints expressed in SpEL.
- `@PostFilter` for filtering output iterable value based on security constraints expressed in SpEL.
- `@PreAuthorize` for authorizing method execution based on security constraints expressed in SpEL.
- `@PostAuthorize` for authorizing return from the method execution based on security constraints expressed in SpEL.



Spring Security - SpEL

Relevant SpEL expressions:

- `hasRole('ROLE_ADMIN')` checks whether the currently logged in user has the 'ADMIN' role.
- `and`, `or` logical operators.
- `principal` the currently logged in user, e.g., `principal.username`.
- `filterObject` the object filtered from the collection, e.g., `filterObject.customer`.

Become familiar with these annotations before starting the tasks.



Security in E-shop

Important security-related classes and notions used in the e-shop involve

- `SecurityConfig` and all the beans used in it
- `SecurityUtils`
 - Namely, the thread local-based `SecurityContextHolder`
- Aforementioned Spring security annotations



Security in E-shop

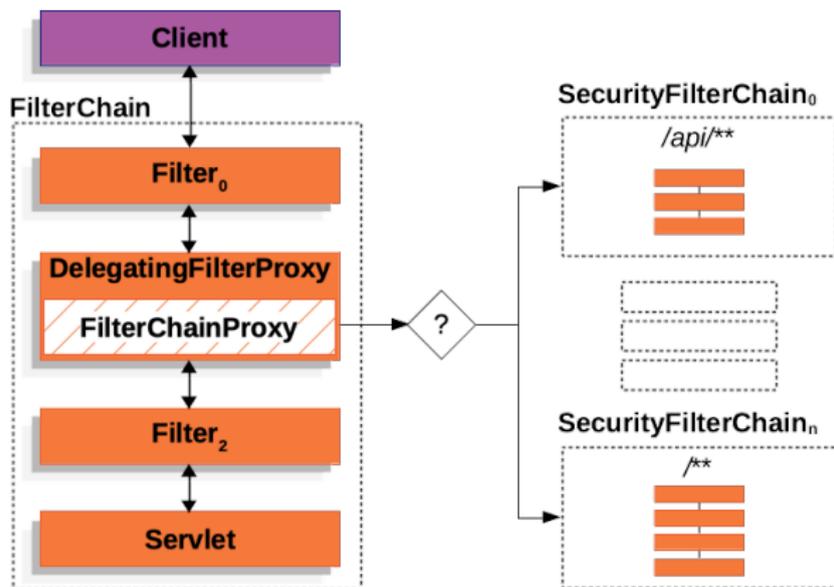


Figure: Source: <https://docs.spring.io/spring-security/reference/servlet/architecture.html>



Security in E-shop – Authentication

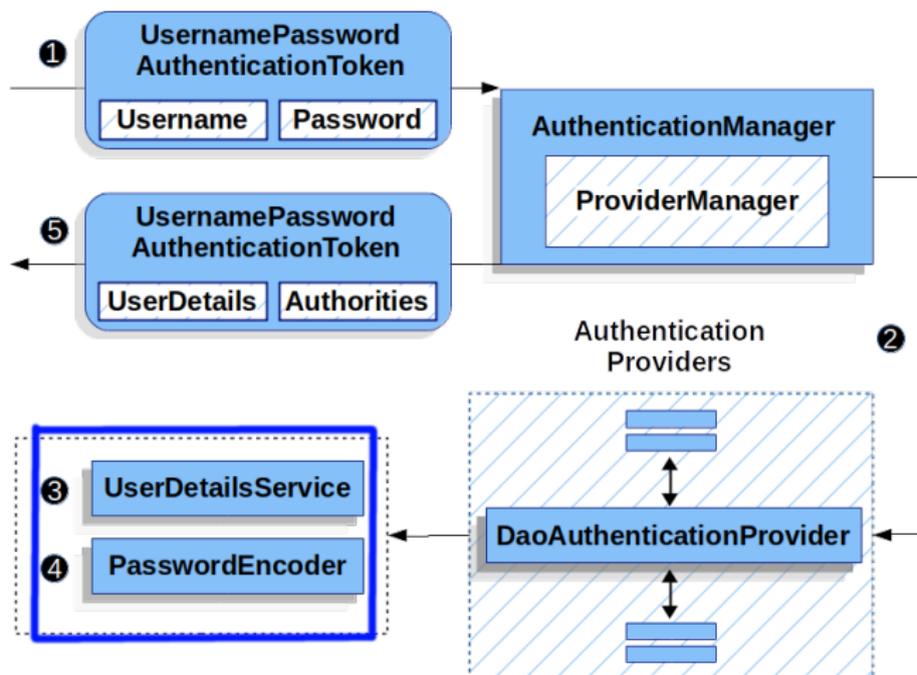


Figure: Source: <https://docs.spring.io/spring-security/reference/servlet/authentication/passwords/dao-authentication-provider.html>



Security in E-shop – Demo

- Let's start e-shop and put some breakpoints to see what happens behind the scene
- Put breakpoints into:
 - `AbstractUserDetailsAuthenticationProvider.authenticate` and log in
 - `AuthorizationFilter.doFilter` and access any API endpoint



Tasks



Syncing Your Fork

- 1 Ensure you have no uncommitted changes in the working tree
 - `git status` → your branch is up to date, nothing to commit
- 2 Fetch branches and commits from the upstream repository (EAR/B231-eshop)
 - `git fetch upstream`
- 3 Check out the task branch from **upstream** (one line!)
 - `git checkout -b b231-seminar-09-task upstream/b231-seminar-09-task`
- 4 Do the task
- 5 Commit and push the solution to **your** fork
 - `git push -u origin b231-seminar-09-task`



Task – 1 point

- 1 Data modifying operations in `CategoryController` should be allowed only for administrators. Use appropriate Spring security features to secure the corresponding endpoints.
- 2 Method `OrderService.findAll` should return only the orders of the current user or all orders if the current user is administrator.

Acceptance criteria:

- 1 All tests in `CategoryControllerSecurityTest` pass.
- 2 All tests in `OrderServiceSecurityTest` pass.

Hints: Administrators are users with `Role.ADMIN`.



Task – 1 point

- 1 The task branch contains code that is vulnerable to SQL injection
- 2 Exploit this vulnerability and then fix it
 - You can use e-shop UI or the REST API directly
- 3 Send an email to your teacher describing the exploit
 - How to exploit the vulnerability?
 - What code did you inject?
 - What was the effect?
- 4 Fix the vulnerability and push your changes

Hints: Login screen allows accessing the vulnerability



Resources

- <https://docs.spring.io/spring-framework/reference/core/expressions.html>
- <https://docs.spring.io/spring-security/reference/servlet/authorization/method-security.html>

