Spring Security

Аутентификация. Продолжение

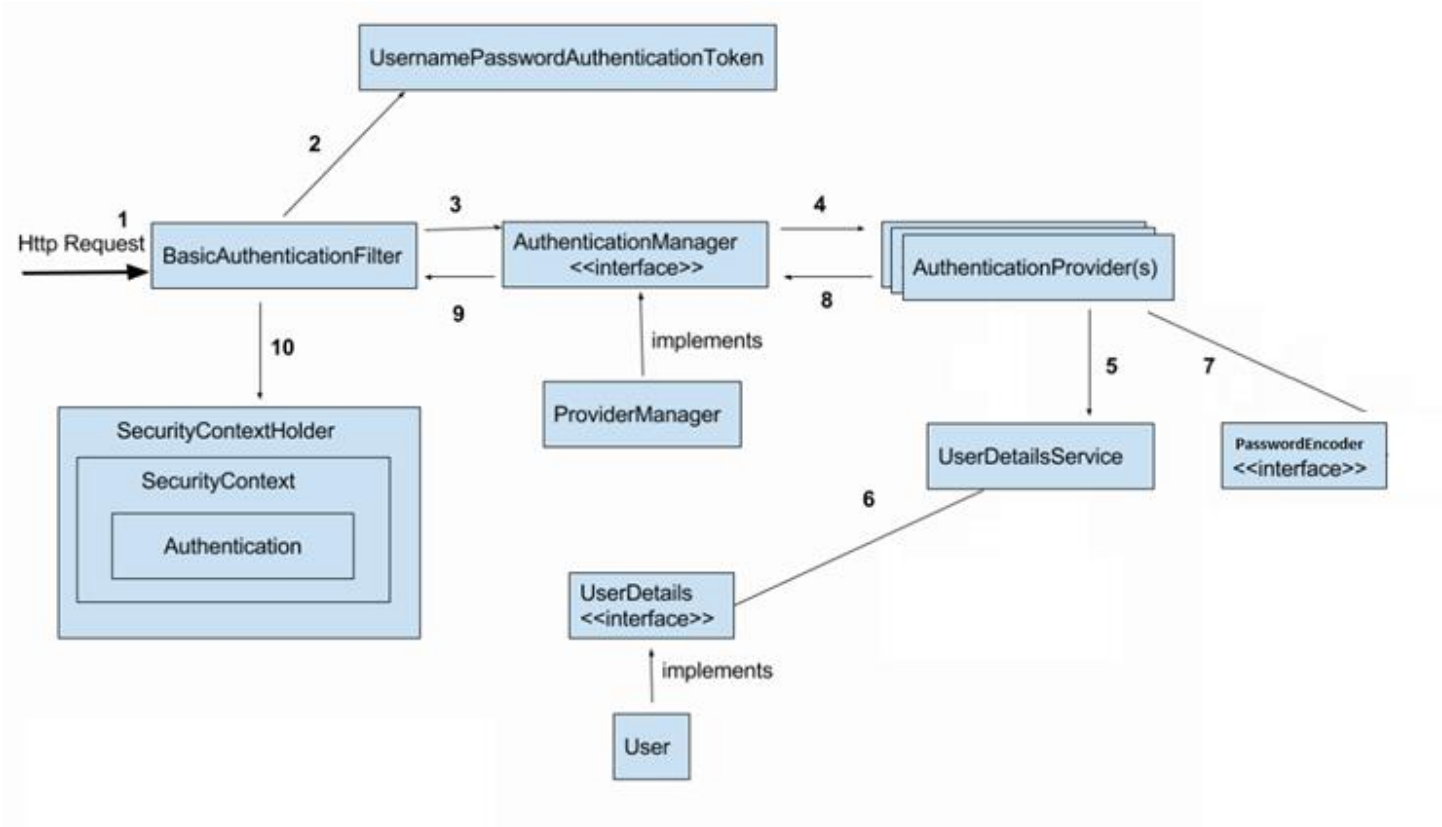
Идентификация, аутентификация, авторизация.

Идентификация – проверяет за кого пытается выдать себя пользователь

Аутентификация – проверяет действительно ли пользователь является тем, за кого он хочет себя выдать

Авторизация – проверяет какие права доступны аутентифицированному пользователю.

Архитектура аутентификации Servlet стека



Аутентификацию в Spring Security можно условно разделить на 2 вида:

1. Первичная аутентификация (Basic, FormLogin)
2. Вторичная аутентификация (Anonymous, Remember-Me)

Анонимная аутентификация

Выполняется фильтром AnonymousAuthenticationFilter, который находится в цепочке после фильтров первичной аутентификации. Идея данного подхода держать в SecurityContext какого-то пользователя (аналог гостевой учетной записи), чтобы можно было работать с SecurityContext всегда одинаковым образом. К примеру при логгировании имени пользователя или каком-то аудите. Такое может потребоваться, если ресурс может быть доступен и аутентифицированным пользователям и нет.

По умолчанию пользователь получает имя anonymousUser и одно право ROLE_ANONYMOUS. Это можно настраивать.

Демо

Remember-Me

Выполняется фильтром RememberMeAuthenticationFilter. Работает в связке с аутентификацией через форму, когда у пользователя есть сессия на сервере. Так как эта сессия ограничена, то данный механизм используется для увеличения времени, в течение которого пользователь может не логиниться заново в приложении.

Существует 2 подхода:

1. Simple Hash-Based Token
2. Persistent Token

Демо

Simple Hash-Based Token

Данный подход основан на хешировании. Сгенерированный токен передается через Cookie.

```
base64(username + ":" + expirationTime + ":" +
md5Hex(username + ":" + expirationTime + ":" password + ":" + key))

username:          As identifiable to the UserDetailsService
password:          That matches the one in the retrieved UserDetails
expirationTime:    The date and time when the remember-me token expires, expressed in
milliseconds
key:               A private key to prevent modification of the remember-me token
```

Persistent Token

Данный подход основан на хранении токенов. Для этого используется таблица со структурой. Значение токена так же передается в Cookie.

```
create table persistent_logins (username varchar(64) not null,
                                series varchar(64) primary key,
                                token varchar(64) not null,
                                last_used timestamp not null)
```

Демо

Мультифакторная аутентификация

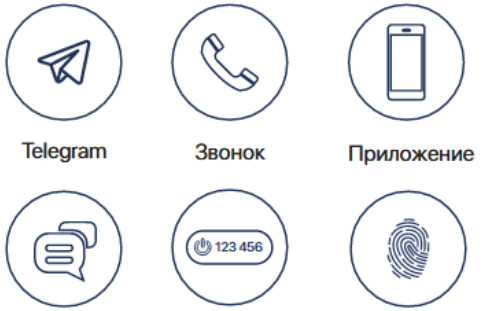
1 Первый фактор

Что пользователь знает:

Логин и пароль

2 Второй фактор

Что пользователь имеет или кем является:



3 Доступ

Доступ разрешён.