深入浅出 Spring Security

一场系统安全之旅

江南一点雨

华为云MVP、华为云云享专家

目录

1 Spring Security 基本认证

2 Spring Security 认证流程分析

3 配置多个数据源

4 添加登录验证码

Spring Security 基本认证

基本认证

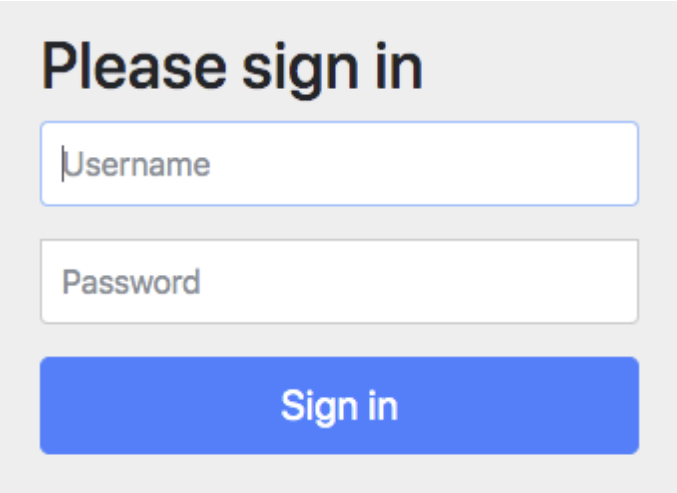
在Spring Boot项目中使用Spring Security非常方便，创建一个新的Spring Boot项目，我们只需要引入Web和Spring Security依赖即可。

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
```

Spring Security 基本认证

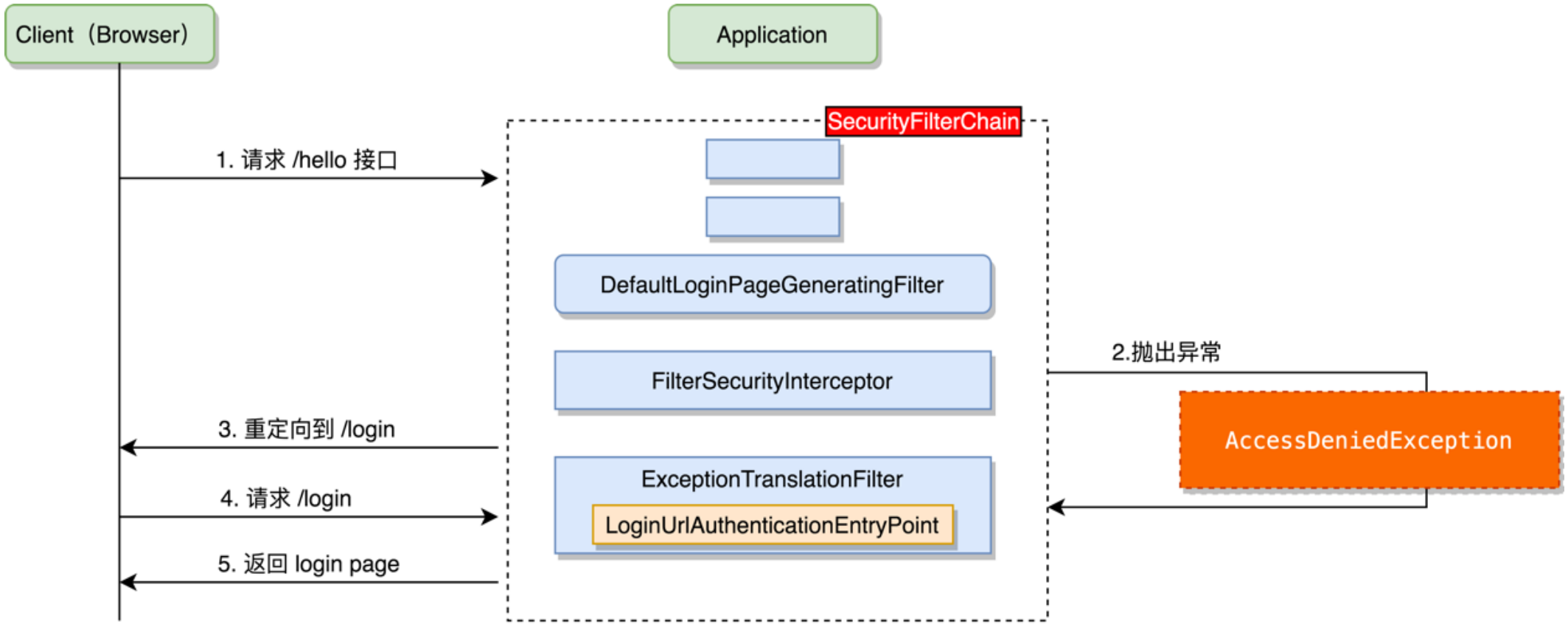
基本认证

引入 Spring Security 依赖后，项目中的所有接口就都被保护起来了，此时访问接口就可以看到登录页面了。



Spring Security 认证流程分析

认证流程



Spring Security 认证流程分析

AuthenticationManager

AuthenticationManager是一个认证管理器，它定义了Spring Security过滤器要如何执行认证操作。AuthenticationManager在认证成功后，会返回一个Authentication对象，这个Authentication对象会被设置到SecurityContextHolder中。

AuthenticationManager是一个接口，它有着诸多的实现类，开发者也可以自定义AuthenticationManager的实现类，不过在实际应用中，我们使用最多的是ProviderManager。在Spring Security框架中，默认也是使用ProviderManager。

Spring Security 认证流程分析

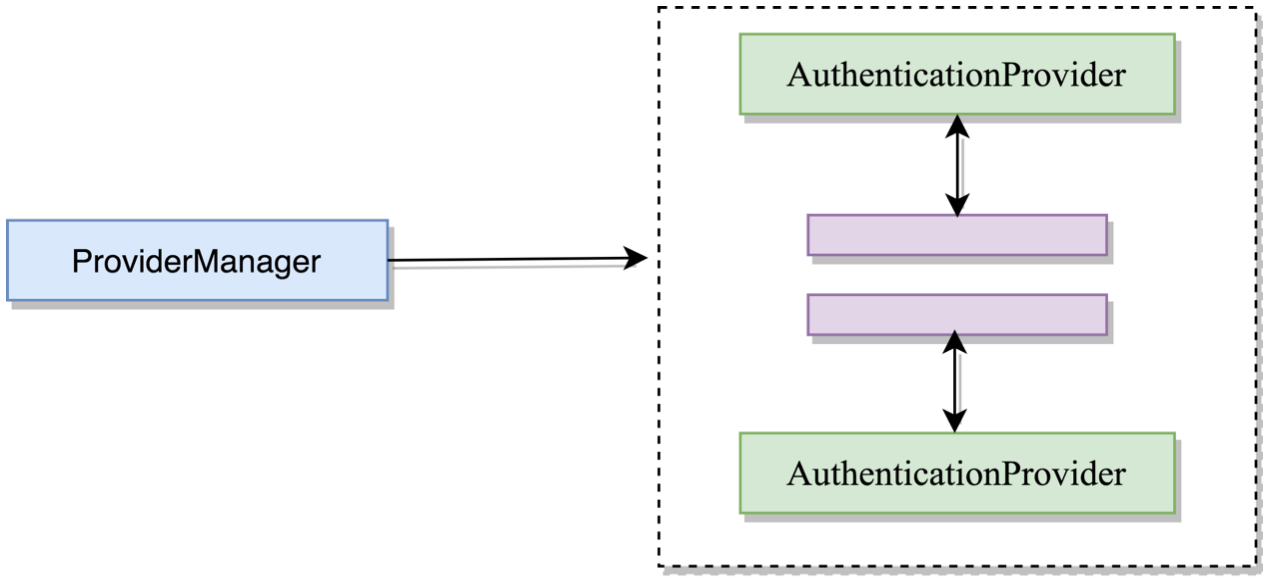
AuthenticationProvider

Spring Security支持多种不同的认证方式，不同的认证方式对应不同的身份类型，AuthenticationProvider就是针对不同的身份类型执行具体的身份认证。例如，常见的DaoAuthenticationProvider用来支持用户名/密码登录认证，RememberMeAuthenticationProvider用来支持“记住我”的认证。

Spring Security 认证流程分析

ProviderManager

ProviderManager是AuthenticationManager的一个重要实现类。



Spring Security 认证流程分析

ProviderManager

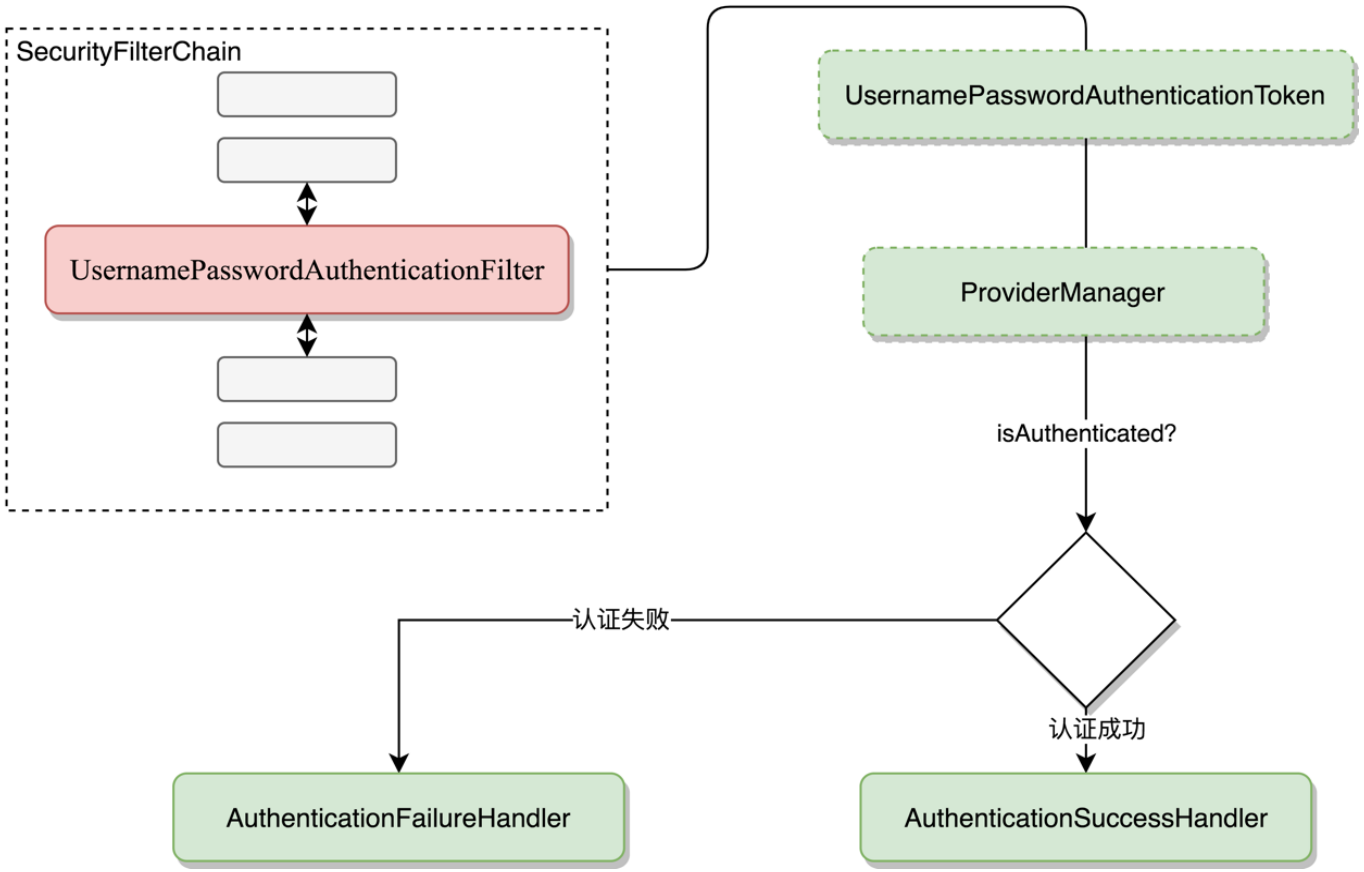
在Spring Security中，由于系统可能同时支持多种不同的认证方式，例如同时支持用户名/密码认证、RememberMe认证、手机号码动态认证等，而不同的认证方式对应了不同的AuthenticationProvider，所以一个完整的认证流程可能由多个AuthenticationProvider来提供。多个AuthenticationProvider将组成一个列表，这个列表将由ProviderManager代理。换句话说，在ProviderManager中存在一个AuthenticationProvider列表，在ProviderManager中遍历列表中的每一个AuthenticationProvider去执行身份认证，最终得到认证结果。

ProviderManager本身也可以再配置一个AuthenticationManager作为parent，这样当ProviderManager认证失败之后，就可以进入到parent中再次进行认证。理论上来说，ProviderManager的parent可以是任意类型的AuthenticationManager，但是通常都是由ProviderManager来扮演parent的角色，也就是ProviderManager是ProviderManager的parent。

Spring Security 认证流程分析

AbstractAuthenticationProcessingFilter

AbstractAuthenticationProcessingFilter可以用来处理任何提交给它的身份认证。



配置多个数据源

同时支持多张用户表

多个数据源是指在同一个系统中，用户数据来自不同的表，在认证时，如果第一张表没有查找到用户，那就去第二张表中查询，依次类推。

要实现这个需求就很容易，认证要经过AuthenticationProvider，每一个AuthenticationProvider中都配置了一个UserDetailsService，而不同的UserDetailsService则可以代表不同的数据源。所以我们只需要手动配置多个AuthenticationProvider，并为不同的AuthenticationProvider提供不同的UserDetailsService即可。

配置多个数据源

同时支持多张用户表

```
@Configuration
public class SecurityConfig extends WebSecurityConfigurerAdapter {
    @Bean
    @Primary
    UserDetailsService us1() {
        return new InMemoryUserDetailsManager(User.builder()
      .username("javaboy").password("{noop}123").roles("admin").build());
    }
    @Bean
    UserDetailsService us2() {
        return new InMemoryUserDetailsManager(User.builder()
            .username("sang").password("{noop}123").roles("user").build());
    }
    @Override
    @Bean
    public AuthenticationManager authenticationManagerBean() throws Exception {
        DaoAuthenticationProvider dao1 = new DaoAuthenticationProvider();
        dao1.setUserDetailsService(us1());
        DaoAuthenticationProvider dao2 = new DaoAuthenticationProvider();
        dao2.setUserDetailsService(us2());
        ProviderManager manager = new ProviderManager(dao1, dao2);
        return manager;
    }
    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.authorizeRequests()
                .anyRequest().authenticated()
                //省略
    }
}
```

添加登录验证码

登录验证码

登录验证码也是项目中一个常见的需求，但是Spring Security对此并未提供自动化配置方案，需要开发者自行定义。一般来说，有两种实现登录验证码的思路：

（1）自定义过滤器。
（2）自定义认证逻辑。

添加登录验证码

登录验证码

生成验证码，可以自定义一个生成工具类，也可以使用一些现成的开源库来实现，例如 kaptcha。

```
@Autowired
Producer producer;
@GetMapping("/vc.jpg")
public void getVerifyCode(HttpServletResponse resp, HttpSession session)
                                                    throws IOException {
    resp.setContentType("image/jpeg");
    String text = producer.createText();
    session.setAttribute("kaptcha", text);
    BufferedImage image = producer.createImage(text);
    try(ServletOutputStream out = resp.getOutputStream()) {
        ImageIO.write(image, "jpg", out);
    }
}
```

添加登录验证码

登录验证码

谢 谢