# 主要架构组件

1. SecurityContextHolder：SecurityContextHolder是Spring Security存储身份验证的详细信息的地方。
2. SecurityContext：从SecurityContextHolder中获取，包含当前认证用户的认证信息。
3. Authentication：可以是AuthenticationManager的输入，以提供用户为身份验证提供的凭据，也可以是来自SecurityContext的当前用户
4. GrantedAuthority：授予的权限（此接口的实现就是定义授予哪些权限）
5. AuthenticationManager：定义Spring Security的过滤器如何执行身份验证的API。
6. [AuthenticationProvider](https://docs.spring.io/spring-security/site/docs/5.4.7-SNAPSHOT/reference/html5/" \l "servlet-authentication-authenticationprovider)：此接口实现用于执行具体的身份验证
7. [Request Credentials with AuthenticationEntryPoint](https://docs.spring.io/spring-security/site/docs/5.4.7-SNAPSHOT/reference/html5/" \l "servlet-authentication-authenticationentrypoint)：
8. [AbstractAuthenticationProcessingFilter](https://docs.spring.io/spring-security/site/docs/5.4.7-SNAPSHOT/reference/html5/" \l "servlet-authentication-abstractprocessingfilter)：用于身份验证的基本过滤器。这还可以很好地了解高层次的身份验证流程以及各个部分如何协同工作。

# 权限认证

权限认证是通过FilterSecurityInterceptor来实现的，它是AbstractSecurityInterceptor的一个子类。AbstractSecurityInterceptor有权限决策获取，如，图-权限决策实现，所示：

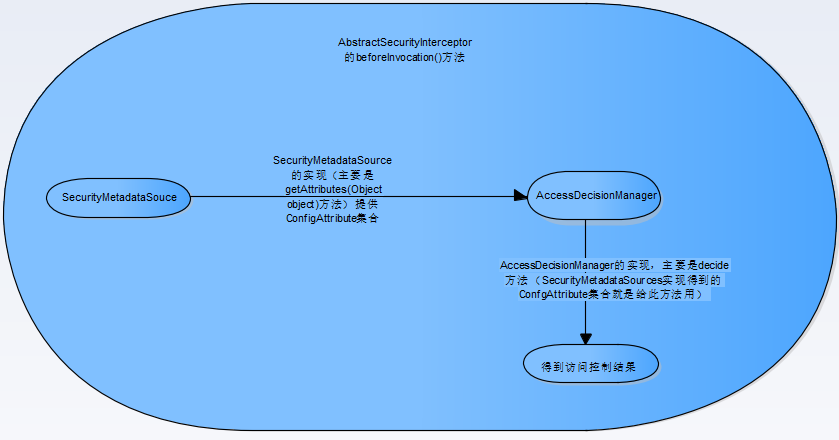


图-权限决策实现

FilterSecurityInterceptor（继承自AbstractSecurityInterceptor）：在此类中获取到需要的SecurityMetadataSouce

SecurityMetadataSouce(默认是使用的DefaultFilterInvocationSecurityMetadataSource)，如，图-SecurityMetaDataSource继承关系，所示:

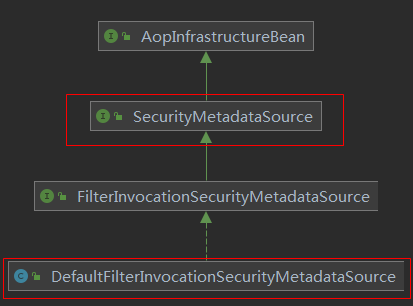


图-SecurityMetaDataSource继承关系

AccessDecisionManager（默认实现是AffirmativeBased，它继承于AbstractAccessDecisionManager(两个supports方法的实现在这里面)）决定了最终的访问控制决策。它由AbstractSecurityInterceptor调用，主要是体现在beforeInvocation(Object objec)方法中。

AccessDecisionManager的三个方法介绍：

1. **decide：**主要是处理访问控制决策
2. **supports(****ConfigAttribute attribute)：**此方法主要是用于定义配置的ConfigAttribute是否被允许（自定义实现AcceDescisionManager由开发者定义）。允许AbstractSecurityInterceptor检查配置的AccessDecisionManager和/或RunAsManager、和/或AfterInvocationManager，可以使用的每个配置属性。具体调用是在AbstractSecurityInterceptor的afterPropertiesSet()方法中。如果AccessDecisionManager支持传递的attribute则返回true。

AbstractSecurityInterceptor的afterPropertiesSet()方法里的对supports方法的调用，如图1-1：

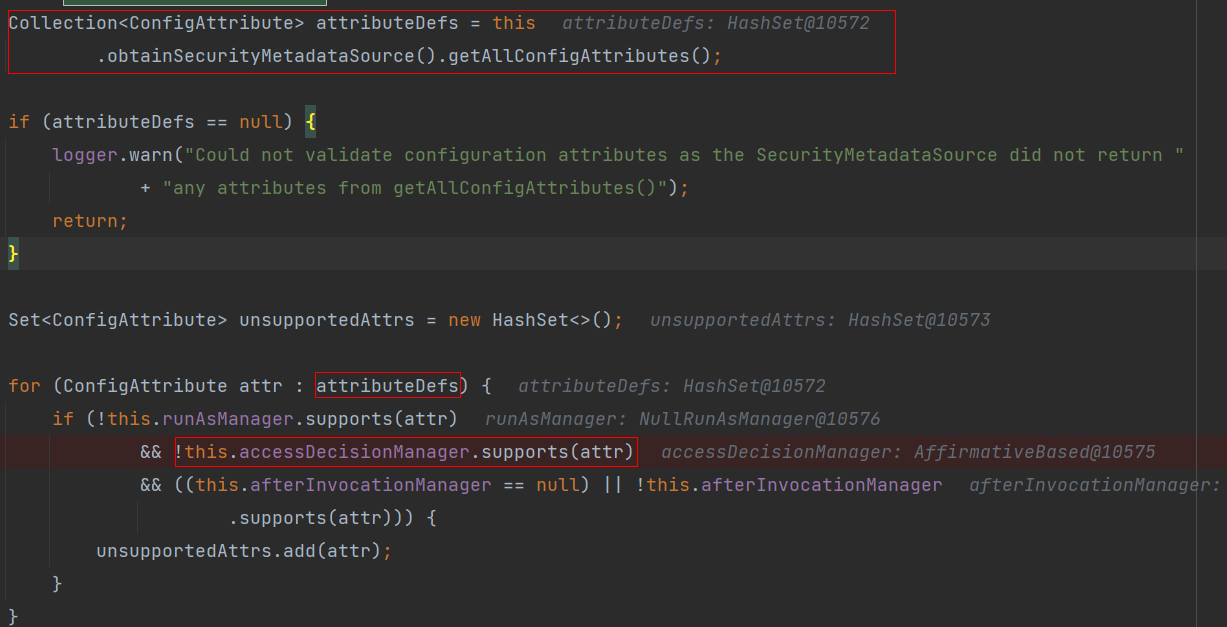


图1-1

3、**supports(Class<?> clazz)：**指示AccessDecisionManager实现是否能够为指定的受保护对象类型提供访问控制决策。具体使用可以查看AccessDecisionVoter实现的supports方法。此方法的调用也是在AbstractSecurityInterceptor的afterPropertiesSet()方法里，如图1-2

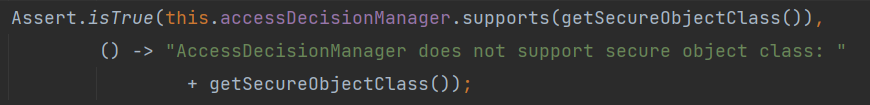


图1-2

AccessDecisionVoter（默认实现是WebExpressionVoter）：此接口定义权限的决策（允许、拒绝、弃权）

默认的访问控制决策实现是，AccessDecisionManager实现的decide方法调用所有AccessDecisionVoter实现的vote方法，然后再进行一些加工处理，最终得到访问控制权限的结果。

AccessDecisionManager和AccessDecisionVoter、ConfigAttribute之间的关系，如，图-关系图，所示：

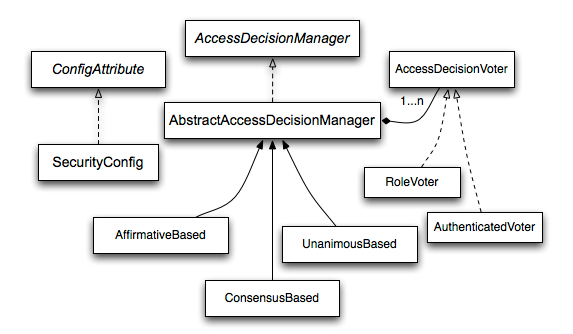


图-关系图

## 自定义权限认证

1. 可以采用自定义AccessDecisionVoter
2. 自定义实现AccessDecisionManager+FilterSecurityInterceptor
3. 只自定义实现AccessDecisionManager
4. 自定义实现FilterInvocationSecurityMetadataSource+AccessDecisionManager
5. 自定义实现FilterInvocationSecurityMetadataSource+AccessDecisionManager+ilterSecurityInterceptor
6. 自定义实现FilterInvocationSecurityMetadataSource+AccessDecisionManager

下面是使用自定义AccessDecisionManager的decide方法片段

