AccessDecisionVoter Türleri

RoleVoter

- Default kullanılan AccessDecisionVoter implemantasyonlarından biri **RoleVoter**'dır
- Eğer secure object'in herhangi bir **ConfigAttribute** değeri **ROLE_** ön ekine sahip ise **RoleVoter** yetkilendirme ile ilgili **olumlu veya olumsuz bir karar** verebilir
- O anki **kullanıcının sahip olduğu roller** ile **ConfigAttribute'daki roller karşılaştırılarak** eşleşen rollerin olup olmadığına bakılır

AuthenticatedVoter

- Yetki ifadeleri içerisinde aşağıdaki **placeholder**'ları işleyerek aktif kullanıcının erişimini denetler
  - IS_AUTHENTICATED_FULLY
  - IS_AUTHENTICATED_REMEMBERED
  - IS_AUTHENTICATED_ANONYMOUSLY
- **Default** durumda bu voter da AccessDecisionManager'a register edilmektedir

AuthenticatedVoter

```
<security:http use-expressions="false">

   <security:intercept-url pattern="/index.jsp"
access="IS_AUTHENTICATED_ANONYMOUSLY" />

   <security:intercept-url pattern="/**"
access="IS_AUTHENTICATED_FULLY" />

</security:http>
```

Spring Security 4 ile birlikte default durumda expression tabanlı access ifadeleri aktif olduğu için bu placeholder'ları kullanabilmek için expression kullanımı disable yapılmalıdır

WebExpressionVoter

- Web resource'ların expression tabanlı yetkilendirilmesinde **WebExpressionVoter** devreye girmektedir
- Yetki ifadeleri ise WebExpressionVoter içerisinde **SecurityExpressionHandler** tarafından ele alınmaktadır
- WebExpressionVoter bu amaçla **DefaultWebSecurityExpressionHandler** sınıfından bir nesne oluşturup kullanmaktadır

PreInvocationAuthorization AdviceVoter

- **PreAuthorize** anotasyonu ile belirtilen yetki ifadelerini ele almak için **PreInvocationAuthorizationAdviceVoter** kullanılmaktadır
- Bu voter içerisinde de iş **SecurityExpressionHandler**'a havale edilmektedir
- Default **SecurityExpressionHandler** implemantasyonu olarak **DefaultMethodSecurityExpressionHandler** kullanılmaktadır

RoleHierarchyVoter

- Bir rolün diğer bazı **rolleri içermesi** yaygın bir ihtiyaçtır
- Örneğin, **EDITOR** rolü kendisine ilaveten **READER** rolünün yaptığı herşeyi yapar, yada **ADMIN** rolü hem EDITOR hem de READER rollerini kapsar gibi
- Bir role verilen **erişim izinlerine** otomatik olarak bu rolü içeren **alt roller de sahip** olabilir
- **RoleHierarchyVoter** ile roller arasındaki hiyerarşik ilişkiler belirtilir

RoleHierarchyVoter Kullanımı

- AccessDecisionManager'ın voter olarak **RoleHierarchyVoter** türünde bir bean'i kullanması sağlanmalıdır
- **RoleHierarchyVoter** ve **AccessDecisionManager** bean'ları Spring konfigürasyon dosyasında tanımlanarak bu özelleştirme yapılabilir

RoleHierarchyVoter Konfigürasyonu

```
<bean id="roleHierarchyVoter"
class="org.springframework.security.access.vote.RoleH
ierarchyVoter">
   <constructor-arg ref="roleHierarchy" />
</bean>

<bean id="roleHierarchy"
class="org.springframework.security.access.hierarchic
alroles.RoleHierarchyImpl">
   <property name="hierarchy">
      <value>
         ROLE_ADMIN > ROLE_EDITOR
         ROLE_EDITOR > ROLE_USER
      </value>
   </property>
</bean>
```

RoleHierarchyVoter Konfigürasyonu

```
<bean id="accessDecisionManager"
class="org.springframework.security.access.vote.AffirmativeB
ased">
    <constructor-arg>
        <list>
            <ref bean="roleHierarchyVoter" />
        </list>
    </constructor-arg>
</bean>

<security:http access-decision-manager-
ref="accessDecisionManager">
...
</security:http>
```

AccessDecision Manager'ın manuel tanımlandığı durumda AuthenticatedVoter gibi voter'larında tanımlanması gerekebilir

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com