CUSTOM PERMISSIONS IN SPRING SECURITY

Custom FitnessPermissionEvaluator class that implements PermissionEvaluator interface

**package** com.pluralsight.security;

**import** java.io.Serializable;

**import** javax.sql.DataSource;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.security.access.PermissionEvaluator;

**import** org.springframework.security.core.Authentication;

**import** org.springframework.security.core.userdetails.User;

**public** **class** FitnessPermissionEvaluator **implements** PermissionEvaluator {

**private** DataSource dataSource;

**public** DataSource getDataSource() {

**return** dataSource;

}

**public** **void** setDataSource(DataSource dataSource) {

**this**.dataSource = dataSource;

}

**public** **boolean** hasPermission(Authentication authentication, Object targetDomainObject, Object permission) {

JdbcTemplate template=**new** JdbcTemplate(dataSource);

Object[] args= {((User)authentication.getPrincipal()).getUsername(),targetDomainObject.getClass().getName(),permission.toString()};

**int** count=template.queryForObject("select count(\*) from permissions p where p.username=? and p.target=? and p.permission=?", args,Integer.**class**);

**if**(count==1)

{

**return** **true**;

}

**else** {

**return** **false**;

}

}

**public** **boolean** hasPermission(Authentication authentication, Serializable targetId, String targetType,

Object permission) {

**return** **false**;

}

}

Go to the database :

Create a new table called permissions with a foreign key to username of users table and an index of unique of the three columns of username,target,permission.

Add a new row to the table with values : akshay,com.pluralsight.model.Goal , createGoal as permission

Go back to the application , we have a problem as in our Spring security context can see into the MVC context but not vice versa and all our jsp configurations are taken care of by the internal resource view resolver which is present in our servlet-config.xml file . Hence we need to specify our permission configuration as well as our global annotation configuration to use the pre-post annotations only within the servlet-config.xml

Servlet-config.xml : Add the following :

<security:global-method-security pre-post-annotations=*"enabled"*>

<security:expression-handler ref=*"fitnessExpressionHandler"*/>

</security:global-method-security>

<bean id=*"fitnessExpressionHandler"* class=*"org.springframework.security.access.expression.method.DefaultMethodSecurityExpressionHandler"*>

<property name=*"permissionEvaluator"*>

<bean id=*"permissionEvaluator"* class=*"com.pluralsight.security.FitnessPermissionEvaluator"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

</property>

Once we are done with this , now we have to go to our GoalController and specify a permission on a method level where we already specified our expression annotation of @PreAuthorize on the method to create a goal with requestMethod as POST.

GoalController.java

@PreAuthorize("hasRole('ROLE\_ADMIN') and hasPermission(#goal,'createGoal')")

@RequestMapping(value = "addGoal", method = RequestMethod.***POST***)

**public** String updateGoal(@Valid @ModelAttribute("goal") Goal goal, BindingResult result) {

System.***out***.println("result has errors: " + result.hasErrors());

System.***out***.println("Goal set: " + goal.getMinutes());

**if**(result.hasErrors()) {

**return** "addGoal";

}

**return** "redirect:index.jsp";

}

You can see that we have the @PreAuthorize annotation that has an expression defined which says that a user with the role as admin and with the permission to create a goal can only access that method and only then will the method be executed.

If either of the condition fails then the user will have a 405 error page displayed.

Custom Permission Evaluator – Step 1

Permission Evaluator Configuration in servlet-config.xml – Step 2

Specify the method level permission you want to give to the user with the particular role – Step 3

The interface PermissionEvaluator has two methods :

**public** **boolean** hasPermission(Authentication authentication, Object targetDomainObject, Object permission) {

// **TODO** Auto-generated method stub

**return** **false**;

}

**public** **boolean** hasPermission(Authentication authentication, Serializable targetId, String targetType,

Object permission) {

// **TODO** Auto-generated method stub

**return** **false**;

}