Bruce Phillips Blog on Java, ColdFusion, Flex and Spry

An Introduction to Shiro (formerly JSecurity) – A Beginner's Tutorial Part 5

Posted At : May 1, 2009 10:33 AM | Posted By : Bruce Phillips
Related Categories: **Java**

Introduction

NOTE: Updated in January 2011.

In **part 4 of this series**, I explained how you can use Shiro's tag library to control what is rendered in the JSPs. In **part 3 of this series**, I explained how you can use a users role to control access to specific parts of a web application. In this part of the series, I demonstrate how you can add another level of security to your web application by using permissions. Each role (e.g. admin, user) can have one or more permissions associated with it. Using Shiro you can restrict what someone can access based upon permission.

Using Permission to Control Access

In this example, our web application has three areas needing to be secured:

/users
/staff
/admin

Our business rules for securing each area are:

/users – user is authenticated and has role of user
/staff – user is authenticated and has role of staff or role of admin
/admin – user is authenticated and has role of admin

Since roles staff and admin both need to access the staff area, we can use permissions to control access to that area. We can assign the same permission to both the staff and admin role. We'll call that permission "secure." Any user with permission of "secure" can access the pages in the staff area. If in the future we add additional roles that need to access the staff area, we can give those new roles the "secure" permission also.

So our revised security business rules for each area are:

/users – user is authenticated and has role of user
/staff – user is authenticated and has permission of secure
/admin – user is authenticated and has role of admin

Here are our three users, their roles, and their permissions:

username	password	role	permission
bruce@hotmail.com	bruce	admin	secure
jack@hotmail.com	jack	staff	secure
sue@hotmail.com	sue	user	none

Applying our security business rules to these users we can identify who will be able to access which areas:

/user – only sue
/staff – both bruce and jack but not sue
/admin - only bruce

Example Application

You can **download an example application** (an archived Eclipse dynamic web project using Maven). You'll also need to **download a new version of the securityDB Derby database** that includes the roles_permissions table.

After downloading the securityDB Derby database, unzip it to the folder c:/derby. It's OK to overwrite the database that was there as this new version of the securityDB database should work for the previous example applications.

You can import the downloaded Eclipse archived project (named permissionsecuritywithtags) into Eclipse and then run it on a Tomcat server.

You can also use the Maven jetty plugin (see reference below for how to install Maven if you've don't already have Maven) to run the web application if you're not using Eclipse and Tomcat. Just open a command window and navigate to where you unzipped the permissionsecuritywithtags.zip download. Make sure you're in the permissionsecuritywithtags directory. Then do the following (in this example I unzipped permissionsecuritywithtags.zip to c:\jsecurity_examples):

ARCHIVES BY SUBJECT

ColdFusion (26) [RSS]
Flex (75) [RSS]
Java (76) [RSS]
Running (10) [RSS]
Spry (13) [RSS]

CALENDAR

<< August 2012 >>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

LATEST FROM CFBLOGGERS.ORG

Reading A Remote File With CFHTTP Using Railo And ColdFusion
Upcoming Appearances August 2012
Don't be Jerry Maguire
Building Explains Building Native Extension For iOS And Android
Reminder - Open Session on PhoneGap Tomorrow!

LINKS

Bruce Phillips' Main Web
Blog Home
Flex.org - The Directory for Flex

NAVIGATION

Home

RECENT ENTRIES

No recent entries.

RECENT COMMENTS

Android App Development - Using The ActionBar Widget
mike said: That was fantastic, been looking for that for a while now. thanks for putting in the effort. regards **[More]**

Using Blackboard Learn 9 Web Services - Part 4 Getting, Creating, Updating, Deleting Users
Bruce said: Baci - I"ve not used that method before and don't know much about course organizations. **[More]**

Using Blackboard Learn 9 Web Services - Part 4 Getting, Creating, Updating, Deleting Users
Baci said: Bruce, Thanks for being really the only resource as we're trying to implement some Bb webservic stuf... **[More]**

An Introduction to Shiro (formerly

```
c:\jsecurity_examples\ permissionsecuritywithtags \mvn clean
c:\jsecurity_examples\ permissionsecuritywithtags \mvn jetty:run
```

Once you see [INFO] Started Jetty Server in the command window, open your web browser and go to this URL: **http://localhost:8080/permissionsecuritywithtags/** . You should see the contents of the index.jsp. To stop the Jetty server type control-c in the command window.

Login as each user (bruce@hotmail.com, jack@hotmail.com, and sue@hotmail.com). When you're logged in as sue@hotmail.com try to go to either **http://localhost:8080/permissionsecuritywithtags/staff/index.jsp** or **http://localhost:8080/permissionsecuritywithtags/admin/index.jsp**. You should be redirected to the unauthorized web page. When you're logged in as jack@hotmail.com try to go to the **http://localhost:8080 /permissionsecuritywithtags/admin/index.jsp** page.

Implementing Permission Security in Shiro

So how do we implement security that uses permissions in Shiro? When you attempt to authenticate a user, Shiro will look for any permissions associated with the role that is associated with that user. To use Shiro's default configuration you just need a table named roles_permissions with a column named role_name and a column named permission. The default permissions query is specified in **class JdbcRealm**, which my class RoleSecurityJdbcRealm extended. In web.xml I specified this class as the value for the IniShiroFilter's realm (a realm is a resource that Shiro will use to authenticate users).

If your project cannot follow Shiro's defaults, you can configure Shiro to use your projects conventions (see the Shiro references below).

In Servlet class LoginUser I created a Subject object and call the Subject class's login method passing it the UsernamePasswordToken object that has the user's username and password. If Shiro can successfully authenticate this user by querying the users table, Shiro will then query the user_roles table for roles associated with this username, and then query the roles_permissions table for permissions associated with each role_name associated with this username. All of this information will be stored in the Subject object and placed into session scope. For more information on interfaces Subject and its associated interfaces AuthenticationInfo and and AuthorizationInfo see the **Shiro API**.

If you examine the web.xml you'll see the following statements in the configuration for the IniShiroFilter:

```
[main]
realmA = name.brucephillips.somesecurity.dao.RoleSecurityJdbcRealm
realmA.permissionsLookupEnabled=true

[filters]
roles.unauthorizedUrl = /unauthorized.jsp
perms.unauthorizedUrl = /unauthorized.jsp

#only let authenticated users
#with the appropriate role or permission
#view the web pages in the staff, user,
#and admin areas
[urls]
```

Under the [main] section the realmA.permissionsLookupEnabled=true configures Shiro to also lookup the permissions associated with a role. By default permissions lookup is false.

The statement perms.unauthorizedUrl = /unauthorized.jsp tells the filter to redirect people who attempt to view a web page in an area of the site for which they don't have the correct permission to the unauthorized.jsp web page.

The statement /staff/** = authc, perms[secure] means only allow people who have logged in successfully and have a permission of secure to view pages in the /staff folder.

In the Servlet LoginUser.java you'll find this statement around line number 134:

```
if (subject.isPermitted("secure") )
```

This statement uses the isPermitted method of the Subject class. This method returns true if the Subject object has the permission sent as the argument, otherwise the isPermitted method returns false.

Shiro also has a tag you can use to check a user's permission. In the JSP /index.jsp I use the hasPermission tag (around line 21) to render content for only those users with the secure permission.

Summary

Shiro is a comprehensive security library that gives you control over all aspects of your web application. By using permissions you can enable more specific access rules.

References

1. An Introduction to Shiro (formerly JSecurity) – A Beginner's Tutorial Part 4, **http://www.brucephillips.name /blog/index.cfm/2009/4/5/An-Introduction-to-Ki-formerly-JSecurity--A-Beginners--Tutorial-Part-4**
2. An Introduction to Shiro (formerly JSecurity) – A Beginner's Tutorial Part 3, **http://www.brucephillips.name /blog/index.cfm/2009/4/5/An-Introduction-to-Ki-formerly-JSecurity--A-Beginners--Tutorial-Part-3**
3. Permission Security With Tags Example Application, **http://www.brucephillips.name/jsecurity_examples /permissionsecuritywithtags_mvn.zip**
4. Apache Shiro **http://shiro.apache.org/**
5. Apache Shiro API, **http://shiro.apache.org/static/current/apidocs/**
6. Apache Shiro Tags API, **http://shiro.apache.org/static/current/apidocs/org/apache/shiro/web/tags /package-summary.html**

JSecurity) – A Beginner's Tutorial Part 4
mark said: In secure/index.jsp had to change jsec:principal to shiro:principal. **[More]**

NetBeans Java IDE 7.2 - Comments From An Eclipse User
Ralf Eichinger said: I used Eclipse for years now and tried to switch to Netbeans 7.1.2. Basically Netbeans does the job,... **[More]**

RSS

RSS FEED

SEARCH

Search

SUBSCRIBE

Enter your email address to subscribe to this blog.

Subscribe

TAGS

coldfusion flex java running spry

7. Apache Shiro Mailing Lists, **http://shiro.apache.org/mailing-lists.html**
8. Shiro Custom Tags TLD, **http://www.brucephillips.name/jsecurity_examples/ki%20(jsecurity)%20tld.pdf**
9. Using Custom Tags, J2EE Tutorial, **http://java.sun.com/javaee/5/docs/tutorial/doc/bnaiy.html**
10. Apache Derby, **http://db.apache.org/derby/**
11. Apache Tomcat, **http://tomcat.apache.org/**
12. Jetty, **http://jetty.mortbay.org/jetty5/index.html**
    tp://incubator.apache.org/projects/ki.html
13. Maven: The Definitive Guide, **http://www.sonatype.com/books/maven-book/reference/public-book.html**
14. Developing with Eclipse and Maven, **http://www.sonatype.com/books/m2eclipse-book/reference/index.html**

Comments (6) | **Print** | **Send** | **del.icio.us** | **Digg It!** | **Linking Blogs** | 9036 Views

Related Blog Entries

- **An Introduction to Shiro (formerly JSecurity) – A Beginner's Tutorial Part 4** (April 5, 2009)

Comments (Comment Moderation is enabled. Your comment will not appear until approved.)

[**Add Comment**] [**Subscribe to Comments**]

I downloaded the sample project and found it misspelled as "persmissionsecuritywithtags". To get this to run you will need to use **http://localhost:8080/persmissionsecuritywithtags/...**

 # Posted By Gordon Dickens | 5/5/09 2:12 PM

Gordon - thank you for letting me know about the problem.

I uploaded a new archived project with the correct artifactID in the pom.xml. You should be able to download that project, unzip, and run the mvn commands as specified in the article.

Then load the application using this URL:

http://localhost:8080/permissionsecuritywithtags/

 # Posted By Bruce | 5/5/09 7:25 PM

I wanted to convert the demo over to use the Apache Ki classes (replacing the former JSecurity) and had to make the following changes:
1. changed the pom.xml to import the following dependencies:
org.apachi.key/ki-all/1.0-incubating-SNAPSHOT
org.slf4j/slf4j-simple/1.5.6
commons-beanutils/commons-beanutils/1.7.0
commons-logging/commons-logging/1.1.1

2. Changed the tag URL in the jsp pages to:
<%@ taglib prefix="jsec" uri="**http://ki.apache.org/tags**"; %>

3. Changed the JSecurityFilter in web.xml
<filter-name>KiFilter</filter-name>
<filter-class>org.apache.ki.web.servlet.KiFilter</filter-class>
...
<filter-mapping>
<filter-name>KiFilter</filter-name>
...

4. Modified all the Java classes to use the classes from package: org.apache.ki.*

Hope this helps.

 # Posted By Gordon Dickens | 5/8/09 10:22 AM

Gordon - thank you for posting how to convert an application to use the Apache Ki classes. I need to do this for a project at my job so what you've done is very helpful.

 # Posted By Bruce | 5/8/09 11:33 AM

can you please provide DDL to define shiro database tables?

 # Posted By Larry | 6/19/12 8:54 AM

Gordon - try generating the DDL from the securityDB database, which is a Derby database. Most IDE's have a database plugin that will work with Derby databases.

I don't have the DDL handy.

Bruce

 # Posted By Bruce | 6/19/12 10:23 AM

[**Add Comment**]

BlogCFC was created by **Raymond Camden**. This blog is running version 5.9.1.002. **Contact Blog Owner**