Authentication and Authorization

Overview

The Fedora 4 Authentication (AuthN) and Authorization (AuthZ) framework is designed to be flexible and extensible, to allow any organization to configure access to suit its needs.

The following sections explain the Fedora 4 AuthN/Z framework, and provide instructions for configuring some out-of-the-box access controls.

For clarity's sake, a distinction is made between Authentication and Authorization:

- **Authentication** answers the question "who is the person, and how do I verify that they are who they say they are?" Fedora 4 relies on the web servlet container to answer this question.
- **Authorization** answers the question, "does this person have permission to do what they want to do?". Fedora 4 provides four different ways to answer this question:
  - Bypass authorization. Anyone who has authenticated through the web application container (Tomcat, Jetty, WebSphere, etc.) has permission to do everything – in effect all, authenticated users are superusers.
  - WebAC authorizations. Authenticated users' access to resources is mediated by WebAC Access Control Lists stored in the repository.
  - **[Deprecated]** Basic Access Roles authorizations (RBACL). Authenticated users are mapped onto one or more preconfigured roles; a user's role determines what they have permission to do.
  - **[Deprecated]** XACML authorizations. Policies created using the XACML framework are used to determine what operations are permissible to whom, using user and resource properties exposed to the XACML engine.

Servlet Container Authentication Configuration

Fedora 4 uses servlet container authentication (Realms) to provide minimal protection for your repository, including the set up of "superuser" accounts. User credentials are configured in your web application container, usually in a properties file or XML file. By configuring superuser accounts you can require authentication for all management (write) operations. This document describes how to set up Fedora and either Tomcat or Jetty to enable HTTP Basic Authentication, using simple user files. Consult your web application server documentation for other ways to configure and manage users; Fedora can handle any user principal passed to it by the servlet container, as provisioned by any of the container's supported authentication mechanisms.

The superuser role is **fedoraAdmin**. This is comparable to the **fedoraAdmin** superuser role in Fedora 3, used for Fedora 3 API-M operations.

The Fedora authorization modules reside in separate source code modules from the core Fedora web-application.

- WebAC : https://github.com/fcrepo4/fcrepo-module-auth-webac
- RBACL : https://github.com/fcrepo4/fcrepo-module-auth-rbacl
- XACML : https://github.com/fcrepo4/fcrepo-module-auth-xacml

As of Fedora 4.7.4, the RBACL and XACML authorization modules are officially deprecated, and will not be included in future releases of Fedora. Subsequent Fedora releases will only include the WebAC authorization module.

As a result, each release includes pre-built Fedora "webapp-plus" war files that have the authorization modules included. You are recommended to use one of these "webapp-plus" war files as a starting point for having an authorization-enabled deployment.

- Webapp-plus releases: https://github.com/fcrepo4-exts/fcrepo-webapp-plus/releases

You can then follow the guidelines in the Best Practices - Fedora Configuration document to specify site-specific "repo.xml" and "repository.json" configurations, as further described below.

1. ## Configure your repo.xml file

   Add the beans *authenticationProvider* and *fad* to your repo.xml file, and make the *modeshapeRepofactory* bean dependent on *authenticationProvider*. Use the class **org.fcrepo.auth.ServletContainerAuthenticationProvider** as your authentication provider. Here is an example repo.xml that configures authentication and authorization using the Basic Roles authorization delegate.

   To specify a local repo.xml configuration, provide the system property as follows:

   ```
   JAVA_OPTS="... -Dfcrepo.spring.repo.configuration=file:/local/repo.xml"
   ```

   repo.xml with authentication configured

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:util="http://www.springframework.org/schema/util"
       xsi:schemaLocation="
       http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-
   3.0.xsd
       http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
   context-3.0.xsd
       http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util.
   xsd">

       <!-- Context that supports the actual ModeShape JCR itself -->

       <context:annotation-config/>

       <bean name="modeshapeRepofactory"
           class="org.fcrepo.kernel.modeshape.spring.ModeShapeRepositoryFactoryBean"
           p:repositoryConfiguration="${fcrepo.modeshape.configuration:classpath:/config/servlet-auth
   /repository.json}"
           depends-on="authenticationProvider"/>

       <bean class="org.modeshape.jcr.ModeShapeEngine" init-method="start"/>

       <bean id="connectionManager" class="org.apache.http.impl.conn.PoolingHttpClientConnectionManager" />

       <!-- Optional PrincipalProvider that will inspect the request header, "some-header", for user role
   values -->
       <bean name="headerProvider" class="org.fcrepo.auth.common.HttpHeaderPrincipalProvider">
           <property name="headerName" value="some-header"/>
           <property name="separator" value=","/>
       </bean>

       <util:set id="principalProviderSet">
           <ref bean="headerProvider"/>
       </util:set>

       <bean name="fad" class="org.fcrepo.auth.roles.basic.BasicRolesAuthorizationDelegate"/>

       <bean name="authenticationProvider" class="org.fcrepo.auth.common.
   ServletContainerAuthenticationProvider">
           <property name="fad" ref="fad"/>
           <property name="principalProviders" ref="principalProviderSet"/>
       </bean>

       <!-- For the time being, load annotation config here too -->
       <bean class="org.fcrepo.metrics.MetricsConfig"/>
   </beans>
   ```

2. ## Configure your repository.json file

   Modify the security section to enable both authenticated (via authentication provider) and internal sessions between Fedora and ModeShape.

   To specify a local repository.json configuration, provide the system property as follows:

   ```
   JAVA_OPTS="... -Dfcrepo.modeshape.configuration=file:/local/repository.json"
   ```

   It should contain a "security" element that matches this block:

   repository.json security

   ```
   "security" : {
           "anonymous" : {
               "roles" : ["readonly","readwrite","admin"],
               "useOnFailedLogin" : false
           },
           "providers" : [
               { "classname" : "org.fcrepo.auth.common.ServletContainerAuthenticationProvider" }
           ]
       },
   ```

3. ## Configure your web.xml

   Configure your **web.xml**.

   Modify fcrepo-webapp/src/main/webapp/WEB-INF/web.xml by uncommenting the security configuration

   ```
   <!--Uncomment section below to enable Basic-Authentication-->
   <security-constraint>
     <web-resource-collection>
       <web-resource-name>Fedora4</web-resource-name>
       <url-pattern>/*</url-pattern>
       <http-method>DELETE</http-method>
       <http-method>PUT</http-method>
       <http-method>HEAD</http-method>
       <http-method>OPTIONS</http-method>
       <http-method>PATCH</http-method>
       <http-method>GET</http-method>
       <http-method>POST</http-method>
     </web-resource-collection>
     <auth-constraint>
       <role-name>fedoraUser</role-name>
       <role-name>fedoraAdmin</role-name>
     </auth-constraint>
     <user-data-constraint>
       <transport-guarantee>NONE</transport-guarantee>
     </user-data-constraint>
   </security-constraint>
   <login-config>
     <auth-method>BASIC</auth-method>
     <realm-name>fcrepo</realm-name>
   </login-config>
   ```

   The "auth-constraint" element must contain the roles defined as your users (see below for jetty and tomcat).

4. ## Configure your web application container

   - **Jetty**
     - Create your **jetty-users.properties** file. This file contains entries in the format *username: password [, role, ...]*, where
       - *username* is the user's login id (the principal)
       - *password* is the user's password
       - *role* is the servlet role they are assigned upon login; jetty allows you to specify any number of roles (or no role at all). Fedora currently supports two roles: **fedoraAdmin**, which is the superuser role, and has rights to do everything; and **fedoraUser**, which is a user role, and must be granted permissions by the Policy Enforcement Point to perform actions.

Sample **jetty-users.properties** file that contains three users, two of whom are regular users, and the third of whom (fedoraAdmin) is a Fedora superuser:

jetty-users.properties

```
testuser: password1,fedoraUser
adminuser: password2,fedoraUser
fedoraAdmin: secret3,fedoraAdmin
```

- Configure your Jetty login realm.
  - Standalone
    Modify your **jetty.xml** file to configure the login realm and include the jetty-users.properties file:

    jetty.xml login service

```
<Configure class="org.eclipse.jetty.webapp.WebAppContext">

  <!-- Set this to the webapp root of your Fedora 4 repository -->
  <Set name="contextPath">/</Set>
  <!-- Set this to the path of of fcrepo4 WAR file -->
  <Set name="war"><SystemProperty name="jetty.home" default="."/>/webapps/fcrepo4</Set>

  <Get name="securityHandler">
    <Set name="loginService">
      <New class="org.eclipse.jetty.security.HashLoginService">
        <Set name="name">fcrepo4</Set>
        <!-- Set this to the path to your jetty-users.properties file -->
        <Set name="config"><SystemProperty name="jetty.home" default="."/>/path/to
/jetty-users.properties</Set>
      </New>
    </Set>
  </Get>

</Configure>
```

  - Embedded in Maven
    - The fcrepo-webapp Maven project includes jetty-maven-plugin. The property *jetty.users.file* sets the location of the **jetty-users.properties** file. Run the fcrepo-webapp server with the following system property:

      ```
      -Djetty.users.file=/path/to/jetty-users.properties
      ```

See the Jetty Authentication documentation for more details.

Tomcat

- Create or edit your `$CATALINA_HOME/conf/tomcat-users.xml` file. It has entries of the form

  ```
  <user name="principal" password="password" roles="role1, role2, ..." />
  ```

  where:

  - *name* is the user's login id (the principal)
  - *password* is the user's password
  - *roles* are the servlet roles they are assigned upon login; tomcat allows you to specify any number of roles (or no role at all). Fedora currently supports two roles: **fedoraAdmin**, which is the superuser role, and has rights to do everything; and **fedoraUser**, which is a user role, and must be granted permissions by the Policy Enforcement Point to perform actions.

  Sample **tomcat-users.xml** file that contains three users, two of whom are regular users, and the third of whom (fedoraAdmin) is a Fedora superuser:

tomcat-users.xml

```
<tomcat-users>
  <role rolename="fedoraUser" />
  <role rolename="fedoraAdmin" />
  <user name="testuser" password="password1" roles="fedoraUser" />
  <user name="adminuser" password="password2" roles="fedoraUser" />
  <user name="fedoraAdmin" password="secret3" roles="fedoraAdmin" />
</tomcat-users>
```

- Configure your Tomcat login realm.
  Modify your file `$CATALINA_HOME/conf/server.xml` file to configure the login realm with the Fedora 4 webapp context:

Tomcat server.xml Realm

```
<Context>
...
  <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
         resourceName="UserDatabase" />
</Context>
```

See the Tomcat Realms documentation for more details.

Bypass Authorization

Running Fedora without authorization means that the REST API is available to any request coming from the container and lacks any finer-grained security. This is useful when Fedora is running behind another application that connects to Fedora and implements its own security checks. In addition, this configuration is useful for temporary demonstrations and for running software tests that do not require security.

This configuration does not preclude the use of container authentication to secure Fedora. However, container roles are not used for any further authorization within Fedora. All requests are treated as superusers.

The security bypass for REST endpoint is accomplished by supplying an alternate ModeShape authentication provider for servlet credentials. This servlet authentication provider permits all actions at the Modeshape level and does not use a PEP (Policy Enforcement Point).

Step-by-Step Configuration

1. If you previously configured a PEP, open your repo.xml file and remove any beans that are instances of "org.fcrepo.auth.common.ServletContainerAuthenticationProvider".
2. Also remove the PEP bean, if one was configured.
3. Remove the depends-on attribute on the modeshapeRepofactory bean, if there is one.
4. Open your repository.json file
5. Under security, configure the "BypassSecurityServletAuthenticationProvider", as shown in the example below.

Example repository.json (security section)

```
"security" : {
  "anonymous" : {
    "roles" : ["readonly","readwrite","admin"],
    "useOnFailedLogin" : false
  },
  "providers" : [
    { "classname" : "org.fcrepo.auth.commons.BypassSecurityServletAuthenticationProvider" }
  ]
},
```

Authorization Delegates

Overview

Fedora Authorization Delegates allow you to implement one interface to enforce access control over your Fedora repository. This interface, FedoraAuthorizationDelegate, has callbacks that allow you to restrict ModeShape operations and filter search results. After following these configuration steps, Fedora's REST endpoints will respond with 403 response codes when the requested action is unauthorized by the authorization delegate.

Use of an authorization delegate and Fedora-specific authorization is optional. You can also configure Fedora to run without API security. You may want to only enforce container authentication or leave the service running completely unsecured, behind a firewall for instance. For details, see How to configure Fedora without authorization.

Fedora Administrators (fedoraAdmin user role)

The authorization delegate is not consulted when servlet credentials identify a client with the **fedoraAdmin** role. When the container has authenticated the connected client as a **fedoraAdmin**, all actions are permitted and PEP is bypassed.

FedoraAuthorizationDelegate Implementations

There are three reference implementations available:

- WebAC Authorization Delegate
- **[Deprecated]** Basic Role-based Authorization Delegate - An authorization delegate that operates on three fixed roles that may be assigned throughout the repository tree. (reader, writer, admin)
- **[Deprecated]** XACML Authorization Delegate

You can also create an authorization delegate implementation and perform security checks differently, possibly including calls to remote services.

Configuration

Two files contain the configuration options for authorization delegates:

- **repo.xml**: the global repository configuration file. Three beans enable the PEP extension:
  - *modeshapeRepoFactory*: should contain a dependency on the *authenticationProvider* bean
  - *authenticationProvider*: should specify the *ServletContainerAuthenticationProvider* class, so that the servlet container handles authentication
    - This bean should have a property "fad" that points to the *fad* bean, to enable the servlet container authentication provider to use the authorization delegate
  - *fad*: should point to your class with the authorization delegate implementation
- **repository.json**: the ModeShape configuration file. It contains a security section, where the **internal session** authentication between Fedora and the ModeShape storage layer is configured. Note that the roles configured here do not apply to end user authentication and authorization..

Step-by-step Configuration

1. Open the repo.xml file in your Fedora web application.
2. Add your authorization delegate implementation as a bean in this file and give it the ID of "fad". Your authorization delegate bean may include more specific configuration details than the example.
3. Now add the Fedora ModeShape Authentication Provider bean. (see repo.xml example)
4. Make sure that your modeshapeRepofactory bean has the depends-on attribute pointing at the authenticationProvider (see repo.xml example).
5. Open your repository.json file.
6. Add org.fcrepo.auth.ServletContainerAuthenticationProvider as a provider in the security section. (see repository.json example)

Example repo.xml (repository and security beans)

```
<bean name="modeshapeRepofactory" class="org.fcrepo.kernel.spring.ModeShapeRepositoryFactoryBean"
  depends-on="authenticationProvider">
    <property name="repositoryConfiguration" value="${fcrepo.modeshape.configuration:repository.json}" />
</bean>
<bean name="fad" class="your.own.implementation"/>
<bean name="authenticationProvider" class="org.fcrepo.auth.ServletContainerAuthenticationProvider">
  <property name="fad" ref="fad"/>
</bean>
```

Example repository.json (security section)

```
"security" : {
  "anonymous" : {
    "roles" : ["readonly","readwrite","admin"],
    "useOnFailedLogin" : false
  },
  "providers" : [
    { "classname" : "org.fcrepo.auth.ServletContainerAuthenticationProvider" }
  ]
},
```

WebAC Authorization Delegate

WebAC Authorization Delegate

Basic Role-based Authorization Delegate

As of Fedora 4.7.4, the RBACL authorization module is officially deprecated, and will not be included in future releases of Fedora. Subsequent Fedora releases will only include the WebAC authorization module.

Basic Role-based Authorization Delegate (RBACL)

Access Roles Module

XACML Authorization Delegate

As of Fedora 4.7.4, the XACML authorization module is officially deprecated, and will not be included in future releases of Fedora. Subsequent Fedora releases will only include the WebAC authorization module.

XACML Authorization Delegate