Configuring run-as and Default Subjects, and principal-role mapping

Introduction

Starting from version 2.0.1, Geronimo adopts the basic principle that all security flows from Subjects that result from logging in to a security realm. In previous Geronimo releases, security information for run-as and default subjects was constructed entirely outside any security realm. As a result of following the new principle, run-as and default identities can now participate fully in security using such features as named credentials to access such external systems as connectors and web services, and the JACC system is now more fully pluggable.

However, because run-as and default subjects now result from logging in to a security realm, to use such a subject you need to supply the login information for each such subject. This information is encapsulated in a CredentialStore. We supply a simple CredentialStore implementation using XML in your Geronimo plan. Note that this includes plain text passwords for the run-as and default subjects. This might not be a suitable implementation for many environments.

Each application can choose to use a default, global, credential store or specify a specific store, perhaps specific to that application.

Configuring a SimpleCredentialStoreImpl

For each Subject accessible through a credential store, you need to specify an id, the realm to log in to, and credentials, which depend on the security realm requirements but are typically the name and password. The schema is as follows:

Error rendering macro 'code': Invalid value specified for parameter 'java.lang.NullPointerException'

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    Licensed to the Apache Software Foundation (ASF) under one or more
    contributor license agreements.  See the NOTICE file distributed with
    this work for additional information regarding copyright ownership.
    The ASF licenses this file to You under the Apache License, Version 2.0
    (the "License"); you may not use this file except in compliance with
    the License.  You may obtain a copy of the License at

        http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.
-->

<!-- $Rev$ $Date$ -->

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:cs="http://geronimo.apache.org/xml/ns/credentialstore-1.0"
            targetNamespace="http://geronimo.apache.org/xml/ns/credentialstore-1.0"
            elementFormDefault="qualified" attributeFormDefault="unqualified"
            version="1.0">

    <xsd:annotation>
        <xsd:documentation>
            This is an XML Schema Definition for credential store configuration.
            CredentialStore configuration is
            specified by the element credential-store with namespace
            specified as xmlns =
            "http://geronimo.apache.org/xml/ns/credentialstore-1.0".
        </xsd:documentation>
    </xsd:annotation>

    <xsd:element name="credential-store" type="cs:credential-storeType">
        <xsd:annotation>
            <xsd:documentation>
                The root element for Geronimo credential store configuration. This
                is a tree structure of realm, id, and sets of credentials such as name and password
            </xsd:documentation>
        </xsd:annotation>
    </xsd:element>

    <xsd:complexType name="credential-storeType">
        <xsd:annotation>
            <xsd:documentation>
                Defines the list of realms
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
```

```
        <xsd:element name="realm" type="cs:realmType" minOccurs="0" maxOccurs="unbounded">
            <xsd:annotation>
                <xsd:documentation>
                    The realm element contains the credentials for subjects in that realm.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="realmType">
    <xsd:sequence>
        <xsd:element name="subject" type="cs:subjectType" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required">
        <xsd:annotation>
            <xsd:documentation>
                The name attribute specifies the login realm name
            </xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="subjectType">
    <xsd:sequence>
        <xsd:element name="id" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                    The id element serves to identify the subject externally. For subjects with meaningful
                    names it might be convenient to use the name as id.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="credential" type="cs:credentialType" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="credentialType">
    <xsd:sequence>
        <xsd:element name="type" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                    Class name or alias of the callback handler that will accept this credential
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
        <xsd:element name="value" type="xsd:string">
            <xsd:annotation>
                <xsd:documentation>
                    credential value as a string.
                </xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>


</xsd:schema>
```

At the moment, Geronimo supplies callback handlers for name and password. For other security realm requirements (e.g. certificates), you will have to write a callback handler.

A simple example of credential store configuration would look like this:

Credential Store Example

```
    <gbean name="CredentialStore" class="org.apache.geronimo.security.credentialstore.SimpleCredentialStoreImpl"
>
        <xml-attribute name="credentialStore">
            <credential-store xmlns="http://geronimo.apache.org/xml/ns/credentialstore-1.0">
                <realm name="my-properties-realm">
                    <subject>
                        <id>admin-run-as</id>
                        <credential>
                            <type>org.apache.geronimo.security.credentialstore.NameCallbackHandler</type>
                            <value>system</value>
                        </credential>
                        <credential>
                            <type>org.apache.geronimo.security.credentialstore.PasswordCallbackHandler</type>
                            <value>manager</value>
                        </credential>
                    </subject>
                    <subject>
                        <id>user-run-as</id>
                        <credential>
                            <type>org.apache.geronimo.security.credentialstore.NameCallbackHandler</type>
                            <value>user</value>
                        </credential>
                        <credential>
                            <type>org.apache.geronimo.security.credentialstore.PasswordCallbackHandler</type>
                            <value>user-password</value>
                        </credential>
                    </subject>
                    <subject>
                        <id>default</id>
                        <credential>
                            <type>org.apache.geronimo.security.credentialstore.NameCallbackHandler</type>
                            <value>default</value>
                        </credential>
                        <credential>
                            <type>org.apache.geronimo.security.credentialstore.PasswordCallbackHandler</type>
                            <value>default</value>
                        </credential>
                    </subject>
                </realm>
            </credential-store>
        </xml-attribute>
    </gbean>
```

Again, note that the PasswordCallbackHandler value element contains a plain text password for the user.

Configuring your application to use a particular CredentialStore

Note that this aspect of Geronimo security is completely pluggable and only the default implementation is described here.

Geronimo security for JavaEE applications requires including a <security> element in (one of) the GHeronimo plans for your application. This describes the principal-role mappings to connect the Subjects from your security realm to the roles used in the spec deployment descriptors (and annotations). It also describes how to interpret run-as roles as subjects through specifying a credential store and the id and realm for each role used as a run-as. Similarly a default subject can be specified in the credential store.

The schema for security configuration is as follows:

Error rendering macro 'code': Invalid value specified for parameter 'java.lang.NullPointerException'

```
<?xml version="1.0" encoding="UTF-8"?>
<!--

    Licensed to the Apache Software Foundation (ASF) under one or more
    contributor license agreements.  See the NOTICE file distributed with
    this work for additional information regarding copyright ownership.
```

```
    The ASF licenses this file to You under the Apache License, Version 2.0
    (the "License"); you may not use this file except in compliance with
    the License.  You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.
-->

<!-- $Rev$ $Date$ -->

<xsd:schema
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:j2ee="http://java.sun.com/xml/ns/j2ee"
        xmlns:geronimo="http://geronimo.apache.org/xml/ns/security-2.0"
        targetNamespace="http://geronimo.apache.org/xml/ns/security-2.0"
        xmlns:app="http://geronimo.apache.org/xml/ns/j2ee/application-2.0"
        xmlns:sys="http://geronimo.apache.org/xml/ns/deployment-1.2"
        elementFormDefault="qualified"
        attributeFormDefault="unqualified"
        version="2.0">

    <xsd:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation="http://www.w3.org/2001/xml.xsd"/>
    <xsd:import namespace="http://geronimo.apache.org/xml/ns/j2ee/application-2.0" schemaLocation="geronimo-
application-2.0.xsd"/>
    <xsd:import namespace="http://geronimo.apache.org/xml/ns/deployment-1.2"  schemaLocation="geronimo-module-1.2.
xsd"/>

    <xsd:element name="security" type="geronimo:securityType" substitutionGroup="app:security"/>
    <xsd:element name="credential-store" type="sys:patternType"/>
    <xsd:element name="default-subject" type="geronimo:subject-infoType"/>

    <xsd:complexType name="securityType">
        <xsd:annotation>
            <xsd:documentation>
                Security entries

                If this element is present, all web and EJB modules MUST make the
                appropriate access checks as outlined in the JACC spec.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension base="app:abstract-securityType">

                <xsd:sequence>
                    <xsd:element name="description" type="geronimo:descriptionType" minOccurs="0"
                                 maxOccurs="unbounded"/>
                    <xsd:element name="credential-store-ref" type="sys:patternType" minOccurs="0"/>
                    <xsd:element name="default-subject" type="geronimo:subject-infoType" minOccurs="0"/>
                    <xsd:element name="role-mappings" type="geronimo:role-mappingsType" minOccurs="0"/>
                </xsd:sequence>
                <xsd:attribute name="doas-current-caller" type="xsd:boolean" default="false">
                    <xsd:annotation>
                        <xsd:documentation>
                            Set this attribute to "true" if the work is to be performed
                            as the calling Subject.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
                <xsd:attribute name="use-context-handler" type="xsd:boolean" default="false">
                    <xsd:annotation>
                        <xsd:documentation>
                            Set this attribute to "true" if the installed JACC policy
                            contexts will use PolicyContextHandlers.
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
                <xsd:attribute name="default-role" type="xsd:string">
                    <xsd:annotation>
                        <xsd:documentation>
                            Used by the the Deployer to assign method permissions for
                            all of the unspecified methods, either by assigning them
                            to security roles, or by marking them as unchecked. If
                            the value of default-role is empty, then the unspecified
                            methods are marked unchecked
                        </xsd:documentation>
                    </xsd:annotation>
                </xsd:attribute>
            </xsd:extension>
        </xsd:complexContent>
```

```
    </xsd:complexType>

    <xsd:complexType name="descriptionType">
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute ref="xml:lang"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>

    <xsd:complexType name="named-username-password-credentialType">
        <xsd:sequence>
            <xsd:element name="name" type="xsd:string"/>
            <xsd:element name="username" type="xsd:string"/>
            <xsd:element name="password" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="role-mappingsType">
        <xsd:sequence>
            <xsd:element name="role" type="geronimo:roleType" minOccurs="1" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="roleType">
        <xsd:sequence>
            <xsd:element name="description" type="geronimo:descriptionType" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="run-as-subject" type="geronimo:subject-infoType" minOccurs="0"/>
            <xsd:element name="realm-principal" type="geronimo:realmPrincipalType" minOccurs="0" maxOccurs="
unbounded"/>
            <xsd:element name="login-domain-principal" type="geronimo:loginDomainPrincipalType" minOccurs="0"
                         maxOccurs="unbounded"/>
            <xsd:element name="principal" type="geronimo:principalType" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="distinguished-name" type="geronimo:distinguishedNameType" minOccurs="0"
                         maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="role-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <xsd:complexType name="realmPrincipalType">
        <xsd:complexContent>
            <xsd:extension base="geronimo:loginDomainPrincipalType">
                <xsd:attribute name="realm-name" type="xsd:string" use="required"/>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="loginDomainPrincipalType">
        <xsd:complexContent>
            <xsd:extension base="geronimo:principalType">
                <xsd:attribute name="domain-name" type="xsd:string" use="required"/>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="principalType">
        <xsd:sequence>
            <xsd:element name="description" type="geronimo:descriptionType" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="class" type="xsd:string" use="required"/>
        <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <xsd:complexType name="distinguishedNameType">
        <xsd:sequence>
            <xsd:element name="description" type="geronimo:descriptionType" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <xsd:complexType name="subject-infoType">
        <xsd:sequence>
            <xsd:element name="description" type="geronimo:descriptionType" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="realm" type="xsd:string"/>
            <xsd:element name="id" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>

    <!--<xsd:complexType name="credential-storeType">-->
        <!--<xsd:sequence>-->
            <!--<xsd:element name="pattern" type="sys:patternType">-->
                <!--<xsd:annotation>-->
                    <!--<xsd:documentation>-->
                        <!--The pattern element defines a components of the-->
```

```
                        <!--abstract name of GBean referred. It (optionally) includes-->
                        <!--the groupId, artifactId, version,-->
                        <!--module, type, and name of the GBean module.-->
                    <!--</xsd:documentation>-->
                <!--</xsd:annotation>-->
            <!--</xsd:element>-->
        <!--</xsd:sequence>-->
    <!--</xsd:complexType>-->

</xsd:schema>
```

The credential store to use is specified in the credential-store-ref. Normally you only need only supply the name component of the credential store name: for most purposes you are likely to include an app specific credential store in the application plan, but otherwise you need to assure that the credential store gbean is in the ancestor configurations of the application.

A default subject or each run-as role specifies the information needed to get the subject using a subject-infoType element.

Example Security Configuration

```
      <security use-context-handler="false" xmlns="http://geronimo.apache.org/xml/ns/security-2.0">
        <default-subject>
          <realm>my-properties-realm</realm>
          <id>default</id>
        </default-subject>
        <role-mappings>
          <role role-name="Administrator">
            <principal class="org.apache.geronimo.security.realm.providers.GeronimoUserPrincipal" name="system"
/>
          </role>
          <role role-name="User">
            <run-as-subject>
                <realm>my-properties-realm</realm>
                <id>user-run-as</id>
            </run-as-subject>the loi
            <principal class="org.apache.geronimo.security.realm.providers.GeronimoGroupPrincipal" name="user"/>
          </role>
        </role-mappings>
      </security>
```

The sample above shows the simplest principal-role mapping: you specify the principal class and name for each principal that maps to a certain role. Normally this will be entirely sufficient to distinguish principals. However, you might have several login modules or security realms that can produce the same principal but with different meanings. In this case you can include the login domain name or realm name to distinguish the principals.

Additional principal specifications

```
<!-- normal, no domain or realm info -->
<principal class="org.apache.geronimo.security.realm.providers.GeronimoGroupPrincipal" name="user"/>

<!-- login domain name specified -->
<login-domain-principal domain-name="mydomain" class="org.apache.geronimo.security.realm.providers.
GeronimoGroupPrincipal" name="user"/>

<~-- realm name and login domain name specified>
<realm-principal realm-name="my-properties-realm" domain-name="mydomain" class="org.apache.geronimo.security.
realm.providers.GeronimoGroupPrincipal" name="user"/>
```