Service Authentication

Service Authentication

Status: PROTOTYPE
Created: 4. April 2013
Author: fmeschbe
Issue: SLING-2944

Problem

Since the early days of Sling we had methods to get an administrative JCR Session and later an administrative ResourceResolver. These methods were intended to provide services with access to the repository with less restrictions than regular users and to also allow those services to access the Resource tree (and JCR Repository) without hard-coding a password in the code or even having the password as some plain text in configuration.

Over the years, it turned out that these `loginAdministrative` methods have been abused.

The goal of this proposal is to come up with new API to replace the `loginAdministrative` methods.

One example of a service, which currently uses administrative privileges but which would benefit from a carefully crafted service user is the Tenant Manager

Requirements

- Don't use administrative JCR Sessions or ResourceResolvers all over
- Allow services access to JCR Sessions and ResourceResolvers without requiring to hard-code or configure passwords
- Allow services to use "service users" which have been specially configured for service level access (as is usually done on unixish systems)
- Allow administrators to configure the assignment of service users to services

Solution

New loginService methods

Two new methods are introduced to replace `loginAdministrative` methods:

- `ResourceResolver getServiceResourceResolver(Map<String, Object> authenticationInfo) throws LoginException;`
- `Session loginService(String serviceInfo, String workspace) throws LoginException, RepositoryException;`

The bundle identifying the actual service is not part of the new API. The bundle is taken from the call stack by leveraging the OSGi Service Factory mechanism: Each bundle using the `ResourceResolverFactory` or `SlingRepository` service actually gets an instance bound to the using bundle. That bundle is used to identify the service.

The `serviceInfo` parameter or `sling.service.info` property of the `authenticationInfo` map may be used to provide additional information on the service. See the *New ServiceUserMapper Service* section below for information on additional service information.

Communicating Service Information to ResourceProviderFactories

The `ResourceProviderFactory` interface is not extended for the new service login. Rather the required information – using bundle and additional service information – is passed to the `getResourceProvider` method as part of the `authenticationInfo` map:

- `ResourceResolverFactory.USER` – name of the service user (never `null`)
- `ResourceProviderFactory.SERVICE_BUNDLE` – the service `Bundle` object (never `null`)
- `ResourceResolverFactory.SERVICE_INFO` – additional service information (optional; may be `null`)

In case the `ResourceProviderFactory` makes use of another service to provide the `ResourceProvider` the provided service bundle should be used to acquire the service to allow the service to support service logins using the `ServiceUserMapper` service. An example of such an implementation would be the JCR based `ResourceProviderFactory` which gets the `SlingRepository` service using the service bundle.

New ServiceUserMapper Service

A service is introduced which allows to map a service to a user name. A service is identified by a service name related to the OSGi Bundle implementing the service and an additional service information string. For example a bundle implementing mail support may represent the *MailServer* service while the actual mail sender may identify itself with the *sender* information and some mail queue handler may identify itself with the *queue* information. This allows separate users to be used for sending messages and handling the message queue or using the same user for both services, depending on the requirements and needs of the system administrator.

The `ServiceUserMapper` service has two methods:

- `String getServiceName(Bundle bundle, String serviceInfo);` – Returns the value of the service identification string to use for the bundle providing the service. In the above example of the message sender service, when call with the mail server bundle and `serviceInfo="sender"` the returned value might be `MailServer:sender`.
- `String getUserForService(Bundle bundle, String serviceInfo);` – Returns the name of the user to be used for the given service.

This `ServiceUserMapper` service is intended to be used by implementations of the new `loginService` methods to allow mapping services to user names and to provide for a central point of configuring the mapping.

Deprecate loginAdministrative

The following methods are deprecated:

- `SlingRepository.loginAdministrative`
- `ResourceResolverFactory.getAdministrativeResourceResolver`
- `ResourceProviderFactory.getAdministrativeResourceProvider`

The implementations we have in Sling's bundle will remain implemented but there will be a configuration switch to disable support for these methods: If the method is disabled, a `LoginException` is always thrown from these methods. The JavaDoc of the methods is augmented with this information.

Prototype Implementation

A prototype implementation of this concept is available in the deprecate_login_administrative whiteboard.