Authentication Initiation

Authentication Initiation

Status: IMPLEMENTED
Created: 21. April 2009
Author: fmeschbe
JIRA: SLING-938, SLING-939
References: Sling Dev: Refining the authentication process, Authentication
Updated: 24. April 2009, fmeschbe, Rename `Authenticator.requestAuthentication` method to `login` and add `NoAuthenticationHandler` exception

Current State

Authentication processing is currently implemented using the following components:

- The `SlingAuthenticator` class is managed by the `SlingMainServlet` and is used as a backend for the OSGi `HttpContext.handleSecurity` method.
- The `AuthenticationHandler` interface is a service interface, which may be used to extend authentication protocols. Currently Sling includes two implementations of this interface for the standard HTTP Basic authentication and for OpenID authentication.
- To initiate a login one or more servlets or scripts have to be written, which create requests suitable for authentication. For example the `LoginServlet` of the `org.apache.sling.httpauth` bundle uses an HTML form and Ajax requests to setup the HTTP BASIC authentication.
- The HTTP Basic authentication handler supports a `sling:authRequestLogin` parameter to have a client request ask for authentication.

This approach currently has two major issues:

1. Initiating authentication can only be done by the user requesting the login servlet or script. For example to initiate HTTP Basic authentication using the `org.apache.sling.httpauth` bundle, the user has to actively request the `/system/sling/login.html` page.
2. There is a conceptual break between login initiation (or form rendering) and authentication handling.
3. The `sling:authRequestLogin` parameter is only obeyed if HTTP Basic authentication is used and the handler is deemed responsible for the request.

To solve these issues, I suggest we add a new API, which may be used to force the initiation of authentication from servlets or scripts. This new API would be implemented such, that a `AuthenticationHandler` is selected based on some path (just like for the actual authentication) and the `AuthenticationHandler.requestAuthentication` method is called.

Currently the `AuthenticationHandler.requestAuthentication` method is only called if an authentication request, which has been initiated from the client side for example by submitting a form, failed and must be reissued. With the new approach, this method may also be called to initiate authentication.

Non Goals

Logout

This concept is not about request "logout". To complement the authentication initiation process to enable rendering authentication handler agnostic login buttons in some GUIs it would likewise be interesting to provide a generic logout mechanism which allows rendering a authentication handler agnostic logout button.

Currently, the only way to implement a "logout" is to initiate another authentication process. This is of course the only way to "logout" when using plain HTTP Basic authentication. For other authentication schemes like OpenID or Single Sign-On solutions, the situation may be different and an actual "logou" mechanism may be existing.

Non-JCR-centric AuthenticationInfo

This concept is not about modifying the `AuthenticationInfo` class, which is used by the `AuthenticationHandler` interface to give back the login credentials for the authenticator to actually log in. This is rather the issue of the general refactoring of the `ResourceResolver` access.

API extension

A new `Authenticator` interface is defined and exported in the `org.apache.sling.engine` bundle :

org.apache.sling.engine.auth.Authenticator

```
/**
 * The <code>Authenticator</code> interface defines the service interface of the
 * authenticator used by the Sling engine. This service provides a method to
 * find an {@link AuthenticationHandler} and call its
 * {@link AuthenticationHandler#requestAuthentication(HttpServletRequest, HttpServletResponse)}
 * method.
 * <p>
 * This interface is not intended to be implemented by applications but may be
 * used to initiate the authentication process form a request processing servlet
 * or script.
 *
 * @since 2.0.4
 */
public interface Authenticator {

    /**
     * Finds an {@link AuthenticationHandler} for the given request and call its
     * {@link AuthenticationHandler#requestAuthentication(HttpServletRequest, HttpServletResponse)}
     * method to initiate an authentication process with the client.
     * <p>
     * This method must be called on an uncommitted response since the
     * implementation may want to reset the response to start the authentication
     * process with a clean response. If the response is already committed an
     * <code>IllegalStateException</code> is thrown.
     * <p>
     * After this method has finished, request processing should be terminated
     * and the response be considered committed and finished.
     *
     * @param request The object representing the client request.
     * @param response The object representing the response to the client.
     * @throws NoAuthenticationHandlerException If no authentication handler
     *             claims responsibility to authenticate the request.
     * @throws IllegalStateException If the response has already been committed.
     */
    public void login(HttpServletRequest request,
            HttpServletResponse response);

}
```

This interface is implemented by the `SlingAuthenticator` class which is also registered under this service interface. The `SlingAuthenticator` implementation in fact already has an implementation of this method, which finds an `AuthenticationHandler` for the request and calls its `requestAuthentication` method.

The `login` method has three possible exit states:

Exit State	Description
Normal	An `AuthenticationHandler` could be selected to which the login request could be forwarded.
`NoAuthenticationHandlerException`	No `AuthenticationHandler` could be selected to forward the login request to. In this case, the caller can proceed as appropriate. For example a servlet, which should just login a user may send back a 403/FORBIDDEN status because login is not possible. Or a 404/NOT FOUND handler, which tried to login as a fallback, may continue and send back the regular 404/NOT FOUND response.
`IllegalStateException`	The response has already been committed and the login request cannot be processed. Normally to request login, the current response must be reset and a new response has to be prepared. This is only possible if the request has not yet been committed.

Generalize sling:authRequestLogin

The request parameter `sling:authRequestLogin` should be generalized and supported by the `SlingAuthenticator`: If none of the registered authentication handlers is able to extract credentials this parameter should cause the authenticator to call `Authenticator.login` method to initiate a login process.

HTTP Basic Authentication Changes

The implementation of the HTTP Basic Authentication bundle `org.apache.sling.httpauth` is modified as follows:

- The AuthorizationHeaderAuthenticationHandler is modified to render the login form when the requestAuthentication method is called.
- The LoginServlet is modified to just call the Authenticator.login method instead of rendering the form itself. If no Authenticator service is available or if no AuthenticationHandler is willing to perform authentication, the LoginServlet sends back a 403/FORBIDDEN response.