WebNow Single Sign-On Solutions
Technical Guide

ImageNow Version: 6.7.x

Written by: Product Documentation, R&D
Date: September 2016
## Table of Contents

Overview ........................................................................................................................................... 4
SSO for WebNow ................................................................................................................................. 4
Web-based SSO solutions ...................................................................................................................... 5
  SSO solution using Apache Tomcat with Realms ............................................................................... 6
  Testing an SSO integration model ...................................................................................................... 7
  SSO integration options ..................................................................................................................... 8
Overview

Nearly all organizations today have multiple applications that need to operate together. One of the challenges present in all integration scenarios is user password management. Disparate systems for user accounts and passwords cause a number of side effects. For example, lost time due to forgotten passwords, frustration and increased maintenance due to multiple passwords per user, and a lack of seamless transitions between the applications that need to interact.

Single sign-on (SSO) technology is designed to simplify user management. There are many solution options available in the marketplace today. While they vary in scope, many provide both authentication and authorization. Authentication identifies who the logged in user is, while Authorization restricts what the user can access and what actions they can take. These solutions also serve a variety of application types. Windows applications and browser applications are two main categories. Solutions that are focused on browser applications are often referred to as web-based SSO functionality.

SSO for WebNow

Starting with WebNow version 6.5, we provide the ability to utilize SSO integration, in addition to the authentication options already available in ImageNow Server. URL integration is frequently used to launch WebNow from a link embedded in another application, so it can benefit from the use of SSO in these scenarios. Currently, only authentication can be delegated to the SSO provider. Many web-based SSO providers have page-based authorization that is enforced per URL. These do not work with a rich applet client. As such, authorization for WebNow is still managed from ImageNow Management Console to determine user access privileges. Also, the user names must exist in both ImageNow Server and in the SSO provider. A user replication agent may be useful in synchronizing user lists. WebNow is designed to integrate with SSO providers that are installed and maintained by the customer.

Because authentication is handled by an external entity, a stateful session is introduced outside of the ImageNow/WebNow system. The SSO provider manages the lifecycle of this session, determines if the user is logged in, and manages timeouts for the session. When SSO mode is enabled for WebNow, there are a few resulting behavioral changes. The first is that there is a programmatic hook at load time in WebNow to read the name of the authenticated user from the SSO provider. Secondly, normal login into WebNow is disabled, since authentication has been delegated. This includes the login screen and URL parameters. Lastly, another hook is present when the user clicks the disconnect button in WebNow (url-redirect-on-disconnect= in WebNow.settings). This allows using a URL endpoint to notify the SSO provider system that the SSO session should be invalidated. This can be useful when the SSO-managed session is not bound to the lifetime of the browser application window itself.

While solutions vary, the basic user story follows this general pattern:

1. User requests a URL resource (http://example.com/webnow).
2. An SSO provider intercepts the non-authenticated request and either prompts or redirects the user for credentials.
3. An Identity Provider (IdP) authenticates the credentials against the user store. If the user authenticates successfully, the IdP directs the user to the originally requested resource.
4. WebNow loads and reads the pre-authenticated username from the SSO provider. This is typically communicated via an HTTP header.
Web-based SSO solutions

Shibboleth is a web-based SSO solution. SiteMinder is another web-based SSO solution, and there are many others that share a similar architecture. In this example, Apache HTTPServer runs the Shibboleth service provider module that offers gated access to `/webnow`. The actual content for `/webnow` is served to the user via proxy from the Tomcat server using an AJP connector. When the user first requests `http://example.com/webnow`, Shibboleth redirects the user to another site (`http://testshib.org`) that serves as the authority for the valid account names and passwords. There, users are prompted for their credentials. When a user authenticates successfully, the identity-authority site redirects the user back to `http://example.com/webnow` with some information about the now logged on user attached to the request.

![Diagram of Shibboleth SSO solution](image)

Components:

   Shibboleth Identity Provider
   Shibboleth Service Provider (mod_shibboleth)
   AJP Connector to Tomcat (mod_ajp)
   WebNow

Typically, this information is communicated between these layers using an XML based protocol called Security Assertion Markup Language (SAML). SAML relies on the same key-pair encryption technology employed by HTTPS to establish both trust and to secure communications between the Identity Provider and Apache HTTPServer even though they do not connect to each other directly. Their only contact with each other is via the information embedded along with the HTTP requests that are redirected.

When the new request has been passed to `http://example.com/webnow`, the Shibboleth module once again takes over and the following occurs:

- It validates the authentication information embedded into the request.
- Approves access.
- Embeds the user name into the request as the REMOTE_USER header for informational use (configurable).
• Apache HTTP server uses the configured AJP connector to request the content from …/webnow from the Tomcat server.

• WebNow loads on the Tomcat server and reads the name of the pre-authenticated user from the REMOTE_USER header.

• The content is sent back to the end user’s browser via reverse proxy through the AJP connector.

To the end user, it appears the WebNow content comes from the Apache HTTP server component.

SSO solution using Apache Tomcat with Realms

One very simple model for basic SSO authentication is Apache’s Tomcat server with the Realms component. While the model is not the same as most web-based SSO solutions, the same concepts apply. We do not recommend using basic authentication for communicating credentials in any production environment, but we include it in the discussion here because it is a no-cost option that is easy to set up in a test environment for proof-of-concept testing with WebNow. Tomcat already ships with a declarative authentication model that applies to the “/manager” application.

A Realm is configured in Tomcat that is backed by a user store. There are several types of user stores available in Tomcat, but the one in use for the manager application uses a file called tomcat-users.xml to define roles and users. When the user requests http://example.com/webnow, the browser will receive this response:

HTTP/1.1 401 Unauthorized

…

WWW-Authenticate: Basic realm="WebNow Realm"

The browser has a handler for basic authentication, so it prompts the user for credentials. Afterwards, the browser re-issues a request for http://example.com/webnow, this time with the following inclusion:

Authorization: Basic dGVzdDE6aW1hZ2Vub3c=

The Tomcat server then validates the credentials and allows access to WebNow. It should be noted that Apache Realms can also be used with other methods of communicating credentials that are more secure than basic. Also, RealmBase can be extended for custom authentication needs. In practice, this user experience is very similar to Integrated Windows authentication in IIS that can be applied per-virtual directory.
Testing an SSO integration model

If you would like to try this SSO model in a test environment, use the following configuration changes for Tomcat:

tomcat-users.xml

```xml
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
   <!--Add the following role and a test user account -->
   <role rolename="WebNow"/>
   <user username="test" password="imagenow" roles="WebNow"/>
</tomcat-users>
```

web.xml

This is located inside webapps\webnow\WEB-INF.

```xml
<!-- Include the following inside the <web-app>element -->
<!-- Define reference to the user database for looking up roles -->
<resource-env-ref>
   <description>
   Link to the UserDatabase instance from which we request lists of defined role names. Typically, this will be connected to the global user database with a ResourceLink element in server.xml or the context configuration file for the Manager web application.
   </description>
   <resource-env-ref-name>users</resource-env-ref-name>
   <resource-env-ref-type>org.apache.catalina.UserDatabase</resource-env-ref-type>
</resource-env-ref>

<!-- Define a security constraint to provide gated access to all URL paths in WebNow -->
<security-constraint>
   <web-resource-collection>
      <web-resource-name>WebNow HTTP</web-resource-name>
      <url-pattern>/*</url-pattern>
   </web-resource-collection>
   <auth-constraint>
      <role-name>WebNow</role-name>
   </auth-constraint>
</security-constraint>
<login-config>
   <auth-method>BASIC</auth-method>
   <realm-name>WebNow Realm</realm-name>
</login-config>
<security-role>
   <role-name>WebNow</role-name>
</security-role>
```
SSO integration options

The configuration steps required for ImageNow Server and WebNow have been documented here:

Support is being added for new providers incrementally, but for those that have not yet been mentioned, following are the basic steps to determine whether an SSO provider should be integrated with WebNow:

- The SSO provider must detect any unauthorized request to WebNow and authenticate the user to the user store before granting access to WebNow.
- The SSO provider must communicate information about the logged-in user to WebNow through an HTTP header.

All access to WebNow must be protected by the SSO provider. Since the authentication step occurs before WebNow loads, it implies a level of trust between WebNow and the SSO provider. Because of this, you need to ensure that only SSO-provider approved requests are granted access to WebNow to prevent spoofing. Each SSO implementation has its own means of security against spoofing. Direct access to the Apache Tomcat application should be limited to connections from the HTTP server component. For Integrated Windows Authentication in IIS and Realms in Tomcat, the container provides this check for you every time a request comes in for the content at .../webnow.