Rest 3LO and Learn SSO

Mark Kauffman

{% assign sluggedName = page.name | replace: '.md' %}

modified: {{ page.last_modified_at | date: '%b-%d-%y' }}

REST Integrations 3-Legged OAuth and Learn Custom Login Pages For System Administrators

Your Custom Login Page and REST 3-Legged OAuth This article is to help Learn administrators ensure that a custom SSO login page works with REST 3LO applications. Several clients have reported that when trying to complete the following workflow with a 3rd-party REST Application their users get stuck on the Learn landing page. Expected 3-legged authentication workflow:

1. Access a 3rd-party application that has a REST integration with a Learn system (App).
2. Use functionality on the App that requires a login to Learn. (Performs an /authorizationcode REST Request)
3. Login to Learn via a custom login .jsp page.
4. Get redirected to back the App. The App can now make REST requests to the Learn system.

When the custom login .jsp page is not built to take the 3LO process into account, the user logs into Learn and 'sticks' on their Learn landing page. They are never redirected back to the App.

What Went Wrong The problem is caused by the login link or button that the user clicks on in the third step above. We've always found that link to have been hard-coded, usually to use the SSO integration for the Learn system. The hard-coded link does not contain the necessary parameters that are sent to the Learn system with the /authorizationcode request. The link can not be hard-coded. It must be dynamic, as shown in this video{:target="_blank"}.

How to Fix As mentioned in the video, when the custom login .jsp uses the <loginUI:loginForm/> tag to create the link/button that the user clicks on to login to the system, that link will be dynamically generated and preserve all of the information that was sent to the Learn system with the /authorizationcode request from the 3rd-party App. You cannot merely have the tag on the page. You must use the tag to create the link because that tag uses the necessary Learn internal Java code to correctly build the sign-in link.

Here is a sample custom login page that works for 3LO and SSO*. You can study how the CSS modifies the content shown by the <loginUI:loginForm/> tag so that it shows up as a "button" with a link that is dynamically generated and maintains the 3LO /authorizationcode request parameters.

Note that you may also need to study how your SSO system needs to be configured to pass through parameters. Such is out of the scope of this document - which is only meant to describe the requirements for your custom login page.

Key takeaway: A custom login page can work with REST API integrations only if the Learn-provided tag is used to generate the link a user access to sign in. You may customize how that looks with some CSS, but you can never use other HTML to provide the user a login link.

*Credits to Chris Bray from the University of Arkansas for the UARKexampleCustomLogin.jsp example and Dan Gioia of St. Louis Community College for much of the CSS Chris used in the .jsp. Thank you both!