ExpiringPasswordInterceptConfiguration

This feature is available in V3.3 and later.

Current File(s): *conf/intercept/expiring-password-intercept-config.xml*, *views/intercept/expiring-password.vm*

Format: Native Spring

Overview

The "expiring-password" interceptor flow is an example of how to use an interceptor to detect an expiring password and provide an advisory page to the user before completing the request. There are many ways to do this, and when LDAP is used there are some features available within that login flow for doing this kind of thing, but a more general approach is to track password expiration in a database or directory. This flow includes an example condition that examines such an attribute, and based on configurable policy, displays a template to the user.

All interceptors are enabled or disabled on a per-relying-party basis using properties in the profile bean(s) you want to enable the flow for. See the ProfileInterceptConfiguration topic for an example.

General Configuration

The primary configuration involved with this flow is to define the condition you want it to evaluate, and to customize the view template displayed.

The bean named **shibboleth.expiring-password.Condition** in *intercept/expiring-password-intercept-config.xml* must be defined by you with the condition you want to apply. The bean must be of type Predicate<ProfileRequestContext>, but beyond that, it can do anything, and if the condition evaluates "false", then the view will be displayed.

The example provided uses a built-in class that can evaluate an IdPAttribute produced by the attribute resolver and parses its value into a timestamp to evaluate against a threshold. It can be configured with a format to use to parse out the timestamp and an offset to apply. The offset essentially determines how soon before the actual time that the condition will evaluate to false.

The view to display is contained in *views/intercept/expiring-password.vm,* and apart from displaying some internationalized message content, it automatically forwards to complete the original request after 20 seconds using a meta-refresh.

The other configurable feature is an anti-nag device, a cookie that tracks when the view is displayed and based on the value, prevents re-display of the view unless a configurable amount of time has elapsed.

Reference

Beans

Bean ID	Type	Function
shibboleth.expiring-password.Condition	Predicate<ProfileRequestContext>	Condition evaluated by the system-supplied intercept flow to decide whether the advisory page should be displayed
shibboleth.expiring-password.NotifyCookieName	String	Name of a cookie to use to prevent repeated viewing of the advisory page
shibboleth.expiring-password.NotifyInterval	Long	Minimum number of milliseconds between repeated viewings of the page if the tracking cookie contains a value

Notes

TBD