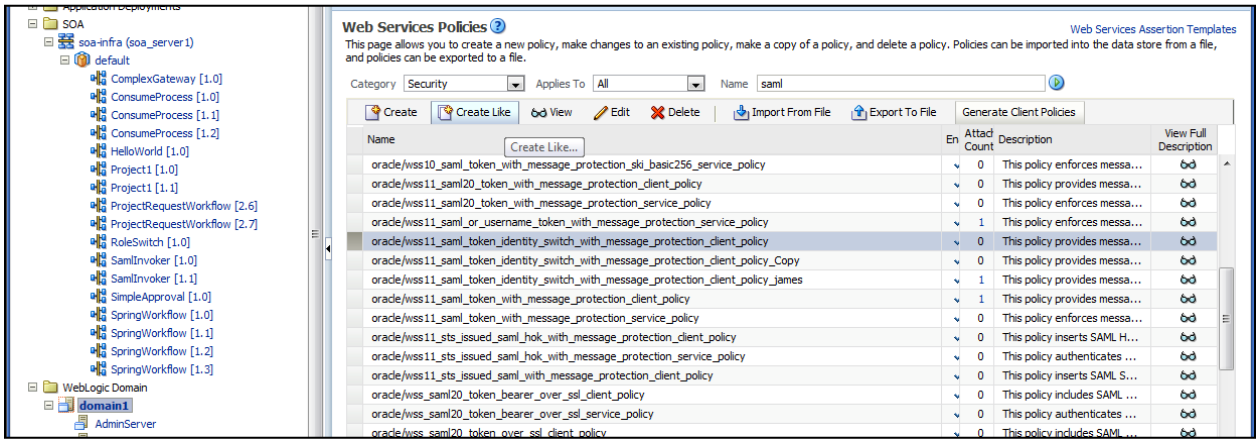
BPM 11g: Configuring SAML Web Service Clients for Identity Switching without Message Protection

Oracle Web Services Manager (WSM) includes **wss11_saml_token_identity_switch_with_message_protection_client_policy**, which enables identity switching. Identity switching means that the policy propagates a different identity than the one based on the authenticated Subject.

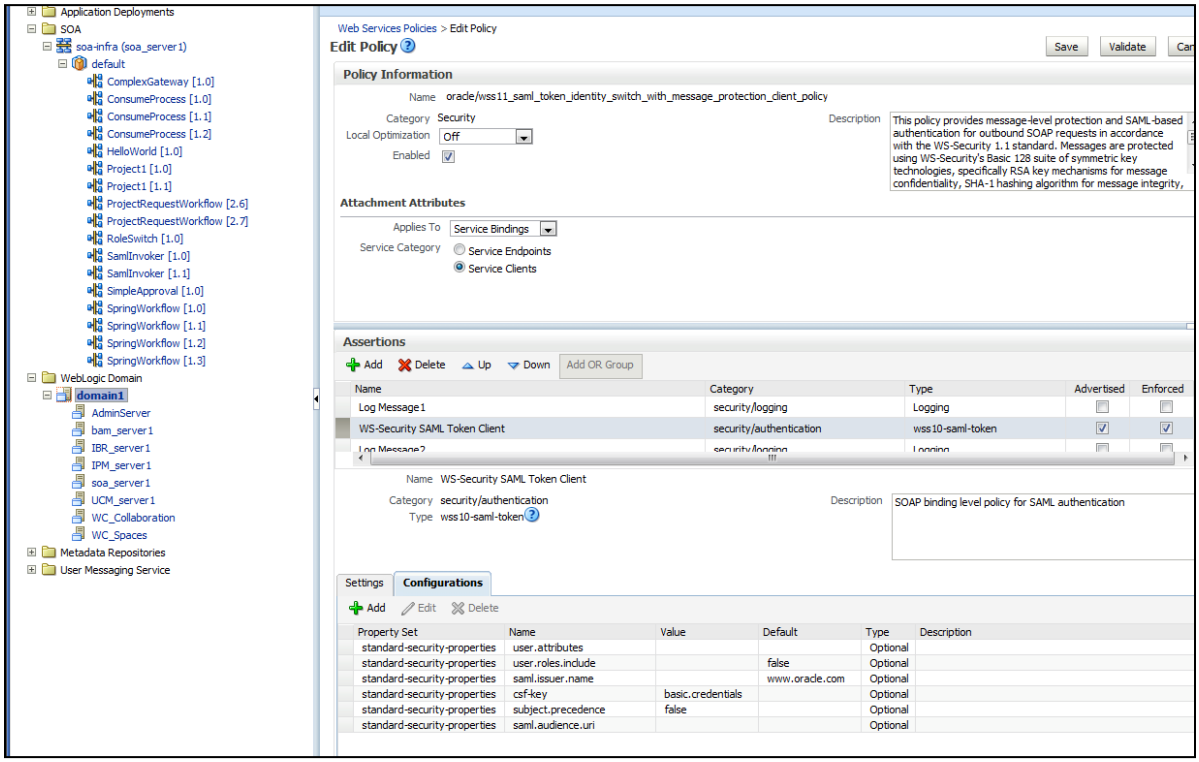
The Service-Oriented Architecture (SOA) application requires you to specify which user identity to use in client-side Web service policies, and then dynamically switches the user associated with the SAML token in the outbound Web service request. Instead of using the username from the Subject, this policy allows you to set a new user name when sending the SAML Web service request.

The **wss11_saml_token_identity_switch_with_message_protection_client_policy** creates the SAML token based on the user ID set via the property **javax.xml.ws.security.auth.username**.

The initial identity switching policy requires message encryption, which requires the server-side policy to be the same. You will not want this policy when working in P6. To change the policy, you need to create a new client-side policy based on the existing identity switching policy (this is done through Enterprise Manager (EM), using the ***"create like"*** option). Within the new policy definition, you can remove the existing assertion (SAML 1.1 SAML with Certificates) and replace it with a new assertion based on an appropriate template, which in this case is WS-Security SAML Token Client.

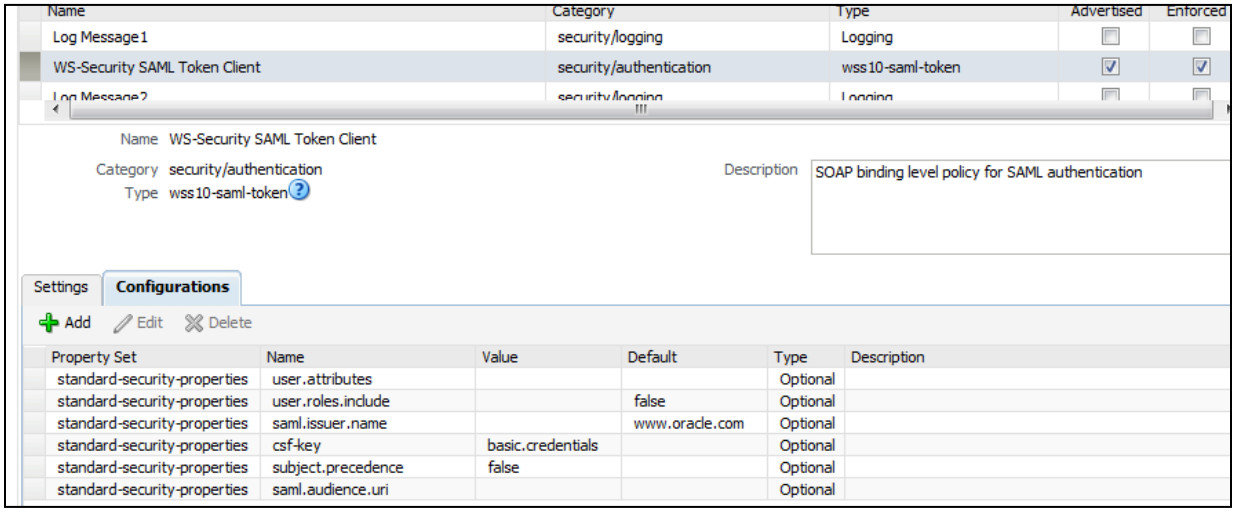


The screenshot below shows where to configure the policy assertions:



You can also copy the custom SAML Identity Policy. To do this, copy the **oracle_wss11_saml_token_identity_switch_with_message_protection_client_policy.txt** file located here:
http://download.oracle.com/docs/cd/E20686_01/English/Technical_Documentation/Oracle_BPM/oracle_wss11_saml_token_identity_switch_with_message_protection_client_policy.txt

The screenshot below shows a clearer version of the configuration:

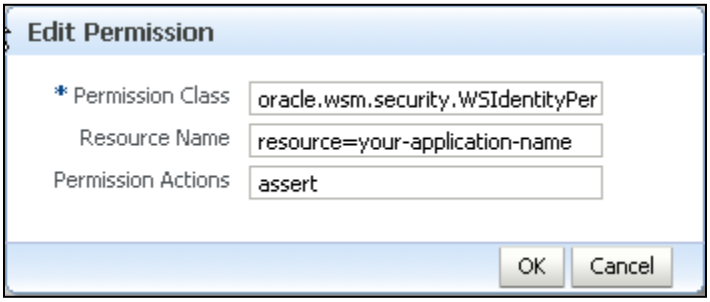


Set the WSIdentityPermission

The Web service client (for example, the SOA reference binding component) to which you attached the **wss11_saml_token_identity_switch_with_message_protection_client_policy** must have the **oracle.wsm.security.WSIdentityPermission**.

To use Fusion Middleware Control and add the **oracle.wsm.security.WSIdentityPermission** to the SOA reference binding component as a System Grant, perform the following steps:

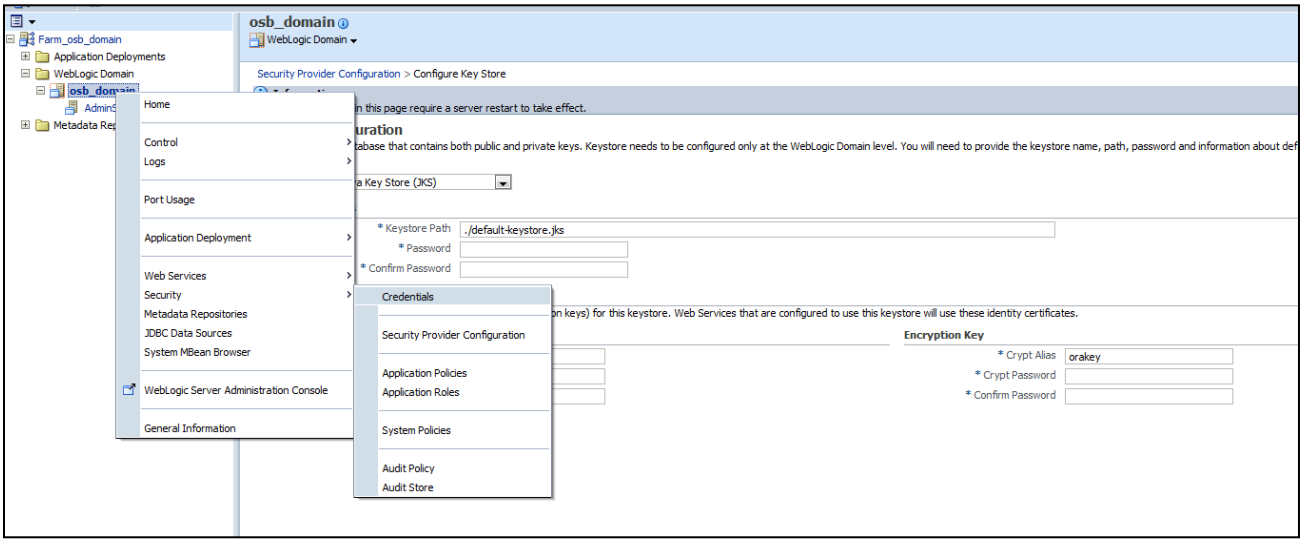
1. In the navigator pane, expand **WebLogic Domain** to show the domain where you need to configure the application. Select the domain.
2. Using Fusion Middleware Control, click **WebLogic Domain**, then **Security**, and then **System Policies**. System policies are the system-wide policies applied to all applications deployed to the current WebLogic Domain.
3. From the **System Policies** page, select the arrow icon in the **Permission** field to search the system security grants.
4. Select one of the codebase permissions to use as a starting point and click **Create Like**.
5. In the **Grant Details** section of the page, enter **file:${common.components.home}/modules/oracle.wsm.agent.common_11.1.1/wsm-agent-core.jar** in the **Codebase** field.
6. In the **Permissions** section of the page, select the starting point permission class and click **Edit**.
7. Enter **oracle.wsm.security.WSIdentityPermission** in the **Permission Class** field. The resource name is the composite name for SOA, and the application name for a J2EE client. The action is always *assert*.



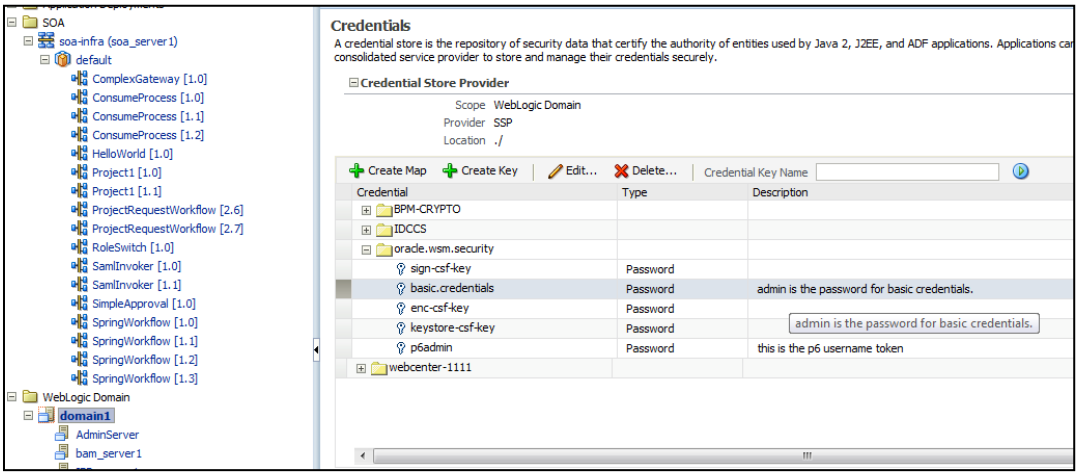
Create the basic.credentials Key

You also need to add the basic.credentials key to the csf store via EM. You might need to create a default keystore if you have not done that already.

1. Right-click domain -> security -> credentials



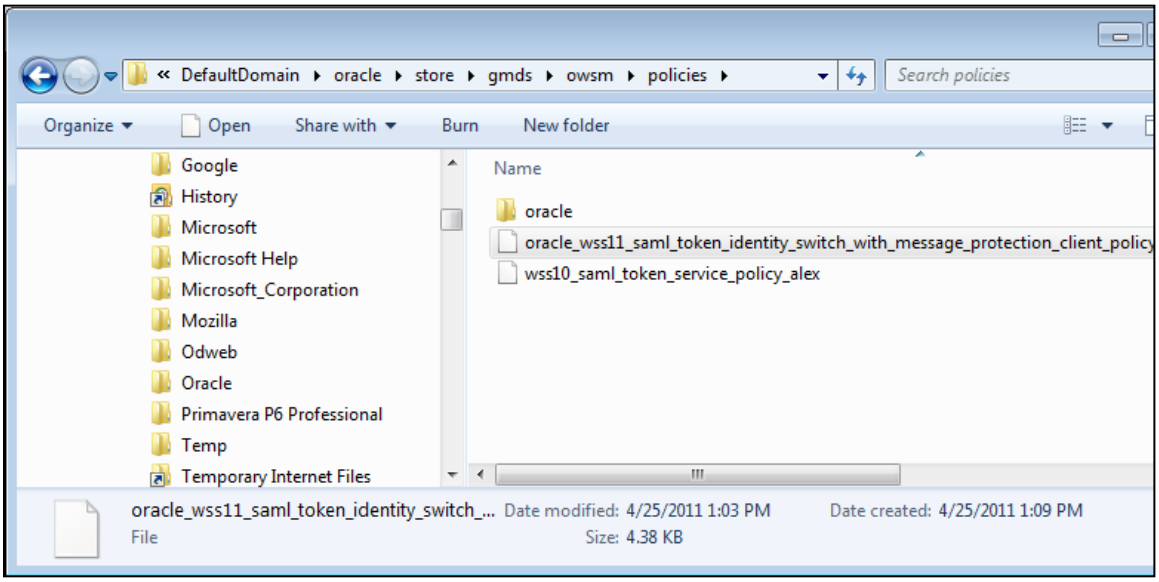
2. Create a **basic.credentials** key.



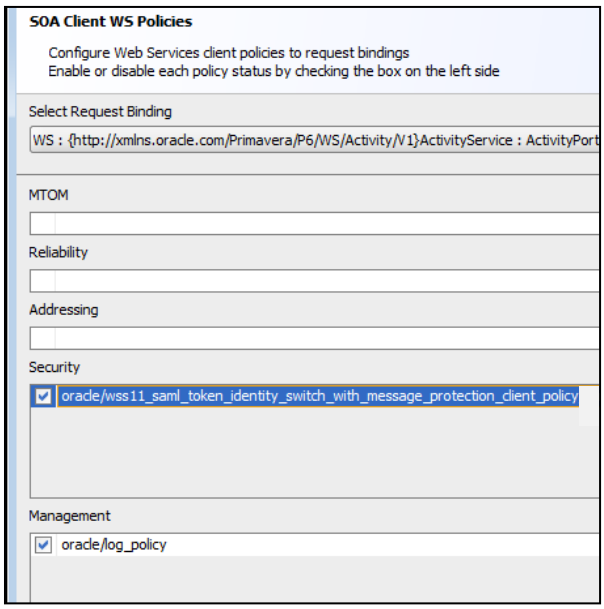
Apply the New Policy

1. Before applying the new policy, you need to import into JDeveloper. Copy the new custom policy to your JDev store directory (either use the attached policy from this document or export your custom policy from EM). The location of the store could appear as follows:

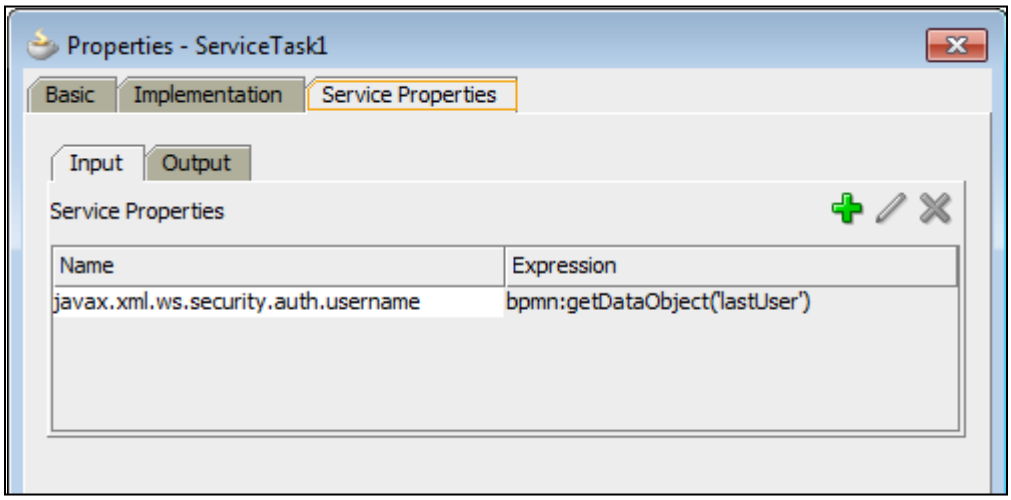
   USER_HOME\AppData\Roaming\JDeveloper\system11.1.1.4.37.59.23\DefaultDomain\oracle\store\gmds\owsm\policies

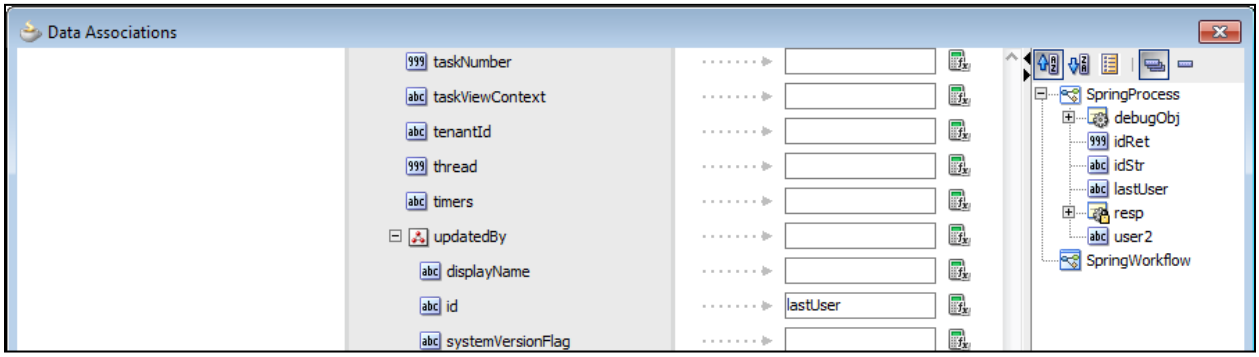


2. Once this is done, apply this new client policy to your service reference in your composite app via EM.



With this policy in place you can then leverage the javax.xml.ws.security.auth.username inbound service property—if you are hardcoding, set the value without quotes. In the example below, the value is set to jcooper; however, you can also extract the username from the payload of execData variable.





You do not have to import the policy to JDev, you can deploy the composite without a client-side policy, and then set the client policy through EM.

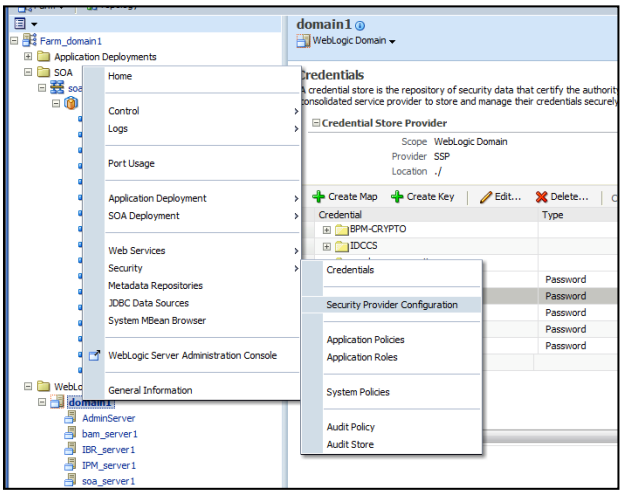
EM has a feature for setting the client-side policies that shows you compatible client-side policies based on the service you are calling.

References:

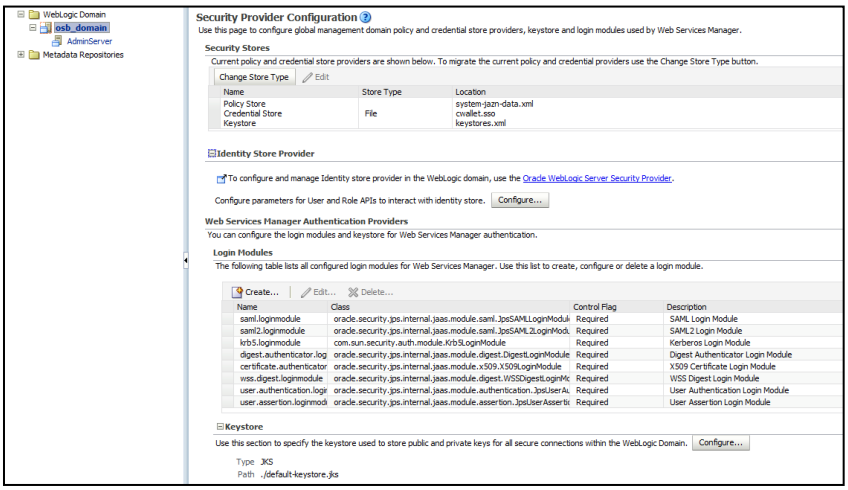
http://download.oracle.com/docs/cd/E17904_01/web.1111/b32511/setup_config.htm#WSSEC3585

Configuring a Keystore if One Is Not Configured

1. Right-click your WebLogic domain and select “Security Provider Configuration.”



2. Select configure in the Keystore Section.



3. Provide credentials.