Oracle Entitlements Server 11gR2 Integration Guide

Published: May 2013

Applies To

Oracle API Gateway v11.1.2.x

Contents

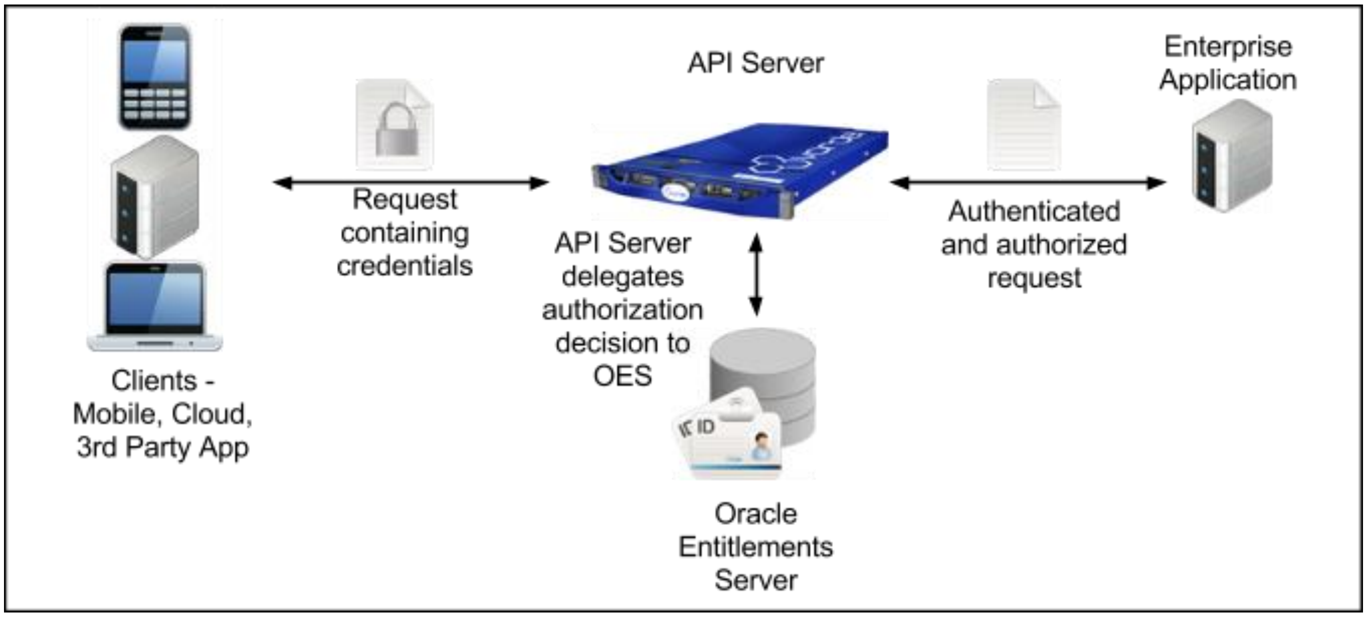
- Introduction
- Prerequisites
- Integration Configuration Steps
  - OES Configuration:
    - Step 1 - Create the Application
    - Step 2 - Create the Security Module
    - Step 3 - Create the Resource Type
    - Step 4 - Create the Resource
    - Step 5 - Configure the Authorization Policy
  - OES Client Setup
  - Distribute the OES Policy
  - Modify the API Server Classpath
  - Start the API Server
  - Configure the API Server to Delegate Authorization to OES
    - Step 1 - Configure the Authentication Filter
    - Step 2 - Configure the OES 11g Authorization Filter
    - Step 3 - Add the Success Message Filter
    - Step 4 - Add the Failure Message Filter
    - Step 5 - Add a Relative Path for the OES Authorization Policy
    - Step 6 - Deploy the Policy
- Testing the Integration Steps
- Conclusion

Introduction

This document describes how to configure the API Gateway to authorize an authenticated user against Oracle Entitlements Server (OES) 11gR2. This will be demonstrated by the following:
- The API Gateway will authenticate a user against its Local User Repository
- The API Gateway will then delegate the authorization decision for the specified resource to OES.

The **OES 11g Authorization** filter will be used to delegate the authorization decision to OES. This filter assumes that an authentication filter has been configured prior to it. Therefore, by the time the authorization filter executes, the `authentication.subject.id` message attribute is populated and its value is used as the subject in the authorization request to OES.

The diagram below shows the sequence of events that occurs when a client sends a message to the API Server. The request sender is authenticated by the API Server and is then authorized against Oracle Entitlements Server. If the user is permitted to access the requested resource, the request will be routed to the Enterprise Application. Otherwise an appropriate fault message will be returned to the client.


API Server
Enterprise Application
Request containing credentials
Authenticated and authorized request
API Server delegates authorization decision to OES
Clients - Mobile, Cloud, 3rd Party App
Oracle Entitlements Server

Deployment of API Server with OES

Prerequisites

API Gateway:
You must have installed API Gateway 11.1.2.x.

OES User:
This guide assumes that a user called “weblogic” is available in OES. Please refer to the OES documentation for instructions on how to add a user.

API Server Local User Store

Furthermore, this guide assumes that you have added the "weblogic" user to the API Gateway Local Repository User Store. The policy we will set up later requires an authenticated user's request to be authorized against OES. By adding the "weblogic" user to the Local User Store, the client can authenticate as this user. The username will then be stored in the `authentication.subject.id` message attribute, which is then passed to the **OES 11g Authorization** filter and subsequently on to OES in order to make the authorization decision. Please refer to the API Server documentation for more information on adding users.

OES Client:

The OES Client (Security Module) must be installed on the machine running the API Server. The OES Client has its own installer, which is available from www.oracle.com. In the integration steps described in this guide, we used the following version of the OES Client:

- "Oracle Entitlement Server Security Module 11g - 11.1.2.0.0"

The OES Client installer requires that a JRE is available on the target machine. In the absence of a preferred JVM on the target machine, the API Server ships with a JRE that can be used. For example, on Windows, the JRE is located in `<APISERVER_INSTALL>\win32\jre`. On Unix, the JRE is located in `<APISERVER_INSTALL>/platform/jre`.

Launch the installer from the command line and pass the JRE location using the "jreLoc" argument as follows:

On Windows:

```
C:\setup.exe -jreLoc <APISERVER_INSTALL>\win32\jre
```

On UNIX:

```
./runInstaller –jreLoc <APISERVER_INSTALL>/platform/jre
```

OES 11gR2:

You have installed and configured OES 11gR2 and have started it using the following command on UNIX-based systems (assuming a weblogic domain of "oes_domain" has been configured):

```
# cd ~/Middleware/user_projects/domains/oes_domain
# ./startWebLogic.sh
```

Curl Test Utility:
To test the integration steps outlined in this guide, we will use the Curl testing utility to POST requests to the API Server. It is available from the following URL:
http://curl.haxx.se/download.html

You can, however, use any client capable of sending HTTP POST requests with HTTP basic authentication.

Integration Configuration Steps

OES Configuration:

The following steps describe how to essentially create a resource and an authorization policy for that resource in Oracle Entitlements Server. For more detailed instructions about each of these steps, please refer to the OES documentation.

You can use the **OES Authorization Policy Manager** web-based administration interface to configure the resource. The web interface is available at the following URL, where [OES_HOST] refers to the IP or hostname of the machine on which OES is running:

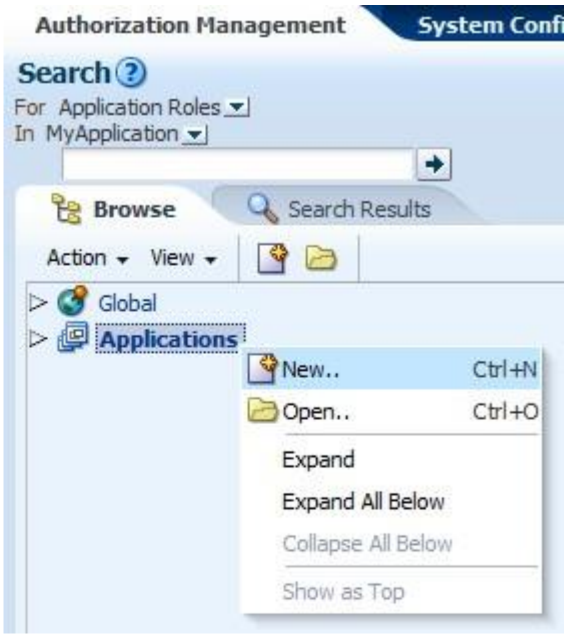
```
http://[OES_HOST]:7001/apm
```

Login using your Weblogic credentials and complete the following steps.

Step 1 - Create the Application

You can add a new Application in OES by carrying out these steps:

1. Open the **Authorization Management** tab.
2. Right-click on the **Applications** node in the tree and select the **New** option:



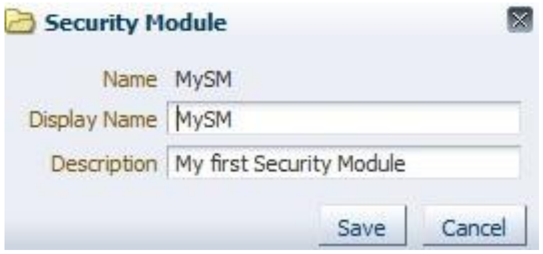
Adding a new application from the Authorization Management tab

3. Enter a name and a description for the new Application, for example, “MyApplication” and “First App” for the name and description, respectively.
4. Click on the “Save” button at the top right-hand corner of the screen.

Step 2 - Create the Security Module

The next step is to create a new Security Module.

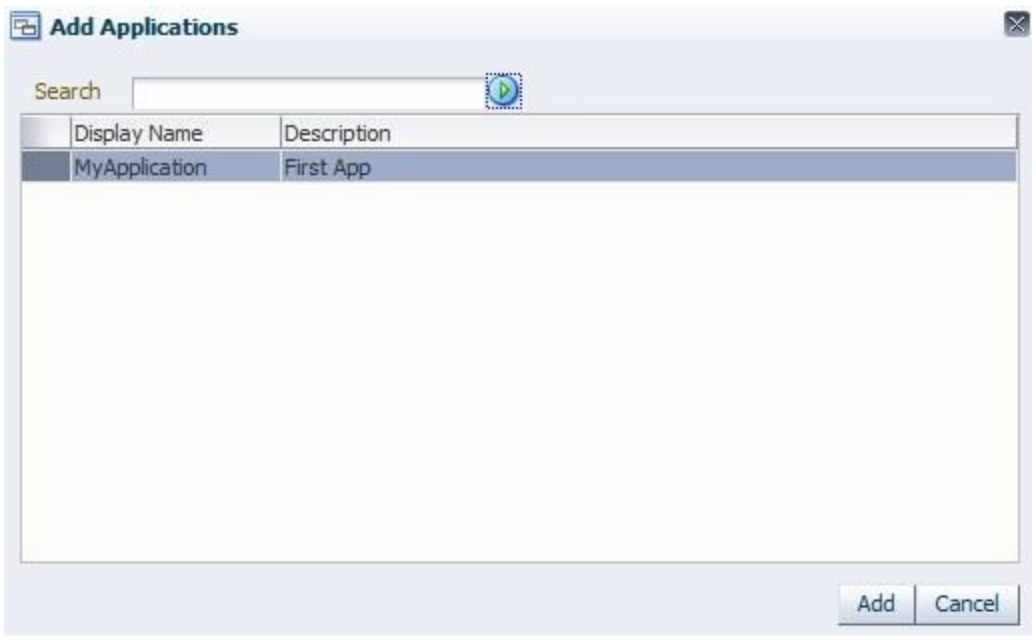
1. Open the **System Configuration** tab.
2. Double-click on the **Security Modules** node in the tree.
3. Click on the **New** button at the top of the Security Modules table.
4. Enter “MySM” as the **Name** of the new Security Module:


Security Module

Name	MySM
Display Name	MySM
Description	My first Security Module

Enter a name, display name, and an optional description for the new Application

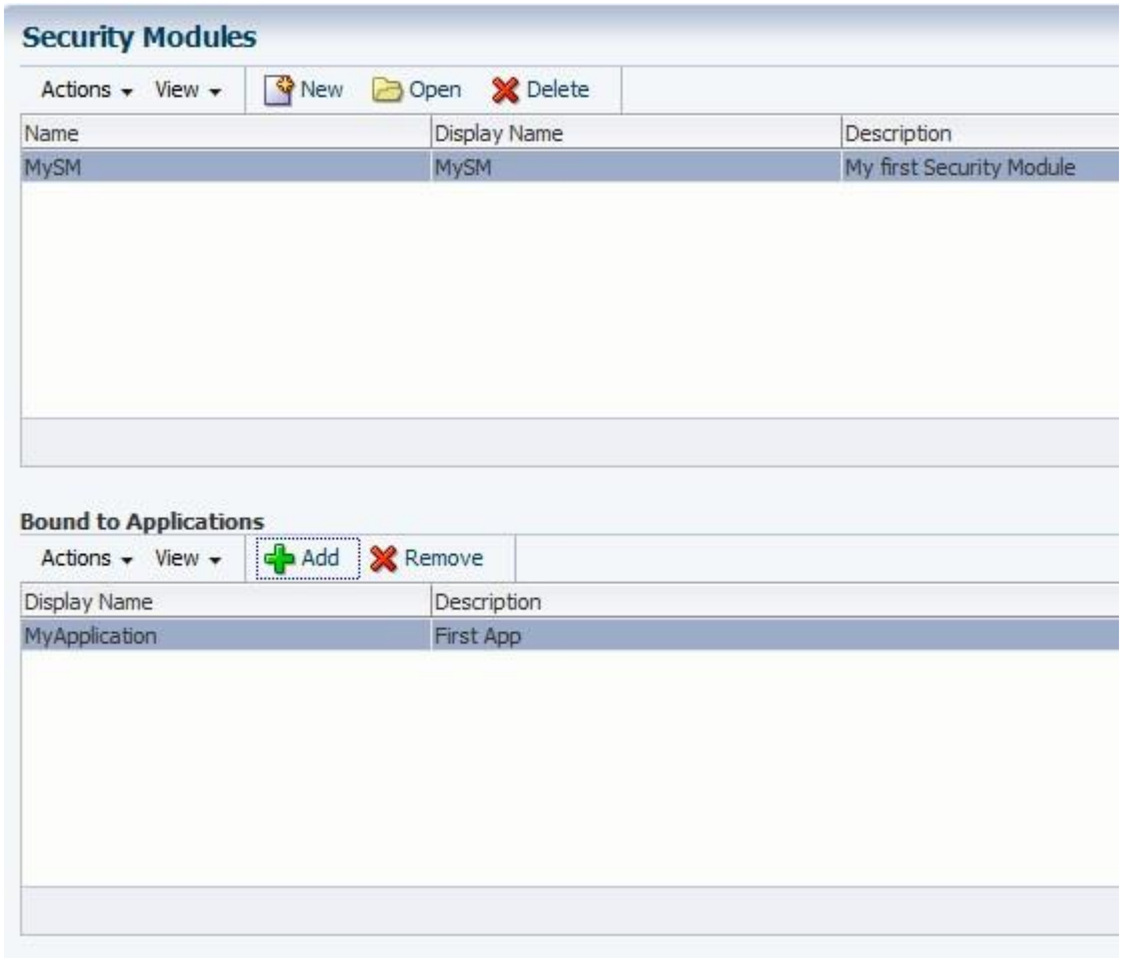
5. Click the **Add** button at the top of the **Bound to Applications** table.
6. Leave the **Search** field blank and press the search button to the right of the field.
7. Select “MyApplication” from the search results and click the **Add** button.



Display Name	Description
MyApplication	First App

Binding the “MyApplication” application to the Security Module

8. The “MyApplication” should now appear in the **Bound to Applications** table as follows:



Name	Display Name	Description
MySM	MySM	My first Security Module

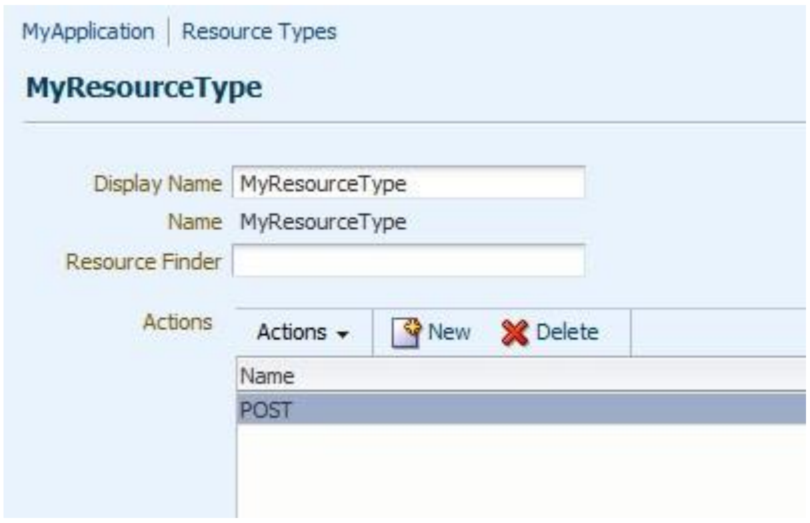
Display Name	Description
MyApplication	First App

The completed Security Module configuration

Step 3 - Create the Resource Type

We will now configure the resource type for the resource that users will be authorized for.

1. Open the **Authorization Management** tab once more.
2. Expand the **Applications** node and then expand the newly created **MyApplication** node.
3. Double-click the **Resource Types** node.
4. Click the **New** button above the table showing the existing **Resource Types**.
5. Enter “MyResourceType” in the **Name** field.
6. Add an action for this resource by clicking the **New** button above the **Actions** table. Add “POST” as an action.
7. Click the **Save** button to save the new **Resource Type**, which should now appear as in the following screenshot:



MyResourceType

Display Name: MyResourceType
Name: MyResourceType
Resource Finder:
Actions

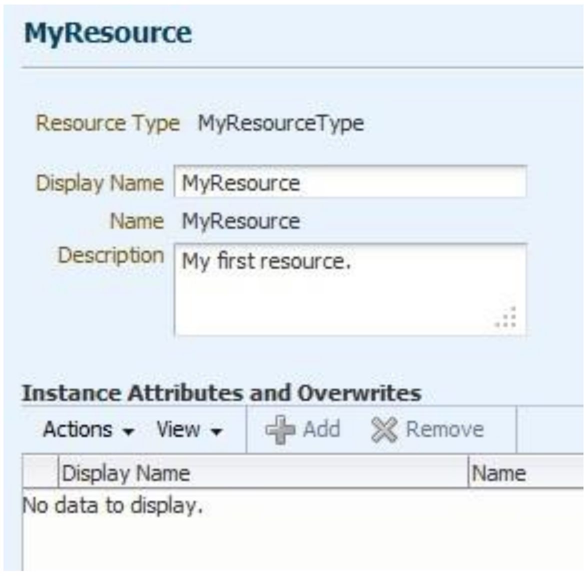
Name
POST

Adding the new Resource Type

Step 4 - Create the Resource

The next step is to add the **Resource.**

1. Expand the **Default Policy Domain** and then the **Resource Catalog** nodes.
2. Double-click on the **Resources** node.
3. Click on the **New** button above the table listing all existing **Resources**.
4. Select “MyResourceType” as the **Resource Type**.
5. Enter “MyResource” as the **Resource** name.
6. Click the **Save** button to save the **Resource**, which should now appear as in the following screenshot:



MyResource

Resource Type: MyResourceType
Display Name: MyResource
Name: MyResource
Description: My first resource.

Instance Attributes and Overwrites

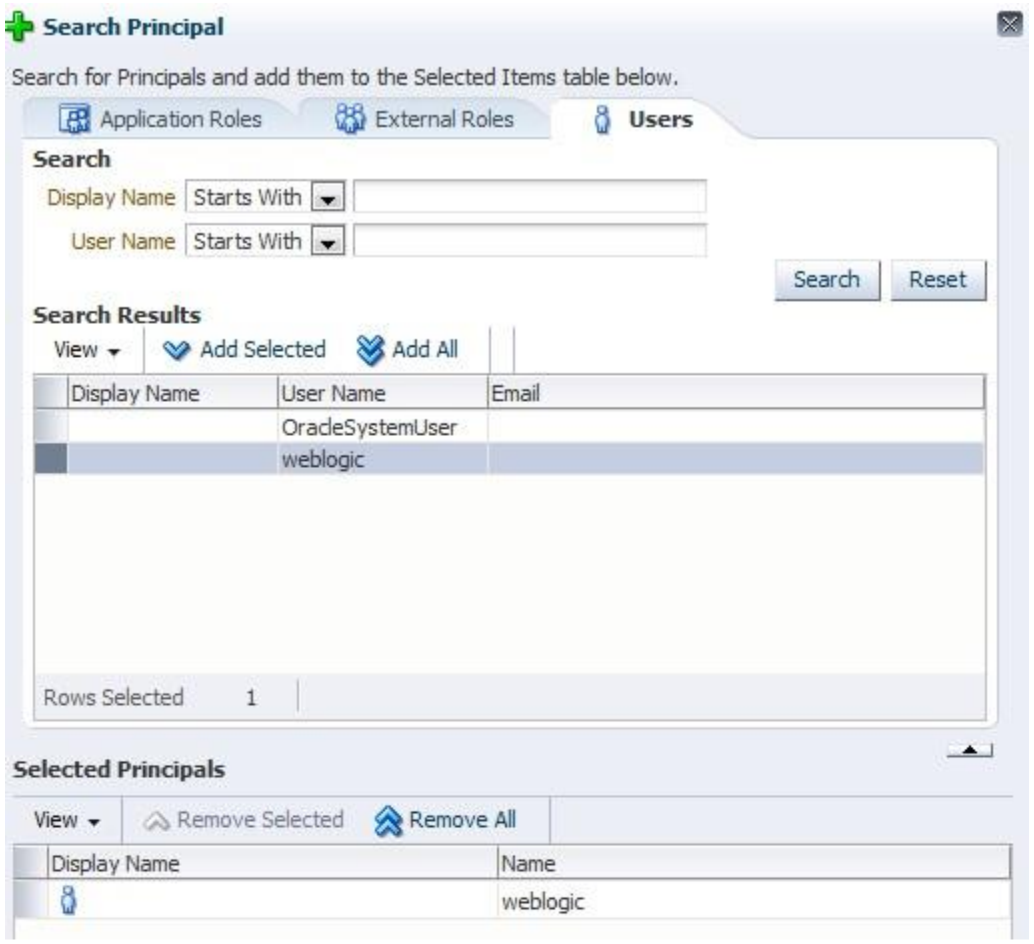
Display Name	Name
No data to display.	

Adding the new Resource

Step 5 - Configure the Authorization Policy

We now have to create the actual **Authorization Policy** that will determine whether we want to **Permit** or **Deny** access to the resource.

1. Double-click the **Authorization Policies** node beneath the **Default Policy Domain** node in the tree.
2. Click the **New** button above the table showing the existing **Authorization Policies**.
3. Enter “MyPolicy” as the name of the new policy.
4. Chose to “Permit” access to the resource target using the corresponding **Effect** checkbox.
5. Configure **Principals** (i.e. uses and/or roles) that can access the resource by clicking on the “plus” button to the right of the **Principals** table.
6. Select the **Users** tab in the **Search Principal** window:



Display Name	User Name	Email
	OracleSystemUser	
	weblogic	

Display Name	Name
	weblogic

Adding the “weblogic” user to the list of Principals governed by this Authorization Policy

7. Select the default “weblogic” user from the table and click the **Add Selected** button to add the “weblogic” user to the **Selected Principals** table, as shown in the screenshot above. Note that the prerequisites section required a “weblogic” user to be available in OES.

8. Click the **Add Principals** button at the bottom of the window.
9. Next we need to specify a resource target that this policy will act on. Click the "plus" button to the right of the **Targets** table.
10. Click on the **Resources** tab on the **Search Targets** window.
11. Select "MyResourceType" in the **Resource Type** dropdown and click the **Search** button.
12. Select "MyResource" from the table and click the **Add Selected** button to add the resource to the **Selected Targets** table.
13. Click the **Add Targets** button at the bottom of the screen.

OES Client Setup

The OES Client distributes policies to individual Security Modules that protect applications and services. Policy data is distributed in a controlled manner or in a non-controlled manner. The distribution mode is defined in the `jps-config.xml` configuration file for each Security Module. The specified distribution mode applies to all Application Policy objects bound to that Security Module. Consult with the OES administrator to find out the distribution mode. For the purposes of this integration guide, *controlled* distribution mode will be used.

Complete the following steps to configure the OES client:

1. Open a command prompt and cd to your OES client installation directory, which will be referred to as `OES_CLIENT_HOME` throughout the remainder of this guide.
2. Set the JAVA_HOME environment variable:

On Windows:

```
> set JAVA_HOME=C:\jdk1.7.0_03
```

On UNIX:

```
> export JAVA_HOME=/home/oesuser/Oracle/Middleware/jdk160_29
```

2. Edit the following file:
   `OES_CLIENT_HOME/oessm/SMConfigTool/smconfig.java.controlled.prp`
3. Ensure that the following values are set:

Parameter	**Description**
oracle.security.jps.runtime.pd.client. policyDistributionMode	Accept the default value "controlled-push" as the distribution mode.

oracle.security.jps.runtime.pd.client.RegistrationServerHost	Enter the address of the Oracle Entitlements Server Administration Server.
oracle.security.jps.runtime.pd.client.RegistrationServerPort	Enter the SSL port number of the Oracle Entitlements Server Administration Server. You can find the SSL port number from the WebLogic Administration console. By default, 7002 is used.

4. On Unix-based systems, run the `config.sh` script located in the `OES_CLIENT_HOME/oessm/bin` directory. On Windows, run the `config.cmd` script located in `OES_CLIENT_HOME\oessm\bin`. The following commands show how to configure the OES client on Windows and Unix-based systems for a Security Module with an ID of "MySM":

On Windows:

```
> config.cmd -smConfigId MySM -prpFileName
OES_CLIENT_HOME\oessm\SMConfigTool\smconfig.java.controlled.prp
```

On Unix:

```
> ./config.sh –smConfigId MySM -prpFileName
OES_CLIENT_HOME/oessm/SMConfigTool/smconfig.java.controlled.prp
```

5. When prompted, specify the following:
    - OES username (Administration Server's user name)
    - OES password (Administration Server's password)
    - New key store password for enrollment

Sample output:

```
> ./config.sh -smConfigId MySM -prpFileName
/home/oesuser/Oracle/Middleware/oes_client/oessm/SMConfigTool/
smconfig.java.controlled.prp
Configuring for Controlled Policy Distribution Mode
Security Module configuration is created at: /home/oesuser/Oracle/
Middleware/oes_client/oes_sm_instances/MySM
Enter password for key stores: ******
Enter password for key stores again: ******
Passwords are saved in credential store.
Keystores are initialized successfully.
```

```
Please enter a value for OES Admin Server User name:weblogic
Please enter a value for OES Admin Server Password:
Enrollment is proceeded successfully.
```

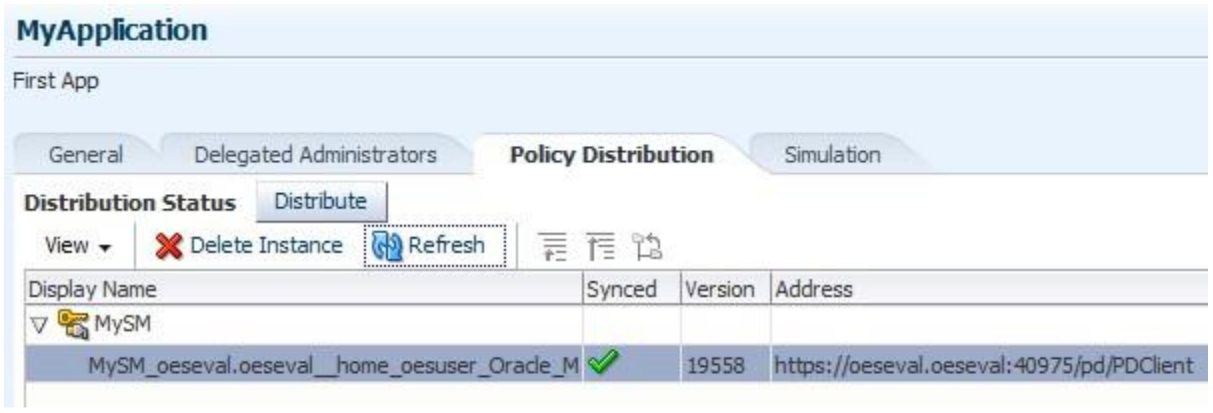
6. Ensure that the Security Module has been configured correctly by checking that the `OES_CLIENT_HOME/oes_sm_instances/MySM` folder has been created.
7. Depending on your OES Client setup, the registration process may not have generated a `cwallet.sso` file in your `OES_CLIENT_HOME/oes_sm_instances/MySM/config` directory. If there is no `cwallet.sso` file present in the `config` directory, you can copy the one generated in the `config/enroll` directory to the `config` directory using the following commands:

```
# cd oes_sm_instances/MySM/config
[oesuser@oeseval config]$ ls
enroll  jps-config.xml  system-jazn-data.xml
# cp enroll/cwallet.sso ./
# ls cwallet.sso  enroll  jps-config.xml  system-jazn-data.xml
```

Distribute the OES Policy

Once the OES client has registered with OES, it is then possible to distribute the policy for that application so that clients making authorization requests for this resource will now be subject to the policy enforcement rules.

1. Double-click the **MyApplication** node in the tree on the **Authorization Management** tab.
2. Open the **Policy Distribution** tab for the application configuration.
3. Expand the **“MySM”** entry in the table.
4. You should see an entry representing the recently registered OES client instance, as shown in the following diagram:



Display Name	Synced	Version	Address
▽ MySM			
MySM_oeseval.oeseval__home_oesuser_Oracle_M	✓	19558	https://oeseval.oeseval:40975/pd/PDClient

5. Select the “**MySM”** application and click the **Distribute** button to push the authorization policy configured for this application.
6. You may need to click the **Refresh** button to update the “Synced” status. You should see a green tick to indicate a successful distribution.

Note that you may have to restart weblogic in order for your newly registered Security Module to appear in the list on the **Policy Distribution** tab.

Modify the API Server Classpath

The API Server’s classpath must be extended to include the OES client JAR on its classpath. To achieve this, create a jvm.xml file at the following location:
`<APISERVER_INSTALL>/conf/jvm.xml`

Edit this `jvm.xml` so that its contents are as follows, providing values for `OES_CLIENT_HOME` and `SM_NAME` that are based on the location where the OES client was installed and the SM name used when enrolling the OES client (i.e. “MySM”):

```
<ConfigurationFragment>
<!-- Change these ENV VARS to match the location where the OEM Client has
been installed and configured -->
<Environment name="OES_CLIENT_HOME" value="/home/oes/Oracle/Middleware/
oes_client" />
<Environment name="SM_NAME" value="MySM" />
<Environment name="INSTANCE_HOME"
value="$OES_CLIENT_HOME/oes_sm_instances/$SM_NAME" />
<!-- Add OES Client JAR to the classpath -->
<ClassPath name="$OES_CLIENT_HOME/modules/oracle.oes.sm_11.1.1/oesclient.jar"
/>

<!-- Add OES JARs to the classpath →
<ClassPath name=”[PATH_TO_OES_JARS]/api.jar”/>
<ClassPath name=”[PATH_TO_OES_JARS]/asi_classes.jar”/>

<VMArg name="-Doracle.security.jps.config=$INSTANCE_HOME/config/jps-config.
xml"/>
<!-- Optional argument to add enhanced logging (via log4j) for the OES Client
-->
```

```
<VMArg name=”-Djava.util.logging.config.file=$INSTANCE_HOME/
logging.properties”/> </ConfigurationFragment>
```

Start the API Server

Start the API Server so that it runs with the OES client classpath and the associated environment settings. Assuming an API instance name of “APIServer1” belonging to a group called “Group1”, you can start the API server using the following command depending on your target platform, where [APISERVER_HOME] refers to the root of your API Server installation:

On Windows:

```
[APISERVER_HOME]\Win32\bin>startinstance -n "APIServer1" -g "Group1"
```

On UNIX:

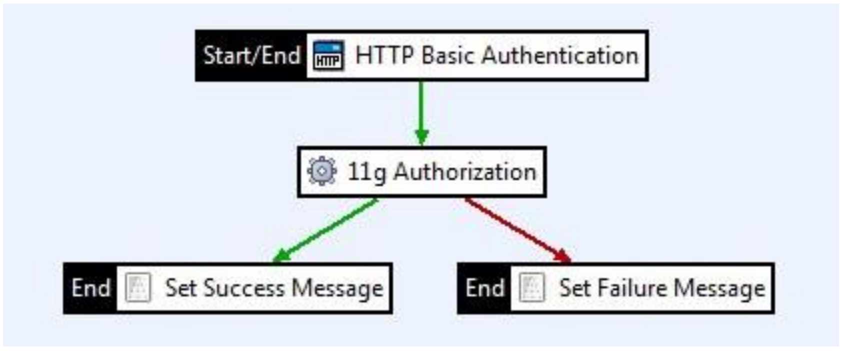
```
[APISERVER_HOME]/posix/bin>startinstance -n "APIServer1" -g "Group1"
```

Configure the API Gateway to Delegate Authorization to OES

This section explains how to configure the API Gateway to delegate authorization decisions to Oracle Entitlements Server. The following steps are required:

- Configure the Authentication filter so that the `authentication.subject.id` attribute is populated with the subject to be used in OES authorization.
- Configure the Oracle Entitlements Server **11g Authorization** filter.
- Configure some appropriate success and failure messages.

The resulting policy created in the API Server will appear as follows:



The complete policy showing the Authentication and OES 11g Authorization filters

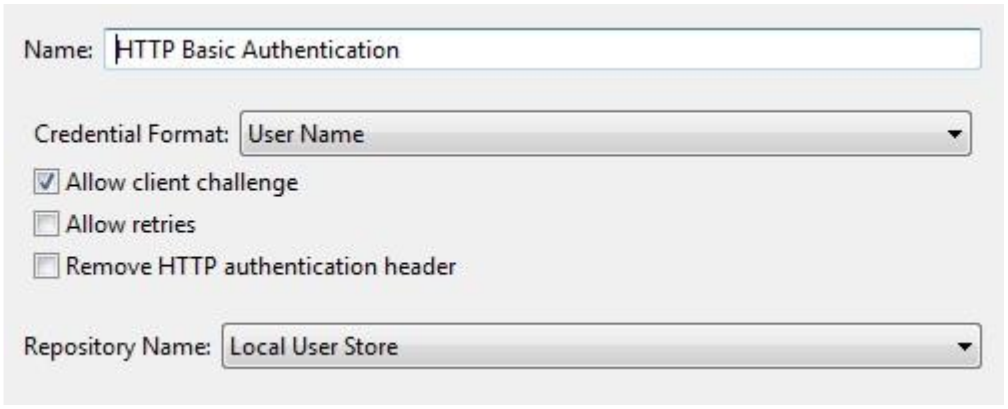
Step 1 - Configure the Authentication Filter

In this setup, it is assumed that the user profile to be authorized via OES is also stored in the Local User Store of the API Server. The API Server can actually delegate authentication decisions to other LDAP directories, databases, and other 3rd party Identity Management systems, including Oracle CA SiteMinder, Oracle Access Manager, and RSA Access Manager. For simplicity, in this guide the API Server's Local User Store is used.

To configure the authentication filter, perform the following steps:

1. Start the Policy Studio.
2. Create a new Policy called "OES Authorization".
3. Drag a **HTTP Basic** filter from the **Authentication** category in the palette and drop it onto the canvas and configure it as follows:
   - **Name:** Enter "HTTP Basic Authentication" in the field provided.
   - **Credential Format:** Select **"**User Name" from the dropdown.
   - **Allow Client Challenge:** Check the "Allow client challenge" checkbox.
   - **Repository Name:** Select "Local User Store" from the dropdown.

The completed configuration for the **HTTP Basic Authentication** filter appears as follows:



Name: HTTP Basic Authentication

Credential Format: User Name

[✓] Allow client challenge

[ ] Allow retries

[ ] Remove HTTP authentication header

Repository Name: Local User Store

The configuration for the HTTP Basic Authentication filter

4. Click OK.
5. Set this authentication filter to be the starting filter of the policy by right-clicking on the filter in the canvas and selecting the "**Set as Start"** menu option.

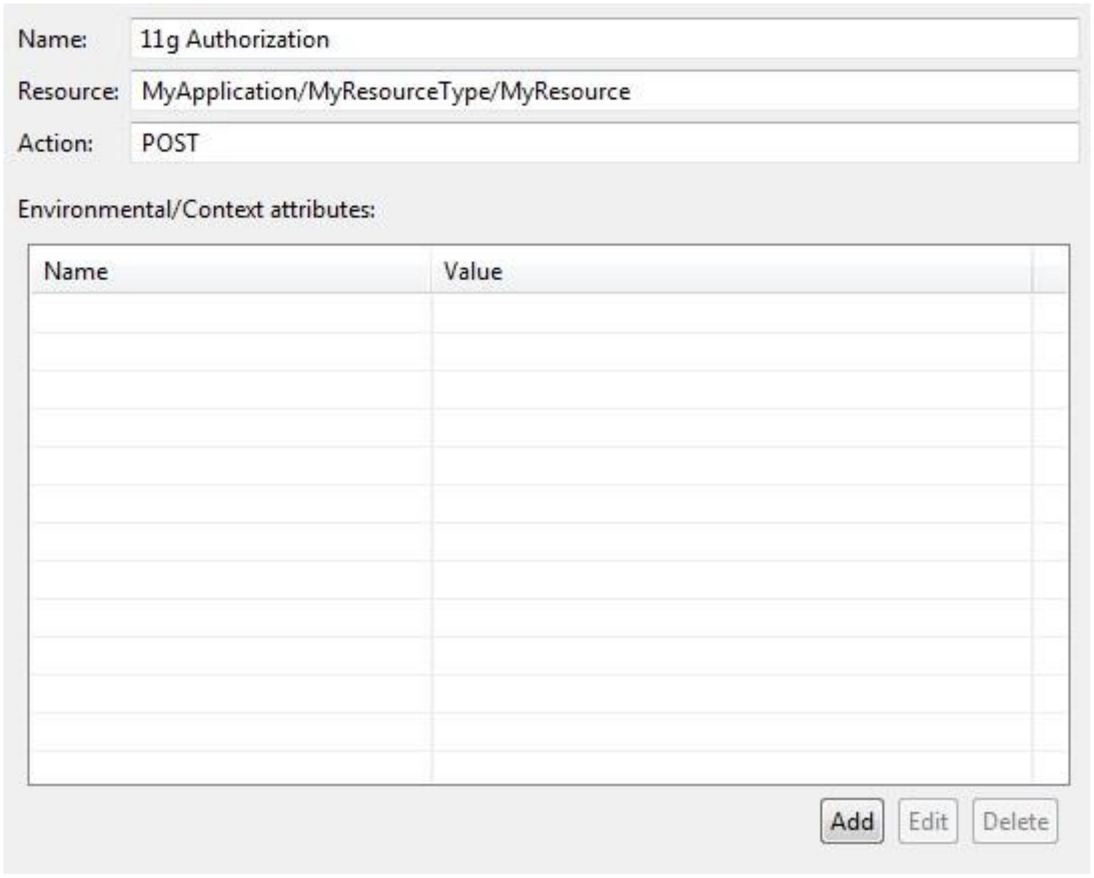
Step 2 - Configure the OES 11g Authorization Filter

To configure the OES **11g Authorization** filter, perform the following steps:

1. From the **Oracle Entitlements Server** category in the palette on the right of the Policy Studio, drag the **11g Authorization** filter onto the canvas.
2. Configure the following fields:

- **Resource:** Enter a formatted string representing the resource that we created in OES and for which the API Server will ask OES for authorization decisions. The resource we created earlier in OES can be represented with the following string: `MyApplication/MyResourceType/MyResource`
- **Action:** The rules created in the OES policy can permit/deny access to a resource based on the requested action, for example, POST, GET, DELETE, etc. In this example, we will be POST-ing the request to the resource, so we need to enter "POST" in the **Action** field. Remember, we configured POST access to the this resource earlier when configuring the OES policy.
- You can optionally configure environment context attributes. However, for the purposes of this integration guide it is not necessary to configure this section.

The completed configuration appears as follows:



Name: 11g Authorization

Resource: MyApplication/MyResourceType/MyResource

Action: POST

Environmental/Context attributes:

Name	Value

The OES 11g Authorization filter configuration

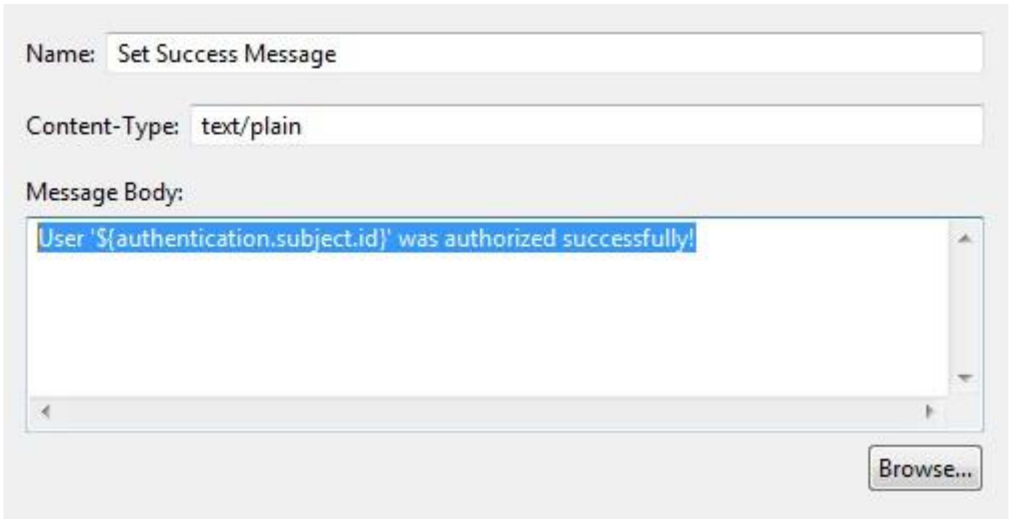
3. Click OK.
4. Set the success path from the **HTTP Basic Authentication** filter to point at the newly created OES **11g Authorization** filter.

Step 3 - Add the Success Message Filter

To display a “success” message to the user after successfully authorizing the user we can add a **Set Message** filter as follows:

1. Drag a **Set Message** filter from the **Conversion** category in the palette and drop it onto the canvas.
2. Configure the following fields on this filter:
   - **Name:** Enter “Set Success Message” in the text field.
   - **Content-type:** Enter “text/plain” as the content-type of the message to return to the client.
   - **Message Body:** Enter the following message to return to the client: “User '${authentication.subject.id}' was authorized successfully!”

The configuration for the **Set Success Message** filter should now look like this:



Name: Set Success Message

Content-Type: text/plain

Message Body:

User '${authentication.subject.id}' was authorized successfully!

Setting the success message using the Set Message filter

3. Click OK.
4. Set the *success* path of the **11g Authorization** filter to the **Set Success Message** filter.

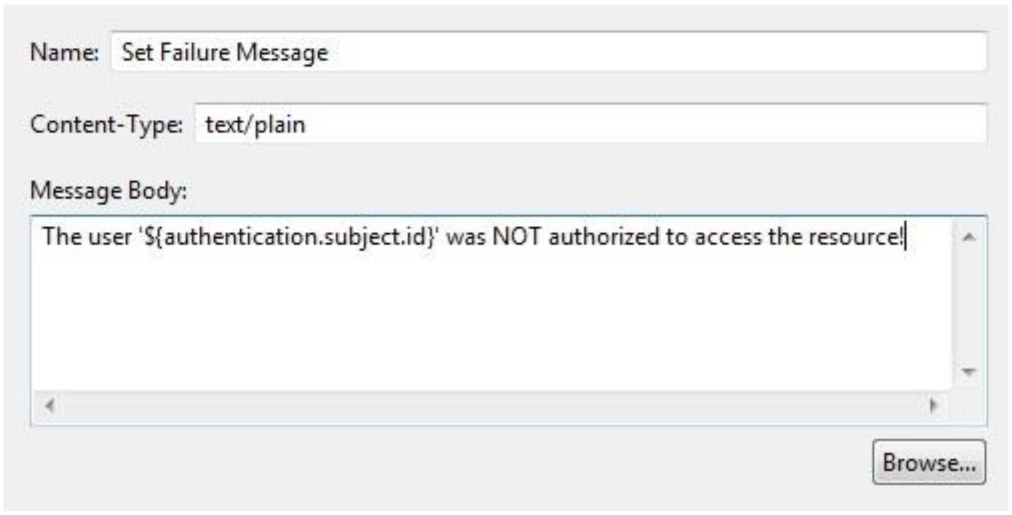
Step 4 - Add the Failure Message Filter

If OES returns false for the authorization request we should return an appropriate error message to the client. To display a “failure” message to the user after an unsuccessful authorization event we can add another **Set Message** filter as follows:

3. Drag a **Set Message** filter from the **Conversion** category in the palette and drop it onto the canvas.
4. Configure the following fields on this filter:
   - **Name:** Enter “Set Failure Message” in the text field.

- **Content-type:** Enter “text/plain” as the content-type of the message to return to the client.
- **Message Body:** Enter the following message to return the client: “The user '${authentication.subject.id}' was NOT authorized to access the resource!”

The configuration for the **Set Failure Message** filter should now look like this:


Name: Set Failure Message

Content-Type: text/plain

Message Body:

The user '${authentication.subject.id}' was NOT authorized to access the resource!

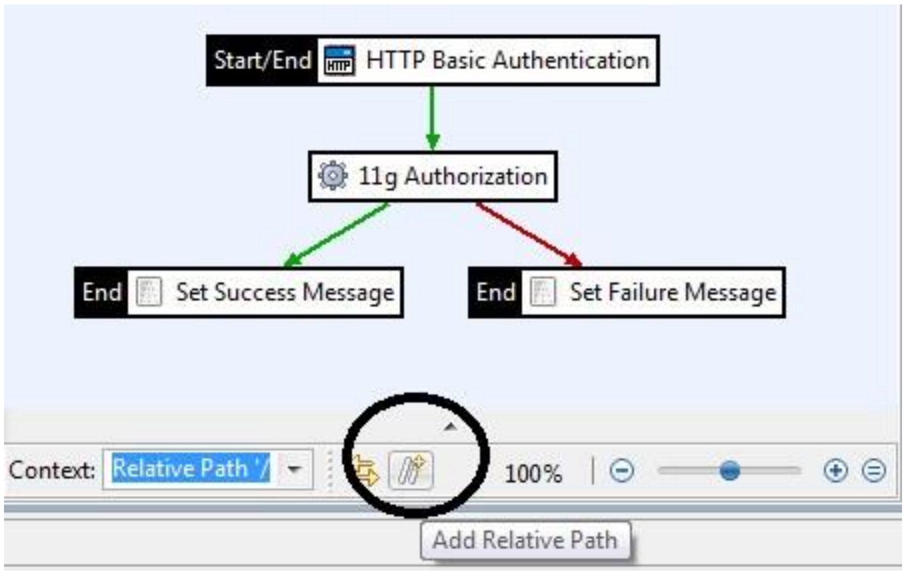
Setting the failure message using another Set Message filter

5. Click OK.
6. Set the *failure* path of the **11g Authorization** filter to the **Set Success Message** filter.

Step 5 - Add a Relative Path for the OES Authorization Policy

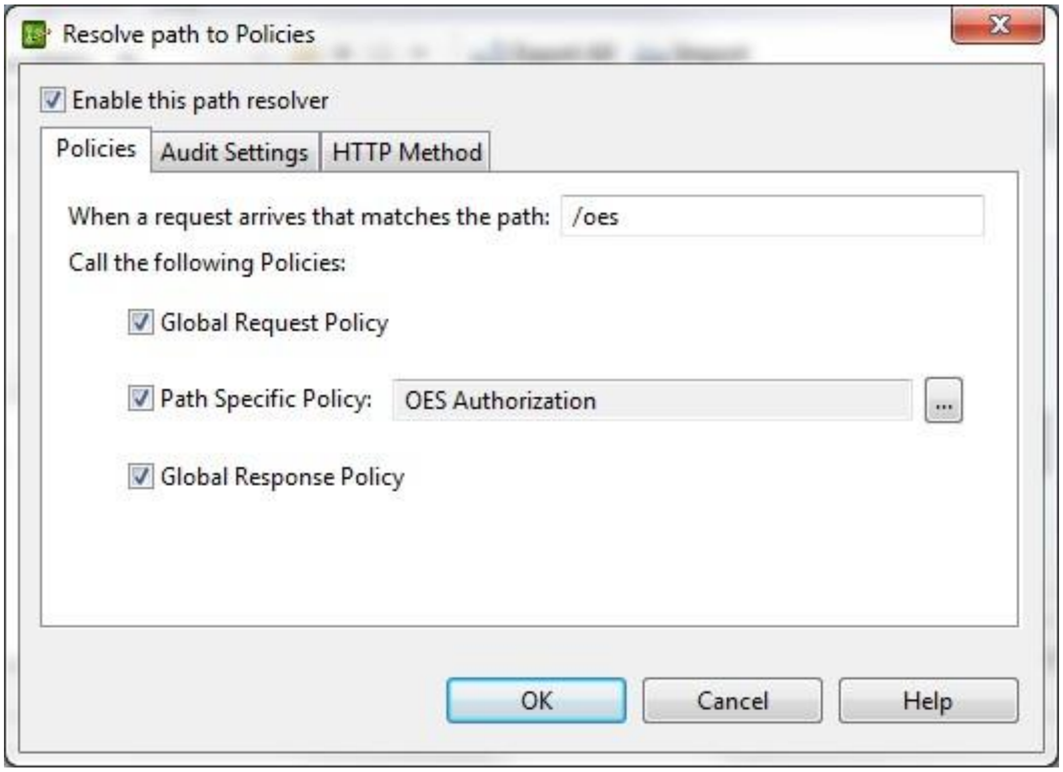
In order for the API Server to invoke this policy for certain requests we need to create a **Relative Path** and map it to the policy. All requests received on this path will be automatically forwarded to the policy for processing.

To add a **Relative Path** for this policy click on the **Add Relative Path** button in the toolbar beneath the canvas. The **Add Relative Path** button is circled in the following screenshot:



Adding a Relative Path to map incoming requests to the newly created OES Authorization policy

Enter the path on which to receive requests for this policy in the field provided in the **Resolve Path to Policies** dialog:



Resolve path to Policies

☑ Enable this path resolver

Policies | Audit Settings | HTTP Method

When a request arrives that matches the path: /oes

Call the following Policies:

☑ Global Request Policy

☑ Path Specific Policy: OES Authorization

☑ Global Response Policy

Configuring the mapping from Relative Path to a Path Specific Policy

Enter a relative path of “/oes” in the field provided. You can see that this path is automatically mapped to the “OES Authorization” policy created earlier in this section.

Step 6 - Deploy the Policy

To push the configuration changes to the live API Server instance you must *deploy* the new policy. You can do this by clicking the **F6** button.

Testing the Integration Steps

Having completed the integration steps, we can now test the setup using the **Curl** testing utility. It is important to note that you must have added a “weblogic” user to the API Server Local User Store as outlined in the prerequisites section of this guide.

Assuming you are running the API Server on a machine called “apiserver” on the default port of 8080, you can send a POST request to the authorization policy on the API Server using HTTP basic authentication with the following command:

```
>curl  --user weblogic:weblogic --data "test=data" http://
apiserver:8080/oes User 'weblogic’ was
authorized successfully!
```

We can see that the success message has been returned by the API Server meaning that the “weblogic” user has been successfully authorized by OES to access the resource. A quick look at the API Server’s trace output shows the **OES 11g Authorization Filter** making the successful authorization request for the “MyApplication/MyResourceType/MyResource” resource:

```
DEBUG   23/Oct/2012:15:28:45.183 [42687940] run circuit "OES
Authorization"...
DEBUG   23/Oct/2012:15:28:45.183 [42687940] run filter [HTTP Basic
Authentication] {
DEBUG   23/Oct/2012:15:28:45.183 [42687940]
     VordelRepository.checkCredentials: username=weblogic
DEBUG   23/Oct/2012:15:28:45.183 [42687940]      Attempt to
map incoming format Username to repository authZ format
Username
DEBUG   23/Oct/2012:15:28:45.183 [42687940] } = 1, filter [HTTP Basic
Authentication]
DEBUG   23/Oct/2012:15:28:45.183 [42687940] Filter [HTTP Basic
Authentication] completes in 0 milliseconds.
DEBUG   23/Oct/2012:15:28:45.183 [42687940] run filter [11g
Authorization] {
DEBUG   23/Oct/2012:15:28:45.183 [42687940]      creating subject
from 'weblogic'
DEBUG   23/Oct/2012:15:28:45.183 [42687940]      checking 'POST' to
resource: MyApplication/MyResourceType/MyResource
DEBUG   23/Oct/2012:15:28:45.185 [42687940]      Request: {weblogic,
POST, MyApplication/MyResourceType/MyResource}
Result: true
DEBUG   23/Oct/2012:15:28:45.185 [42687940]      result from OES:
true
DEBUG   23/Oct/2012:15:28:45.185 [42687940]      no obligations in
response
DEBUG   23/Oct/2012:15:28:45.185 [42687940] } = 1, filter [11g
Authorization]
DEBUG   23/Oct/2012:15:28:45.185 [42687940] Filter [11g
Authorization] completes in 2 milliseconds.
```

Now let's see what happens when we authenticate with a user that is present in the API Server's Local User Store, but that hasn't been configured with authorization rights in OES. To demonstrate this, we can send up a request using the credentials of the default API Server "admin" user:

```
>curl  --user admin:changeme --data "test=data" http://
apiserver:8080/oes User 'admin' was NOT authorized to
access the resource!
```

If we take a quick look at the API Server's trace output once more, we can see that the **11g Authorization** filter has blocked the authorization request:

```
DEBUG   23/Oct/2012:15:29:22.678 [41f67940] run circuit "OES
Authorization"...
DEBUG   23/Oct/2012:15:29:22.678 [41f67940] run filter [HTTP Basic
Authentication] {
DEBUG   23/Oct/2012:15:29:22.678 [41f67940]
      VordelRepository.checkCredentials: username=admin
DEBUG   23/Oct/2012:15:29:22.678 [41f67940]      Attempt to map
incoming format Username to repository authZ format Username
DEBUG   23/Oct/2012:15:29:22.678 [41f67940] } = 1, filter [HTTP Basic
Authentication]
DEBUG   23/Oct/2012:15:29:22.678 [41f67940] Filter [HTTP Basic
Authentication] completes in 0 milliseconds.
DEBUG   23/Oct/2012:15:29:22.678 [41f67940] run filter [11g
Authorization] {
DEBUG   23/Oct/2012:15:29:22.678 [41f67940]      creating subject
from 'admin'
DEBUG   23/Oct/2012:15:29:22.678 [41f67940]      checking 'POST' to
resource: MyApplication/MyResourceType/MyResource
DEBUG   23/Oct/2012:15:29:22.679 [41f67940]      Request: {admin,
POST, MyApplication/MyResourceType/MyResource}
Result: false
DEBUG   23/Oct/2012:15:29:22.679 [41f67940]      result from OES:
false
ERROR   23/Oct/2012:15:29:22.679 [41f67940]      Failed to authZ to
OES
DEBUG   23/Oct/2012:15:29:22.680 [41f67940] } = 0, filter [11g
Authorization]
DEBUG   23/Oct/2012:15:29:22.680 [41f67940] Filter [11g
Authorization] completes in 2 milliseconds.
```

Conclusion

This document outlines a simple demonstration to configure the API Gateway to authorize a user’s request for a particular resource against an 11gR2 Oracle Entitlements Server.

This configuration can be part of a larger policy, including features such as XML threat detection and conditional routing, features which are out of scope of this document but are covered in other documents which can be obtained from Oracle Technology Network.