An Oracle White Paper
June 2011

IBM Tivoli Directory Server – Oracle Enterprise Gateway Integration Guide

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

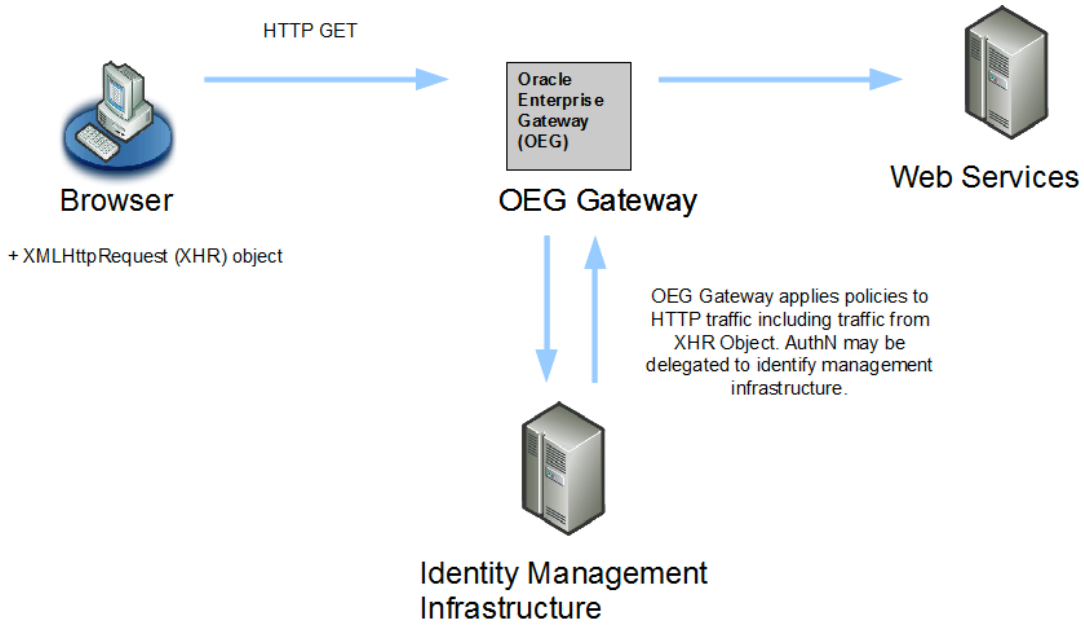
1. Introduction

1.1. Purpose

This document describes how to configure the Gateway to authenticate via an LDAP directory server and to extract attributes/roles from the LDAP repository. This will be demonstrated by the following:

1. The Gateway will be configured to authenticate a user located in a LDAP directory.
2. Upon successful authentication the Gateway will be configured to extract attributes belonging to this user from the LDAP directory.
3. A SAML Authentication Assertion will be injected into the message as proof of the authentication event, which can then be consumed by a downstream SAML-aware Web Service.

Flow of request:



This guide applies to software products version 11.1.1.x.
In this guide the LDAP Server used is **IBM Directory Server.**

1.2. LDAP Architecture

LDAP refers to *Lightweight Directory Access Protocol*. LDAP is based on a simplified version of X.500 directories. It is used to access a hierarchical directory of information on a directory server.

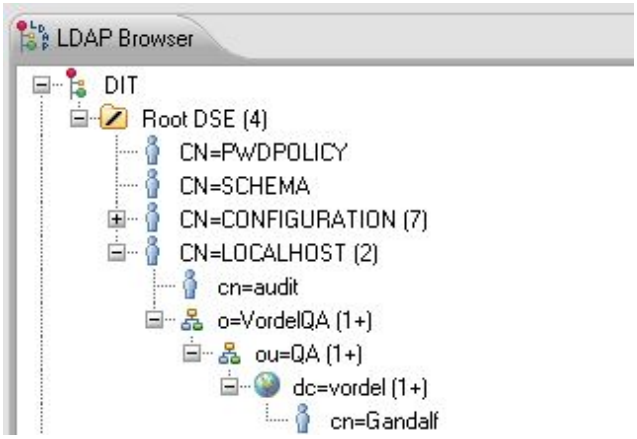
1.3. Setup Used for this Guide:

- OEG Gateway 11.1.1.5.0
- IBM Directory Server
- Apache Directory Studio (used as LDAP browser)

2. Directory Details

2.1. Directory Structure

The details of this directory are displayed here in a LDAP browser:



2.2. Connection Details

The connection details for this LDAP directory is as follows:

- LDAP URL: ldap://ibmds.qa.vordel.com:389
- user: cn=root
- password: vordel

NOTE: The connection details will differ depending on the local implementation and is defined by the Directory Administrator. As these details are going to be used in the configuration of the Gateway, it is useful to reference them here for purpose of this guide.

3. Authenticate User with HTTP Basic HTTP Filter

3.1. Create a policy to authenticate a user in the LDAP directory

3.1. The first policy that will be created is to authenticate an existing user located in IBM Directory Server. Before creating this policy it will be necessary to create a LDAP Connection and a LDAP Repository.
Creating a policy to authenticate an existing user located in an LDAP directory:

1. Click on **External Connections** on the left hand side of **Policy Studio**
1. Expand the **External Connections** Tree on the left hand side of **Policy Studio**
2. Right Click on **LDAP Connections** and Click **Add a LDAP Connection**
3. **Name:** For this guide **'IBMDS'** is used
4. For the **Type** dropdown box select "Simple"
5. Enter the Connection details to connect to the LDAP directory
6. **Realm:** Leave blank
7. Click on **Test Connection** to verify that the connection to the LDAP database has been configured successfully
8. Click on **OK**
9. The new **LDAP Connection** should be visible in the **LDAP Connections** Tree
10. Within the **External Connections** Tree expand the **Authentication Repository Profiles** Tree
11. Right Click on **LDAP Repositories** and Click **Add a new Repository**
12. **Repository Name:** For this guide **'IBMDS'** is used
13. **LDAP Store:** For the **LDAP Directory** choose the previously created LDAP connection **'IBMDS'** from the drop down list
14. Now the **User Search Conditions needs** to be specified
15. For this guide the following details are used based on the Directory information above:

- Base Criteria: CN=LOCALHOST
- User Class: 'inetOrgPerson' LDAP Class (from the drop down list)
- User Search Attribute: cn

16. For "Attributes for use in subsequent filters" the following values are used:

- Login Authentication Attribute: This can be left blank (or use Distinguished Name)
- Authorization Attribute: cn
- Authorization Attribute Format: X.509 Distinguished Name

17. Click on **OK**
18. The new **LDAP Repository** should now be visible in the **LDAP Repositories** Tree
19. Click on **Policies** and then Right Click on the **Policies** Tree on the left hand side of **Policy Studio**

20. Click **Add Policy** and name the Policy **'IBMDS'**
21. Click on the Policy and drag a **"HTTP Basic" filter** located in the **"Authentication" group** of the filter palette located on the right hand side of Policy Studio
22. **Name** of the filter can be left default or changed to any descriptive name.
23. **Credential Format:** select User Name from the drop down list
24. **Repository Name:** For this guide **'IBMDS'** is chosen from the drop down list
25. Click on **Finish**
26. The HTTP Basic filter is now properly configured and for testing purposes a **"Reflect" filter** will be added
27. Drag a reflect filter from the **"Utility" palette** and connect the HTTP Basic filter to it with a success path connector.

Explanation of values used in Step 16:
There are 2 steps involved when using the LDAP Authentication filter to authenticate a user:

1. Retrieve the user's Distinguished Name (DName) from the LDAP directory using search criteria.
2. Bind to the LDAP directory using the retrieved Distinguished Name (DName) and the user's password.

Step1: Retrieve the User's DName

The first step is to find the entry for the user in the Directory Server using search criteria. If "Gandalf" needs to be authenticated and the setup is used as per step 17:

- **Base Criteria:** CN=LOCALHOST
- **User Class**: 'inetOrgPerson' LDAP Class (from the drop down list)
- **User Search Attribute**: cn

The following LDAP search filter will be generated from these settings:
(&(objectclass=inetOrgPerson)(cn=Gandalf))

The search filter can be described as follows:
Look for the object in the hierarchy of type "inetOrgPerson ", where the attribute "cn" can be used to identify the user in the hierarchy under the base object "LOCALHOST".

In general, the two fields **User Class** and **User Search Attribute** from the search criteria section are combined to create a search filter of the form:
(&(objectclass=****User Class value goes here *****)(****User Search Attribute goes here ****=****Authentication username from HTTP Header goes here ****))

NOTE:
The configuration details for IBM Directory Server deserve a special mention as an example of a directory server which does not return a full Distinguished Name (DName) as the result of a standard LDAP user search. Instead, it returns a *contextualized* DName which is relative to the specified **Base Criteria**. In other words, the returned name is relative to the **Base Criteria**. In such cases, the Gateway can build up the full DName by combining the **Base Criteria** and the returned name. Let's look at an example of how this works in practice.
Instead of returning a full Distinguished Name (DName), such as "cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST", IBM Directory Server will return just "cn=Gandalf, OU=QA,o=VordelQA", assuming of course that "CN=LOCALHOST" was specified as the **Base Criteria**. To allow the Gateway to do this, you can leave the **Login Authentication Attribute** field blank. The Gateway will then automatically concatenate the specified **Base Criteria** (i.e. "C=LOCALHOST") with the contextualized DName returned from the directory server (i.e."cn=Gandalf,OU=QA,o=VordelQA ") to obtain the fully qualified DName (i.e."cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST").

Step 2: Bind to the LDAP Directory

Once the user's Distinguished Name (DName) has been retrieved, the Gateway will attempt to "bind" to the Directory Server on behalf of the user. The "bind" operation requires the DName returned from the search and the password provided by the user for authentication. If the Gateway can bind to the Directory Server on behalf of the client then the HTTP Basic authentication filter will pass, otherwise it will fail.
We need to configure the **Login Authentication Attribute** field before the Gateway can bind to the LDAP directory. In an LDAP directory tree, there must be one user attribute that uniquely distinguishes any one user from all the others. This is usually the Distinguished Name of the user; however, this is called different things in different LDAP directories. In IBM Directory Server, the Distinguished Name is referred to as the *distinguishedName*, and so "Distinguished Name" should be selected from the **Login Authentication Attribute** dropdown in order to uniquely identify the client authenticating to the Gateway.

The default entries in the **Login Authentication Attribute** dropdown are friendly names for common LDAP attributes that typically contain DNames:
Entry Domain Name=entrydn
Distinguished name=distinguishedName

Note that for integration with IBM Directory Server, the **Login Authentication Attribute** field can also be left blank in which case the Gateway will automatically

establish the correct attribute by querying the object name in the response from the LDAP directory after running the search filter.

Explanation of values used in Step 17:
The Authentication Repository configuration window also has a section titled "Attributes for use in subsequent filters".
For integration with IBM Directory Server, this section has been configured as follows:

- Login Authentication Attribute: Leave Blank
- Authorization Attribute: cn
- Authorization Attribute Format: X.509 Distinguished Name

In the **"Attributes for use in subsequent filters"** section, the following fields are available:

1. **Login Authentication Attribute:**

As stated earlier, the attribute specified here is used to uniquely identify the user in the LDAP directory. Typically this attribute contains the DName of the user and is used in the "bind" operation.

2. **Authorization Attribute:**

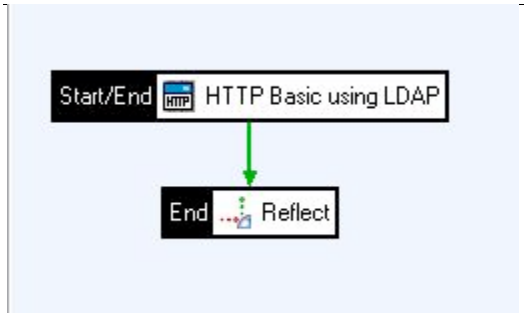
Once the client has been successfully authenticated, it is possible to use any one of that user's stored attributes in a subsequent Authorization Filter. In this case we simply want to use the user's Distinguished Name for an Authorization Filter, so we enter "cn" into the dropdown. However, any user attribute could be entered here, as long as the subsequent Authorization Filter supports it. The value of the LDAP attribute specified here will be stored in the authentication.subject.id Vordel message attribute.

3. **Authorization Attribute Format:**

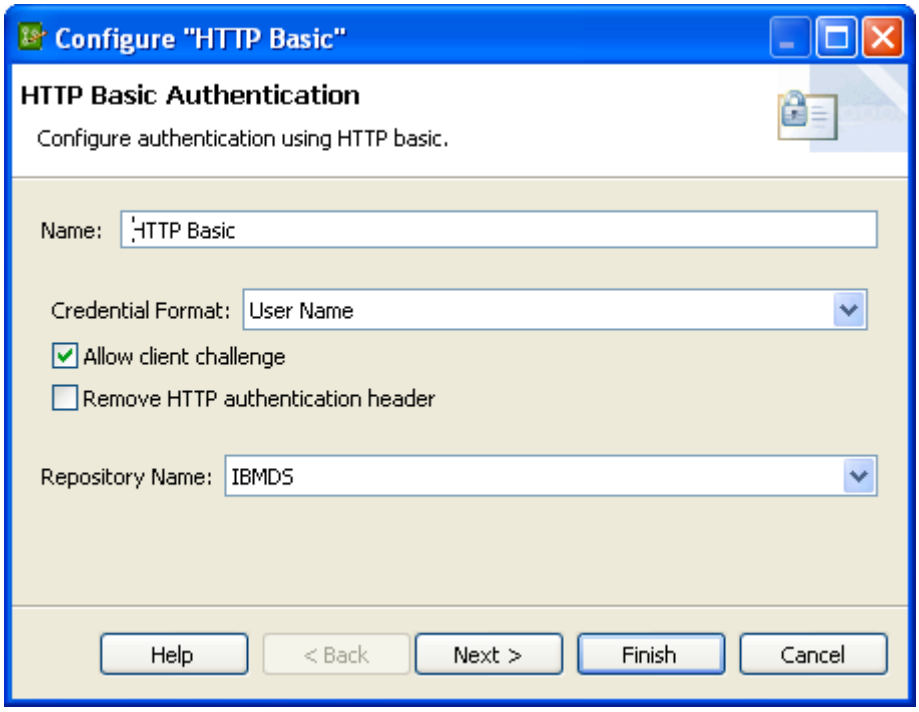
Since any user attribute can be specified in the **Authorization Attribute** above, it is necessary to inform the Gateway of the type of this attribute. This information will be used internally by the Gateway in subsequent Authorization Filters. Simply select "X.509 Distinguished Name" from the dropdown.

NOTE: The settings referred to here are specific to the IBM Directory Server being used for purpose of this guide and will differ from case to case.

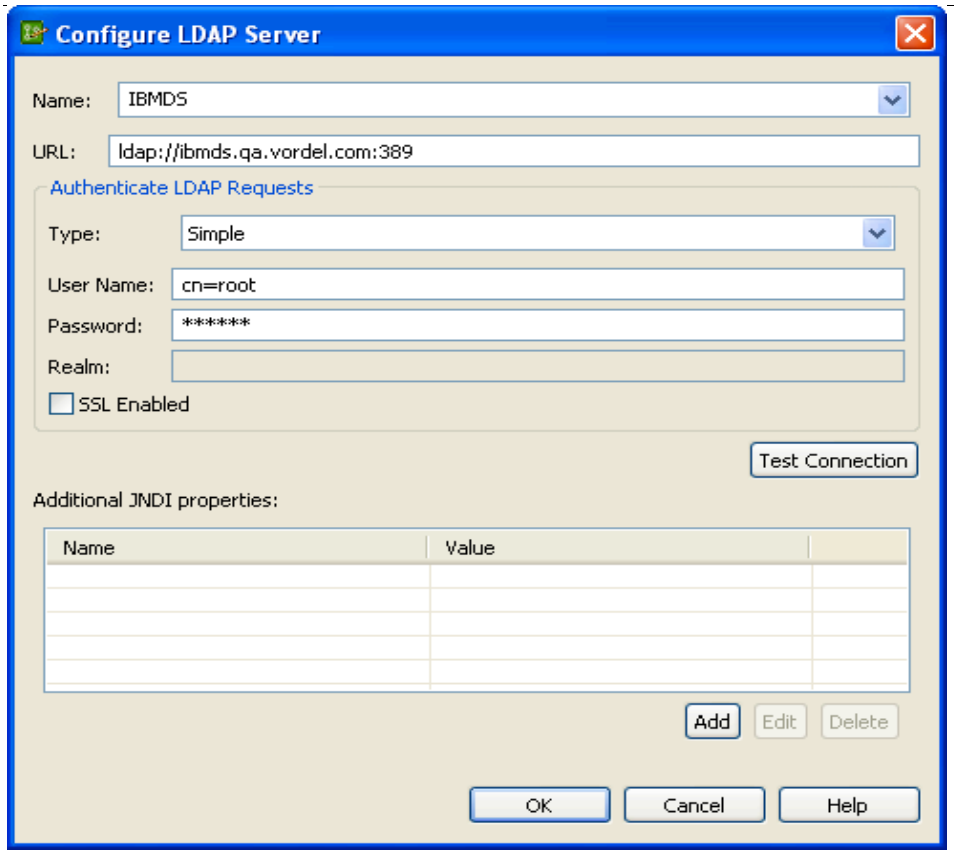
The completed Policy will look as follows:



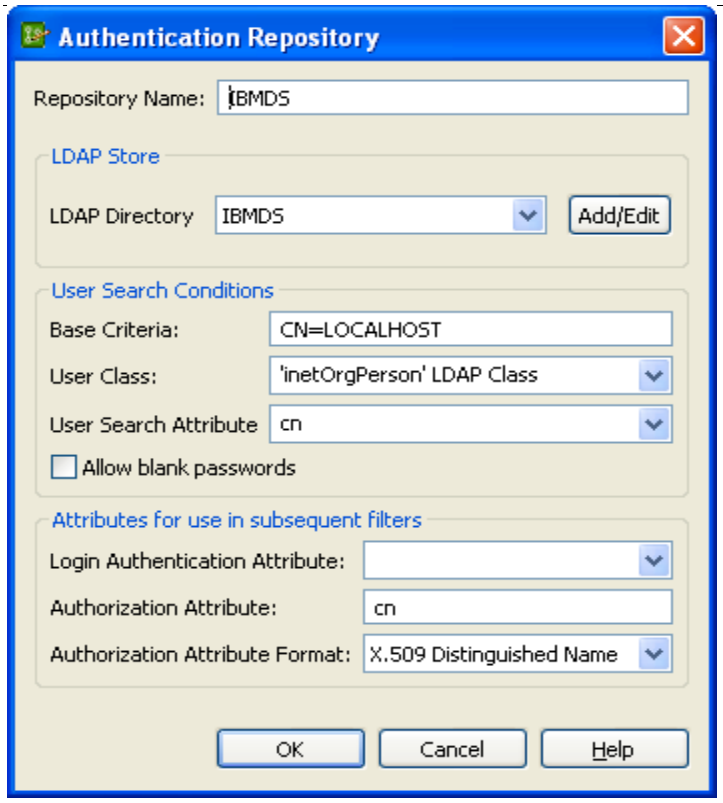
The configuration of the HTTP Basic filter as described above:



The Connection settings for the LDAP directory:



The Authentication Repository configuration:



3.2. Create a new relative path for the Policy

- Open **Policy Studio**
- Expand **Processes** and then **OEG Gateway**
- Right Click on **Default Services** and select "Add Relative Path"
- Name the Relative path as follows: /IBMDS
- Map the path to the newly created policy titled "IBM Directory Server"
- Click **OK**

The Add a relative path window:

3.3. Ensure policies are updated on the Gateway

- Open the **Policy Studio**
- Click on **Settings**
- Select **Deploy** to ensure that the changes made are propagated to the live Gateway.

3.4. Test the configuration in OEG Service Explorer

To test the policy OEG Service Explorer can be used to send through a message embedded with user credentials (Username/Password)

Set up a message in OEG Service Explorer

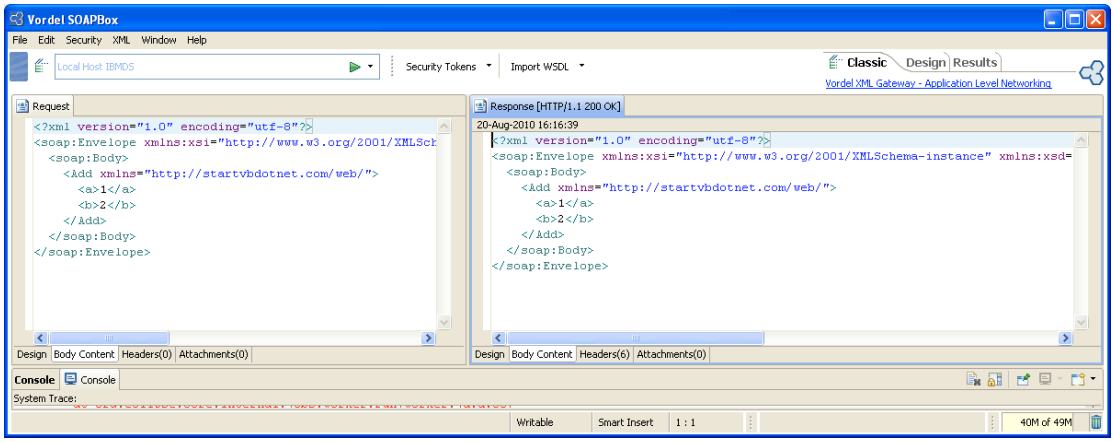
1. Open **OEG Service Explorer**
2. Load a message request
3. Click on '**Request Settings'** on the drop down list on the green '**Send Request'** button
4. Make sure that the URL is set correctly. In this case it will be http://localhost:8080/IBMDS
5. Click on the '**Security**' tab
6. Click on the **HTTP Authentication** tab
7. Select **HTTP Basic**

8. Enter the **Username** and **Password** of the user that will be Authenticated via LDAP
9. User Credentials:

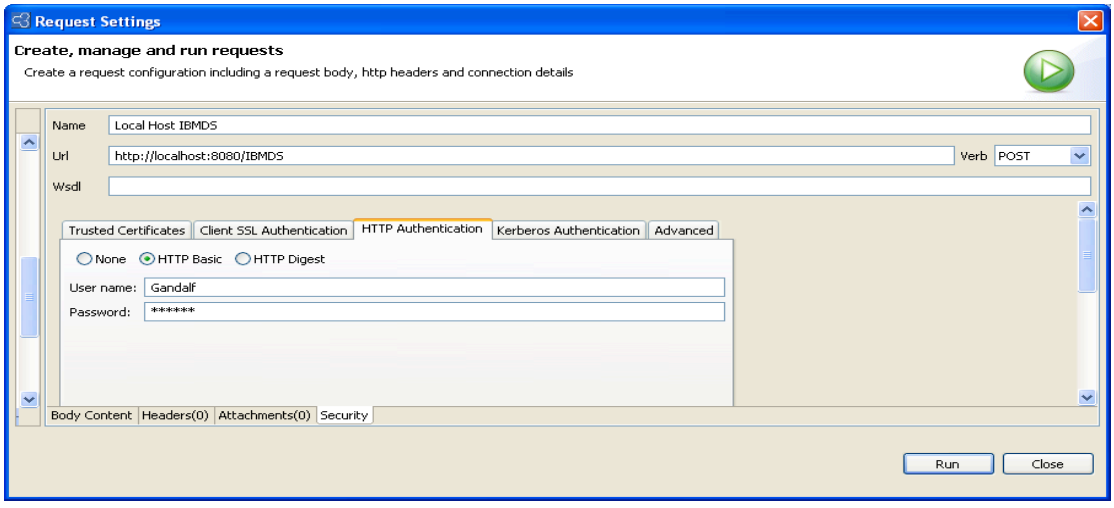
- User: Gandalf
- Password: vordel

10. Click on **Run**
11. The message would now have been sent through

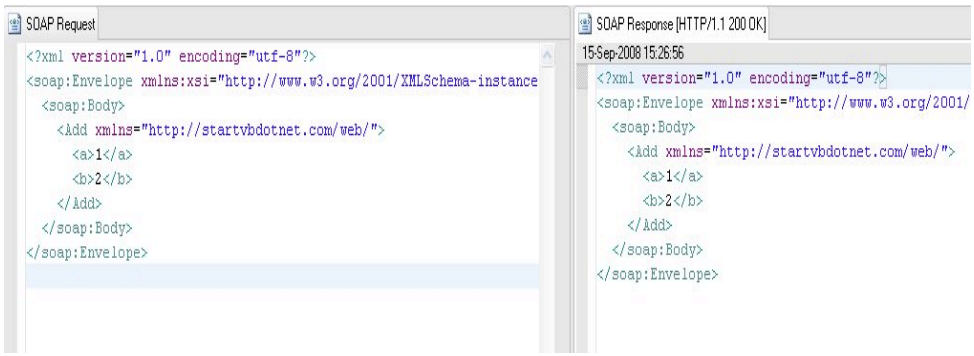
The Message loaded and Connection Settings option:



The Connection Settings Screen:



The Message Request with Embedded User credentials was processed and authenticated successfully via LDAP:



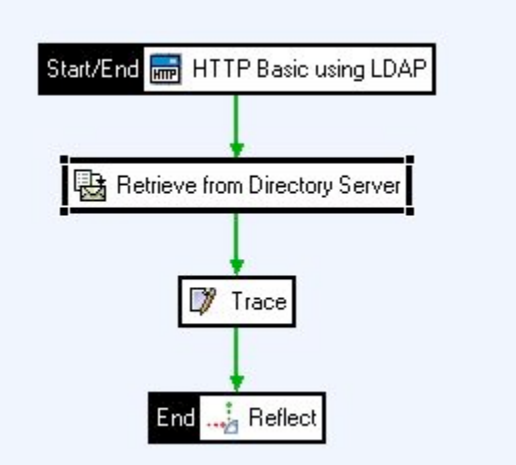
4. Adding a Retrieve from Directory Server filter

By having successfully authenticated a user from using an LDAP lookup, it is now possible to retrieve attributes from this user.

4.1. Modifying the Policy to include an "Retrieve from Directory Server" filter

1. Open **Policy Studio**
2. Click the "Authentication via LDAP" policy
3. From the "**Attributes" group** in the filter palette drag a "**Retrieve from Directory Server**" filter to the circuit
4. Also drag a "**Trace" filter** from the "**Utilities" group** of the filter palette.
5. The flow of the filters should now be, HTTP Basic->Retrieve from Directory Server->Trace->Reflect all connected with success path connectors

The modified Policy:



4.2. Configuring the Retrieve from Directory Server Filter:

1. **LDAP Directory:** (choose LDAP directory from the drop down list as configured in section 2)
2. **Retrieve Unique User Identity:** Two options are available here to choose from

- From **Message Attribute**: select "`authentication.subject.id`" (as this attribute is provided by having authenticated using the Basic HTTP filter)
  Select this option if the user ID is stored in a message attribute. A user's credentials are stored in the `authentication.subject.id` message attribute after authenticating to the Gateway and so this is the most likely attribute to enter in this field. Typically this will contain the Distinguished Name (DName) or username of the authenticated user. The name extracted from the selected message attribute will be used to query the directory server.
  Use the Steps 3 and 4 below

- From **LDAP Search**: This option can be used to specify a search location in the directory for a required attribute.
  Select this option to configure the Gateway to retrieve the user's identity from an LDAP search. Click the **Configure Directory Search** button to configure the search criteria to use to retrieve the user's identity. This option can be selected in cases where the `authentication.subject.id` attribute has not been pre-populated by an authentication filter. In this case the user's unique Distinguished Name must be retrieved from the LDAP repository.
  Use the Steps 5 and 6 below

Retrieve Unique User Identity from Message Attribute:-

3. **Base Criteria**: dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST

4. **Search Filter:** (&(objectclass=inetOrgPerson)(cn=${authentication.subject.id}))

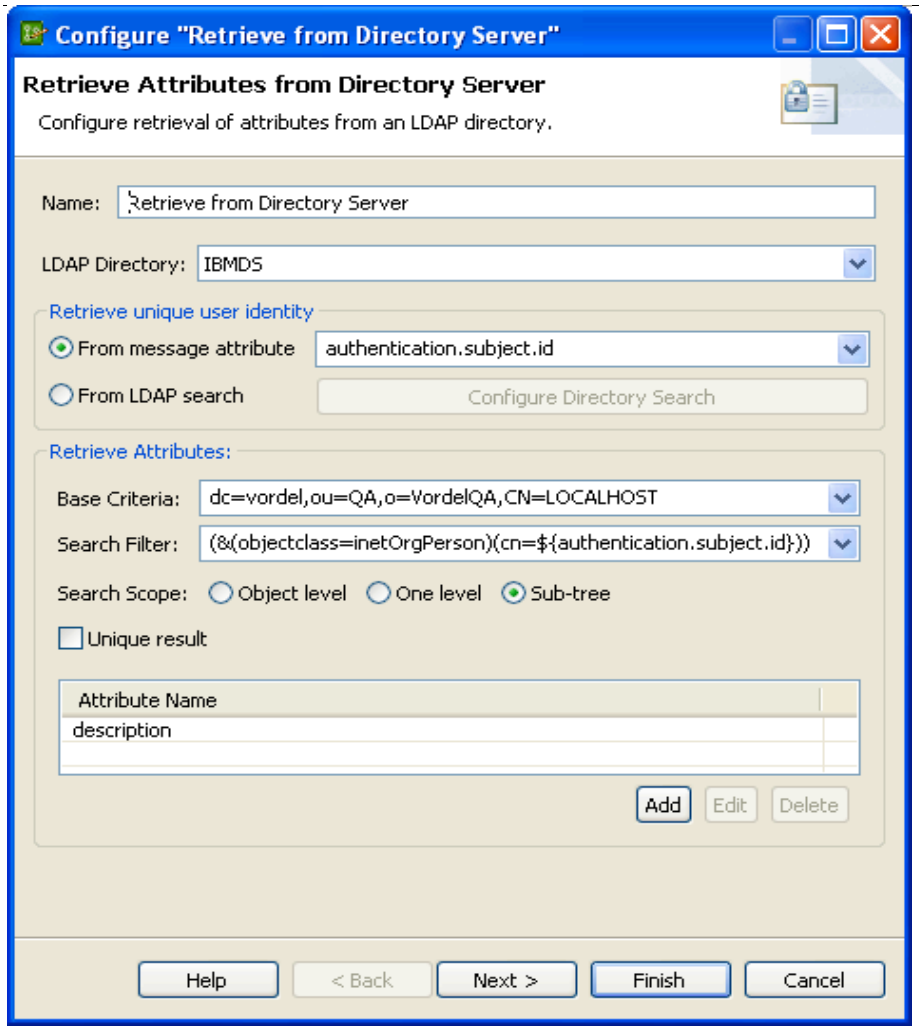
Retrieve Unique User Identity from LDAP Search:-

5. **Base Criteria**: dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST
6. **Query Search Filter:**
   (&(objectclass=inetOrgPerson)(cn=${authentication.subject.id}))
7. **Search Scope:** Sub Tree is selected
8. The **Attribute Name** table lists the attributes that the Gateway will retrieve from the user profile. If no attributes are explicitly listed here, the Gateway will extract all user attributes. In both cases, the retrieved attributes will be set to the `attribute.lookup.list` message attribute. For this guide and additional user attribute has been added:
   Attribute name: description
   Value: Wizard

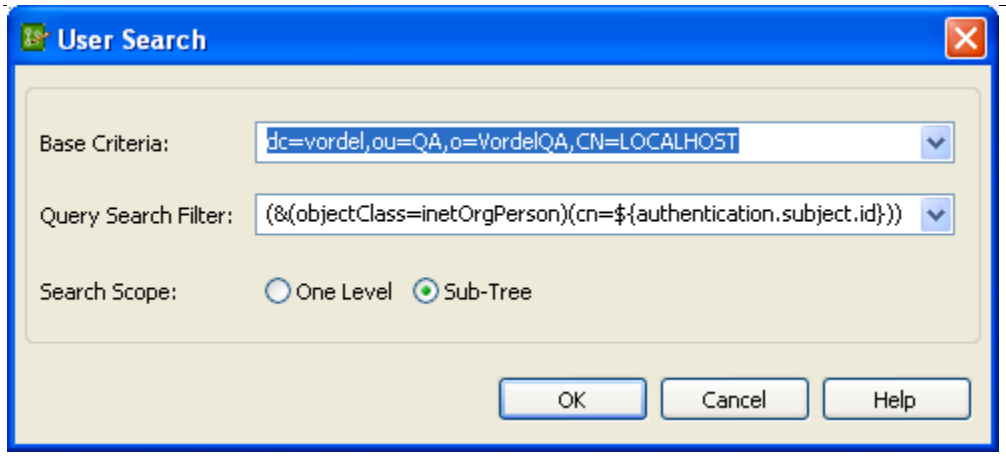
   So the Attribute value added is **"description"**. This should return the value **"Wizard"** when the messaged is being process by the Gateway.
   The search options above are using the base criteria of the directory structure as far down as the Common Name Object: User
   The Query syntax used can also be validated by performing a search in an LDAP browser using the same string:

(&(objectclass=user)(CN=Gandalf))

NOTE: The settings referred to here are specific to the IBM Directory Server setup being used for purpose of this guide and will differ from case to case.
The Retrieve from Directory Server configuration:



The "Configure Directory Search" configuration screen:



4.3. Ensure policies are updated on the Gateway

- Open the **Policy Studio**
- Click on **Settings**
- Select **Deploy** to ensure that the changes made are propagated to the live Gateway

4.4. Test the configuration in OEG Service Explorer

With the "Retrieve from Directory Server" and "Trace" filter added it is worthwhile to test the Policy again using OEG Service Explorer
Set up a message in OEG Service Explorer

1. Open **OEG Service Explorer**
2. Load a message request
3. Click on '**Request Settings'** on the drop down list on the green '**Send Request'** button
4. Make sure that the URL is set correctly. In this case it will be http://localhost:8080/IBMDS
5. Click on the **'Security'** tab
6. Click on the **HTTP Authentication** tab
7. Select **HTTP Basic**
8. Enter the **Username** and **Password** of the user that will be Authenticated via LDAP
9. User credentials:
   - Username: Gandalf
   - Password: vordel
10. Click on **Finish**
11. The message would now have been sent through

With the "Trace" Filter added it is also possible to view the attribute retrieval that occurred:
Extract from Trace above showing Attribute retrieval:

```
---------------------------------------------------------------------------------------------------------
---------
DEBUG   17:19:51:543 [0b4c] run filter [Retrieve from
Directory Server] {
DEBUG   17:19:51:543 [0b4c]     Searching for a user
identity with base
[dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST] and filter
[(&(objectclass=inetOrgPerson)(cn=Gandalf))]
DEBUG   17:19:51:543 [0b4c]     The ldap service provider
is com.sun.jndi.ldap.LdapCtxFactory
DEBUG   17:19:51:543 [0b4c]     adding the additional jndi
properties: {}
DEBUG   17:19:51:543 [0b4c]     The context is created for
the LDAP lookup
DEBUG   17:19:51:543 [0b4c]
LdapLookup.getUserIdentity: The user's unique  identity is
[cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST]
DEBUG   17:19:51:543 [0b4c]     LookupHandler.process:
userIdentity:
cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST
DEBUG   17:19:51:543 [0b4c]     Looking up user cache with
the key:
cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST
DEBUG   17:19:51:543 [0b4c]     User's attribute from
cache: (null)
DEBUG   17:19:51:543 [0b4c]     No attributes for user in
cache so do lookup
DEBUG   17:19:51:543 [0b4c]     The user identity whose
attributes are looked for is
[cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST]
DEBUG   17:19:51:543 [0b4c]     Searching for a attributes
with base [dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST] and
filter [(&(objectclass=inetOrgPerson)(cn=Gandalf))]
DEBUG   17:19:51:543 [0b4c]     The context is created for
the LDAP lookup
DEBUG   17:19:51:543 [0b4c]     Retrieving attributes for
the result cn=Gandalf
DEBUG   17:19:51:543 [0b4c]
LdapLookup.addToAttributeHashMap: attribute=[description]
value=[Wizard]
```

```
DEBUG   17:19:51:543 [0b4c]
LdapAttrLookupHandler.getAttributes:
Attributes={description=key=[description]
name=[description] values=[Wizard]
namespace=[##nonamespace##]
namespaceForAssertion=[urn:vordel:attribute:1.0]
useForAssertion=[true]}
DEBUG   17:19:51:543 [0b4c]     Retrieved attributes:
1===> key=[description] name=[description] values=[Wizard]
namespace=[##nonamespace##]
namespaceForAssertion=[urn:vordel:attribute:1.0]
useForAssertion=[true]
DEBUG   17:19:51:543 [0b4c]     Copy user attribute
[description] value=[Wizard] to message attribute
[user.description]
DEBUG   17:19:51:543 [0b4c] } = 1, in 0 milliseconds
DEBUG   17:19:51:543 [0b4c] run filter [Trace] {
DEBUG   17:19:51:543 [0b4c]         Trace {
DEBUG   17:19:51:543 [0b4c]
attribute.lookup.list {
DEBUG   17:19:51:543 [0b4c]                 Value:
{description=key=[description] name=[description]
values=[Wizard] namespace=[##nonamespace##]
namespaceForAssertion=[urn:vordel:attribute:1.0]
useForAssertion=[true]}
DEBUG   17:19:51:543 [0b4c]                 Type:
java.util.HashMap
DEBUG   17:19:51:543 [0b4c]             }
DEBUG   17:19:51:543 [0b4c]
attribute.subject.format {
DEBUG   17:19:51:543 [0b4c]                 Value:
Unspecified
DEBUG   17:19:51:543 [0b4c]                 Type:
java.lang.String
DEBUG   17:19:51:543 [0b4c]             }
DEBUG   17:19:51:543 [0b4c]
attribute.subject.id {
DEBUG   17:19:51:543 [0b4c]                 Value:
cn=Gandalf,dc=vordel,ou=QA,o=VordelQA,CN=LOCALHOST
DEBUG   17:19:51:543 [0b4c]                 Type:
java.lang.String
DEBUG   17:19:51:543 [0b4c]             }
DEBUG   17:19:51:559 [0b4c]
```

```
authentication.subject.id {
DEBUG   17:19:51:559 [0b4c]                 Value:
Gandalf
DEBUG   17:19:51:559 [0b4c]                 Type:
java.lang.String
DEBUG   17:19:51:559 [0b4c]             }
……….
DEBUG   17:19:51:574 [0b4c]                 Value:
Wizard
DEBUG   17:19:51:574 [0b4c]                 Type:
java.lang.String
DEBUG   17:19:51:574 [0b4c]             }
DEBUG   17:19:51:574 [0b4c]         }
DEBUG   17:19:51:574 [0b4c] } = 1, in 31 milliseconds
DEBUG   17:19:51:574 [0b4c] run filter [Reflect] {
```

As can be seen in the trace the attribute "description" has been retrieved as specified in the "Retrieve from Directory Server" filter with a value of "Wizard". If an attribute value is not specified it will retrieve all relevant attributes for the particular user.

5. Adding an Insert SAML Authentication Assertion filter

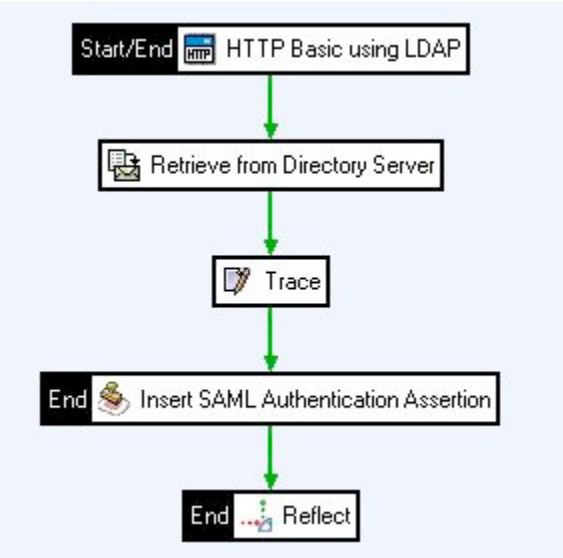
With the Basic HTTP authorization and Attribute retrieval from the directory server having been completed successfully, the policy will yet again be modified to include a SAML Assertion filter.

5.1. Add an "Insert SAML Authentication Assertion" filter

Complete the following steps to refresh the policies:

1. Open **Policy Studio**
2. Click the "**IBMDS"** policy
3. From the "**Authentication"** group in the filter palette drag a "**Insert SAML Authentication Assertion"** filter to the circuit
4. The flow of the filters should now be, HTTP Basic->Retrieve from Directory Server->Trace->Insert SAML Authentication Assertion->Reflect all connected with success path connectors

The modified Policy after having added the "Insert SAML Authentication Assertion" filter:



NOTE: The "Trace" filter here is not a prerequisite and has only been added for showing attribute retrieval in the trace output as demonstrated in section 4. This could be left out or removed from the flow above.

5.2. Configuring the "Insert SAML Authentication Assertion" filter:

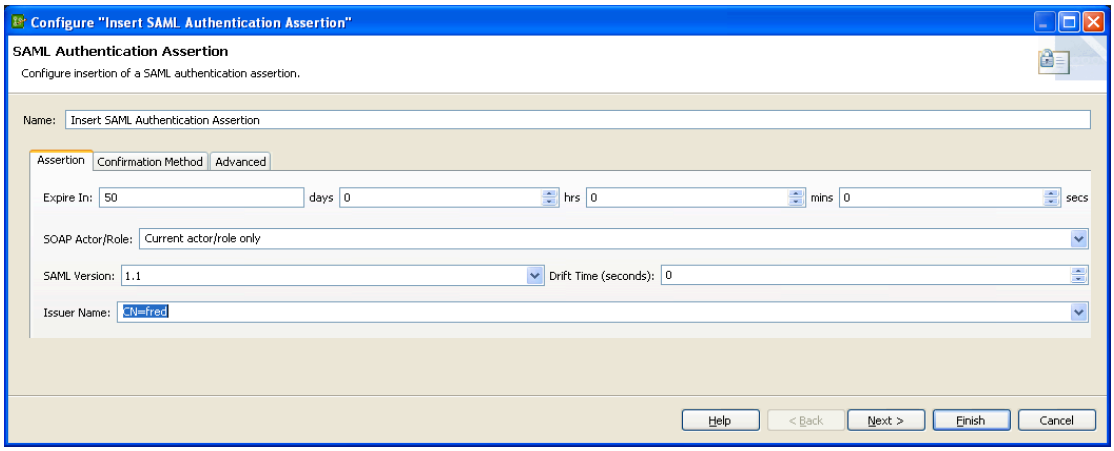
On the **Assertion Tab:**

1. **Expiry Date:** Set to any desired value
2. **SOAP Actor/Role:** Choose "Current Actor/Role Only" from the drop down list
3. Select any value from the drop down field for **Issuer Name**

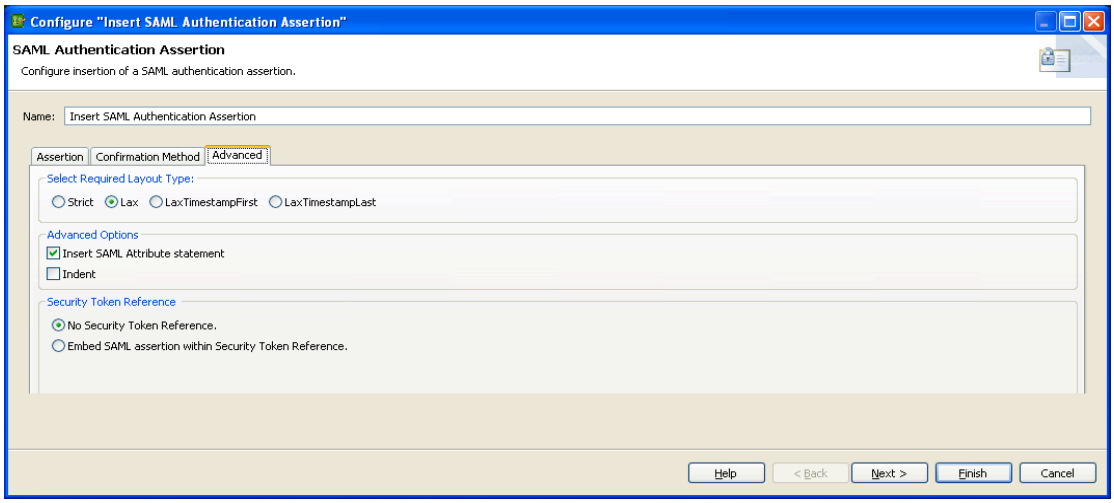
On the **Advanced Tab**

4. Tick **'Insert SAML Attribute Statement'**
5. The rest of the options could be left default.
6. Click on **Finish**

The "SAML Authentication Assertion" Filter configuration – Assertion Tab:



The "SAML Authentication Assertion" Filter configuration – Advanced Tab:



5.3. Test the configuration in OEG Service Explorer

With the "Insert SAML Authentication Assertion" filter added to the policy OEG Service Explorer will be used to verify the configuration.

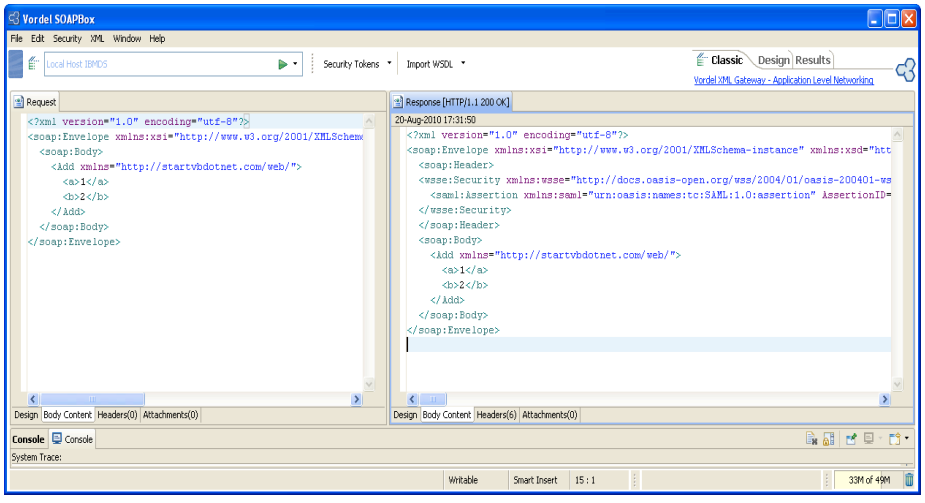
Set up a message in OEG Service Explorer

1. Open **OEG Service Explorer**
2. Load a message request
3. Click on **Settings** just above the **Send Request** button
4. Then Click on **Connection Settings**
5. Make sure that the URL is set correctly. In this case it will be http://localhost:8080/LDAP
6. Click on **OK**
7. Click on the **HTTP Authentication** tab
8. Select **HTTP Basic**
9. Enter the **Username** and **Password** of the user that will be Authenticated via LDAP
10. User Credentials:

- Username: Gandalf
- Password: vordel

11. Click on **Finish**
12. The message would now have been sent through

The Message can be seen having gone through successfully with the SAML Authentication Assertion embedded in the message:



A snippet of the SAML Assertion after successfully sending message:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
  <wsse:Security xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd">
    <saml:Assertion
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
AssertionID="Id-0000012a90578a75-00000000004daccc-32"
IssueInstant="2010-08-20T16:31:50Z" Issuer="CN=AAA
Certificate Services, O=Comodo CA Limited, L=Salford,
ST=Greater Manchester, C=GB" MajorVersion="1"
MinorVersion="1"><saml:Conditions NotBefore="2010-08-
20T16:31:50Z" NotOnOrAfter="2010-10-
09T16:31:50Z"/><saml:AuthenticationStatement
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:passw
ord" AuthenticationInstant="2010-08-
20T16:31:50Z"><saml:Subject><saml:NameIdentifier
Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">Gandalf</saml:NameIdentifier><saml
:SubjectConfirmation><saml:ConfirmationMethod>urn:oasis:na
mes:tc:SAML:1.0:cm:sender-
vouches</saml:ConfirmationMethod></saml:SubjectConfirmatio
n></saml:Subject></saml:AuthenticationStatement><saml:Attr
ibuteStatement><saml:Subject><saml:NameIdentifier
Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">Gandalf</saml:NameIdentifier><saml
:SubjectConfirmation><saml:ConfirmationMethod>urn:oasis:na
mes:tc:SAML:1.0:cm:sender-
vouches</saml:ConfirmationMethod></saml:SubjectConfirmatio
n></saml:Subject><saml:Attribute
AttributeName="description"
AttributeNamespace="urn:vordel:attribute:1.0"><saml:Attrib
uteValue>Wizard</saml:AttributeValue></saml:Attribute></sa
ml:AttributeStatement></saml:Assertion>
  </wsse:Security>
  </soap:Header>
  <soap:Body>
    <Add xmlns="http://startvbdotnet.com/web/">
      <a>1</a>
      <b>2</b>
```

```
    </Add>
  </soap:Body>
</soap:Envelope>
```

As can been seen the attribute specified under the Attribute Name section of the "Retrieve from Directory Server" filter will return the valid attribute. If left blank it will return all attributes for that user.

6. Conclusion

This configuration can be part of a larger policy, including features such as XML threat detection and conditional routing, features which are out of the scope of this document but are covered in other documents which can be obtained from Oracle at http://www.oracle.com.

Oracle Enterprise Gateway
May 2011
Author:

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Oracle is committed to developing practices and products that help protect the environment

SOFTWARE. HARDWARE. COMPLETE.