An Oracle White Paper
May 2011

Microsoft Active Directory – Oracle Enterprise Gateway Integration Guide

Disclaimer

1. Introduction 4
1.1. Purpose... 4
1.2. LDAP Architecture 5
1.3. Setup Used for this Guide:.. 5
2. Directory Details.. 5
2.1. Directory Structure 5
2.2. Connection Details... 7
3. Authenticate User with HTTP Basic HTTP Filter 7
3.1. Create a policy to authenticate a user in the LDAP directory 7
3.2. Create a new relative path for the Policy 13
3.3. Ensure policies are updated on the Gateway 14
3.4. Test the configuration in OEG Service Explorer 14
4. Adding a Retrieve from Directory Server filter 18
4.1. Modifying the Policy to include an 'Retrieve from Directory Server' filter..18
4.2. Configuring the Retrieve from Directory Server Filter: 18
4.3. Ensure policies are updated on the Gateway 22
4.4. Test the configuration in OEG Service Explorer 22
5. Adding an 'Insert SAML Authentication Assertion' filter.................... 25
5.1. Add an 'Insert SAML Authentication Assertion' filter.................. 25
5.2. Configuring the 'Insert SAML Authentication Assertion' filter:.... 26
5.3. Test the configuration in OEG Service Explorer 27
6. Conclusion 29
7. Appendix 30

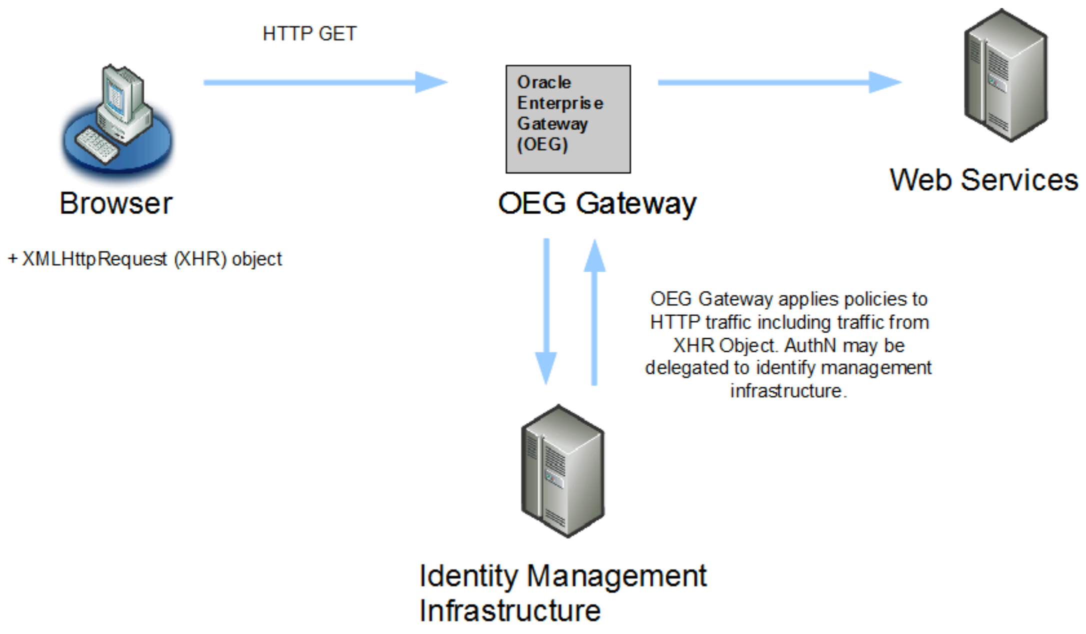
1. Introduction

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:


HTTP GET
Oracle Enterprise Gateway (OEG)
Browser
OEG Gateway
Web Services
+ XMLHttpRequest (XHR) object
OEG Gateway applies policies to HTTP traffic including traffic from XHR Object. AuthN may be delegated to identify management infrastructure.
Identity Management Infrastructure

This guide applies to OEG software products, from version 11.1.1.x upwards.
In this guide the LDAP Server used is **Microsoft Active Directory.**

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.

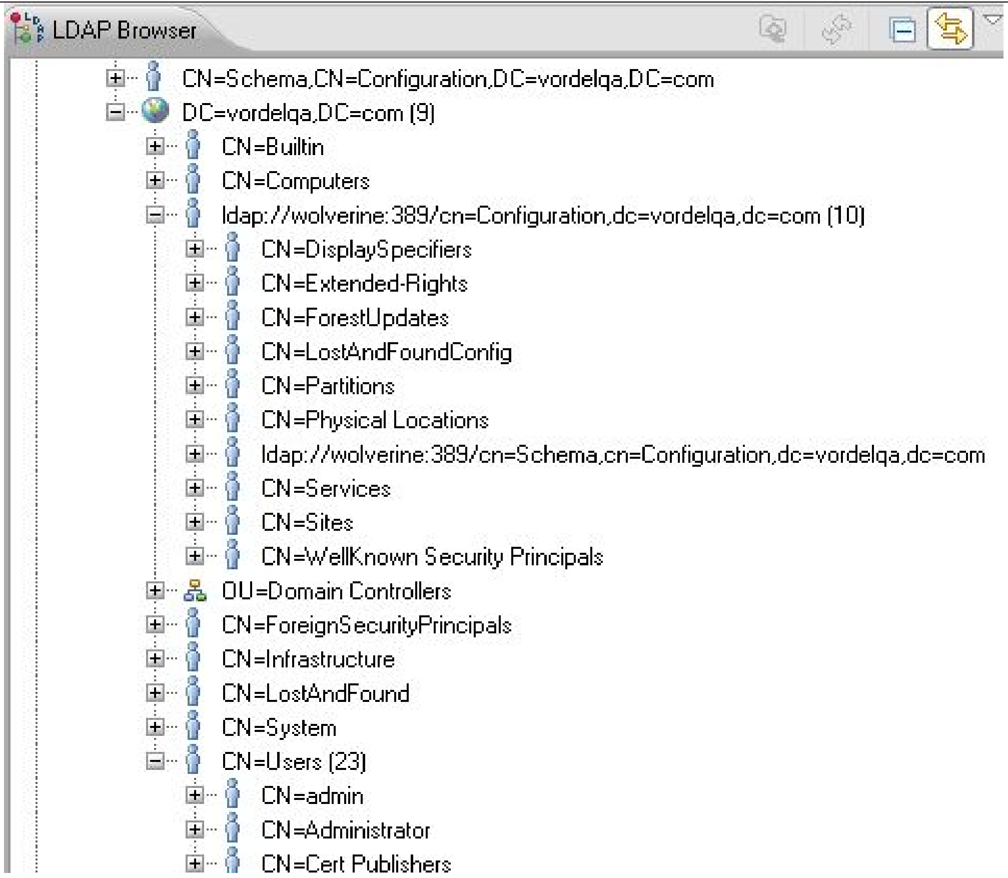
1.3. Setup Used for this Guide:

- OEG Gateway 11.1.1.5.0
- Microsoft Active Directory
- 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:



```
CN=Schema,CN=Configuration,DC=vordelqa,DC=com
DC=vordelqa,DC=com (9)
    CN=Builtin
    CN=Computers
    ldap://wolverine:389/cn=Configuration,dc=vordelqa,dc=com (10)
        CN=DisplaySpecifiers
        CN=Extended-Rights
        CN=ForestUpdates
        CN=LostAndFoundConfig
        CN=Partitions
        CN=Physical Locations
        ldap://wolverine:389/cn=Schema,cn=Configuration,dc=vordelqa,dc=com
        CN=Services
        CN=Sites
        CN=WellKnown Security Principals
    OU=Domain Controllers
    CN=ForeignSecurityPrincipals
    CN=Infrastructure
    CN=LostAndFound
    CN=System
    CN=Users (23)
        CN=admin
        CN=Administrator
        CN=Cert Publishers
```

2.2. Connection Details

The connection details for this LDAP directory is as follows:

- Hostname: wolverine
- Port: 389
- URL: ldap://wolverine:389
- User: cn=Administrator,cn=users,dc=vordelqa,dc=com
- 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

The first policy that will be created is to authenticate an existing user located in Active Directory. 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
2. Expand the External Connections Tree on the left hand side of Policy Studio
3. Right Click on LDAP Connections and Click Add a LDAP Connection
4. Name: For this guide 'Active Directory' is used
5. For the Type dropdown box select "Simple"
6. Enter the Connection details to connect to the LDAP directory
7. Realm: Leave blank
8. Click on Test Connection to verify that the connection to the LDAP database has been configured successfully
9. Click on OK
10. The new LDAP Connection should be visible in the LDAP Connections Tree
11. Within the External Connections Tree expand the Authentication Repository Profiles Tree
12. Right Click on LDAP Repositories and Click Add a new Repository
13. Repository Name: For this guide 'Active Directory' is used
14. LDAP Store: For the LDAP Directory choose the previously created LDAP connection 'Active Directory' from the drop down list

15. Now the User Search Conditions needs to be specified
16. For this guide the following details are used based on the Directory information above:
    - Base Criteria: CN=Users,DC=vordelqa,DC=com
    - User Class: 'User' LDAP Class (from the drop down list)
    - User Search Attribute: cn

Please see the section below for more information on configuring these fields.

17. For "Attributes for use in subsequent filters" the following values are used:
    - Login Authentication Attribute: Distinguished Name (or can be left blank)
    - Authorization Attribute: distinguishedName
    - Authorization Attribute Format: X.509 Distinguished Name

Please see the section below for more information on configuring these fields.

18. Click on OK
19. The new LDAP Repository should now be visible in the LDAP Repositories Tree
20. Click on Policies and then Right Click on the Policies Tree on the left hand side of Policy Studio
21. Click Add Policy and name the Policy 'Active Directory'
22. 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
23. Name of the filter can be left default or changed to any descriptive name
24. Credential Format: Select User Name from the drop down list
25. Repository Name: For this guide 'Active Directory' is chosen from the drop down list
26. Click on Finish
27. The HTTP Basic filter is now properly configured and for testing purposes a "Reflect" filter will be added
28. 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 "Hubert Farnsworth" needs to be authenticated and the setup is used as per step 17:

- Base Criteria: CN=Users,DC=vordelqa,DC=com
- User Class: 'User' LDAP Class (from the drop down list)
- User Search Attribute: cn

The following LDAP search filter will be generated from these settings:
(&(objectclass=User)(cn=Administrator))
The search filter can be described as follows:
Look for the object in the hierarchy of type "User", where the attribute "cn" can be used to identify the user in the hierarchy under the base object "Users".
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 ****))

If the user is found in the Directory Server, the Distinguished Name is returned, in this case:
cn=Administrator,CN=Users,DC=vordelqa,DC=com

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 Active Directory, 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 Active Directory, 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 Active Directory, this section has been configured as follows:
Login Authentication Attribute: Distinguished Name (or can be left blank)
Authorization Attribute: distinguishedName
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:

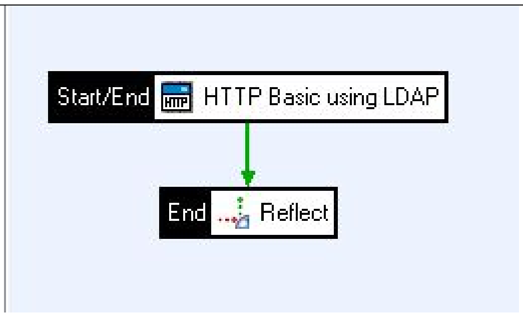
Once the client has been successfully authenticated, it is possible to use any one of that user's stored attributes in a subsequent Authorization Filter. In this case we simply want to use the user's Distinguished Name for an Authorization Filter, so we enter "distinguishedName" into the dropdown. However, 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:

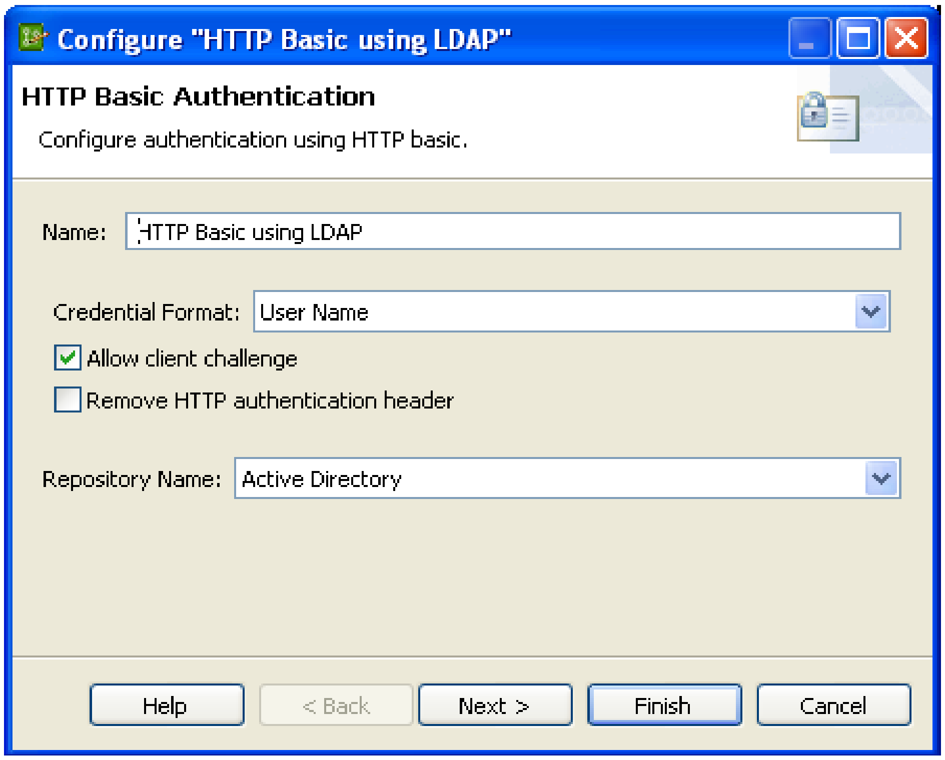
Since any user attribute can be specified in the Authorization Attribute above, it is necessary to inform the Gateway of the type of this attribute. This information will be used internally by the Gateway in subsequent Authorization Filters. Simply select "X.509 Distinguished Name" from the dropdown.

NOTE: The settings referred to here are specific to the Active Directory 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:



Configure "HTTP Basic using LDAP"

HTTP Basic Authentication

Configure authentication using HTTP basic.

Name: HTTP Basic using LDAP

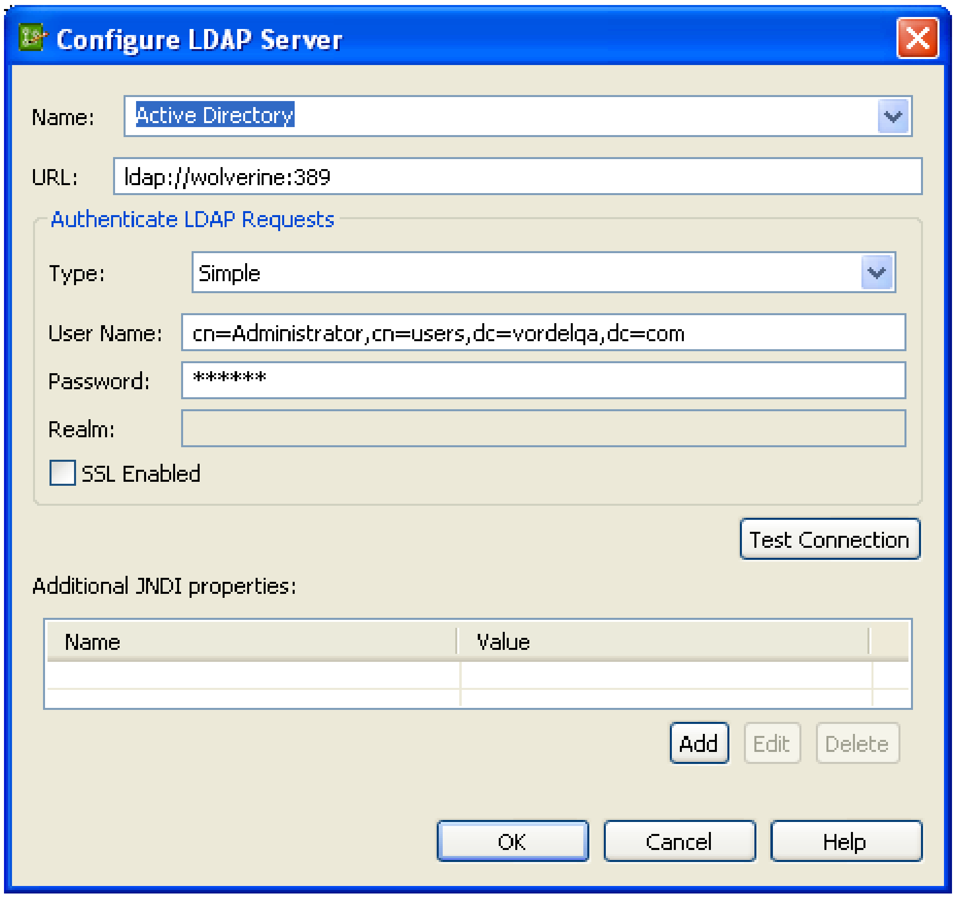
Credential Format: User Name

[✓] Allow client challenge

[ ] Remove HTTP authentication header

Repository Name: Active Directory

The Connection settings for the LDAP directory:



Configure LDAP Server

Name: Active Directory

URL: ldap://wolverine:389

Authenticate LDAP Requests

Type: Simple

User Name: cn=Administrator,cn=users,dc=vordelqa,dc=com

Password: *******

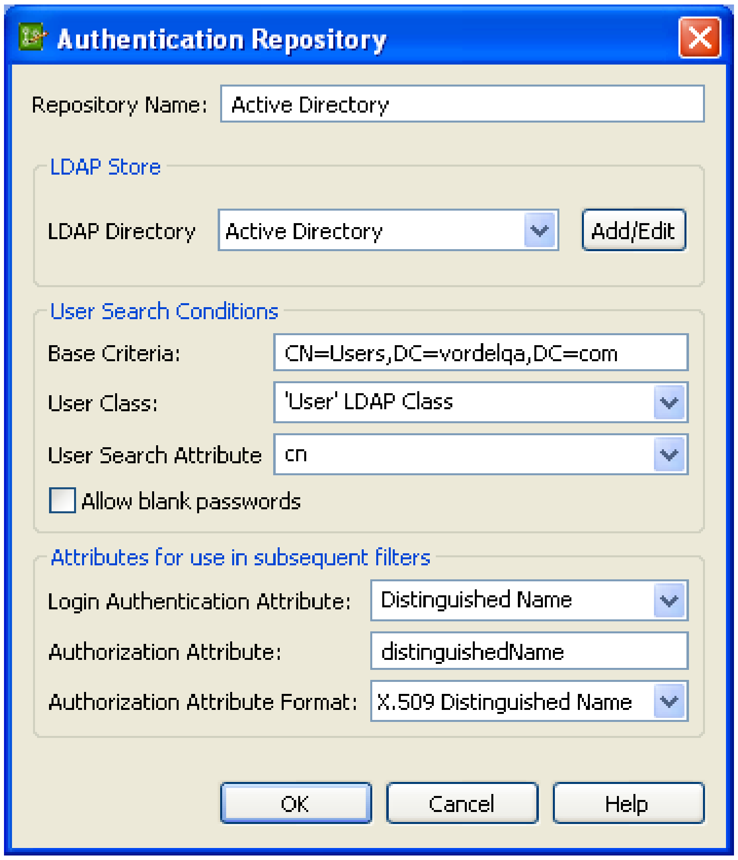
Realm:

☐ SSL Enabled

Additional JNDI properties:

Name	Value

The Authentication Repository configuration:



Repository Name: Active Directory

LDAP Store

LDAP Directory: Active Directory

User Search Conditions

Base Criteria: CN=Users,DC=vordelqa,DC=com

User Class: 'User' LDAP Class

User Search Attribute: cn

☐ Allow blank passwords

Attributes for use in subsequent filters

Login Authentication Attribute: Distinguished Name

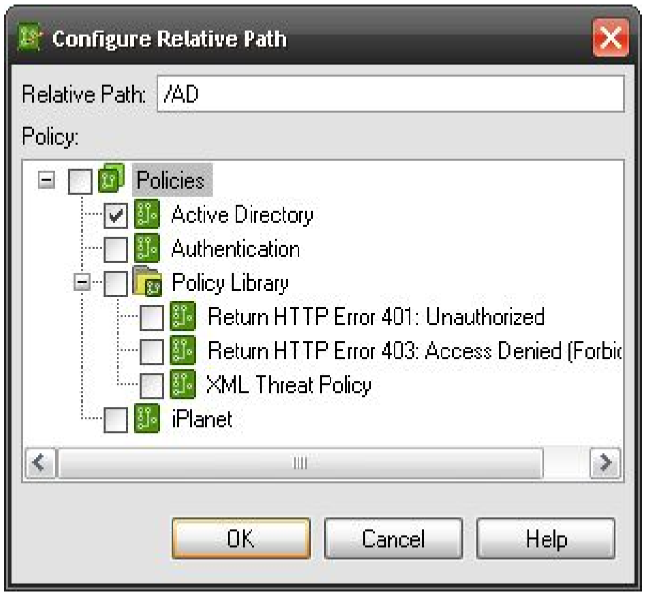
Authorization Attribute: distinguishedName

Authorization Attribute Format: X.509 Distinguished Name

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: /AD
- Map the path to the newly created policy titled "Active Directory"
- Click OK

The Add a relative path window:



Configure Relative Path

Relative Path: /AD

Policy:

- [ ] Policies
  - [✓] Active Directory
  - [ ] Authentication
  - [ ] Policy Library
    - [ ] Return HTTP Error 401: Unauthorized
    - [ ] Return HTTP Error 403: Access Denied (Forbic
    - [ ] XML Threat Policy
  - [ ] iPlanet

3.3. Ensure policies are updated on the Gateway

- Open the Policy Studio
- Click on 'Settings'
- Select 'Refresh Server' to ensure that the changes made are propagated to the live Gateway.

3.4. Test the configuration in OEG Service Explorer

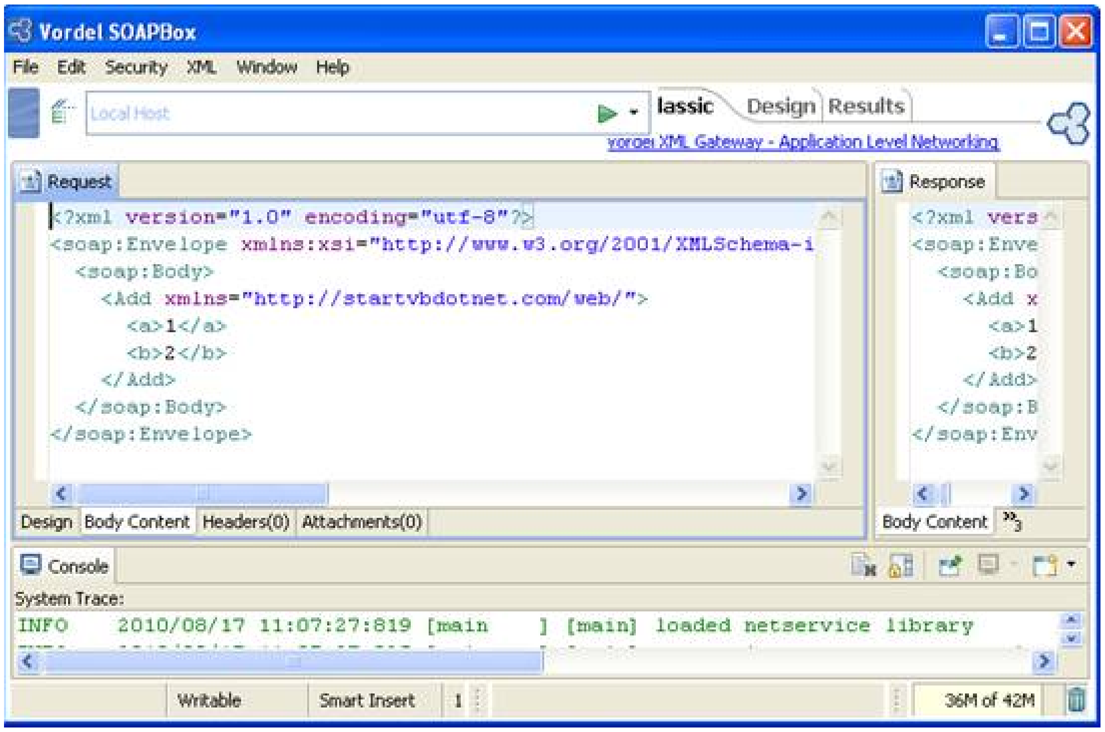
To test the policy OEG Service Explorer can be used to send through a message embedded with user credentials (Username/Password)

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/AD
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: Administrator
   - Password: vordel
10. Click on 'Run'
11. The message has been sent

The Message loaded and Connection Settings option:

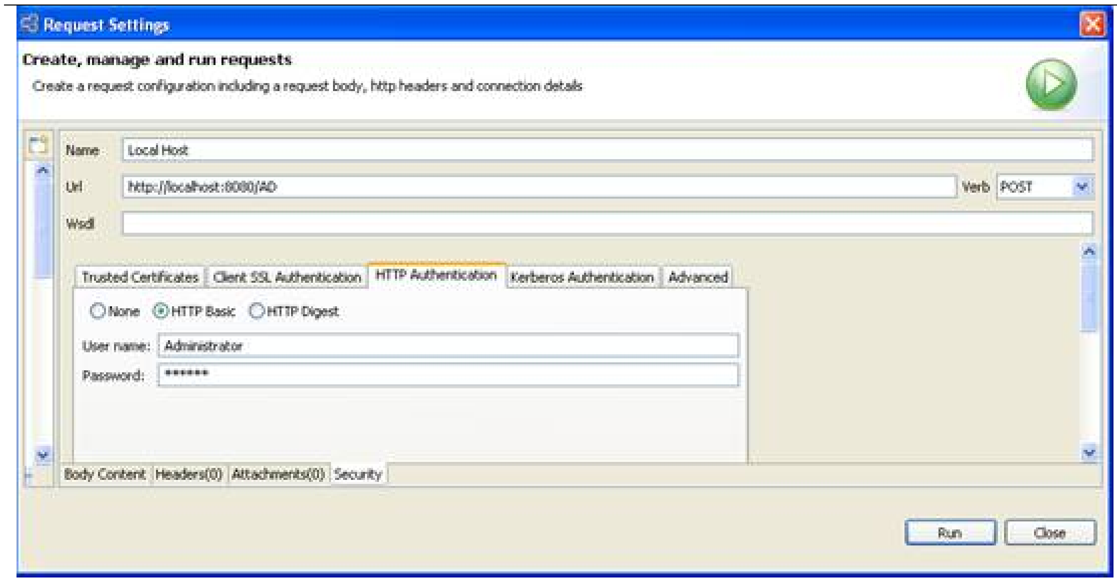

Vordel SOAPBox
File Edit Security XML Window Help
Local Host
lassic Design Results
vordel XML Gateway - Application Level Networking
Request
Response
Design Body Content Headers(0) Attachments(0)
Body Content
Console
Writable Smart Insert 1
36M of 42M

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-i
  <soap:Body>
    <Add xmlns="http://startvbdotnet.com/web/">
      <a>1</a>
      <b>2</b>
    </Add>
  </soap:Body>
</soap:Envelope>
```

System Trace:

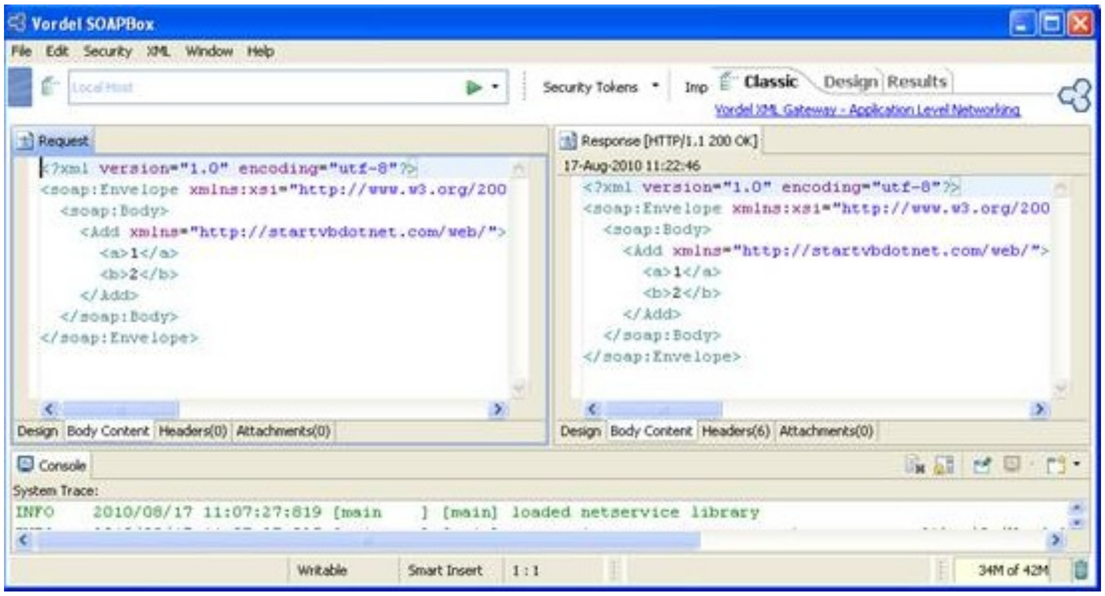
```
INFO    2010/08/17 11:07:27:819 [main    ] [main] loaded netservice library
```

The Request Settings Screen:


Request Settings
Create, manage and run requests
Create a request configuration including a request body, http headers and connection details
Trusted Certificates
Client SSL Authentication
HTTP Authentication
Kerberos Authentication
Advanced
Body Content
Headers(0)
Attachments(0)
Security
Run
Close

Name: Local Host
Url: http://localhost:8080/AD
Verb: POST
Wsdl:
None [✓ HTTP Basic] HTTP Digest
User name: Administrator
Password: ******

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



Request:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/200
  <soap:Body>
    <Add xmlns="http://startvbdotnet.com/web/">
      <a>1</a>
      <b>2</b>
    </Add>
  </soap:Body>
</soap:Envelope>
```

Response [HTTP/1.1 200 OK]
17-Aug-2010 11:22:46

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/200
  <soap:Body>
    <Add xmlns="http://startvbdotnet.com/web/">
      <a>1</a>
      <b>2</b>
    </Add>
  </soap:Body>
</soap:Envelope>
```

System Trace:

```
INFO    2010/08/17 11:07:27:819 [main     ] [main] loaded netservice library
```

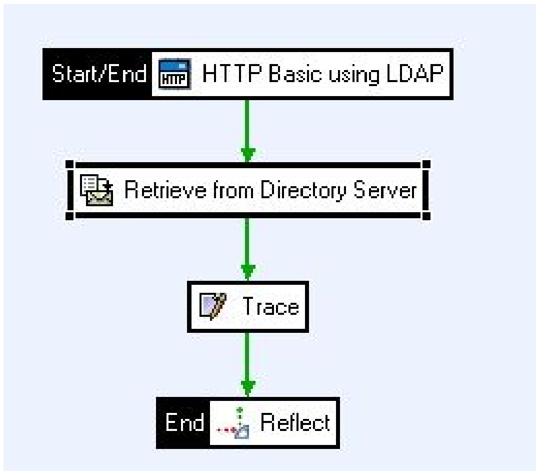
4. Adding a Retrieve from Directory Server filter

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 'Active Directory' 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 'Utility' 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: CN=Users,DC=vordelqa,DC=com
4. Search Filter: (&(objectclass=User)(cn=${authentication.subject.id}))

Retrieve Unique User Identity from LDAP Search:-

5. Base Criteria: CN=Users,DC=vordelqa,DC=com
6. Query Search Filter: (&(objectclass=User)(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: whenCreated

Value: 20040924100844.0Z

So the Attribute value added is “whenCreated”. This should return the value “20040924100844.0Z” when the messaged is being process by the Gateway.

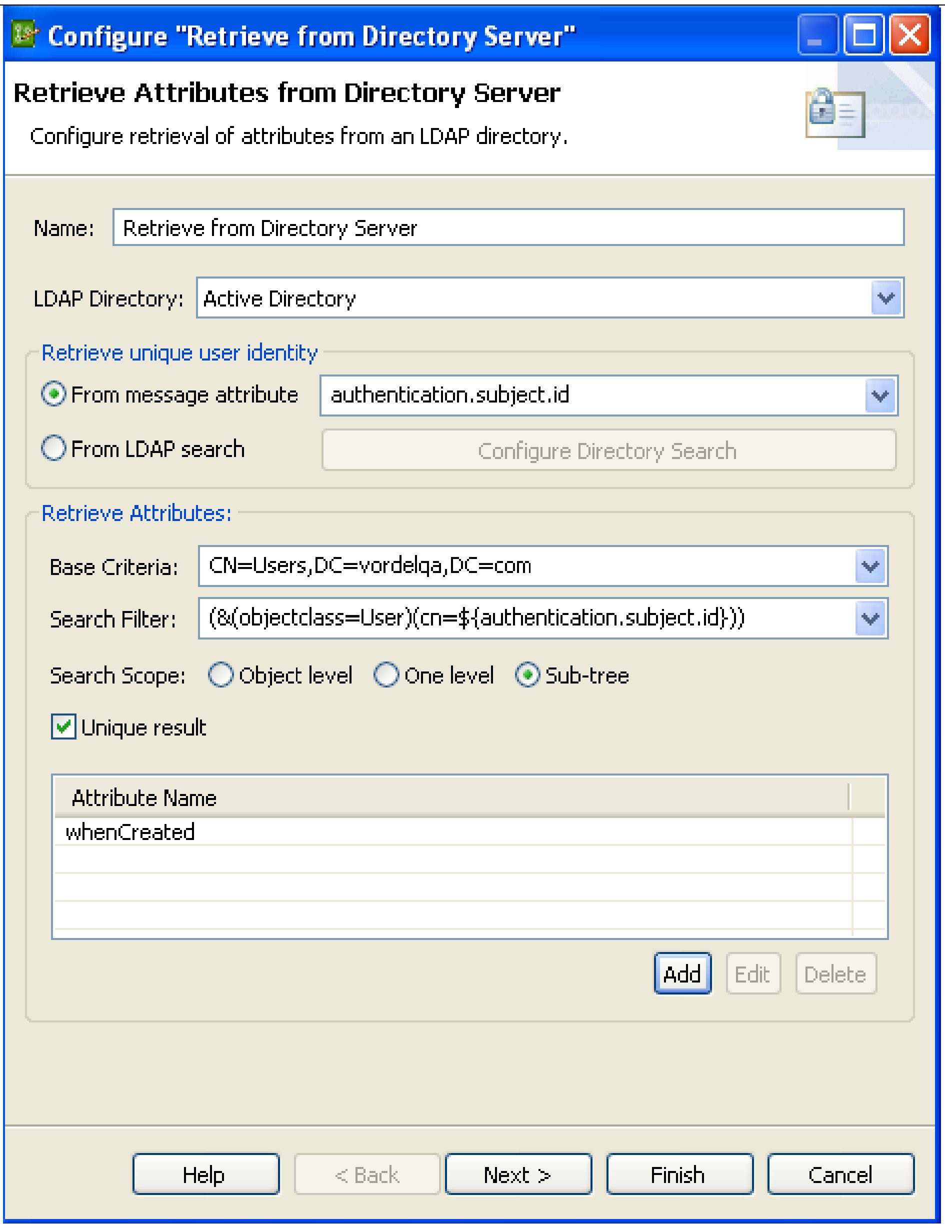
The search options above are using the base criteria of the directory structure as far down as the Common Name Object: User

The Query syntax used can also be validated by performing a search in an LDAP browser using the same string:

(&(objectclass=user)(CN=Administrator))

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

The Retrieve from Directory Server configuration:



Configure "Retrieve from Directory Server"

Retrieve Attributes from Directory Server

Configure retrieval of attributes from an LDAP directory.

Name: Retrieve from Directory Server

LDAP Directory: Active Directory

Retrieve unique user identity

- [✓ From message attribute] authentication.subject.id
- From LDAP search — Configure Directory Search

Retrieve Attributes:

Base Criteria: CN=Users,DC=vordelqa,DC=com

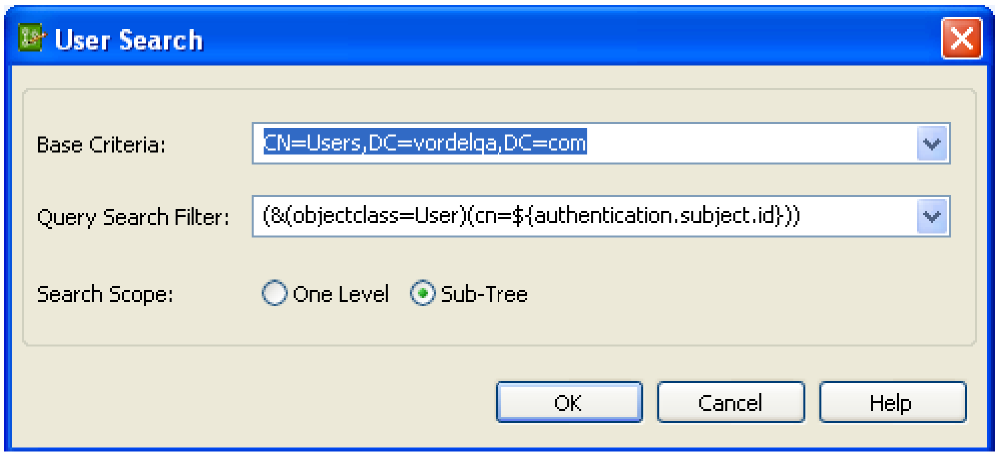
Search Filter: (&(objectclass=User)(cn=${authentication.subject.id}))

Search Scope: Object level One level [✓ Sub-tree]

[✓] Unique result

Attribute Name
whenCreated

The "Configure Directory Search" configuration screen used for the option Retrieve unique user identity From LDAP Search:



Base Criteria: CN=Users,DC=vordelqa,DC=com

Query Search Filter: (&(objectclass=User)(cn=${authentication.subject.id}))

Search Scope: One Level [✓ Sub-Tree]

4.3. Ensure policies are updated on the Gateway

- Open the Policy Studio
- Click on 'Settings'
- Select 'Refresh Server' 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/AD

5. Click on the 'HTTP Authentication' tab
6. Select 'HTTP Basic'
7. Enter the 'Username' and 'Password' of the user that will be Authenticated via LDAP
8. User credentials:
   - Username: Administrator
   - Password: vordel
9. Click on 'Run'
10. The message has been sent

With the 'Trace' Filter added it is also possible to view the attribute retrieval that occurred:

Extract from Trace above showing Attribute retrieval:

```
-------------------------------------------------------------------------------------------

DEBUG   10:56:36:644 [1528] run filter [Retrieve from Directory Server] {

DEBUG   10:56:36:644 [1528]     Searching for a user identity with base
[CN=Users,DC=vordelqa,DC=com] and filter [(&(objectclass=User)(cn=Administrator))]

DEBUG   10:56:36:644 [1528]     The ldap service provider is com.sun.jndi.ldap.LdapCtxFactory

DEBUG   10:56:36:644 [1528]     adding the additional jndi properties: {}

DEBUG   10:56:36:644 [1528]     The context is created for the LDAP lookup

DEBUG   10:56:36:660 [1528]     LdapLookup.getUserIdentity: The user's unique  identity is
[CN=Administrator,CN=Users,DC=vordelqa,DC=com]

DEBUG   10:56:36:660 [1528]     LookupHandler.process: userIdentity:
CN=Administrator,CN=Users,DC=vordelqa,DC=com

DEBUG   10:56:36:660 [1528]     Looking up user cache with the key:
CN=Administrator,CN=Users,DC=vordelqa,DC=com

DEBUG   10:56:36:660 [1528]     User's attribute from cache: (null)

DEBUG   10:56:36:660 [1528]     No attributes for user in cache so do lookup

DEBUG   10:56:36:660 [1528]     The user identity whose attributes are looked for is
[CN=Administrator,CN=Users,DC=vordelqa,DC=com]

DEBUG   10:56:36:660 [1528]     Searching for a attributes with base
[CN=Users,DC=vordelqa,DC=com] and filter [(&(objectclass=User)(cn=Administrator))]
```

```
DEBUG   10:56:36:676 [1528]     The context is created for the LDAP lookup

DEBUG   10:56:36:676 [1528]     Retrieving attributes for the result CN=Administrator

DEBUG   10:56:36:676 [1528]     LdapLookup.addToAttributeHashMap: attribute=[whenCreated]
value=[20040924100844.0Z]

DEBUG   10:56:36:676 [1528]     LdapAttrLookupHandler.getAttributes:
Attributes={whenCreated=key=[whenCreated] name=[whenCreated] values=[20040924100844.0Z]
namespace=[##nonamespace##] namespaceForAssertion=[urn:vordel:attribute:1.0]
useForAssertion=[true]}

DEBUG   10:56:36:676 [1528]     Retrieved attributes: 1===> key=[whenCreated]
name=[whenCreated] values=[20040924100844.0Z] namespace=[##nonamespace##]
namespaceForAssertion=[urn:vordel:attribute:1.0] useForAssertion=[true]


DEBUG   10:56:36:691 [1528]     Copy user attribute [whenCreated] value=[20040924100844.0Z] to
message attribute [user.whenCreated]

DEBUG   10:56:36:691 [1528] } = 1, in 47 milliseconds

DEBUG   10:56:36:691 [1528] run filter [Trace] {

DEBUG   10:56:36:691 [1528]         Trace {

DEBUG   10:56:36:691 [1528]             attribute.lookup.list {

DEBUG   10:56:36:691 [1528]                 Value:     {whenCreated=key=[whenCreated]
name=[whenCreated] values=[20040924100844.0Z] namespace=[##nonamespace##]
namespaceForAssertion=[urn:vordel:attribute:1.0] useForAssertion=[true]}

………………………………………………………..

DEBUG   10:56:36:879 [1528]             user.whenCreated {

DEBUG   10:56:36:879 [1528]                 Value:     20040924100844.0Z

DEBUG   10:56:36:879 [1528]                 Type:      java.lang.String

DEBUG   10:56:36:879 [1528]             }

DEBUG   10:56:36:879 [1528]         }

DEBUG   10:56:36:879 [1528] } = 1, in 188 milliseconds
```

As can be seen in the trace the attribute “whenCreated” has been retrieved as specified in the ‘Retrieve from Directory Server’ filter with a value of ‘20040924100844.0Z’. 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

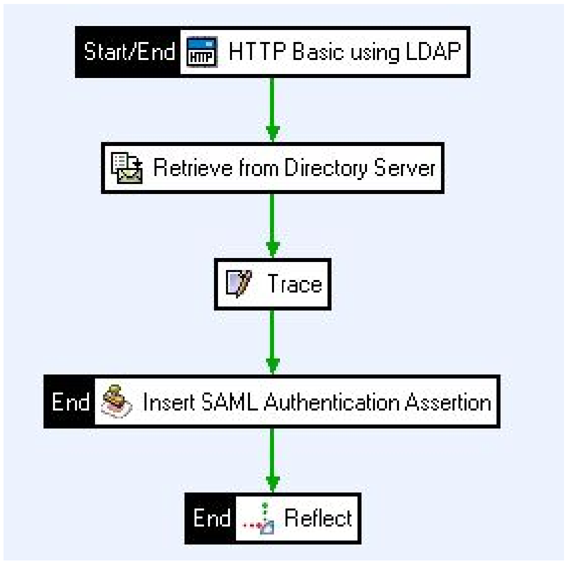
With the Basic HTTP authorization and Attribute retrieval from the directory server having been completed successfully, the policy will yet again be modified to include a SAML Assertion filter.

5.1. Add an ‘Insert SAML Authentication Assertion’ filter

Complete the following steps to refresh the policies:

1. Open Policy Studio
2. Click the ‘Active Directory’ 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:


Start/End
HTTP Basic using LDAP
Retrieve from Directory Server
Trace
End
Insert SAML Authentication Assertion
End
Reflect

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:

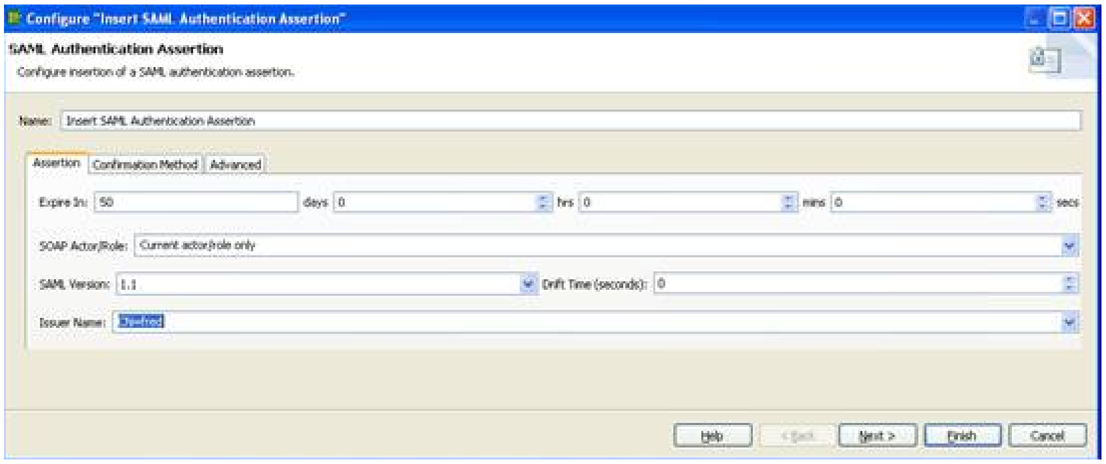
On the Assertion Tab:

1. Expire in: 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:


Configure "Insert SAML Authentication Assertion"

SAML Authentication Assertion

Configure insertion of a SAML authentication assertion.

Name: Insert SAML Authentication Assertion

Assertion | Confirmation Method | Advanced

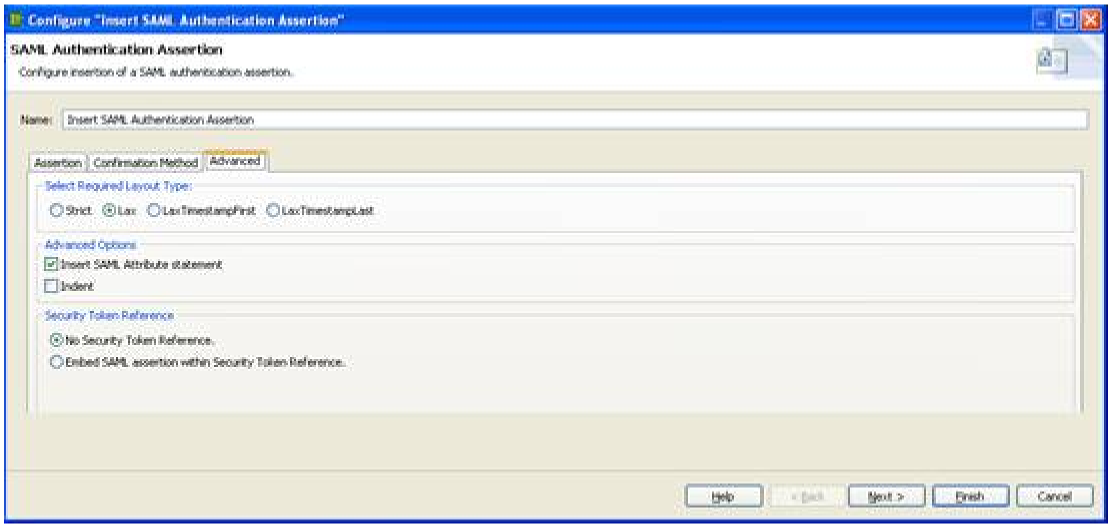
Expire In: 50 days 0 hrs 0 mins 0 secs

SOAP Actor/Role: Current actor/role only

SAML Version: 1.1 Drift Time (seconds): 0

Issuer Name: CN=fred

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


Configure "Insert SAML Authentication Assertion"

SAML Authentication Assertion

Configure insertion of a SAML authentication assertion.

Name: Insert SAML Authentication Assertion

Assertion | Confirmation Method | Advanced

Select Required Layout Type:

Strict [✓ Lax] LaxTimestampFirst LaxTimestampLast

Advanced Options

[✓ Insert SAML Attribute statement]

Indent

Security Token Reference

[✓ No Security Token Reference.]

Embed SAML assertion within Security Token Reference.

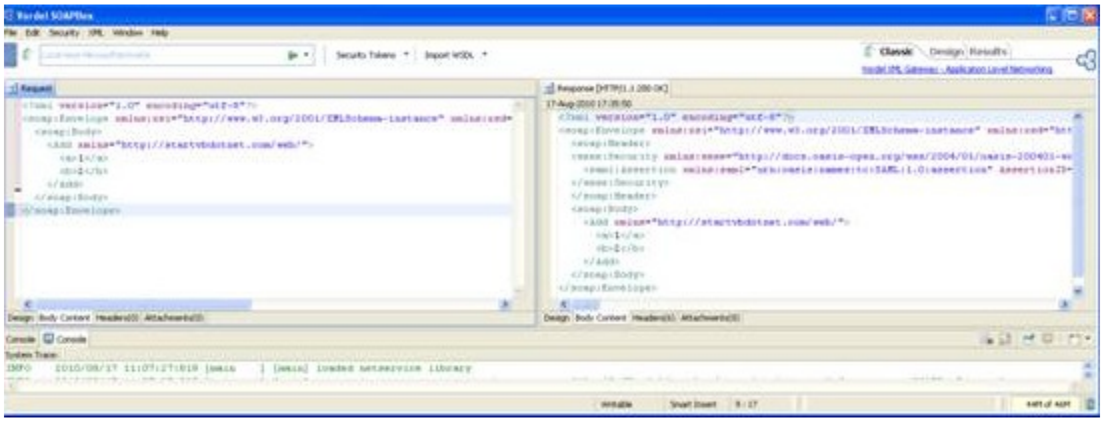
5.3. Test the configuration in OEG Service Explorer

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 '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/AD
5. Click on OK
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: Administrator
   - Password: vordel
10. Click on 'Run'
11. 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:



The successful message response in more detail:

And a snippet of the message response

```
<?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-0000012a80e82138-00000000015ff184-5"
IssueInstant="2010-08-17T16:35:50Z" Issuer="CN=AAA Certificate
Services, O=Comodo CA Limited, L=Salford, ST=Greater Manchester,
C=GB" MajorVersion="1" MinorVersion="1"><saml:Conditions
NotBefore="2010-08-17T16:35:50Z" NotOnOrAfter="2010-10-
06T16:35:50Z"/><saml:AuthenticationStatement
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
AuthenticationInstant="2010-08-
17T16:35:50Z"><saml:Subject><saml:NameIdentifier
Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">CN=Administrator,CN=Users,DC=vordelqa,DC=c
om</saml:NameIdentifier><saml:SubjectConfirmation><saml:Confirmati
onMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-
vouches</saml:ConfirmationMethod></saml:SubjectConfirmation></saml
:Subject></saml:AuthenticationStatement><saml:AttributeStatement><
saml:Subject><saml:NameIdentifier
Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName">CN=Administrator,CN=Users,DC=vordelqa,DC=c
```

```
om</saml:NameIdentifier><saml:SubjectConfirmation><saml:Confirmati
onMethod>urn:oasis:names:tc:SAML:1.0:cm:sender-
vouches</saml:ConfirmationMethod></saml:SubjectConfirmation></saml
:Subject><saml:Attribute AttributeName="whenCreated"
AttributeNamespace="urn:vordel:attribute:1.0"><saml:AttributeValue
>20040924100844.0Z</saml:AttributeValue></saml:Attribute></saml:At
tributeStatement></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 document is a simplistic demonstration on how to setup the connection and authenticate from a OEG Gateway to a LDAP directory, in this case Microsoft Active Directory.

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.

7. Appendix

Creating a secure connection using SSL to Active Directory:

The Certificate Authority that issued the LDAP Server certificate is required by Gateway and Policy Studio keystore.

Once the CA certificate is obtained it is necessary to import it into the Gateway and Policy Studio JAVA keystores.

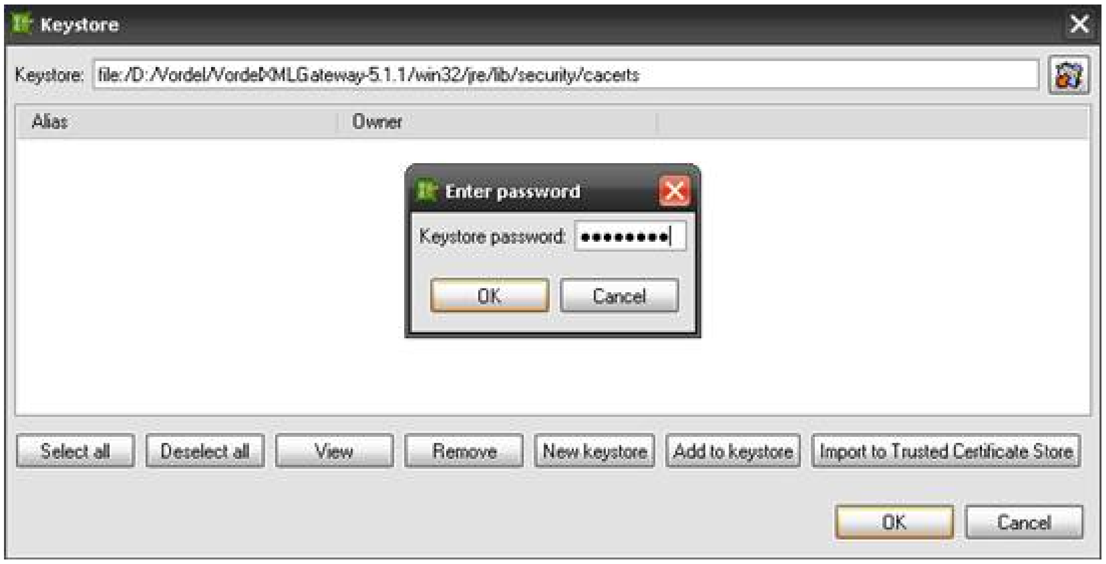
- Open Policy Studio

- Browse to 'Certificates'
- Click on 'Create' and click on 'Import Certificate'
- Browse to the LDAP Certificate and click on 'Open'
- Tick the 'Use Subject' box next to the 'Alias' field and click on 'Ok'
- The LDAP server certificate is now imported into the Gateway Certificate store
- It now needs to be added to the JAVA keystore
- Click on 'Keystore' in the 'Certificate' window
- Next to the 'Keystore' field click on the browse button
- Browse to the following file:

OEGGateway_Dir/win32/jre/lib/security/cacerts (Windows)

OEGGateway_Dir/posix/jre/lib/security/cacerts (Linux/Unix)

- Click on 'Open' and enter the Keystore password. Default password is: changeit



Keystore: file:/D:/Vordel/VordelXMLGateway-5.1.1/win32/jre/lib/security/cacerts

Keystore password: •••••••••

- Click 'Add to Keystore'
- Browse to the LDAP Certificate imported before and select it and click on 'OK'
- Refresh the Gateway by hitting the 'F6' key or select Settings, Deploy
- Now LDAP server certificate needs to be added to the Policy Studio JAVA keystore to test the LDAP connection over SSL

- Click on ‘Keystore’ in the ‘Certificate’ window
- Browse to the following file:

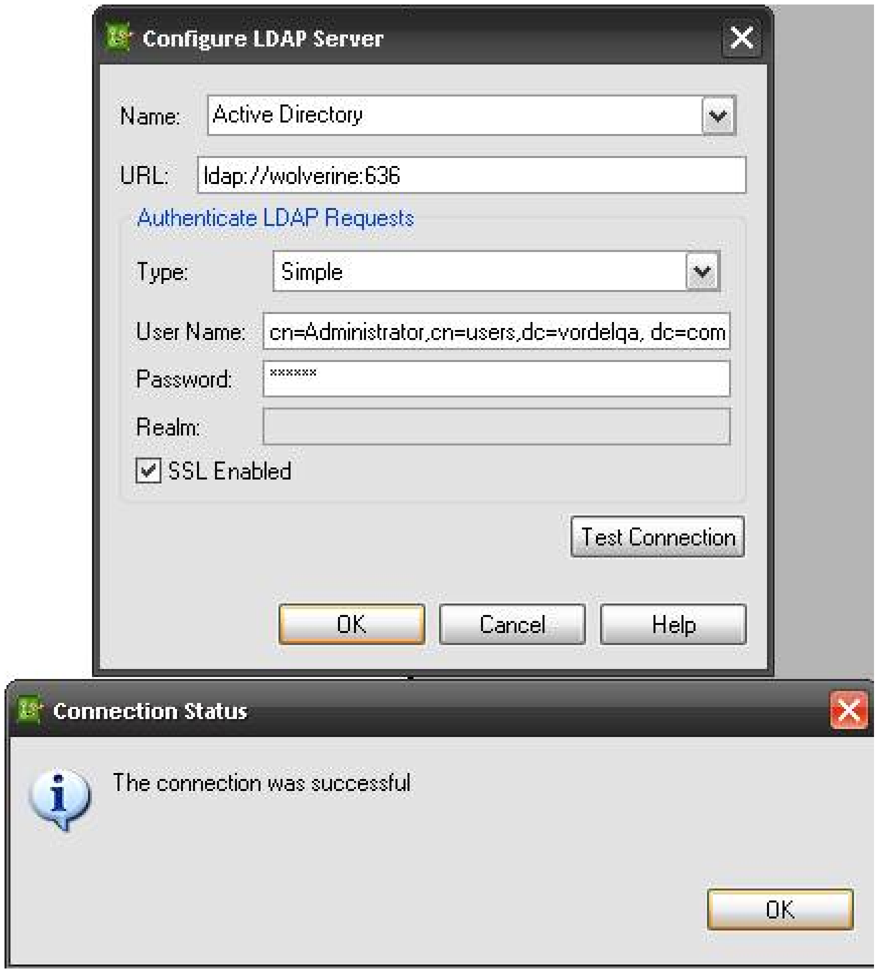
PolicyStudio_Dir/win32/jre/lib/security/cacerts (Windows)

PolicyStudio_Dir/posix/jre/lib/security/cacerts (Linux/Unix)

- Click on ‘Open’ and enter the Keystore password. Default password is: changeit
- Click ‘Add to Keystore’
- Browse to the LDAP Certificate imported before and select it and click on ‘OK’
- Restart Policy Studio

Testing the LDAP connection over SSL

- Browse to ‘External Connections’
- Expand ‘External Connection’ and ‘LDAP Connections’
- Right click on the ‘Active Directory’ connection and click on ‘Edit’
- In the URL field change the LDAP port to 636 as shown below:
- ldaps://ldap_host:636
- Tick the ‘SSL Enabled’ check box and click on ‘Test Connection’


Configure LDAP Server

Name: Active Directory

URL: ldap://wolverine:636

Authenticate LDAP Requests

Type: Simple

User Name: cn=Administrator,cn=users,dc=vordelqa, dc=com

Password: ******

Realm:

[✓] SSL Enabled

Connection Status

The connection was successful

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

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

SOFTWARE. HARDWARE. COMPLETE.