Migrating from OWSM 10gR3 Gateway to Oracle Enterprise Gateway 11.1.1

1. Introduction

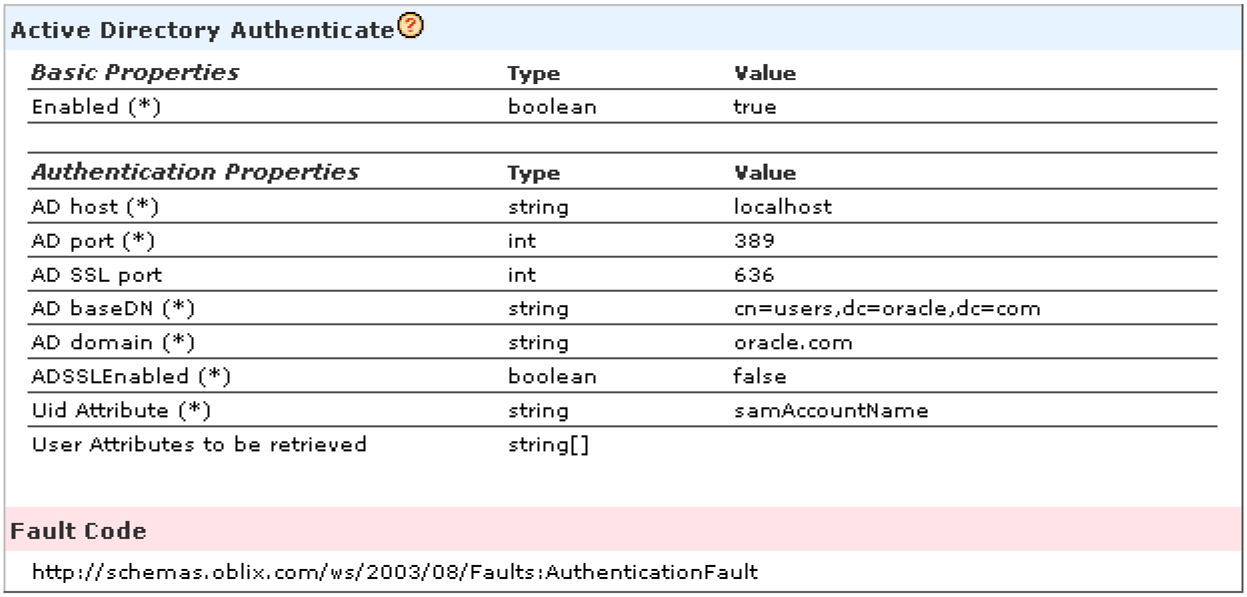
This document describes how to migrate OWSM 10gR3 Gateway configuration to Oracle Enterprise Gateway 11.configuration.

Mapping of OWSM to Oracle Enterprise Gateway filters

This section shows the mapping from steps available in OWSM to the equivalent filter / policy settings in the Oracle Enterprise Gateway.

Step: Active Directory Authenticate

OWSM 10gR3 contains a step called "Active Directory Authenticate" where by credentials are authenticate against Active Directory, the default settings for this is as follows:



Active Directory Authenticate

Basic Properties	Type	Value
Enabled (*)	boolean	true

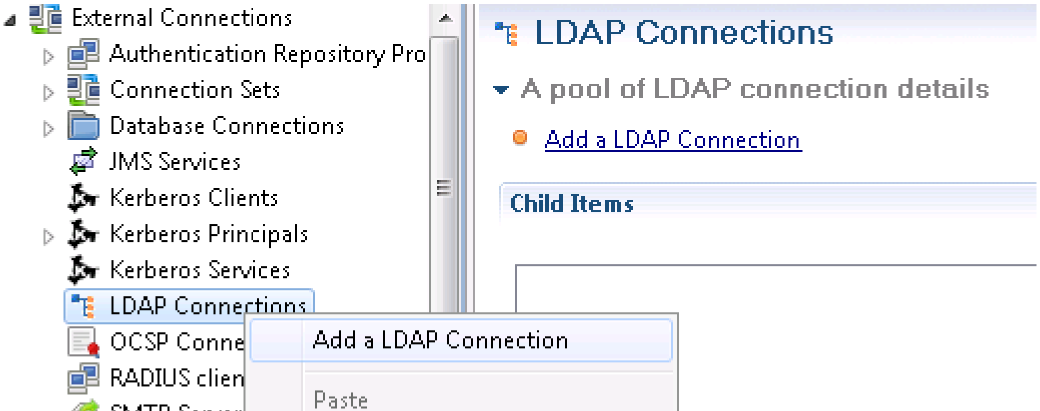
Authentication Properties	Type	Value
AD host (*)	string	localhost
AD port (*)	int	389
AD SSL port	int	636
AD baseDN (*)	string	cn=users,dc=oracle,dc=com
AD domain (*)	string	oracle.com
ADSSLEnabled (*)	boolean	false
Uid Attribute (*)	string	samAccountName
User Attributes to be retrieved	string[]	

Fault Code

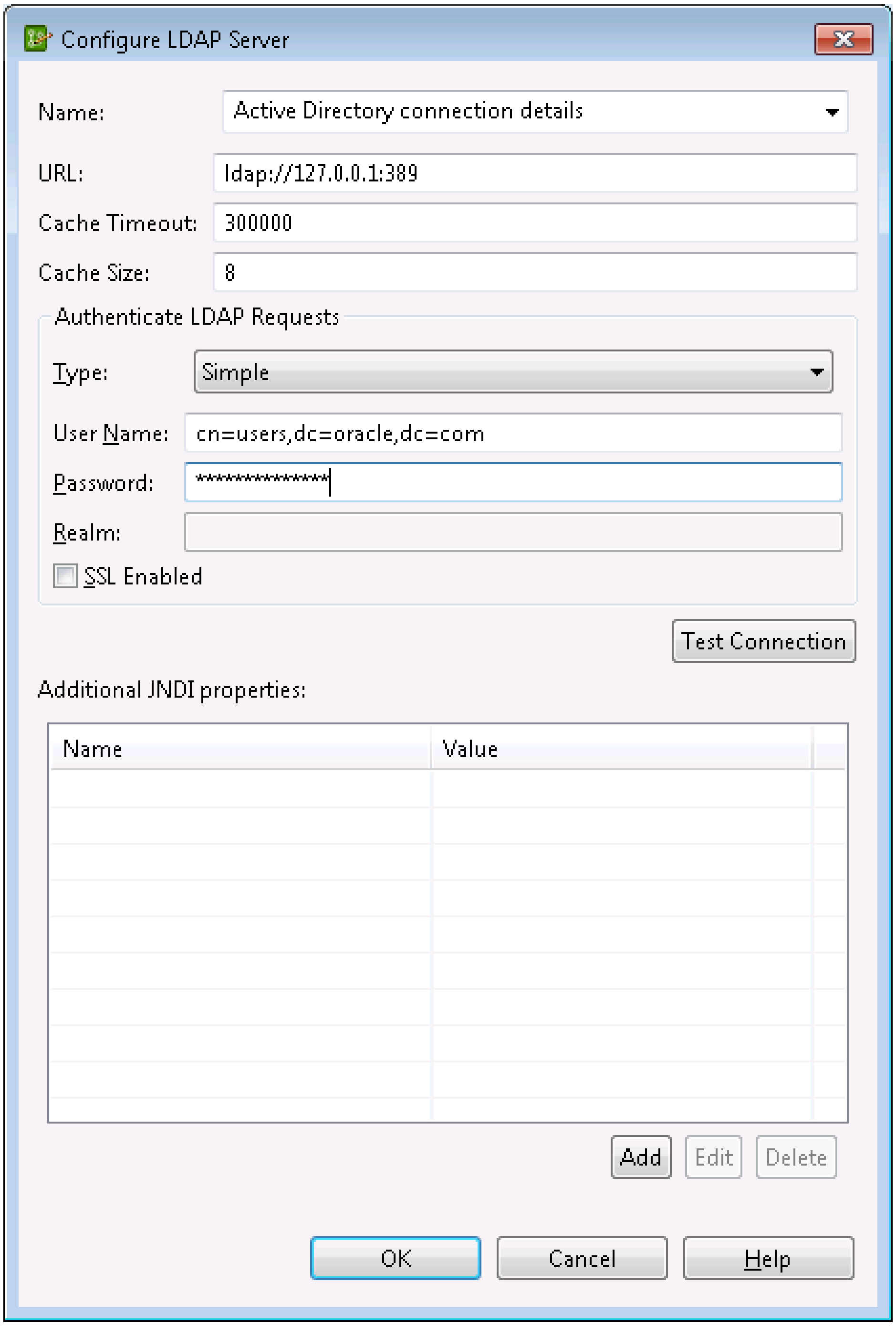
http://schemas.oblix.com/ws/2003/08/Faults:AuthenticationFault

Oracle Enterprise Gateway provides the same functionality by following these steps:

- External connections > LDAP Connections > Add a LDAP Connection
-



Enter configuration details for the LDAP connection



Configure LDAP Server

Name: Active Directory connection details

URL: ldap://127.0.0.1:389

Cache Timeout: 300000

Cache Size: 8

Authenticate LDAP Requests

Type: Simple

User Name: cn=users,dc=oracle,dc=com

Password: ***************

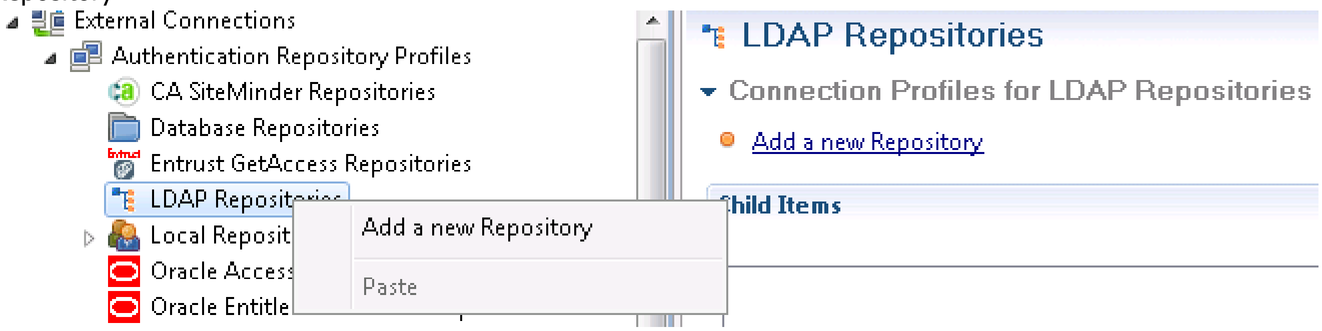
Realm:

□ SSL Enabled

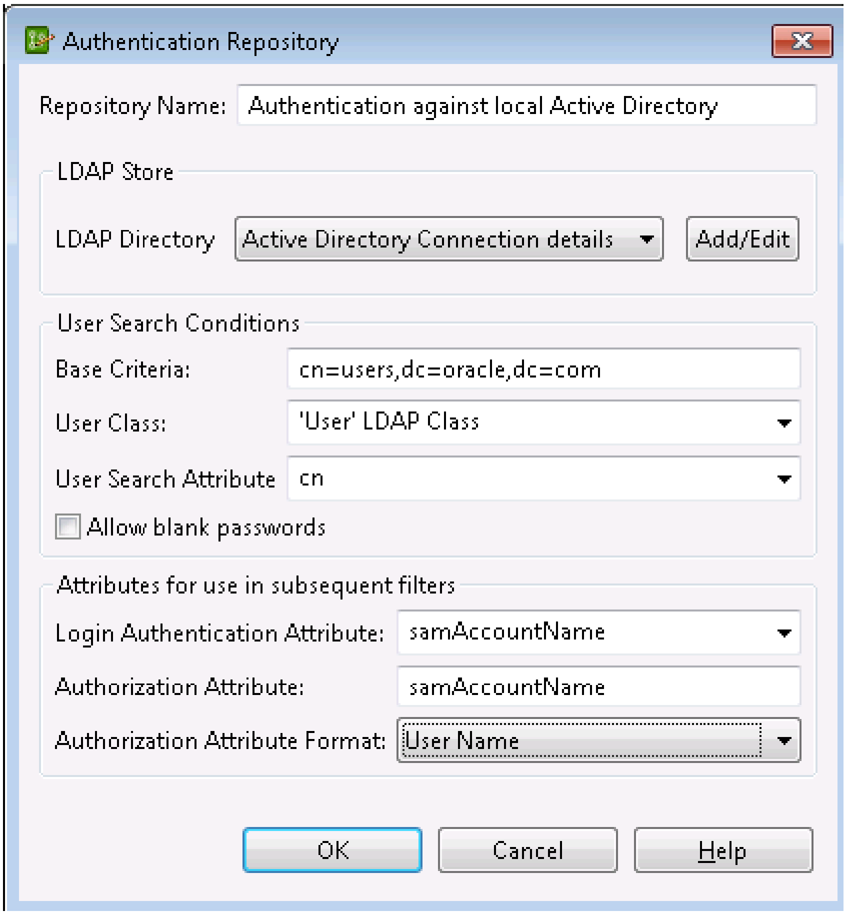
Additional JNDI properties:

Name	Value

- Create an LDAP authentication repository which then can be used in HTTP Basic, Digest, and WS-Security username password filters for authentication.
- External connections > Authentication Repository Profiles > LDAP Repositories > Add a new Repository



- Configure the repository as required:



Authentication Repository

Repository Name: Authentication against local Active Directory

LDAP Store

LDAP Directory: Active Directory Connection details

User Search Conditions

Base Criteria: cn=users,dc=oracle,dc=com

User Class: 'User' LDAP Class

User Search Attribute: cn

☐ Allow blank passwords

Attributes for use in subsequent filters

Login Authentication Attribute: samAccountName

Authorization Attribute: samAccountName

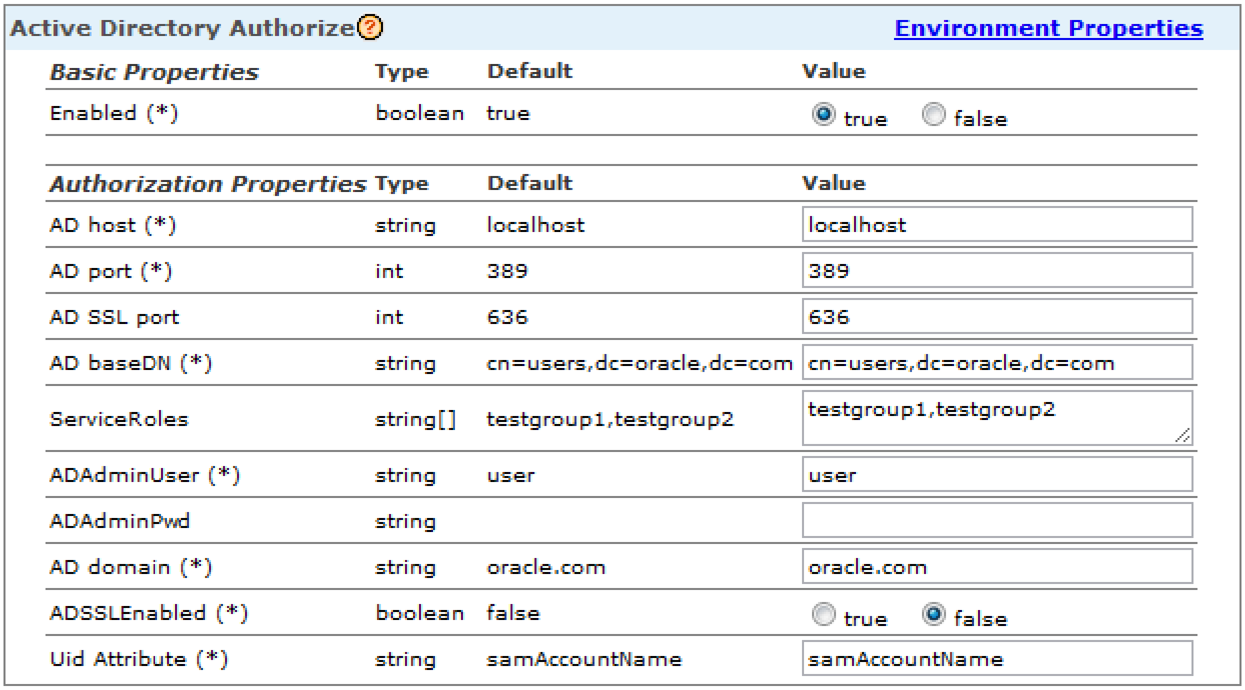
Authorization Attribute Format: User Name

- In a filter that requires authentication against Active Directory, simply select the repository named above.

Step: Active Directory Authorize

Authorizes request by retrieving roles from Active Directory and checking against roles allowed by service.

OWSM Configuration:



Active Directory Authorize

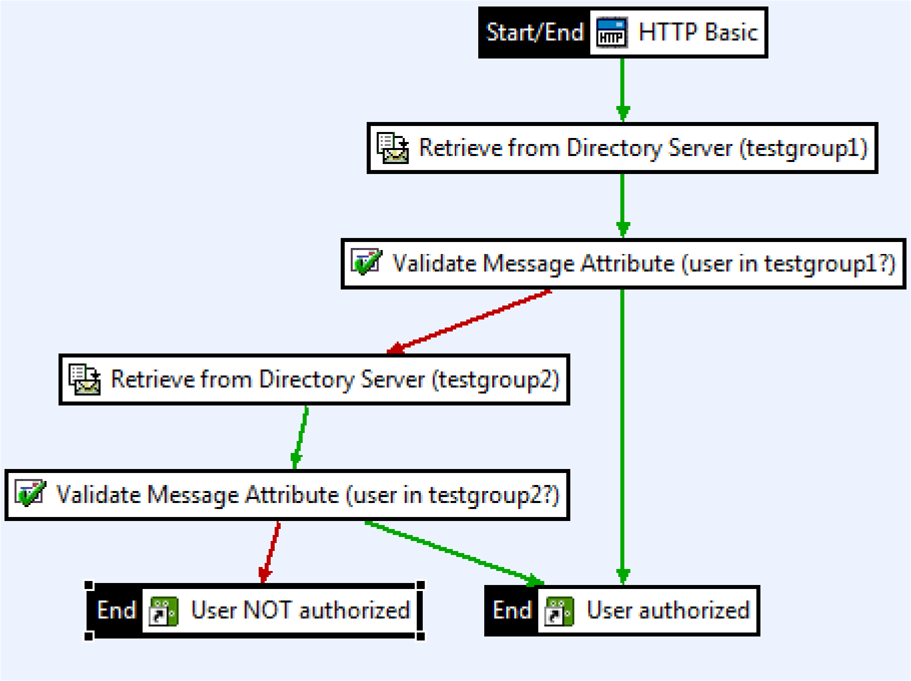
Environment Properties

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

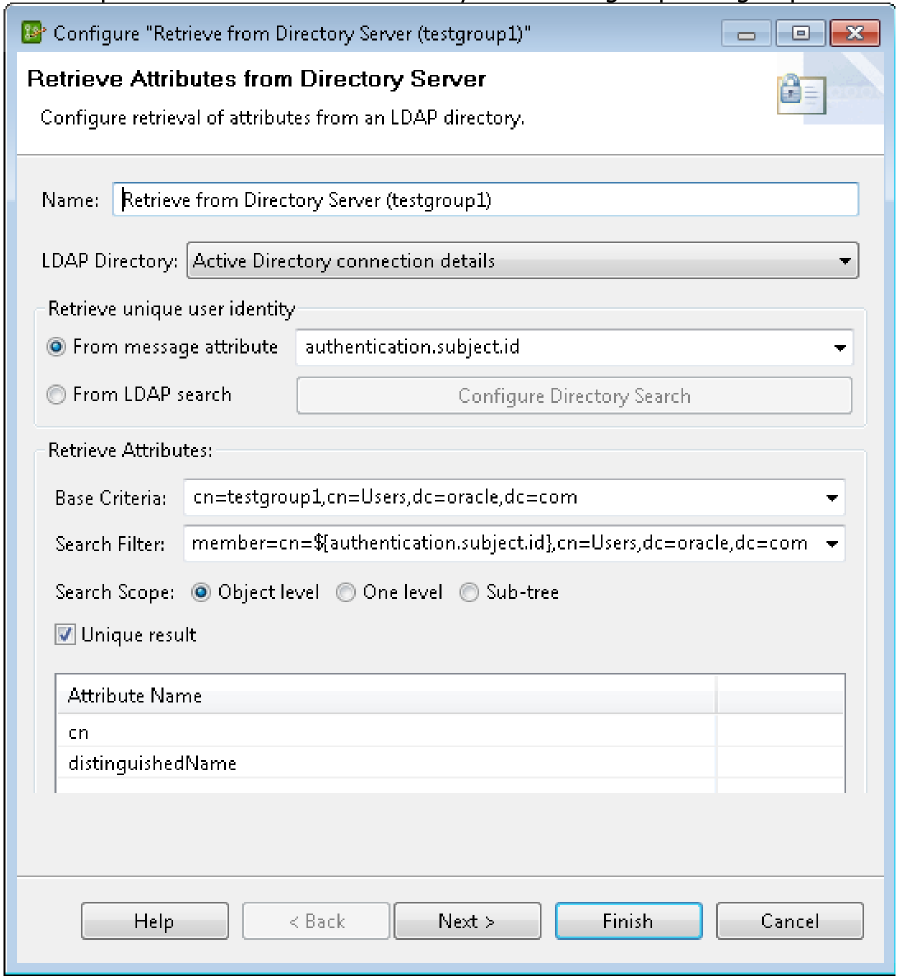
Authorization Properties	Type	Default	Value
AD host (*)	string	localhost	localhost
AD port (*)	int	389	389
AD SSL port	int	636	636
AD baseDN (*)	string	cn=users,dc=oracle,dc=com	cn=users,dc=oracle,dc=com
ServiceRoles	string[]	testgroup1,testgroup2	testgroup1,testgroup2
ADAdminUser (*)	string	user	user
ADAdminPwd	string		
AD domain (*)	string	oracle.com	oracle.com
ADSSLEnabled (*)	boolean	false	○ true ● false
Uid Attribute (*)	string	samAccountName	samAccountName

Oracle Enterprise Gateway Configuration:

1. Use LDAP connection and repository setup as detailed in **Active Directory Authenticate** above.
2. After successful Authentication with HTTP Basic, Digest or WS-Security username password filters send circuit to a **"Retrieve Attribute from Directory Server"** filter followed by **"Validate Message Attribute"** filter. This combination can be used to check a group for a user's membership.
3. Repeat for each group that needs checking
e.g.


Start/End HTTP Basic
Retrieve from Directory Server (testgroup1)
Validate Message Attribute (user in testgroup1?)
Retrieve from Directory Server (testgroup2)
Validate Message Attribute (user in testgroup2?)
End User NOT authorized
End User authorized

4. Example of a Retrieve from Directory Server for group “testgroup1”



Configure "Retrieve from Directory Server (testgroup1)"

Retrieve Attributes from Directory Server

Configure retrieval of attributes from an LDAP directory.

Name: Retrieve from Directory Server (testgroup1)

LDAP Directory: Active Directory connection details

Retrieve unique user identity

● From message attribute: authentication.subject.id

○ From LDAP search

Retrieve Attributes:

Base Criteria: cn=testgroup1,cn=Users,dc=oracle,dc=com

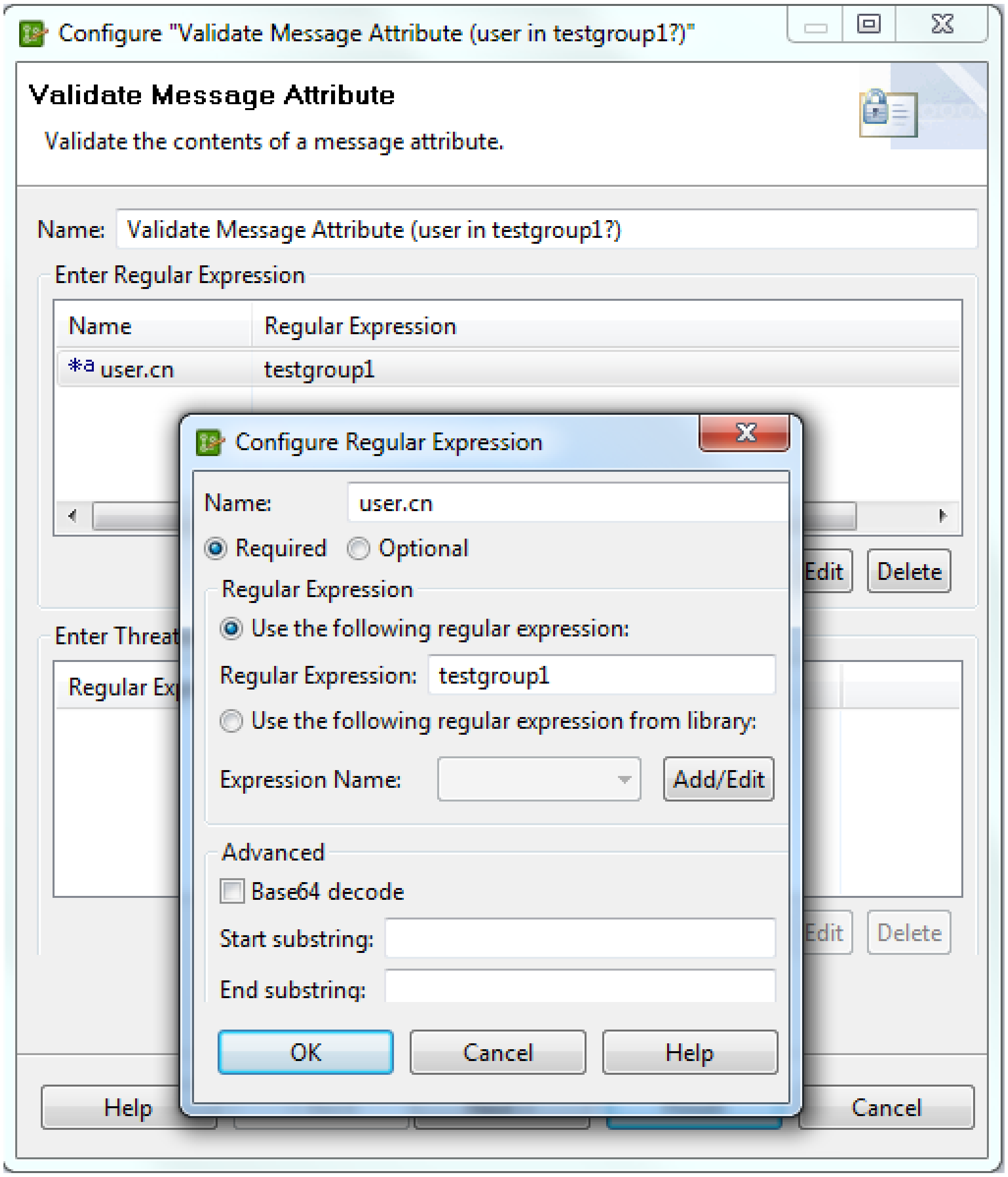
Search Filter: member=cn=${authentication.subject.id},cn=Users,dc=oracle,dc=com

Search Scope: ● Object level ○ One level ○ Sub-tree

☑ Unique result

Attribute Name
cn
distinguishedName

5. Example of a **Validate Message Attribute** for group “testgroup1”
This filter returns success if the user is found to be a a member of testgroup1. The authorized user is forwarded to the next group check and finally to the user authorized policy which is shown as a shortcut in the circuit above.



Configure "Validate Message Attribute (user in testgroup1?)"

Validate Message Attribute

Validate the contents of a message attribute.

Name: Validate Message Attribute (user in testgroup1?)

Enter Regular Expression

Name	Regular Expression
*a user.cn	testgroup1

Enter Threat

Regular Exp

Configure Regular Expression

Name: user.cn

● Required ○ Optional

Regular Expression

● Use the following regular expression:

Regular Expression: testgroup1

○ Use the following regular expression from library:

Expression Name:

Advanced

□ Base64 decode

Start substring:

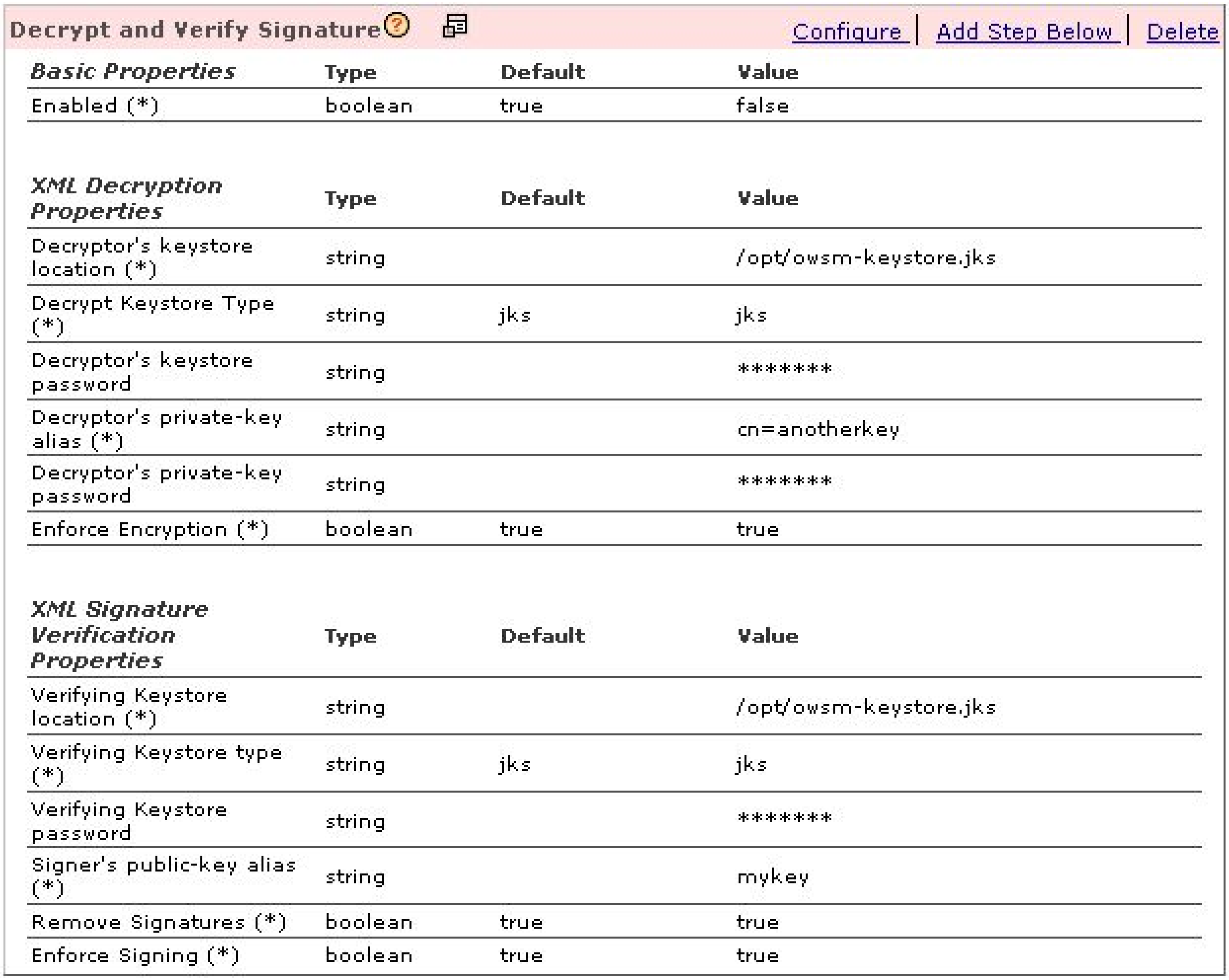
End substring:

6. Similarly the user not authorized branch is a shortcut. This could be to a policy that eventually reflects an appropriate failure message back to the client.

Step: Decrypt and Verify Signature

XML Decryption And Signature Verification

OWSM Configuration:



Decrypt and Verify Signature

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	false

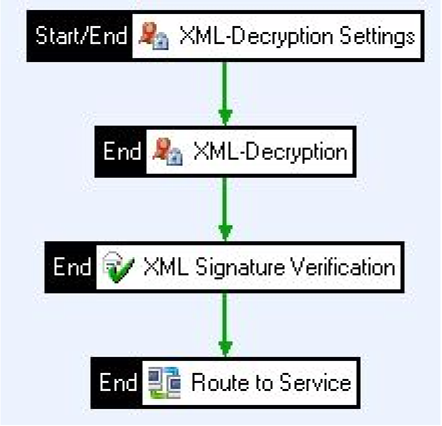
XML Decryption Properties	Type	Default	Value
Decryptor's keystore location (*)	string		/opt/owsm-keystore.jks
Decrypt Keystore Type (*)	string	jks	jks
Decryptor's keystore password	string		*******
Decryptor's private-key alias (*)	string		cn=anotherkey
Decryptor's private-key password	string		*******
Enforce Encryption (*)	boolean	true	true

XML Signature Verification Properties	Type	Default	Value
Verifying Keystore location (*)	string		/opt/owsm-keystore.jks
Verifying Keystore type (*)	string	jks	jks
Verifying Keystore password	string		*******
Signer's public-key alias (*)	string		mykey
Remove Signatures (*)	boolean	true	true
Enforce Signing (*)	boolean	true	true

Oracle Enterprise Gateway Configuration:
To decrypt and verify a request in the Gateway perform the XML Decryption and Verify Signature steps in one policy. Please refer to the XML Decryption and Verify Signature the sections for configuration of the relevant options.

NOTE: Ensure that the relevant certificate/key is loaded into the Gateway Certificate store and that the correct certificate is selected under Decryption and Signature Verification filters.

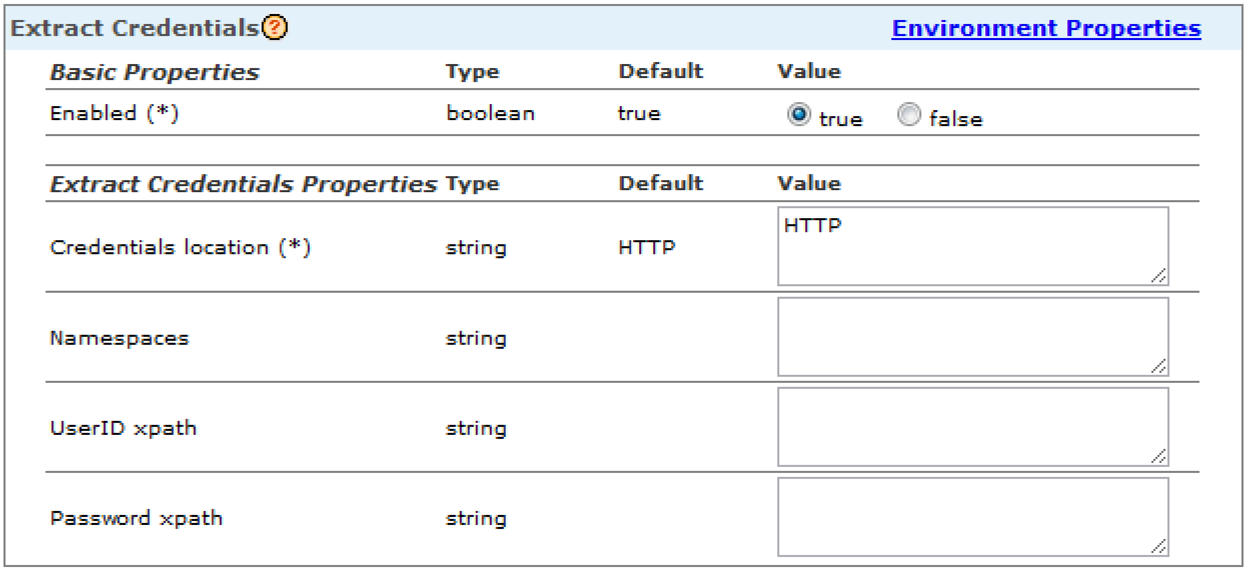
A sample policy where the request is decrypted and the signature verified:


Start/End XML-Decryption Settings
End XML-Decryption
End XML Signature Verification
End Route to Service

Step: Extract Credentials

Locates and extracts credentials and presents the credentials in a form that they can be authenticated. You must know from where the credentials are to be extracted.

OWSM Configuration:



Extract Credentials ? — Environment Properties

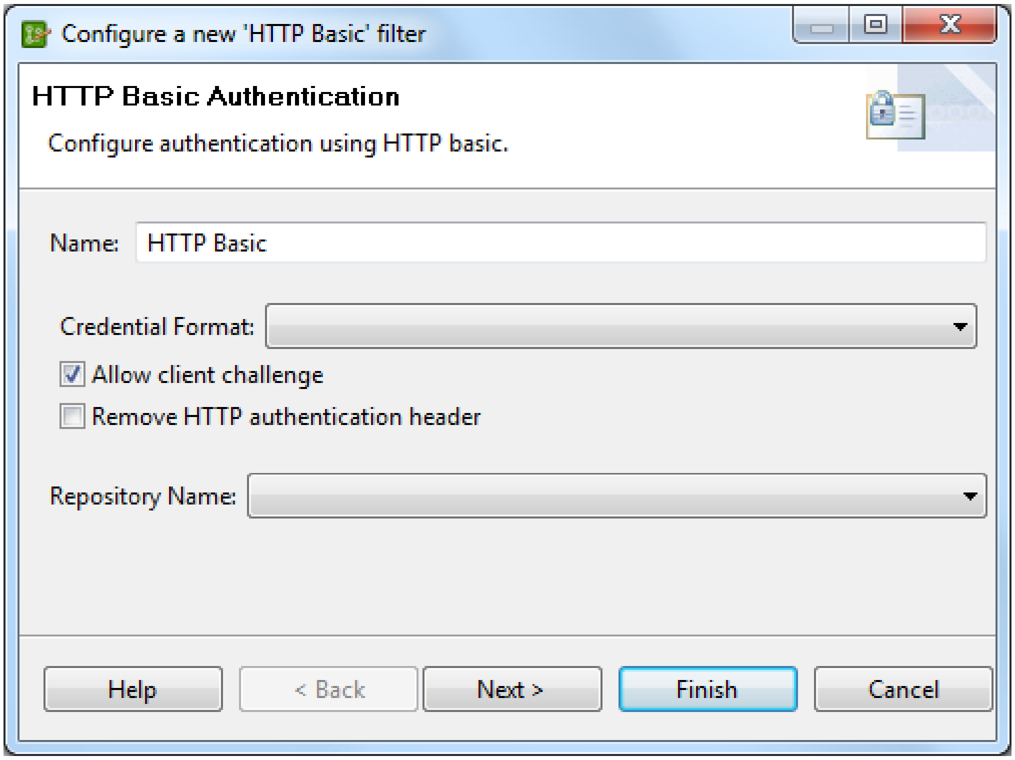
Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

Extract Credentials Properties	Type	Default	Value
Credentials location (*)	string	HTTP	HTTP
Namespaces	string		
UserID xpath	string		
Password xpath	string		

Credential location allowed values are HTTP, JMS, WS-BASIC or XPath

Oracle Enterprise Gateway Configuration:

HTTP Basic - filter to check against a previously created repository name. e.g. LDAP, Active Directory, Oracle Access Manager



Configure a new 'HTTP Basic' filter

HTTP Basic Authentication

Configure authentication using HTTP basic.

Name: HTTP Basic

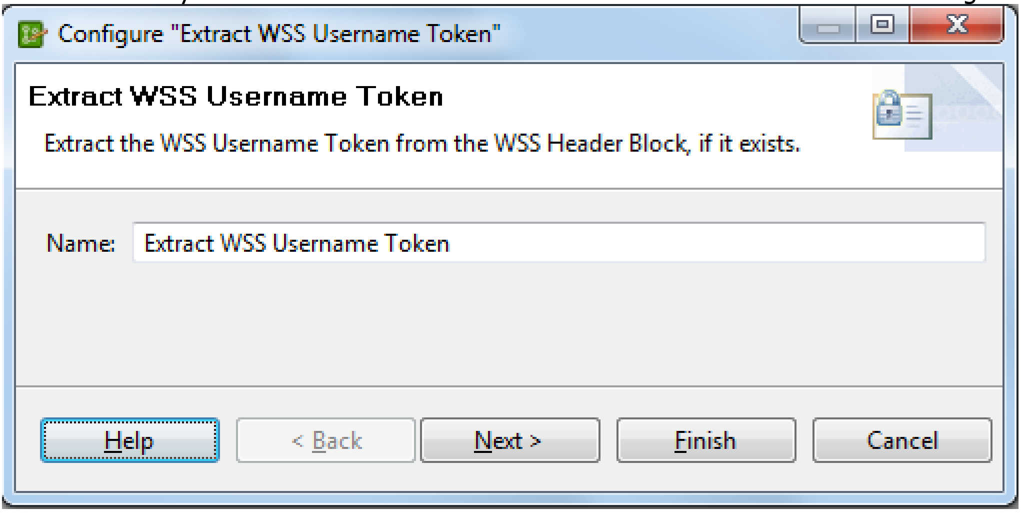
Credential Format:

☑ Allow client challenge

□ Remove HTTP authentication header

Repository Name:

WS-BASIC - WS-Security UsernameToken will be stored in the wss.usernameToken message attribute.



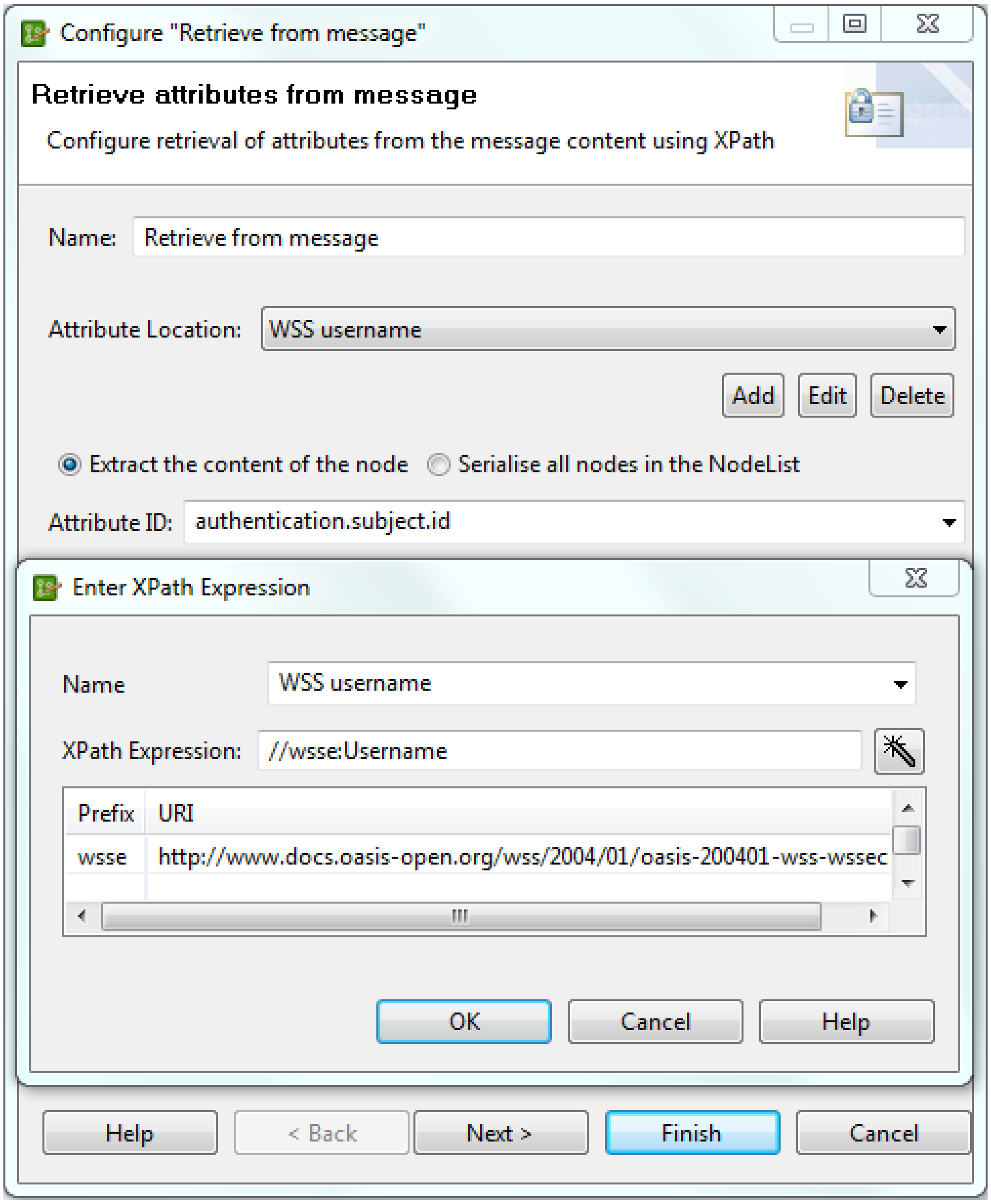
Configure "Extract WSS Username Token"

Extract WSS Username Token

Extract the WSS Username Token from the WSS Header Block, if it exists.

Name: Extract WSS Username Token

XPath e.g. retrieve WS-Security username and password then store in message attributes authentication.subject.id and authentication.subject.password
Within the policy drag a "Retrieve from message" filter from the Attributes category.



Configure "Retrieve from message"

Retrieve attributes from message

Configure retrieval of attributes from the message content using XPath

Name: Retrieve from message

Attribute Location: WSS username

● Extract the content of the node ○ Serialise all nodes in the NodeList

Attribute ID: authentication.subject.id

Enter XPath Expression

Name: WSS username

XPath Expression: //wsse:Username

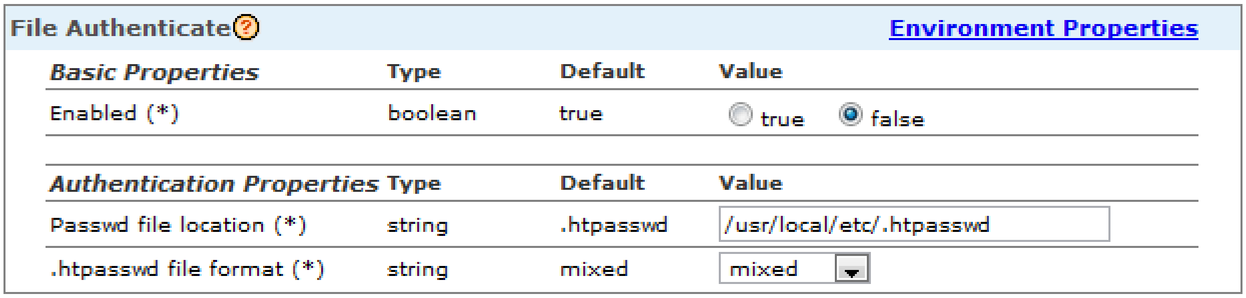
Prefix	URI
wsse	http://www.docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssec

Then do the same for wsse:Password

File Authenticate

Step Name: Authenticate username and password against a local .htpasswd file. This step depends on Extract Credentials Step

OWSM Configuration:



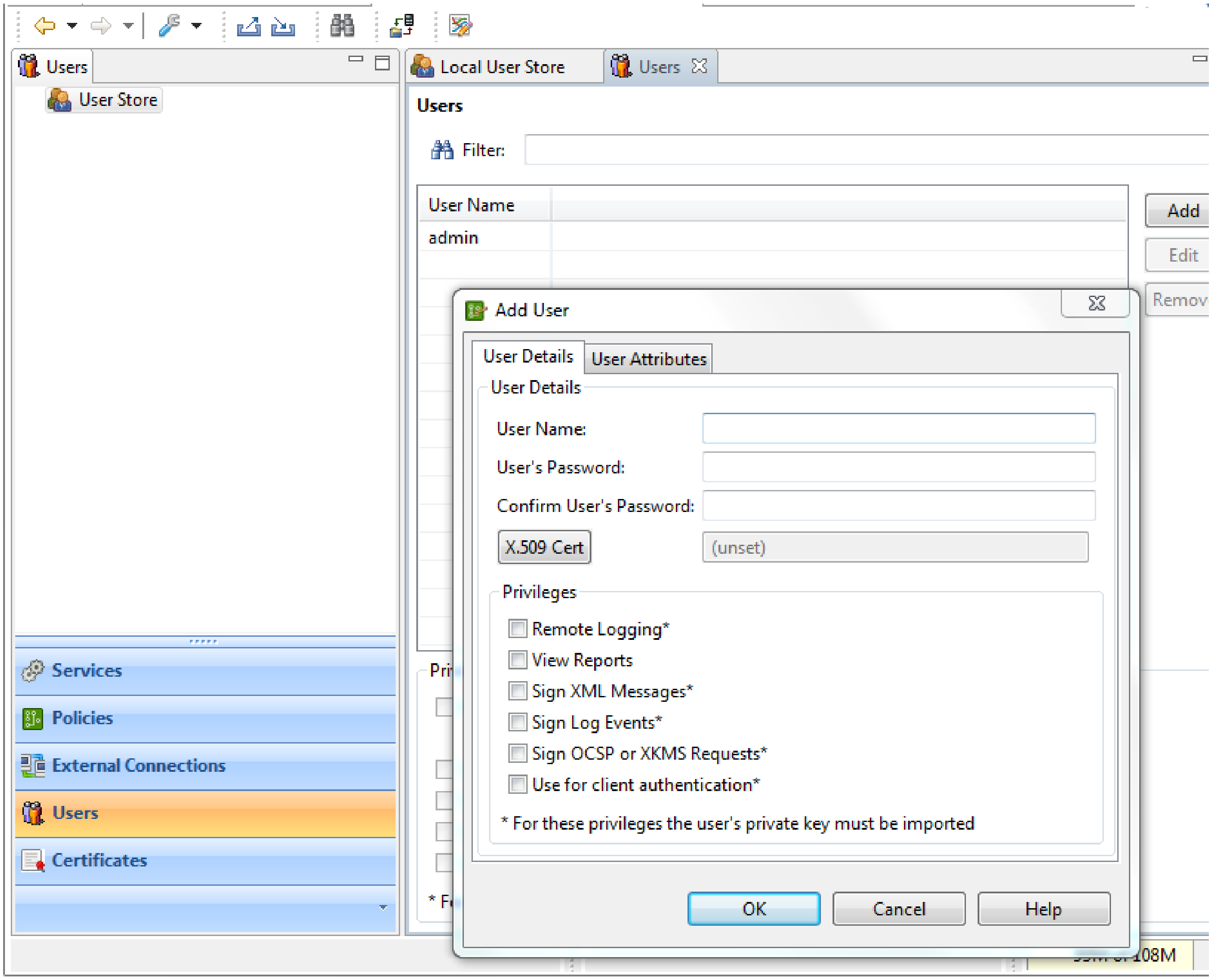
File Authenticate ? **Environment Properties**

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	○ true ● false

Authentication Properties	Type	Default	Value
Passwd file location (*)	string	.htpasswd	/usr/local/etc/.htpasswd
.htpasswd file format (*)	string	mixed	mixed

Oracle Enterprise Gateway Configuration:
Oracle Enterprise Gateway is not able to use a htpasswd format file like OWSM 10gR3 but the Local User Store would offer equivalent functionality and more. User credentials are securely stored in the Local User Store for authentication with a filter for example HTTP Basic.

1. To add a user to the Local User Store



Add User

User Details | User Attributes

User Details

User Name:

User's Password:

Confirm User's Password:

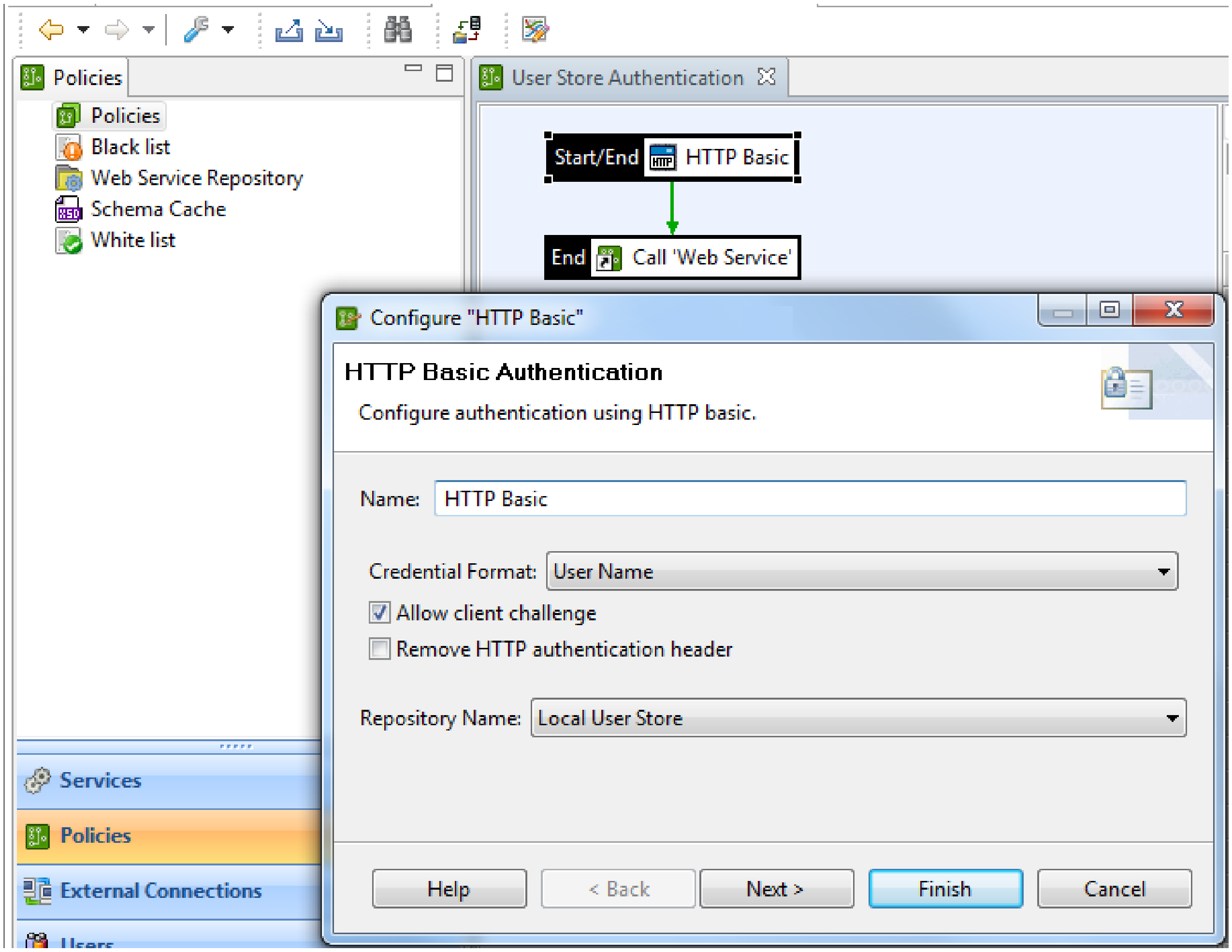
X.509 Cert (unset)

Privileges

□ Remote Logging*
□ View Reports
□ Sign XML Messages*
□ Sign Log Events*
□ Sign OCSP or XKMS Requests*
□ Use for client authentication*

* For these privileges the user's private key must be imported

2. Then use an authentication filter against the users in the store. e.g A circuit to use HTTP Basic authentication.



Configure "HTTP Basic"

HTTP Basic Authentication

Configure authentication using HTTP basic.

Name: HTTP Basic

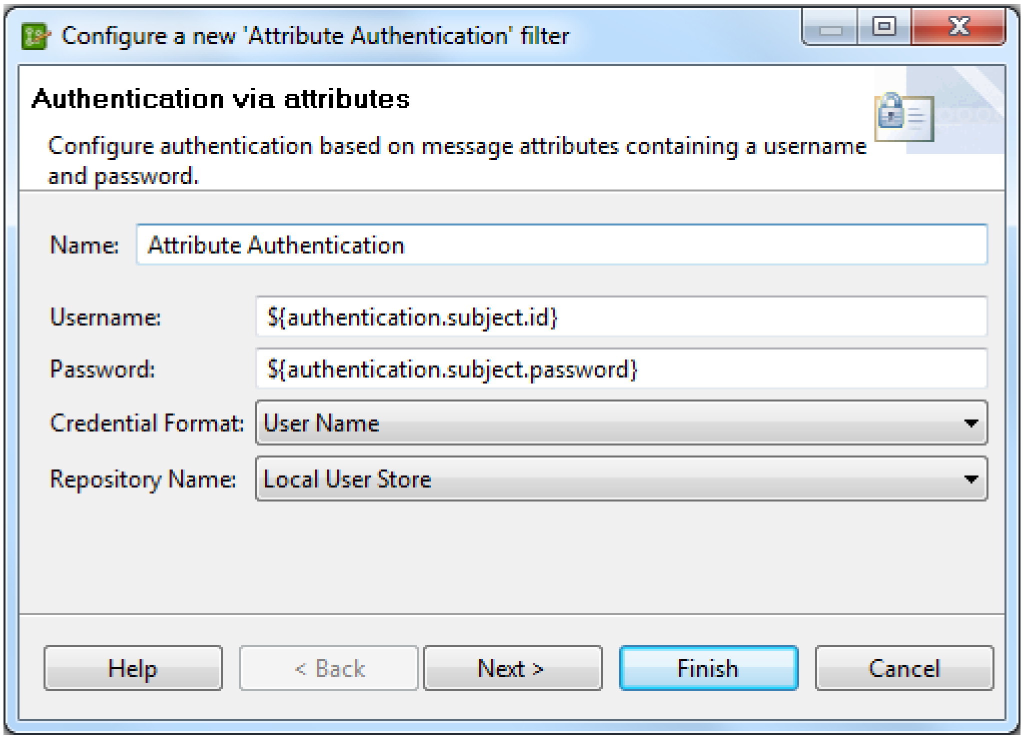
Credential Format: User Name

☑ Allow client challenge

□ Remove HTTP authentication header

Repository Name: Local User Store

3. The Attributes Authentication filter can be used to authenticate any message attributes in the Local User Store.



Configure a new 'Attribute Authentication' filter

Authentication via attributes

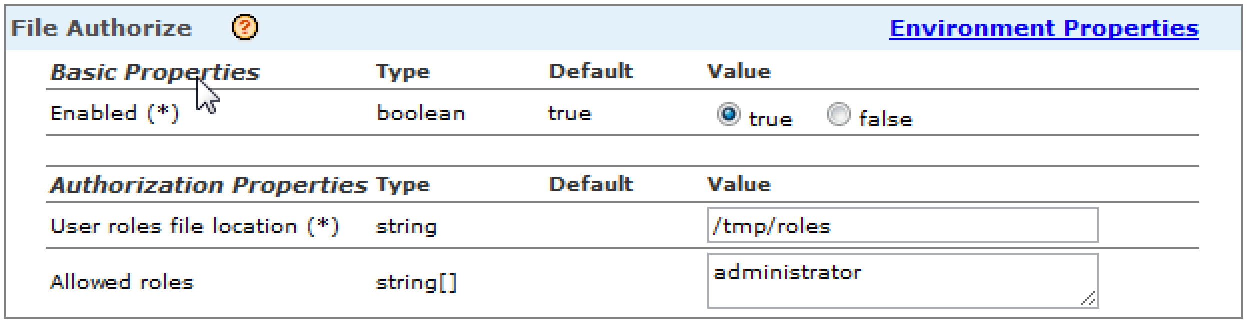
Configure authentication based on message attributes containing a username and password.

Name:	Attribute Authentication
Username:	${authentication.subject.id}
Password:	${authentication.subject.password}
Credential Format:	User Name
Repository Name:	Local User Store

Step: File Authorize

Authorize remote user against a local roles file. This step depends on Extract Credentials Step

OWSM Configuration:



File Authorize

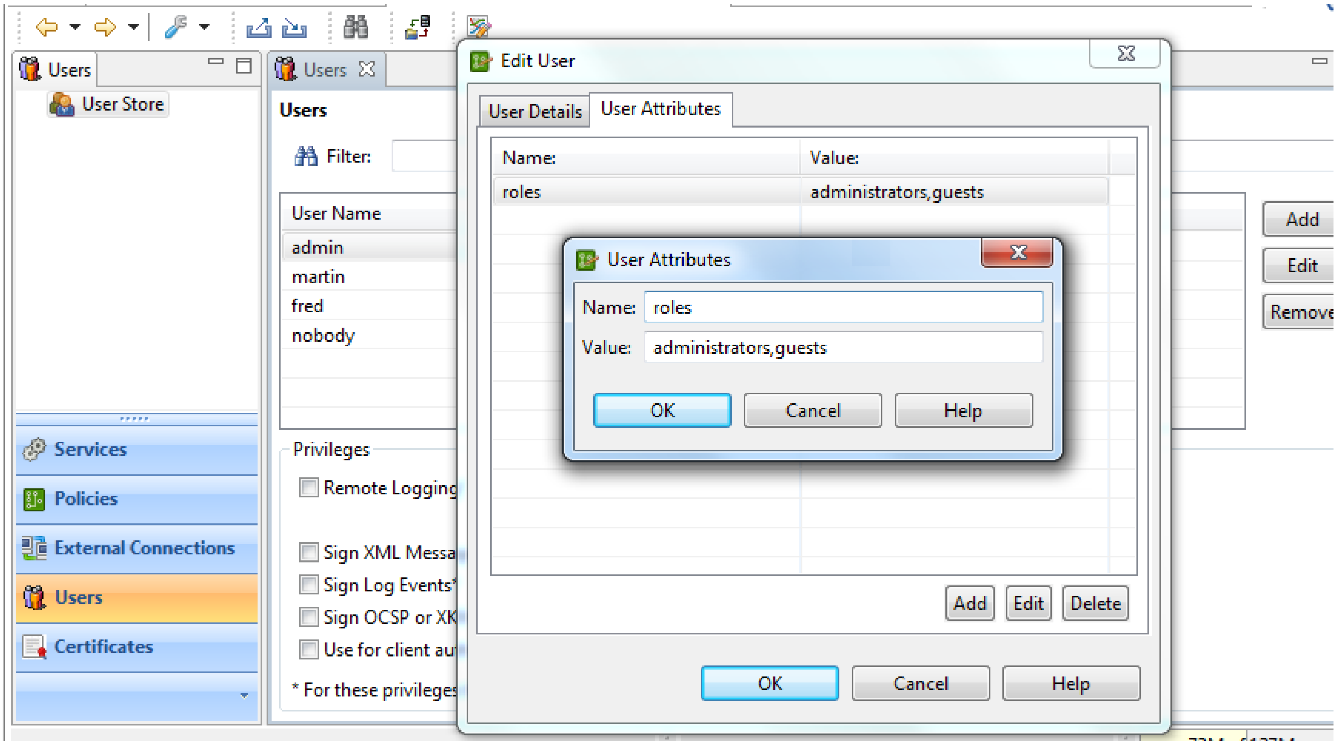
Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

Authorization Properties	Type	Default	Value
User roles file location (*)	string		/tmp/roles
Allowed roles	string[]		administrator

Oracle Enterprise Gateway Configuration:

Similar to File Authentication step the Fusion Gateway can implement role based Authorization via the Gateways own Local User Store. Authorization can be added to the previous step after successfully authenticating the user via the HTTP Basic filter.

1. Add a comma separated list of roles to the user attributes of a Local Store user. This is very similar to the roles list used in the OWSM authorization file.



User Name
admin
martin
fred
nobody

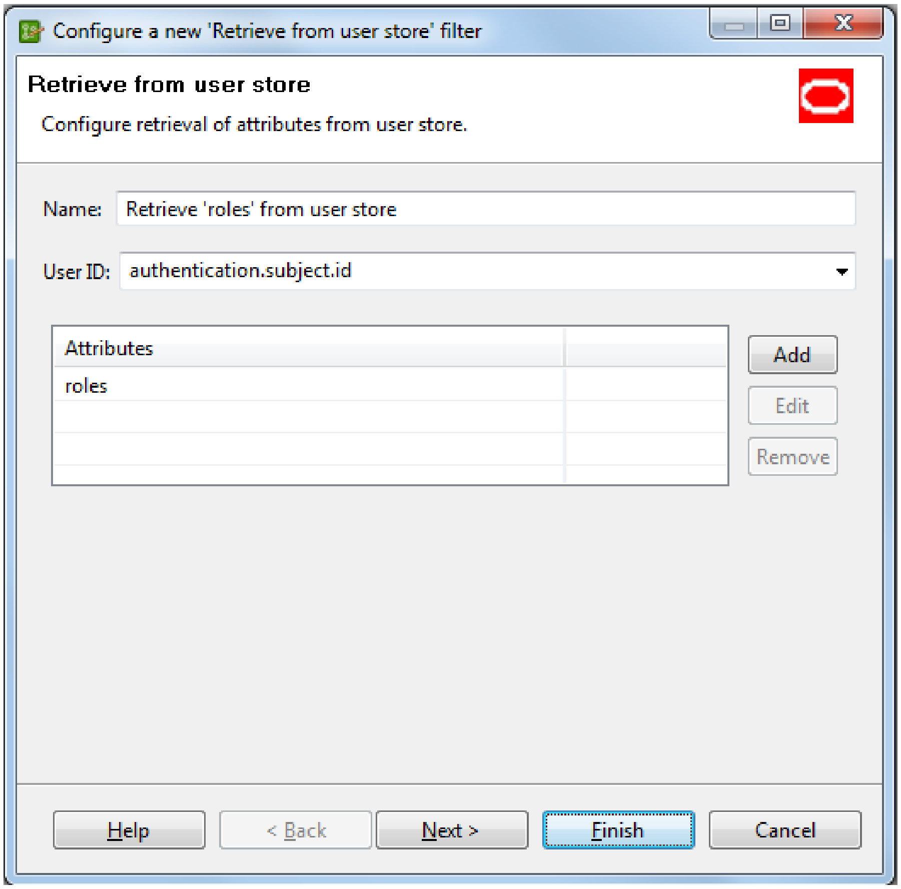
□ Remote Logging
□ Sign XML Messa
□ Sign Log Events*
□ Sign OCSP or XK
□ Use for client au

* For these privileges

Name:	Value:
roles	administrators,guests

Name: roles
Value: administrators,guests

2. After **HTTP Basic** authentication add a "**Retrieve from User Store"** filter to get the user's roles. Note that the message attribute authentication.subject.id will contain the username after successfully running HTTP Basic filter.



Retrieve from user store

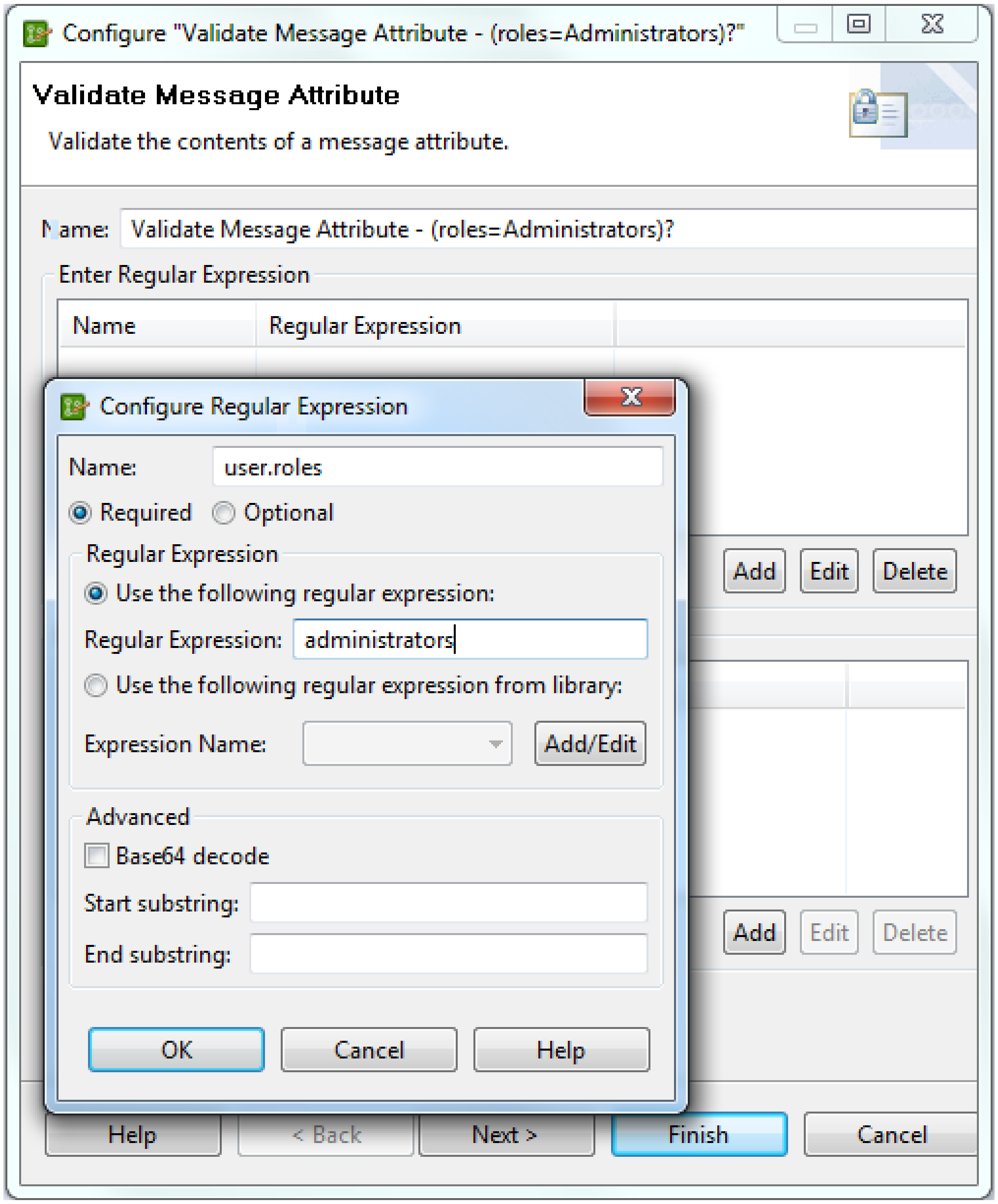
Configure retrieval of attributes from user store.

Name: Retrieve 'roles' from user store

User ID: authentication.subject.id

Attributes	
roles	

3. Next test if the user has a particular role on their roles list with a **Validate Message Attribute** filter. Click **Add** to enter a regular expression to test if a particular role like administrators is on the list. Note the Retrieve from user store copies the message attributes to user.<user attribute name> (so user.roles in this case)



Configure "Validate Message Attribute - (roles=Administrators)?"

Validate Message Attribute

Validate the contents of a message attribute.

Name: Validate Message Attribute - (roles=Administrators)?

Enter Regular Expression

Name	Regular Expression

Configure Regular Expression

Name: user.roles

● Required ○ Optional

Regular Expression

● Use the following regular expression:

Regular Expression: administrators

○ Use the following regular expression from library:

Expression Name:

Advanced

☐ Base64 decode

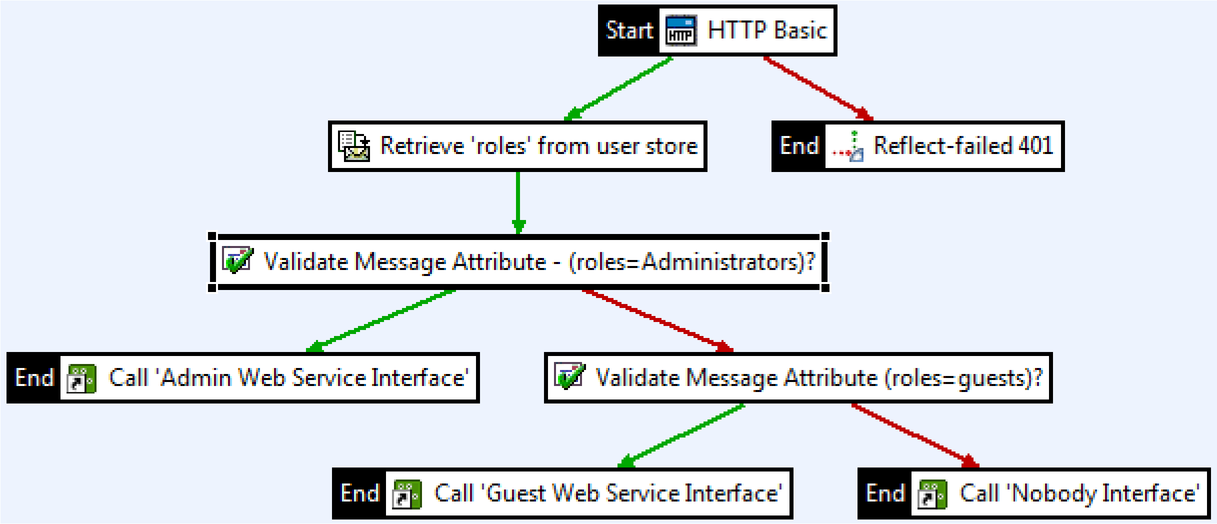
Start substring:

End substring:

4. If success then so far the forward to the path for administrators.

5. Add more checks for other roles that require testing.

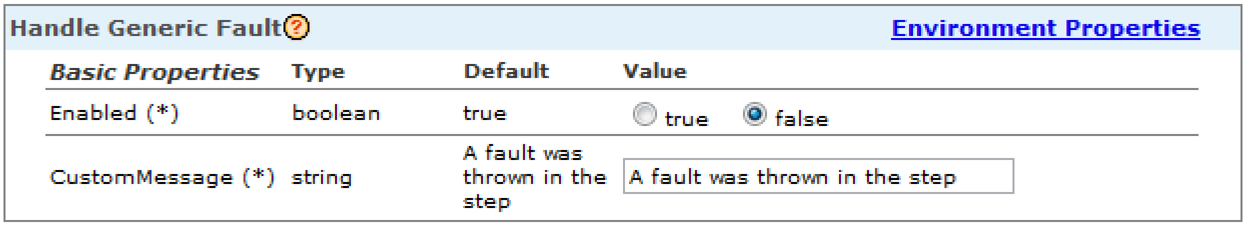
6. An example circuit might look like the following.


Start
HTTP Basic
Retrieve 'roles' from user store
End
Reflect-failed 401
Validate Message Attribute - (roles=Administrators)?
End
Call 'Admin Web Service Interface'
Validate Message Attribute (roles=guests)?
End
Call 'Guest Web Service Interface'
End
Call 'Nobody Interface'

Step: Handle Generic Fault

Example generic fault handler step

OWSM Configuration:


Handle Generic Fault

Environment Properties

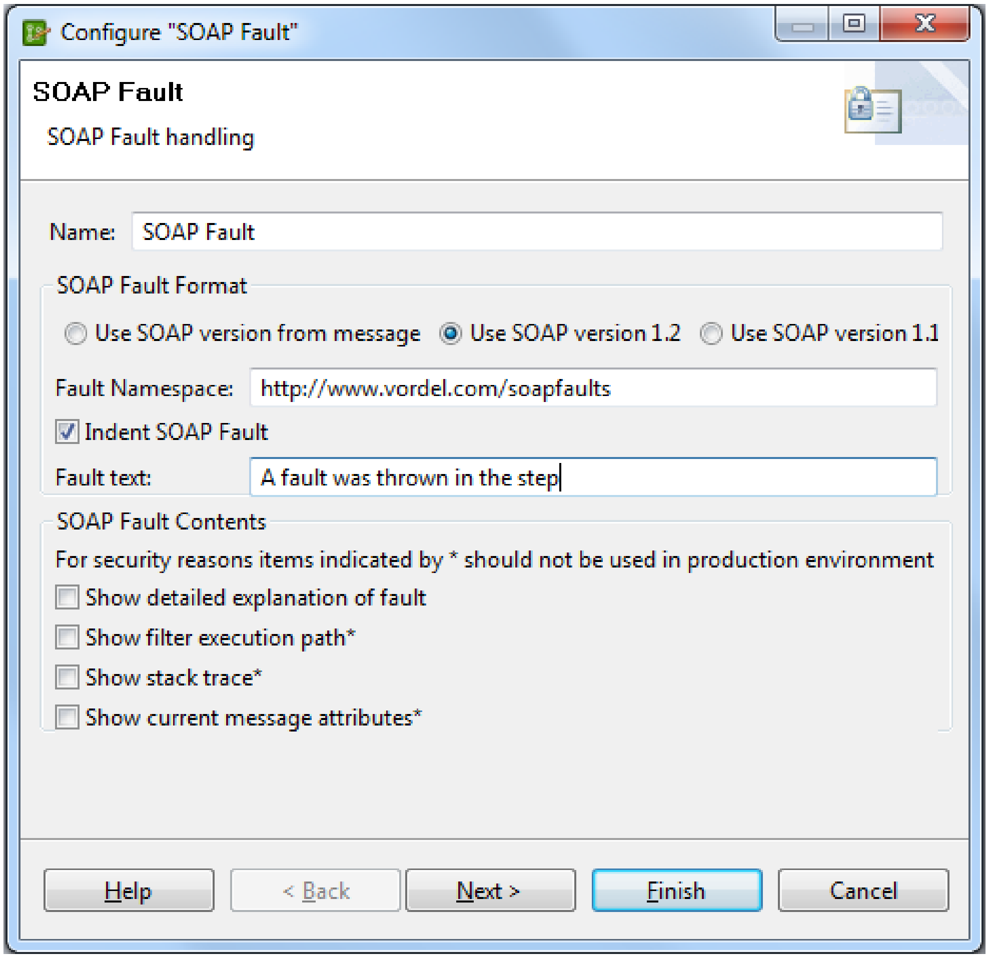
Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	○ true ● false
CustomMessage (*)	string	A fault was thrown in the step	A fault was thrown in the step

Provides custom message in the SOAP fault when errors are encountered.

Oracle Enterprise Gateway Configuration:

SOAP Fault Filter

1. By default, the Gateway returns a very basic SOAP Fault to the client when a message filter has failed.
2. SOAP Fault processor can be added to a policy to return more complicated error information to the client.
3. Using the **SOAP Fault** processor, administrators have the flexibility to configure just how much information to return to clients, depending on their individual requirements.


Configure "SOAP Fault"

SOAP Fault

SOAP Fault handling

Name: SOAP Fault

SOAP Fault Format

○ Use SOAP version from message ● Use SOAP version 1.2 ○ Use SOAP version 1.1

Fault Namespace: http://www.vordel.com/soapfaults

☑ Indent SOAP Fault

Fault text: A fault was thrown in the step

SOAP Fault Contents

For security reasons items indicated by * should not be used in production environment

☐ Show detailed explanation of fault

☐ Show filter execution path*

☐ Show stack trace*

☐ Show current message attributes*

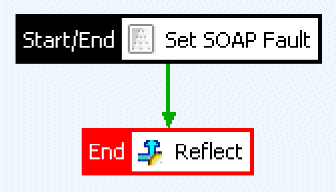
4. The SOAP Fault filter can either be returned in a circuit path or it can be set as the fault handler for a particular policy. In this case it will be invoked when a filter aborts.

Customized SOAP Faults

1. It is possible to produce a customized SOAP fault to return to the client.
The **Set Message** filter can change the contents of the message body to any arbitrary content. When an exception occurs in a policy, it is possible to use this filter to customize the body of the SOAP fault.
e.g.

2. Create a Fault Policy

Create a policy called "Fault Circuit". This policy will use the **Set Message** filter to customize the body of the SOAP fault. When configuring this filter, enter the contents of the customized SOAP fault that you want to return to clients in the text area provided.

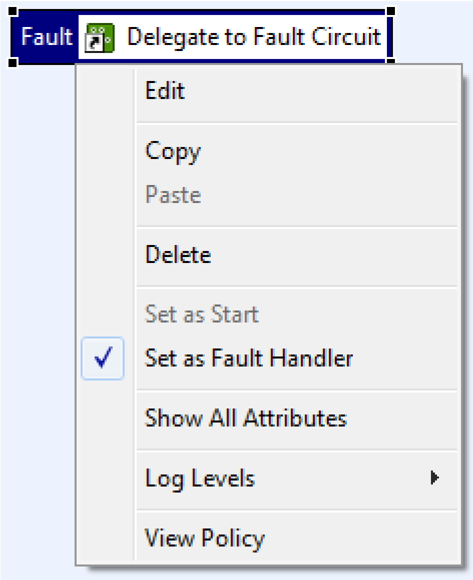


The second **Reflect** filter is used to return the SOAP Fault back to the client.

3. Drop a **Policy Shortcut** onto the policy canvas.

4. Point the shortcut to the newly created Fault Policy

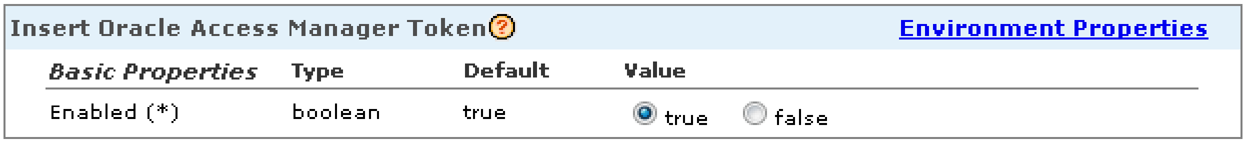
5. Right click on the Policy Shortcut and set it as the fault handler



Step: Insert Oracle Access Manager Token

Inserts an ObSSOCookie in the SOAP security header.

OWSM Configuration:



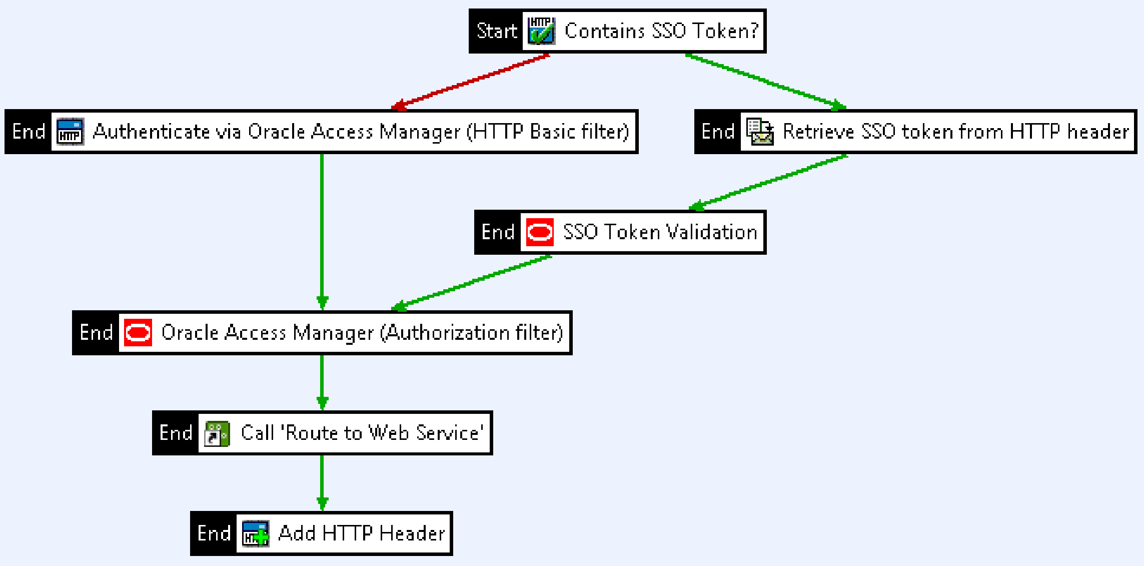
Insert Oracle Access Manager Token

Environment Properties

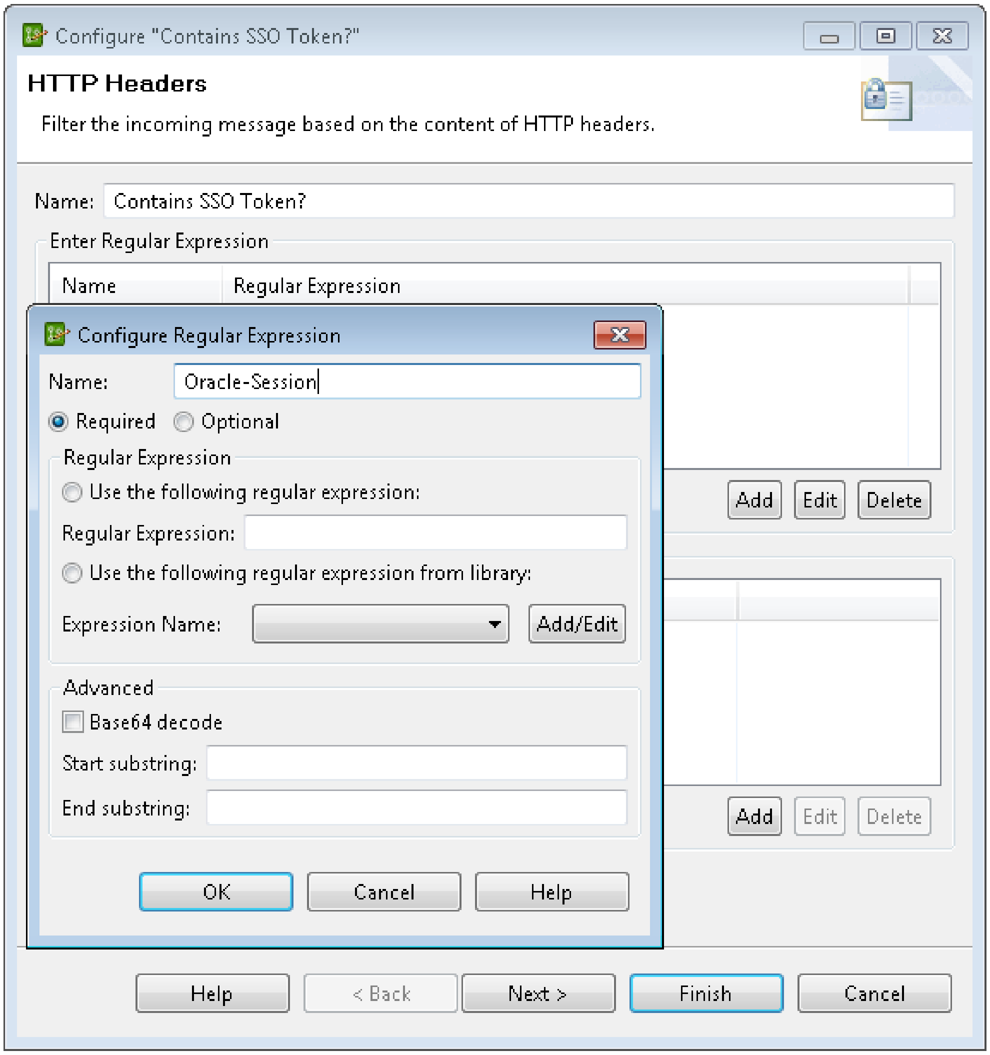
Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

Oracle Enterprise Gateway Configuration:

The equivalent functionality to ObSSOCookie can be achieved with the Fusion Gateway by passing the SSO token that was inserted into an HTTP header during the Oracle Access Manager Authorization phase. An incoming policy can be setup to check for the sso token header and call the SSO token validate filter rather than authenticating the user again. For example the circuit could add the following filters to the Oracle Access Manager Authenticate Authorization :-


Start Contains SSO Token?
End Authenticate via Oracle Access Manager (HTTP Basic filter)
End Retrieve SSO token from HTTP header
End SSO Token Validation
End Oracle Access Manager (Authorization filter)
End Call 'Route to Web Service'
End Add HTTP Header

1. Configure a **'Validate HTTP Headers'** filter to validate that an existing token exists in the HTTP Header of the message.


Configure "Contains SSO Token?"

HTTP Headers

Filter the incoming message based on the content of HTTP headers.

Name: Contains SSO Token?

Enter Regular Expression

Name	Regular Expression

Configure Regular Expression

Name: Oracle-Session

● Required ○ Optional

Regular Expression

○ Use the following regular expression:

Regular Expression:

○ Use the following regular expression from library:

Expression Name:

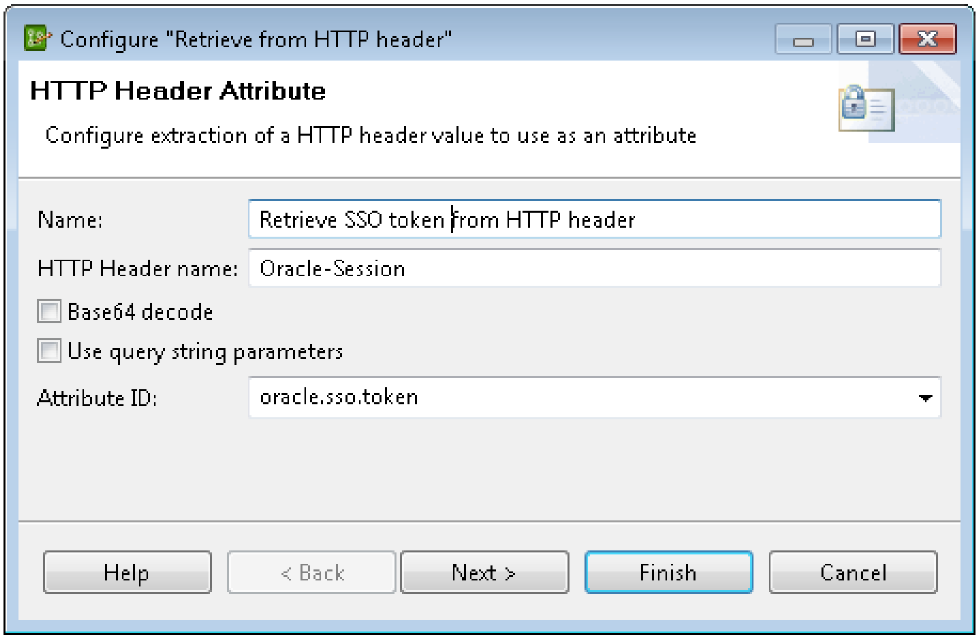
Advanced

□ Base64 decode

Start substring:

End substring:

2. Configure a **'Retrieve from Attribute'** filter that will retrieve the existing token attribute from the HTTP Header of the message.



Configure "Retrieve from HTTP header"

HTTP Header Attribute

Configure extraction of a HTTP header value to use as an attribute

Name: Retrieve SSO token from HTTP header

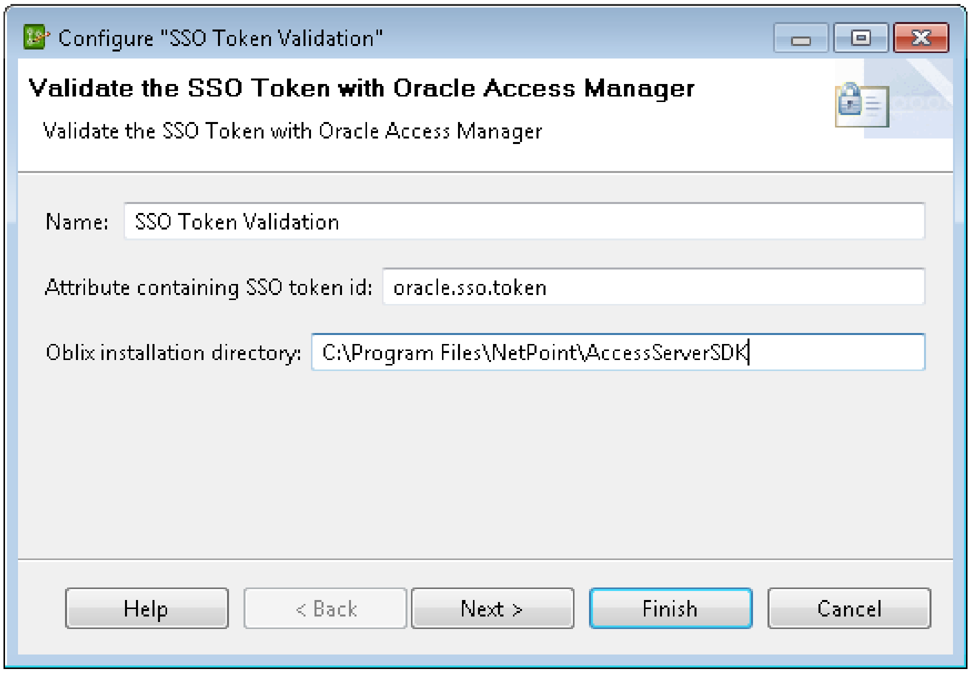
HTTP Header name: Oracle-Session

□ Base64 decode

□ Use query string parameters

Attribute ID: oracle.sso.token

3. Configure an **'Oracle Access Manager Session Validation'** filter that will check for the validity of the SSO token.



Configure "SSO Token Validation"

Validate the SSO Token with Oracle Access Manager

Validate the SSO Token with Oracle Access Manager

Name: SSO Token Validation

Attribute containing SSO token id: oracle.sso.token

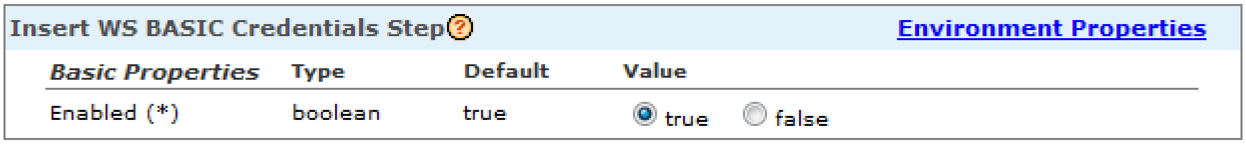
Oblix installation directory: C:\Program Files\NetPoint\AccessServerSDK

4. Connect to the **'Oracle Access Manager Authorization'** filter that was specified in the Oracle Access Manager Authenticate Authorize step.

Step: Insert WSBASIC Credentials

Inserts user name password credentials in a security soap header

OWSM Configuration:

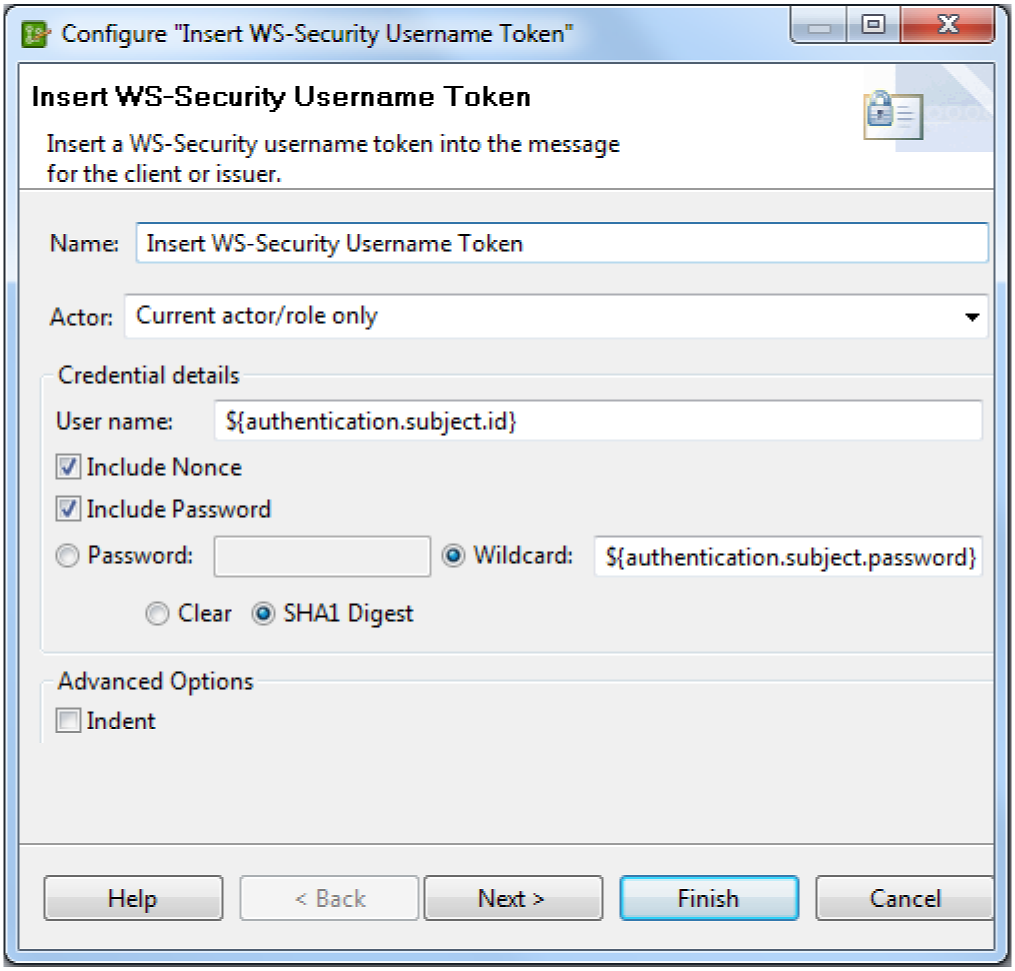


Insert WS BASIC Credentials Step

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

Oracle Enterprise Gateway Configuration:

Drag an **"Insert WS-Security UsernameToken"** filter from the Authentication category to the target circuit path. This should follow on from extracting and authenticating credentials from another filter like HTTP Basic..



Configure "Insert WS-Security Username Token"

Insert WS-Security Username Token

Insert a WS-Security username token into the message for the client or issuer.

Name: Insert WS-Security Username Token

Actor: Current actor/role only

Credential details

User name: ${authentication.subject.id}

☑ Include Nonce

☑ Include Password

○ Password: ● Wildcard: ${authentication.subject.password}

○ Clear ● SHA1 Digest

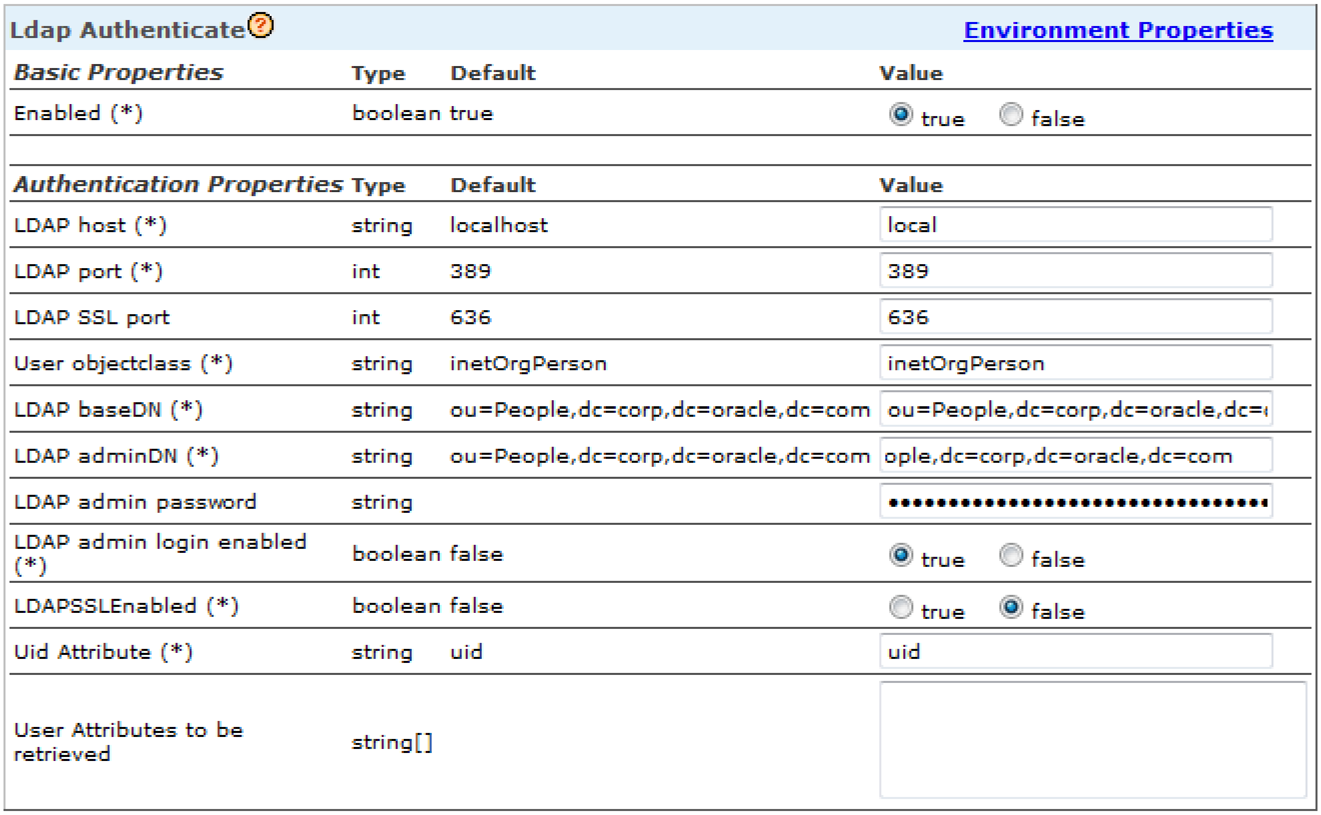
Advanced Options

□ Indent

Step: Ldap Authenticate

Performs the authentication with a LDAP Server

OWSM Configuration:



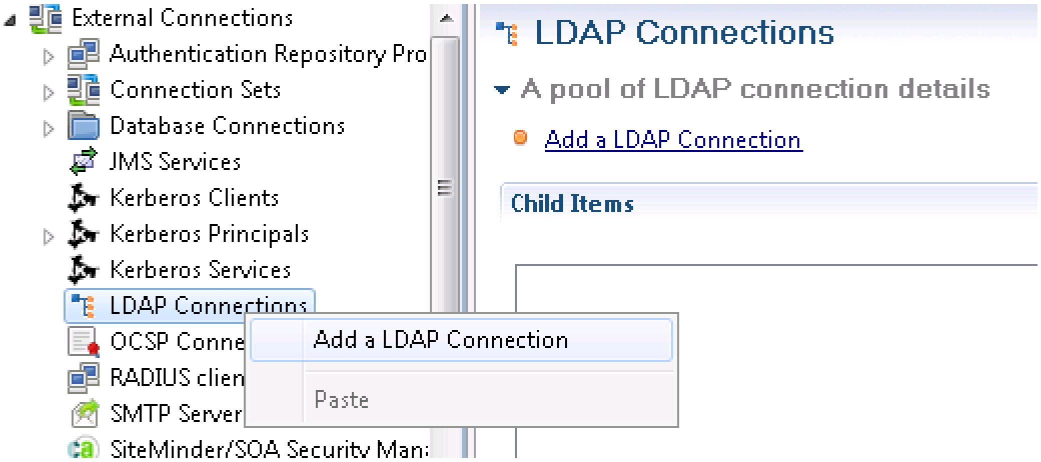
Ldap Authenticate **Environment Properties**

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

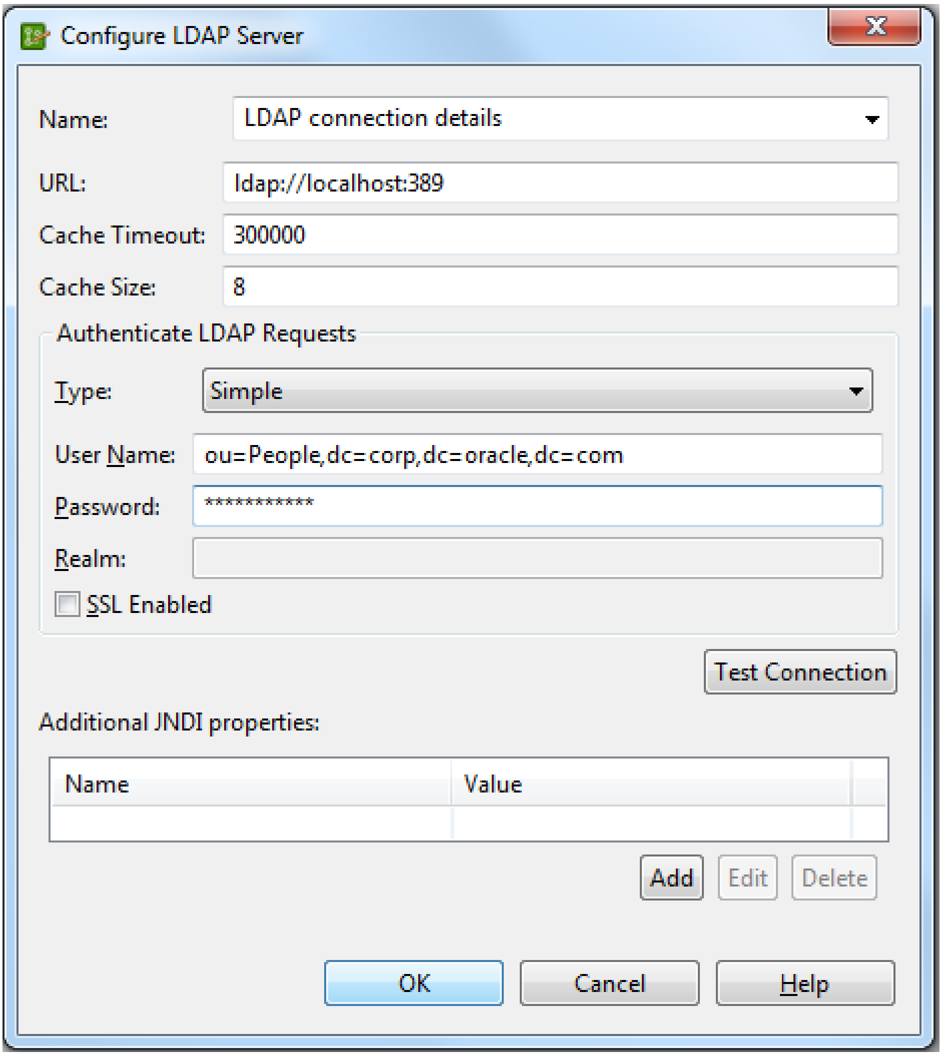
Authentication Properties	Type	Default	Value
LDAP host (*)	string	localhost	local
LDAP port (*)	int	389	389
LDAP SSL port	int	636	636
User objectclass (*)	string	inetOrgPerson	inetOrgPerson
LDAP baseDN (*)	string	ou=People,dc=corp,dc=oracle,dc=com	ou=People,dc=corp,dc=oracle,dc=
LDAP adminDN (*)	string	ou=People,dc=corp,dc=oracle,dc=com	ople,dc=corp,dc=oracle,dc=com
LDAP admin password	string		••••••••••••••••••••••••••••••••
LDAP admin login enabled (*)	boolean	false	● true ○ false
LDAPSSLEnabled (*)	boolean	false	○ true ● false
Uid Attribute (*)	string	uid	uid
User Attributes to be retrieved	string[]		

Oracle Enterprise Gateway Configuration:

External connections > LDAP Connections > Add a LDAP Connection



Enter configuration details for the LDAP



Configure LDAP Server

Name: LDAP connection details

URL: ldap://localhost:389

Cache Timeout: 300000

Cache Size: 8

Authenticate LDAP Requests

Type: Simple

User Name: ou=People,dc=corp,dc=oracle,dc=com

Password: ***********

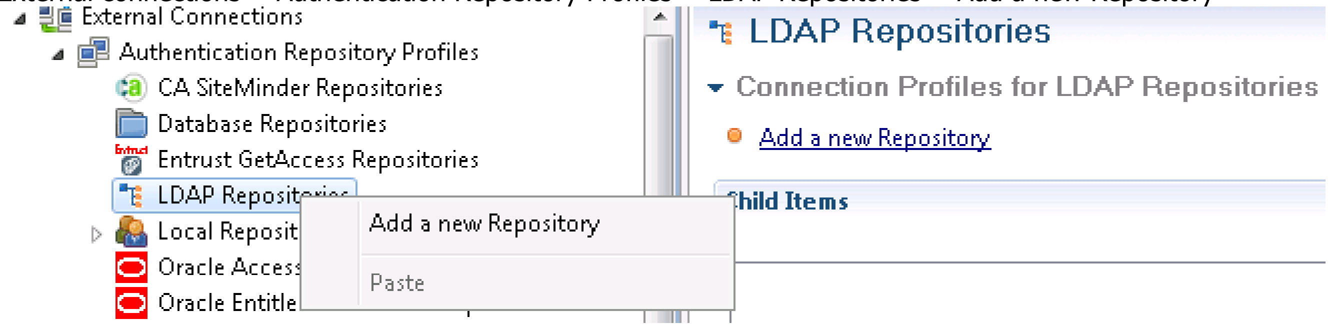
Realm:

☐ SSL Enabled

connection.

Create an LDAP authentication repository entry which then can be used in HTTP Basic, Digest, and WS-Security username password filters for authentication.

External connections > Authentication Repository Profiles > LDAP Repositories > Add a new Repository



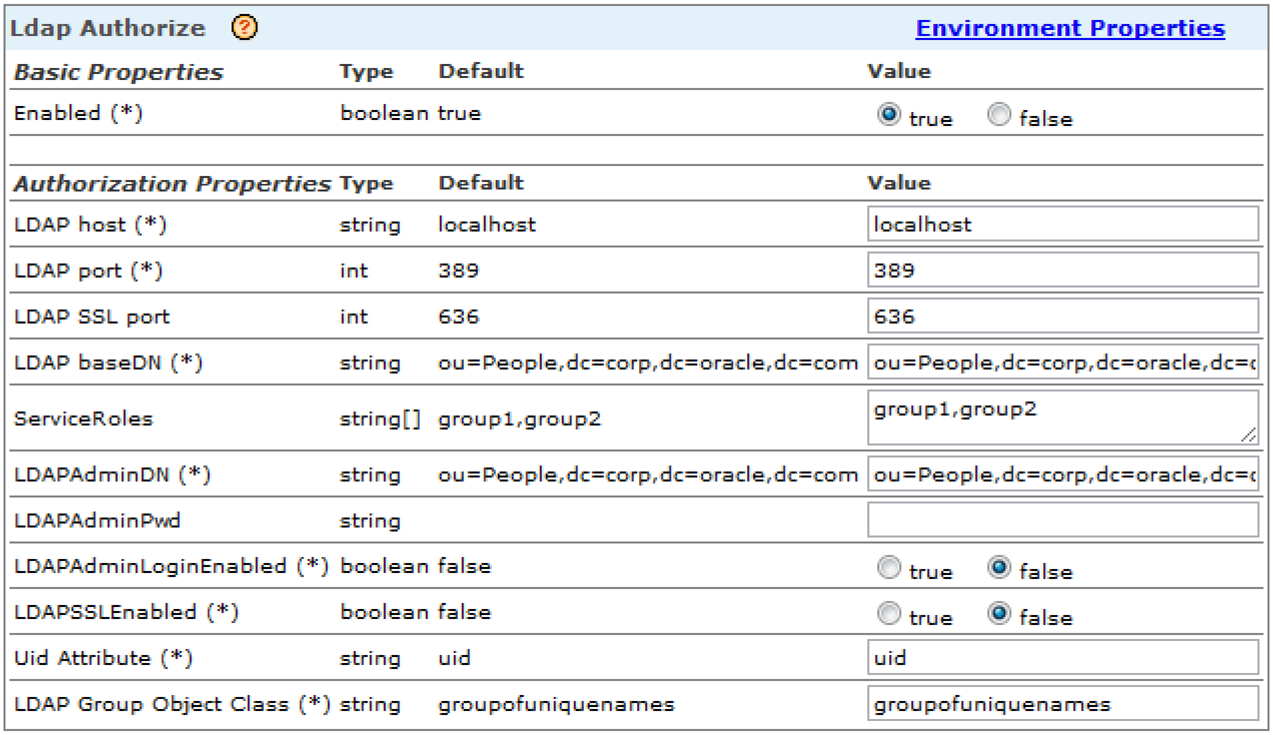
Configure the repository as required:

In a filter that requires authentication against LDAP, simply select the repository named above.

Step: Ldap Authorize

Authorizes request by retrieving role from LDAP and checking against roles allowed by service.

OWSM Configuration:



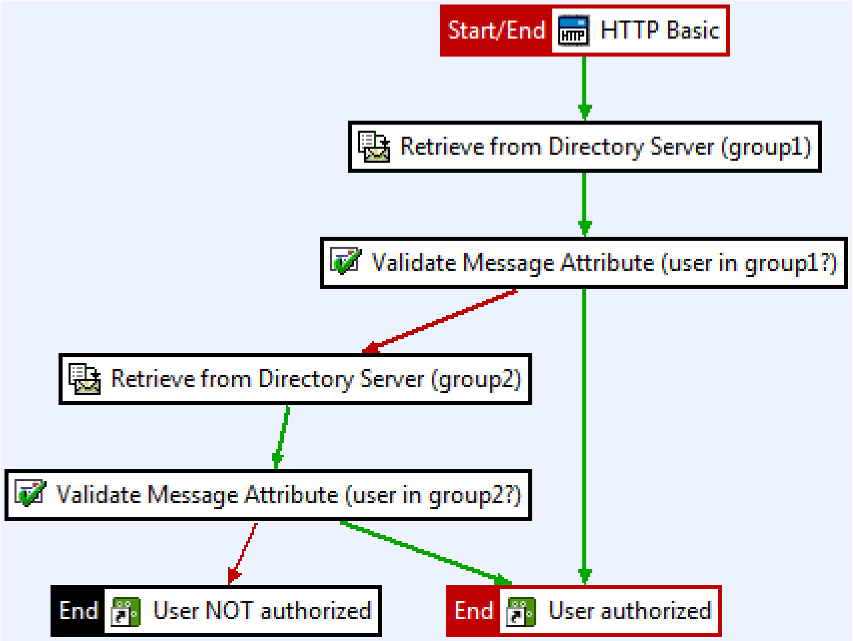
Ldap Authorize ?

Environment Properties

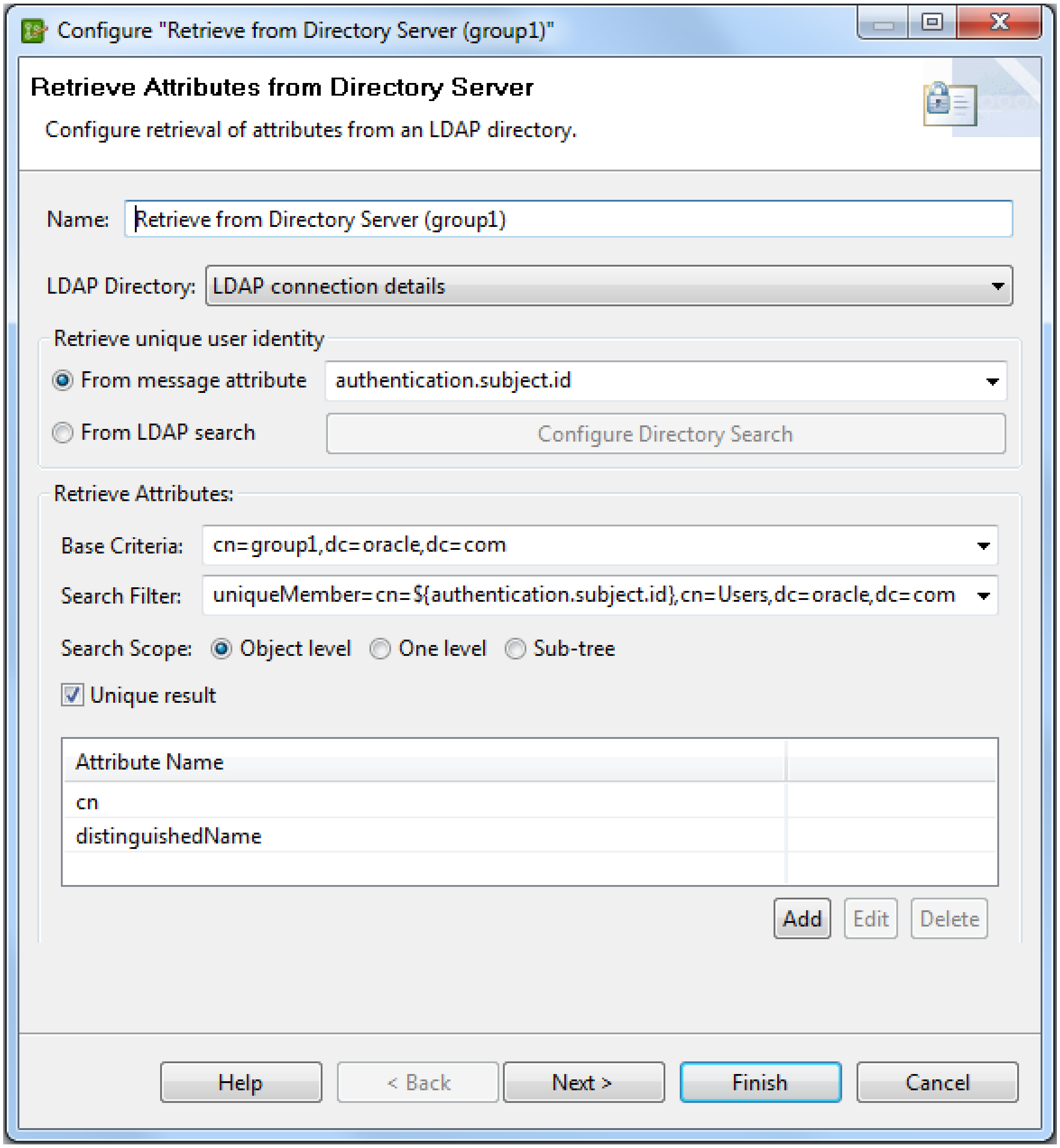
Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false
Authorization Properties	**Type**	**Default**	**Value**
LDAP host (*)	string	localhost	localhost
LDAP port (*)	int	389	389
LDAP SSL port	int	636	636
LDAP baseDN (*)	string	ou=People,dc=corp,dc=oracle,dc=com	ou=People,dc=corp,dc=oracle,dc=
ServiceRoles	string[]	group1,group2	group1,group2
LDAPAdminDN (*)	string	ou=People,dc=corp,dc=oracle,dc=com	ou=People,dc=corp,dc=oracle,dc=
LDAPAdminPwd	string		
LDAPAdminLoginEnabled (*)	boolean	false	○ true ● false
LDAPSSLEnabled (*)	boolean	false	○ true ● false
Uid Attribute (*)	string	uid	uid
LDAP Group Object Class (*)	string	groupofuniquenames	groupofuniquenames

Oracle Enterprise Gateway Configuration:

1. Use LDAP connection and repository setup as detailed in LDAP Authenticate above.
2. After Authentication with HTTP Basic, Digest or WS-Security username password filters send circuit to a **"Retrieve Attribute from Directory Server"** filter and then **"Validate Message Attribute"** filter. This combination can be used to check a role (group) for a user's membership.
3. Repeat for each group that needs checking. e.g.


Start/End
HTTP Basic
Retrieve from Directory Server (group1)
Validate Message Attribute (user in group1?)
Retrieve from Directory Server (group2)
Validate Message Attribute (user in group2?)
End
User NOT authorized
End
User authorized

4. Example of a Retrieve from Directory Server for group “group1”



Configure "Retrieve from Directory Server (group1)"

Retrieve Attributes from Directory Server

Configure retrieval of attributes from an LDAP directory.

Name: Retrieve from Directory Server (group1)

LDAP Directory: LDAP connection details

Retrieve unique user identity

● From message attribute: authentication.subject.id

○ From LDAP search

Retrieve Attributes:

Base Criteria: cn=group1,dc=oracle,dc=com

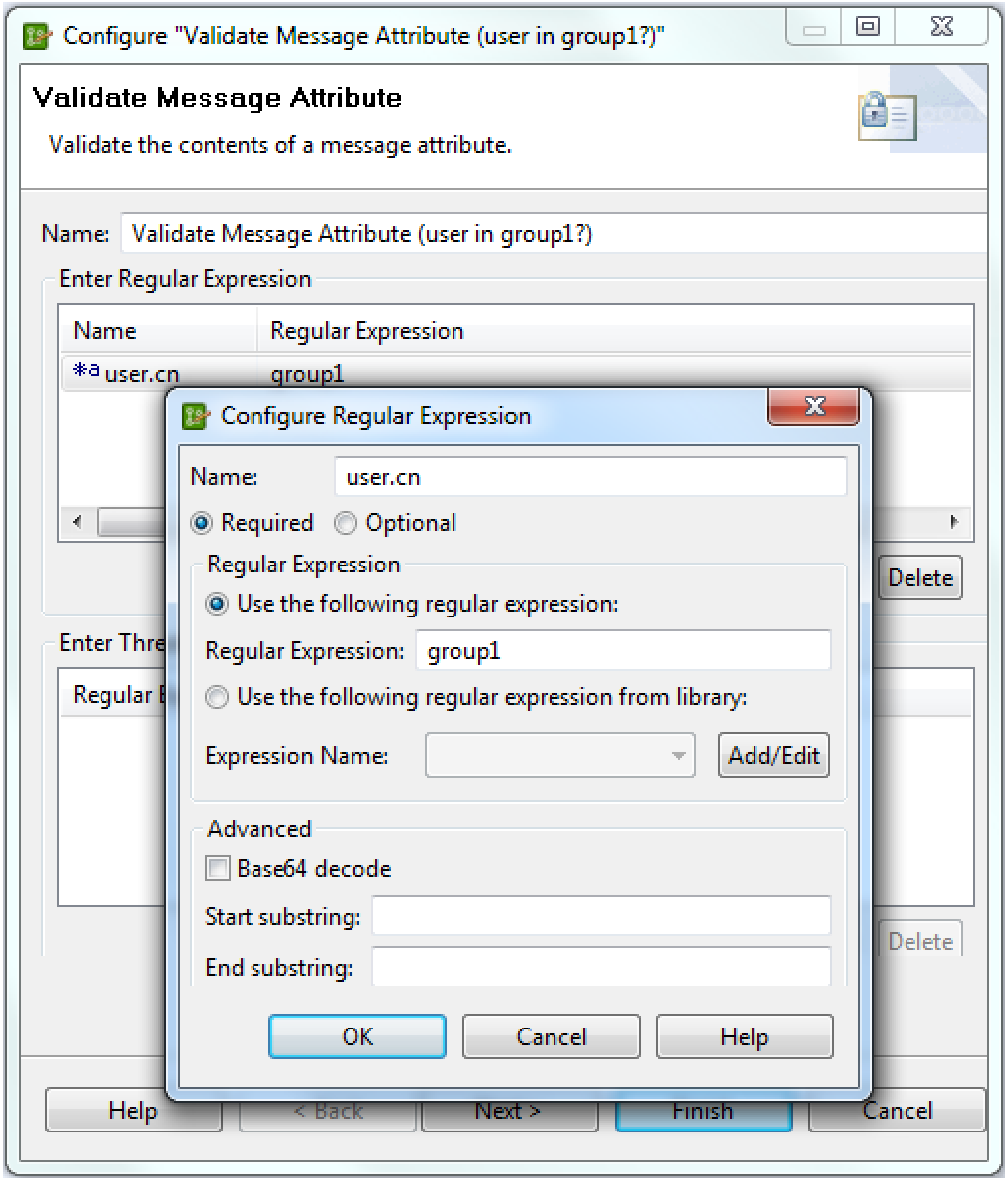
Search Filter: uniqueMember=cn=${authentication.subject.id},cn=Users,dc=oracle,dc=com

Search Scope: ● Object level ○ One level ○ Sub-tree

☑ Unique result

Attribute Name
cn
distinguishedName

5. Example of a Validate Message Attribute for group "group1"
This filter returns success if the user is found to be a member of the role group1. The authorized user is forwarded to the next role check and finally to the user authorized policy which is shown as a shortcut in the circuit above.



Configure "Validate Message Attribute (user in group1?)"

Validate Message Attribute

Validate the contents of a message attribute.

Name: Validate Message Attribute (user in group1?)

Enter Regular Expression

Name	Regular Expression
*a user.cn	group1

Configure Regular Expression

Name: user.cn

● Required ○ Optional

Regular Expression

● Use the following regular expression:

Regular Expression: group1

○ Use the following regular expression from library:

Expression Name:

Advanced

□ Base64 decode

Start substring:

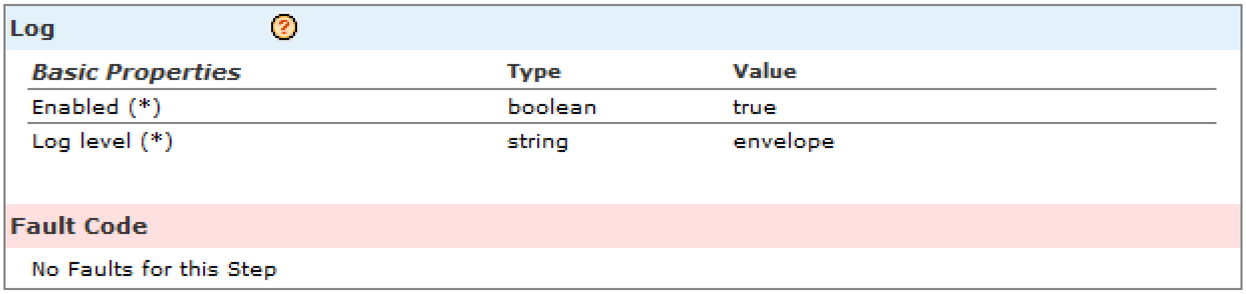
End substring:

6. Similarly the user not authorized branch is a shortcut. This could be to a policy that eventually reflects an appropriate failure message back to the client.

Step: Log

Log the request/response message

OWSM Configuration:



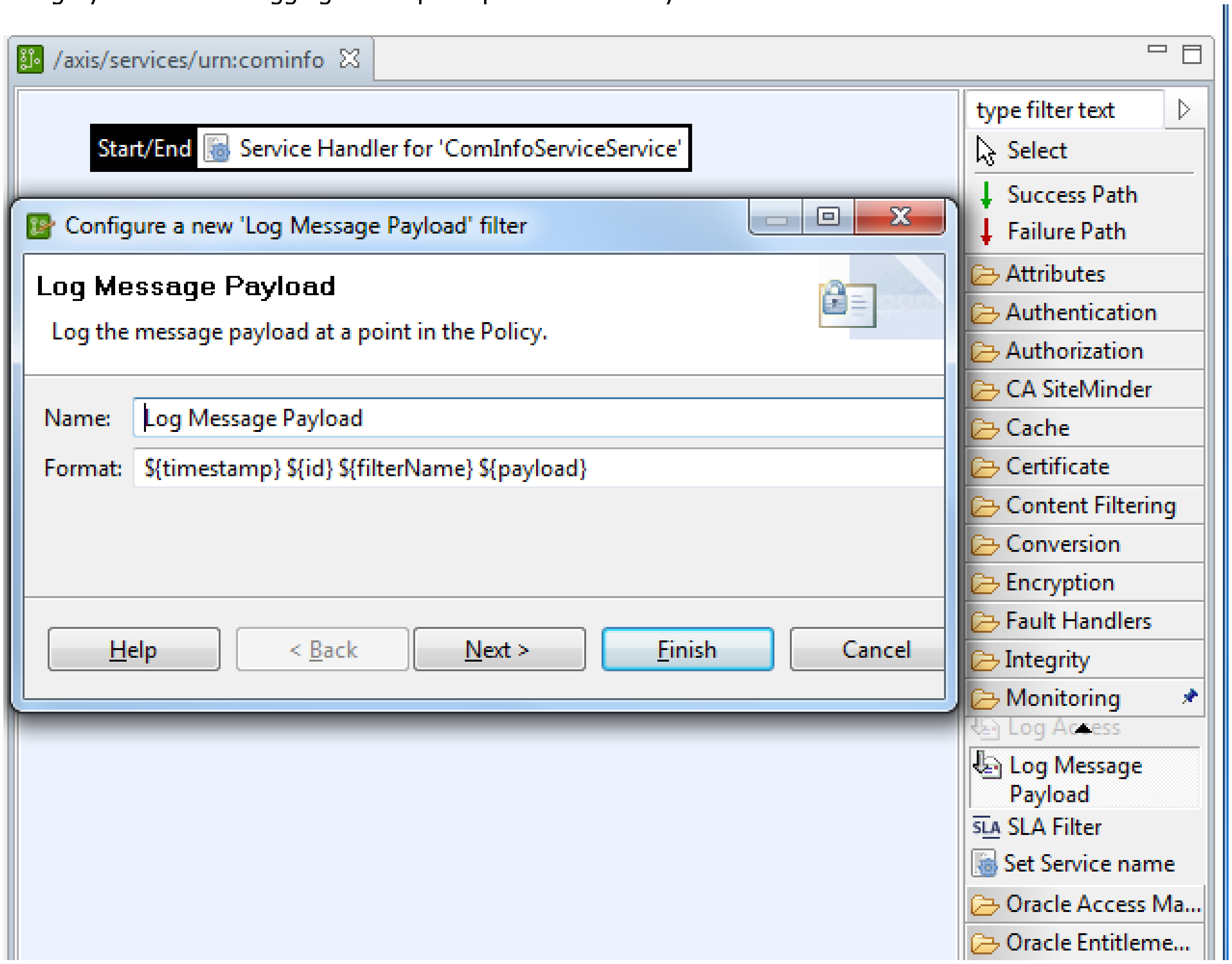
Log

Basic Properties	Type	Value
Enabled (*)	boolean	true
Log level (*)	string	envelope

Fault Code

No Faults for this Step

Oracle Enterprise Gateway Configuration:
Insert **"Log Message Payload"** filter from the Monitoring category into desired logging interception points in circuitry.



Configure a new 'Log Message Payload' filter

Log Message Payload

Log the message payload at a point in the Policy.

Name: Log Message Payload

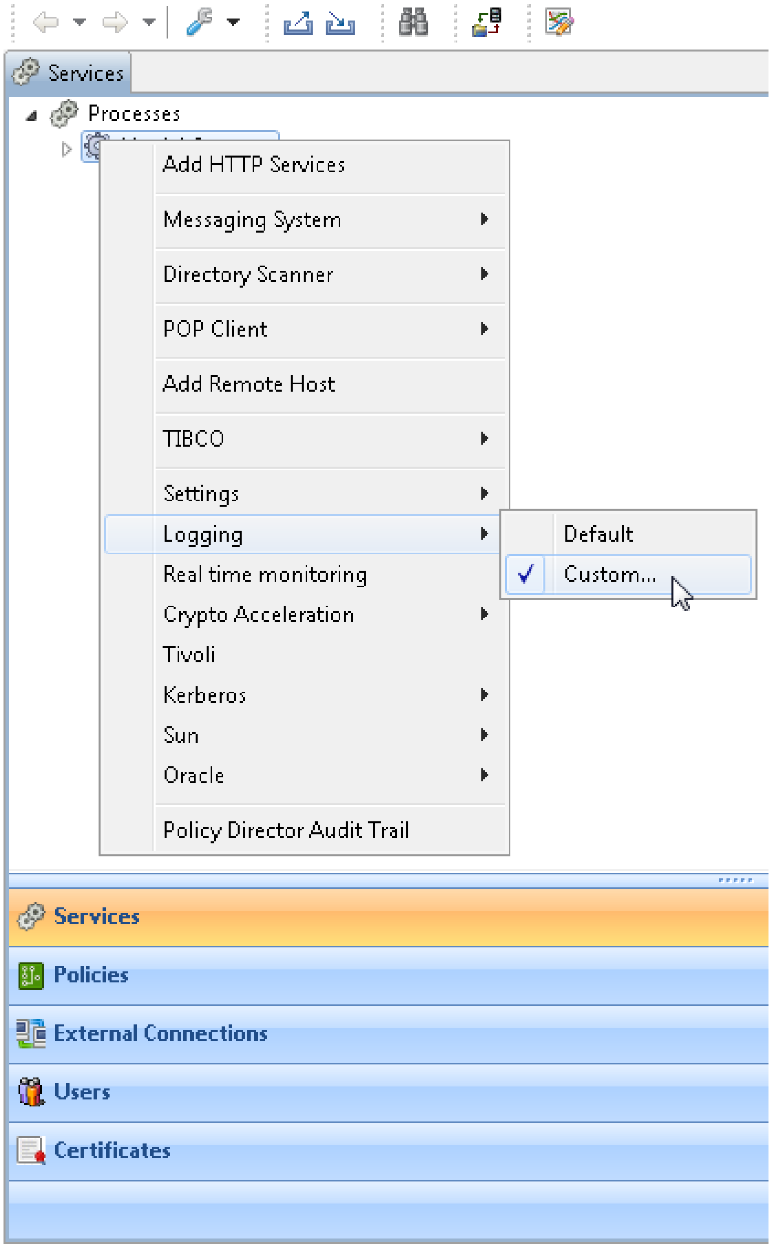
Format: ${timestamp} ${id} ${filterName} ${payload}

The Format field supports the following wildcards :-

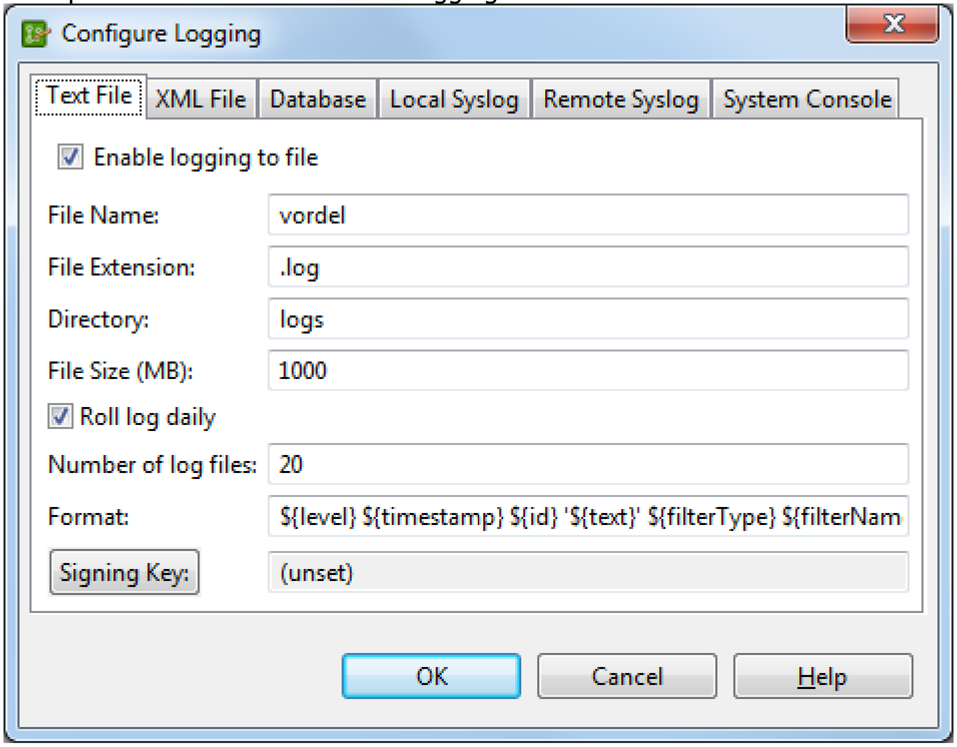
- **level:**
- The log level (i.e. fatal, fail, success).
- **id:**
- The unique transaction ID assigned to the message.
- **ip:**
- The IP address of the client that sent the request.

- **timestamp:**
- The time that the message was processed in user-readable form.
- **filterName:**
- The name of the filter that generated the log message.
- **filterType:**
- The type of the filter that logged the message.
- **text:**
- The text of the log message that was configured in the filter itself.
- **payload:**
- The complete contents of the HTTP request, including HTTP headers, body, and attachments.

Then enable logging to file, database, syslog or console from a right click on the Fusion Gateway process.



Select the **"Logging->Custom ..."** to bring up the configure logging window.
Example below shows the text file logging has been enabled.



☑ Enable logging to file

File Name: vordel

File Extension: .log

Directory: logs

File Size (MB): 1000

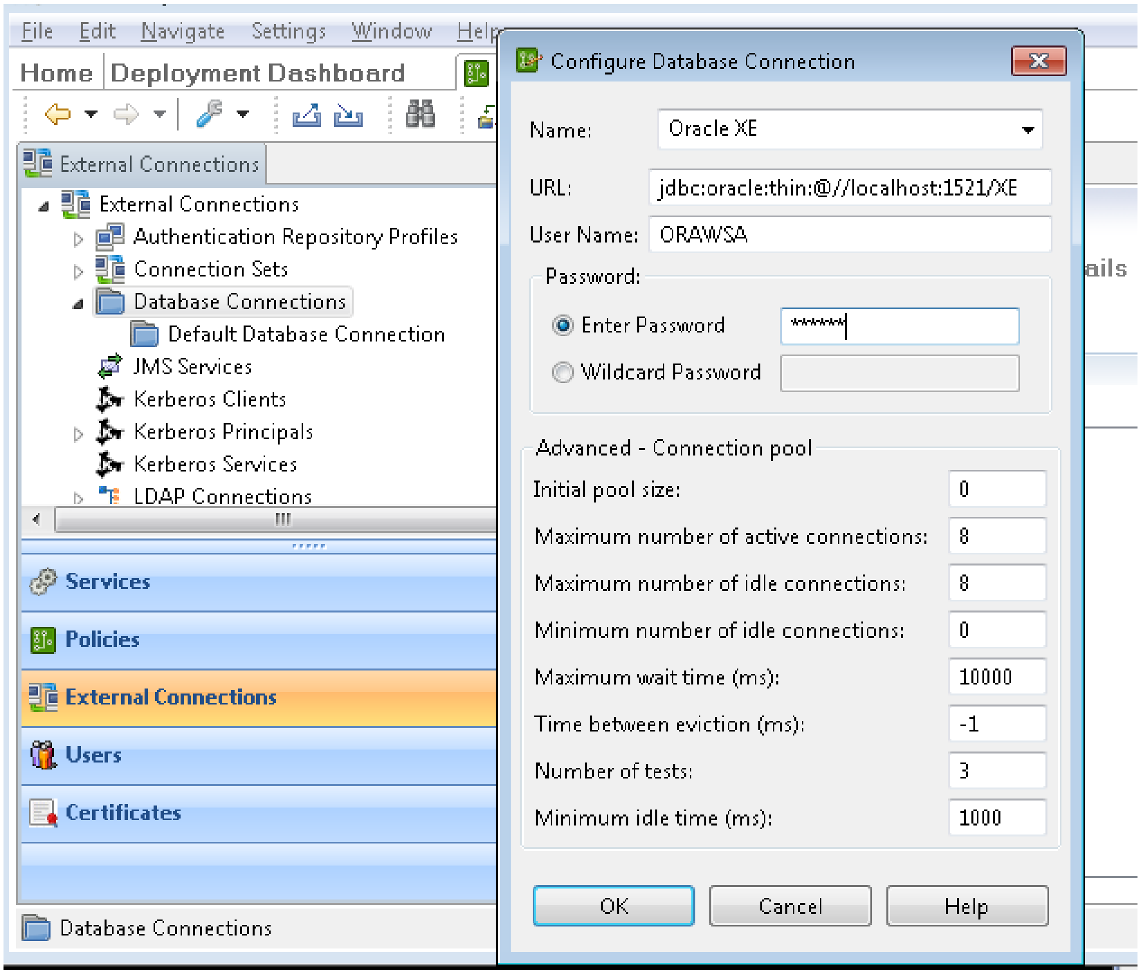
☑ Roll log daily

Number of log files: 20

Format: ${level} ${timestamp} ${id} '${text}' ${filterType} ${filterNam

Signing Key: (unset)

or for a database logging first add the database connection details in the External Connections selection panel. Right click on **"Database Connections"** and select "**Add a Database Connection".**
Fill out the database details like the example below and click OK.



Configure Database Connection

Name: Oracle XE

URL: jdbc:oracle:thin:@//localhost:1521/XE

User Name: ORAWSA

Password:

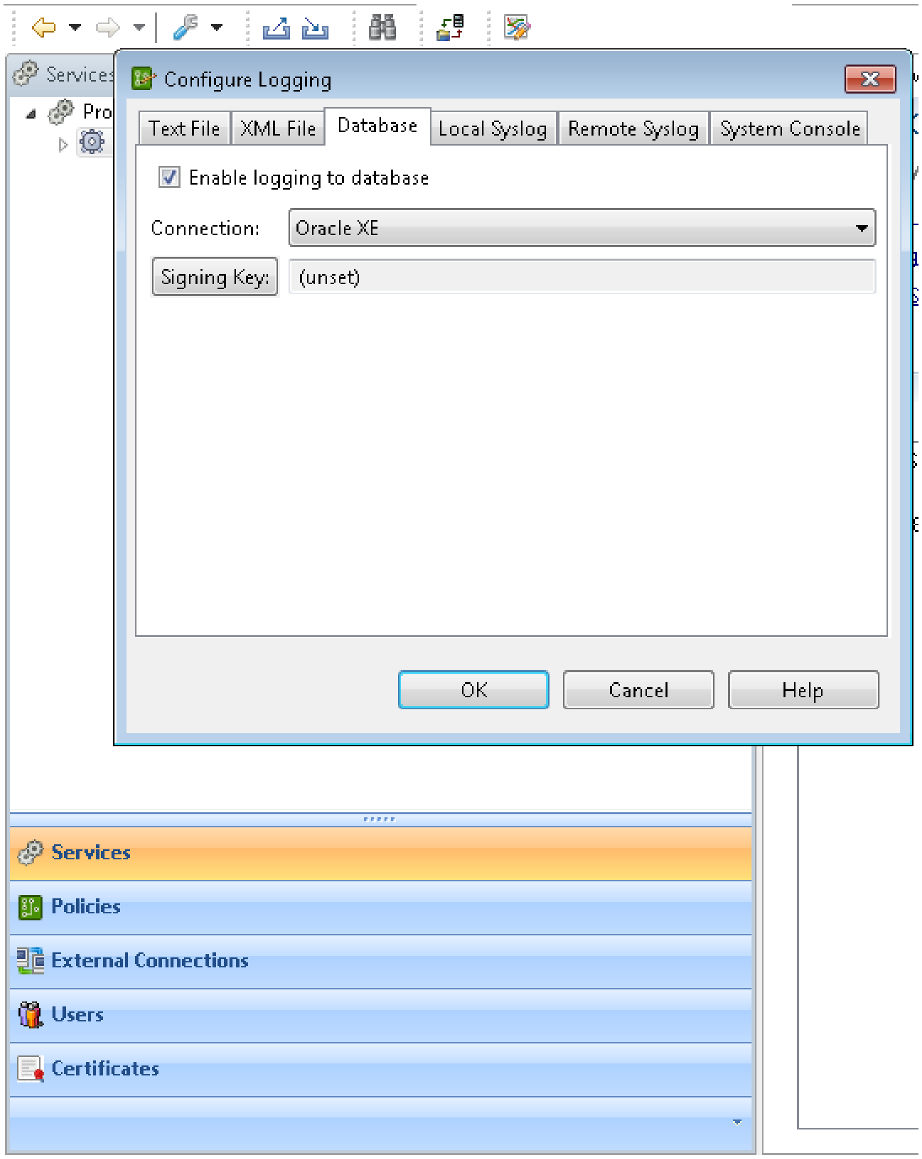
● Enter Password *******

○ Wildcard Password

Advanced - Connection pool

Initial pool size:	0
Maximum number of active connections:	8
Maximum number of idle connections:	8
Minimum number of idle connections:	0
Maximum wait time (ms):	10000
Time between eviction (ms):	-1
Number of tests:	3
Minimum idle time (ms):	1000

then set the logging to the configured database connection from the database tab in the **“configure logging**” database tab



Configure Logging

☑ Enable logging to database

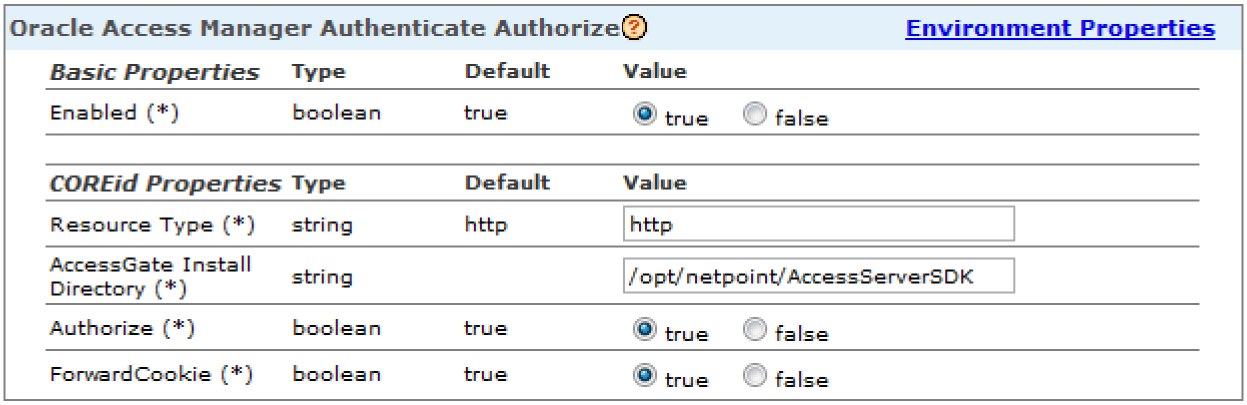
Connection: Oracle XE

Signing Key: (unset)

Step: Oracle Access Manager Authenticate Authorize

Authenticate and Authorize URLs access with Oracle Access Manager Access Server

OWSM Configuration:



Oracle Access Manager Authenticate Authorize ? **Environment Properties**

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

COREid Properties	Type	Default	Value
Resource Type (*)	string	http	http
AccessGate Install Directory (*)	string		/opt/netpoint/AccessServerSDK
Authorize (*)	boolean	true	● true ○ false
ForwardCookie (*)	boolean	true	● true ○ false

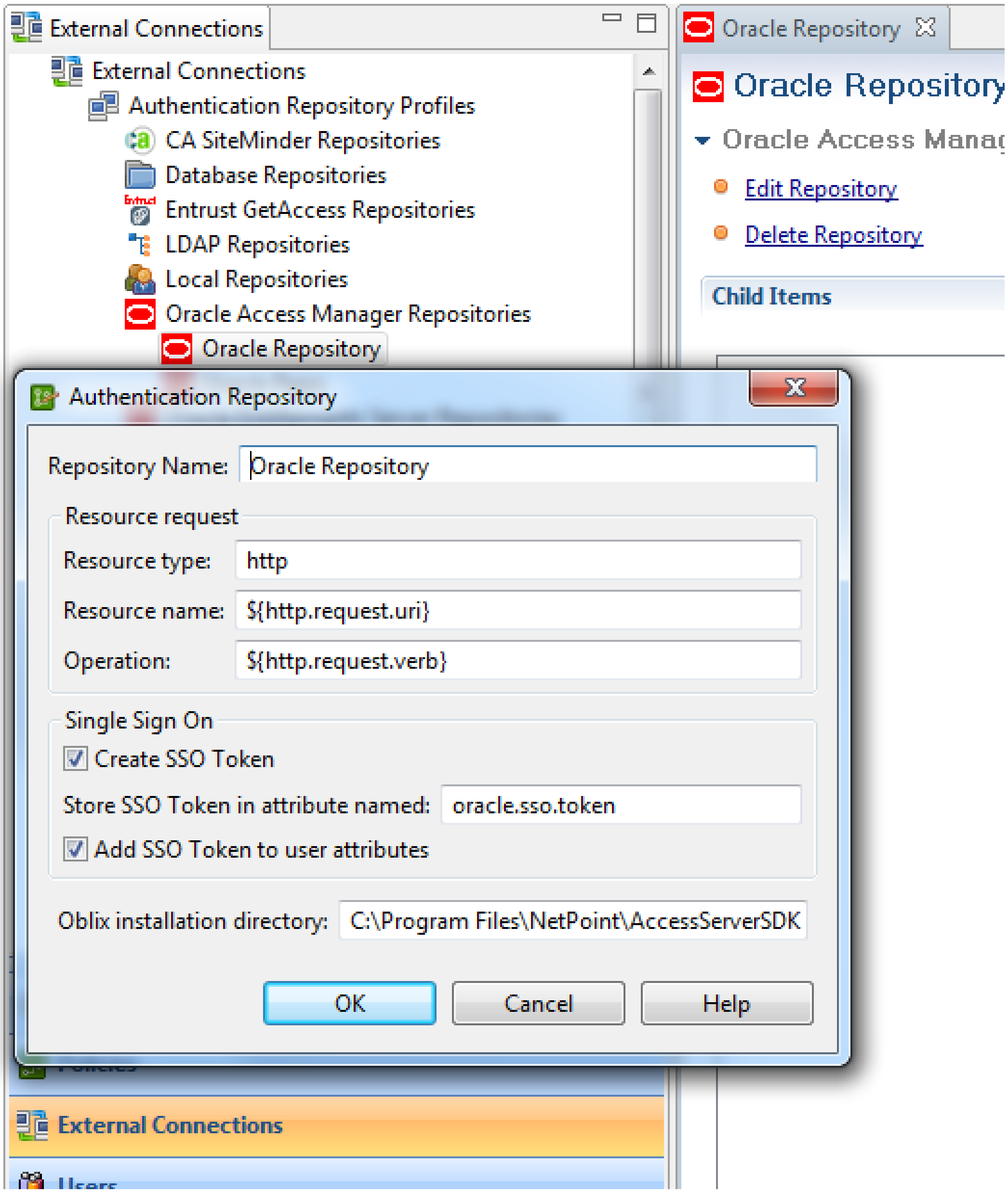
Oracle Enterprise Gateway Configuration:

An Oracle Access Manager repository needs to be registered in the Gateway. To do this

1. Click on the **"External Connections"** navigation bar.

2. Right Click on the **"Oracle Access Manager Repositories"** and **"add a new Repository"**

3. Fill in the repository details as below including the path to the AccessServerSDK on the local gateway machine.



Authentication Repository

Repository Name: Oracle Repository

Resource request

Resource type: http

Resource name: ${http.request.uri}

Operation: ${http.request.verb}

Single Sign On

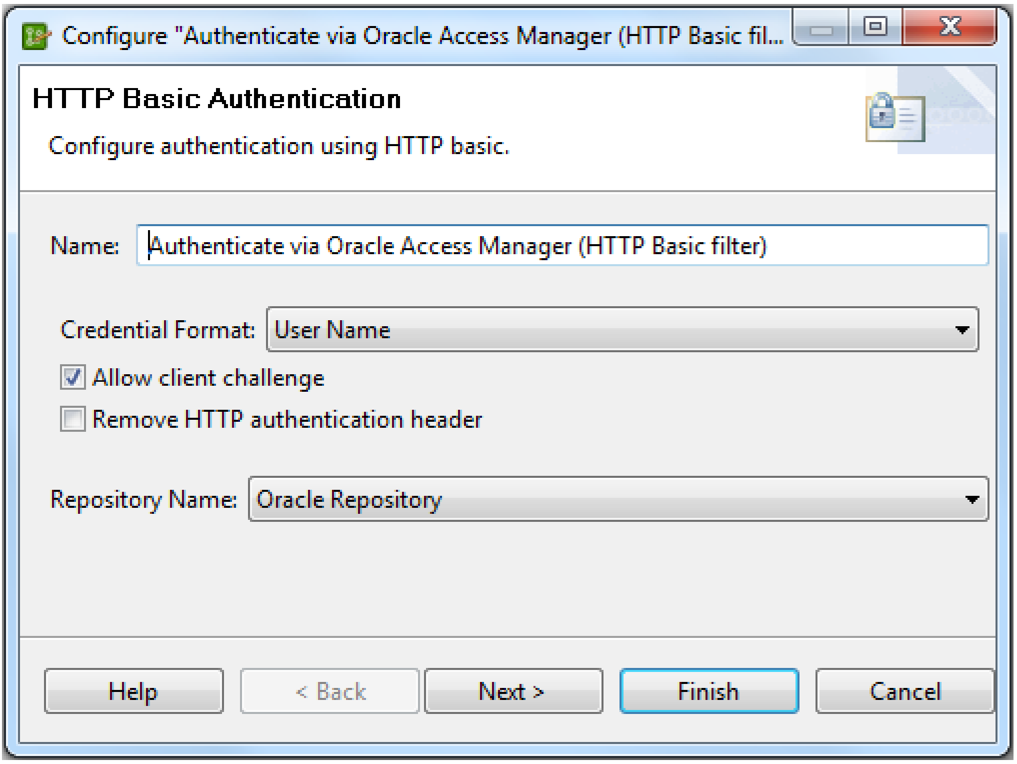
☑ Create SSO Token

Store SSO Token in attribute named: oracle.sso.token

☑ Add SSO Token to user attributes

Oblix installation directory: C:\Program Files\NetPoint\AccessServerSDK

4. Authentication to Oracle Access Manager is handled by either the **HTTP Basic** (used in this example) or **HTTP Digest** filter from the Authentication


Configure "Authenticate via Oracle Access Manager (HTTP Basic fil...

HTTP Basic Authentication

Configure authentication using HTTP basic.

Name: Authenticate via Oracle Access Manager (HTTP Basic filter)

Credential Format: User Name

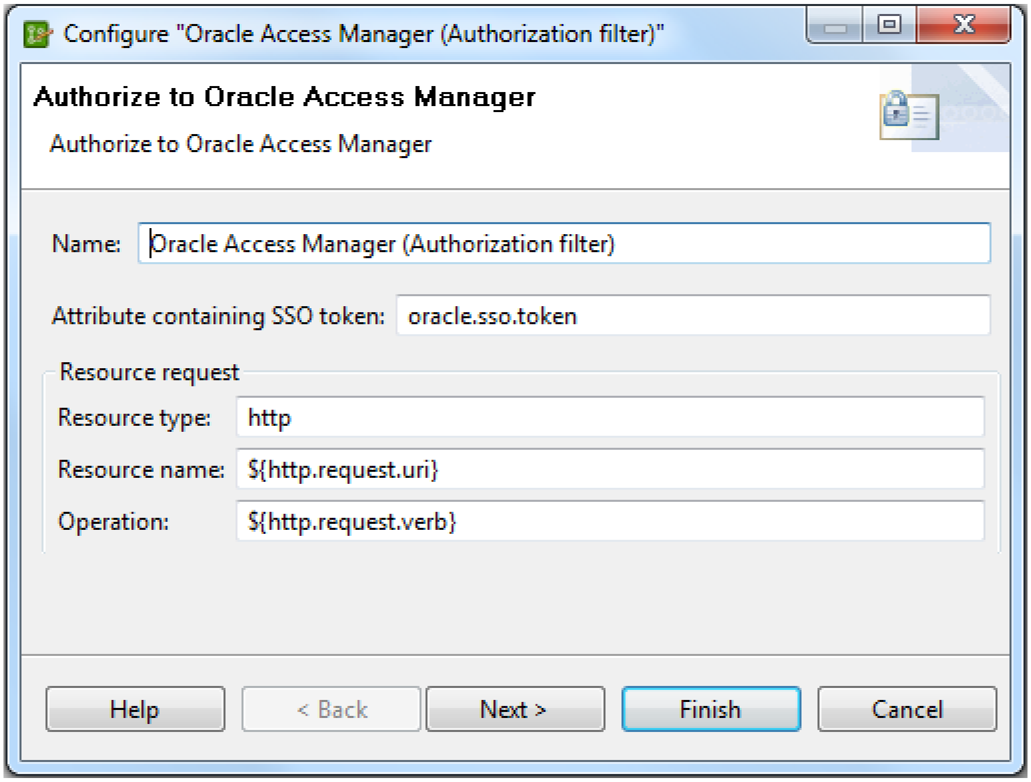
☑ Allow client challenge

□ Remove HTTP authentication header

Repository Name: Oracle Repository

category.

5. An **Oracle Access Manager Authorization** filter is used next to authorize an authenticated user for the current resource uri and http verb against Oracle Access Manager. The user must first have been authenticated to OAM via the **HTTP Basic** or **HTTP Digest** filter.


Configure "Oracle Access Manager (Authorization filter)"

Authorize to Oracle Access Manager

Authorize to Oracle Access Manager

Name: Oracle Access Manager (Authorization filter)

Attribute containing SSO token: oracle.sso.token

Resource request

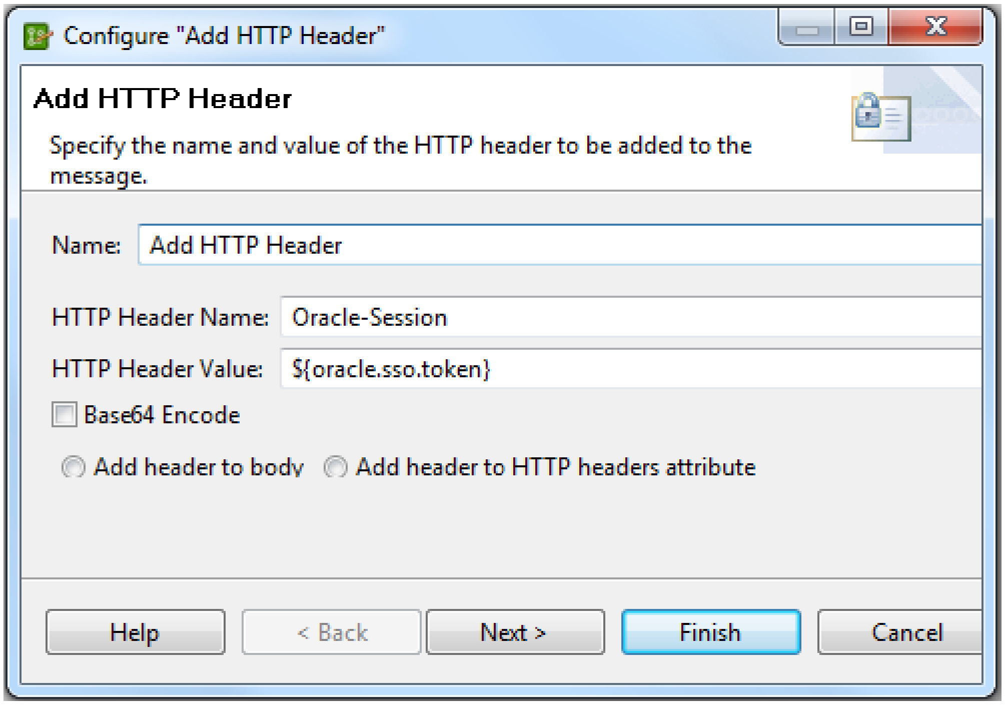
Resource type: http

Resource name: ${http.request.uri}

Operation: ${http.request.verb}

6. Once the user has been authenticated and authorized the circuit can be sent to the target Web Service.

7. A "Single Sign On" (SSO) token can be created and stored in a Fusion Gateway message attribute, so that it can be added later to the response that the client receives. The check box for creating an SSO token was ticked in the repository configuration. Add the session token to the HTTP headers with the **'Add HTTP Header'** filter from the **'Conversion'** group.



Configure "Add HTTP Header"

Add HTTP Header

Specify the name and value of the HTTP header to be added to the message.

Name: Add HTTP Header

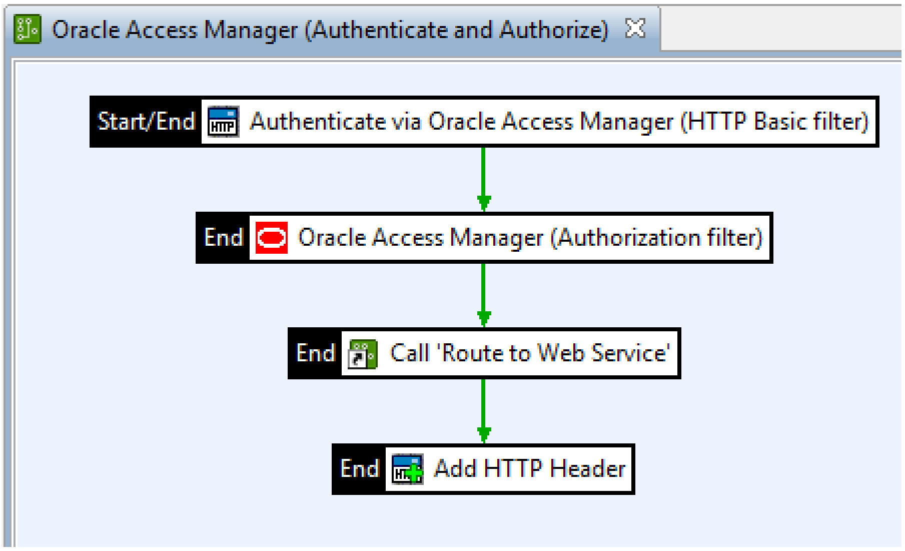
HTTP Header Name: Oracle-Session

HTTP Header Value: ${oracle.sso.token}

□ Base64 Encode

○ Add header to body ○ Add header to HTTP headers attribute

8. The full policy diagram will look like



Oracle Access Manager (Authenticate and Authorize)

Start/End Authenticate via Oracle Access Manager (HTTP Basic filter)

End Oracle Access Manager (Authorization filter)

End Call 'Route to Web Service'

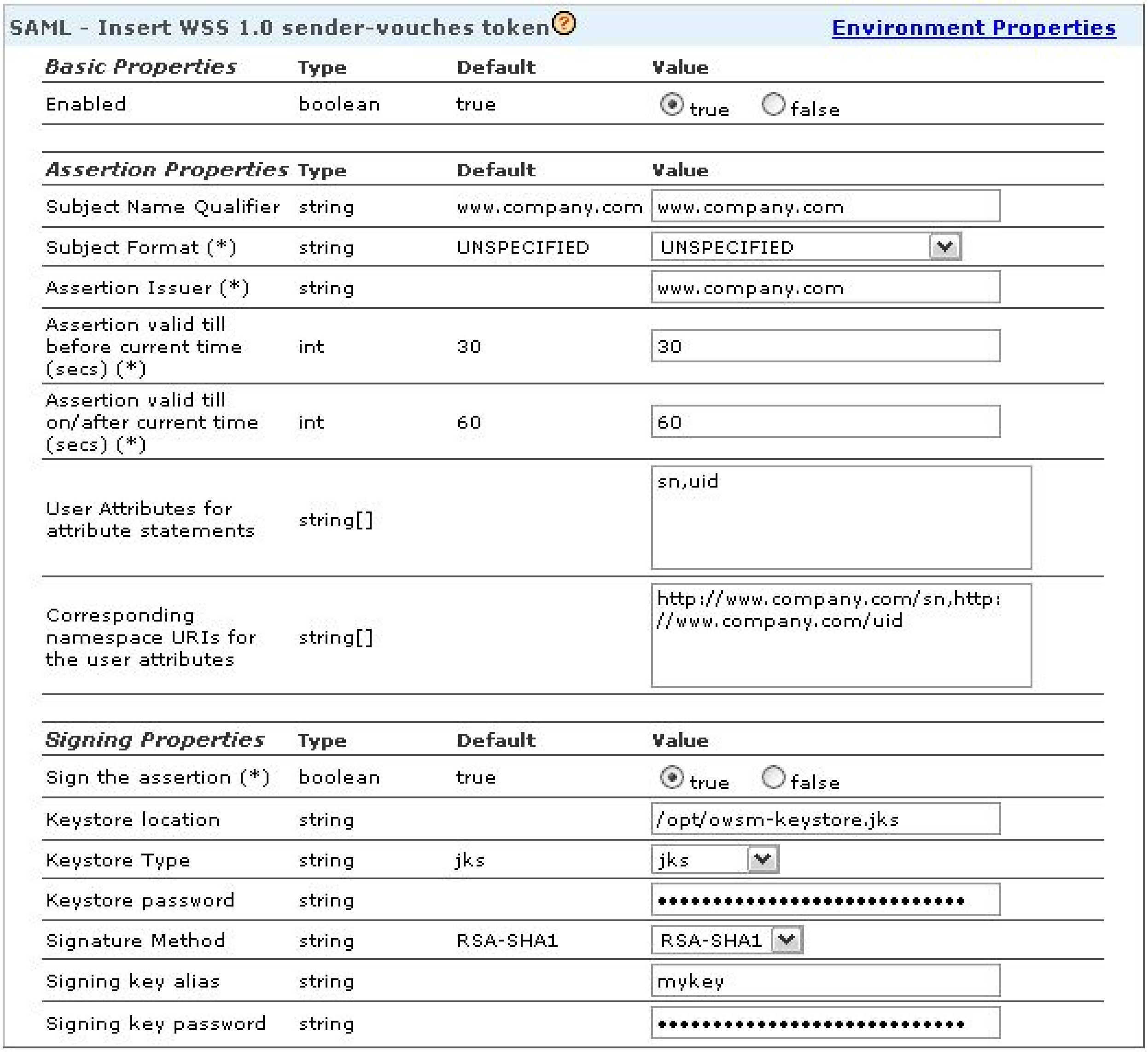
End Add HTTP Header

this.

Step: SAML - Insert WSS 1.0 sender-vouches token

Step to Insert SAML token as per WSS 1.0 token profile with Sender-Vouches confirmation method

OWSM Configuration:



SAML - Insert WSS 1.0 sender-vouches token

Environment Properties

Basic Properties	Type	Default	Value
Enabled	boolean	true	● true ○ false

Assertion Properties	Type	Default	Value
Subject Name Qualifier	string	www.company.com	www.company.com
Subject Format (*)	string	UNSPECIFIED	UNSPECIFIED
Assertion Issuer (*)	string		www.company.com
Assertion valid till before current time (secs) (*)	int	30	30
Assertion valid till on/after current time (secs) (*)	int	60	60
User Attributes for attribute statements	string[]		sn,uid
Corresponding namespace URIs for the user attributes	string[]		http://www.company.com/sn,http://www.company.com/uid

Signing Properties	Type	Default	Value
Sign the assertion (*)	boolean	true	● true ○ false
Keystore location	string		/opt/owsm-keystore.jks
Keystore Type	string	jks	jks
Keystore password	string		•••••••••••••••••••••••••••
Signature Method	string	RSA-SHA1	RSA-SHA1
Signing key alias	string		mykey
Signing key password	string		•••••••••••••••••••••••••••

Oracle Enterprise Gateway Configuration:

Refer to the **“Appendix”** section for how to add Certificates and Keys to the Gateway Certificate Store.

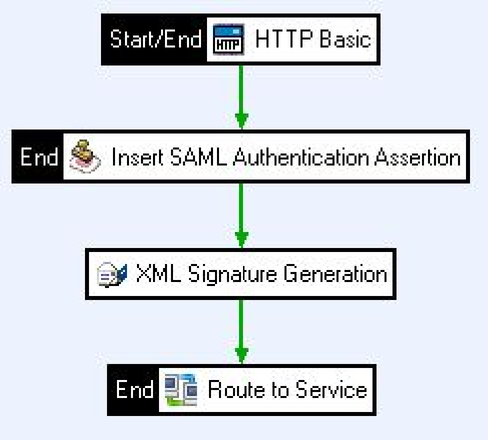
The **“Insert SAML Authentication Assertion”** filter requires user credentials to be passed to it from a predecessor filter in a policy. A **“HTTP Basic”** filter will be added at the start of the policy. Use the **“Sender Vouches”** confirmation method to assert that the Gateway is acting on behalf of the authenticated end-user. No other information relating to the context of the assertion is sent. It is recommended that both the assertion **and** the SOAP Body must be signed if this option is selected. These message parts can be signed by using the **“XML Signature Generation”** filter.

1. Add a **"HTTP Basic"** filter from the **"Authentication"** filter category. For **"Credential Format"** select **"User Name"** from the drop down list.
2. For **"Repository Name"** select the configured authentication repository. By default only the **"Local User Store"** will be available. Authentication repositories can be configured under the **"External Connections"** module in Policy Studio and can be a database, LDAP etc... For this demonstration the **"Local User Store"** will be selected.
3. Next add a **"Insert SAML Authentication Assertion"** filter located in the **"Authentication"** filter category.
4. Under the **"Assertion"** tab configure the expiry of the assertion token. Select **"Current Actor/Role"** only for the **"SOAP Actor/Role"** value.
5. Select the version of SAML assertion required. Here 1.1 is selected.
6. For **"Issuer Name"** select the certificate containing the Distinguished Name (DName) that will be used as the Issuer of the SAML assertion. This DName will appear in the SAML assertion as the value of the Issuer attribute of the <saml:Assertion> element.
7. Under the **"Confirmation Method"** tab select **"Sender Vouches"** from the drop down list that will assert that the Gateway is acting on behalf of the authenticated end-user. No other information relating to the context of the assertion is sent.
8. The rest of the options can be left default. For more information on the configuration options for this filter please refer to the Gateway documentation or click on the help tab in the filter itself.
9. It is recommended that both the assertion and the SOAP Body must be signed if the **"Sender Vouches"** option is selected. These message parts can be signed by using the "XML Signature Generation" filter.
10. Connect the **"HTTP Basic"** to the **"Insert SAML Authentication Assertion"** via a success path.
11. Add a **"XML Signature Generation"** filter from the **"Integrity"** filter category.
12. Under the **"Signing Key"** tab choose the type of key to be used to sign the request. It is possible to use either a symmetric or an asymmetric key to sign the message content. Select the appropriate radio button and configure the fields on the corresponding tab.
13. Select the location of the Private Key to be used to sign the request. This can either be a key stored in the Certificate Store or in an available message attribute if the key has already been used used by predecessor filters.
14. Once the key type has been configured, click on the **"Key Info"** tab where it can be configured how the <KeyInfo> block of the generated XML Signature will appear.
15. To achieve the same structure as that of the signed OWSM assertion select the **"Security Token Reference"** radio button, select **"X509V3"** from the drop down field. Then select the **"Embed"** radio button and tick the **"Include TokenType"** checkbox.
16. The **"What to Sign"** tab is used to configure to identify the parts of the message that must be signed. Each signed part will be referenced from within the generated XML Signature. Here it is recommended that the assertion and SOAP body be signed.
17. Select the **"Use WS-Ids"** option which uses wsu:Id attribute to reference the signed data. Also ensure that the ensure that **"Use SAML Ids for SAML Elements"** and **"Add and Dereference Security Token Reference for SAML"** is checked.
18. Select the part of the request that needs to be signed from either the **"Nodes Location"**, **"XPath"**, **"XPath Predicates"** or **"Message Attribute"** tab. The assertion and SOAP body is selectable under the **"Nodes Location"** tab.
19. Under the "Advanced/Options" tab the **"Insert CarriedKeyName for EncryptedKey"** can be

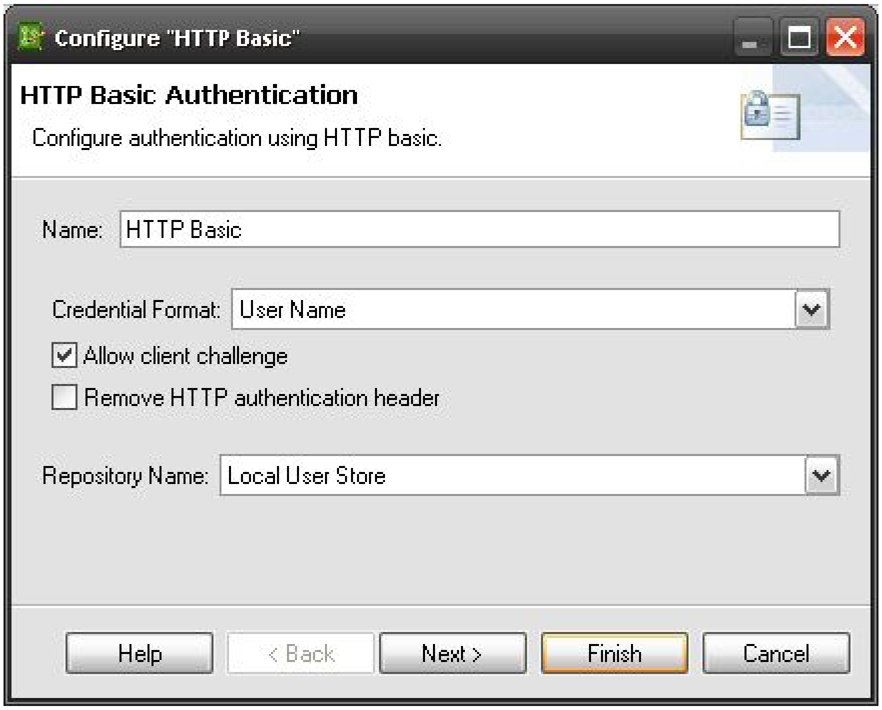
unchecked. The rest of the options can be left default. For more information on the configuration options for this filter please refer to the Gateway documentation or click on the help tab in the filter itself.

20. Connect the **"Insert SAML Authentication Assertion"** to the **"XML Signature Generation"** filter via a success path.

The sample policy:



The HTTP Basic Filter:



HTTP Basic Authentication

Configure authentication using HTTP basic.

Name: HTTP Basic

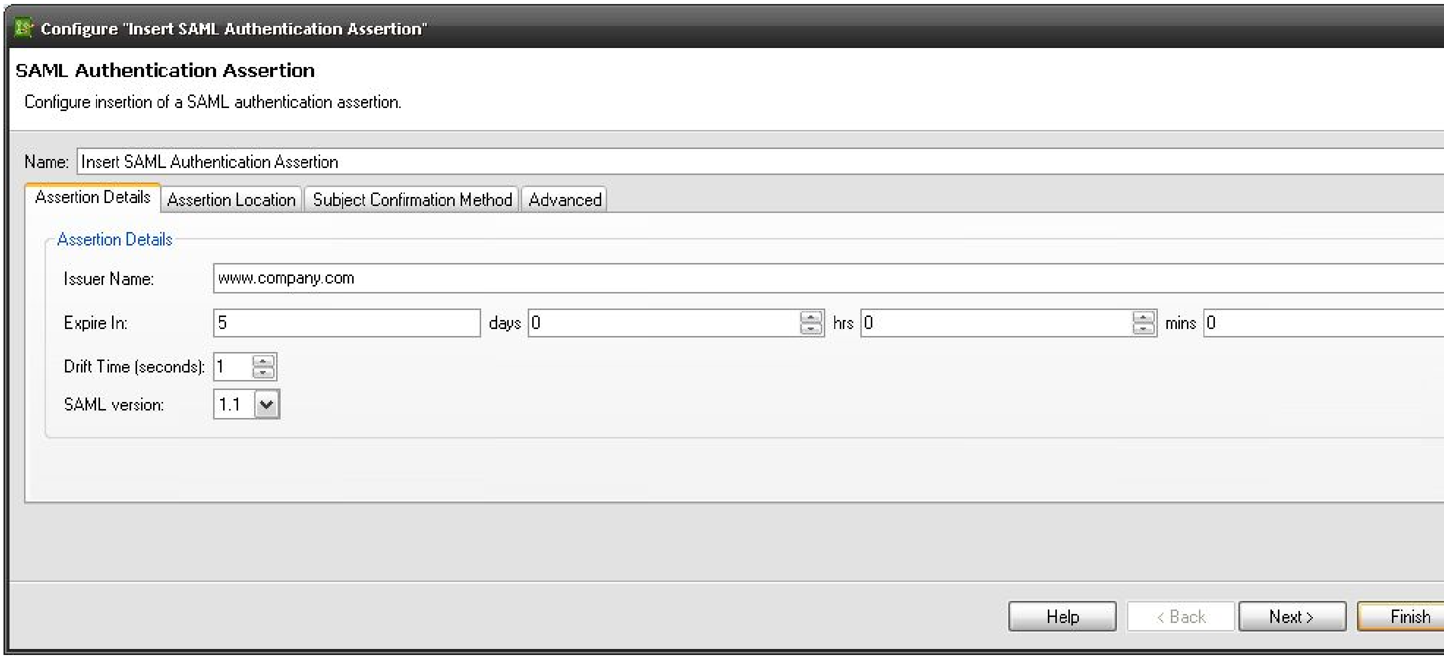
Credential Format: User Name

☑ Allow client challenge

□ Remove HTTP authentication header

Repository Name: Local User Store

The Insert SAML Authentication Assertion filter:



Configure "Insert SAML Authentication Assertion"

SAML Authentication Assertion

Configure insertion of a SAML authentication assertion.

Name: Insert SAML Authentication Assertion

Assertion Details

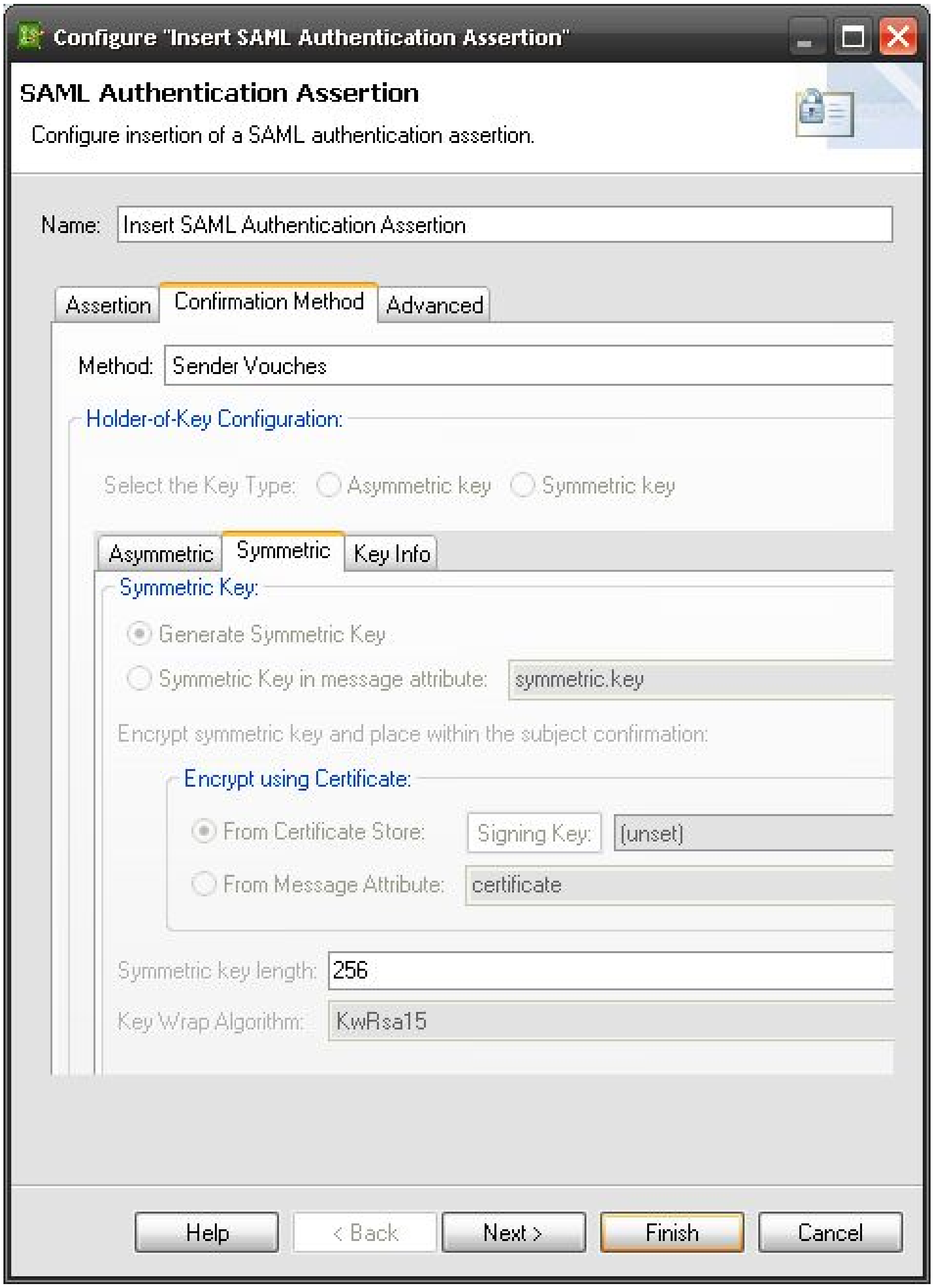
Issuer Name: www.company.com

Expire In: 5 days 0 hrs 0 mins 0

Drift Time (seconds): 1

SAML version: 1.1

The Confirmation Method tab:



Configure "Insert SAML Authentication Assertion"

SAML Authentication Assertion

Configure insertion of a SAML authentication assertion.

Name: Insert SAML Authentication Assertion

Method: Sender Vouches

Holder-of-Key Configuration:

Select the Key Type: ○ Asymmetric key ○ Symmetric key

Symmetric Key:

● Generate Symmetric Key

○ Symmetric Key in message attribute: symmetric.key

Encrypt symmetric key and place within the subject confirmation:

Encrypt using Certificate:

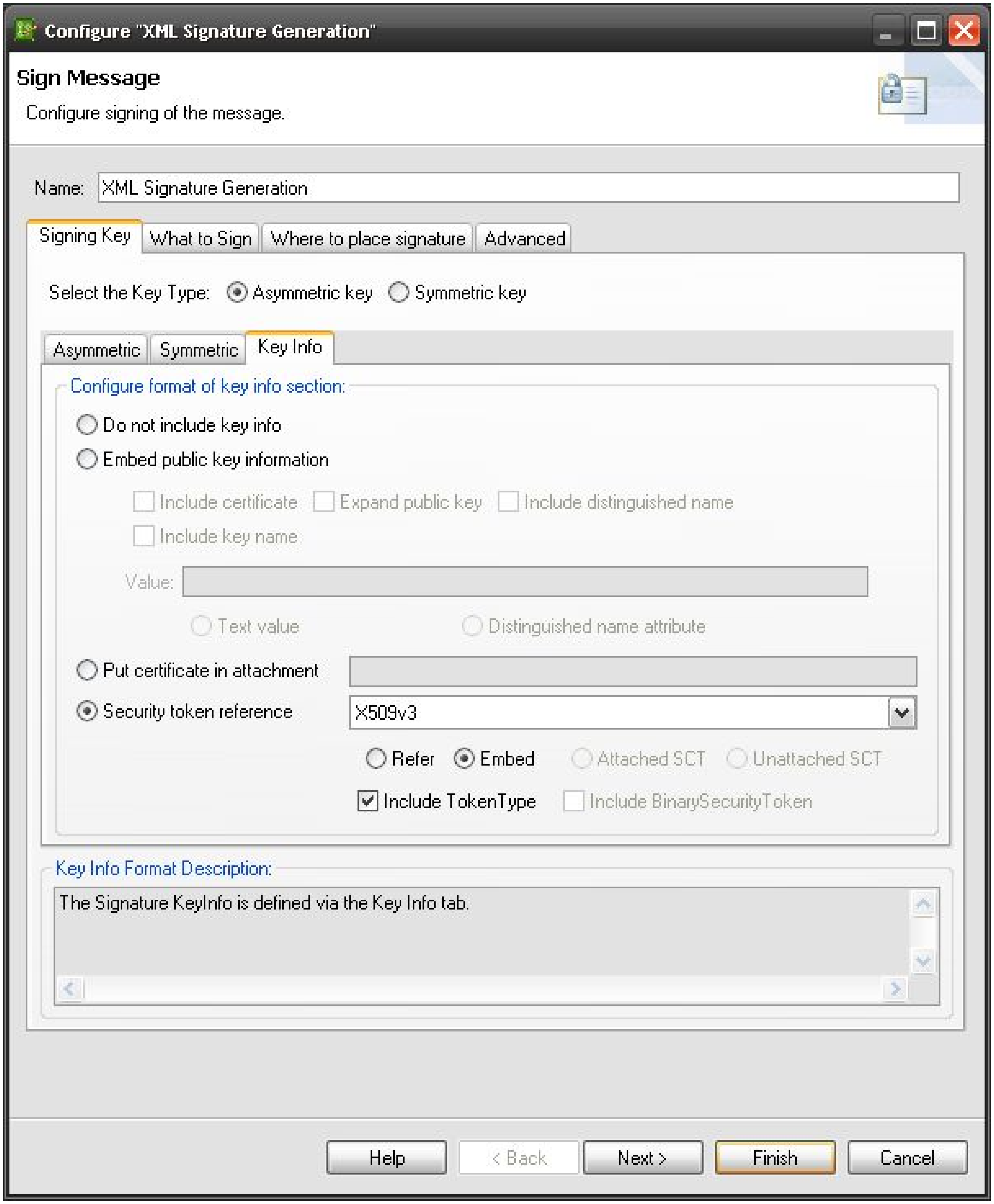
● From Certificate Store: Signing Key: [unset]

○ From Message Attribute: certificate

Symmetric key length: 256

Key Wrap Algorithm: KwRsa15

The Key Info tab in the XML Signature Generation filter:



Sign Message

Configure signing of the message.

Name: XML Signature Generation

Select the Key Type: ● Asymmetric key ○ Symmetric key

Configure format of key info section:

○ Do not include key info

○ Embed public key information

□ Include certificate □ Expand public key □ Include distinguished name

□ Include key name

Value:

○ Text value ○ Distinguished name attribute

○ Put certificate in attachment

● Security token reference X509v3

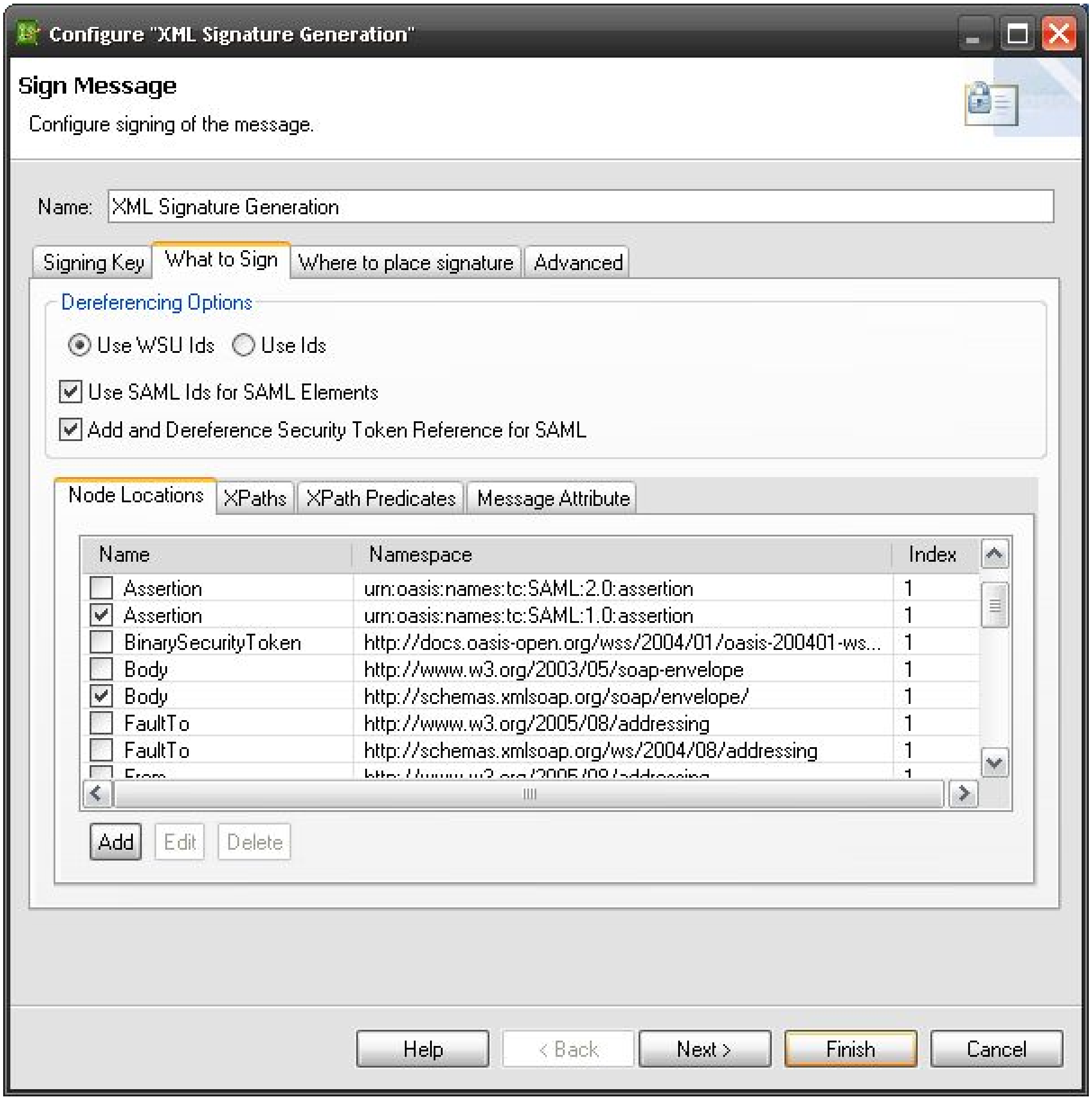
○ Refer ● Embed ○ Attached SCT ○ Unattached SCT

☑ Include TokenType □ Include BinarySecurityToken

Key Info Format Description:

The Signature KeyInfo is defined via the Key Info tab.

The What to Sign tab in the XML Signature Generation filter:



Sign Message

Configure signing of the message.

Name: XML Signature Generation

Dereferencing Options

● Use WSU Ids ○ Use Ids

☑ Use SAML Ids for SAML Elements

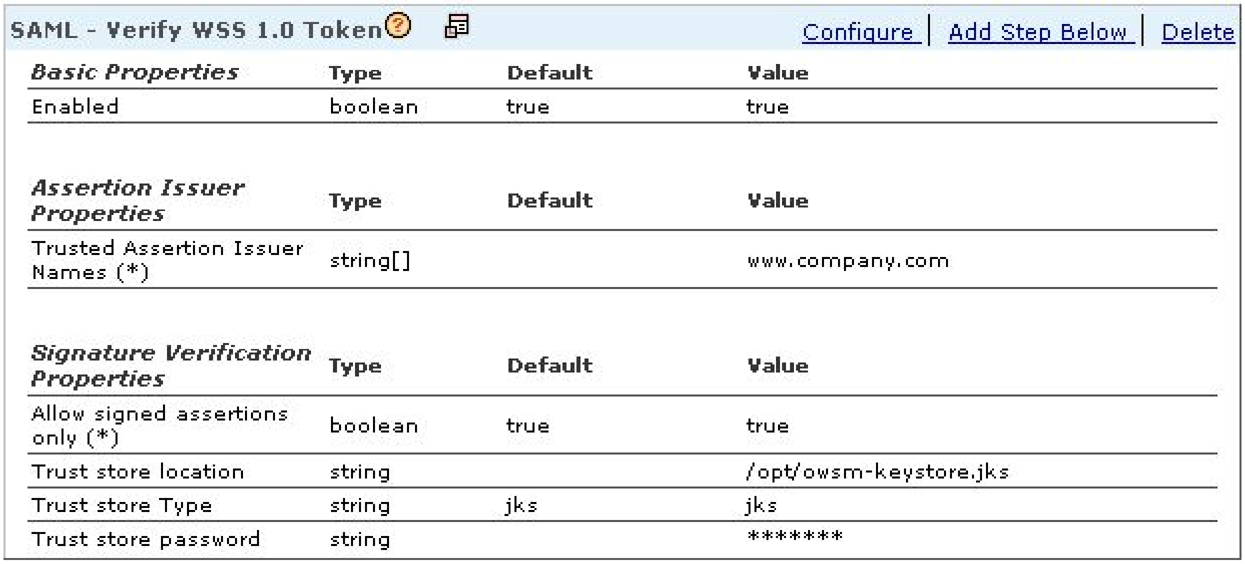
☑ Add and Dereference Security Token Reference for SAML

	Name	Namespace	Index
□	Assertion	urn:oasis:names:tc:SAML:2.0:assertion	1
☑	Assertion	urn:oasis:names:tc:SAML:1.0:assertion	1
□	BinarySecurityToken	http://docs.oasis-open.org/wss/2004/01/oasis-200401-ws...	1
□	Body	http://www.w3.org/2003/05/soap-envelope	1
☑	Body	http://schemas.xmlsoap.org/soap/envelope/	1
□	FaultTo	http://www.w3.org/2005/08/addressing	1
□	FaultTo	http://schemas.xmlsoap.org/ws/2004/08/addressing	1
□	From	http://www.w3.org/2005/08/addressing	1

Step: SAML - Verify WSS 1.0 Token

Verify SAML tokens as per WSS SAML token profile 1.0

OWSM Configuration:



SAML - Verify WSS 1.0 Token

Basic Properties	Type	Default	Value
Enabled	boolean	true	true

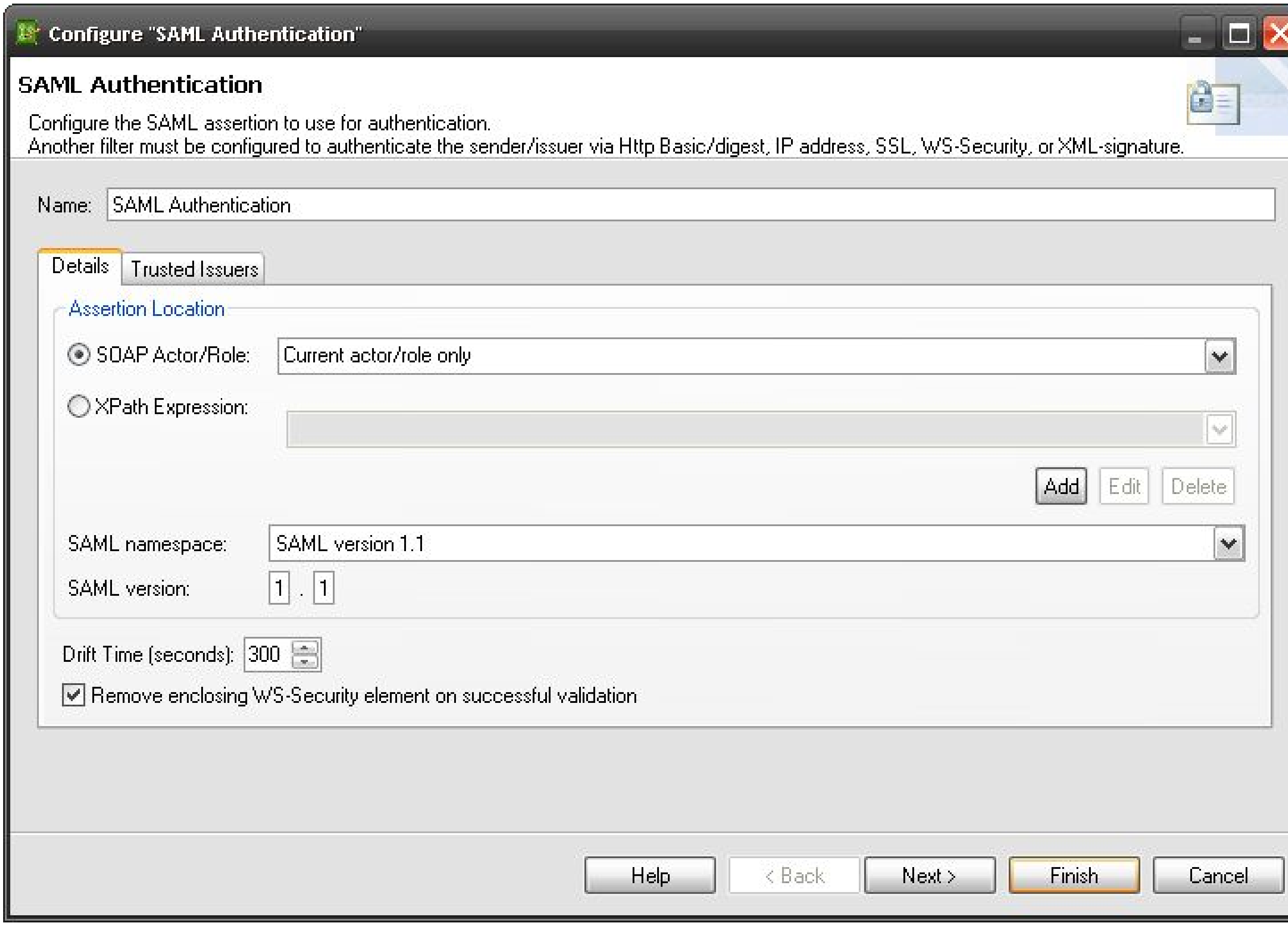
Assertion Issuer Properties	Type	Default	Value
Trusted Assertion Issuer Names (*)	string[]		www.company.com

Signature Verification Properties	Type	Default	Value
Allow signed assertions only (*)	boolean	true	true
Trust store location	string		/opt/owsm-keystore.jks
Trust store Type	string	jks	jks
Trust store password	string		*******

Oracle Enterprise Gateway Configuration:

1. Add a **"SAML Authentication"** filter from the **"Authentication"** filter category.
2. Under the **"Details"** tab, **"Assertion Location"** section, choose the location of the SAML assertion by selecting the **"SOAP Actor/Role"** option or the **"XPath"** option.
3. Enforcement of the SAML namespace (SAML version) can be configured if required.
4. The rest options can be left default. For more information on the configuration options for this filter please refer to the Gateway documentation or click on the help tab in the filter itself.
5. Under the **"Trusted Issuer"** tab the issuer of the SAML assertion that requires verification needs to be entered or selected. To verify the request that has been created by OWSM for test purposes, enter www.company.com as the trusted issuer value.

The SAML Authentication verification filter:



Configure "SAML Authentication"

SAML Authentication

Configure the SAML assertion to use for authentication.
Another filter must be configured to authenticate the sender/issuer via Http Basic/digest, IP address, SSL, WS-Security, or XML-signature.

Name: SAML Authentication

Assertion Location

● SOAP Actor/Role: Current actor/role only

○ XPath Expression:

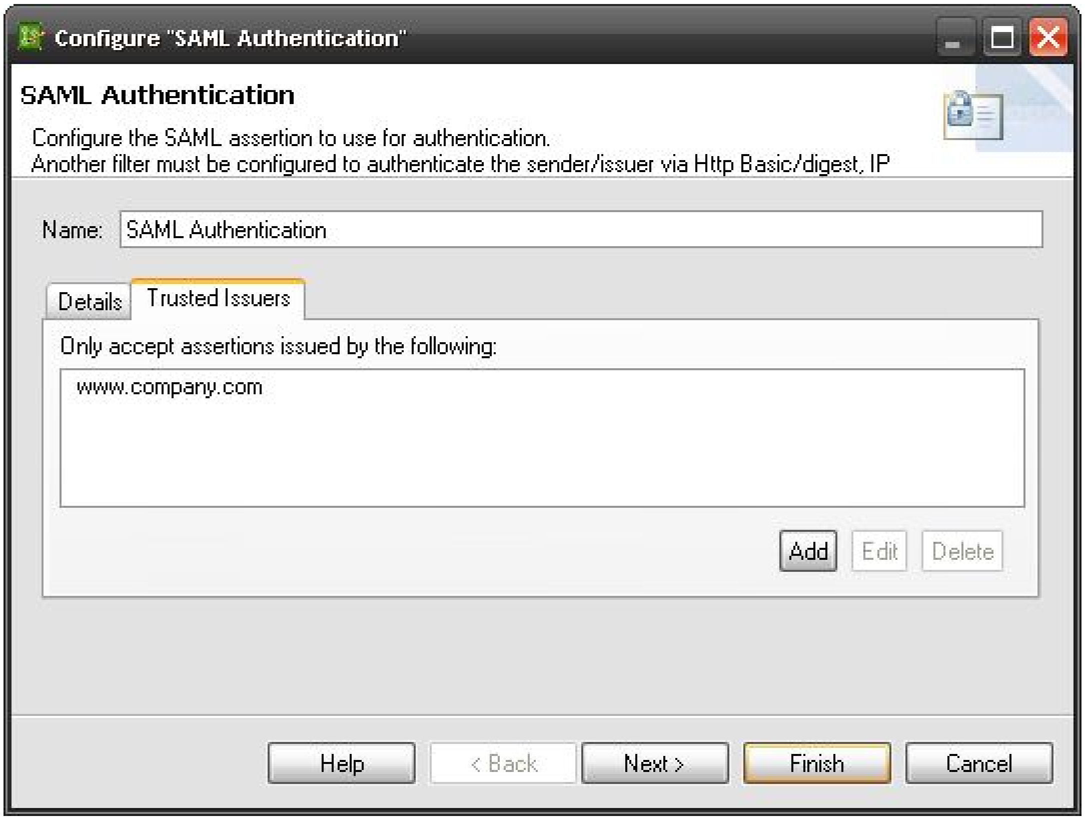
SAML namespace: SAML version 1.1

SAML version: 1 . 1

Drift Time (seconds): 300

☑ Remove enclosing WS-Security element on successful validation

The Trusted Issuer tab configuration:



Configure "SAML Authentication"

SAML Authentication

Configure the SAML assertion to use for authentication.
Another filter must be configured to authenticate the sender/issuer via Http Basic/digest, IP

Name: SAML Authentication

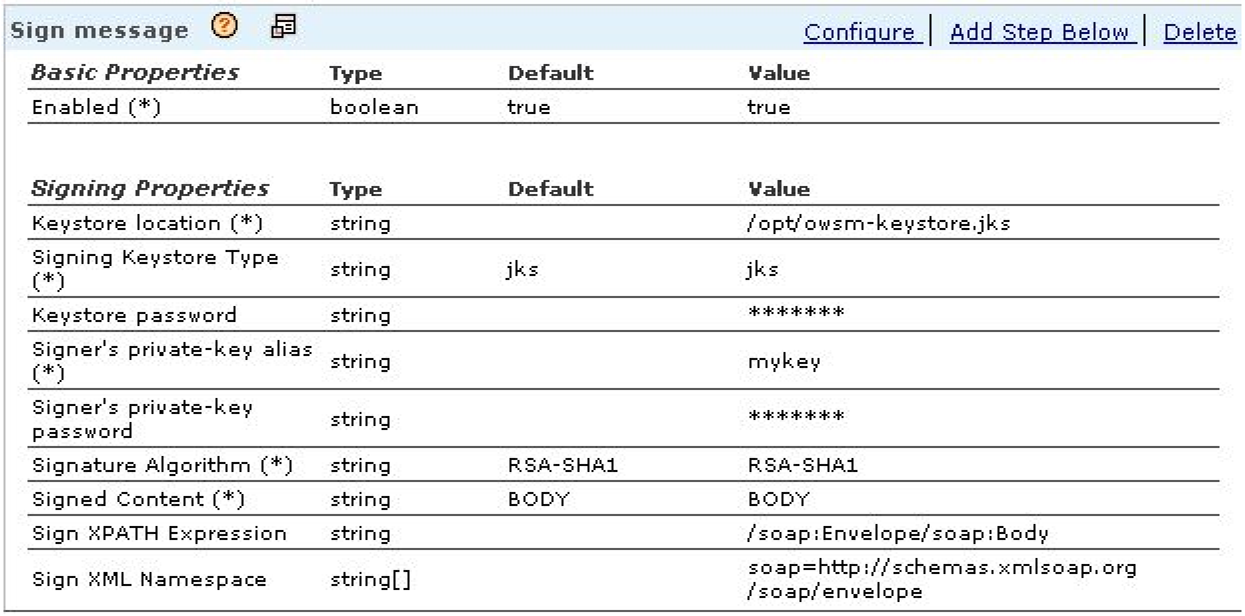
Only accept assertions issued by the following:

www.company.com

Step: Sign message

Digitally Sign the message

OWSM Configuration:



Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	true

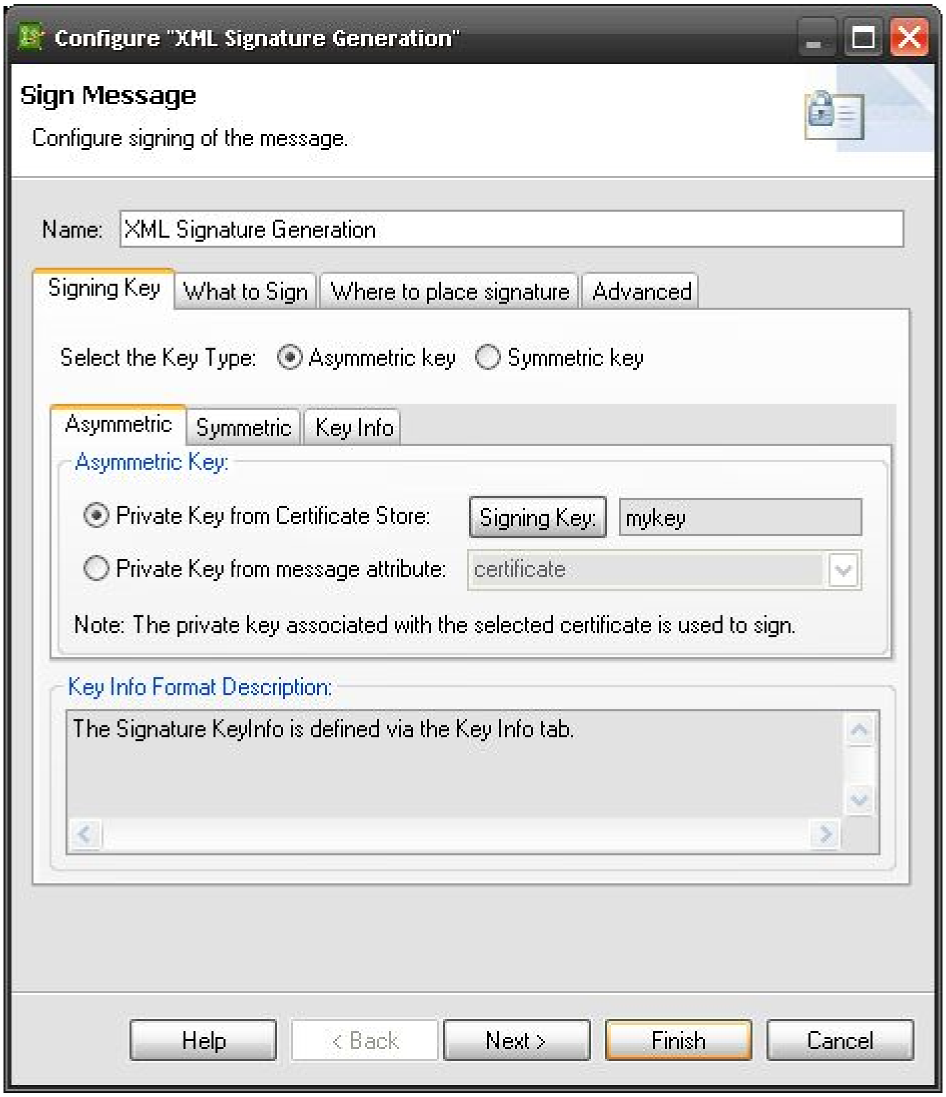
Signing Properties	Type	Default	Value
Keystore location (*)	string		/opt/owsm-keystore.jks
Signing Keystore Type (*)	string	jks	jks
Keystore password	string		*******
Signer's private-key alias (*)	string		mykey
Signer's private-key password	string		*******
Signature Algorithm (*)	string	RSA-SHA1	RSA-SHA1
Signed Content (*)	string	BODY	BODY
Sign XPATH Expression	string		/soap:Envelope/soap:Body
Sign XML Namespace	string[]		soap=http://schemas.xmlsoap.org/soap/envelope

Oracle Enterprise Gateway Configuration:
Refer to the **"Appendix"** section for how to add Certificates and Keys to the Gateway Certificate Store.

Messages can be signed in the Gateway using the **"XML Signature Generation"** filter. For configuring the settings to be the same as that configured in OWSM, please follow the steps below.

1. Add a **"XML Signature Generation"** filter located in the **"Integrity"** filter category.
2. Under the **"Signing Key"** tab choose the **"Asymmetric Key"** option.
3. Select the location of the Private Key to be used to sign the request. This can either be a key stored in the Certificate Store or in an available message attribute if the key has already been used used by predecessor filters.
4. Once the key type has been configured, click on the **"Key Info"** tab where it can be configured how the <KeyInfo> block of the generated XML Signature will appear.
5. To achieve the same structure as that of a signed OWSM request, select the **"Security Token Reference"** radio button, select **"X509v3"** from the drop down field. Then select the **"Embed"** radio button and tick the **"Include TokenType"** checkbox.
6. The "**What to Sign"** tab is used to configure to identify the parts of the message that must be signed. Each signed part will be referenced from within the generated XML Signature.
7. Select either the **"Use WS-Ids"** option which uses wsu:Id attribute to reference the signed data or the **"Use Ids"** option that uses generic Id's that references the signed data.
8. Select the part of the request that needs to be signed from either the **"Nodes Location"**, **"XPath"**, **"XPath Predicates"** or **"Message Attribute"** tab.
9. Under the **"Where to place Signature"**, select where the signature is to be placed. As per OWSM, the option to select is the **"In the WS-Security Element for SOAP Actor/Role"**. Then select **"Current Actor/Role only"** from the drop down field.
10. The **"Advanced"** tab contains additional but not mandatory options for example inserting a timestamp, select a particular algorithm for signing if required and other options. Ensure that **"ShaRsa1"** for the **"Algorithm Method"** and **"Sha1"** is selected for the **"Digest Algorithm"** option under the **"Algorithm Suite"** tab. Under the **"Options"** tab the **"Insert CarriedKeyName for EncryptedKey"** can be unchecked.
11. Please refer to Gateway documentation for more information on all the options available in the **"Sign Message"** filter.

The Signing Key tab settings:



Configure "XML Signature Generation"

Sign Message

Configure signing of the message.

Name: XML Signature Generation

Select the Key Type: ● Asymmetric key ○ Symmetric key

Asymmetric Key:

● Private Key from Certificate Store: Signing Key: mykey

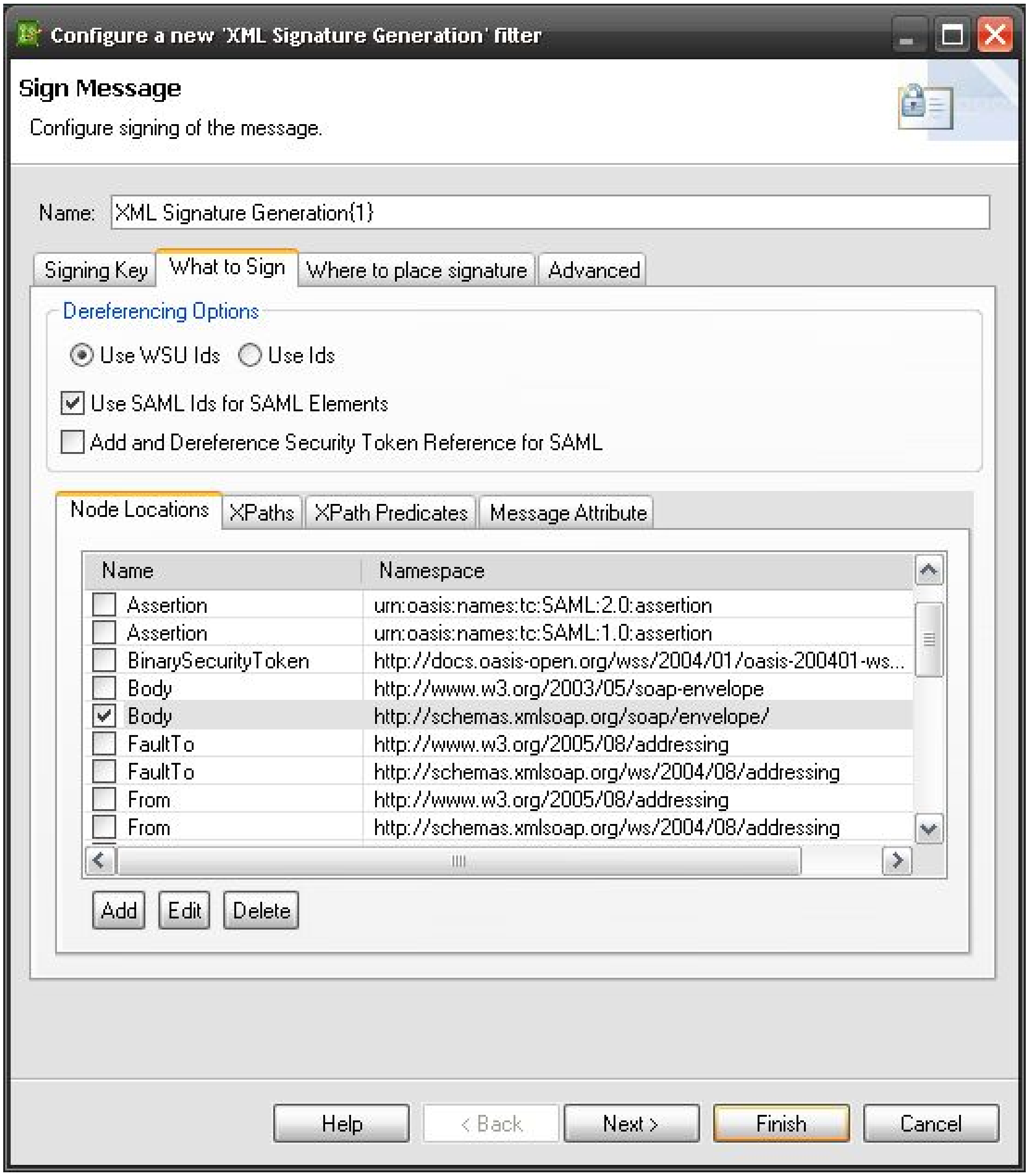
○ Private Key from message attribute: certificate

Note: The private key associated with the selected certificate is used to sign.

Key Info Format Description:

The Signature KeyInfo is defined via the Key Info tab.

The What to Sign tab settings:



Sign Message

Configure signing of the message.

Name: XML Signature Generation{1}

Dereferencing Options

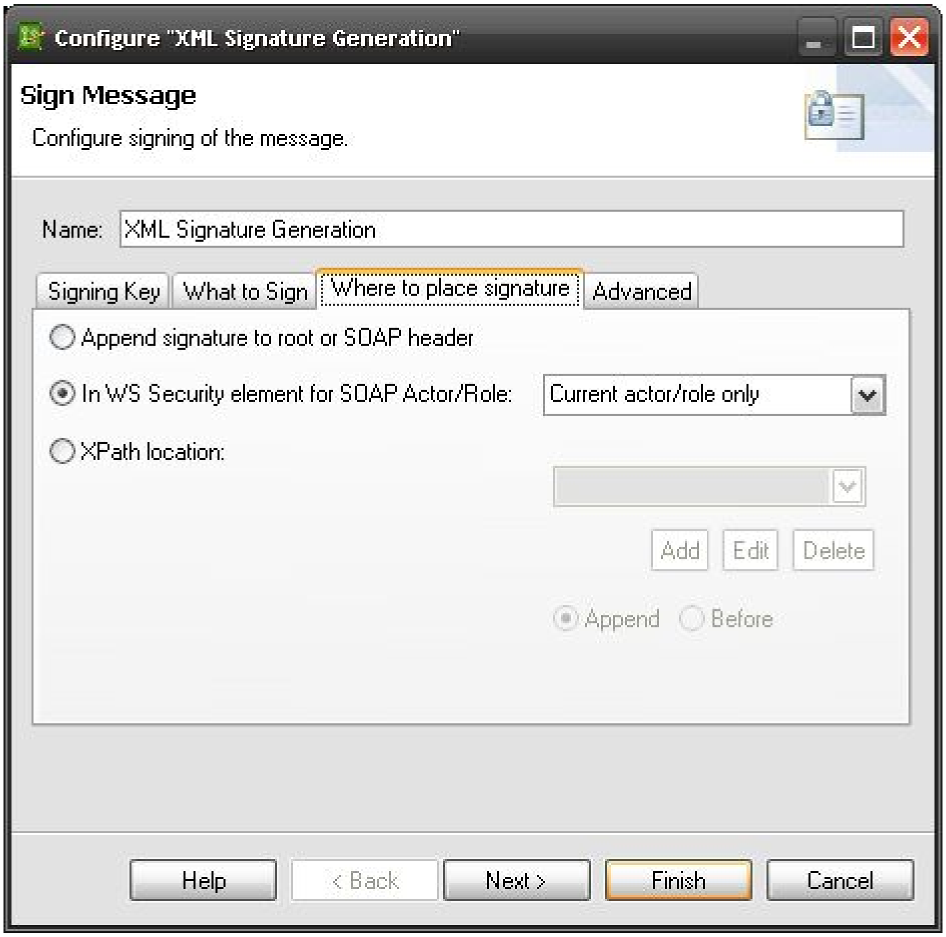
● Use WSU Ids ○ Use Ids

☑ Use SAML Ids for SAML Elements

□ Add and Dereference Security Token Reference for SAML

Name	Namespace
□ Assertion	urn:oasis:names:tc:SAML:2.0:assertion
□ Assertion	urn:oasis:names:tc:SAML:1.0:assertion
□ BinarySecurityToken	http://docs.oasis-open.org/wss/2004/01/oasis-200401-ws...
□ Body	http://www.w3.org/2003/05/soap-envelope
☑ Body	http://schemas.xmlsoap.org/soap/envelope/
□ FaultTo	http://www.w3.org/2005/08/addressing
□ FaultTo	http://schemas.xmlsoap.org/ws/2004/08/addressing
□ From	http://www.w3.org/2005/08/addressing
□ From	http://schemas.xmlsoap.org/ws/2004/08/addressing

The Where to Place Signature tab settings:



Configure "XML Signature Generation"

Sign Message

Configure signing of the message.

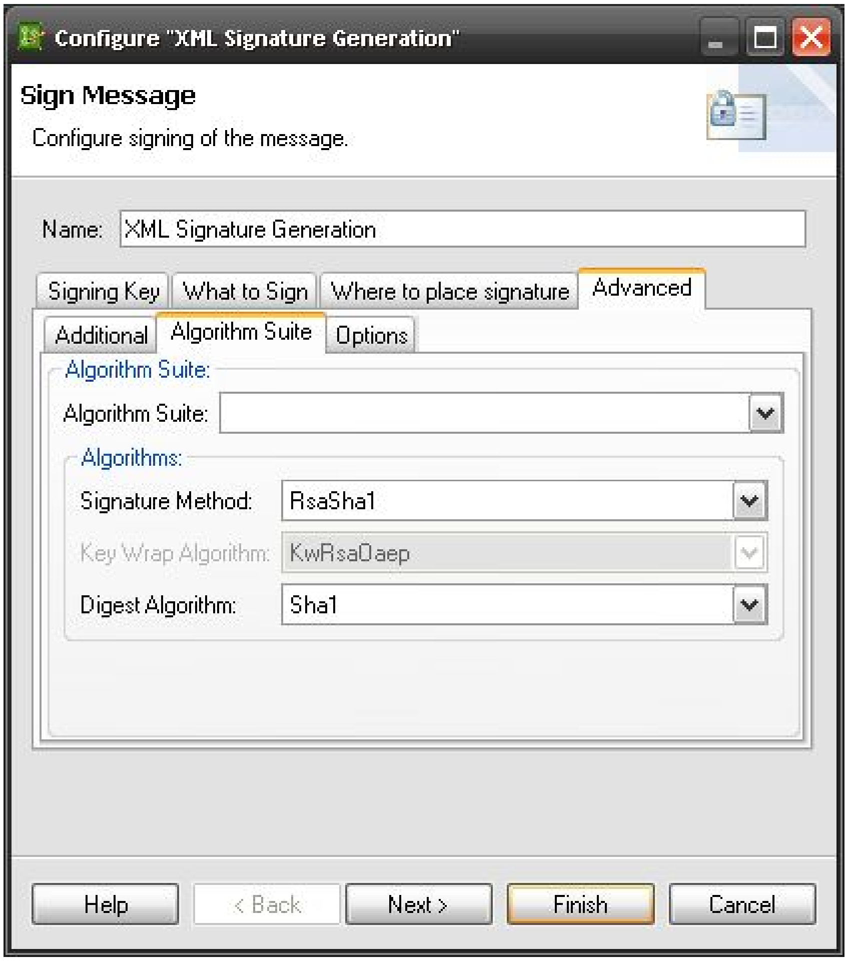
Name: XML Signature Generation

○ Append signature to root or SOAP header

● In WS Security element for SOAP Actor/Role: Current actor/role only

○ XPath location:

The Advanced Tab settings:



Sign Message

Configure signing of the message.

Name: XML Signature Generation

Algorithm Suite:

Algorithm Suite:

Algorithms:

Signature Method: RsaSha1

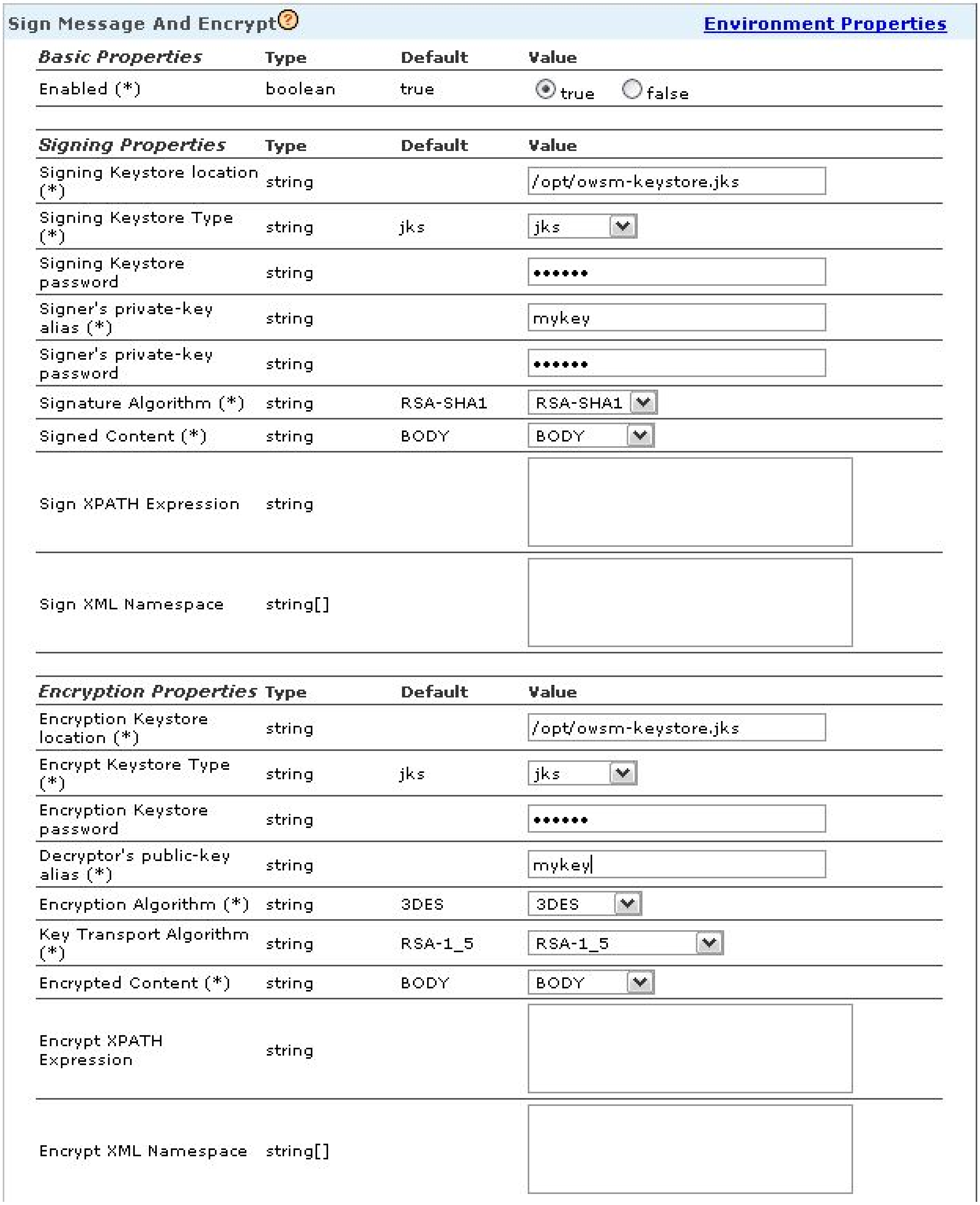
Key Wrap Algorithm: KwRsaOaep

Digest Algorithm: Sha1

Step: Sign Message And Encrypt

XML Signature and Encryption

OWSM Configuration:



Sign Message And Encrypt ? Environment Properties

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false

Signing Properties	Type	Default	Value
Signing Keystore location (*)	string		/opt/owsm-keystore.jks
Signing Keystore Type (*)	string	jks	jks
Signing Keystore password	string		••••••
Signer's private-key alias (*)	string		mykey
Signer's private-key password	string		••••••
Signature Algorithm (*)	string	RSA-SHA1	RSA-SHA1
Signed Content (*)	string	BODY	BODY
Sign XPATH Expression	string		
Sign XML Namespace	string[]		

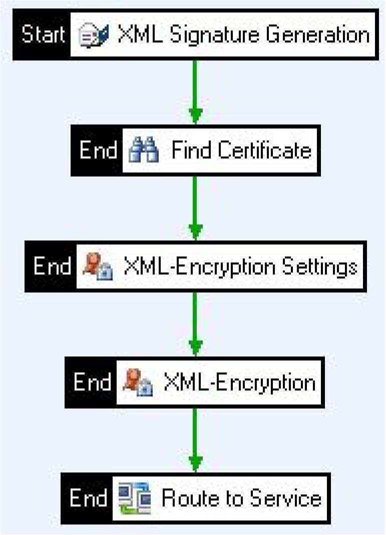
Encryption Properties	Type	Default	Value
Encryption Keystore location (*)	string		/opt/owsm-keystore.jks
Encrypt Keystore Type (*)	string	jks	jks
Encryption Keystore password	string		••••••
Decryptor's public-key alias (*)	string		mykey
Encryption Algorithm (*)	string	3DES	3DES
Key Transport Algorithm (*)	string	RSA-1_5	RSA-1_5
Encrypted Content (*)	string	BODY	BODY
Encrypt XPATH Expression	string		
Encrypt XML Namespace	string[]		

Oracle Enterprise Gateway Configuration:

To sign and encrypt a request in the Gateway perform the XML Sign and XML Encrypt steps in one policy. Please see the XML Sign and XML Encrypt sections here for configuration of the relevant options.

NOTE: It is recommended to use different keys for signing and encryption.

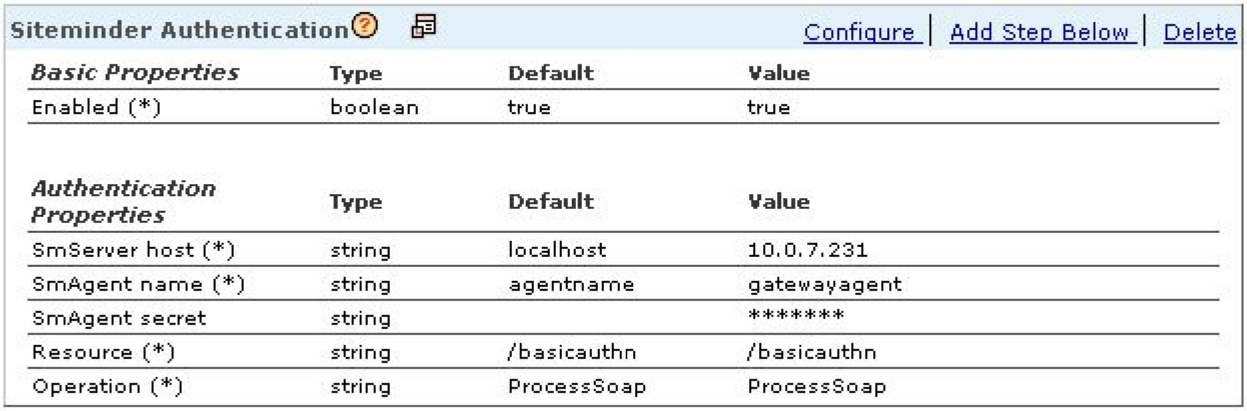
A sample policy where the request is signed and encrypted.



Step: Siteminder Authentication

Authentication via SiteMinder

OWSM Configuration:



Siteminder Authentication

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	true

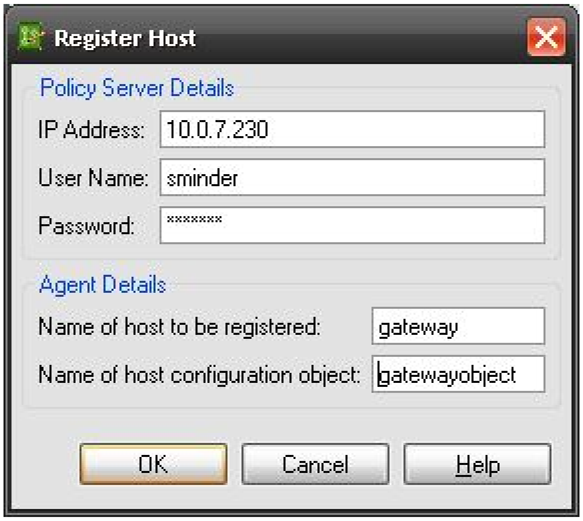
Authentication Properties	Type	Default	Value
SmServer host (*)	string	localhost	10.0.7.231
SmAgent name (*)	string	agentname	gatewayagent
SmAgent secret	string		*******
Resource (*)	string	/basicauthn	/basicauthn
Operation (*)	string	ProcessSoap	ProcessSoap

Oracle Enterprise Gateway Configuration:
Register a Siteminder Agent in the Gateway:
The Siteminder agent needs to be registered under the **"Siteminder/SOA Security Manager Connections"** object under the **"External Connections"** module.

1. Click on the **"External Connections"** module.
2. Right click on **"SiteMinder/SOA Security Manager Connections"** and select **"Add SiteMinder Connection".**
3. Click on the **"Register"** button.

4. Enter the **"IP Address"** of the SiteMinder Policy Server.
5. Enter the **"User Name"** and **"Password"** of the account used to connect to the SiteMinder Policy Server.
6. Under **"Agent Details"** enter a value for the **"Name of the host to be registered".** This can be **ANY** descriptive value and should not already exist under the **"Trusted Host"** list on the SiteMinder Policy Server.
7. Enter the **"Name of the host configuration object"** as is configured in SiteMinder.

The Siteminder Agent configuration window:



Register Host

Policy Server Details

IP Address: 10.0.7.230

User Name: sminder

Password: *******

Agent Details

Name of host to be registered: gateway

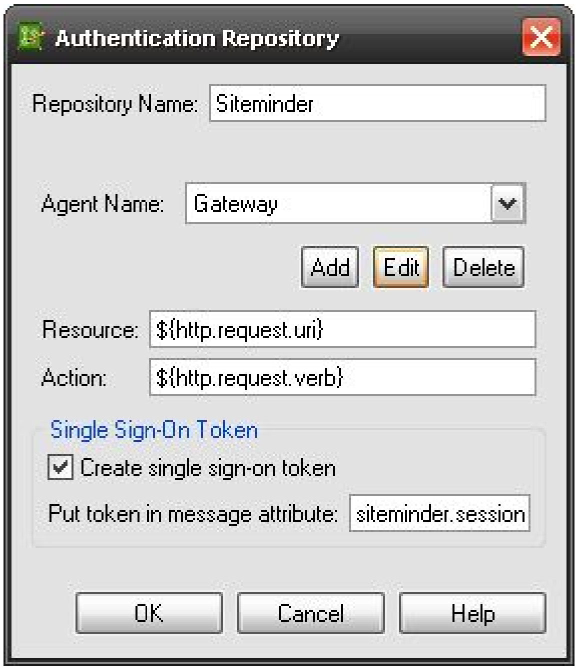
Name of host configuration object: gatewayobject

Add a Siteminder Authentication Repository:

Once the Siteminder Agent has been registered, expand the **"Authentication Repository Profiles"** object under the **"External Connections"** tree.

1. Right click on **"CA SiteMinder Repositories"** and select **"Add a new Repository".**
2. Enter a name for the repository. For this guide **"SiteMinder"** is used.
3. Select the configured agent in the drop down menu for the **"Agent Name"** value.
4. The rest of the options can be left default. Click on the **"Help"** button if more details are required for the configuration options here.

The Siteminder Authentication Repository configuration window:



Repository Name: Siteminder

Agent Name: Gateway

Resource: ${http.request.uri}

Action: ${http.request.verb}

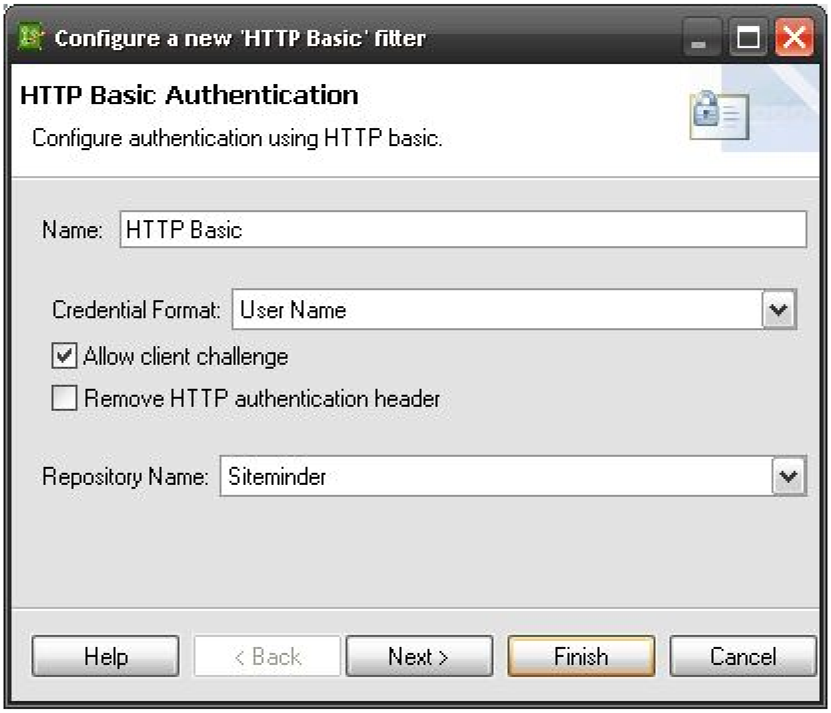
Single Sign-On Token

☑ Create single sign-on token

Put token in message attribute: siteminder.session

The Siteminder Authentication Repository is now available for any filter that can connect to a configured Repository for example the **"HTTP Basic"** filter where available repositories where users are stored can be selected from a drop down menu. The "HTTP Basic" filter is available in the **"Authentication"** filter group.

The HTTP Basic filter configured to connect to the Siteminder Repository.



HTTP Basic Authentication

Configure authentication using HTTP basic.

Name: HTTP Basic

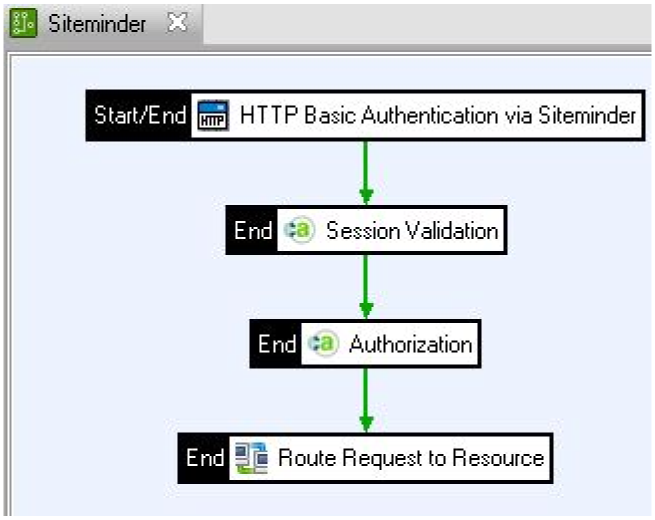
Credential Format: User Name

☑ Allow client challenge

☐ Remove HTTP authentication header

Repository Name: Siteminder

Example of the HTTP Basic filter for Authentication via Siteminder in a policy:

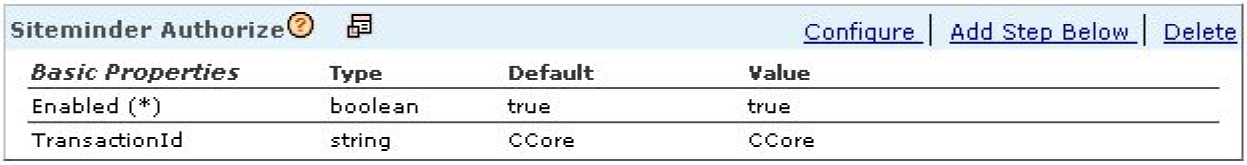

Siteminder
Start/End
HTTP Basic Authentication via Siteminder
End
Session Validation
End
Authorization
End
Route Request to Resource

For more information on how the Gateway integrates with SiteMinder please refer to the Fusion Gateway documentation as well as the Fusion Gateway Integration with Siteminder Guide.

Step: Siteminder Authorize

SiteMinder Authorization to be used after SiteMinder Authentication Step

OWSM Configuration:



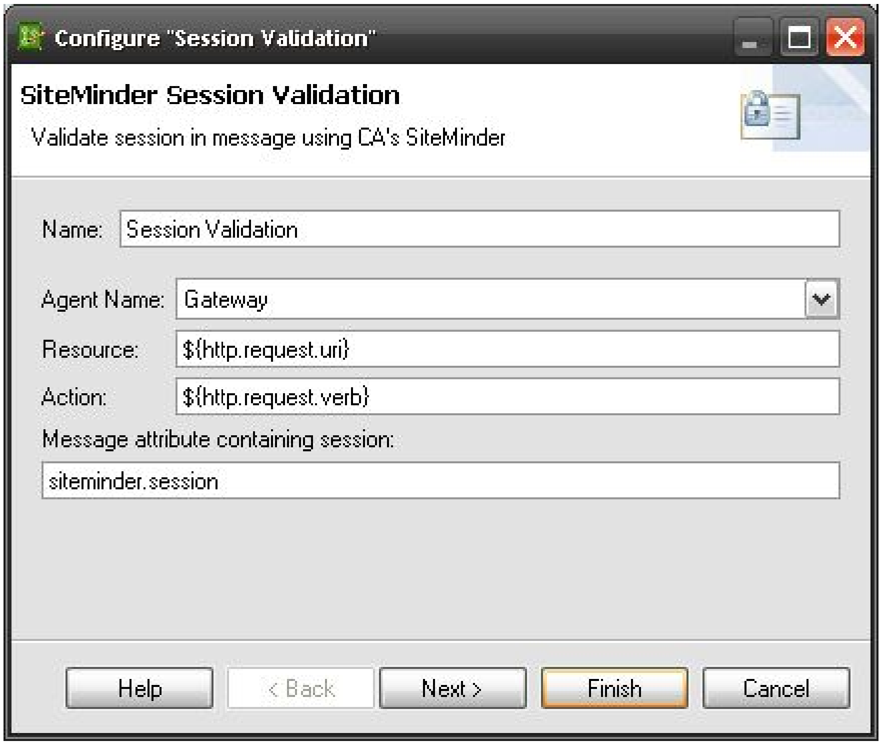
Siteminder Authorize

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	true
TransactionId	string	CCore	CCore

Oracle Enterprise Gateway Configuration:

Authorization via Siteminder is configured by adding the Siteminder Validation and Siteminder Authorization filter to a policy, normally after having authenticated via Siteminder.

1. Add a **"Session Validation"** filter located in the **"CA SiteMinder"** filter category.
2. For the **"Agent Name"** select the available SiteMinder agent in the drop down list as configured in the **Siteminder Authentication** section above.
3. Ensure that the **"Message Attribute Containing Session"** is set to **"siteminder.session".**
4. For the **"Resource"** and **"Action"** fields the values can be left default. The Gateway will retrieve these attributes from the request and the wildcarded values will be set to those matching attributes.



Configure "Session Validation"

SiteMinder Session Validation

Validate session in message using CA's SiteMinder

Name: Session Validation

Agent Name: Gateway

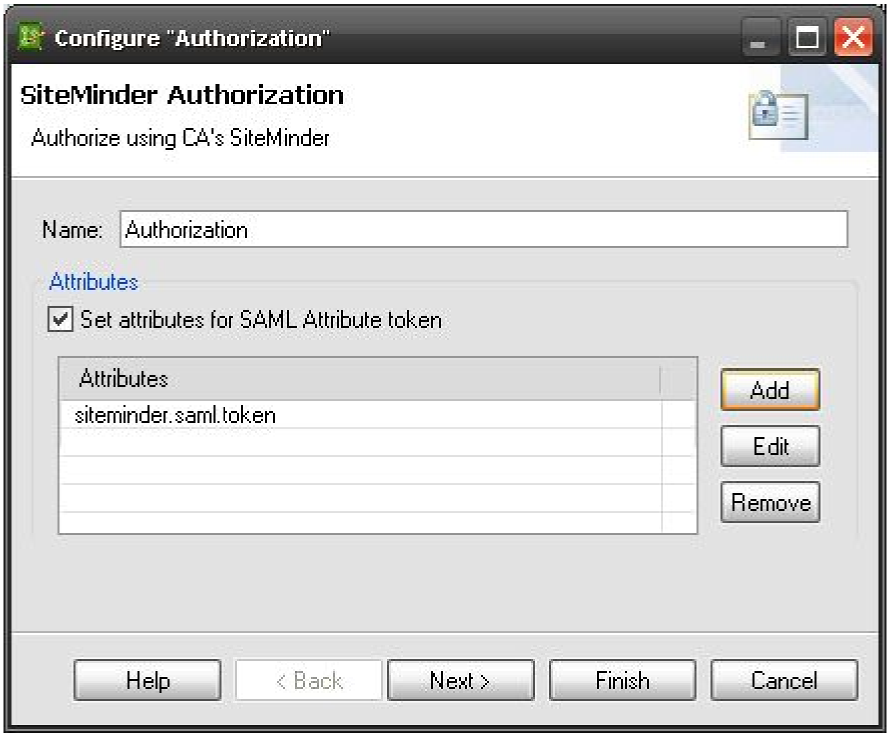
Resource: ${http.request.uri}

Action: ${http.request.verb}

Message attribute containing session:

siteminder.session

5. Next is to add the **"Siteminder Validation"** filter also located in the **"CA SiteMinder"** filter category.
6. Additionally an attribute can be configured that is used as a SAML Attribute Token for consumption downstream if required. By default this is not enabled.



Configure "Authorization"

SiteMinder Authorization

Authorize using CA's SiteMinder

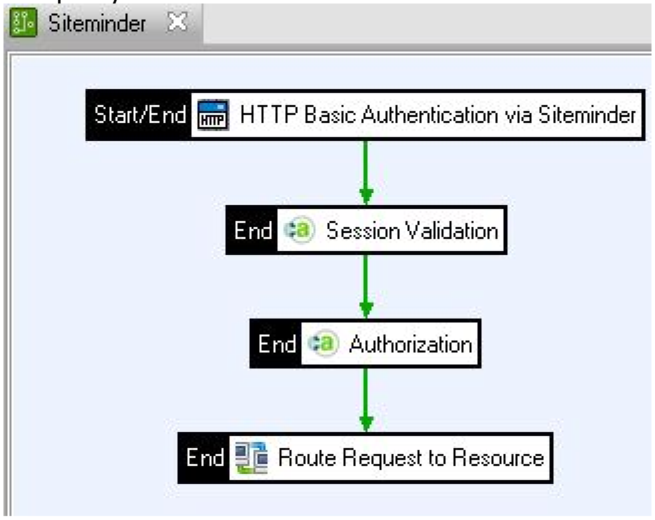
Name: Authorization

Attributes

☑ Set attributes for SAML Attribute token

Attributes
siteminder.saml.token

The policy:

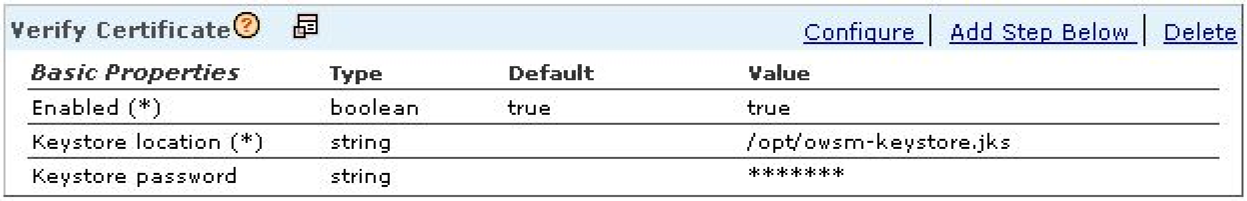

Siteminder
Start/End
HTTP Basic Authentication via Siteminder
End
Session Validation
End
Authorization
End
Route Request to Resource

For more information on how the Gateway integrates with SiteMinder please refer to the Fusion Gateway documentation as well as the Fusion Gateway Integration with Siteminder Guide.

Step: Verify Certificate

Verify a certificate against a local keystore.

OWSM Configuration:



Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	true
Keystore location (*)	string		/opt/owsm-keystore.jks
Keystore password	string		*******

Oracle Enterprise Gateway Configuration:

Refer to the **"Appendix"** section for how to add Certificates and Keys to the Gateway Certificate Store.

After the certificates and keys have been loaded to the Gateway Certificate Store, any filter using the Certificate Store has access to these Certificate and Keys.

For Certificate Validation the Gateway has the following filters available under the **"Certificates"** filter category that can be used in any policy.

The **"Certificate Chain"** filter which ensures that a certificate has been issued by a trusted source is the closest match to the **"Verify Certificate"** option in OWSM.

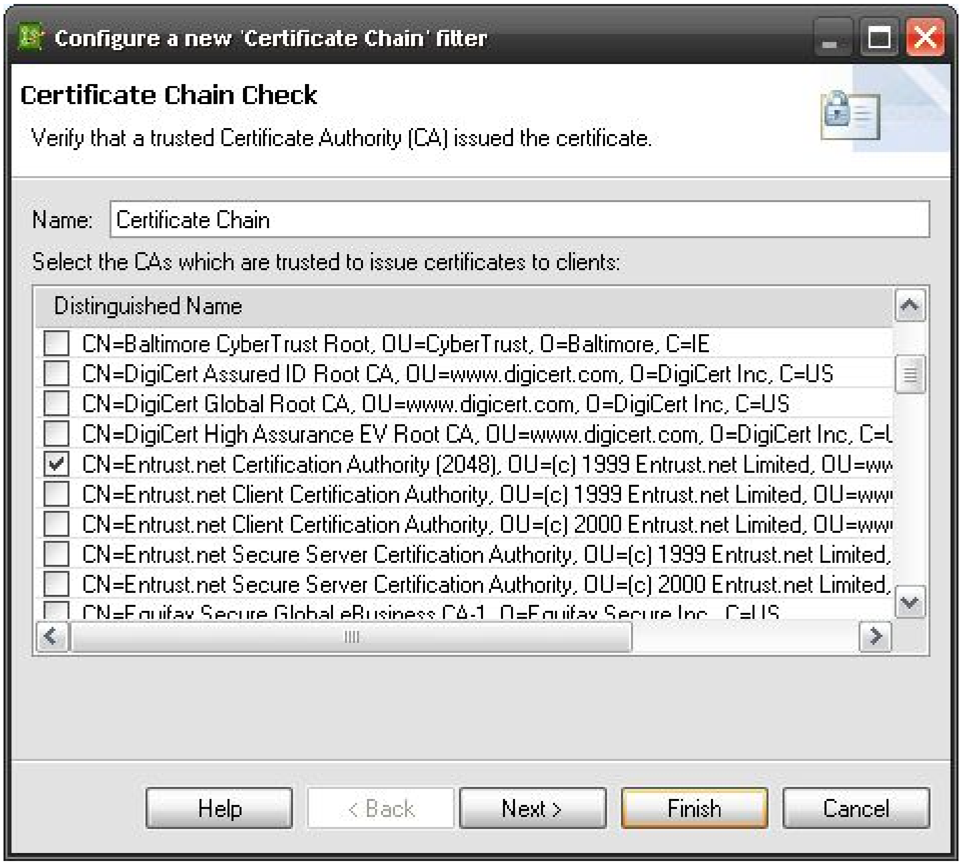
To configure the **"Certificate Chain"** filter:

1. Add a **"Certificate Chain"** filter from the **"Certificates"** filter category into the policy palette.

2. Select the Certificate Authority which is to be trusted to issue certificates to clients by selecting the checkbox next to the certificate D-Name. Multiple CA certificates can be selected.

3. The **"Certificate Chain"** filter has to be preceded by a filter that generates the **"Certificates"** attribute, which is a list containing certificates that has been retrieved by predecessor filters in the Gateway. Normally this will be one of the following filters: SAML PDP XML-Signature Response Verification, SAML PDP Response XML-Signature Verification, HTTP Header Authentication, Integrity, XML-Signature Verification, SAML, XML-Signature Verification, SAML Authorization, XML-Signature Verification, SSL Authentication and XML-Signature Authentication for example.

The Cert Chain filter:



Configure a new 'Certificate Chain' filter

Certificate Chain Check

Verify that a trusted Certificate Authority (CA) issued the certificate.

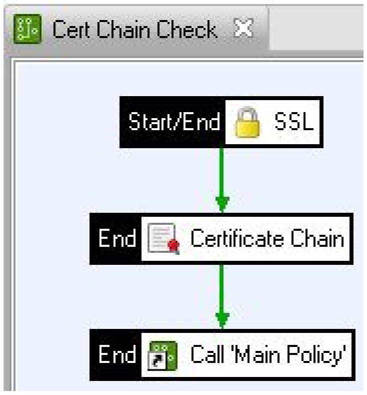
Name: Certificate Chain

Select the CAs which are trusted to issue certificates to clients:

Distinguished Name

- □ CN=Baltimore CyberTrust Root, OU=CyberTrust, O=Baltimore, C=IE
- □ CN=DigiCert Assured ID Root CA, OU=www.digicert.com, O=DigiCert Inc, C=US
- □ CN=DigiCert Global Root CA, OU=www.digicert.com, O=DigiCert Inc, C=US
- □ CN=DigiCert High Assurance EV Root CA, OU=www.digicert.com, O=DigiCert Inc, C=U
- ☑ CN=Entrust.net Certification Authority (2048), OU=(c) 1999 Entrust.net Limited, OU=ww
- □ CN=Entrust.net Client Certification Authority, OU=(c) 1999 Entrust.net Limited, OU=ww
- □ CN=Entrust.net Client Certification Authority, OU=(c) 2000 Entrust.net Limited, OU=ww
- □ CN=Entrust.net Secure Server Certification Authority, OU=(c) 1999 Entrust.net Limited,
- □ CN=Entrust.net Secure Server Certification Authority, OU=(c) 2000 Entrust.net Limited,
- □ CN=Equifax Secure Global eBusiness CA-1, O=Equifax Secure Inc., C=US

The Cert Chain filter used in a simple policy



In addition to the Cert Chain filter for certificate validation the Gateway is also capable of providing certificate validation using the following technologies:

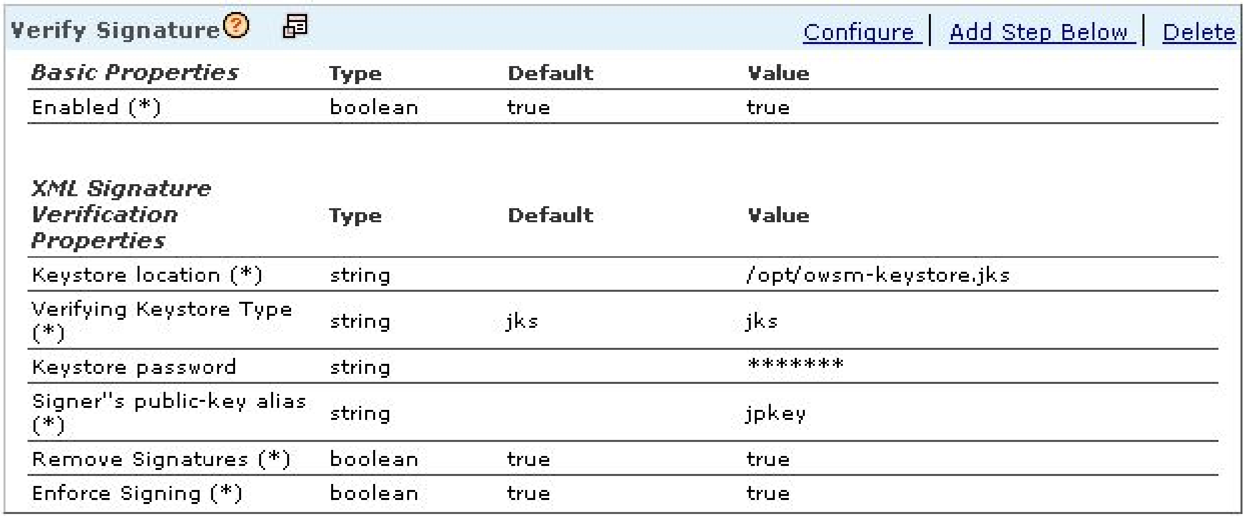
- **CRL Dynamic** (Validate certificate against a Certificate Revocation List URL)
- **CRL Static** (Validate certificate against a local Certificate Revocation List file)
- **CRL LDAP** (Validate certificate against a Certificate Revocation List located in a LDAP directory)
- **CRL Responder** (Allows Gateway to behave as a Certificate Revocation List) responder)
- **Certificate Chain** (Ensures that a certificate has been issued by a trusted source)
- **Certificate Validity** (Performs a simple check on a certificate to make sure that it has not expired.)
- **OCSP** (Validate certificate against an Online Certificate Status Protocol responder)
- **XKMS** (Validate certificate against a XML Key Management Service responder)

For more information about configuring these filters please refer to the Fusion Gateway Administrators Guide. Please also refer to the SSL Integrity and Confidentiality guide.

Step: Verify Signature

XML Signature Verification

OWSM Configuration:



Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	true

XML Signature Verification Properties	Type	Default	Value
Keystore location (*)	string		/opt/owsm-keystore.jks
Verifying Keystore Type (*)	string	jks	jks
Keystore password	string		*******
Signer''s public-key alias (*)	string		jpkey
Remove Signatures (*)	boolean	true	true
Enforce Signing (*)	boolean	true	true

Oracle Enterprise Gateway Configuration:

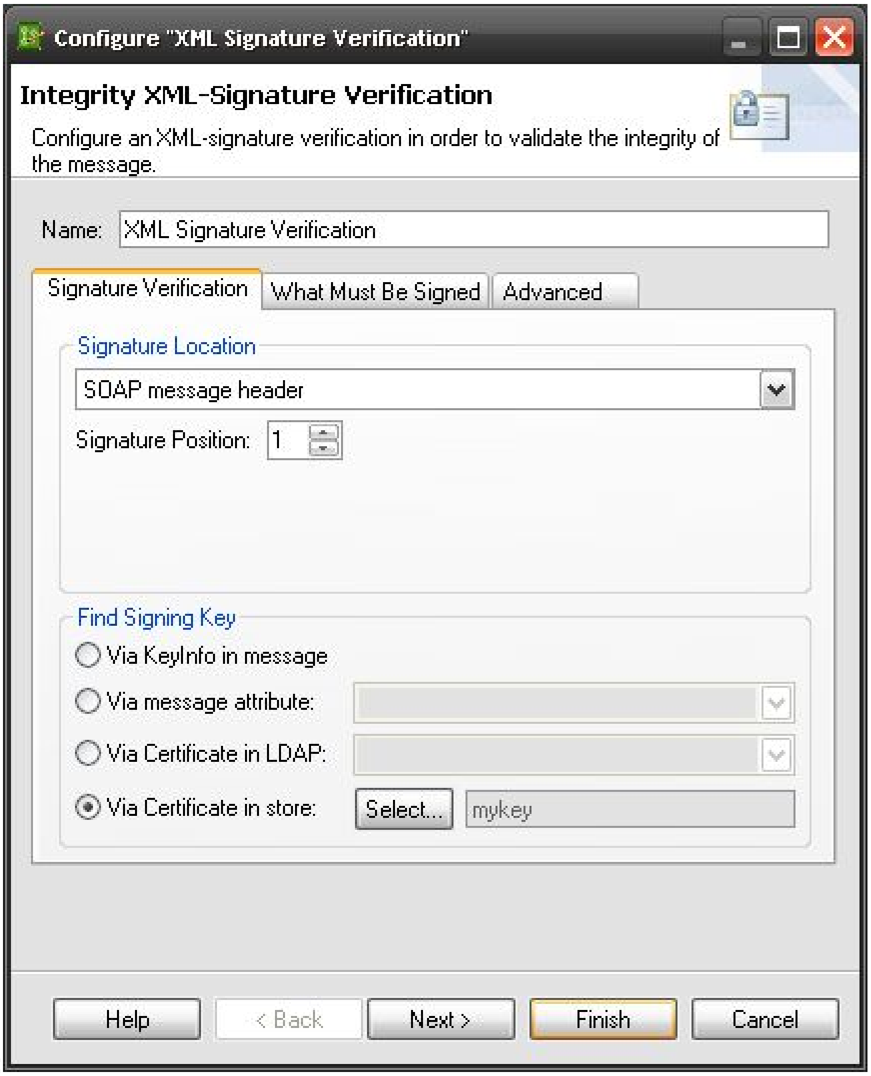
The Gateway can verify signed messages using the **"XML Signature Verification"** filter.

1. Add a **"XML Signature Verification"** filter from the **"Integrity"** filter category to the policy palette.
2. Under the **"Signature Location"** section choose the location where the signature resides from the drop down list.
3. Set the **"Signature Position"** to **1**.
4. Under the **"Find Signing"** key section select the location of the public key that will be used to verify the signature.
5. Under the **"What Must Be Signed"** tab the particular part of the message can be selected that

should be signed, although not mandatory, if selected strengthens the integrity verification process, if it is known that a specific part of the message should be signed.

6. The **"Advanced"** tab covers additional options, for example to fail if no signature is present, which can be enabled or disabled. Please refer to Gateway documentation for explanation of all features in the filter.

The Signature Verification configuration window:



Configure "XML Signature Verification"

Integrity XML-Signature Verification

Configure an XML-signature verification in order to validate the integrity of the message.

Name: XML Signature Verification

Signature Location: SOAP message header

Signature Position: 1

Find Signing Key

○ Via KeyInfo in message

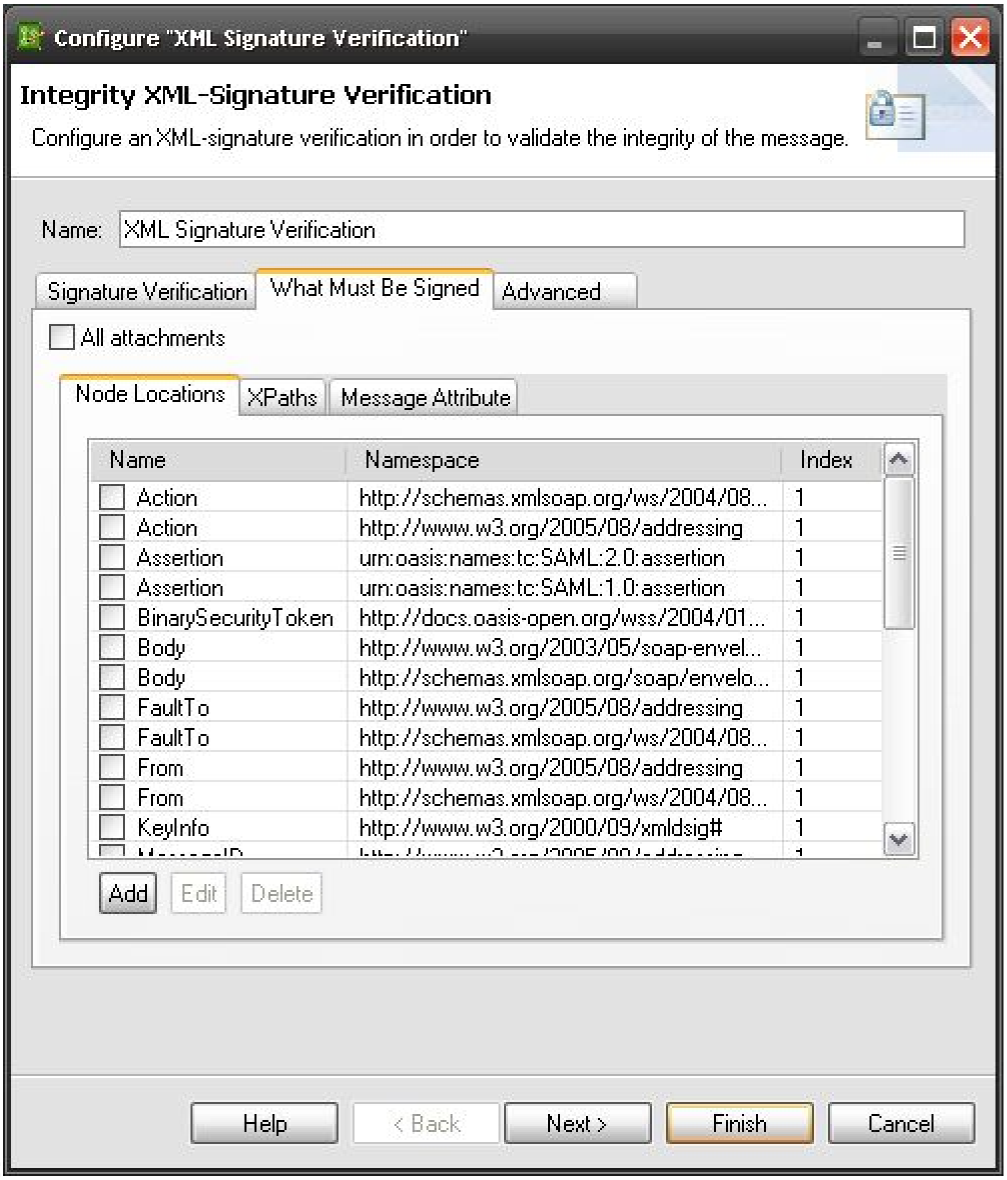
○ Via message attribute:

○ Via Certificate in LDAP:

● Via Certificate in store: mykey

The What Must be Signed configuration

window:



Configure "XML Signature Verification"

Integrity XML-Signature Verification

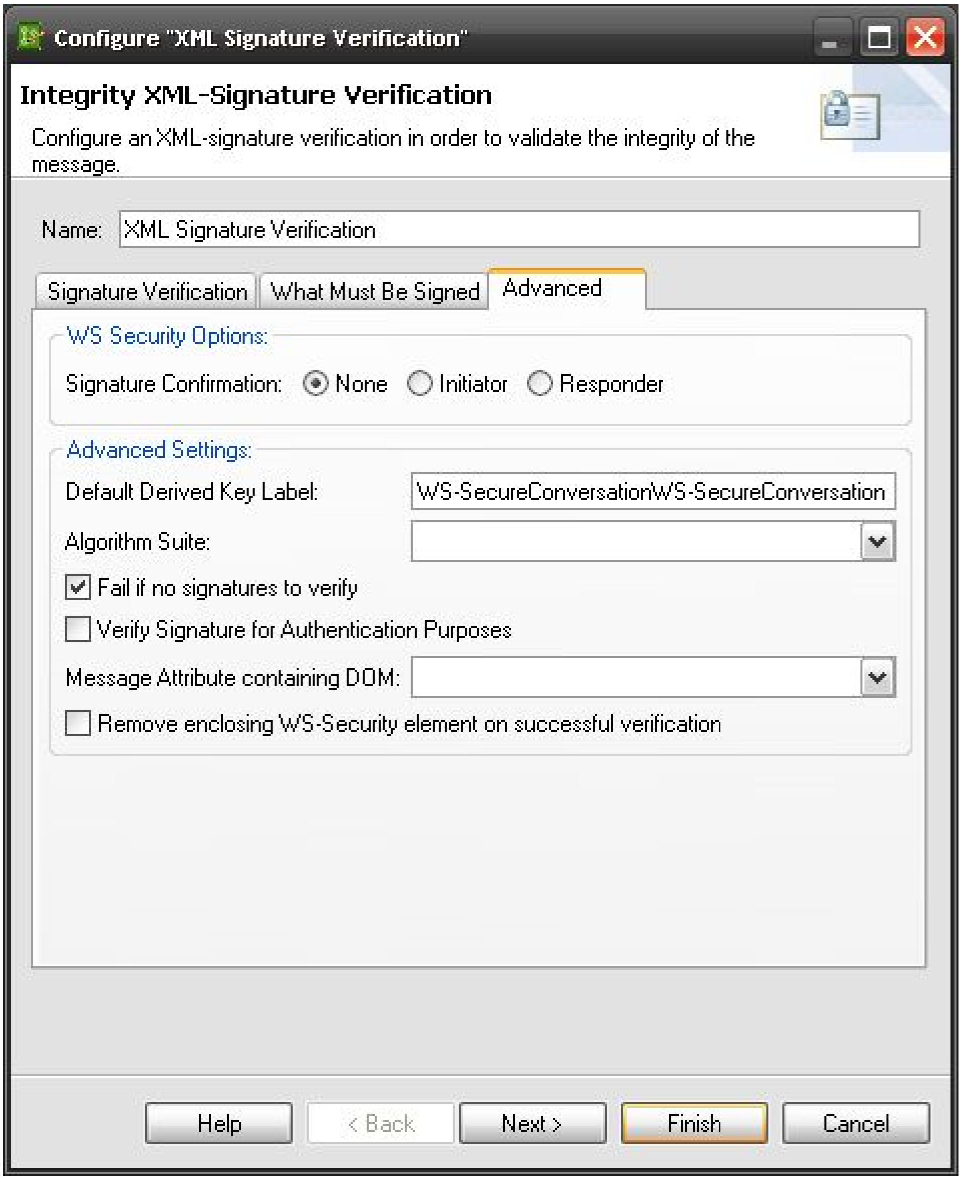
Configure an XML-signature verification in order to validate the integrity of the message.

Name: XML Signature Verification

□ All attachments

Name	Namespace	Index
□ Action	http://schemas.xmlsoap.org/ws/2004/08...	1
□ Action	http://www.w3.org/2005/08/addressing	1
□ Assertion	urn:oasis:names:tc:SAML:2.0:assertion	1
□ Assertion	urn:oasis:names:tc:SAML:1.0:assertion	1
□ BinarySecurityToken	http://docs.oasis-open.org/wss/2004/01...	1
□ Body	http://www.w3.org/2003/05/soap-envel...	1
□ Body	http://schemas.xmlsoap.org/soap/envelo...	1
□ FaultTo	http://www.w3.org/2005/08/addressing	1
□ FaultTo	http://schemas.xmlsoap.org/ws/2004/08...	1
□ From	http://www.w3.org/2005/08/addressing	1
□ From	http://schemas.xmlsoap.org/ws/2004/08...	1
□ KeyInfo	http://www.w3.org/2000/09/xmldsig#	1
□ MessageID	http://www.w3.org/2005/08/addressing	1

The Advanced configuration

window:

Configure "XML Signature Verification"

Integrity XML-Signature Verification

Configure an XML-signature verification in order to validate the integrity of the message.

Name: XML Signature Verification

WS Security Options:

Signature Confirmation: ● None ○ Initiator ○ Responder

Advanced Settings:

Default Derived Key Label: WS-SecureConversationWS-SecureConversation

Algorithm Suite:

☑ Fail if no signatures to verify

☐ Verify Signature for Authentication Purposes

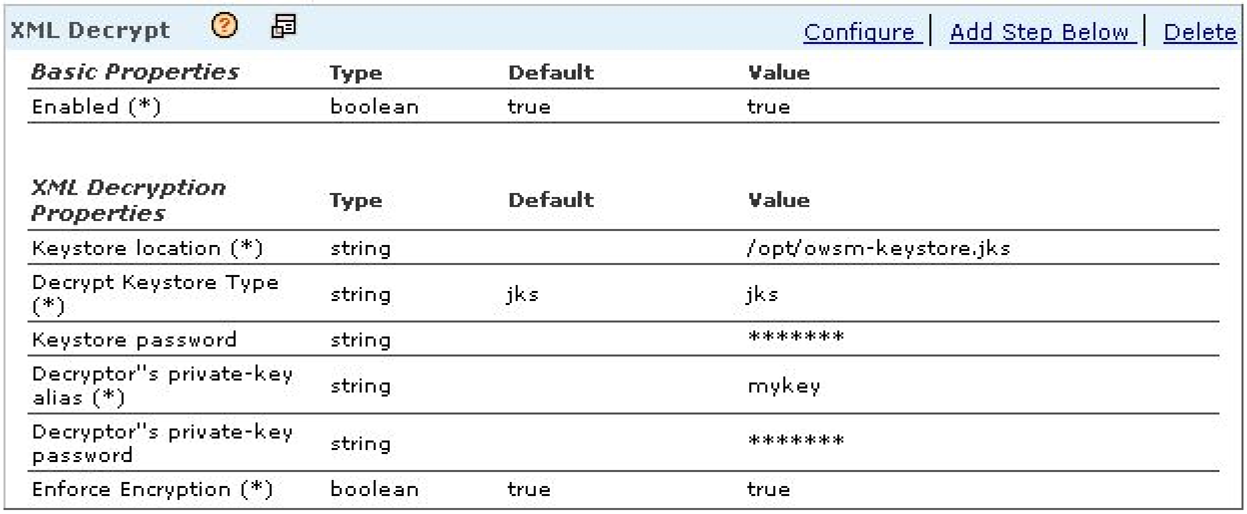
Message Attribute containing DOM:

☐ Remove enclosing WS-Security element on successful verification

Step: XML Decrypt

XML Decryption

OWSM Configuration:



Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	true

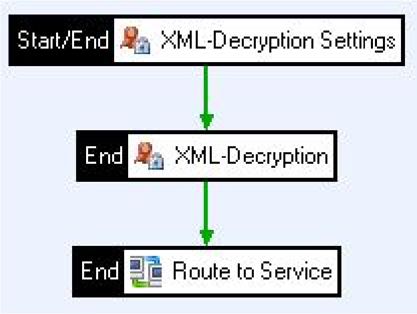
XML Decryption Properties	Type	Default	Value
Keystore location (*)	string		/opt/owsm-keystore.jks
Decrypt Keystore Type (*)	string	jks	jks
Keystore password	string		*******
Decryptor''s private-key alias (*)	string		mykey
Decryptor''s private-key password	string		*******
Enforce Encryption (*)	boolean	true	true

Oracle Enterprise Gateway Configuration:

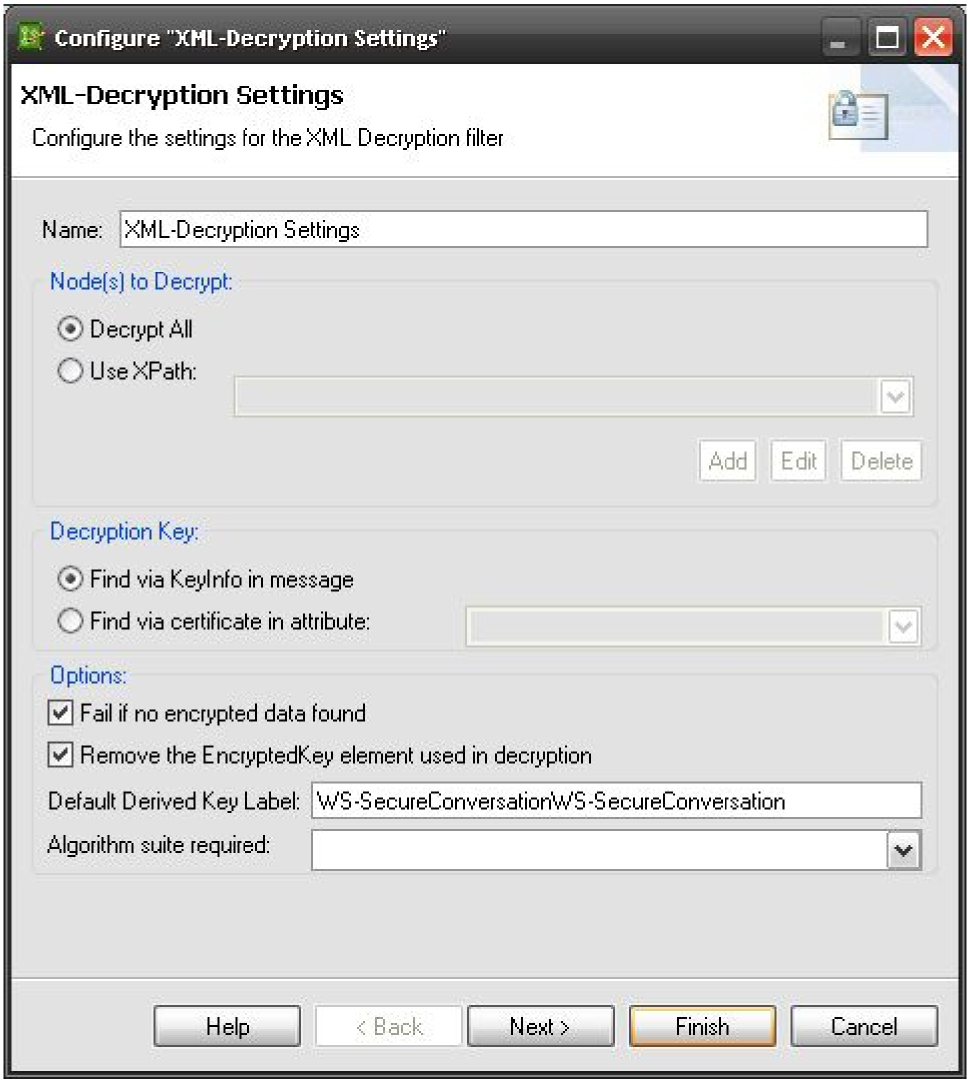
To decrypt messages in the Gateway, the **"Decryption Settings"** filter followed by the **"Decryption"** filter is used.

1. Add a **"Decryption Settings"** filter from the **"Encryption"** filter category.
2. Under the **"Nodes to Decrypt"** section there are two options to select from. Select **"Decrypt All"** to decrypt all encrypted content in a request or **"Use XPath"** to explicitly select the EncryptionData block in a request.
3. Under the **"Decryption Key"** section select where the decryption key will be retrieved from. The options are from the **"Key Info"** section in the request or via a message attribute generated by predecessor filters in the policy.
4. Under the options section the filter can be configured to fail if not encryption data was found in the request and also to remove the encrypted key element used for decryption.
5. Add an **"Decryption"** filter from the **"Encryption"** filter category. No configuration is required in this filter. Connect to the **"Decryption Settings"** filter via a success path.
6. For information on all the options please refer to the Gateway documentation.

A sample Decryption Policy:



The Decryption Settings filter:



XML-Decryption Settings

Configure the settings for the XML Decryption filter

Name: XML-Decryption Settings

Node(s) to Decrypt:

● Decrypt All

○ Use XPath:

Decryption Key:

● Find via KeyInfo in message

○ Find via certificate in attribute:

Options:

☑ Fail if no encrypted data found

☑ Remove the EncryptedKey element used in decryption

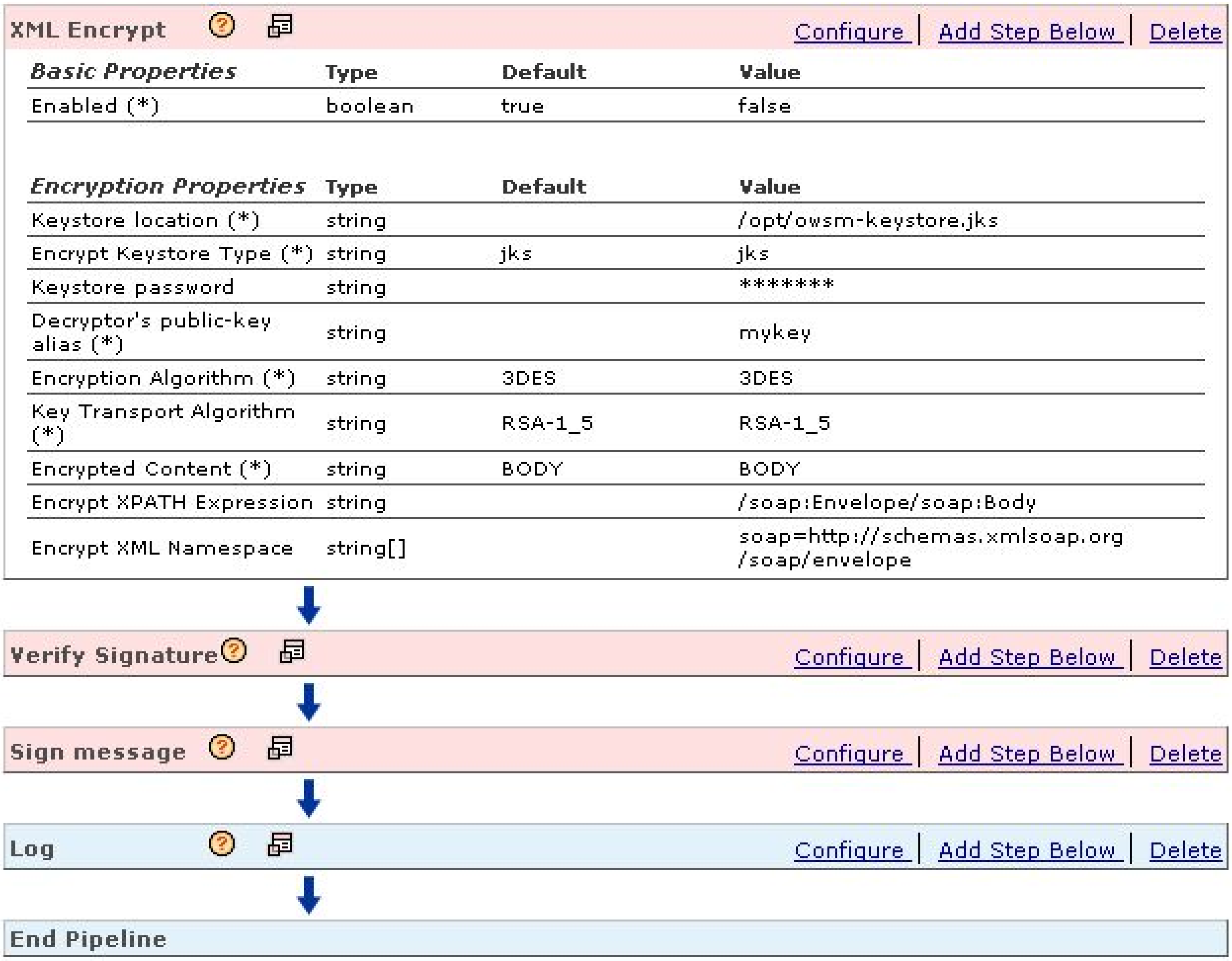
Default Derived Key Label: WS-SecureConversationWS-SecureConversation

Algorithm suite required:

Step: XML Encrypt

XML Encryption

OWSM Configuration:



XML Encrypt

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	false

Encryption Properties	Type	Default	Value
Keystore location (*)	string		/opt/owsm-keystore.jks
Encrypt Keystore Type (*)	string	jks	jks
Keystore password	string		*******
Decryptor's public-key alias (*)	string		mykey
Encryption Algorithm (*)	string	3DES	3DES
Key Transport Algorithm (*)	string	RSA-1_5	RSA-1_5
Encrypted Content (*)	string	BODY	BODY
Encrypt XPATH Expression	string		/soap:Envelope/soap:Body
Encrypt XML Namespace	string[]		soap=http://schemas.xmlsoap.org/soap/envelope

Verify Signature

Sign message

Log

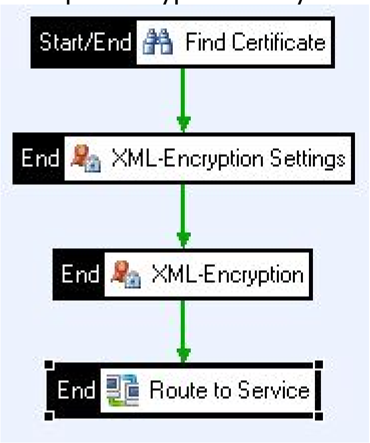
End Pipeline

Oracle Enterprise Gateway Configuration:
To encrypt messages in the Gateway, the **"Encryption Settings"** filter followed by the **"Encryption"** filter is used. The **"Encryption Settings"** filter is dependant on the previous filters in the policy chain to supply the **"Certificate"** attribute that holds the key that will be used to encrypt the request. For this the **"Find Certificate"** filter will be used to locate the key in the Gateway certificate store.

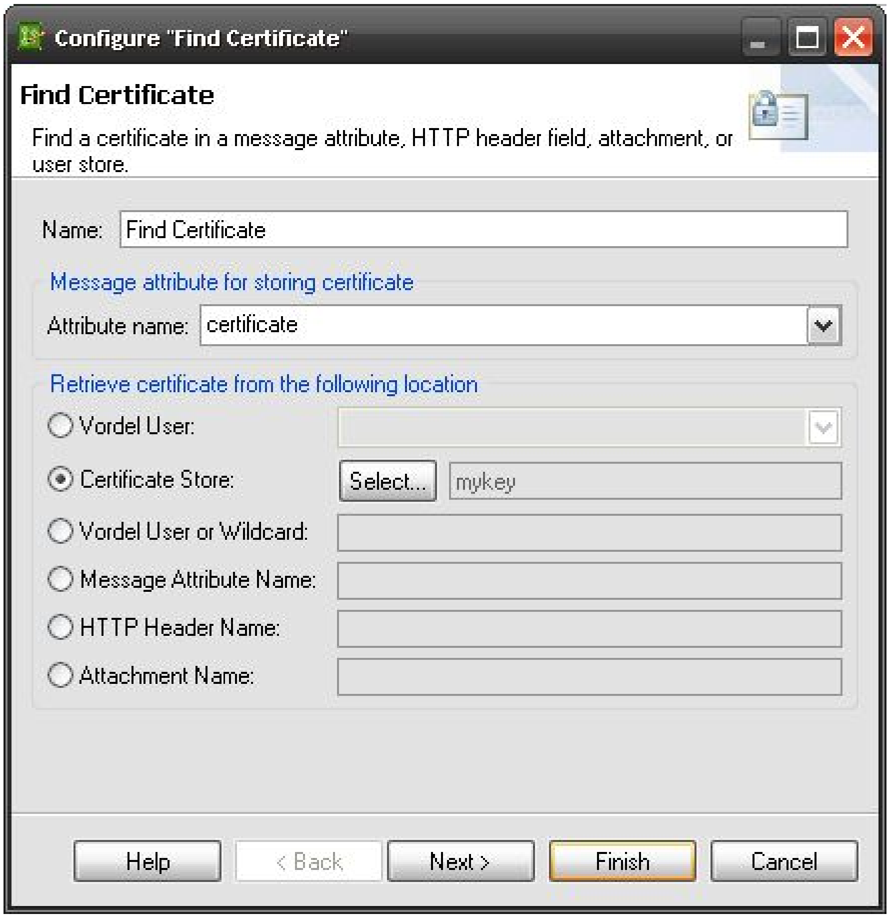
1. Add a **"Find Certificate"** filter from the **"Certificates"** filter category.
2. Select the option where the key is to be obtained from. In this example the **"Certificate Store"** option has been selected with the relevant certificate/key combination chosen from the list of certificates and keys then click on **"Finish"**.
3. Next Drag a **"XML-Encryption Settings"** filter from the **"Encryption"** filter category.
4. Under the **"Encryption Key"** tab the settings can be left as default.
5. Under the **"Key Info"** tab select the **"Security Token Reference"** radio button and select **"X509v3"** from the drop down list. Also select **"Embed"** option and select the **"Include Security Token"** check box.
6. Under the **"Recipients"** tab the options can be left as default.
7. Under the **"What to Encrypt"** tab select the part of the request that needs to be encrypted from either the **"Nodes Location"**, **"XPath"** or **"Message Attribute"** tab.
8. The **"Advanced"** tab contains additional options for example inserting a timestamp, select a particular algorithm for encryption if required and other options. Select **"TripleDes"** under for the **"Encryption Algorithm"** option under the **"Algorithm Suite"** tab**.**

9. Click on **"Finish"** once the configuration of the **"Encryption Settings"** filter is complete and connect to the **"Find Certificates"** filter via a success path.
10. Add an **"Encryption"** filter from the **"Encryption"** filter category. No configuration is required in this filter. Connect to the **"Encryption Settings"** filter via a success path.
11. Please refer to the Gateway documentation for more information on all the options in the **"Encryption Settings"** filter.

Example Encryption Policy:



The Find Certificates filter:


Configure "Find Certificate"

Find Certificate

Find a certificate in a message attribute, HTTP header field, attachment, or user store.

Name: Find Certificate

Message attribute for storing certificate

Attribute name: certificate

Retrieve certificate from the following location

○ Vordel User:

● Certificate Store: Select... mykey

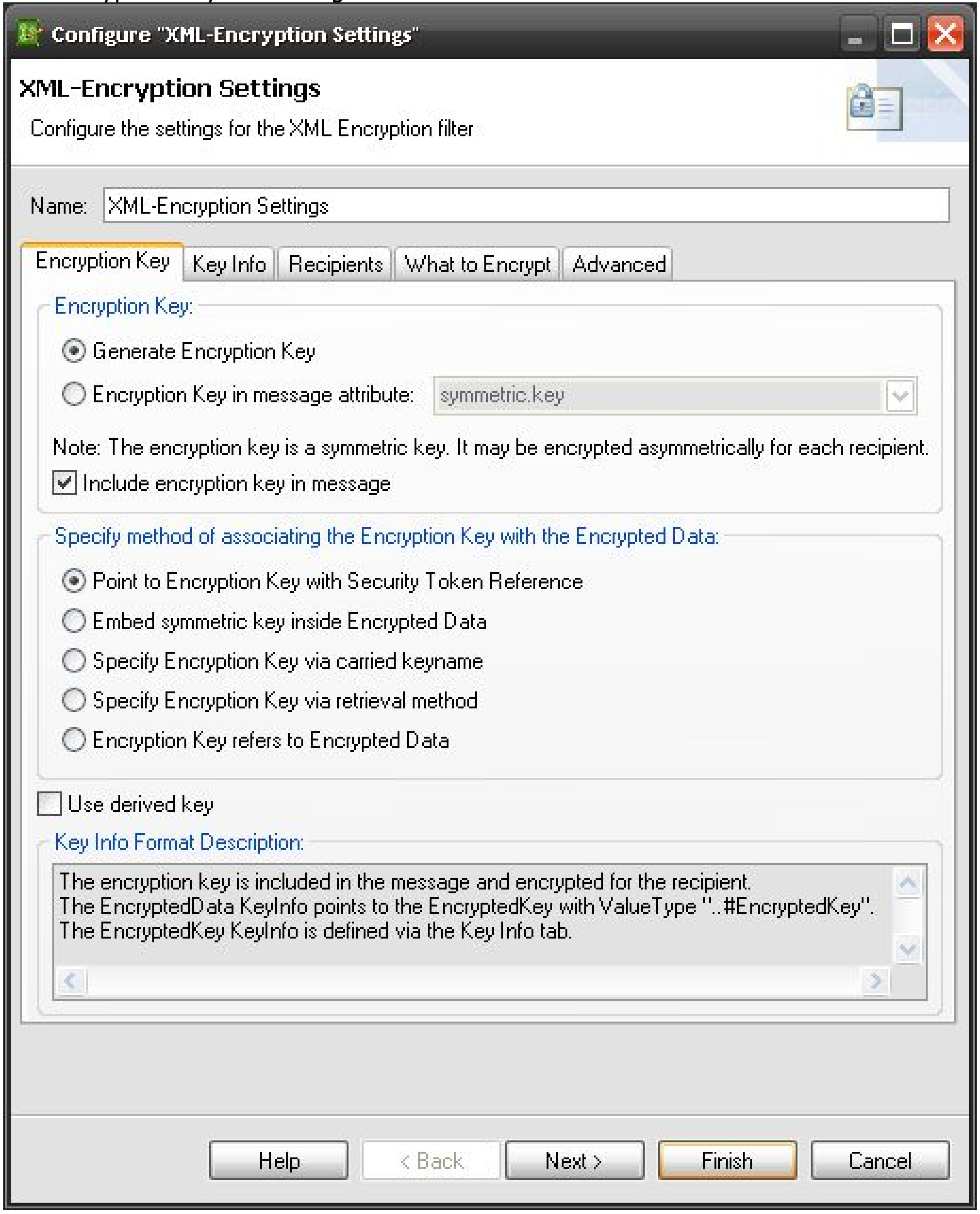
○ Vordel User or Wildcard:

○ Message Attribute Name:

○ HTTP Header Name:

○ Attachment Name:

The Encryption Key tab settings:



Configure "XML-Encryption Settings"

XML-Encryption Settings

Configure the settings for the XML Encryption filter

Name: XML-Encryption Settings

Encryption Key:

● Generate Encryption Key

○ Encryption Key in message attribute: symmetric.key

Note: The encryption key is a symmetric key. It may be encrypted asymmetrically for each recipient.

☑ Include encryption key in message

Specify method of associating the Encryption Key with the Encrypted Data:

● Point to Encryption Key with Security Token Reference

○ Embed symmetric key inside Encrypted Data

○ Specify Encryption Key via carried keyname

○ Specify Encryption Key via retrieval method

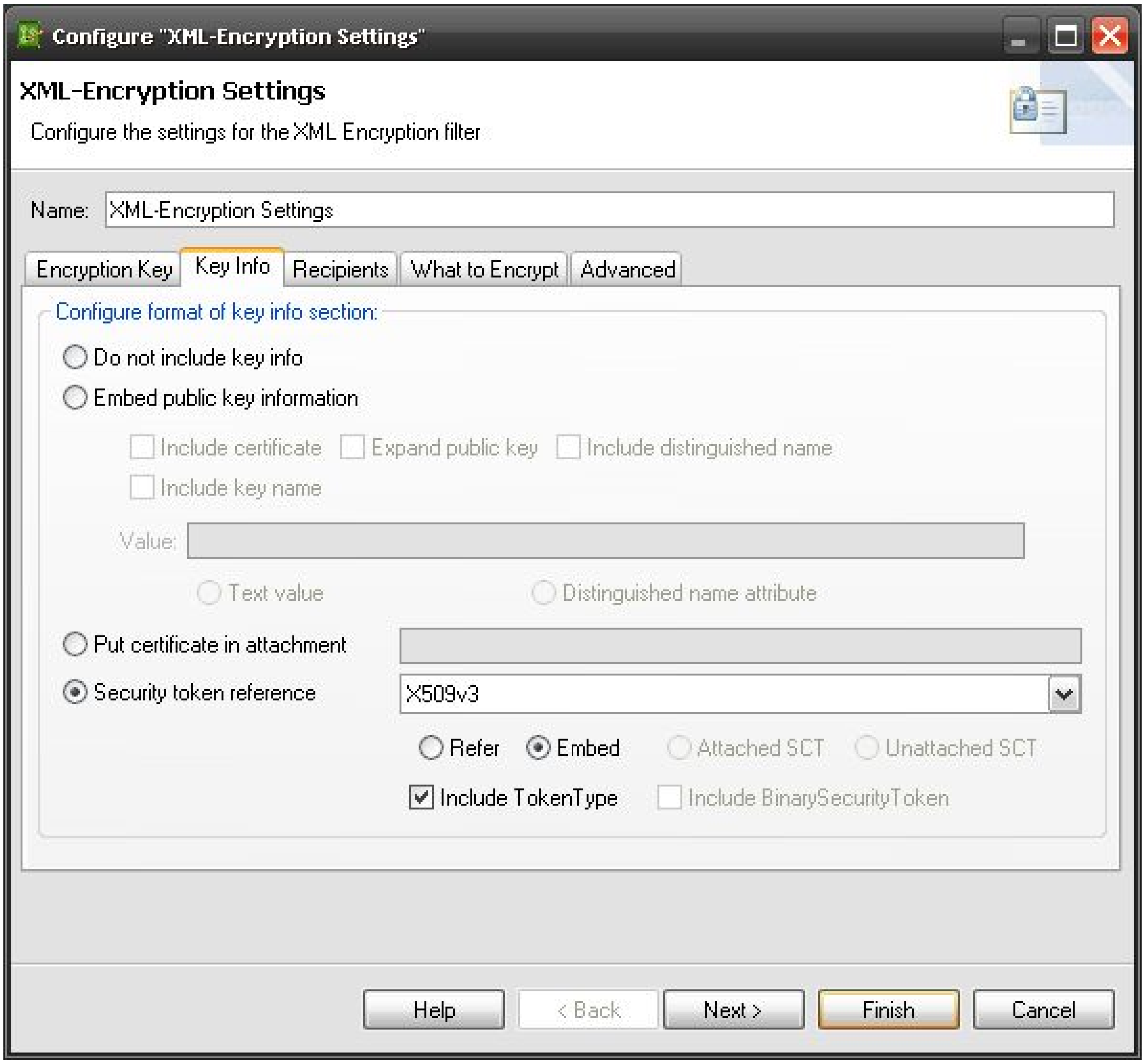
○ Encryption Key refers to Encrypted Data

□ Use derived key

Key Info Format Description:

The encryption key is included in the message and encrypted for the recipient.
The EncryptedData KeyInfo points to the EncryptedKey with ValueType "..#EncryptedKey".
The EncryptedKey KeyInfo is defined via the Key Info tab.

The Key Info tab Settings:



XML-Encryption Settings

Configure the settings for the XML Encryption filter

Name: XML-Encryption Settings

Configure format of key info section:

○ Do not include key info

○ Embed public key information

□ Include certificate □ Expand public key □ Include distinguished name

□ Include key name

Value:

○ Text value ○ Distinguished name attribute

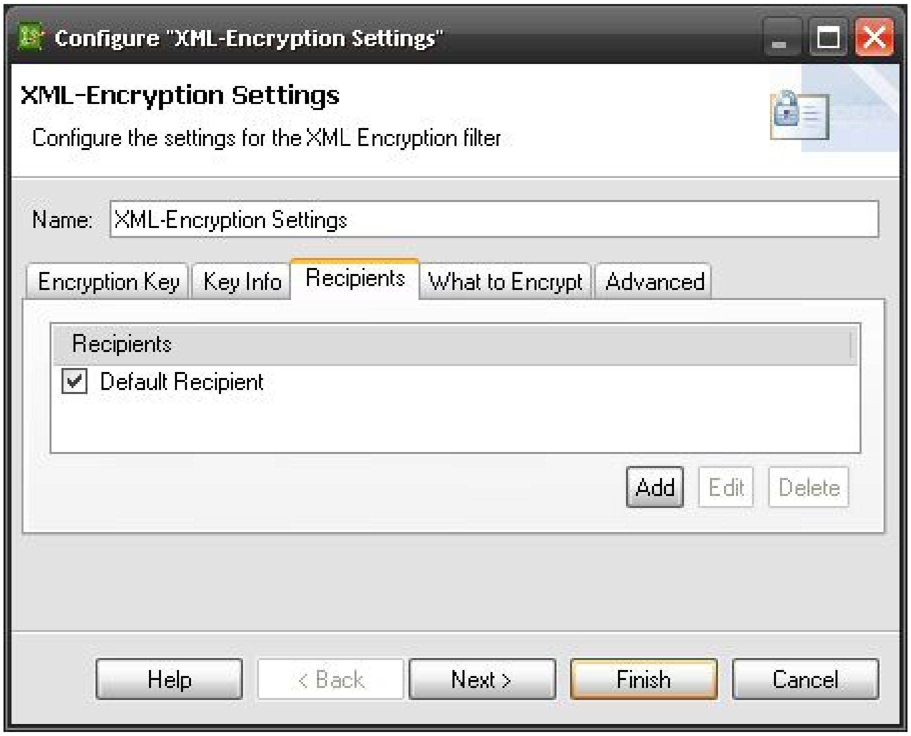
○ Put certificate in attachment

● Security token reference X509v3

○ Refer ● Embed ○ Attached SCT ○ Unattached SCT

☑ Include TokenType □ Include BinarySecurityToken

The Recipients tab settings:



Configure "XML-Encryption Settings"

XML-Encryption Settings

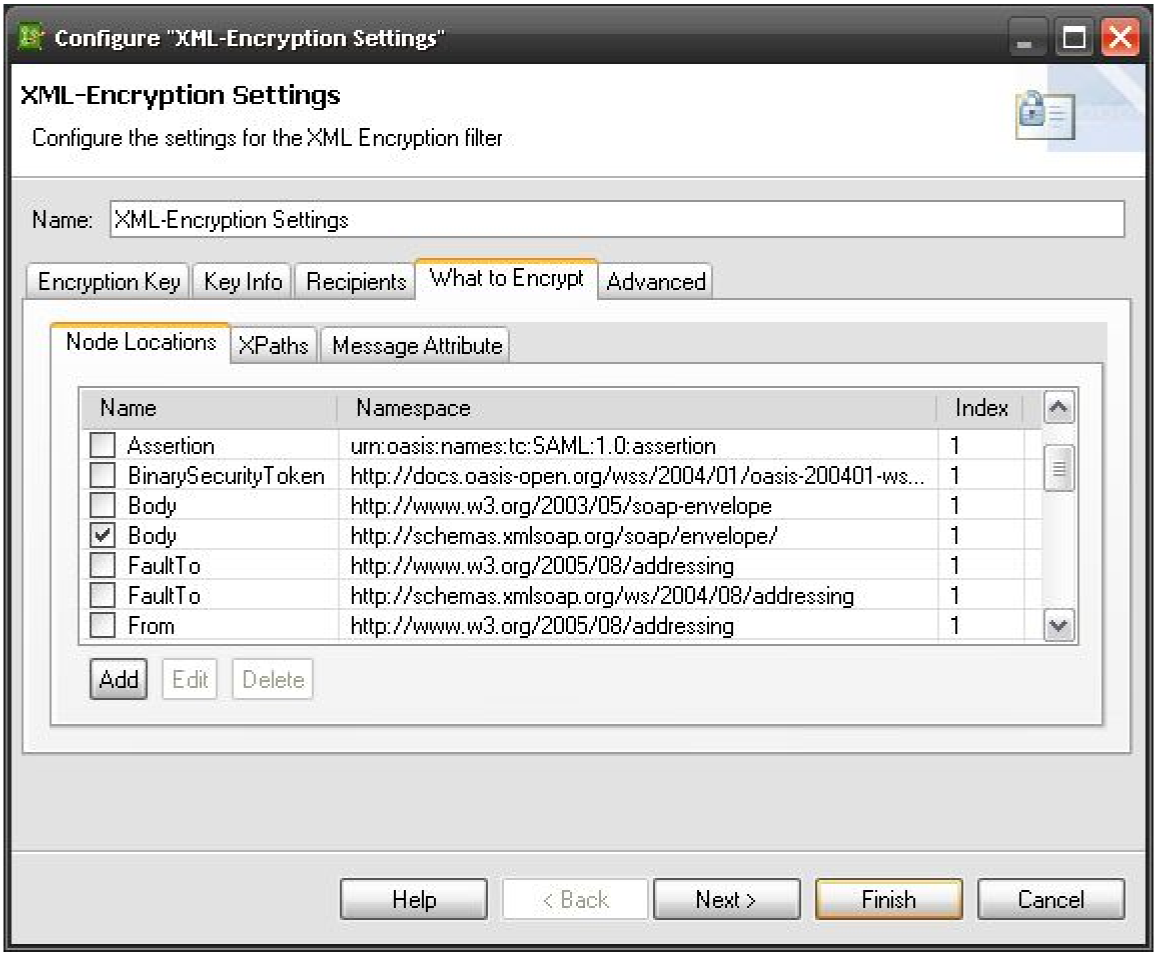
Configure the settings for the XML Encryption filter

Name: XML-Encryption Settings

Recipients

☑ Default Recipient

The What to Encrypt tab settings:



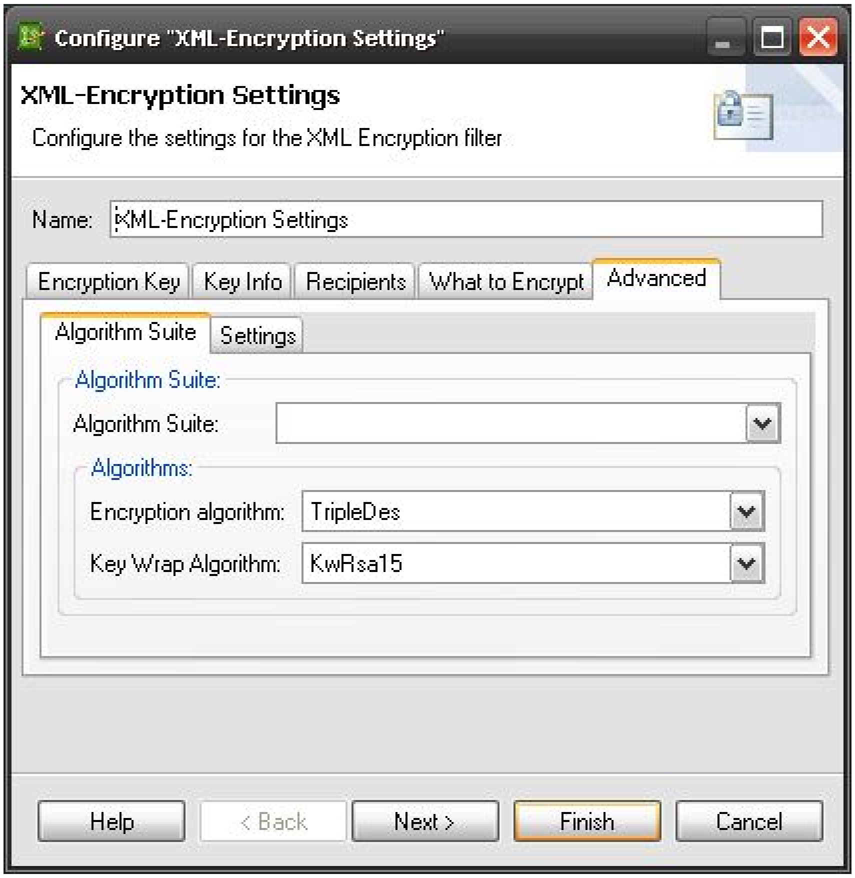
XML-Encryption Settings

Configure the settings for the XML Encryption filter

Name: XML-Encryption Settings

Name	Namespace	Index
□ Assertion	urn:oasis:names:tc:SAML:1.0:assertion	1
□ BinarySecurityToken	http://docs.oasis-open.org/wss/2004/01/oasis-200401-ws...	1
□ Body	http://www.w3.org/2003/05/soap-envelope	1
☑ Body	http://schemas.xmlsoap.org/soap/envelope/	1
□ FaultTo	http://www.w3.org/2005/08/addressing	1
□ FaultTo	http://schemas.xmlsoap.org/ws/2004/08/addressing	1
□ From	http://www.w3.org/2005/08/addressing	1

The Advanced tab settings:



Configure "XML-Encryption Settings"

XML-Encryption Settings

Configure the settings for the XML Encryption filter

Name: XML-Encryption Settings

Algorithm Suite:

Algorithm Suite:

Algorithms:

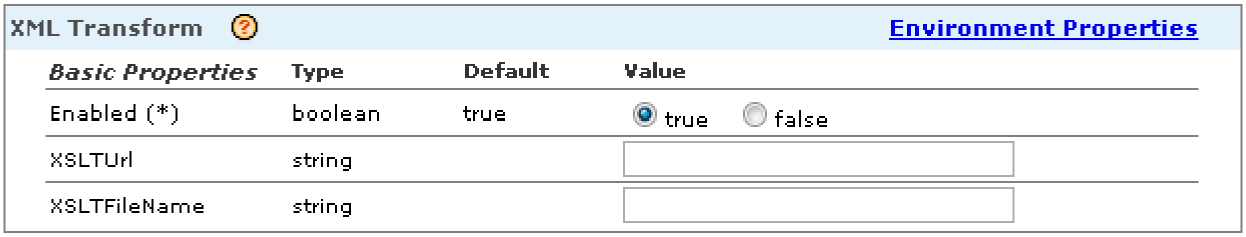
Encryption algorithm: TripleDes

Key Wrap Algorithm: KwRsa15

Step: XML Transform

Transform message using XSL

OWSM Configuration:



XML Transform

Basic Properties	Type	Default	Value
Enabled (*)	boolean	true	● true ○ false
XSLTUrl	string		
XSLTFileName	string		

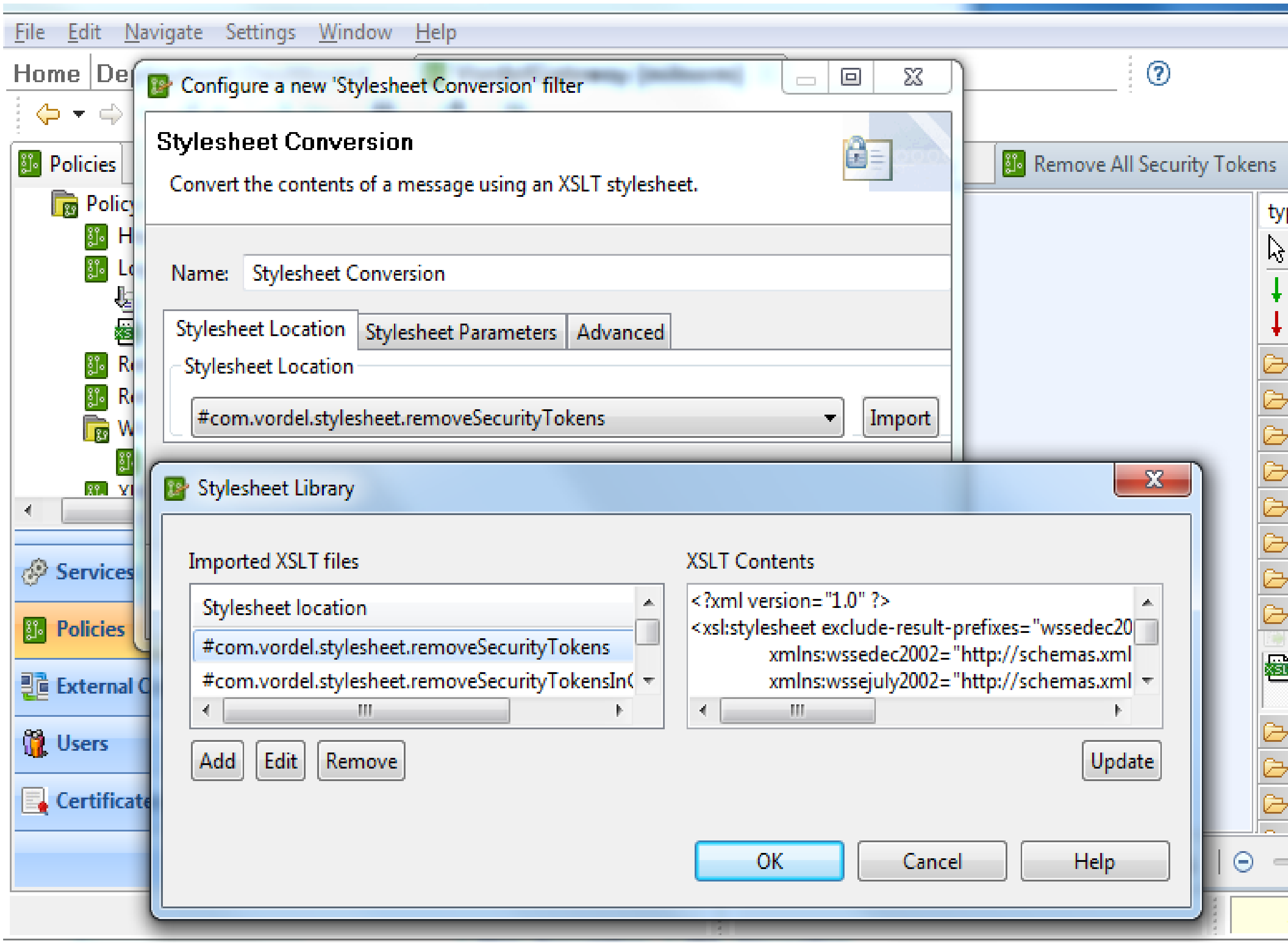
Where

XSLTUrl is the URL that specifies the location of the XSLT file.

XSLTFileName is the Path to the XSLT file on the machine where Oracle WSM is installed.

Oracle Enterprise Gateway Configuration:

Drag and drop a Stylesheet Conversion filter from the Utility class into the circuit at the desired point.


File Edit Navigate Settings Window Help
Home
Policies
Services
Policies
External
Users
Certificate
Remove All Security Tokens
Configure a new 'Stylesheet Conversion' filter
Stylesheet Library
Add
Edit
Remove
Update
OK
Cancel
Help
Import

Stylesheet Conversion

Convert the contents of a message using an XSLT stylesheet.

Name: Stylesheet Conversion

Stylesheet Location | Stylesheet Parameters | Advanced

Stylesheet Location: #com.vordel.stylesheet.removeSecurityTokens

Imported XSLT files

Stylesheet location
#com.vordel.stylesheet.removeSecurityTokens
#com.vordel.stylesheet.removeSecurityTokensIn(

XSLT Contents

```
<?xml version="1.0" ?>
<xsl:stylesheet exclude-result-prefixes="wssedec20
        xmlns:wssedec2002="http://schemas.xml
        xmlns:wssejuly2002="http://schemas.xml
```

Appendix:

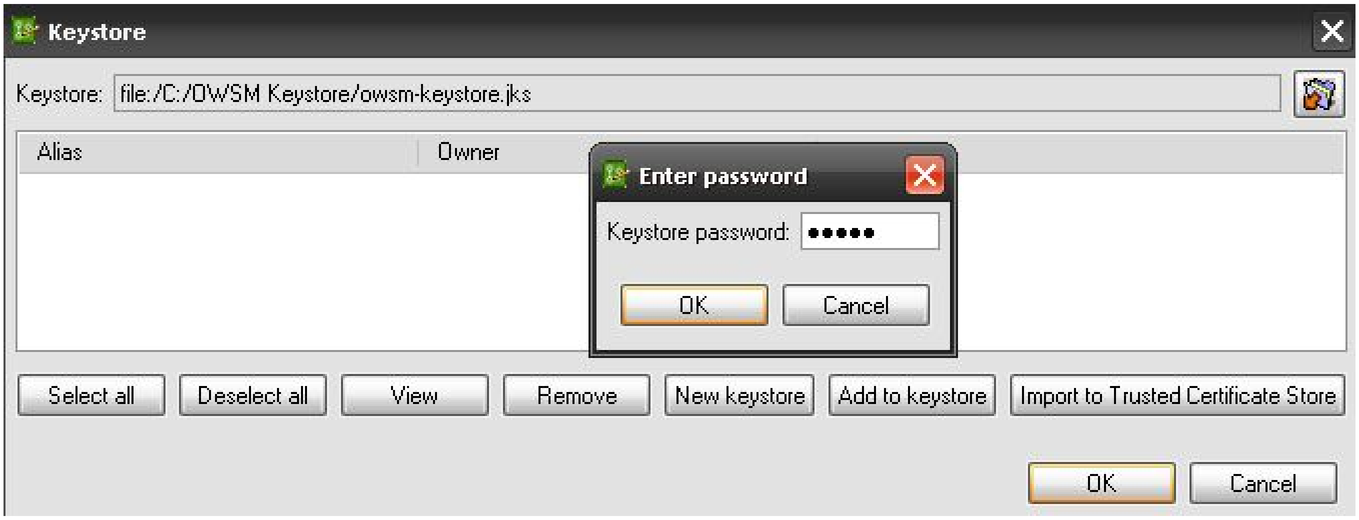
Oracle Enterprise Gateway Certificate Store:

Adding Certificates and Keys to the Oracle Enterprise Gateway Certificate Store:

If OWSM is using certificates or keys contained in java keystore file then this can be imported into the Gateway’s configuration via the following steps:

1. Add Certificates and Keys from the OWSM Keystore to the Oracle Enterprise Gateway Certificate Store

Click on the **“Certificates”** module in Policy studio then select **“Keystore”**, click on the **“Browse”** button and browse to the keystore file and connect to it using Keystore password



Keystore: file:/C:/OWSM Keystore/owsm-keystore.jks

Keystore password: ●●●●●

A list of certificates and keys will be show. Select the certificates and keys to export to the Gateway Certificate store. Then click on **"Import to Trusted Certificate Store".**

2. To add a Certificate or Certificate and Key to the Oracle Enterprise Gateway Certificate store that does not reside in another Keystore:

1. Click on the **"Certificates"** module in the left side of Policy Studio.
2. Click on the **"Certificates"** object at the top.
3. On the right hand side of Policy Studio click on **"Create/Import"** button.
4. Click on the **"Import Certificate"** button if it is a certificate with public key
that needs importing.
5. Click on the **"Import Certificate and Key"** option to import a certificate and private key.
6. Certificate and Key can also be imported separately by following step 4 for importing the certificate, then click on the **"Private Key"** tab where the corresponding private key can be imported.