

NetIQ Access Manager Template Condition Policy Readme

Policy extension examples include:

- PolicyConditionExtnFactoryTemplate.java
- PolicyConditionExtnTemplate.java

Purpose

You can use this example as a template to implement a policy extension of type *Condition* that is `com.novell.nxpe.NxpeCondition`. This example provides a basic framework that can be used as a starting point for creating such policy extensions.

It provides details about:

- How to configure and install a *Condition* policy extension in the Administration Console.
- Major implementation details of the Condition policy extension factory and extension classes.

Functionality

This policy extension performs a simple condition check on user's LDAP UserDN and returns *True* if it is not null or empty otherwise returns *False*. In this example, this *Condition* policy extension is used in an Access Gateway Authorization policy to decide whether to permit the access to a protected resource.

Implementation Details

- This *Condition Policy Extension* example consists of two Java classes:
 - PolicyConditionExtnFactoryTemplate.java:** Implements the `com.novell.nxpe.NxpeConditionFactory` interface. This factory creates the `PolicyConditionExtnTemplate` objects.
 - PolicyConditionExtnTemplate.java:** Implements the `com.novell.nxpe.NxpeCondition` interface. It contains the methods required to evaluate the condition.
- This class needs a configuration parameter for evaluating condition LDAP User DN of the user. Configuration parameters are set for a policy extension through the Administration Console. In the policy extension, the values for these configuration parameters are retrieved at the time of evaluation from the `NxpeInformationContext` object that is sent to it by the policy engine.
- The *evaluate* method in the `NxpeCondition` interface contains the condition evaluation logic and it is called by the policy engine when the condition extension needs to be evaluated for a policy. It executes the logic present in the policy extension and returns the results.
- The `getLDAPUserDN` method in `PolicyConditionExtnTemplate` is used to retrieve the value of the LDAP User DN configuration parameter from the `NxpeInformationContext` object.
- The `NxpeResult` class is used to return codes and the `NxpeException` class is used for

exceptions in PolicyConditionExtnTemplate.

- The Condition Policy extensions can be used in the Access Gateway Authorization policies and Identity Server Role policies. In this example the *Condition Policy* extension is used in a Access Gateway Authorization policy.

How to Customize This Example to Meet Your Needs?

- **Factory:** Modify the PolicyConditionExtnFactoryTemplate class to control the creation of PolicyConditionExtnTemplate as needed. Synchronize the methods of PolicyConditionExtnTemplate accordingly to avoid problems due to concurrency.
- **Configuration parameters:** Identify and set the required configuration parameters and provide their mappings in the Administration Console.
- **Implement getter methods** for all of your configuration parameters in the PolicyConditionExtnTemplate class as it has been done for retrieving LDAP User DN in this example.
- **Modify evaluate method** to contain your own condition evaluation logic in PolicyConditionExtnTemplate.
- **For creating condition extension for role policy**, create a policy extension of the type *Identity Server: Roles* from your customized extension classes and then create a Role policy that uses this extension. For more information, see [Section 2.2, Creating Role](#) in the NetIQ Access Manager 3.2 Policy Guide.

For more details on the implementation, see comments in the PolicyConditionExtnFactoryTemplate and PolicyConditionExtnTemplate classes or [NetIQ Access Manager 3.2 Developer Kit](#).

Installation

Before you start the installation, you need the following:

- The following Java classes and nxpe.jar:
NameAttributeFromMailID.java
NameAttributeFromMailIDFactory.java

Download novell-nacm3_2.tar.gz (SDK tar) from http://www.novell.com/developer/ndk/novell_access_manager_developer_tools_and_examples.html.

For information on how to obtain nxpe.jar, see section 4.1.1 Prerequisites in the NetIQ Access Manager Developer Kit 3.2.

- Java SDK (jdk1.6) to compile the Java sources.
- Apache Ant. You can download it from <http://ant.apache.org/bindownload.cgi>.
- An Access Manager setup with the Identity Server, Access Gateway, and policies.
- The user must have an email ID set in the user store.
- A basic knowledge of Java programming.

Setup

1. Extract the SDK tar (novell-nacm3_2.tar.gz). Go to nacm-3_2_0 > samples-jar folder. The PolicyExtensions.jar will be present at this location. This jar contains all the policy extension example's compiled classes. This jar can be used for this example. If you want to customize this example before installation, modify its java source, compile, and generate a Jar file.

Steps :

Go to nacm-3_2_0 > samples > PolicyExtension > TemplateConditionExtension_Example and run the build.xml.

This will generate the PolicyExtensions.jar in the same location where build.xml is available.

To run build.xml:

You need to copy nxpe.jar in the nacm-3_2 > samples > PolicyExtension > nxpe folder before running build.xml.

Note the location where Java and Ant are installed on your system. Set them in the Path environment variable and run the *ant dist* command. For example, run the following commands in the Linux environment:

- export ANT_HOME=/usr/apache-ant-1.8.1<set it as per your environment>
- export JAVA_HOME=/usr/java/jdk1.6.0_27<set it as per your environment>
- export PATH=\$ANT_HOME/bin:\$JAVA_HOME/bin:\$PATH
- ant all

2. Create a policy extension of the type *Access Gateway: Authorization*. For steps, see [Installing the Extension on the Administration Console](#) in the [NetIQ Access Manager 3.2 Policy Guide](#).

2.1. Create a new policy extension:

- Name: TemplatePolicyConditionExtension
- Type: Access Gateway: Authorization
- Class Name: com.novell.nam.custom.policy.condition.PolicyConditionExtnFactoryTemplate
- File name: name of the jar file created in step 1

New

Name:

Description:

Policy Type:

Type:

Class Name: (e.g. com.test.MyClass)

File Name:

Note: If the .jar file containing the class is not listed, you must upload it. To upload a .jar file, select Upload.

2.2. Policy extensions configuration parameters: Configuration parameters are sent to the policy extension for execution.

- Name: LDAP_User_DN (Name can be anything)
- ID: 41 (This must match the parameter ID as mentioned in the policy extension implementation in code)

Corresponding extract from PolicyDataExtnTemplate.java:

```
private static final String LDAP_USER_DN_NAME = "LDAP User DN";
private static final int EV_LDAP_USER_DN = 41;
```

This parameter has to be mapped to LDAP User DN of the user in the credential profile as shown in the following image:

NetIQ Access Manager
ADMIN
DS6_TULIKA_TREE

Access Manager | Devices | Policies | Auditing

Extensions ▾

Policy Extension: TemplatePolicyConditionExtension

Type: Condition

Policy Type: Access Gateway: Authorization

Description: A template Policy Condition Extension

Class Name: com.novell.nam.custom.policy.condition.PolicyConditionExtnFactoryTen

File Name: PolicyExtensions

Configuration Parameters:

<input type="checkbox"/> Name	ID	Mapping
<input type="checkbox"/> LDAP_User_DN	41	Authentication Contract:[Current] ▾

OK Cancel

- Distribute the policy extension. For the steps, see [“Distributing Policy Extension”](#). Create an authorization policy from the data extension, assign the policy to the Access Gateway, and distribute the policy extension created in step 3.

The *TemplateConditionPolicy* authorization policy created in this step will be assigned to a protected resource. For more information, see [Assigning an Authorization Policy to a Protected Resource](#). This policy will evaluate the TemplatePolicyConditionExtension – Policy “Condition” Extension created in step 2 and will decide whether to provide access to the protected resource based on the value of LDAP User DN.

New Access Gateway: Authorization policy

New

Name: TemplateConditionPolicy

Type: Access Gateway: Authorization

OK Cancel

Rules defined in this policy:

- Condition Group: Extension:TemplatePolicyConditionExtension
- Action: Permit

Access to the protected resource is provided if the user's LDAP User DN is not null or empty otherwise access will be denied.

The screenshot shows the NetIQ Access Manager interface. At the top, the user is logged in as ADMIN with the role DS6_TULIKA_TREE. The main navigation bar includes 'Access Manager', 'Devices', 'Policies', and 'Auditing'. The current page is 'Policies > Edit Policy > Edit Rule: TemplateConditionPolicy - Rule 1'.

Rule Details:

- Type: Access Gateway: Authorization
- Description: This is a simple sample conditon policy that returns "True" when LDAP UserDN is not nu
- Priority: 1

Conditions:

- Condition Group 1 (checked)
 - Operator: If
 - Condition Extension: TemplatePolicyConditionExtension
 - Result on Condition Error: False

Actions:

- Do: Permit

Changes made on this panel must be applied from the [Policies](#) Panel.

Buttons: OK, Cancel

Copyright © 2012 Novell, Inc. All Rights Reserved.

Novell grants permission, free of charge, to any person obtaining copies of this software and its associated documentation files (the "Software"), to deal in the Software without restriction, including to use, copy, adapt, publish, distribute, display, perform, sublicense, and sell copies of the Software, subject to the following condition: You must include the above copyright notice and this permission notice in all full or partial copies of the Software.

NOVELL PROVIDES THE SOFTWARE "AS IS," WITHOUT ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING WITHOUT THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. NOVELL, THE AUTHORS OF THE SOFTWARE, AND THE OWNERS OF COPYRIGHT IN THE SOFTWARE ARE NOT LIABLE FOR ANY CLAIM, DAMAGES, OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT, OR OTHERWISE, ARISING FROM, OUT OF, OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.