

User Guide for JBoss XACML

A Guide for Developers

Anil J. Saldhana

User Guide for JBoss XACML: A Guide for Developers

by Anil J. Saldhana

Table of Contents

Target Audience	v
Preface	vi
1. Introduction to JBoss XACML	1
Pre-requisites	1
2. JBoss XACML Configuration	2
3. Introduction to JBoss XACML PDP	6
PDP Construction with a Configuration File	6
PDP Construction with Object Model	6
4. Examples	8
Web Binding	8
5. Policy Enforcement Points (PEPs)	13
6. Dependencies for JBoss XACML	17
7. Acknowledgments for JBoss XACML	18

Target Audience

This guide is aimed at developers who want to use the JBoss XACML to implement complex Policy Infrastructure.

Preface

Commercial development support, production support and training for the JBoss XACML is available through JBoss Inc. [<http://www.jboss.com>] The JBoss XACML is a project of JBoss Security in the JEMS product suite.

Authors:

- Anil Saldhana - JBoss Security and Identity Management Project Lead

Chapter 1. Introduction to JBoss XACML

The JBoss XACML provides a standards based, robust Policy Infrastructure library .

JBossXACML is based on Oasis XACML v2.0. It defines an API to read one or more PolicySets (and/or Policy files) via a simple configuration file. It also provides a JAXB v2.0 compatible object model that can be used to construct policies and requests in XACML.

At its core, JBossXACML makes use of the standards compliant, BSD style licensed SunXACML library available at <http://sunxacml.sourceforge.net/>

Pre-requisites

To make full use of JBoss XACML, you will need a basic understanding of XACML v2.0, their policies etc

Chapter 2. JBoss XACML Configuration

An configuration file can be used to define the various policies and policy sets that the PDP can use for evaluation. The schema file driving the configuration is shown below.

```
<xs:schema>
<xs:element>
<xs:annotation>
<xs:documentation>Root Element for JBoss XACML</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:complexType>
<xs:sequence>
<xs:element></xs:element>
<xs:element></xs:element>
  </xs:sequence>
</xs:complexType>
<xs:element></xs:element>
<xs:complexType>
<xs:sequence>
<xs:element></xs:element>
<xs:element></xs:element>
  </xs:sequence>
</xs:complexType>
<xs:complexType>
<xs:sequence>
<xs:element></xs:element>
<xs:element></xs:element>
```

```
        </xs:sequence>
    </xs:complexType>

<xs:complexType>
<xs:sequence>
<xs:element></xs:element>
        </xs:sequence>
    </xs:complexType>

<xs:element></xs:element>

<xs:complexType>
<xs:sequence>
<xs:element></xs:element>
        </xs:sequence>
    </xs:complexType>

<xs:element>
<xs:complexType>
<xs:sequence>
<xs:any></xs:any>
        </xs:sequence>
</xs:element>

<xs:attribute></xs:attribute>
        </xs:complexType>
    </xs:element>

<xs:complexType>
<xs:sequence>
<xs:element></xs:element>
        </xs:sequence>
</xs:complexType>

<xs:attribute></xs:attribute>
        </xs:complexType>
</xs:schema>
```

JBoss XACML Configuration

The following is an example of a configuration file.

```
<ns:jbossdpd>
<ns:Policies>
<ns:PolicySet>
<ns:Location>test/policies/interop/xacml-policySet.xml</ns:Location>

<ns:Policy>
<ns:Location>test/policies/interop/xacml-policy2.xml</ns:Location>
  </ns:Policy>

<ns:Policy>
<ns:Location>test/policies/interop/xacml-policy3.xml</ns:Location>
  </ns:Policy>

<ns:Policy>
<ns:Location>test/policies/interop/xacml-policy4.xml</ns:Location>
  </ns:Policy>

<ns:Policy>
<ns:Location>test/policies/interop/xacml-policy5.xml</ns:Location>
  </ns:Policy>

  </ns:PolicySet>
</ns:Policies>

<ns:Locators>
<ns:Locator>
  </ns:Locator>
</ns:Locators>
</ns:jbossdpd>
```

JBoss XACML Configuration

As shown in the configuration file, you can define a policy set with enclosing policies. If you do not need any policy sets, but want to define 1 or more policies, you can do so. The location of a policy/policy set has to conform to the java.net.URL format (or you can provide a relative location as shown in the above configuration file).

The PDP makes use of PolicyLocator(s) for policies. Each of these policy locators are asked for a policy when a request is being evaluated. The default locator that is provided with JBossXACML is the **org.jboss.security.xacml.locators.JBossPolicySetLocator**

You will need to provide at least 1 locator.

Chapter 3. Introduction to JBoss XACML PDP

There is an implementation of the **org.jboss.security.xacml.interfaces.PolicyDecisionPoint** in JBoss called as **org.jboss.security.xacml.core.JBossPDP**. There are few variants of the construction of the JBossPDP as shown below.

```
public JBossPDP(InputStream configFile);
public JBossPDP(InputSource configFile);
public JBossPDP(Node configFile);
public JBossPDP(XMLStreamReader configFile);
public JBossPDP(URL configFileURL);
```

PDP Construction with a Configuration File

A PDP can be provided with policies via a configuration file or one can make use of the JAXB2 object model in JBoss XACML to construct the Policy/PolicySet instances and then provide them to the PDP. An example usage of the PDP with a configuration file is shown below:

```
import org.jboss.security.xacml.interfaces.PolicyDecisionPoint;
import org.jboss.security.xacml.core.JBossPDP;

ClassLoader tcl = Thread.currentThread().getContextClassLoader();
InputStream is =
    tcl.getResourceAsStream("test/config/interopPolicySetConfig.xml");
PolicyDecisionPoint pdp = new JBossPDP(is);
```

PDP Construction with Object Model

There is an object model provided with JBoss XACML in the package **org.jboss.security.xacml.core.model.policy**. This is the package that one will work with in the construction of elements associated with a Policy or PolicySet. There is a utility factory class that is provided in **org.jboss.security.xacml.factories.PolicyAttributeFactory** which can be used to create the AttributeValueTypes used in a Policy.

The interface **org.jboss.security.xacml.interfaces.XACMLPolicy** represents either a Policy or a PolicySet and is represented in the method in the PDP when you want to pass a set of pre-constructed instances, as shown below:

```
PolicyType policyType = constructPolicy();
PolicyDecisionPoint pdp = new JBossPDP();

XACMLPolicy policy = PolicyFactory.createPolicy(policyType);
Set<XACMLPolicy> policies = new HashSet<XACMLPolicy>();
policies.add(policy);

//Pass a set of policies (and/or policy sets) to the PDP
```

Introduction to JBoss XACML PDP

```
pdp.setPolicies(policies);
```

Chapter 4. Examples

The following sections display usage of JBossXACML in various bindings.

Web Binding

The Policy File that we will use as an example for the Web Binding Layer is shown below.

```
<Policy xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
  RuleCombiningAlgId=
    "urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides"
  Version="2.0" PolicyId="ExamplePolicy">
  <Target>
    <Resources>
      <Resource>
        <ResourceMatch
          MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue
            DataType=
              "http://www.w3.org/2001/XMLSchema#anyURI">http://test/developer-guide.html</Attrib
            <ResourceAttributeDesignator
              DataType=
                "http://www.w3.org/2001/XMLSchema#anyURI"
              AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"/>
            </ResourceMatch>
          </Resource>
        </Resources>
      </Target>
      <Rule Effect="Permit" RuleId="ReadRule">
        <Target>
          <Actions>
            <Action>
              <ActionMatch
                MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                <AttributeValue
                  DataType=
                    "http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
                <ActionAttributeDesignator
                  DataType="http://www.w3.org/2001/XMLSchema#string"
                  AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"/>
                </ActionMatch>
              </Action>
            </Actions>
          </Target>
          <Condition>
            <Apply
              FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-is-in">
              <AttributeValue
                DataType=
                  "http://www.w3.org/2001/XMLSchema#string">developer</AttributeValue>
              <SubjectAttributeDesignator
                DataType="http://www.w3.org/2001/XMLSchema#string"
                AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role" />
              </Apply>
            </Condition>
          </Rule>
        <!-- If none of the rules apply, deny the request -->
      </Policy>
```

Examples

```
<Rule Effect="Deny" RuleId="DenyRule"/>
</Policy>
```

This policy file basically provides access to the url when the subject has a role of "developer". All other requests are denied because of the explicit rule at the bottom of the policy file, without which the PDP would have returned a decision of NotAPPLICABLE.

If we would like to construct the policy dynamically, then we will need to use the Object Model provided with JBoss XACML that is based on JAXB v2.0 and above. The code listing is shown below:

```
import java.net.URI;
import java.security.Principal;
import java.security.acl.Group;
import java.util.HashSet;
import java.util.Set;

import javax.servlet.http.HttpServletRequest;
import javax.xml.bind.JAXBElement;

import junit.framework.TestCase;

import org.jboss.security.xacml.core.JBossPDP;
import org.jboss.security.xacml.core.model.policy.ActionMatchType;
import org.jboss.security.xacml.core.model.policy.ActionType;
import org.jboss.security.xacml.core.model.policy.ActionsType;
import org.jboss.security.xacml.core.model.policy.ApplyType;
import org.jboss.security.xacml.core.model.policy.AttributeValueType;
import org.jboss.security.xacml.core.model.policy.ConditionType;
import org.jboss.security.xacml.core.model.policy.EffectType;
import org.jboss.security.xacml.core.model.policy.ExpressionType;
import org.jboss.security.xacml.core.model.policy.FunctionType;
import org.jboss.security.xacml.core.model.policy.ObjectFactory;
import org.jboss.security.xacml.core.model.policy.PolicyType;
import org.jboss.security.xacml.core.model.policy.ResourceMatchType;
import org.jboss.security.xacml.core.model.policy.ResourceType;
import org.jboss.security.xacml.core.model.policy.ResourcesType;
import org.jboss.security.xacml.core.model.policy.RuleType;
import org.jboss.security.xacml.core.model.policy.SubjectAttributeDesignatorType;
import org.jboss.security.xacml.core.model.policy.TargetType;
import org.jboss.security.xacml.factories.PolicyAttributeFactory;
import org.jboss.security.xacml.factories.PolicyFactory;
import org.jboss.security.xacml.interfaces.PolicyDecisionPoint;
import org.jboss.security.xacml.interfaces.PolicyLocator;
import org.jboss.security.xacml.interfaces.RequestContext;
import org.jboss.security.xacml.interfaces.XACMLConstants;
import org.jboss.security.xacml.interfaces.XACMLPolicy;
import org.jboss.security.xacml.interfaces.XMLSchemaConstants;
import org.jboss.security.xacml.locators.JBossPolicyLocator;
import org.jboss.test.security.xacml.factories.util.XACMLTestUtil;

public void testWebBinding() throws Exception
{
    PolicyType policyType = constructPolicy();
    PolicyDecisionPoint pdp = new JBossPDP();

    XACMLPolicy policy = PolicyFactory.createPolicy(policyType);
```

Examples

```
Set<XACMLPolicy> policies = new HashSet<XACMLPolicy>();
policies.add(policy);

pdp.setPolicies(policies);

//Add the basic locators also
PolicyLocator policyLocator = new JBossPolicyLocator();
policyLocator.setPolicies(policies);
//Locators need to be given the policies

Set<PolicyLocator> locators = new HashSet<PolicyLocator>();
locators.add(policyLocator);
pdp.setLocators(locators);
assertNotNull("JBossPDP is != null", pdp);

Principal p = new Principal()
{
    public String getName()
    {
        return "testuser";
    }
};

//Create Role Group
Group grp = XACMLTestUtil.getRoleGroup("developer");

String requestURI = "http://test/developer-guide.html";
HttpRequestUtil util = new HttpRequestUtil();
HttpServletRequest req = util.createRequest(p, requestURI);

//Check PERMIT condition
WebPEP pep = new WebPEP();
RequestContext request = pep.createXACMLRequest(req, p, grp);
if(debug)
    request.marshall(System.out);

    assertEquals("Access Allowed?",
XACMLConstants.DECISION_PERMIT,
        XACMLTestUtil.getDecision(pdp,request));
}

public void testNegativeAccessWebBinding() throws Exception
{
    PolicyType policyType = constructPolicy();
    PolicyDecisionPoint pdp = new JBossPDP();

    XACMLPolicy policy = PolicyFactory.createPolicy(policyType);
    Set<XACMLPolicy> policies = new HashSet<XACMLPolicy>();
    policies.add(policy);

    pdp.setPolicies(policies);

    //Add the basic locators also
    PolicyLocator policyLocator = new JBossPolicyLocator();
    //Locators need to be given the policies
    policyLocator.setPolicies(policies);

    Set<PolicyLocator> locators = new HashSet<PolicyLocator>();
    locators.add(policyLocator);
    pdp.setLocators(locators);
    assertNotNull("JBossPDP is != null", pdp);

    Principal p = new Principal()
```

Examples

```
{
    public String getName()
    {
        return "testuser";
    }
};

//Create Role Group
Group grp = XACMLTestUtil.getRoleGroup("imposter");
String requestURI = "http://test/developer-guide.html";
HttpRequestUtil util = new HttpRequestUtil();
HttpServletRequest req = util.createRequest(p, requestURI);

//Check DENY condition
WebPEP pep = new WebPEP();
RequestContext request = pep.createXACMLRequest(req, p, grp);
if(debug)
    request.marshall(System.out);

assertEquals("Access Disallowed?", XACMLConstants.DECISION_DENY,
    XACMLTestUtil.getDecision(pd, request));
}

private PolicyType constructPolicy() throws Exception
{
    ObjectFactory objectFactory = new ObjectFactory();

    String PERMIT_OVERRIDES=
"urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides";
    PolicyType policyType = new PolicyType();
    policyType.setPolicyId("ExamplePolicy");
    policyType.setVersion("2.0");
    policyType.setRuleCombiningAlgId(PERMIT_OVERRIDES);

    //Create a target
    TargetType targetType = new TargetType();

    ResourceType resourcesType = new ResourceType();
    ResourceType resourceType = new ResourceType();
    ResourceMatchType rmt = new ResourceMatchType();
    rmt.setMatchId(XACMLConstants.FUNCTION_ANYURI_EQUALS);
    rmt.setResourceAttributeDesignator(
PolicyAttributeFactory.createAttributeDesignatorType(
    XACMLConstants.RESOURCE_IDENTIFIER,
    XMLSchemaConstants.DATATYPE_ANYURI));
    rmt.setAttributeValue(PolicyAttributeFactory.createAnyURIAttributeType(
        new URI("http://test/developer-guide.html")));
    resourceType.getResourceMatch().add(rmt);
    resourcesType.getResource().add(resourceType);

    targetType.setResources(resourcesType);

    policyType.setTarget(targetType);

    //Create a Rule
    RuleType permitRule = new RuleType();
    permitRule.setRuleId("ReadRule");
    permitRule.setEffect(EffectType.PERMIT);

    ActionType permitRuleActionType = new ActionType();
    ActionMatchType amct = new ActionMatchType();
```

Examples

```
        amct.setMatchId("urn:oasis:names:tc:xacml:1.0:function:string-equal");
        amct.setAttributeValue(
            PolicyAttributeFactory.createStringAttributeType("read"));
        amct.setActionAttributeDesignator(
            PolicyAttributeFactory.createAttributeDesignatorType(
                XACMLConstants.ACTION_IDENTIFIER,
                XMLSchemaConstants.DATATYPE_STRING));
        permitRuleActionType.getActionMatch().add(amct);
        TargetType permitRuleTargetType = new TargetType();
        permitRuleActionsType.getAction().add(permitRuleActionType);
        permitRuleTargetType.setActions(permitRuleActionsType);
        permitRule.setTarget(permitRuleTargetType);

        ConditionType permitRuleConditionType = new ConditionType();
        FunctionType functionType = new FunctionType();
        functionType.setFunctionId(XACMLConstants.FUNCTION_STRING_EQUAL);
        JAXBElement<ExpressionType> jaxbElementFunctionType =
            objectFactory.createExpression(functionType);
        permitRuleConditionType.setExpression(jaxbElementFunctionType);

        ApplyType permitRuleApplyType = new ApplyType();
        permitRuleApplyType.setFunctionId(XACMLConstants.FUNCTION_STRING_IS_IN);

        SubjectAttributeDesignatorType sadt =
            PolicyAttributeFactory.createSubjectAttributeDesignatorType(
                XACMLConstants.SUBJECT_ROLE_IDENTIFIER,
                XMLSchemaConstants.DATATYPE_STRING);
        JAXBElement<SubjectAttributeDesignatorType> sadtElement =
            objectFactory.createSubjectAttributeDesignator(sadt);
        AttributeValueType avt =
            PolicyAttributeFactory.createStringAttributeType("developer");
        JAXBElement<AttributeValueType> jaxbAVT =
            objectFactory.createAttributeValue(avt);
        permitRuleApplyType.getExpression().add(jaxbAVT);
        permitRuleApplyType.getExpression().add(sadtElement);

        permitRuleConditionType.setExpression(
            objectFactory.createApply(permitRuleApplyType));

        permitRule.setCondition(permitRuleConditionType);

        policyType.
            getCombinerParametersOrRuleCombinerParametersOrVariableDefinition().
            add(permitRule);
        //Create a Deny Rule
        RuleType denyRule = new RuleType();
        denyRule.setRuleId("DenyRule");
        denyRule.setEffect(EffectType.DENY);
        policyType.
            getCombinerParametersOrRuleCombinerParametersOrVariableDefinition().
            add(denyRule);

        return policyType;
    }
}
```

Chapter 5. Policy Enforcement Points (PEPs)

PEPs are an important component of any Policy based infrastructure. PEPs are needed to construct the policy requests to be passed to the PDPs for evaluation.

JBossXACML provides an object model that is based on JAXB v2.0 and later, to construct XACML Policy requests. The package **org.jboss.security.xacml.core.model.context** contains the object classes. Please make a note of this package used for building request and responses. This is different from the package used for policy construction. There may be types that have the same name in the two different packages.

There is a utility factory class that can be used to construct the various attributes of the Request. The class is **org.jboss.security.xacml.factories.RequestAttributeFactory**.

An example of construction of XACML Request is shown below:

```
import java.io.InputStream;

import junit.framework.TestCase;

import org.jboss.security.xacml.core.JBossPDP;
import org.jboss.security.xacml.core.model.context.ActionType;
import org.jboss.security.xacml.core.model.context.AttributeType;
import org.jboss.security.xacml.core.model.context.EnvironmentType;
import org.jboss.security.xacml.core.model.context.RequestType;
import org.jboss.security.xacml.core.model.context.ResourceType;
import org.jboss.security.xacml.core.model.context.SubjectType;
import org.jboss.security.xacml.factories.RequestAttributeFactory;
import org.jboss.security.xacml.factories.RequestResponseContextFactory;
import org.jboss.security.xacml.interfaces.PolicyDecisionPoint;
import org.jboss.security.xacml.interfaces.RequestContext;
import org.jboss.security.xacml.interfaces.XACMLConstants;
import org.jboss.test.security.xacml.factories.util.XACMLTestUtil;

public void testInteropTestWithObjects() throws Exception
{
    ClassLoader tcl = Thread.currentThread().getContextClassLoader();
    InputStream is =
tcl.getResourceAsStream("test/config/interopPolicySetConfig.xml");
    assertNotNull("InputStream != null", is);
    PolicyDecisionPoint pdp = new JBossPDP(is);
    assertNotNull("JBossPDP is != null", pdp);

    assertEquals("Case 1 should be deny", XACMLConstants.DECISION_DENY,
        XACMLTestUtil.getDecision(pdp,
            getRequestContext("false", "false", 10)));
    assertEquals("Case 2 should be deny", XACMLConstants.DECISION_PERMIT,
        XACMLTestUtil.getDecision(pdp,
            getRequestContext("false", "false", 1)));
    assertEquals("Case 3 should be deny", XACMLConstants.DECISION_PERMIT,
        XACMLTestUtil.getDecision(pdp,
            getRequestContext("true", "false", 5)));
    assertEquals("Case 4 should be deny", XACMLConstants.DECISION_DENY,
        XACMLTestUtil.getDecision(pdp,
            getRequestContext("false", "false", 9)));
    assertEquals("Case 5 should be deny", XACMLConstants.DECISION_DENY,
```

Policy Enforcement Points (PEPs)

```
        XACMLTestUtil.getDecision(pdp,
getRequestContext("true", "false", 10));
        assertEquals("Case 6 should be deny", XACMLConstants.DECISION_DENY,
        XACMLTestUtil.getDecision(pdp,
getRequestContext("true", "false", 15));
        assertEquals("Case 7 should be deny", XACMLConstants.DECISION_PERMIT,
        XACMLTestUtil.getDecision(pdp,
getRequestContext("true", "true", 10));
    }

private RequestContext getRequestContext(String reqTradeAppr, String reqCreditAppr,
int buyPrice) throws Exception
{
    RequestType request = new RequestType();
    request.getSubject().add(
createSubject(reqTradeAppr, reqCreditAppr, buyPrice));
    request.getResource().add(createResource());
    request.setAction(createAction());
    request.setEnvironment(new EnvironmentType());

    RequestContext requestCtx =
RequestResponseContextFactory.createRequestCtx();
    requestCtx.setRequest(request);
    if(debug)
        requestCtx.marshall(System.out);

    return requestCtx;
}

private SubjectType createSubject(String reqTradeAppr,
String reqCreditAppr,
int buyPrice)
{
    //Create a subject type
    SubjectType subject = new SubjectType();
    subject.setSubjectCategory(
"urn:oasis:names:tc:xacml:1.0:subject-category:access-subject");
    //create the subject attributes
    AttributeType attSubjectID =
RequestAttributeFactory.createStringAttributeType(
        "urn:oasis:names:tc:xacml:1.0:subject:subject-id",
        "xacml20.interop.com", "123456");
    subject.getAttribute().add(attSubjectID);

    AttributeType attUserName =
RequestAttributeFactory.createStringAttributeType(
        "urn:xacml:2.0:interop:example:subject:user-name",
        "xacml20.interop.com", "John Smith");
    subject.getAttribute().add(attUserName);

    AttributeType attBuyNumShares =
RequestAttributeFactory.createIntegerAttributeType(
        "urn:xacml:2.0:interop:example:subject:buy-num-shares",
        "xacml20.interop.com", 1000);
    subject.getAttribute().add(attBuyNumShares);

    AttributeType attBuyOfferShare =
RequestAttributeFactory.createIntegerAttributeType(
        "urn:xacml:2.0:interop:example:subject:buy-offer-price",
        "xacml20.interop.com", buyPrice);
    subject.getAttribute().add(attBuyOfferShare);
}
```

Policy Enforcement Points (PEPs)

```
        AttributeType attRequestExtCred =
RequestAttributeFactory.createStringAttributeType(
    "urn:xacml:2.0:interop:example:subject:req-credit-ext-approval",
    "xacml20.interop.com", reqCreditAppr);
    subject.getAttribute().add(attRequestExtCred);

        AttributeType attRequestTradeApproval =
RequestAttributeFactory.createStringAttributeType(
    "urn:xacml:2.0:interop:example:subject:req-trade-approval",
    "xacml20.interop.com", reqTradeAppr);
    subject.getAttribute().add(attRequestTradeApproval);

    return subject;
}

public ResourceType createResource()
{
    ResourceType resourceType = new ResourceType();

        AttributeType attResourceID =
RequestAttributeFactory.createStringAttributeType(
    "urn:oasis:names:tc:xacml:1.0:resource:resource-id",
    "xacml20.interop.com", "CustomerAccount");
    resourceType.getAttribute().add(attResourceID);

        AttributeType attOwnerID =
RequestAttributeFactory.createStringAttributeType(
    "urn:xacml:2.0:interop:example:resource:owner-id",
    "xacml20.interop.com", "123456");
    resourceType.getAttribute().add(attOwnerID);

        AttributeType attOwnerName =
RequestAttributeFactory.createStringAttributeType(
    "urn:xacml:2.0:interop:example:resource:owner-name",
    "xacml20.interop.com", "John Smith");
    resourceType.getAttribute().add(attOwnerName);

        AttributeType attAccountStatus =
RequestAttributeFactory.createStringAttributeType(
    "urn:xacml:2.0:interop:example:resource:account-status",
    "xacml20.interop.com", "Active");
    resourceType.getAttribute().add(attAccountStatus);

        AttributeType attCreditLine =
RequestAttributeFactory.createIntegerAttributeType(
    "urn:xacml:2.0:interop:example:resource:credit-line",
    "xacml20.interop.com", 15000);
    resourceType.getAttribute().add(attCreditLine);

        AttributeType attCurrentCredit =
RequestAttributeFactory.createIntegerAttributeType(
    "urn:xacml:2.0:interop:example:resource:current-credit",
    "xacml20.interop.com", 10000);
    resourceType.getAttribute().add(attCurrentCredit);

        AttributeType attTradeLimit =
RequestAttributeFactory.createIntegerAttributeType(
    "urn:xacml:2.0:interop:example:resource:trade-limit",
    "xacml20.interop.com", 10000);
    resourceType.getAttribute().add(attTradeLimit);
    return resourceType;
}
```

Policy Enforcement Points (PEPs)

```
private ActionType createAction()
{
    ActionType actionType = new ActionType();
    AttributeType attActionID =
RequestAttributeFactory.createStringAttributeType(
    "urn:oasis:names:tc:xacml:1.0:action:action-id",
    "xacml20.interop.com", "Buy");
    actionType.getAttribute().add(attActionID);
    return actionType;
}
```

Chapter 6. Dependencies for JBoss XACML

JBoss XACML needs the following as dependencies.

1. JDK 5.0 or above
2. JBoss-JavaEE 5.0 (This basically provides javax.xml.stream support which exists in JDK6)
3. JAXB2 libraries (JAXB-API, JAXB-IMPL)
4. SunXACML 2.0

Chapter 7. Acknowledgments for JBoss XACML

This is an acknowledgment that JBoss XACML software contains SunXACML(BSD style licensed) with license as follows.

Copyright 2003 Sun Microsystems, Inc. All Rights Reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistribution of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistribution in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Sun Microsystems, Inc. or the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided "AS IS," without a warranty of any kind. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN MICROSYSTEMS, INC. ("SUN") AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You acknowledge that this software is not designed or intended for use in the design, construction, operation or maintenance of any nuclear facility.