XACML Policy Test Case:

- Access Control Policy for XML Resource -

December 16, 2001

Author: Michiharu Kudo

USE CASE:

An example XML instance is an order request document sent from ABC company to XYZ company and it is stored in the repository of the XYZ company. It contains sensitive information under <Secret> element. A security administrator in XYZ company wants to specify the policy to hide <Secret> element whenever the access to this document comes from outside the company.

```
<OrderRequest>
<Header>
  <From>
    <CompanyName>ABC</CompanyName>
    <Secret>
      <mode>E10</mode>
      <authcode>opensesame</authcode>
    </Secret>
  </From>
  <To><CompanyName>XYZ</CompanyName></To>
</Header>
<Body>
  <Total><Money>187.60</Money></Total>
</Body>
</OrderRequest>
```

We assume that there are two roles here, InternalUser and ExternalUser. The access control policy he needs to specify is the following:

- When an internal user submits an access request, the PDP says that the requester can read every element under the root node.
- When an external user submits an access request, the PDP MUST say that he cannot read any element below the <Secret> element. As a result, the external user receives the following XML fragment from the PEP.

```
<OrderRequest>
<Header>
  <From><CompanyName>ABC</CompanyName></From>
  <To><CompanyName>XYZ</CompanyName></To>
</Header>
<Body>
  <Total><Money>187.60</Money></Total>
</Body>
</OrderRequest>
```

Considering above intention, the PDP returns the access decision in response to the access request as follows:

Access Request 1: Can an InternalUser read the OrderRequest XML instance?
Access Decision 1: Yes.

Access Request 2: Can an ExternalUser read the OrderRequest XML instance?
Access Decision 2: Yes, but he can read only "/OrderRequest/Header/From/CompanyName", "/OrderRequest/Header/To/CompanyName", and "/OrderRequest/Body/Total/Money" but not others

To be consistent with the SAML authorization decision assertion, the additional decision information such as "he can read only ..." is stored in the SAML <Conditions> element. We assume here that the PEP understands the description specified in the <Conditions> element and it filters disallowed elements out when it returns the requested XML instance to the requester.

POLICY SPECIFICATION IN SIMPLIFIED WAY

We describe how to specify the above policy and present three manners here (note that this is not exhaustive.) We assume that the requested action is read.

(1) Grant only policy: specify the positive permissions on each readable element.

(2) Grant only policy based on the algorithm that supports a permission propagation through XML structure. In the case of this policy evaluation algorithm, if a positive permission is specified on the specific element X, it implies that all the descendant elements under X are allowed to access.

(3) Grant and deny policy based on the previous propagation algorithm: If a negative permission is specified on the specific element X, it implies that all the descendant elements under X are disallowed to access. If any conflicts occur (that means both the grant and the deny permissions hold at the same time on the same element), the denial permission takes precedence.

We first describe each policy in simplified manner.

(1) Grant Only Policy

```
IF   (principal == "InternalUser") AND
     ((resource.element == "/OrderRequest/Header/From/CompanyName") OR
      (resource.element == "/OrderRequest/Header/From/Secret/mode") OR
      (resource.element == "/OrderRequest/Header/From/Secret/authcode") OR
      (resource.element == "/OrderRequest/Header/To/CompanyName") OR
      (resource.element == "/OrderRequest/Body/Total/Money"))
THEN grant

IF   (principal == "ExternalUser") AND
     ((resource.element == "/OrderRequest/Header/From/CompanyName") OR
      (resource.element == "/OrderRequest/Header/To/CompanyName") OR
      (resource.element == "/OrderRequest/Body/Total/Money"))
THEN grant
```

The first rule lists every element. The second rule lists only readable elements. The problem of this approach is that if the number of elements becomes greater, it would not be wise to specify each element in the policy. Another problem is that it would become unclear that the current set of rules enforce the secrecy policy that the <Secret> sub-tree is not disclosed to any ExternalUser.

(2) Grant Only Policy (based on the propagation algorithm):

```
IF (principal == "InternalUser") AND
   (resource.element == "/")
THEN grant

IF   (principal == "ExternalUser") AND
     ((resource.element == "/OrderRequest/Header/From/CompanyName") OR
      (resource.element == "/OrderRequest/Header/To/CompanyName") OR
      (resource.element == "/OrderRequest/Body/Total/Money"))
THEN grant
```

The point of this approach is that the policy does not have to list every element that is allowed to access. The first rule implies that every element under the root node ("/") is readable by the InternalUser. The certain algorithm handles the permission propagation computation toward the leaf elements.

(3) Grant and Deny Policy (based on the propagation algorithm):

```
IF (principal == "InternalUser") AND
```

```
    (resource.element == "/")
THEN grant

IF (principal == "ExternalUser") AND
    (resource.element == "/")
THEN grant

IF (principal == "ExternalUser") AND
    (resource.element == "/OrderRequest/Header/From/Secret")
THEN deny
```

The second rule says that the ExternalUser is allowed to read every node below the root node, while the third rule says that the ExternalUser is not allowed to read any node below the <Secret> element. In the propagation algorithm, the conflict resolution occurs and the access to the <Secret>, <mode>, and <authcode> elements is determined as denial because of the denial takes precedence policy.

REQUIREMENTS:

The above Use Case imposes the following requirements on an access control policy language.

1. Ability to describe a grant-and-denial policy as well as a grant-only policy, or ability to provide the language syntax and the semantics with extensibility and flexibility for satisfying the description in the XACML charter section.

2. Ability to use a specific semantic basis (or its implementation) defined by the user to make an access decision.

(The user-defined semantic basis means a high-level description of the semantics as described in this use case. We assume that the user implements the algorithm. Those implementations should be considered as a valid XACML processor if it supports a mandatory XACML standard algorithm. If those extended algorithms are important to certain application area, it should be included in the standard specification in addition to the mandatory one.)

3. Ability to support the triple-based syntax like <principal, resource, action> IN ADDITION TO the current flat-structure based syntax.

(It is reasonable to claim this because the policy used in this use case simply consists of the triple <principal, resource, (action)>, which represents the notion of the access control policy very well.

Note that this is not saying that the triple-based syntax replaces the current Boolean-based specification. Both can coexist.)

COMPARISON OF POLICY SPECIFICATION

We compare expressiveness of the policy specification in order to explain the intention for the requirement 1 and 3. We first describe the policy for the first policy example (Grant Only Policy) based on the current specification proposal. Then we describe the third policy example (Grant and Deny Policy) based on the current specification proposal. Lastly, we describe the third policy (Grant and Deny Policy) based on possible triple-like syntax just for feeling an appearance. (The applicability elements are omitted. Parameters might be wrongly specified.)

A. Policy specification for the Grant Only Policy example based on the current specification proposal

```
<policy>
  <rule><preCondition>
  <or>
    <rule><preCondition>
    <and>
      <rule><preCondition>
        <predicate>
          <equality>
            <referencedData>
              <roleAttribute>/saml/Attribute/AttributeName/Role</roleAttribute>
            </referencedData>
            <secondOperand>
              <hardcodedValue>InternalUser</hardcodedValue>
            </secondOperand>
          </subsetOf>
        </predicate>
      </preCondition></rule>
      <rule><preCondition>
        <or>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>"/OrderRequest/Header/From/CompanyName"</hardcodedValue>
                </secondOperand>
              </equality>
            </predicate>
          </preCondition></rule>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>"/OrderRequest/Header/From/Secret/mode"</hardcodedValue>
                </secondOperand>
              </equality>
```

```
          </predicate>
        </preCondition></rule>
        <rule><preCondition>
          <predicate>
            <equality>
              <referencedData>
                <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
              </referencedData>
              <secondOperand>
                <hardcodedValue>"/OrderRequest/Header/From/Secret/authcode"</hardcodedValue>
              </secondOperand>
            </equality>
          </predicate>
        </preCondition></rule>
        <rule><preCondition>
          <predicate>
            <equality>
              <referencedData>
                <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
              </referencedData>
              <secondOperand>
                <hardcodedValue>"/OrderRequest/Header/To/CompanyName"</hardcodedValue>
              </secondOperand>
            </equality>
          </predicate>
        </preCondition></rule>
        <rule><preCondition>
          <predicate>
            <equality>
              <referencedData>
                <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
              </referencedData>
              <secondOperand>
                <hardcodedValue>"/OrderRequest/Body/Total/Money"</hardcodedValue>
              </secondOperand>
            </equality>
          </predicate>
        </preCondition></rule>
      </or>
    </preCondition></rule>
  <and>
  </preCondition></rule>
  <rule><preCondition>
  <and>
    <rule><preCondition>
      <predicate>
        <equality>
          <referencedData>
            <roleAttribute>/saml/Attribute/AttributeName/Role</roleAttribute>
          </referencedData>
          <secondOperand>
            <hardcodedValue>ExternalUser</hardcodedValue>
          </secondOperand>
        </subsetOf>
      </predicate>
    </preCondition></rule>
    <rule><preCondition>
      <or>
        <rule><preCondition>
          <predicate>
            <equality>
              <referencedData>
                <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
              </referencedData>
              <secondOperand>
```

```
                    <hardcodedValue>"/OrderRequest/Header/From/CompanyName"</hardcodedValue>
                  </secondOperand>
                </equality>
              </predicate>
            </preCondition></rule>
            <rule><preCondition>
              <predicate>
                <equality>
                  <referencedData>
                    <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
                  </referencedData>
                  <secondOperand>
                    <hardcodedValue>"/OrderRequest/Header/To/CompanyName"</hardcodedValue>
                  </secondOperand>
                </equality>
              </predicate>
            </preCondition></rule>
            <rule><preCondition>
              <predicate>
                <equality>
                  <referencedData>
                    <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
                  </referencedData>
                  <secondOperand>
                    <hardcodedValue>"/OrderRequest/Body/Total/Money"</hardcodedValue>
                  </secondOperand>
                </equality>
              </predicate>
            </preCondition></rule>
          </or>
        </preCondition></rule>
      <and>
      </preCondition></rule>
    </or>
    </preCondition></rule>
</preCondition></rule>
</policy>
```

B. Policy specification for the Grant And Deny Policy example based on the current specification proposal.

```
<policy>
   <rule><preCondition>
   <or>
        <rule><preCondition>
        <and>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <roleAttribute>/saml/Attribute/AttributeName/Role</roleAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>InternalUser</hardcodedValue>
                </secondOperand>
              </equality>
            </predicate>
          </preCondition></rule>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
```

```
                </referencedData>
                <secondOperand>
                  <hardcodedValue>"/"</hardcodedValue>
                </secondOperand>
              </equality>
            </predicate>
          </preCondition></rule>
        </and>
        </preCondition></rule>
        <rule><preCondition>
        <and>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <roleAttribute>/saml/Attribute/AttributeName/Role</roleAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>ExternalUser</hardcodedValue>
                </secondOperand>
              </equality>
            </predicate>
          </preCondition></rule>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>"/"</hardcodedValue>
                </secondOperand>
              </equality>
            </predicate>
          </preCondition></rule>
        </and>
        </preCondition></rule>
</or>
</preCondition></rule>
<rule type=”deny”><preCondition>
<and>
        <rule><preCondition>
        <and>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <roleAttribute>/saml/Attribute/AttributeName/Role</roleAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>ExternalUser</hardcodedValue>
                </secondOperand>
              </equality>
            </predicate>
          </preCondition></rule>
          <rule><preCondition>
            <predicate>
              <equality>
                <referencedData>
                  <classificationAttribute>/saml/Request/AuthorizationQuery/Object</classificationAttribute>
                </referencedData>
                <secondOperand>
                  <hardcodedValue>"/OrderRequest/Header/From/Secret"</hardcodedValue>
                </secondOperand>
              </equality>
```

```
            </predicate>
          </preCondition></rule>
        </and>
        </preCondition></rule>
      </and>
      </preCondition></rule>
</policy>
```

C. A Sample policy specification for the Grant And Deny Policy example based on the TRIPLE-BASED policy specification. This includes several modifications and additions to the currently proposed language syntax.

```
<policy>
<grant>
   <rule>
      <principal>
         <and>
            <equality type=”roleAttribute” value=”InternalUser”/>
         </and>
      </principal>
      <resource>
         <and>
            <equality type=”classificationAttribute” value=”/”/>
         </and>
      </resource>
   </rule>
   <rule>
      <principal>
         <and>
            <equality type=”roleAttribute” value=”ExternalUser”/>
         </and>
      </principal>
      <resource>
         <and>
            <equality type=”classificationAttribute” value=”/”/>
         </and>
      </resource>
   </rule>
</grant>
<deny>
   <rule>
      <principal>
         <and>
            <equality type=”roleAttribute” value=”ExternalUser”/>
         </and>
      </principal>
      <resource>
         <and>
            <equality type=”classificationAttribute” value=”/OrderRequest/Header/From/Secret”/>
         </and>
      </resource>
   </rule>
</deny>
</policy>
```