XACML and Discovery Services

04 / 21 / 2011

GREYC Laboratory

Jérôme Le Moulec
Jacques Madelaine
Adrien Laurence

XACML Generalities

XACML (eXtensible Access Control Markup Language) :

- declarative **access control policy** language
- implemented in **XML**
- Current **version : 2.0** (February 2005)
- Draft XACML 3.0 in April 2009

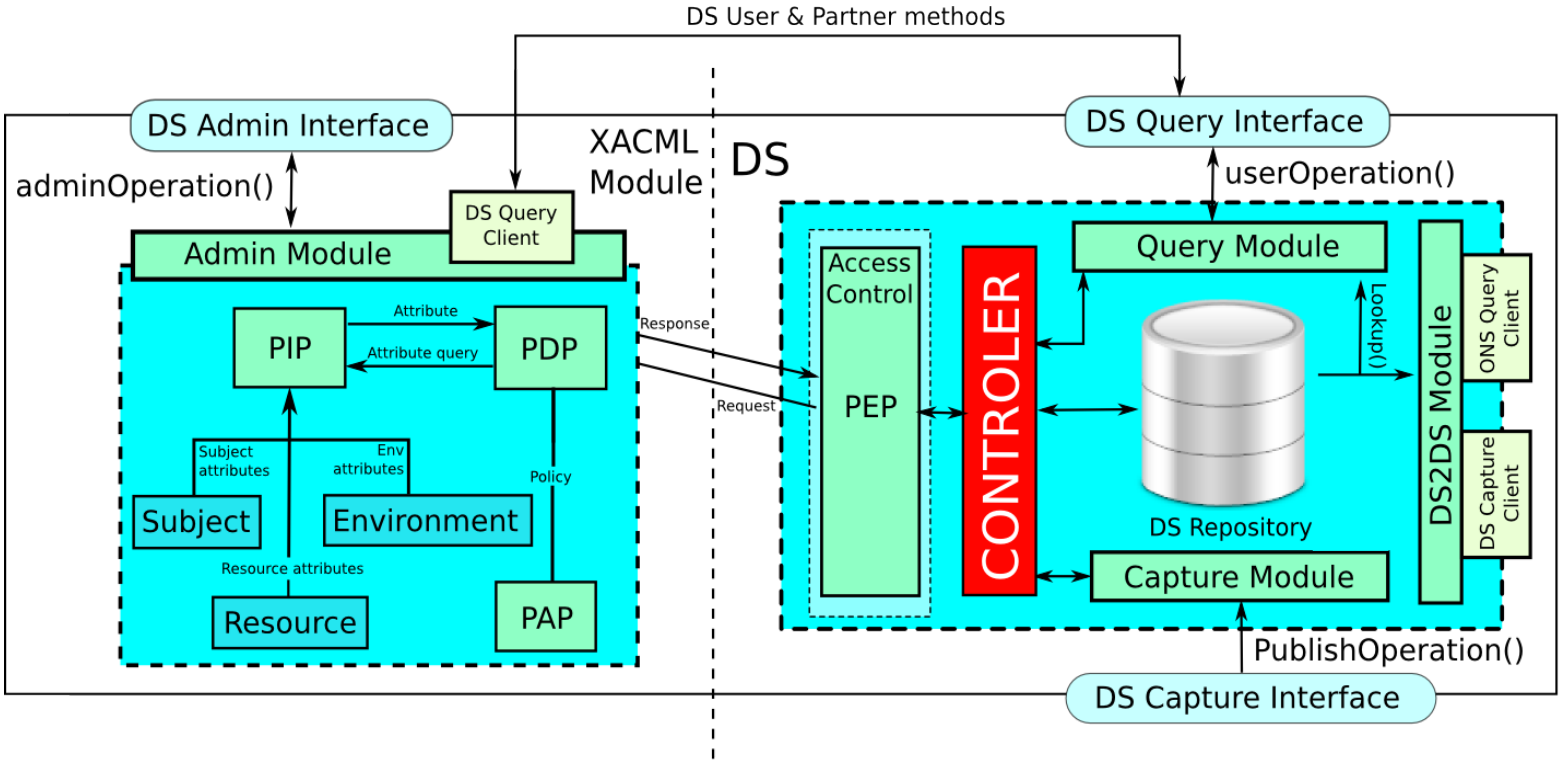
XACML Components

- **PAP:** Policy Administration Point - Point which manages policies
- **PDP:** Policy Decision Point - Point which evaluates and issues authorization decisions
- **PEP:** Policy Enforcement Point - Point which intercepts user's access request to a resource and enforces PDP's decision.
- **PIP:** Policy Information Point - Point which can provide external information to a PDP, such as LDAP attribute information.

XACML Policies

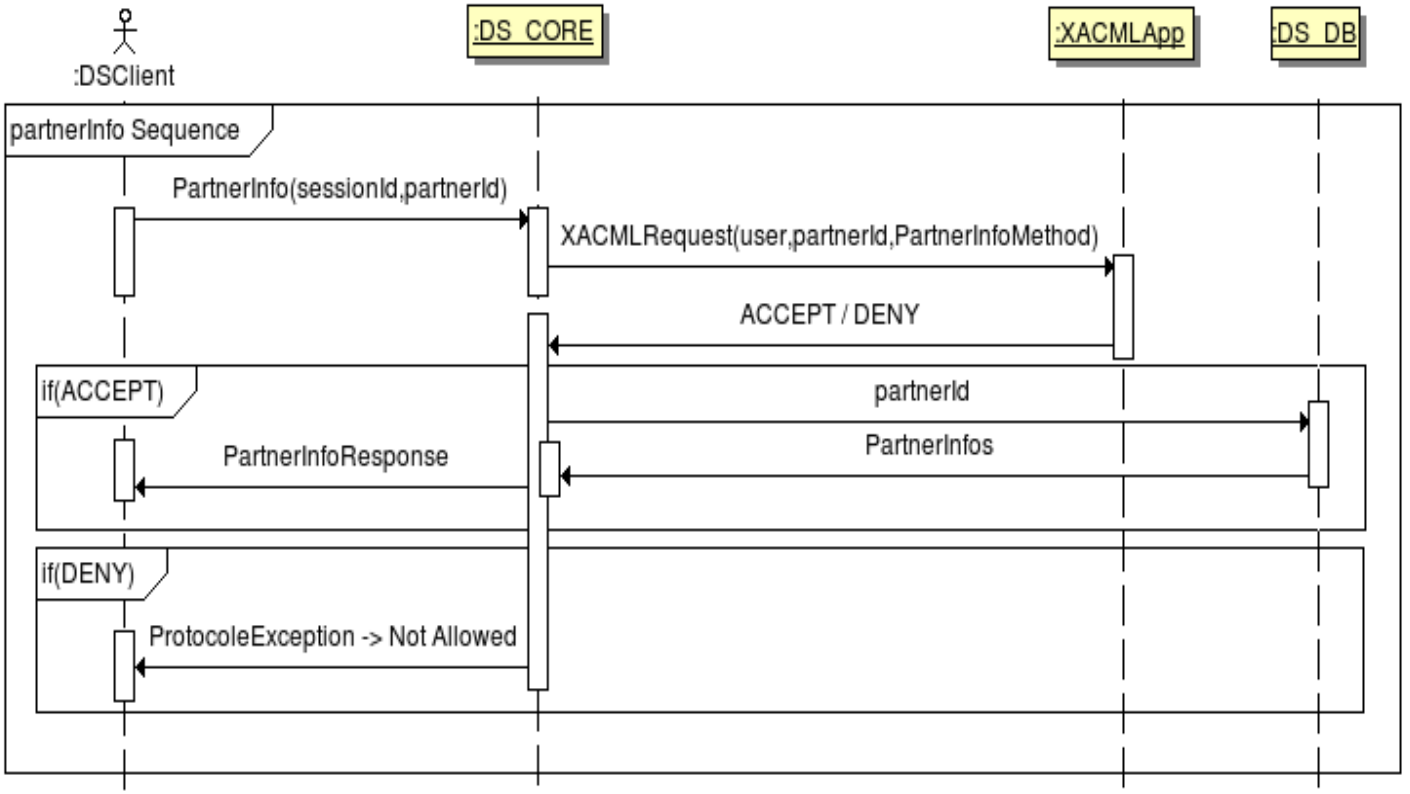
- **Policy**: comprises four main components:
  - a target;
  - a rule-combining algorithm-identifier;
  - a set of rules;
  - obligations.
- **Policy Set**: comprises four main components:
  - a target;
  - a policy-combining algorithm-identifier;
  - a set of policies;
  - obligations.

DS architecture using XACML



- Two different applications
  - XACML module → process XACML requests using policy repository;
  - DS core → create and send XACML request and process DS requests.
- Connected over the IP network;
  - DS application → send XACML request to the XACML module;
  - XACML module → send back ACCEPT or DENY.

DS XACML request processing



- User call PartnerInfo method on the DS application;
- DS application send corresponding XACML request to the XACML application;
- XACML application process the request with the corresponding XACML policy file and send back ACCEPT or DENY as response;
- DS application process the PartnerInfo method and send back the result or throw a protocolException

XACML for DS access control

XACML Policies for DS access control

- **DS User Group :**
  - a set of method signature
  - a set of users
  - a set of resource filters (Query & Capture)

- **Partner access control :**
  - a set of DS user groups

XACML Policies for DS access control

- **Policy (A DS USER GROUP)** comprises four main components:
  - a target → a set of method signature
  - a rule-combining algorithm-identifier → group-rule
  - a set of rules
    - One associating users to this group (RuleID=UserGroup)
    - Four specifying filters (only for query and capture modules).
  - Obligations. → currently not used.
- **Policy Set (PARTNER ACCES CONTROL)** comprises four main components:
  - a target
    - moduleID (Query, Capture or Admin);
    - partnerID : owner of the policy.
  - a policy-combining algorithm-identifier → permit-overrides
  - a set of policies → user group definitions
  - obligations → currently not used.

XACML PolicySet for DS AC

A policy file describes :

- **The module** (QueryModule,CaptureModule,AdminModule) depending on the file location.
- A list of policies defining a **user group** list.

```
<PolicySet PolicySetId="epcistest" PolicyCombiningAlgId="permit-overrides">
    <Target>
        <!-- MODULE AND OWNER POLICY -->
    </Target>
    <Policy>
        <!-- A GROUP DEFINITION -->
    </Policy>
    <Policy>
        <!-- AN OTHER GROUP DEFINITION -->
    </Policy>
    ...
</PolicySet>
```

XACML PolicySet Target for DS AC

The header defining the **owner** of the policy and to wich **module** it is applyed

```
<PolicySet policySetId="epcistest" PolicyCombiningAlgId="permit-overrides">
  <Target>

    <Subjects>   <!-- MODULE -->
     <SubjectMatch MatchId="string-equal">
       <SubjectValue DataType="string">QueryModule</AttributeValue>
       <SubjectAttributeDesignator AttributeId="subject-id" DataType="string"/>
     </ResourceMatch>
    </Subjects>

    <Resources>  <!-- RESSOURCE OWNER -->
      <Resource>
        <ResourceMatch MatchId="string-equal">
         <AttributeValue DataType="string">producer1</AttributeValue>
         <ResourceAttributeDesignator AttributeId="owner-id" DataType="string"/>
        </ResourceMatch>
      </Resource>
     </Resources>

     <Actions/>
  </Target>

   ... <!-- POLICYSET CONTENT  -->

</PolicySet>
```

XACML Policy for DS AC

The Policy entity defines a user group. It contains :

• A list of methods in the target. The access rights for these methods are provided by filter rules.
• A rule called 'UserGroup' containing the user/profile id that will be able to use the methods given in the target.
• Four other rules used to allow filters for this group (BizStep,EPC,EventClass and EventTime) **just for Query and Capture module xml files**.

```
<Policy PolicyId="group_name" RuleCombiningAlgId="group-rule">
     <Target>
         <Subjects/>
         <Resources/>
         <Actions>
             <!-- Action list concerned by this policy (ex: userUpdate) -->
         </Actions>
     </Target>
     <Rule RuleId="UserGroup" Effect="Permit">...</Rule> <!-- user list -->
     <Rule RuleId="BizStep" Effect="Permit">...</Rule>  <!-- BizStep filter list -->
     <Rule RuleId="EPC" Effect="Permit">...</Rule> <!-- EPC filter list -->
     <Rule RuleId="EventClass" Effect="Permit">...</Rule> <!-- Event class list -->
     <Rule RuleId="EventTime" Effect="Permit">...</Rule> <!-- Event time list -->
</Policy>
```

XACML POLICY METHOD FILTER

```
<Policy PolicyId="test_env_gp" RuleCombiningAlgId="group-rule">
    <Target>
        ...
        <Actions>

           <Action>
            <ActionMatch MatchId="string-equal">
               <AttributeValue DataType="string">eventCreate</AttributeValue>
               <ActionAttributeDesignator AttributeId="action-id" DataType="string"/>
            </ActionMatch>
           </Action>

           <Action>
            <ActionMatch MatchId="string-equal">
               <AttributeValue DataType="string">eventLookup</AttributeValue>
               <ActionAttributeDesignator AttributeId="action-id" DataType="string"/>
            </ActionMatch>
           </Action>

        </Actions>
    </Target>

    <!-- rules -->

</Policy>
```

XACML QUERY POLICY

The UserGroup rule associating **user2** and **user3** in a same group :

```
<Rule RuleId="UserGroup" Effect="Permit">
 <Target/>
 <Condition FunctionId="global-permit-one-deny">

  <Apply FunctionId="string-equal">
   <Apply FunctionId="string-one-and-only">
    <SubjectAttributeDesignator AttributeId="user-id" DataType="string" />
   </Apply>
   <AttributeValue DataType="string">user2</AttributeValue>
  </Apply>

  <Apply FunctionId="string-equal">
   <Apply FunctionId="string-one-and-only">
    <SubjectAttributeDesignator AttributeId="user-id" DataType="string"/>
   </Apply>
   <AttributeValue DataType="string">user3</AttributeValue>
  </Apply>

 </Condition>
</Rule>
```

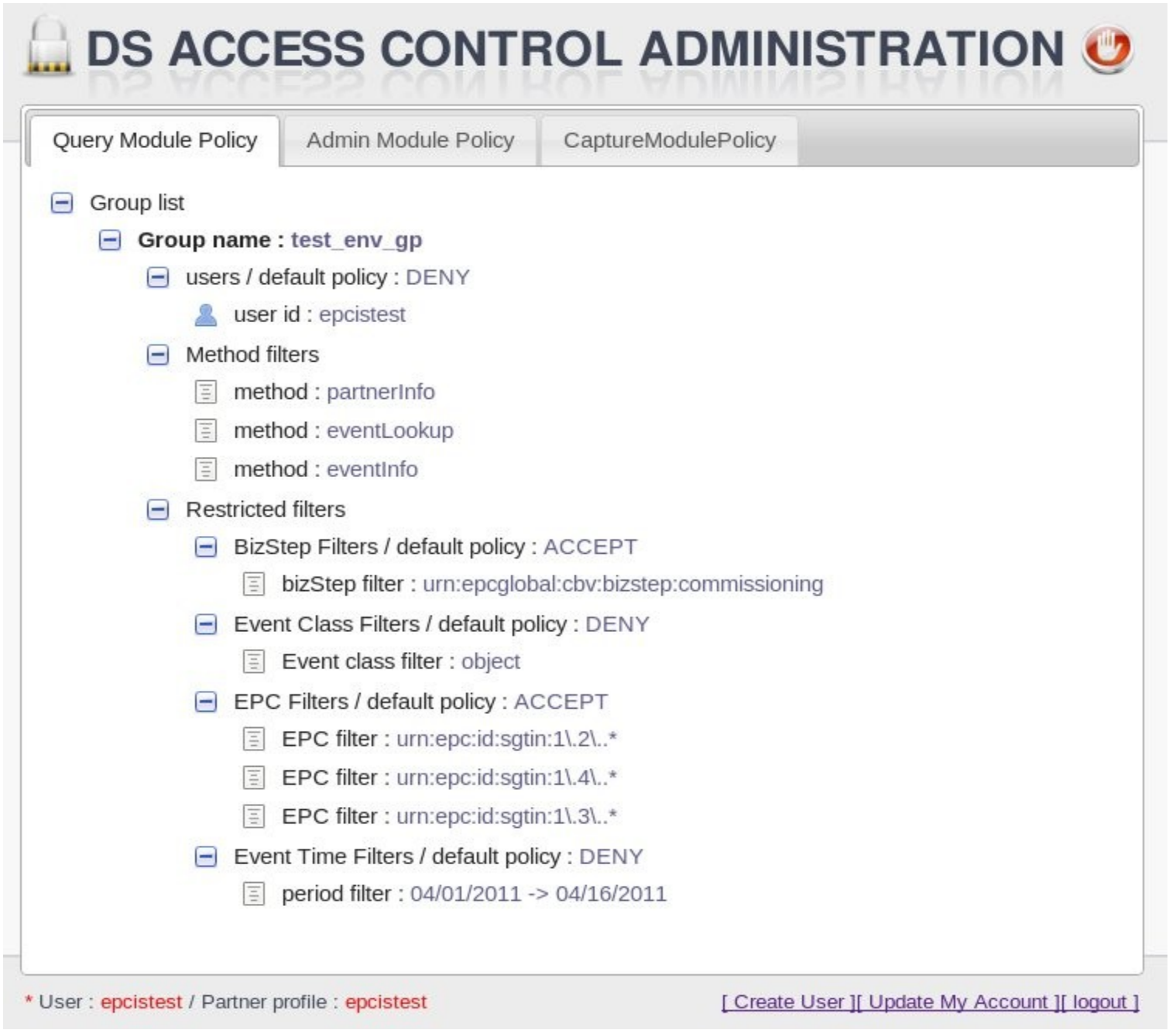
XACML QUERY POLICY

The **BizStep restriction** and **EPC restriction** rules :

```
<Rule RuleId="BizStep" Effect="Permit">
 <Target/>
 <Condition FunctionId="global-permit-one-deny">
  <Apply FunctionId="string-equal">
   <Apply FunctionId="string-one-and-only">
    <ResourceAttributeDesignator AttributeId="bizStep-id" DataType="string"/>
   </Apply>
   <AttributeValue DataType="string">urn:epcglobal:cbv:bizstep:com</AttributeValue>
  </Apply>
 </Condition>
</Rule>

<Rule RuleId="Epc" Effect="Permit">
 <Target/>
 <Condition FunctionId="global-deny-one-permit">
  <Apply FunctionId="unicaen:revert-regexp-string-match">
   <Apply FunctionId="string-one-and-only">
    <ResourceAttributeDesignator AttributeId="epc-id" DataType="string" />
   </Apply>
   <AttributeValue DataType="string">urn:epc:id:sgtin:1\.2\..*</AttributeValue>
  </Apply>
 </Condition>
</Rule>
```

Method defined for access control

Admin methods		Capture methods	Query methods
userInfo	updateGroupName	eventCreate	partnerInfo
partnerInfo	switchUserPermissionPolicy	voidEvent	eventLookup
createPartnerGroup	savePolicyPartner	multipleEventCreate	eventInfo
addPartnerToGroup	updateAdminGroupName		
addBizStepRestriction	createAdminPartnerGroup		
addEPCRestriction	deleteAdminPartnerGroup		
addEPCClassRestriction	addAdminPartnerToGroup		
addTimeRestriction	removeAdminPartnerFromGroup		
addUserPermission	switchAdminUserPermissionPolicy		
switchBizStepPolicy	removeAdminUserPermission		
switchEPCPolicy	addAdminUserPermission		
switchEPCClassPolicy	saveAdminPolicyPartner		
switchTimePolicy	partnerUpdate		
removeBizStepRestriction	userCreate		
removeEPCRestriction	userUpdate		
removeEPCClassRestriction	userLookup		
removeTimeRestriction	userDelete		
removeUserPermission	partnerDelete		
deletePartnerGroup			
removePartnerFromGroup			


DS ACCESS CONTROL ADMINISTRATION
Query Module Policy
Admin Module Policy
CaptureModulePolicy
Group list
Group name : test_env_gp
users / default policy : DENY
user id : epcistest
Method filters
method : partnerInfo
method : eventLookup
method : eventInfo
Restricted filters
BizStep Filters / default policy : ACCEPT
bizStep filter : urn:epcglobal:cbv:bizstep:commissioning
Event Class Filters / default policy : DENY
Event class filter : object
EPC Filters / default policy : ACCEPT
EPC filter : urn:epc:id:sgtin:1\.2\..*
EPC filter : urn:epc:id:sgtin:1\.4\..*
EPC filter : urn:epc:id:sgtin:1\.3\..*
Event Time Filters / default policy : DENY
period filter : 04/01/2011 -> 04/16/2011
* User : epcistest / Partner profile : epcistest
[Create User][Update My Account][logout]

XACML Requests

XACML Requests

The PEP component creates XACML requests. An XACML request comprises three parts :

- **Subject** : the user who wants to process the given action and the corresponding module.
- **Ressource**:
  * The information about the policies' owner that will be processed for this request.
  * epc, bizStep, eventTime and eventClass for Query and Capture modules
- **Action**: the method name that needs an explicit access to be executed.

```
<Request>
    <Subject>
        ... <!-- who makes the query ? -->
    </Subject>
    <Resource>
        ... <!-- ressource definition -->
    </Resource>
    <Action>
        ... <!-- method name -->
    </Action>
</Request>
```

XACML Request - subject

Example of Subject sent by **user8** in the **capture module:**

```
<Request>

   <Subject SubjectCategory="access-subject">

        <Attribute AttributeId="module-id" DataType="string">
            <AttributeValue>Capture</AttributeValue>
        </Attribute>

        <Attribute AttributeId="user-id" DataType="string">
            <AttributeValue>user8</AttributeValue>
        </Attribute>

   </Subject>

   <Resource>
    ... <!-- ressource definition -->
   </Resource>

   <action>
    ... <!-- method name -->
   </action>

</Request>
```

XACML Request Resources for DS AC

Example of the resource definition for an “eventCreate” processed by a user of the producer1 profile:

```
...
<Resource>

    <Attribute AttributeId="resource-id" DataType="string">
        <AttributeValue>DSevent</AttributeValue>
    </Attribute>
    <Attribute AttributeId="owner-id" DataType="string">
        <AttributeValue>producer1</AttributeValue>
    </Attribute>

    <Attribute AttributeId="epc-id" DataType="string">
        <AttributeValue>urn:epc:id:sgtin:1.2.3</AttributeValue>
    </Attribute>
    <Attribute AttributeId="eventTime-id" DataType="dateTime">
        <AttributeValue>1970-01-01T01:00:08+01:00</AttributeValue>
    </Attribute>
    <Attribute AttributeId="bizStep-id" DataType="string">
        <AttributeValue>bizStep1</AttributeValue>
    </Attribute>
    <Attribute AttributeId="eventClass-id" DataType="string">
        <AttributeValue>ObjectEvent</AttributeValue>
    </Attribute>

</Resource>
...
```

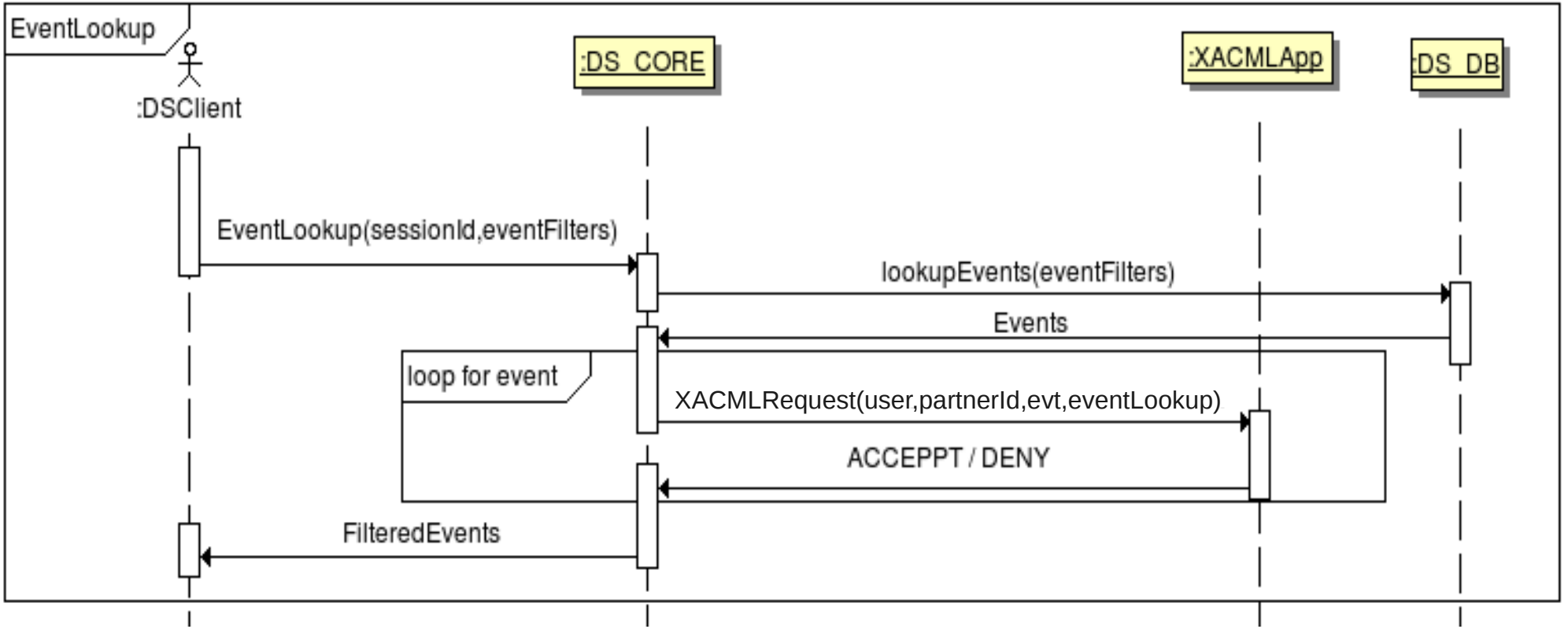
XACML Request - action

The corresponding **action** definition:

```
<Request>

   <Subject SubjectCategory="access-subject">
    ... <!-- who makes the query ? -->
   </Subject>

   <Resource>
    ... <!-- ressource definition -->
   </Resource>

   <Action>
        <Attribute AttributeId="action-id" DataType="string">
            <AttributeValue>eventCreate</AttributeValue>
        </Attribute>
   </Action>

</Request>
```

USE CASES

DS EventLookup processing



DS UserCreate processing