OASIS eXtensible Access Control Markup Language (XACML)

Technical Committee

Issues List

Version 10

August 19, 2002

Ken Yagen, Editor

Purpose

This document catalogs issues for the eXtensible Access Control Markup Language (XACML) developed the Oasis eXtensible Access Control Markup Language Technical Committee.

Introduction

The issues list presented here documents issues brought up in response to draft documents as well as other issues mentioned on the xacml mailing list, in conference calls, and in other venues. The structure of this document was taken from the Security Assertion Markup Language (SAML) Issues List document maintained at the Security Services Technical Committee document repository. Each issue is formatted as follows:

ISSUE:[Document/Section Abbreviation-Issue Number: Short name] Issue long description. Possible resolutions, with optional editor resolution Decision

The issues are informally grouped according to general areas of concern. For this document, the "Issue Number" is given as "#-##", where the first number is the number of the issue group.

To make reading this document easier, the following convention has been adopted for shading sections in various colors.

Gray is used to indicate issues that were previously closed.

Blue is used to indicate issues that have been flagged as ready to close in the most recent revision. These require review and voting by the committee and they can be closed.

Yellow is used to indicated issues which have recently been created or modified or are actively being debated.

Other open issues are not marked, i.e. left white.

Issues with lengthy write-ups, that have been closed “for some time” will be removed from this document, in order to reduce its overall size. The headings, a short description and resolution will be retained. All vote summaries from closed issues will also been removed.

Use Case Issues

Group 1: Group Name

Design Issues

Group 1: Group Name

Policy Model Issues

Group 1: Rules

ISSUE:[PM-1-01: Negative Authorizations]

Authorizations can be either positive (permit) or negative (deny). Should we allow both?

See also PM-1-01-A which was split off from this issue.

Potential Resolutions:

[Text Removed in Version 08]

Proposed Resolution:

XACML allows policy writers to specify positive (permit) or negative (deny) authorization. The negative authorization is specified using the effect element with "deny" in the rule with corresponding rule set combiner such as "meta-policy-1" meaning the global-deny semantics. Using the rule combiner (XACML extension point), the semantics of the negative authorization varies depending on the user-defined rule combiner. PM-1-01-A discusses about the global-deny semantics.

Champion: Michiharu

Status: Closed

ISSUE:[PM-1-01-A: Implementing global deny and Meta-Policies]

Implementing global "deny" semantics using schema 0.8 and meta-policies

[Text Removed in Version 08]

Proposed Resolution:

the syntax for <rule> allows for the <rule> to return an <effect> of "permit" or "deny". It is up to the combiner in the <policyStatement> that uses a <rule> to determine the effect of a <rule> that returns "deny". Likewise, it is up to the combiner in the <policyCombinationStatement> that uses a <policyStatement> to determine the effect of a <policyStatement> that returns "deny".

The following example combiners can be used to implement "global deny" semantics for a <rule>. Since an "indeterminate" rule might have evaluated to "deny" if sufficient information had been supplied, these examples treat "indeterminate" results like "deny".

GLOBAL DENY RULE COMBINER:

```
for <rule> in <ruleSet> {
  boolean atLeastOnePermit = false;
  effect = eval(<rule>);
  if (effect == "deny" || effect == "indeterminate") {
     return "deny";
  } else if (effect == "permit") {
     atLeastOnePermit = true;
  }
}
if (atLeastOnePermit) {
  return "permit";
} else {
  return "not applicable";
}
```

GLOBAL DENY POLICY COMBINER:

```
for <policy> in <policySet> {
  boolean atLeastOnePermit = false;
  effect = eval(<policy>);
  if (effect == "deny" || effect == "indeterminate") {
    return "deny";
  } else if (effect == "permit") {
    atLeastOnePermit = true;
  }
}
if (atLeastOnePermit) {
  return "permit";
} else {
  return "not applicable";
}
```

Policy and policy combination writers that do not wish to support "global deny" semantics can specify different combiners.

Policy combination writers should publish the combiner they use to policy writers so that consistent semantics are maintained: if a policy combination writer is implementing "global deny", then the policy writers should be aware that returning an effect of "deny" will by itself result in denial of access.

Champion: Anne

Status: Closed

ISSUE:[PM-1-02: Post-Conditions]

[Text Removed in Version 08]

Proposed Resolution:

[From Michiharu and Anne]

[We use the term "obligation" to mean what we have previously been calling "post condition". The issue of the term is addressed in PM-1-03.]

Obligations are annotations that MAY be specified in a policyStatement and/or policyCombinationStatement that should be returned in conjunction with an authorization decision meaning that the obligations(s) SHOULD be executed by the PEP. The obligation is specified using URI reference with optional arguments. The actual meaning of each obligation depends on the application. It also depends on the configuration of the PEP and/or PDP. If the PEP does not recognize an obligation, the PEP should deny access.

The set of obligations returned by each level of evaluation includes only those obligations returned by rules, policyStatements, or policyCombinationStatements that were actually evaluated by the combiner algorithm, and associated with the effect element being returned by the given level of evaluation. For example, a policy set may include some policies that return Permit and other policies that return Deny for a given request evaluation. If the policy combiner returns a result of Permit, then only those obligations associated with the policies that were evaluated, and that returned Permit are returned to the next higher level of evaluation. If the PDP's evaluation is viewed as a tree of policyCombinationStatements, policyStatements, and rules, each of which returns "Permit" or "Deny", then the set of obligations returned by the PDP will include only the obligations associated with evaluated paths where the effect at each level of evaluation is the same as the effect being returned by the PDP.

Champion: Simon

Status: Closed

ISSUE:[PM-1-03: Post-Conditions as a term]

[Text Removed in Version 08]

Proposed Resolution:

At the March, 2002 Face-to-Face meeting, we agreed to use the term "obligation" to express an annotation associated with an access decision that is returned to a PEP. This term replaces our former use of "post-condition".

Champion: Bill

Status: Closed

ISSUE:[PM-1-04:References to attributes in XACML predicates]

What information needs to be provided in order to refer to an attribute in an XACML policy predicate?

Potential Resolutions:

Proposed Resolution:

References to attributes associated with the access request in XACML predicates consist of a URI to a document instance that contains the value of the attribute to be evaluated, a URI for the schema for the document, a schema-dependent path for locating a particular attribute instance in the document according to the schema, and an optional name for the Attribute Authority trusted to assign values for this attribute. The AA is located using the PKI with which the PDP is configured.

Vote:

2/21: There was considerable discussion about whether this was ready to close. The feeling was that we needed to see a specific proposal either free standing or in the working spec before we could vote to close. The issue was raised as to whether we should use XPath expressions here. It was not closed

Resolution:

Two possible ways provided to refer to attributes: attribute designator and attribute selector (XPATH). PAP has a choice of which to use. Use XPATH if need to refer to attributes within a resource.

Champion: Anne

Status: Closed

ISSUE:[PM-1-05: how NOT-APPLICABLE impacts a combinator expression]

[Text Removed in Version 08]

Proposed Resolution:

A <rule> will return NOT-APPLICABLE under the following conditions:

<rule> Truth Table:

Target	Condition	Effect
match	match	[Effect]
match	no-match	Inapplicable
match	Indet.	Indet.
no-match	match	Inapplicable
no-match	no-match	Inapplicable
no-match	Indet.	Inapplicable

It is up to the combiner in the <policyStatement> that uses a <rule> to determine the effect of a <rule> that returns "Inapplicable". Likewise, it is up to the combiner in the <policyCombinationStatement> that uses a <policyStatement> to determine the effect of a <policyStatement> that returns "Inapplicable".

The example "GLOBAL DENY" combiners proposed in PM-1-01A can be used to implement "remove inapplicable elements from the computation" semantics.

The following example combiners can be used to implement "inapplicable same as deny" semantics. Such semantics might be desired where all rules are intended to be applicable, so a result of inapplicable indicates some breakdown in the consistency of the system.

INAPPLICABLE GLOBAL DENY RULE COMBINER:

```
if (<ruleSet> == null) {
  return "deny";
}
for <rule> in <ruleSet> {
  effect = eval(<rule>);
  if (effect == "deny" ||
      effect == "indeterminate" ||
      effect == "inapplicable") {
     return "deny";
}
return "permit";
```

INAPPLICABLE GLOBAL DENY POLICY COMBINER:

```
if (<policySet> == null) {
  return "deny"
}
for <policy> in <policySet> {
  effect = eval(<policy>);
  if (effect == "deny" ||
      effect == "indeterminate" ||
      effect == "inapplicable") {
    return "deny";
}
return "permit";
```

Champion: Anne

Status: Closed

ISSUE:[PM-1-06: result of <N-OF n=0> combinator expression]

We all agreed that <N-OF n=[something greater than 0]> was an error if there were not at least n predicates to be evaluated. We also agreed that the semantics of <N-OF> were "at least n of". We did not agree on what should be the result of <N-OF n=0>.

Resolution:

<N-OF n=0> results in TRUE, regardless of the results of the predicates in the combinator expression.

Champion: Anne

Status: Closed

ISSUE:[PM-1-07: How can the set of combinators be extended?]

[Text Removed in Version 08]

Proposed Resolution:

The combiner algorithm to be used by a given <policyStatement> or <policyCombinationStatement> is specified using a URI. XACML will specify a small set of mandatory-to-implement combiner algorithms. The algorithm associated with the URI MAY be descriptive text. Users are free to define other algorithms, although not all XACML-compliant PDPs will be able to apply them.

Champion: Anne

Status: Closed

ISSUE:[PM-1-08: syntax for <applicablePolicyReference>]

If a predicate in XACML references an <xacml:applicablePolicy>, what should the syntax for this reference be?

Potential Resolution:

The syntax should include a URI for <xacml:applicablePolicy> and a URI for the Policy Authority trusted to issue and sign this <xacml:applicablePolicy>. The name attribute in the referenced <xacml:applicablePolicy> must match the URI in the <applicablePolicyReference>. A chain of <applicablePolicyReference> that contains a cycle has a result of ERROR.

Keep it simple because we have no experience with how it will be used. URI may not be globally unique, just unique within policy authority

Resolution:

Syntax is just the URI for the policy, not the Policy Authority. Policy ID and Policy Set ID elements in schema are both defined as URI. Specifying the authority is left for a potential enhancement in future versions of XACML.

Champion: Anne

Status: Closed

Group 2: Applicable Policy

ISSUE:[PM-2-01: Referencing Multiple Policies]

[Text Removed in Version 08]

Proposed Resolution:

Multiple policies may be referenced and combined using a <policyCombinationStatement>. This has the following syntax:

```
<policyCombinationStatement>
  <target/>
  <policySet Combiner="myURI">
    <policyDesignator>
      <policyRef> or <policyStatement> or
        <policyCombinationRef> or <policyCombinationStatement> or
        <saml:assertion>
      <policyMetadata>
    </policyDesignator>
    <policyDesignator>...</policyDesignator>
    <obligations />   OPTIONAL
  </policySet>
</policyCombinationStatement>
```

The <policyDesignator> element specifies a policy to include, using one of various ways of referring to a policy. There can be multiple <policyDesignator> elements in a <policyCombinationStatement>. The "combiner" specifies how the various policies are to be combined to produce a result.

Champion: Anne

Status: Closed

ISSUE:[PM-2-02: Target Specification]

According to the current schema each applicable policy can have multiple targets, each of which is an action and a URI identifying a set of resources (possibly with a transfer function to support wildcards). One may want to specify the target with reference to resource attributes (e.g., this policy applies to all files older that two years). How can I specify this?

[Tim] A different transform algorithm is all that is required. In the example, the "classification" is "older than two years", and the transform algorithm specifies how to deduce the age of a file.

Simon will present counter deductions to Anne 's proposal at the F2F

Potential Resolutions:

Ernesto suggests that this issue only mention retrieval of distributed policies and should be updated to reflect the recent discussion and Anne's proposal (See PM-1-01A) about policy combination. Anne volunteers to extend its wording in order to include policy combination as well.

Anne: [This note has to do with the syntax for expressing "applicability" of a single policy, and not with the logical rules for combining an inapplicable policy with other policies!!]

We currently allow a <target> element predicate in <applicablePolicy> element. The purpose of this element is to allow a PDP (or its agent, a PRP) to eliminate policies efficiently if they do not apply to the current authorizationDecisionQuery. Such an element can be used to index policies by Subject or Resource/Action (where some policies will need to be indexed under both Subject and Resource/Action, and some policies will apply to all Subjects and/or Resource/Actions). The idea is that the <target> element predicate is simple to compute, and allows the PDP (or PRP) to narrow down the field of potentially applicable policies efficiently. The PDP (or PRP) can then perform more complex evaluations on the smaller remaining set of policies.

Since the <target> element needs to be a simple predicate that is efficient to compute, it is not sufficiently expressive to rule out all cases where the <policy> may not apply. For example, if the policy applies only to employees who are over 55 years of age, then there is no syntax currently for expressing this in the <target> element.

POTENTIAL RESOLUTION:

We need two levels of applicability predicate: one used for fast narrowing down of the set of potentially applicable policies (and used for indexing), and the second for fully expressing the conditions under which this policy is applicable.

The first level applicability predicate is our current syntax: a regular expression match on a Resource/Action and Subject. It is very simple to compute, and MUST return TRUE for every authorizationDecisionQuery to which the corresponding policy applies. It MAY return TRUE for an authorizationDecisionQuery to which it does not apply. This predicate might be called "indexApplicability" or "basicApplicability" or something similar.

The second level applicability predicate is an optional new element in the <applicablePolicy>. It may use any comparison of attributes and values that could be used in the policy itself. This predicate might be called "fullApplicability" or something similar. This second level predicate is optional because for many policies, only the first level predicate may be required to fully capture the exact set of conditions under which the policy applies.

A policy evaluation returns "NOT-APPLICABLE" if either the first level applicability predicate OR the second level applicability predicate evaluates to FALSE. The second level predicate need be computed ONLY IF the first level predicate evaluates to TRUE.

The <policy> element may assume that the first and second level applicability predicates have been evaluated to TRUE. This may save some duplicate predicates.

Resolution:

Resolved in the schema in definition of Resource Attribute Designator and rules for evaluating target and condition.

Champion: Simon G.

Status: Closed

ISSUE:[PM-2-03: Meaningful Actions]

[Text Removed in Version 08]

Proposed Resolution:

The XACML syntax shall not address the question of which actions are valid for a particular resource classification.

Champion: Simon G.

Status: Closed

ISSUE:[PM-2-04: Indexing Policy]

Also related to target are indexing issues and how to retrieve, given a request, the applicable policy for it [Tim].

Potential Resolutions:

[Tim] Section 6.4 of version 0.8 of the language proposal is reserved for tackling this question in the LDAP case. Do we need to tackle other cases?

[Tim] The XACML specification shall provide normative, but non-mandatory to implement, text that profiles LDAP for distribution of XACML instances. [PM-2-04]

[Tim] The XACML specification shall provide normative, but non-mandatory to implement, text that profiles "the Web" for distribution of XACML instances. [PM-2-04]

Resolution:

LDAP profile deferred to post 1.0 so no longer an issue. Have target inside policy, but don't specify how to find the particular policy given the request. Left up to local implementations.

Champion: Tim

Status: Deferred

ISSUE:[PM-2-05: Ensuring Completeness]

[Text Removed in Version 08]

Proposed Resolution [Polar]:

This resolution is against the Version 12 document:

I would suggest that we add a Normative section for Operational Semantics. I suggest that we put it between Section 8 and Section 9 (of course altering the numbering of 9 to 10, etc). We may add more normative parts for other operational parts of the model. However, I think the only one we have to really worry about is the PDP, which is the XACML policy language evaluator.

However, given the enormous flexibility of our model, I don't think we can actually state specify by XACML language alone, what happens behind the PDP, a.k.a retrieving policies, attributes, (lazy evaluation) etc. It appears that our PDP can be an interconnected collection of PRPs, PIPs, and even other PDPs recursively. I think it best just to state the compliance rules for a PDP for our viable language elements.

The basic crux of the argument is that the when faced with evaluating a XACML policy or policy set it will do so in accordance to the semantics that we lay out in this document. (I've kept the terminology somewhat non-saml specific (i.e. "authorization decision request"), and apply that conformance to the SAML profile section.

Here it goes:

8.0 Operational Model (Normative)

8.1 Policy Decision Point (PDP)

Given a valid XACML "policy statement" or a "policy set statement", a compliant XACML PDP MUST evaluate that statement in accordance to the semantics specified in Sections 5, 6, and 7 when applied to an "authorization decision request". The PDP MUST return a "authorization decision", with one value of "permit", "deny", or "indeterminate". The PDP MAY return an "authorization decision" of "indeterminate" with an error code of "insufficient information", signifying that more information needed. In this case, the "authorization decision" MAY list any the names of any attributes of the subject and the resource that are needed by the PDP to refine its "authorization decision".

Decision Convergence

A client of a PDP MAY resubmit a refined authorization decision request in response to an "authorization decision" of "indeterminate" with an error code of "insufficient information" by adding attribute values for the attribute names that are listed in the response.

When the PDP returns an "authorization decision" of "indeterminate" with and error code of "insufficient information", a PDP MUST NOT list the names of any attribute of the subject or the resource of the "authorization decision request" of which values were already supplied in the "authorization decision request". Note, this requirement forces the PDP to eventually return an "authorization decision" of "permit", "deny", or "indeterminate" with some other reason, in response to successively refined "authorization decision requests".

9. Profiles (Normative, but not mandatory to implement)

9.2 SAML Profile

A compliant SAML based PDP MUST reply to an SAML Authorization Decision Request with a SAML Authorization Decision in accordance with operational semantics of the PDP stated in Section 8.1.

Champion: Pierangela

Status: Closed

ISSUE:[PM-2-06:Encapsulation of XACML policy (was Policy Security)]

[Text Removed in Version 08]

Proposed Resolution:

The XACML syntax will not contain its own security features. An XACML rule has no XACML-specified encapsulation. An XACML policyStatement or policyCombinationStatement MAY be encapsulated in a SAML assertion.

Champion: Tim

Status: Closed

ISSUE:[PM-2-07: valueRef type]

Resolution 5: XACML valueRef elements shall be of type "saml:AttributeValueType".

Resolution:

Attribute has attribute value but are not importing SAML schema. XACML has defined its own schema.

Champion: Tim

Status: Closed

ISSUE:[PM-2-08: Outcome of policies and their combination]

[Probably related to several other issues]

[Text Removed in Version 08]

Proposed Resolution:

[This resolution is related to the proposed resolutions to PM-1-01-A, PM-1-05, PM-1-07, PM-2-01, PM-3-03, PM-3-03A]

The combiner algorithm to be used by a given <policyStatement> or <policyCombinationStatement> is specified using a URI. The algorithm associated with the URI MAY be descriptive text.

XACML will specify a small set of mandatory-to-implement combiner algorithms. Users are free to define other algorithms, although not all XACML-compliant PDPs will be able to apply them.

The combiner algorithm specifies how the associated <ruleSet> or <policySet> is combined, and what the outcome will be.

Champion: Ernesto/Polar

Status: Closed

Group 3: Policy Composition

Assuming an Applicable Policy can refer to several Policy elements, we need to answer the following questions:

ISSUE:[PM-3-01: Combining Policy Elements]

[Text Removed in Version 08]

Proposed Resolution:

PolicyCombinationStatement allows policy writers to specify arbitrary algorithm to combine one or more PolicyStatement and/or one or more PolicyCombinationStatement. A policySetCombiner attribute in the PolicyCombinationStatement is used to identify the combination algorithm. PolicyMetaData MAY be used to combine policies.

Champion: Michiharu

Status: Closed

ISSUE:[PM-3-02: Specifying Policy Outcome]

How the policy outcome should be specified. Possibilities are 2-valued (access decision is ``grant"/"deny") or 3-valued (policy outcome is ``grant"/"deny"/nothing). Note the ``nothing" means that no rule applies, to be solved according to default. (Related work on composition…?)

How does the PEP interpret the answer I don’t know?

Potential Resolutions:

[Tim] Ultimately, the PEP has to know whether or not to grant access. So, someone has to decide, and (by definition) it is the PDP. So, the "don't care" response isn't helpful. However, saml should have an error code to indicate that the PDP is not the appropriate PDP to render a decision on a particular request.

[Tim] The XACML specification shall specify when a PDP should return saml:decision attributes with the values "permit" and "deny". If the PDP is unable to render a decision, then a saml status code shall be returned. No decision value shall be supplied in this case. [PM-3-02]

Resolution:

Four value returns as well as errors allowed. (Permit, Deny, Indeterminate, Not Applicable) If give back an indeterminate, do not specify what the PEP should do. If give back not applicable, then ask another PDP. Do have status codes to provide PEP with means to provide additional information. What PEP does with the decision is out of scope.

Champion: Simon

Status: Closed

ISSUE:[PM-3-03: multiple Base Policies]

Can a PDP have more than one Base Policy?

Potential Resolutions:

Alternative 1:

A PDP MAY have multiple Base Policies, but such Base Policies SHOULD have non-overlapping <xacml:target> elements. The XACML specification does not specify the order in which multiple Base Policies are evaluated, or the result if two or more Base Policies have overlapping <xacml:target> elements.

A PDP that has multiple Base Policies MUST publish its algorithm for the order in which Base Policies are evaluated and the result where two or more Base Policies have overlapping <xacml:target> elements.

Alternative 2:

Base Policies have restricted <target> elements that are easily compared for overlap. In this alternative, the case where base policies overlap is an ERROR. Note that the 0.8 syntax favors this alternative and allows Alternative 3.

Alternative 3:

There is only one Base Policy. Either it has no <target>, and applies to all Resources or it has a <target> element that specifies the set of resources which this PDP is prepared to handle and returns NOT-APPLICABLE if a resource does match that target.

Potential Resolution:

A given PDP uses a single <policyCombinationStatement> or <policyStatement> as the root of its evaluation. The <target> element of this base policy specifies the set of resources, subjects, and actions that this PDP is prepared to handle. This <target> element MAY be universal (allSubjects, allResources, allActions). A PDP returns NOT-APPLICABLE if a request does not match the <target> in its base policy.

[NOTE: Separate issue PM-5-13 of whether this can be overridden by input from the PEP].

Resolution

This concept no longer applies. There is no overriding XACML policy that acts as a base policy.

Champion: Anne

Status: Closed

ISSUE:[PM-3-03A: default PDP result]

If no Base Policy applies to a given Access Request (i.e. all Base Policy evaluations return NOT-APPLICABLE), does the PDP return NOT-APPLICABLE (=SAML INDETERMINATE) to the PEP, or is the PDP configured with a default result to return (e.g. TRUE or FALSE)?

Potential Resolution:

If no Base Policy applies to a given Access Request, then the PDP returns NOT-APPLICABLE (=SAML INDETERMINATE) to the PEP.

Potential Resolution:

A PDP must have a single base policy, which may be either a <policyStatement> or a <policyCombinationStatement>. This base policy will always return a result, whether it is "permit", "deny", "NOT-APPLICABLE", or "Indeterminate".

Resolution:

If no policy applies to a given Access Request, then the PDP returns NOT-APPLICABLE to the PEP.

Champion: Anne

Status: Closed

ISSUE:[PM-3-04: Pseudo Code for Combiner Algorithms]

Shall XACML mandatory-to-implement combiner algorithms be described using some sort of formal language or pseudo-code? If so, what syntax shall we use?

Anne, Ernesto, Carlisle, and Tim recommended that some sort of pseudo-code be used. Java was suggested. Ernesto offered to research various standard pseudo-codes and make a recommendation.

Anne’s Proposed Resolution:

Java syntax should be used to describe any mandatory-to-implement combiner algorithms.

Konstantin’s Proposed Resolution:

Object Constraint Language (OCL) v1.4, as specified in [OMG formal/01-09-77], should be used to describe any mandatory-to-implement combiner algorithms.

Result of Vote:

Six voted to approve OCL as the language to express combiner algorithms; Hal and Ken voted to accept the originally-proposed resolution (i.e., Java); Anne voted for Java or, failing that, C/C++ (but would be happy to accept OCL "if that is what the majority wish"). My personal objection to OCL is that the example that Konstantin posted did not seem as clear to me as the pseudocode example (in particular, I found the operator "exists" to be entirely non-intuitive), so I wonder how many readers/implementers of XACML will struggle with this. I am willing to close this issue since the majority has voted in favour of OCL, but I would prefer to continue discussions on this issue until Thursday's TC call. Remember that the only goal is to be able to specify as clearly as possible what we want the combiner to do. On a first glance, OCL doesn't do that for me. I don't think we need to have a real software language for this, although that might be nice. I don't even think we necessarily have to have a standardized pseudocode; anything will do, as long as it is clear. For the small number of combiner algorithms that we will include in XACML 1.0, what we currently have in v0.12 seems fine to me. Can someone explain why OCL is a better choice than the current Section 7.1 if all we want to do is say what we mean by "deny overrides"?

Discussion on 4/18:

The committee discussed the pros and cons of using it or pseudo code to describe combiner algorithms like "deny overrides." Konstantin had recommended it if we were attempting to define a method of ensuring compliance to the spec, because it is a formal language. The consensus was that it was too unfamiliar for many, but more importantly, XACML requires an explanation of the combiner algorithms, not a specification. So, a less formal English explanation and vendor-neutral pseudo code should be sufficient. No formal vote was taken on the issue, but Tim will incorporate this in the next specification revision.

Resolution:

Informal pseudo code is used, but not in any particular language. Pseudo code was all written by Polar, so it is consistent and there is a plain language equivalent as well.

Champion: Ernesto.

Status: closed

Group 4: Syntax

ISSUE:[PM-4-01: Triplet Syntax (was Syntactic Sugar)]

The current schema assumes authorizations are specified as a pre-condition which is an expression made of predicates on SAML attributes (conditions on principal, resource and environment can be interspersed), let's call it Option ``pre-cond" [Carlisle, Tim, Anne, ...]. In the last conference call it was agreed to leave as an open issue whether to group conditions about principal, resource, and environment in three different elements, let's call it Option ``triplet" [Michiharu, Ernesto, Simon, ...]. The argument for Option ``pre-cond" is that there are predicates that involve both principal and resource attributes (e.g., an authorization that states that users can read the files they own). The counter-objection to this is that you can naturally include all predicates on resources in the resource condition element (which can also refer to principal attributes). The argument for the triplet is that it makes authorization specifications conceptually clearer and closer to current approaches.

[Tim] In the 0.8 schema, valueRef has an attribute to indicate the entity to which it applies (principal, resource, etc.). It only has to be consulted if the attribute type identifier is ambiguous.

Potential Resolutions:

[Tim] The XACML syntax will differentiate between model entities (principal, resource, etc.) in its attribute elements, rather than in its rule elements. [PM-4-01]

Resolution:

XACML differentiates in model entities using SubjectAttributeDesignator (and SubjectAttributeDesignatorWhere), ResourceAttributeDesignator, and ActionAttributeDesignator, EnvironmentAttributeDesignator.

Champion: Pierangela

Status: Closed

ISSUE:[PM-4-02: Policy names as URIs]

Policy names are strings. Should we make then URIs?

Potential Resolutions:

Proposed Resolution:

Policy names should be URIs.

Vote:

2/21 Everybody agreed we should close this, because policy names are URIs in the current spec. Then we noticed that actually Policy Identifiers are URIs and Policy Names are strings. Everybody agreed this is the way it should be. Nobody could think of a reason to have an name and an id which were both URIs. **The Committee voted to close this issue with a resolution to leave the name and id as they are (string and URI respectively.)**

Champion: Tim

Status: Closed

ISSUE:[PM-4-03: Required type in policy]

The "rec:patient/patientName" element is a complex type. So, how should we indicate the required type in the policy?

[From PM-4-09] This only allows for simple types. Do we need to support values of complex type?

Datatypes are described as a table listing which types are supported. Datatypes are specified in the AttributeDesignator and Attribute elements. In future may provide a specific schema for communicating datatypes and extension functions between PDP.

Resolution:

Attributes and Functions have a datatype attribute which is a URI that may identify a simple datatype or a structured element.

Champion: Tim

Status: Closed

ISSUE:[PM-4-04:syntax extension]

Issue: should this element be an extension point to which other policy syntaxes can be added?

Potential Resolutions:

Propose Resolution:

Close this issue. It is incompletely specified: which element? Extension issues are in a separate section.

Vote:

The TC voted to close this issue as a matter of housekeeping and take up specific proposals for XACML extension points as separate issues.

Champion: Tim

Status: Closed

ISSUE:[PM-4-05:Policy Name a URI]

Issue: should we make policy name a URI?

Potential Resolutions:

See PM-4-02

Champion: Tim

Status: Closed as Duplicate

ISSUE:[PM-4-06:Comment element]

Issue: Should we include a "comment" element?

Potential Resolutions:

Proposed Resolution:

We should include a "comment" element.

Vote:

It was suggested that Annotation, which is built into XML schema be used instead. It was explained that this is for commenting Schemas, not instances. It was also pointed out that XML has a provision for imbedded comments. **The committee agreed to close this issue. The resolution is that an element called "Description" will be added to the schema and the text will say explicitly that the contents of this element MAY NOT affect policy evaluation in any way.**

Champion: Tim

Status: Closed

ISSUE:[PM-4-07:policy element in a rule]

Issue: Should we allow a policy element in a rule? Then the same schema could express the policy for combining policies. If so, should it be policy or applicable policy?

Potential Resolutions:

See PM-3-01

Champion: Tim

Status: Closed as Duplicate

ISSUE:[PM-4-08:XML elements include xsi:type]

Issue: Should we require XML elements compared in this way to include an xsi:type attribute?

Have optional datatype attribute in context. Required in policy in designator type and attribute value.

Datatype attribute versus xsi attribute. Can you specify both? What if they conflict?

Potential Resolution:

It is not required, but is allowed and considered helpful. By viewing the policy can tell what type you expect and PDP will interpret attribute it finds as value of the specified type. If value is not of expected type, it is an error.

Resolution:

Datatype takes the place of xsi:type, so xsi:type is not required and if datatype exists, it will take precedence over any xsi:type specified.

Champion: Tim

Status: Closed

ISSUE:[PM-4-09:complex types]

Issue: This only allows for simple types. Do we need to support values of complex type?

Proposed Resolution:

See PM-4-03

Champion: Tim

Status: Closed as Duplicate

ISSUE:[PM-4-10:preserve PAP identity]

Issue: Should the identities and/or signatures of the PAPs be preserved in the composed policy?

Proposed Resolution:

a <policyStatement> or <policyCombinationStatement> may be referenced as a saml assertion. In this case, the PAP identity, signature (if present), and other information is available to the associated combiner algorithm. Otherwise, the PAP identity is not preserved, and is not available to the associated combiner algorithm.

Champion: Tim

Status: Closed

Group 5: SAML Related

In the current schema attributes on resources and principals, which can be used in the Target (for resources) and in predicates, are retrieved using URIs pointing to SAML dataflow.

ISSUE:[PM-5-01: Non-SAML Input]

Can this mechanism be extended to point to non-SAML authorities as required in the Java environment [Sehkar]?

At a minimum, extending SAML expressions but broader to other authorities.

Potential Resolutions:

[Tim] The XACML specification shall be closely coupled to saml entities. However, the use of saml namespace identifiers is not intended to imply that all attributes must be retrieved from saml messages and assertions. [PM-5-01]

Resolution:

Input is non-saml. Any authority that can convert its input to the XACML input format can be used.

Champion: Sehkar

Status: Closed

ISSUE:[PM-5-02: Wildcards on Resource Hierarchies]

How do we express wildcards on the resource hierarchies [Simon G.]?

The current schema includes ResourcetoClassificationTransform to this purpose. Is this sufficient?

Potential Resolutions:

[Tim] We should register an OASIS identifier for the use of regular expressions in this context.

[Tim] The XACML syntax shall use registered URIs to identify algorithms for processing resource classification wildcards. [PM-5-02]

Tied to outcome of resolution PM-5-14

Proposed Resolution:

Use "ResourceToClassificationTransform". Register a URI with OASIS for the use of regular expressions in this context. Other transform algorithms may be specified by the use of other URIs to be registered with OASIS.

Resolution:

"xMatch" Functions, specific to type, can recognize wildcards. These exist for String, RFC822 and X.500 names.

Champion: Simon G.

Status: Closed

ISSUE:[PM-5-03: Roles and Group Hierarchies]

[Text Removed in Version 08]

Proposed Resolution:

XACML will not support role and group hierarchies in the policy language. Attribute authorities may support role and group hierarchies.

Champion: Simon G.

Status: Closed

ISSUE:[PM-5-04: SAML Assertions URI]

[Text Removed in Version 08]

Proposed Resolution:

Attributes in SAML assertions are identified by a namespace, which is a URI, and a name, which is a string.

Champion: Simon

Status: Closed

ISSUE:[PM-5-05: XPath]

Use of Xpath for identifying SAML constructs and the use of Xpath operators

Potential Resolutions:

Simon clarifies that the position he will take is that while the use of Xpaths to extract nodeset is just fine, they do not make good values in expression. The solution in the current schema is cleaner.

Anne offers to look into the issue to provide an alternative point of view.

Resolution:

Allow the use of XPath through attribute selector element, but is not required.

Champion: Simon

Status: Closed

ISSUE:[PM-5-06: Multiple actions in single request]

In the SAML issues document, http://www.oasis-open.org/committees/security/docs/draft-sstc-core-discussion-01.doc

... Issue 5.1.15.2 seeks guidance on whether multiple "actions" can be specified in a single decision request.

Potential Resolutions:

[Tim] I feel that XACML should answer this question and send its conclusion in a liaison to SAML. My feeling is that the answer is "No". If "applicable policy" is to be identified with the resource/action pair, then multiple "applicable policies" are involved when multiple actions are involved. Much "cleaner" for there to be a single "applicable policy" for each decision request. And, therefore, a single action per decision request. It is no great hardship to submit multiple decision requests, in the event that you need a decision for each of several actions.

[Hal] Personally I am in favor of limiting this, but I will state the counter argument for the record. If the possible Actions correspond to what can be in the request, then this works fine. The only reason for multiple actions would be some sort of policy provisioning requirement. However, if the Actions are more like privileges or permission bits, and do not match allowable requests one for one, then some requests may require the AND or OR of several actions. I believe this is the motive behind suggesting multiple actions.

I don't see any rush on this as we are not close to proposing changes to the decision protocol yet.

Resolution:

Multiple actions in a single request are not allowed.

Champion: Tim

Status: Closed

ISSUE:[PM-5-07: Delegation]

[Polar] Has anybody thought about how delegation can be reasoned about in XACML? It appears that SAML only asserts a flat list of attributes with a single principal, or am I off base here? Can I support policies on such operations as:

Paul for Peter says debit Peter's account?

Which mean that Paul (or some other party trusted to do so) has issued Paul the authorization to act on behalf of Peter, in this case to access Peter's account. Or such things, like WebServer quoting JohnDoe says lookup in customer database. Where the WebServer may be trusted to authenticate JohnDoe, but no such proof is necessary other than the WebServer merely claiming to be acting on JohnDoe's behalf?

Potential Resolutions:

[Hal] With regards to SAML, the Access Decision Request was deliberately kept simple with the idea that XACML would give us the tools to do the job properly. I have proposed (see my use cases) that XACML not only be able to express policies, but the method of expressing policy inputs be rolled back into the SAML Access Decision Request (and Assertion).

In my opinion, XACML policies should be able to contain predicates about zero or more of the following subjects:

Requestor Subject

Recipient Subject (can be different from requestor)

Intermediary Subject (can be more than one for a given request)

I propose a single construct for Subjects and their attributes and some kind of modifier indicating the type (refrain from using "role" here) of subject.

[Tim] Delegation could be expressed in attribute assertions. The very issuance of an attribute assertion is a form of delegation. So, XACML should not have to concern itself with the process by which an entity obtained an attribute.

Subject category for intermediator, but don't specifically address it.

Would it be possible to write a policy that requires delegation? Yes, stating subject must have attribute from trusted authority stating subject has delegated right.

It is not disallowed but provide no facilities for supporting it. Can place limitations on delegations for a particular subject.

May want to address it in a future version.

Resolution:

Delegation is not specified in XACML. It is not disallowed, but provided no facilities for supporting it.

Champion: Polar/Hal

Status: Deferred

ISSUE:[PM-5-08: saml;Action is a "string"]

These are some of the potential SAML issues. Most of them were found when attempting to write J2SE policy files in XACML syntax. Further discussion is needed on these issues.

saml:Action is currently specified as a "string". Making Action an abstract type would allow it to be extended. This would allow the content model to be defined by a schema external to the SAML spec.

Thus what constitutes an action could be determined by the J2SE schema.

Potential Resolutions:

[Toshi] In SAML, saml:Action is used only in saml:Actions and saml:Actions have Namespace as an attribute. So it is possible to write action(s) such as:

```
<saml:Actions Namespace="urn:J2SEPermission:java.io.FilePermission">
    <saml:Action>write</saml:Action>
</saml:Actions>
```

or

```
<saml:Actions Namespace="urn:J2SEPermission">
   <saml:Action>java.io.FilePermission:write</saml:Action>
</saml:Actions>
```

But it will be useful if we can write something like:

```
<saml:Action>
    <J2SEPermission class="java.io.FilePermission">write</J2SEPermission>
</saml:Action>
```

Resolution:
Action is a datatype

Champion: Sekhar

Status: Closed

ISSUE:[PM-5-09: saml;AuthorizationQuery requires actions]

If actions are optional for XACML, then why should <saml:Actions> be required in <saml:AuthorizationQuery> ? Both the wording in the SAML assertions draft as well as the SAML schema places such a requirement. saml:Actions should be optional in the AuthorizationQuery to accommodate queries without actions. At least for now, I don't anticipate this as an issue for J2SE.

Potential Resolutions:

[Toshi] In the latest SAML spec (core-25), AuthorizationDecisionQuery element has Resource attribute and Actions element and both of them are "required". Does this cause many problems?

(Resource attribute is "optional" for AuthorizationDecisionStatement element.)

As for J2SE case, I think there is an issue in terminology.

Resolution:

In context, action element is required for XACML. Because it is allowed, can map saml:Action into XACML:Action.

Champion: Sekhar

Status: Closed

ISSUE:[PM-5-10: single subject in AuthorizationQuery]

[editor note: Is this issue covered somewhere else?]

saml:AuthorizationQuery currently only contains a single Subject. While a saml:Subject can support multiple NameIdentifier or SubjectConfirmation or AssertionSpecifier elements, it is required that they all belong to the same principal. So a single subject cannot be used for unrelated principals. In J2SE, there is a need to base access control on multiple principals which are not related and this therefore points to a need for more than one Subject in the saml:AuthorizationQuery

Potential Resolutions:

The way out of this appears to be extend SubjectQueryAbstractType.

Resolution:

XACML supports but does not require multiple subjects. No inconsistency since we are more general.

Champion: Hal

Status: Closed

ISSUE:[PM-5-11:XACML container in SAML]

Issue: should we use a SAML assertion as a container for an XACML applicable policy?

Proposed Resolution:

a SAML assertion MAY be used as a container for an XACML <policyStatement> or <policyCombinationStatement>. The policy combiner MAY ignore the container elements, or MAY reference them in making its decision.

Champion: Tim

Status: Closed

ISSUE:[PM-5-12:derive attribute from saml:AttributeValueType]

Issue: Should we derive the attribute from saml:AttributeValueType? This seems to make sense, but the resulting attribute will have to become an element, with start and stop tags, making it larger and less readable.

Resolutions:

XACML defines its own AttributeType and it can be derived from saml:AttributeValueType.

Champion: Tim

Status: Closed

ISSUE:[PM-5-13: Base Policy supplied as part of AuthorizationDecisionQuery]

Some PEPs have knowledge of the policy associated with a resource (example: a typical FileSystem knows the ACLs associated with a file or directory). To support this case, can a Base Policy or <referencedPolicy> be supplied as part of the SAML AuthorizationDecisionQuery?

Possible Resolutions:

Default policy:

A Base Policy or <referencedPolicy> for evaluating a particular Access Request may be specified as part of the Access Request. If a PDP has no Base Policy(s), then the result of evaluating an Access Request that does not specify a Base Policy to use is NOT-APPLICABLE (=SAML INDETERMINATE).

Two ways

Resolution:

Two ways of doing this:

XACML way: Put policy that PEP is asking to have used in the environment context as an attribute.

Perform outside the scope of XACML: have the PEP requested policy extracted prior to the PDP getting the evaluation request.

Champion: Anne

Status: Closed

ISSUE:[PM-5-14: Resource Structure]

Simon proposes that the resource be written in a request-independent manner. The point that

Simon makes in that while in SAML the resource is just a string, XACML should suggest a structure.

Hal comments that while it is good to retain a simplified structure, we should not be tied to SAML as a specific way of expressing requests. In other words, we need to be compatible with SAML, but should not be tied to it. Carlisle, replies that we actually have that in the charter. Hal says we should be compliant, but we should ask SAML to define a more sophisticated request.

Simon says that the SAML way of expressing resources as a string is limited. For instance, what is the resource in case of XML documents? How do i go fine grained?

Ernesto comments that we should not have a sophisticated resource encoding if SAML does not support it. This can be a parallel effort to influence the next version of SAML.

Do we need another attribute identifier that means resource name, but not URI. Currently support just URI. What if want just a string?

Change the type of the resource URI to string and rename to ResourceName rather than ResourceURI for both input and response.

Could we used Resource-Id with a datatype? Can be string or URI.

Resolution:

Support resources identified by name which is compatible with SAML or structured resources. Resource-Id is defined with an optional datatype that can be a string or URI for specifying the name of the resource.

Champion: Simon

Status: Closed

ISSUE:[PM-5-15: Attribute reference tied to object]

Simon comments that attribute reference should be tied to the object. It's a question of tight coupling or loose coupling of the policy with the request. (This issue will be discussed in relationship with PM-5-14)

Resolution:

All attribute types refer to object in their names.

Champion: Simon

Status: Closed

ISSUE:[PM-5-16: Arithmetic Operators]

The issue was discussed at the F2F where Sekhar said he would have looked at it. Sekhar reports that he could not complete it. Hal comments that we will need black box functions. for instance matching a subject requestor to something in a record that requires some sort of private functions: no set of simple operators that we can define that will be good enough. Ernesto, while agreeing on this, comments that it would be useful to have at least the simplest arithmetic operators be part of the language.

Tim has proposed MathML as a solution and published a MathML XML Schema for review

Resolution:

Now have specified functions which include arithmetic operators. They are identified by URI and are extensible.

Champion: Ernesto, Simon, Tim

Status: Closed

ISSUE:[PM-5-17: Boolean Expression of rules]

The current proposal in the document that a policy could be a boolean expression of rules. Pierangela points out that semantics of such a boolean expression seems to be not clear and while boolean expressions (or rather AND and OR) seems to be needed for combining policies they seems not to be for combining rules within an elementary policy.

Proposed Resolution:

The <condition> element in a <rule> can be a Boolean expression of predicates. <rule>s are combined in a <policyStatement> using a "combiner" algorithm, which specifies how the results of the <rule>s are combined. Likewise, <policyStatement>s and other <policyCombinationStatment>s are combined in a <policyCombinationStatement> using a "combiner" algorithm, which specifies how the results of the <policyStatement>s and <policyCombinationStatement>s are combined. Some combiner algorithms may be expressed using boolean expressions, but other combiner algorithms will use other logic. A combiner algorithm MAY be expressed using descriptive text rather than a formal language or pseudo-code.

Champion: Pierangela

Status: Closed

ISSUE:[PM-5-18: Request/Response Context]

Needs to support multiple responses, hierarchal resources, queries about hierarchal resources.

Michiharu is to provide text on SAML profile.

See Context Schema for specifics.

Resolution:

Support multiple responses, hierarchal resources and queries about hierarchal resources.

Champion: Michiharu

Status: Closed

ISSUE:[PM-5-19: Authorization Decision]

Does this relate to a new authorization decision request type for SAML? (In order to support 5-18)

SAML should support some attribute that specifies scope of the resource. Michiharu posted a proposal to extend authz decision to include obligation element. SAML already supports multiple assertions in response but does not support identifier.

Will require new authorization decision request type and assertion.

Resolution:

Created a new authorization decision type and will be proposing a form of this to SAML along with a new authorization decision assertion.

Champion: Anne

Status: Deferred

Group 6: Predicate Cononicalization

ISSUE:[PM-6-01: SAML Assertions URI]

Values used in predicates can refer to various standard formats (e.g, X.509 [Anne]) that could make the predicates evaluation difficult. For instance, if a principal's name is expressed in X.500 syntax you cannot compare it against a simple string. How do we make the representations canonical?

Potential Resolutions:

[Tim] Policy environments have to use consistent type definitions for the attributes they use.

Resolution:

Have specific functions with type specific semantics.

Champion: Anne

Status: Closed

Group 7: Extensibility

ISSUE:[PM-7-01: XACML extensions]

XACML Extension Model that defines what portion of the XACML specification is a core and to what extent the XACML specification can be extended. Based on this proposal, XACML policy administrators can represent much broader access control policies by extending the core portion of the XACML specification.

This extension model is designed to support an XACML extensibility property stated in the XACML charter. This proposal is based on the current language proposal document but includes several modifications.

Potential Resolutions:

See http://lists.oasis-open.org/archives/xacml/200112/msg00076.html

Resolution:

XACML is extensible through use of URIs. Six areas of extensions include function identifiers, attribute identifiers, datatype, subject category, rule combining algorithm id and policy combining algorithm id.

Champion: Michiharu

Status: Closed

Group 8: Post Conditions

This group was created out of issues raised in Michiharu's proposal for post conditions. See Also Issues PM-1-02 and PM-1-03 for more on post conditions

ISSUE:[PM-8-01:] (4.1) Internal v.s. external post conditions

Proposed Resolution:

XACML does not support any distinction between internal post condition and external post condition. It depends on the configuration of PEP and/or PDP.

Champion: Michiharu

Status: Closed

ISSUE:[PM-8-02:] (4.2) Mandatory v.s. advisory post conditions

Proposed Resolution:

XACML does not support any distinction between mandatory obligation and advisory obligation. The meaning of the obligation is determined in each application.

Champion: Michiharu

Status: Closed

ISSUE:[PM-8-03:] (4.3) Inapplicable

Proposed Resolution:

The obligation is not returned to PEP when the authorization decision is determined as inapplicable or indeterminate.

Champion: Michiharu

Status: Closed

ISSUE:[PM-8-04:] (4.4) Base policy v.s. policy reference

[Text Removed in Version 08]

Proposed Resolution:

The obligation is specified in both policyStatement and policyCombinationStatement. The scope of the obligation is defined in ISSUE: PM-1-02 as "The set of obligations returned by each level of evaluation includes only those obligations associated with the effect element being returned by the given level of evaluation. For example, a policy set may include some policies that return Permit and other policies that return Deny for a given request evaluation. If the policy combiner returns a result of Permit, then only those obligations associated with the policies that returned Permit are returned to the next higher level of evaluation. If the PDP's evaluation is viewed as a tree of policyCombinationStatements, policyStatements, and rules, each of which returns "Permit" or "Deny", then the set of obligations returned by the PDP will include only the obligations associated paths where the effect at each level of evaluation is the same as the effect being returned by the PDP."

Champion: Michiharu

Status: Closed

ISSUE:[PM-8-05:] (4.5) How to return obligations via SAML

[Text Removed in Version 08]

Proposed Resolution:

Here is an authorization decision syntax that returns obligation(s). SAML AuthorizationDecisionStatement is extended to include xacml:obligations element by type extension. "samle" namespace prefix is used to indicate SAML extension for the decision assertion with obligation. Note that the following example just shows the overview for simplicity.

```
<saml:Assertion>
  <saml:AuthorizationDecisionStatement Resource="aaa" Decision="Permit"
xsi:type="samle:AuthorizationDecisionStatementWithObligations">
  <saml:Subject>
   <saml:NameIdentifier SecurityDomain="aaa" Name="Alice"/>
  </saml:Subject>
  <saml:Actions Namespace="http://www.oasis-open.org/xmlactions">
   <saml:Action>Read</saml:Action>
  </saml:Actions>
  <xacml:obligations>
   <xacml:obligation obligationId="myId">
     ...
   </xacml:obligation>
  </xacml:obligations>
  </saml:AuthorizationDecisionStatement>
</saml:Assertion>
```

The following "saml" schema fragment defines an authorization decision with obligations.

```
<complexType name="AuthorizationDecisionStatementWithObligations">
  <complexContent>
    <extension base="saml:AuthorizationDecisionStatementType">
      <sequence>
        <element ref="xacml:obligations"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Champion: Michiharu

Status: Closed

ISSUE:[PM-8-06:] (4.6) When to execute post condition

While post condition implies that specified operations must be dealt with prior to the requested access, it does not necessarily mean that the specified operations must be executed synchronously. Taking the obligatory operation usage scenario in 1.2 for example, it is impossible to execute "delete-in-90days" post condition prior to the requested access. It would be reasonable if such operation is queued in the application and guaranteed to be executed later.

Proposed Resolution:

When and how PEP executes obligation depends on each application. XACML (as PDP) does not assume any specific semantics. While obligation implies that specified operation must be dealt with prior to the requested access, it does not necessarily mean that the specified operations must be executed synchronously. Taking the obligatory operation usage scenario like "customers can register themselves with their private information provided that such information is deleted in 90 days--- obligation is delete-in-90days", it is impossible to execute "delete-in-90days" obligation prior to the requested access. It would be reasonable if such operation is queued in the application and guaranteed to be executed later.

Champion: Michiharu

Status: Closed

ISSUE:[PM-8-07:] (4.7) Extension point

Proposed Resolution:

XACML SHOULD support extension point in the post condition specification and semantics. It includes the process of how to determine the post condition. One example is that the processor selects the post condition that is attached to the rule of the highest priority.

Extension point of obligation is 1. obligationId in policyStatement or policyCombinationStatement and 2. ruleSet combiner or policySet combiner. This allows policy writers to specify arbitrary identifier of the user-defined obligation and to specify the semantics of how obligation is computed in response to the access request.

Champion: Michiharu

Status: Closed

Schema Issues

Group 1: General

ISSUE:[SI-1-01:Graphical Representation of Schema]

Should the core text include a graphical representation of the schema? Simon to investigate graphical schema representation with xml spy. Anne suggested including graphical representation of the schema in the core text. Everybody is encouraged to get schema tools like xml spy or similar.

Proposed Resolution:

Bill is creating a graphical representation of the schema and it will exist as a separate document. Waiting on Bill

Champion: Bill

Status: Open

ISSUE:[SI-1-02:Identify Attributes for Rule and Policy]

We need to verify that <rule> and <policy> elements have identity attributes.

Should these be unique within instance of policy.

Proposed Resolution:

Policy ID, PolicySet ID and Rule ID have been defined. No requirements exist in regards to uniqueness of these ids.

Champion: Tim

Status: Closed

ISSUE:[SI-1-03:Built-In Predicate Functions]

We need to define normative set of predicate functions for strings, dates, etc.

Proposed Resolution:

These have been defined.

Champion: Simon

Status: Closed

ISSUE:[SI-1-04:Attribute Designation in context of condition]

When attributes are referenced in predicate expression within <condition> element it is not clear what object owns this attribute: subject, resource, environment etc.

Proposed Resolution:

AttributeDesignators (subject, resource, action, environment) provide this.

Champion: Simon

Status: Closed

ISSUE:[SI-1-05:Extension Schemas]

Will XACML extensibility be handled via extension schemas, or will the XACML base functions include a mechanism for locating extensions?

For example, if I want to define a new predicate to compare dates expressed in the Mayan calendar format, do I

a) define an extension schema

```
xmlns:mayan="http://http://research.sun.com/people/anderson/mayan.xsd";
```

that defines

```
<xs:element name="MayanDateMatch"
            type="xacml:CompareType"
            substitutionGroup="xacml:predicate"/>
```

then use

```
<MayanDateMatch>
 <saml:AttributeDesignator>...</saml:AttributeDesignator>
 <saml:AttributeDesignator>...</saml:AttributeDesignator>
</MayanDate>
```

in my policy, or

b) make use of built-in XACML extensible predicate element, and use in my policy:

```
<Operator OperatorName="MayanDateMatch"
   OperatorNamespace="http://research.sun.com/people/anderson/";>
   <saml:AttributeDesignator>....</saml:AttributeDesignator>
   <string>"tzolkin=2 Etznab, haab=11 Pop"</string>
</Operator>
```

where the base XACML specification defines something like:

```
<xs:element name="Operator"
      type="xacml:ExtensiblePredicateType"
         substitutionGroup="xacml:predicate"/>
<xs:complexType name="ExtensiblePredicateType">
   <xs:complexContent>
      <xs:extension base="xacml:PredicateAbstractType">
      <xs:choice minOccurs="1">
                 <xs:element ref="saml:AttributeDesignator"/>
                 <xs:element ref="saml:Attribute"/>
                 <xs:element ref="xacml:attributeFunction"/>
         <xs:string/>
      </xs:choice>
      <xs:attribute name="OperatorName"
                 type="xs:anyURI"
```

```
                    use="required"/>
            <xs:attribute name="OperatorNamespace"
                    type="xs:anyURI"
                    use="required"/>
        </xs:complexContent>
    </xs:complexType>
```

Proposed Resolution:

Make use of built-in XACML extensible predicate element (functions).

Champion: Anne

Status: Closed

Miscellaneous Issues

Group 1: Glossary

ISSUE:[MI-1-01: Consistency]

Pierangela mentioned something discussed in PM group that may not coincide with glossary concerning pre and post conditions.

Proposed Resolution:

Any glossary concerns should be resolved as part of the resolution for the particular issue in the PM group.

Champion: Pierangela

Status: Closed

ISSUE:[MI-1-02: Definition of Policy vs. Rule]

[Text Removed in Version 08]

Proposed Resolution:

A "rule" is the smallest unit from which a "policy" is composed. A "rule" uses predicates that refer to attributes and values.

A "policy" is a combination of rules or other policies. A combination of rules is called a <policyStatement>. A combination of <policyStatement>s or other <policyCombinationStatement>s is called a <policyCombinationStatement>. A policy is the smallest administrative unit in XACML, and is the smallest unit that can be signed. A policy does not refer to attributes and values, but only to combinations of rules or other policies.

Champion: Carlisle

Status: Closed

ISSUE:[MI-1-03: Definition and purpose of Target]

[Text Removed in Version 08]

Proposed Resolution:

a <target> element consists of three predicates over elements in a SAML access decision request: one over Subject, one over Resource, and one over Action. Any of these predicates may be universal in that they may result in "true" for "anySubject", "anyResource", or "anyAction".

The <target> element in a <rule>, <policyStatement>, or <policyCombinationStatement> has two purposes. First, it allows <rule>s, <policyStatement>s, and policyCombinationStatement>s to be indexed based on their applicable subject, resource, and/or action. Second, it allows a PDP to quickly and efficiently reduce the set of <rule>s, <policyStatement>s, and <policyCombinationStatement>s that must be evaluated in response to a given access decision request.

These intended purposes place three restrictions on what can be included in a <target>. First, the predicates in a <target> must be very efficient to evaluate. Second, each target must contain at most one each of <subject>, <resource> and <action> mapping predicate, which in turn may match multiple actual runtime values. Third, each predicate in a <target> must refer only to attributes that will always be present in a SAML access decision request, since a <target> must not return a result of "indeterminate".

In a <rule>, the <target> element is logically part of the <condition> element. Were indexing and efficiency not a concern, the tests in the <target> could be incorporated into the <condition>. The <target> element serves as the "first pass" test for whether the rule applies:

```
if (<target> == true) {
    if (<condition> == true) {
        return <effect>;
    }
}
return <not applicable>;
```

Champion: Anne

Status: Closed

Group 2: Conformance

ISSUE:[MI-2-01: Successfully Using]

XACML definition of OASIS requirement to successfully use the specification

Proposed Resolution:

"Successfully Using the XACML Specification"

XACML is an XML schema for representing authorization and entitlement policies. However, it is important to note that a compliant Policy Decision Point (PDP) may choose an entirely different representation for its internal evaluation and decision-making processes. That is, it is entirely permissible for XACML to be regarded simply as a policy interchange format, with any given implementation translating the XACML policy to its own local/native/proprietary/alternate policy language sometime prior to evaluation.

A set of test cases (each test case consisting of a specific XACML policy instance, along with all relevant inputs to the policy decision and the corresponding PDP output decision) will be devised and included on the XACML Web site.

In order to be "successfully using the XACML specification", an implementation MUST, for each test case, have a "policy evaluation component" that can consume the policy instance and the inputs and produce the specified output.

Furthermore, the implementation MUST have a "policy creation component" that allows it to generate schema-valid XACML policy instances that can be consumed/processed by other PDPs.

Note that, aside from the XACML policy instance itself, all PDP inputs and outputs MUST be SAML-compliant (i.e., conform with the assertions and protocol messages defined in the SS-TC SAML specification), although other syntaxes/formats for the PDP input and output MAY be supported in addition to this.

Champion: Carlisle

Status: Closed

Group 3: Patents, IP

ISSUE:[MI-3-01: XrML]

[Ernesto] As I recollect, OASIS requested us to evaluate whether any XACML specification might fall in the scope of patents held by others. I quote from a Dec 13th addition to announcements regarding Xerox's XrML:

(http://xml.coverpages.org/xrml.html) :

"ContentGuard's strategy appears to be to make money by licensing the technology -- whatever some outside body defines it to be. It can do this because its patents cover the idea of a rights language in general, no matter what the specifics of the language are".

I know XrML has already been mentioned in our discussions from the technical point of view, but the wording of this announcements makes me suspect that we should explore the matter further from the patents' point of view.

Potential Resolutions:

Oasis has a specific IPR policy and ContentGuard needs to make Oasis aware of any IP as it relates to XACML or other technical committees in accordance with that policy.

[Hal] Paragraph (C) of OASIS.IPR.3.2. makes the following points:

If OASIS knows about something they "shall attempt to obtain from the claimant of such rights a written assurance ..."

However, "results of this procedure shall not affect advancement of a specification..."

Except that "The results will, however, be recorded..." and "...may also direct that a summary of the results be included in any OASIS document published containing the specification." It also says elsewhere that they will not go out of their way to find IPR that has not been drawn to their attention.

Resolution:

Numerous attempts to get a statement regarding XACML and the XrML patents owned by ContentGuard has failed. ContentGuard has not responded. As Oasis members they are required to respond if there is any overlap, so we must assume they are claiming no IP on XACML.

XrML is now also under the OASIS umbrella and there is some overlap in committee participation which should help.

Champion: Ernesto

Status: Closed

ISSUE:[MI-3-02: IBM Patents]

Hal to provide a section which includes correct formatting of IP statement from IBM in regards to their patents.

Resolution:

Waiting on Hal

Champion: Hal

Status: Open

Group 4: Other Standards

ISSUE:[MI-4-01: RuleML]

[Text Removed in Version 08]

Proposed Resolution:

The issue is a generic suggestion about XACML to be a possible application of a general setting for rule representation, RuleML.

Anne proposes that at the F2F every suggestion of taking into account related languages should be mandatory accompanied by a presentation

After a brief discussion on RuleML, the issue is voted closed. It should be deleted from the next version of the issues document

Champion: Edwin

Status: Closed

ISSUE:[MI-4-02: RAD]

Should XACML look at RAD?

[Polar] In response to some query about the expressiveness of evaluation of policies from different places, I would like to point the group to the CORBA Resource Access Decision specification (RAD).

http://www.omg.org/cgi-bin/doc?formal/01-04-11.pdf

and we may want to include it the document repository. It has in it an Access Decision model in which not only policies are located, but also, a policy evaluation combinator is located for a

particular resource. Note, there is no language component to this specification.

However, it does present a model by which policy can be distributed and evaluated. A combinator, which has an interface operation of "evaluate_policies" takes the list of located policies for the resource, the attribute list of the subject, and the operation (i.e. Action) on the resource) and evaluates the decision.

That way, depending the semantics of the combinator you choose for the resource, your combinator may choose to ignore, or evaluate only some policies based on the evaluations of other policies.

Potential Resolutions:

Polar will bring that one to the discussion, with special reference to policy combination.

Resolution:

Polar has been a participant in discussion and has brought in various aspects that may be applicable.

Champion: Polar

Status: Closed

ISSUE:[MI-4-03: DSML]

Transformations from XACML to DSML

[Gil] Since the last time we talked I had the chance to play with DSML a little. It seems to me that it is theoretically possible to transform an XACML policy document into a DSML document and import that document into LDAP. The DSML document could contain elements that described the (LDAP) schema necessary to store the authorization policy entries in case the target LDAP

didn't already have this schema. It is also possible to export some LDAP entries into a DSML document and transform that DSML document in XACML.

What I don't know (having nothing more than a cursory understanding of XSL/XSLT) is how difficult such transformations would be and if there are any "gotchas" that would keep this from really working.

Potential Resolutions:

[Gil] What I think the XACML spec should do is:

1.) Describe the LDAP schema necessary to store authorization policies. This should be done in "LDAP fashion" with dn's, classnames, etc.

2.) (if possible) Provide the XSLT necessary to transform XACML to DSML and vice versa.

That way people who don't want to be bothered with DSML can work out their own way to store and retrieve XACML data to and from the defined schema.

Resolution:

Did specify a way to refer to LDAP attributes from XACML but did not define a way to store XACML in DSML.

Champion: Gil

Status: Deferred

ISSUE:[MI-4-04: Java Security Model]

Hal says he is not clear about whether XACML should be able to represent the Java security model. Gil comments that XACML would be limited if it cannot express it. Hal notes that what XACML should be able to represent are the same requirements that Java security model represents, but not necessarily in the same way (i.e., representing the same authorizations).

Potential Resolutions:

Anne has investigated and believes XACML is capable of expressing the Java Security model.

Champion: Sekhar

Status: Closed

Document History

- 7 Jan 2002 First Version Published
- 21 Jan 2002 Major edits and additions. Every open item updated.
- 18 Feb 2002 Edits based on F2F and Anne’s edits
- 27 Feb 2002 Edits based on 2/21 voting and post condition issues
- 8 Mar 2002 Version 5 released but title page had version 4 information
- 27 Mar 2002 Closed issues updated from F2F and Policy Model Calls
- 18 Apr 2002 Reflected official email voting results and added schema issues from Simon/Anne
- 10 Jul 2002 Removed much of text of closed issues; Added new SAML issues
- 08 Aug 2002 Reviewed and closed issues up to Group 3 during TC Call.
- 19 Aug 2002 Reviewed remaining open issues in subcommittee call and defined resolutions closing them.