|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **itu-old** | INTERNATIONAL TELECOMMUNICATION UNION | | | | COM 17 – C xx – E |
| **TELECOMMUNICATION STANDARDIZATION SECTOR**  STUDY PERIOD 2013-2016 | | | **December 2012** | |
| **English only**  **Original: English** | |
| **Question(s):** | | 7,10/17 |  | | |
| **STUDY GROUP 17 – CONTRIBUTION xx** | | | | | |
| **Source:** | | Korea (Rep. of) | | | |
| **Title:** | | Proposal for supplement to X.1144 – Enhancements and new features in XACML 3.0 | | | |

This contribution proposes the supplement to X.1144 (XACML 3.0) based on C114 that summarizes the enhancements and new features of XACML3.0 compared to XACML 2.0.

**Supplement xx to ITU-T X-series Recommendations**

**ITU-T X.1144 – Supplement on enhancements and new features in XACML 3.0**

This supplement summarizes the enhancements and new features of XACML3.0 compared to XACML 2.0.

* Advice element:This new feature is similar to obligations with the exception that PEPs do not have to comply with the statement. PEPs can consider or discard the statement.
* Custom categories: In XACML 3.0, users are given the option to create their own custom categories. But in XACML 2.0, attributes have been organized into subject, resource, environment or action.
* Content element: In a XACML 2.0 request, there can only be XML content inside the resource category as part of the ResourceContent element. The ResourceContent element is generalized into a Content element that can be found in any category.
* Enhanced profiles:
  + The Hierarchical Resource Profile presented in XACML 2.0 has been reviewed and enhanced in XACML 3.0 to allow new scheme to encode hierarchy as URI.
  + Multiple decision profile: Multiple resource request (XACML 2.0) was renamed Multiple Decision Profile and enhanced with new variants. The profile lets a requestor-typically the Policy Enforcement Point (PEP) ask several questions in one XACML request. It enhances performance as it reduces communication overhead between PEP and PDP.
* Improvement in XACML Request and Response: As custom categories can be defined, many types of attribute categories can be in the XACML 3.0 request. XACML 2.0 request can contains only Subject, resource, environment or action categories.
* Improvements in XPath: New XPath data type has been introduced with XACML 3.0. In XACML 2.0 XPath has been defined as a String where it cannot be defined the context that the namespace prefix is going to resolve. Also XPath based multiple decisions scheme has been introduced.
* New attribute functions and datatypes: XACML 3.0 brings in new datatypes and new functions that can be used for the attributes and attribute matching. In particular XACML 3.0 utilizes XPath to manipulate attributes.
* New profiles:
  + Delegation: This is a new profile in XACML 3.0 that allows to define policies about who can write policies about what. The ability to delegate administrative rights in XACML is new as of XACML 3.0. Delegation enables global administrators to delegate constrained administrative rights to local administrators. For instance, a global administrator can define access control (AC) policies for an entire set of resources within an organization. The administrator can also delegate the right to an administrator to manage a set of resources. An administrator’s rights to define access control rules are constrained by the delegation policy that the global administrator has defined. Delegation is most useful in federation scenarios, cloud-based scenarios, and in environments where the domains to secure are so vast that they require local knowledge to define relevant policies.
  + XACML 3.0 brings additional profiles. In particular a new profile for export compliance has been produced to help author policies that can cater for export compliance scenarios. Similarly, a new profile for Intellectual Property Control (IPC) has been introduced.
* Obligations in rules: XACML 3.0 provides that rules can contain obligations. There are several improvements with Obligation in XACML 3.0 when compare to 2.0. One is the Obligation Expressions.  This would add dynamic expressions in to the obligation statements. In XACML 2.0, it needs to define the obligation element with the user email statically. But user would not be same for each XACML request. Therefore it is not possible to configure the email statically in the Obligation element. Obligation can only say PEP to “please send email to user” (let the PEP figure out the value of user’s email).

But in XACML 3.0, email of each user can be retrieved using PIP in a dynamic manner as it can define an expression element inside the ObligationExpression. Therefore Obligation can say PEP to “please send email to user@foo.com address”.

In XACML 2.0, obligations can only be added to policies and policy sets. But with XACML 3.0, rules can also contain Obligations.

* Policy combination algorithms: In XACML, policies are combined together to produce a single decision. Each policy can reach different decisions. These decisions must be combined to return a single result. XACML 3.0 enhances XACML 2.0’s existing combination algorithms.
* Scope of XPath expressions: In XACML 2.0, XPath expressions apply to the root of the XACML request. In XACML 3.0, XPath expressions apply to the root of the Content element.
* Target element: XACML 3.0 removes the disjunctive (*or*) and conjunctive (*and*) function of the category elements and introduces the *AnyOf* and *AllOf* elements. The target element still bears the conjunctive function though. Note that XACML 2.0 had already introduced and defined the any-of and all-of functions but did not have the equivalent schema elements. XACML 3.0 specification explains the behavior of the Target element and its children in XACML 3.0.
* Variables in the Obligation and Advice element: The administrator value can be determined at runtime, for instance through a policy information point (PIP). This enables richer scenarios such as: On deny, tell the PEP to send an email to the requestor’s line manager. XACML 2.0 cannot cater for such an obligation since at design-time, it does not know who the requestor is and therefore does not know who their line manager is.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_