Union internationale des télécommunications

UIT-T

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

X.1142

(06/2006)

SÉRIE X: RÉSEAUX DE DONNÉES, COMMUNICATION ENTRE SYSTÈMES OUVERTS ET SÉCURITÉ

Sécurité des télécommunications

Langage de balisage extensible de contrôle d'accès (XACML 2.0)

Recommandation UIT-T X.1142

RECOMMANDATIONS UIT-T DE LA SÉRIE X

RÉSEAUX DE DONNÉES, COMMUNICATION ENTRE SYSTÈMES OUVERTS ET SÉCURITÉ

RÉSEAUX PUBLICS DE DONNÉES	
Services et fonctionnalités	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalisation et commutation	X.50–X.89
Aspects réseau	X.90–X.149
Maintenance	X.150–X.179
Dispositions administratives	X.180–X.199
INTERCONNEXION DES SYSTÈMES OUVERTS	
Modèle et notation	X.200–X.209
Définitions des services	X.210–X.219
Spécifications des protocoles en mode connexion	X.220–X.229
Spécifications des protocoles en mode sans connexion	X.230–X.239
Formulaires PICS	X.240–X.259
Identification des protocoles	X.260–X.269
Protocoles de sécurité	X.270–X.279
Objets gérés des couches	X.280–X.289
Tests de conformité	X.290–X.299
INTERFONCTIONNEMENT DES RÉSEAUX	
Généralités	X.300–X.349
Systèmes de transmission de données par satellite	X.350–X.369
Réseaux à protocole Internet	X.370–X.379
SYSTÈMES DE MESSAGERIE	X.400–X.499
ANNUAIRE	X.500–X.599
RÉSEAUTAGE OSI ET ASPECTS SYSTÈMES	
Réseautage	X.600–X.629
Efficacité	X.630–X.639
Qualité de service	X.640–X.649
Dénomination, adressage et enregistrement	X.650–X.679
Notation de syntaxe abstraite numéro un (ASN.1)	X.680–X.699
GESTION OSI	
Cadre général et architecture de la gestion-systèmes	X.700–X.709
Service et protocole de communication de gestion	X.710–X.719
Structure de l'information de gestion	X.720–X.729
Fonctions de gestion et fonctions ODMA	X.730–X.799
SÉCURITÉ	X.800–X.849
APPLICATIONS OSI	
Engagement, concomitance et rétablissement	X.850–X.859
Traitement transactionnel	X.860–X.879
Opérations distantes	X.880–X.889
Applications génériques de l'ASN.1	X.890–X.899
TRAITEMENT RÉPARTI OUVERT	X.900–X.999
SÉCURITÉ DES TÉLÉCOMMUNICATIONS	**X.1000–**

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

Recommandation UIT-T X.1142

Langage de balisage extensible de contrôle d'accès (XACML 2.0)

Résumé

XACML est un vocabulaire XML pour exprimer les politiques de contrôle d'accès. Le contrôle d'accès consiste à décider si l'accès demandé à une ressource devrait être admis et à mettre en application cette décision. La présente Recommandation définit le cœur de XACML, y compris la syntaxe du langage, les modèles, le contexte avec le modèle de langage de politique, les règles de syntaxe et de traitement. La présente Recommandation spécifie le profil de contrôle d'accès fondé sur le cœur de XACML et le rôle hiérarchique. Elle spécifie un profil de ressources multiples de XACML et un profil SAML 2.0 de XACML. Pour améliorer la sécurité des échanges de politiques fondées sur XACML, la présente Recommandation spécifie aussi un profil de signature numérique XML de XACML pour sécuriser les données. Un profil de confidentialité est spécifié afin de fournir des lignes directrices pour l'implémentation.

La présente Recommandation est techniquement équivalente et compatible avec la norme XACML 2.0 d'OASIS.

Source

La Recommandation UIT-T X.1142 a été approuvée le 13 juin 2006 par la Commission d'études 17 (2005-2008) de l'UIT-T selon la procédure définie dans la Recommandation UIT-T A.8.

AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue.

Le respect de cette Recommandation se fait à titre volontaire. Cependant, il se peut que la Recommandation contienne certaines dispositions obligatoires (pour assurer, par exemple, l'interopérabilité et l'applicabilité) et considère que la Recommandation est respectée lorsque toutes ces dispositions sont observées. Le futur d'obligation et les autres moyens d'expression de l'obligation comme le verbe "devoir" ainsi que leurs formes négatives servent à énoncer des prescriptions. L'utilisation de ces formes ne signifie pas qu'il est obligatoire de respecter la Recommandation.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas des renseignements les plus récents, il est vivement recommandé aux développeurs de consulter la base de données des brevets du TSB sous http://www.itu.int/ITU-T/ipr/.

TABLE DES MATIÈRES

Recommandation UIT-T X.1142

Langage de balisage extensible de contrôle d'accès (XACML 2.0)

1 Domaine d'application

La présente Recommandation définit le langage de balisage de contrôle d'accès extensible (XACML, *extensible access control markup language*) Version 2.0. Elle définit un langage commun pour l'expression de la politique de sécurité. Les motivations sous-jacentes de XACML sont le développement d'un langage de politique fondé sur XML qui puisse être utilisé pour fournir:

- une méthode de combinaison des règles et politiques individuelles en un seul ensemble de politique qui s'applique à une demande de décision particulière;
- une méthode de définition souple de la procédure par laquelle les règles et les politiques se combinent;
- une méthode pour traiter plusieurs sujets agissant à des titres divers;
- une méthode appuyant une décision d'autorisation sur des attributs du sujet et de la ressource;
- une méthode pour traiter des attributs ayant plusieurs valeurs;
- une méthode appuyant une décision d'autorisation sur le contenu d'une ressource d'information;
- un ensemble d'opérateurs logiques et mathématiques sur des attributs de sujet, de ressource et d'environnement;
- une méthode de traitement d'un ensemble distribué de composants de politique, tout en faisant abstraction de la méthode de localisation, de restitution et d'authentification des composants de la politique;
- une méthode d'identification rapide de la politique qui s'applique à une action donnée, sur la base des valeurs des attributs des sujets, de ressource et d'action;
- une couche d'abstraction qui isole l'auteur de la politique des détails de l'environnement d'application;
- une méthode de spécification d'un ensemble d'actions devant être effectuées en conjonction avec la mise en application de la politique.

Les solutions XACML pour chacune de ces exigences figurent dans la présente Recommandation. En particulier, le paragraphe 7 développe le cœur du langage XACML incluant les modèles XACML, le modèle de langage de politique, la syntaxe de politique et les règles de traitement. Le paragraphe 8 développe le cœur XACML et le profil de contrôle d'accès fondé sur le rôle (RBAC, *role based access control*) hiérarchique. Le paragraphe 9 développe le profil XACML à ressource multiple. Le paragraphe 10 discute des technologies pour sécuriser la communication XACML au moyen du développement du profil SAML 2.0 de XACML. Le paragraphe 11 s'appuie sur le paragraphe 10 pour développer un profil de signature numérique XML pour XACML. Le paragraphe 12 expose la combinaison des profils XACML développés dans les paragraphes 7 à 11 pour proposer un profil de ressource hiérarchique de XACML. Les questions de confidentialité sont exposées au paragraphe 13.

2 Références normatives

Les Recommandations suivantes et autres références contiennent des dispositions qui par leur référence dans le présent texte, constituent des dispositions de la présente Recommandation. Au moment de la publication, les numéros d'édition indiqués étaient valides. Toute Recommandation et autre référence est sujette à révision, et les parties aux accords fondés sur la présente Recommandation sont invitées à rechercher s'il est possible d'appliquer l'édition la plus récente des Recommandations et autres références listées ci-dessous. Le Bureau de normalisation des télécommunications de l'UIT tient à jour la liste des Recommandations de l'UIT-T. L'IETF tient à jour la liste des RFC, ainsi que de celles qui ont été rendues obsolètes par des RFC plus récentes. Le W3C maintient à jour une liste des Recommandations et autres publications les plus récentes.

- Recommandation UIT-T X.811 (1995) | ISO/CEI 10181-2:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: cadre d'authentification*.
- Recommandation UIT-T X.812 (1995) | ISO/CEI 10181-3:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: cadre de contrôle d'accès.*
- Recommandation UIT-T X.1141 (2006), *Langage de balisage d'assertion de sécurité (SAML 2.0)*.

- IETF RFC 822 (1982), *Standard for the Format of ARPA Internet Text Messages*.
- IETF RFC 2119 (1997), *Key words for use in RFCs to Indicate Requirement Levels*.
- IETF RFC 2253 (1997), *Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names*.
- IETF RFC 2256 (1997), *A Summary of the X.500 (96) User Schema for use with LDAPv3*.
- IETF RFC 2396 (1998), *Uniform Resource Identifiers (URI): Generic Syntax*.
- IETF RFC 2732 (1999), *Format for Literal IPv6 Addresses in URL's*.
- IETF RFC 2821 (2001), *Simple Mail Transfer Protocol*.
- IETF RFC 3280 (2002), *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*.
- W3C Canonicalization:2002, *Exclusive XML Canonicalization Version 1.0*, W3C Recommendation, Copyright © [18 juillet 2002] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/xml-exc-c14n/.
- W3C Datatypes:2001, *XML Schema Part 2: Datatypes*, W3C Recommendation, Copyright © [2 mai 2001] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.
- W3C Encryption:2002, *XML Encryption Syntax and Processing*, W3C Recommendation, Copyright © [10 décembre 2002] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/.
- W3C MathML:2003, *Mathematical Markup Language (MathML), Version 2.0*, W3C Recommendation, Copyright © [21 octobre 2003] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2003/REC-MathML2-20031021/.
- W3C Signature:2002, *XML-Signature Syntax and Processing*, W3C Recommendation, Copyright © [12 février 2002] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/.
- W3C XML:2004, *Extensible Markup Language (XML) 1.0 (Third Edition)*, W3C Recommendation, Copyright © [4 février 2004] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/REC-xml/.
- W3C XPATH:1999, *XML Path Language, Version 1.0*, W3C Recommendation, Copyright © [16 novembre 1999] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/1999/REC-xpath-19991116.
- W3C XSLT:1999, *XSL Transformations (XSLT) Version 1.0*, W3C Recommendation, Copyright © [16 novembre 1999] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/1999/REC-xslt-19991116.

NOTE – La référence à un document au sein de la présente Recommandation ne lui donne pas comme document autonome, le statut d'une Recommandation.

3 Définitions

Pour les besoins de la présente Recommandation, on applique les définitions suivantes.

3.1 Définitions importées

3.1.1 La présente Recommandation utilise le terme suivant défini dans la Rec. UIT-T X.811:

a) principe.

3.1.2 La présente Recommandation utilise les termes suivants définis dans la Rec. UIT-T X.812:

a) informations de contrôle d'accès;

b) utilisateur.

3.1.3 La présente Recommandation utilise les termes suivants définis dans le Glossaire des services de la Toile du W3C:

a) espace de nom;

b) schéma XML.

3.1.4 La présente Recommandation utilise les termes suivants définis dans la RFC 2828 de l'IETF:

a) accès;

b) contrôle d'accès;

c) point d'administration de politique;

d) point de décision de politique;

e) point de mise en application de politique;

f) architecture de sécurité;

g) politique de sécurité;

h) service de sécurité.

3.1.5 La présente Recommandation utilise les termes suivants définis dans la RFC 2396 de l'IETF:

a) identifiant de ressource uniforme (URI);

b) référence d'URI.

3.1.6 La présente Recommandation utilise le terme suivant défini dans Signature XML du W3C:

a) objet de données.

3.2 Définitions supplémentaires

3.2.1 **accès**: effectuer une action.

3.2.2 **contrôle d'accès**: contrôler l'accès conformément à une politique.

3.2.3 **action**: opération sur une ressource.

3.2.4 **politique applicable**: ensemble des politiques et ensembles de politiques qui gouvernent l'accès pour une demande de décision spécifique.

3.2.5 **attribut**: caractéristique d'un sujet, ressource, action ou environnement à laquelle on peut faire référence dans un prédicat ou une cible.

3.2.6 **autorité d'attribut (AA, *attribute authority*)**: entité qui lie des attributs à des identités. Une telle liaison peut être exprimée en utilisant une assertion d'attribut SAML produite par l'autorité d'attribut.

3.2.7 **décision d'autorisation**: résultat de l'évaluation d'une politique applicable, retournée par le PDP au PEP. Fonction qui évalue comme "Permis", "Refusé", "Indéterminé" ou "NonApplicable", et (facultativement) un ensemble d'obligations.

3.2.8 **sac**: collection non ordonnée de valeurs, dans laquelle des valeurs peuvent être dupliquées.

3.2.9 **condition**: expression de prédicats. Fonction qui évalue comme "Vrai", "Faux" ou "Indéterminé".

3.2.10 **séquence conjonctive**: séquence de prédicats combinés en utilisant l'opération logique 'ET'.

3.2.11 **contexte**: représentation canonique d'une demande de décision et d'une décision d'autorisation.

3.2.12 gestionnaire de contexte: entité système qui convertit les demandes de décision en format de demande d'origine dans le format canonique de XACML et convertit les décisions d'autorisation en forme canonique XACML dans le format de réponse d'origine.

3.2.13 gardien: entité à laquelle sont confiées les informations permettant d'identifier la personne.

3.2.14 objet de données: se réfère à un objet numérique qui est signé. Un objet de données est référencé à l'intérieur d'un élément `<Reference>` en utilisant un URI.

3.2.15 décision: résultat de l'évaluation d'une règle, politique ou ensemble de politiques.

3.2.16 demande de décision: demande d'un PEP à un PDP de prendre une décision d'autorisation.

3.2.17 séquence disjonctive: séquence de prédicats combinés en utilisant l'opération logique 'OU'.

3.2.18 effet: conséquence voulue d'une règle satisfaite ("Permis" ou "Refusé").

3.2.19 environnement: ensemble des attributs qui sont pertinents pour une décision d'autorisation et sont indépendants d'un sujet, ressource ou action particulier.

3.2.20 politique HasPrivilegesOfRole: type facultatif de `<Policy>` qui peut être inclus dans une permission `<PolicySet>` pour permettre la prise en charge d'interrogations demandant si un sujet "a les privilèges" d'un rôle spécifique.

3.2.21 ressource hiérarchique: ressource organisée comme une arborescence ou une forêt (graphe acyclique dirigé) de ressources individuelles appelées nœuds.

3.2.22 rôle junior: dans une hiérarchie de rôle, le rôle A est *junior* pour le rôle B si le rôle B hérite de toutes les permissions associées au rôle A.

3.2.23 permissions multirôle: ensemble de permissions pour lesquelles un utilisateur doit tenir simultanément plus d'un rôle afin d'obtenir l'accès.

3.2.24 attribut désigné: instance spécifique d'un attribut, déterminée par le nom et le type de l'attribut, l'identité du détenteur de l'attribut (qui peut être du type: sujet, ressource, action ou environnement) et (facultativement) l'identité de l'autorité productrice.

3.2.25 nœud: ressource individuelle qui fait partie d'une ressource hiérarchique.

3.2.26 obligation: opération spécifiée dans une politique ou ensemble de politiques qui devrait être effectuée par le PEP conjointement avec la mise en application d'une décision `d'autorisation`.

3.2.27 propriétaire: sujet d'informations d'identification personnelles.

3.2.28 permission: capacité ou droit d'effectuer une action sur une ressource, éventuellement sous certaines conditions spécifiées.

3.2.29 permission <policy set> (PPS): `<PolicySet>` qui contient les permissions réelles associées à un rôle donné.

3.2.30 point d'informations de politique (PIP): entité système qui agit comme source des valeurs d'attribut.

3.2.31 ensemble de politiques: ensemble de politiques, autres ensembles de politiques, algorithme de combinaison de politiques et (facultativement) ensemble d'obligations. Peut être un composant d'un autre ensemble de politiques.

3.2.32 prédicat: déclaration sur des attributs dont la véracité peut être évaluée.

3.2.33 ressource: données, service ou composant de système.

3.2.34 rôle: fonction de tâche dans le contexte d'une organisation qui a une sémantique associée en ce qui concerne l'autorité et la responsabilité conférée à l'utilisateur affecté à ce rôle.

3.2.35 contrôle d'accès fondé sur le rôle (RBAC, *role based access control*): modèle pour le contrôle de l'accès à des ressources où les actions permises sur les ressources sont identifiées par les rôles plutôt que par les identités des sujets individuels.

3.2.36 autorité d'activation de rôle: entité qui alloue les attributs et valeurs de rôle aux utilisateurs ou active les attributs et valeurs de rôle durant une session d'utilisateur.

3.2.37 rôle `<PolicySet>` (RPS) : `<PolicySet>` qui associe les détenteurs d'attribut et valeur d'un rôle donné à une permission `<PolicySet>` qui contient les permissions réelles associées à ce rôle.

3.2.38 **rôle senior**: dans une hiérarchie de rôle, le rôle A est *senior* pour le rôle B si le rôle A hérite de toutes les permissions associées au rôle B.

3.2.39 **règle**: cible, effet et condition. Composant d'une politique.

3.2.40 **algorithme de composition de règles**: procédure pour combiner les décisions provenant de plusieurs règles.

3.2.41 **sujet**: acteur dont les attributs peuvent être désignés par un prédicat.

3.2.42 **cible**: ensemble des demandes de décision, identifié par les définitions de ressources, sujets et actions, qu'une règle, une politique ou un ensemble de politiques est destiné à évaluer.

3.2.43 **unification de type**: méthode par laquelle deux expressions de type sont "unifiées". Les expressions de type sont confrontées à leur structure. Lorsqu'une variable de type apparaît dans une expression, elle est alors "unifiée" pour représenter l'élément de structure correspondant de l'autre expression, qu'elle soit une autre variable ou une sous-expression. Toutes les allocations de variables doivent rester cohérentes dans les deux structures. L'unification échoue si les deux expressions ne peuvent pas être alignées, soit parce qu'elles ont une structure différente, soit en ayant des instances de conflit, comme lorsqu'une variable doit représenter à la fois "**xs:string**" et "**xs:integer**".

4 Abréviations

AA	autorité d'attribut (*attribute authority*)
ASP	fournisseur de service d'application (*application service provider*)
CA	autorité de certification (*certification authority*)
CMP	protocole de gestion de certificat (*certificate management protocol*)
CRL	liste de révocation de certificat (*certificate revocation list*)
ECP	client/mandataire amélioré (e*nhanced client/proxy*)
HTTP	protocole de transfert hypertexte (*hypertext transfer protocol*)
ID	identifiant (*identifier*)
IPSEC	protocole de sécurité IP (*IP security protocol*)
LDAP	protocole allégé d'accès à un répertoire (*lightweight directory access protocol*)
PAP	point d'administration de politique (*policy administration point*)
PDP	point de décision de politique (*policy decision point*)
PEP	point de mise en application de politique (*policy enforcement point*)
PIP	point d'information de politique (*policy information point*)
PKI	infrastructure de clé publique (*public-key infrastructure*)
POP	preuve de possession (*proof of possession*)
PPS	permission `<Policy Set>`
RA	autorité d'enregistrement (*registration authority*)
RBAC	contrôle d'accès fondé sur le rôle (*role based access control*)
RPS	rôle `<PolicySet>`
RSA	algorithme de clé publique Rivest Shamir Adleman (*Rivest Shamir Adleman (public key algorithm*)
SAML	langage de balisage d'assertion de sécurité (*security assertion markup language*)
SP	fournisseur de services (*service provider*)
SSO	signature unique (*single sign on*)
TLS	sécurité de la couche Transport (*transport layer security*)
URI	identifiant de ressource uniforme (*uniform resource identifier*)
URN	nom de ressource uniforme (*uniform resource name*)
XML	langage de balisage extensible (*extensible markup language)*
XPath	langage de chemin XML (*XML path language*)
XSLT	langage de feuille de style extensible (*extensible stylesheet language*)

5 Conventions

La présente Recommandation utilise les mots clés "doit", "ne doit pas", "exige", "devrait", "ne devrait pas", "recommande", "peut", et "facultatif". Dans la présente Recommandation, ces termes sont à interpréter comme décrit dans la RFC 2119.

Dans la présente Recommandation, la phrase "Normatif, mais facultatif" signifie que la fonctionnalité décrite est facultative pour les implémentations XACML conformes, mais si la fonctionnalité est prétendument prise en charge conformément à ce profil, elle doit alors être prise en charge de la façon décrite.

Dans les descriptions de la syntaxe, les éléments entre crochets angulaires ("<", ">") sont à remplacer par les valeurs appropriées, celles entre crochets ("[", "]") englobent des éléments facultatifs, les éléments entre guillemets sont des composants littéraux, et "*" indique que l'élément précédent peut survenir zéro, une, ou plusieurs fois.

6 Aperçu général

Les "économies d'échelle" ont conduit les fabricants de plates-formes de calcul à développer des produits ayant des fonctionnalités très générales, de sorte qu'elles puissent être utilisées dans la plus large gamme possible de situations. "Hors de la boîte", ces produits ont les possibilités maximales d'accession aux données et d'exécution des logiciels, de sorte qu'ils puissent être utilisés dans le plus d'environnements d'application possibles, y compris ceux qui ont les politiques de sécurité les plus permissives. Dans le cas le plus courant d'une politique de sécurité relativement restrictive, les privilèges inhérents à la plate-forme doivent être restreints, par configuration.

La politique de sécurité d'une grande entreprise a de nombreux éléments et points de mise en application. Les éléments de politique peuvent être gérés par le bureau des systèmes d'information, par celui des ressources humaines, par le bureau des affaires juridiques et par le bureau des finances. Et la politique peut être mise en application par l'extranet, la messagerie électronique, le WAN, et les systèmes d'accès à distance, plates-formes qui par nature mettent en application une politique de sécurité permissive. La pratique courante consiste à gérer indépendamment chaque point de mise en application afin d'implémenter la politique de sécurité de façon aussi précise que possible. Par conséquent, modifier la politique de sécurité est une proposition coûteuse et non fiable. Et il est virtuellement impossible d'avoir une vue d'ensemble des sauvegardes effectives à travers toute l'entreprise pour la mise en application de la politique. En même temps, il y a une pression croissante sur la direction de l'entreprise, de la part des consommateurs, des actionnaires et du réglementeur pour qu'elle fasse la preuve qu'elle utilise ce qui se fait de mieux pour le capital d'informations de l'entreprise et de ses consommateurs.

Pour ces raisons, il y a un pressant besoin d'un langage commun pour exprimer les politiques de sécurité. S'il est mis en œuvre dans l'ensemble d'une entreprise, un langage de politique commun permet à l'entreprise de gérer la mise en application de tous les éléments de sa politique de sécurité dans tous les composants des ses systèmes d'information. La gestion de la politique de sécurité peut inclure tout ou partie des étapes suivantes: écriture, révision, essai, approbation, production, combinaison, analyse, modification, retrait, restitution et mise en application de la politique.

7 Cœur de XACML

Le présent paragraphe développe les fondements de XACML qui comprennent les exigences de politique générale, les modèles, le contexte général, la syntaxe de la politique, et donne quelques exemples.

7.1 Fondements

Le présent paragraphe est pour information.

XML est le choix naturel comme base du langage commun de politique de sécurité, du fait que sa syntaxe et sa sémantique peuvent facilement être étendues pour s'accommoder des exigences particulières d'une application, et du large soutien dont il jouit de la part des principaux fabricants de plates-formes et d'outils.

7.1.1 Exigences

Les exigences de base d'un langage de politique pour l'expression de la politique de sécurité des systèmes d'information sont de fournir:

- une méthode de combinaison des règles et politiques individuelles en un seul ensemble de politiques qui s'applique à une demande de décision particulière;
- une méthode de définition souple de la procédure par laquelle sont combinées les règles et politiques;
- une méthode de traitement de sujets multiples agissant à des titres différents;

- une méthode pour fonder les décisions d'autorisation sur les attributs du sujet et de la ressource;
- une méthode pour traiter des attributs multivaleurs;
- une méthode pour fonder une décision d'autorisation sur le contenu d'une ressource d'information;
- un ensemble d'opérateurs logiques et mathématiques sur les attributs de sujet, ressource et environnement;
- une méthode de traitement d'ensembles distribués de composants de politique, tout en faisant abstraction de la méthode de localisation, de restitution et d'authentification des composants de la politique;
- une méthode d'identification rapide de la politique qui s'applique à une action donnée, sur la base des valeurs des attributs de sujet, ressources et action;
- une couche d'abstraction qui isole l'auteur de la politique des détails de l'environnement d'application;
- une méthode de spécification d'un ensemble d'actions qui doivent être effectuées en conjonction avec la mise en application de la politique.

La motivation sous-jacente à XACML est d'exprimer ces idées bien établies dans le champ de la politique de contrôle d'accès en utilisant une extension du langage XML. Les solutions XACML pour chacune de ces exigences sont exposées dans les paragraphes suivants.

7.1.2 Combinaison de règle et de politique

La politique complète applicable à une demande de décision particulière peut être composée d'un certain nombre de règles ou politiques individuelles. Par exemple, dans une application de confidentialité personnelle, le propriétaire des informations personnelles peut définir certains aspects de la politique de divulgation, alors que l'entreprise qui est la gardienne des informations peut définir certains autres aspects. Pour prendre une décision d'autorisation, il doit être possible de combiner les deux politiques séparées pour former la politique unique applicable à la demande.

XACML définit trois éléments de politique de niveau supérieur: `<Rule>` (règle), `<Policy>` (*politique)* et `<PolicySet>` (ensemble de politiques). L'élément `<Rule>` contient une expression booléenne qui peut être évaluée isolément, mais qui n'est pas destinée à un accès isolé de la part d'un PDP. Et donc, il n'est pas destiné à former par lui-même la base d'une décision d'autorisation. Il n'est destiné à une existence isolée qu'au sein d'un PAP XACML, dans lequel il peut former l'unité de base de gestion, et être réutilisé dans plusieurs politiques.

L'élément `<Policy>` contient un ensemble d'éléments `<Rule>` et une procédure spécifiée pour combiner les résultats de leur évaluation. C'est l'unité de base de la politique utilisée par le PDP, et il est donc destiné à former la base d'une décision d'autorisation.

L'élément `<PolicySet>` contient un ensemble d'éléments `<Policy>` ou autres `<PolicySet>` et une procédure spécifiée pour combiner les résultats de leur évaluation. Il est le moyen de combinaison standard de politiques séparées en une seule politique combinée.

7.1.3 Algorithmes de combinaison

XACML définit un certain nombre d'algorithmes de combinaison qui peuvent être identifiés par un attribut `RuleCombiningAlgId` ou `PolicyCombiningAlgId`, respectivement des éléments `<Policy>` ou `<PolicySet>`. L'algorithme de combinaison de règle définit une procédure pour arriver à une décision d'autorisation étant donné les résultats individuels de l'évaluation d'un ensemble de règles. De même, l'algorithme de combinaison de politiques définit une procédure pour arriver à une décision d'autorisation étant donné les résultats individuels de l'évaluation d'un ensemble de politiques. Des algorithmes de combinaison standard sont définis pour:

- Deny-overrides *(refus prioritaire)* (ordonné et non ordonné);
- Permit-overrides *(permission prioritaire)* (ordonné et non ordonné);
- First-applicable *(le premier s'applique)*;
- Only-one-applicable *(une seule est applicable).*

Dans le cas de l'algorithme Deny-overrides, si on rencontre un seul élément `<Rule>` ou `<Policy>` qui évalue à "Deny" *(refus)*, indépendamment du résultat d'évaluation des autres éléments `<Rule>` ou `<Policy>` dans la politique applicable, le résultat combiné est alors "Deny".

De même, dans le cas de l'algorithme Permit-overrides, si un seul résultat "Permit" *(permis)* est rencontré, le résultat combiné est alors "Permit".

Dans le cas de l'algorithme de combinaison "First-applicable", le résultat combiné est le même que le résultat de l'évaluation du premier élément `<Rule>`, `<Policy>` ou `<PolicySet>` dans la liste des règles dont la cible est applicable à la demande de décision.

L'algorithme de combinaison de politique "Only-one-applicable" ne s'applique qu'aux politiques. Le résultat de cet algorithme de combinaison garantit qu'une seule politique ou un seul ensemble de politiques est applicable en vertu de leur cible. Si aucune politique ou ensemble de politiques ne s'applique, le résultat est alors "NotApplicable" *(non applicable)*, mais si plus d'une politique ou ensemble de politiques sont applicables, le résultat est alors "Indeterminate" *(indéterminé)*. Lorsque exactement une politique ou ensemble de politiques est applicable, le résultat de l'algorithme de combinaison est le résultat de l'évaluation de la seule politique ou ensemble de politiques applicable.

Les politiques et ensembles de politiques peuvent prendre des paramètres qui modifient le comportement des algorithm es de combinaison. Cependant, aucun des algorithmes de combinaison standard n'est affecté par les paramètres.

Les utilisateurs de la présente Recommandation peuvent, si nécessaire, définir leurs propres algorithmes de combinaison.

7.1.4 Sujets multiples

Les politiques de contrôle d'accès imposent souvent des exigences aux actions de plus d'un sujet. Par exemple, la politique qui gouverne l'exécution d'une transaction financière de grande importance peut exiger l'approbation de plusieurs individus, agissant à divers titres. Donc, XACML reconnaît qu'il peut y avoir plus d'un sujet pertinent pour une demande de décision. Un attribut appelé "subject-category" *(catégorie de sujet)* est utilisé pour différencier les sujets qui agissent à des titres divers. Certaines valeurs standards sont spécifiées pour cet attribut, et les utilisateurs peuvent en définir d'autres en plus.

7.1.5 Politiques fondées sur des attributs de sujet et de ressource

Une autre exigence commune est de fonder une décision d'autorisation sur des caractéristiques du sujet, autres que son identité. Peut-être, l'application la plus courante de cette idée est le rôle du sujet. XACML fournit des facilités pour la prise en charge de cette approche. Les attributs des sujets contenus dans le contexte de la demande peuvent être identifiés par l'élément `<SubjectAttributeDesignator>`. Cet élément contient un URN qui identifie l'attribut. Autrement, l'élément `<AttributeSelector>` peut contenir une expression XPath sur le contexte de la demande pour identifier une valeur d'attribut de sujet particulière par sa localisation dans le contexte.

XACML fournit une manière standard de référencer les attributs définis dans la RFC 2253 de l'IETF. Elle est destinée à encourager les implémentations à utiliser des identifiants d'attribut standard pour des attributs de sujet communs.

Une autre exigence commune est de fonder une décision d'autorisation sur des caractéristiques de la ressource autres que son identité. XACML fournit la faculté de prendre en charge cette approche. Les attributs de la ressource peuvent être identifiés par l'élément `<ResourceAttributeDesignator>` *(désignateur d'attribut de ressource)*. Cet élément contient un URN qui identifie l'attribut. Autrement, l'élément `<AttributeSelector>` *(sélecteur d'attribut)* peut contenir une expression XPath sur le contexte de la demande pour identifier une valeur d'attribut de ressource particulière par sa localisation dans le contexte.

7.1.6 Attributs multivaleurs

Les techniques les plus communes pour communiquer les attributs (LDAP, XPath, SAML, etc.) prennent en charge plusieurs valeurs par attribut. Donc, lorsqu'un PDP XACML restitue la valeur d'un attribut désigné, le résultat peut contenir plusieurs valeurs. Une collection de telles valeurs est appelée un sac. Un sac diffère d'un ensemble en ce qu'il peut contenir des valeurs dupliquées, alors qu'un ensemble ne le peut pas. Parfois, cette situation représente une erreur. Parfois, la règle XACML est satisfaite si une des valeurs de l'attribut satisfait aux critères exprimés dans la règle.

XACML fournit un ensemble de fonctions qui permettent à l'auteur d'une politique d'être absolument explicite sur la façon dont un PDP devrait traiter les cas de valeurs d'attribut multiples. Ce sont les fonctions "d'ordre supérieur".

7.1.7 Politiques fondées sur le contenu de la ressource

Dans de nombreuses applications, il est exigé qu'une décision d'autorisation soit fondée sur des données contenues dans la ressource d'information à laquelle l'accès est demandé. Par exemple, un composant courant de politique de confidentialité est qu'une personne devrait être admise à lire les enregistrements dont elle est le sujet. La politique correspondante doit contenir une référence au sujet identifié dans la ressource d'information elle-même.

XACML fournit la faculté de faire cela lorsque la ressource d'information peut être représentée comme un document XML. L'élément `<AttributeSelector>` peut contenir une expression XPath sur le contexte de la demande pour identifier des données dans la ressource d'informations à utiliser dans l'évaluation de politique.

7.1.8 Opérateurs

Les politiques de sécurité des informations opèrent sur les attributs des sujets, sur la ressource, sur l'action et sur l'environnement afin d'arriver à une décision d'autorisation. Dans le processus menant à la décision d'autorisation, les attributs de nombreux types différents peuvent devoir être comparés ou calculés. Par exemple, dans une application

financière, le crédit disponible d'une personne peut devoir être calculé en ajoutant sa limite de crédit à sa balance comptable. Le résultat peut alors devoir être comparé à la valeur de la transaction. Cette sorte de situation amène au besoin d'opérations arithmétiques sur les attributs du sujet (balance comptable et limite de crédit) et de la ressource (valeur de la transaction).

Encore plus couramment, une politique peut identifier l'ensemble des rôles auxquels il est permis de jouer une action particulière. L'opération correspondante implique de vérifier s'il y a une intersection non vide entre l'ensemble des rôles joués par le sujet et l'ensemble des rôles identifiés dans la politique. D'où le besoin d'opérations d'ensemble.

XACML inclut un certain nombre de fonctions incorporées et une méthode d'ajout de fonctions non standard. Ces fonctions peuvent être enchâssées pour construire des expressions complexes arbitraires. Ceci est réalisé avec l'élément `<Apply>` *(appliquer)*. L'élément `<Apply>` a un attribut XML appelé `FunctionId` qui identifie la fonction à appliquer au contenu de l'élément. Chaque fonction standard est définie pour une combinaison spécifique de type de données et d'argument, et son type de données de retour est aussi spécifié. Donc, la cohérence du type de données de la politique peut être vérifiée au moment où la politique est écrite ou analysée. Et les types des valeurs de données présentés dans le contexte de la demande peuvent être confrontés aux valeurs attendues par la politique pour garantir un résultat prévisible.

En plus des opérateurs sur les arguments numériques et d'ensembles, des opérateurs sont définis pour les arguments de date, d'heure et de durée.

Les opérateurs de relations (égalité et comparaison) sont aussi définis pour un certain nombre de types de données, y compris les formes de nom, les chaînes, URI, etc., de la RFC 822 de l'IETF et de la Rec. UIT-T X.500.

Il vaut aussi de noter les opérateurs sur les types de données booléens, qui permettent des combinaisons logiques de prédicats dans une règle. Par exemple, une règle peut contenir la déclaration que l'accès peut être permis durant les heures de bureau ET à partir d'un terminal sur le lieu de travail.

7.1.9 Distribution de la politique

Dans un système distribué, des déclarations de politique individuelles peuvent être écrites par plusieurs auteurs de politique et mises en application en plusieurs points de mise en application. En plus de faciliter la collecte et la combinaison de composants de politique indépendants, cette approche permet de mettre à jour à la demande les politiques. Les déclarations de politique XACML peuvent être distribuées de toutes sortes de façons. Mais XACML ne décrit aucune façon normalisée de le faire. Indépendamment des moyens de distribution, les PDP sont supposés confirmer, en examinant l'élément `<Target>` *(cible)* de la politique, que celle-ci est applicable à la demande de décision en cours de traitement.

Les éléments `<Policy>` peuvent être rattachés aux ressources d'informations auxquelles ils s'appliquent. Autrement, les éléments `<Policy>` peuvent être conservés dans une ou plusieurs localisations à partir desquelles ils sont restitués pour l'évaluation. Dans de tels cas, la politique applicable peut être référencée par un identifiant ou localiseur étroitement associé à la ressource d'informations.

7.1.10 Indexation de politique

Pour l'efficacité de l'évaluation et pour faciliter la gestion, la politique globale de sécurité en application dans une entreprise peut être exprimée sous forme de plusieurs composants de politique indépendants. Dans ce cas, il est nécessaire d'identifier et de restituer la déclaration de politique applicable et de vérifier que c'est celle qui est correcte pour l'action demandée avant de l'évaluer. C'est l'objet de l'élément `<Target>` dans XACML.

Deux approches sont acceptées:

1) les déclarations de politique peuvent être mémorisées dans une base de données. Dans ce cas, le PDP devrait former une interrogation de la base de données pour restituer les politiques qui sont applicables à l'ensemble des demandes de décision auxquelles il est supposé répondre. De plus, le PDP devrait évaluer l'élément `<Target>` des déclarations de politique ou ensembles de politiques restitués;

1) autrement, le PDP peut être chargé avec toutes les politiques disponibles et évaluer leurs éléments `<Target>` dans le contexte d'une demande de décision particulière, afin d'identifier les politiques et ensembles de politiques qui sont applicables à cette demande.

7.1.11 Couche d'abstraction

Les PEP surviennent sous des formes diverses. Par exemple, un PEP peut faire partie d'une passerelle d'accès distant, faire partie d'un serveur de la Toile ou faire partie d'un agent d'utilisateur de messagerie électronique. Il n'est pas irréaliste de s'attendre à ce que tous les PEP dans une entreprise produisent couramment sous un format commun, ou produiront bientôt, des demandes de décision à un PDP. A tout le moins, une politique particulière peut devoir être mise en application par plusieurs PEP. Il serait inefficace de forcer un auteur de politique à écrire la même politique de

plusieurs façons différentes afin de s'accommoder des exigences de format de chaque PEP. De même, les attributs peuvent être contenus dans divers types d'enveloppe (par exemple, des certificats d'attribut X.509, des assertions d'attribut SAML, etc.). Donc, il y a besoin d'une forme canonique de la demande et de la réponse traitées par un PDP XACML. Cette forme canonique est appelée le contexte XACML. Sa syntaxe est définie dans le schéma XML.

Naturellement, les PEP conformes à XACML peuvent produire des demandes et recevoir des réponses sous la forme d'un contexte XACML. Mais, lorsque cette situation n'existe pas, une étape intermédiaire est nécessaire pour convertir du format de demande/réponse compris par le PEP au format de contexte XACML compris par le PDP.

L'intérêt de cette approche est que les politiques peuvent être écrites et analysées indépendamment de l'environnement spécifique dans lequel elles sont à mettre en application.

Dans le cas où le format de demande/réponse d'origine est spécifié dans les schémas XML (par exemple, un PEP conforme à SAML), la transformation entre le format d'origine et le contexte XACML peut être spécifiée sous la forme d'une transformation de langage de feuille de style extensible.

De même, dans le cas où la ressource à laquelle l'accès est demandé est un document XML, la ressource elle-même peut être incluse dans le contexte de la demande, ou y être référencée. Alors, grâce à l'utilisation d'expressions XPath dans la politique, les valeurs dans la ressource peuvent être incluses dans l'évaluation de politique.

7.1.12 Actions effectuées en conjonction avec la mise en application

Dans de nombreuses applications, les politiques spécifient des actions qui doivent être effectuées, soit à la place, soit en plus, d'actions qui peuvent être effectuées. XACML procure des facilités pour spécifier des actions qui doivent être effectuées en conjonction avec l'évaluation de politique dans l'élément `<Obligations>`. Il n'y a pas de définition standard pour ces actions dans la version 2.0 de XACML. Donc, un accord bilatéral est nécessaire pour une interprétation correcte entre un PAP et le PEP qui va mettre en application sa politique. Les PEP qui se conforment à la v2.0 de XACML doivent refuser l'accès sauf s'ils comprennent et peuvent s'acquitter de tous les éléments `<Obligations>` associés à la politique applicable. Les éléments `<Obligations>` sont retournés au PEP pour la mise en application.

7.2 Modèles XACML

Le présent paragraphe est pour information.

Le modèle de flux de données et le modèle de langage de XACML sont décrits dans les paragraphes suivants.

7.2.1 Modèle de flux de données

Les acteurs majeurs dans le domaine XACML sont indiqués dans le diagramme des flux de données de la Figure 7-1.

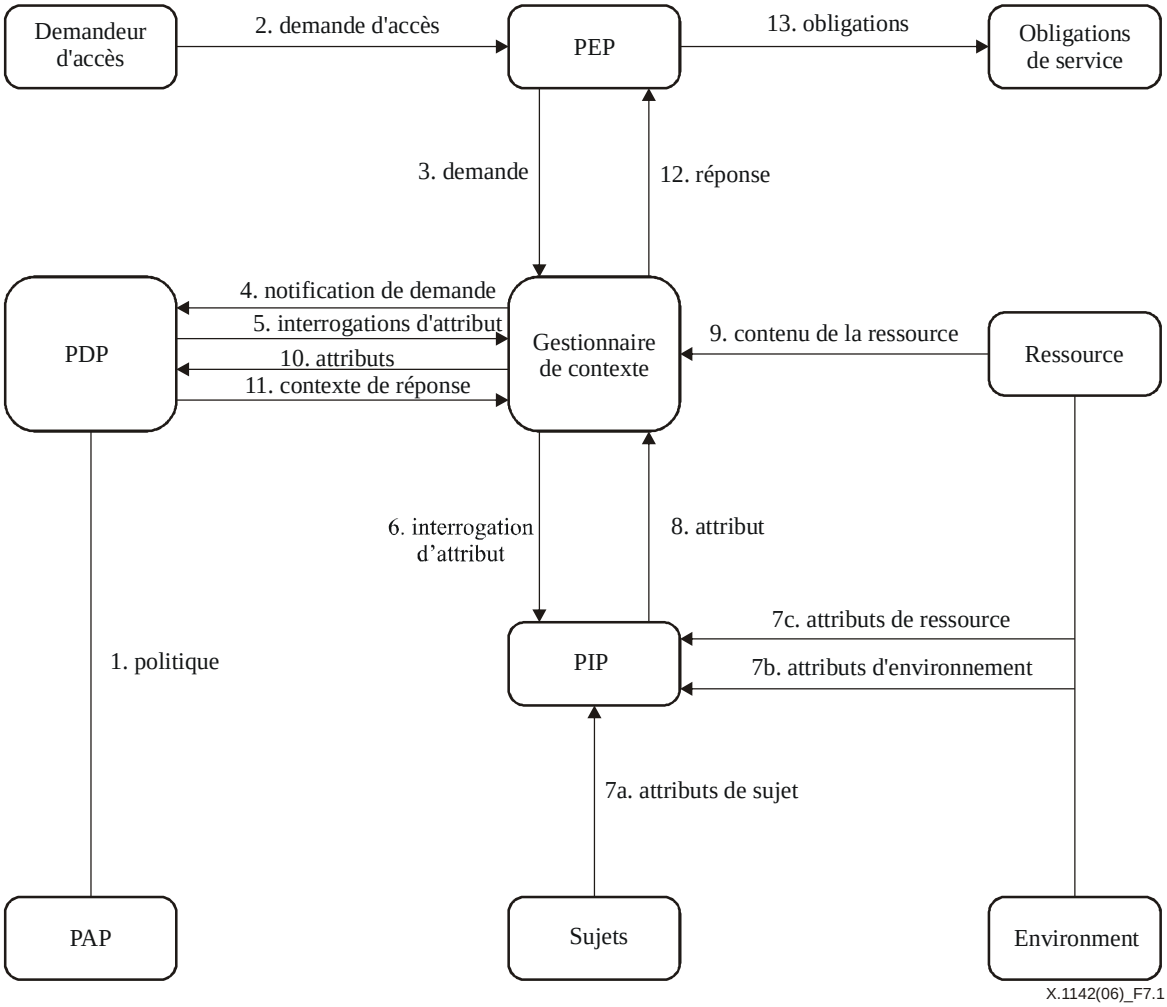

Figure 7-1/X.1142 – Diagramme de flux de données

NOTE – Certains des flux de données présentés dans le diagramme peuvent être facilités par un répertoire. Par exemple, les communications entre le gestionnaire de contexte et le PIP ou les communications entre le PDP et le PAP peuvent être facilitées par un répertoire. La présente Recommandation n'est pas destinée à formuler des restrictions sur la localisation de tels répertoires, ou bien sûr, à prescrire un protocole de communication particulier pour aucun des flux de données.

Le modèle opère selon les étapes suivantes:

1) les PAP écrivent les politiques et ensembles de politiques et les rendent disponibles au PDP. Ces politiques ou ensembles de politiques représentent la politique complète pour une cible spécifiée;
2) le demandeur d'accès envoie une demande d'accès au PEP;
3) le PEP envoie la demande d'accès au gestionnaire de contexte dans son format de demande d'origine, incluant facultativement les attributs des sujets, ressources, actions et environnement;
4) le gestionnaire de contexte construit un contexte de demande XACML et l'envoie au PDP;
5) le PDP demande tous les attributs supplémentaires de sujet, ressource, action et environnement de la part du gestionnaire de contexte;
6) le gestionnaire de contexte demande les attributs à un PIP;
7) le PIP obtient les attributs demandés;
8) le PIP retourne les attributs demandés au gestionnaire de contexte;
9) facultativement, le gestionnaire de contexte inclut les ressources dans le contexte;
10) le gestionnaire de contexte envoie les attributs demandés et (facultativement) les ressources au PDP. Le PDP évalue la politique;
11) le PDP retourne le contexte de réponse (y compris la décision d'autorisation) au gestionnaire de contexte;
12) le gestionnaire de contexte traduit le contexte de réponse dans le format de réponse d'origine du PEP. Le gestionnaire de contexte retourne la réponse au PEP;

13) le PEP remplit ses obligations;

14) (non montré) Si l'accès est permis, le PEP permet alors l'accès à la ressource; autrement, il refuse l'accès.

7.3 Contexte XACML

XACML est conçu pour convenir à divers environnements d'application. Le cœur du langage est isolé de l'environnement d'application par le contexte XACML, comme indiqué à la Figure 7-2, dans laquelle la portée de XACML est indiquée par la zone ombrée. Le contexte XACML est défini dans le schéma XML, qui décrit une représentation canonique des entrées et des sorties du PDP. Les attributs référencés par une instance de politique XACML peuvent être sous la forme d'expressions XPath sur le contexte, ou sous la forme de désignations d'attribut qui identifient l'attribut par sujet, ressource, action ou environnement et son identifiant, type de données et (facultativement) son producteur. Les implémentations doivent convertir les représentations d'attribut dans l'environnement de l'application (par exemple, SAML) et les représentations d'attribut dans le contexte XACML. La façon de le réaliser sort du domaine d'application de la présente Recommandation. Dans certains cas, tels que SAML, cette conversion peut être accomplie de façon automatique grâce à l'utilisation d'une transformation XSLT (voir XSLT:1999 du W3C).

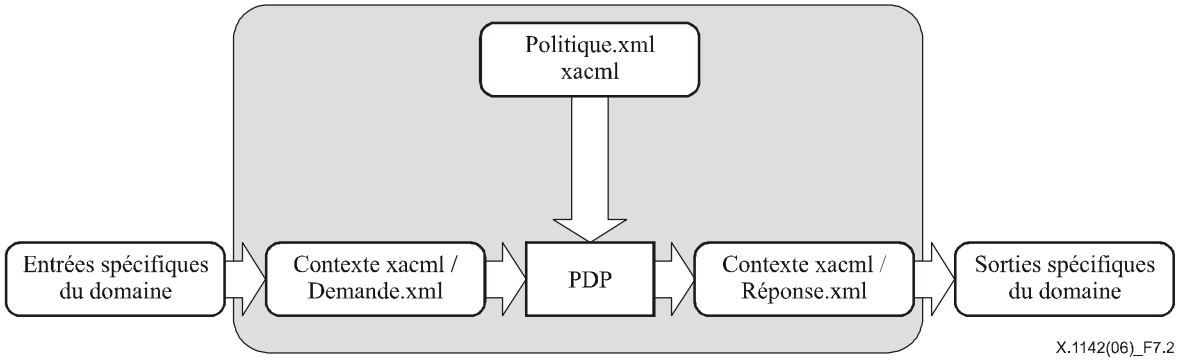

Figure 7-2/X.1142 – Le contexte XACML

Le PDP n'est pas obligé d'opérer directement sur la représentation XACML d'une politique. Il peut opérer directement sur une représentation de remplacement.

7.3.1 Modèle de langage de politique

Le modèle de langage de politique est exposé à la Figure 7-3. Les principaux composants du modèle sont:

- règle;
- politique;
- ensemble de politiques.

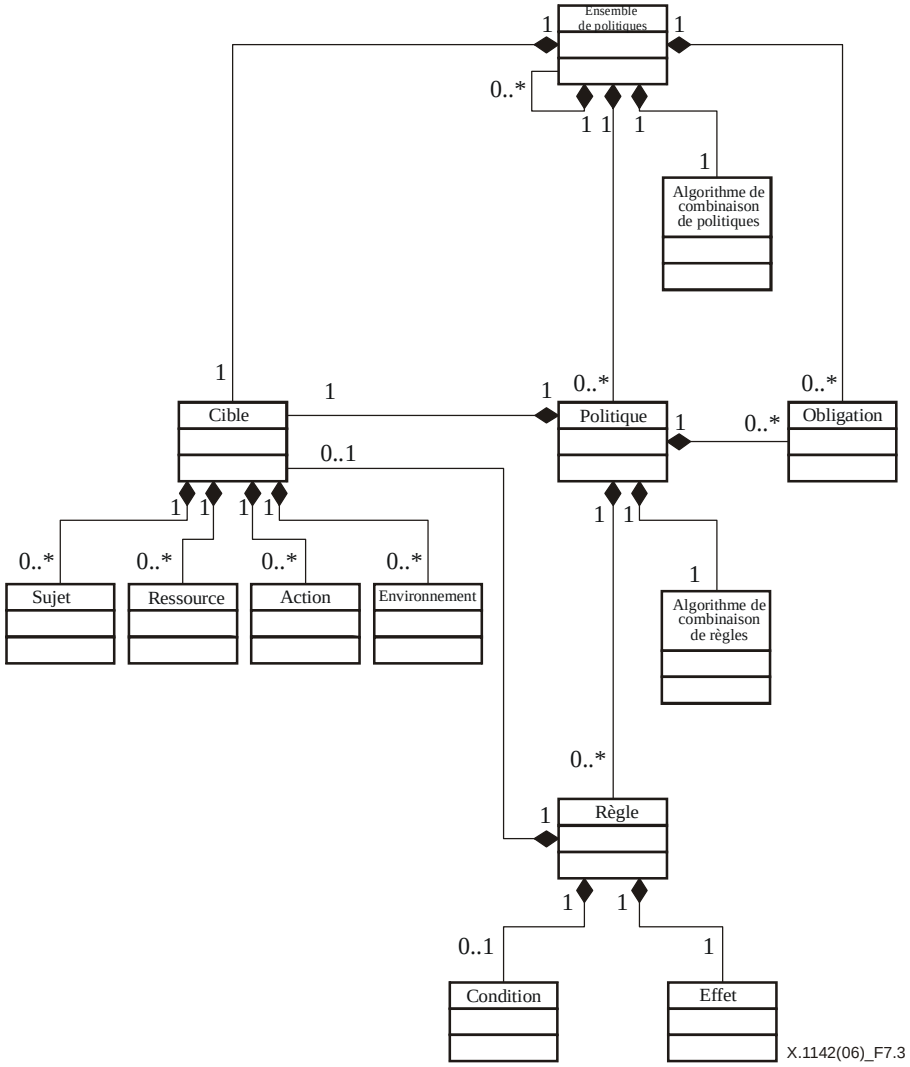

Figure 7-3/X.1142 – Modèle de langage de politique

7.3.1.1 Règle

Une règle est l'unité la plus élémentaire de la politique. Elle ne peut exister isolément qu'au sein des acteurs majeurs du domaine XACML. Pour échanger des règles entre les acteurs majeurs, elle doit être encapsulée dans une politique. Une règle peut être évaluée sur la base de son contenu. Les principaux composants d'une règle sont:

- une cible;
- un effet;
- une condition.

7.3.1.1.1 Cible de règle

La cible définit l'ensemble des:

- ressources;
- sujets;
- actions;
- environnements,

auxquels la règle est destinée à s'appliquer. L'élément `<Condition>` peut encore préciser l'applicabilité établie par la cible. Si la règle est destinée à s'appliquer à toutes les entités d'un type de données particulier, l'entité correspondante est alors omise de la cible. Un PDP XACML vérifie que les correspondances définies par la cible sont satisfaites par les attributs de sujet, ressource, action et environnement dans le contexte de la demande. Les définitions de cible sont discrètes, afin que les règles applicables puissent être efficacement identifiées par le PDP.

L'élément <Target> peut être absent d'une <Rule>. Dans ce cas, la cible de la <Rule> est la même que celle de l'élément <Policy> parent.

Certaines formes de nom de sujet, formes de nom de ressource et certains types de ressource ont une structure interne. Par exemple, la forme de nom de répertoire X.500 et la forme de nom de la RFC 822 de l'IETF sont des formes de nom de sujet structurées, alors qu'un numéro de compte n'a habituellement pas de structure discernable. Les noms de chemin de système de fichier UNIX et les URI sont des exemples de formes de nom de ressource structurées. Et un document XML est un exemple de ressource structurée.

Généralement, le nom d'un nœud (autre qu'un nœud folié) dans une forme de nom structurée est aussi une instance légale de la forme de nom. Ainsi, par exemple, le nom de la RFC 822 de l'IETF "med.example.com" est un nom légal de la RFC 822 de l'IETF qui identifie l'ensemble des adresses de messagerie électronique qui sont hébergées par le serveur med.example.com. Et la valeur de XPath/XPointer `//xacml-context:Request/xacml-context:Resource/xacml-context:ResourceContent/md:record/md:patient/` est une valeur légale de XPath/XPointer qui identifie un ensemble de nœuds dans un document XML.

Une question se pose: comment le PDP devrait-il interpréter un nom qui identifie un ensemble de sujets ou ressources, s'il apparaît dans un contexte de politique ou de demande? Est-il destiné à représenter seulement le nœud explicitement identifié par le nom, ou est-il destiné à représenter tout le sous-arbre subordonné à ce nœud?

Dans le cas des sujets, il n'y a pas d'entité réelle qui corresponde à un tel nœud. Aussi, les noms de ce type se réfèrent toujours à l'ensemble des sujets subordonnés au nœud identifié dans la structure de nom. Par conséquent, les noms de sujet non foliés ne devraient pas être utilisés dans des fonctions d'égalité, et seulement dans des fonctions de correspondance, telles que `"urn:oasis:names:tc:xacml:1.0:function:rfc822Name-match"`, et non dans `"urn:oasis:names:tc:xacml:1.0:function:rfc822Name-equal"`.

7.3.1.1.2 Effet

L'effet de la règle indique les conséquences prévues par l'auteur de la règle d'une évaluation de "vrai" pour la règle. Deux valeurs sont admises: "Permit" *(permis)* et "Deny" *(refusé)*.

7.3.1.1.3 Condition

Condition représente une expression booléenne qui précise l'applicabilité de la règle au-delà des prédicats impliqués par sa cible. Donc, elle peut être absente.

7.3.1.2 Politique

D'après le modèle de flux de données on peut voir que les règles ne sont pas échangées parmi les entités système. Donc, un PAP combine les règles dans une politique. Une politique comprend quatre composants principaux:

- une cible;
- un identifiant d'algorithme de combinaison de règle;
- un ensemble de règles;
- des obligations.

7.3.1.2.1 Politique cible

Un élément XACML <PolicySet>, <Policy> ou <Rule> contient un élément <Target> qui spécifie l'ensemble des sujets, ressources, actions et environnements auxquels il s'applique. La <Target> d'un <PolicySet> ou <Policy> peut être déclarée par l'auteur du <PolicySet> ou <Policy>, ou elle peut être calculée à partir de l'élément <Target> des éléments <PolicySet>, <Policy> et <Rule> qu'elle contient.

Une entité système qui calcule une <Target> de cette façon n'est pas définie par XACML, mais il y a deux méthodes logiques qui peuvent être utilisées. Dans une méthode, l'élément <Target> du <PolicySet> ou <Policy> extérieur (le "composant extérieur") est calculé comme l'union de tous les éléments <Target> des éléments <PolicySet>, <Policy> ou <Rule> référencés (les "composants internes"). Dans une autre méthode, l'élément <Target> du composant externe est calculé comme l'intersection de tous les éléments <Target> des composants internes. Le résultat de l'évaluation dans chaque cas sera très différent: dans le premier cas, l'élément <Target> du composant externe le rend applicable à toute demande de décision qui correspond à l'élément <Target> d'au moins un composant interne; dans le second cas, l'élément <Target> du composant externe ne le rend applicable qu'aux demandes de décision qui correspondent aux éléments <Target> de chaque composant interne. Noter que le calcul de l'intersection d'un ensemble d'éléments <Target> n'est vraisemblablement praticable que si le modèle de données cible est relativement simple.

Dans les cas où la <Target> d'un <Policy> est déclarée par l'auteur de la politique, tous les éléments <Rule> composants du <Policy> qui ont le même élément <Target> que l'élément <Policy> peuvent omettre l'élément <Target>. De tels éléments <Rule> héritent de la <Target> de la <Policy> dans laquelle ils sont contenus.

7.3.1.2.2 Algorithme de combinaison de règles

L'algorithme de combinaison de règles spécifie la procédure par laquelle les résultats de l'évaluation des règles composantes sont combinés lors de l'évaluation de la politique, c'est-à-dire que la valeur de décision placée dans le contexte de réponse par le PDP est la valeur de la politique, comme défini par l'algorithme de combinaison de règles. Une politique peut avoir des paramètres de combinaison qui affectent le fonctionnement de l'algorithme de combinaison de règles.

7.3.1.2.3 Obligations

Des obligations peuvent être ajoutées par l'auteur de la politique.

Lorsqu'un PDP évalue une politique contenant des obligations, il retourne certaines de ces obligations au PEP dans le contexte de réponse.

7.3.1.3 Ensemble de politique

Un ensemble de politique comprend quatre composants principaux:

- une cible;
- un identifiant d'algorithme de combinaison de politiques;
- un ensemble de politiques;
- des obligations.

7.3.1.3.1 Algorithme de combinaison de politiques

L'algorithme de combinaison de politiques spécifie la procédure par laquelle les résultats de l'évaluation des politiques composantes sont combinés lors de l'évaluation de l'ensemble de politiques, c'est-à-dire que la valeur de Decision placée dans le contexte de réponse par le PDP est le résultat de l'évaluation de l'ensemble de politiques, comme défini par l'algorithme de combinaison de politiques. Un ensemble de politiques peut avoir des paramètres de combinaison qui affectent le fonctionnement de l'algorithme de combinaison de politiques.

7.3.1.3.2 Obligations

L'auteur d'un ensemble de politiques peut ajouter des obligations à l'ensemble de politiques, en plus de celles qui sont contenues dans les politiques composantes et des ensembles de politiques.

Lorsqu'un PDP évalue un ensemble de politiques qui contient des obligations, il retourne certaines de ces obligations au PEP dans son contexte de réponse.

7.4 Syntaxe de politique

Les fragments de schéma dans les paragraphes suivants ne sont pas normatifs.

7.4.1 Elément <PolicySet>

L'élément <PolicySet> est un élément de niveau supérieur dans le schéma de politique XACML. <PolicySet> est une agrégation d'autres ensembles de politique et de politiques. Les ensembles de politique peuvent être inclus dans un élément <PolicySet> englobant soit en utilisant directement l'élément <PolicySet>, soit en utilisant indirectement l'élément <PolicySetIdReference>. Les politiques peuvent être incluses dans un élément <PolicySet> englobant soit en utilisant directement l'élément <Policy>, soit en utilisant indirectement l'élément <PolicyIdReference>.

Un élément <PolicySet> peut être évalué, auquel cas on doit utiliser la procédure d'évaluation définie dans la présente Recommandation.

Si un élément <PolicySet> contient des références à d'autres ensembles de politiques ou politiques sous la forme d'URL, ces références peuvent alors être résolubles.

Les ensembles de politiques et politiques inclus dans un élément <PolicySet> doivent être combinés en utilisant l'algorithme identifié par l'attribut PolicyCombiningAlgId. <PolicySet> est traité exactement comme un <Policy> dans tous les algorithmes de combinaison de politiques.

L'élément <Target> définit l'applicabilité de l'élément <PolicySet> à un ensemble de demandes de décision. Si l'élément <Target> au sein de l'élément <PolicySet> correspond au contexte de la demande, l'élément <PolicySet> peut alors être utilisé par le PDP lors de sa prise de décision d'autorisation.

L'élément <Obligations> contient un ensemble d'obligations qui doivent être satisfaites par le PEP en conjonction avec la décision d'autorisation. Si le PEP ne comprend, ou ne peut satisfaire, aucune des obligations, il doit alors agir comme si le PDP avait retourné une valeur de décision d'autorisation de "Deny".

```
<xs:element name="PolicySet" type="xacml:PolicySetType"/>
<xs:complexType name="PolicySetType">
   <xs:sequence>
         <xs:element ref="xacml:Description" minOccurs="0"/>
         <xs:element ref="xacml:PolicySetDefaults" minOccurs="0"/>
         <xs:element ref="xacml:Target"/>
         <xs:choice minOccurs="0" maxOccurs="unbounded">
               <xs:element ref="xacml:PolicySet"/>
               <xs:element ref="xacml:Policy"/>
               <xs:element ref="xacml:PolicySetIdReference"/>
               <xs:element ref="xacml:PolicyIdReference"/>
               <xs:element ref="xacml:CombinerParameters"/>
               <xs:element ref="xacml:PolicyCombinerParameters"/>
               <xs:element ref="xacml:PolicySetCombinerParameters"/>
         </xs:choice>
         <xs:element ref="xacml:Obligations" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="PolicySetId" type="xs:anyURI" use="required"/>
   <xs:attribute name="Version" type="xacml:VersionType" default="1.0"/>
   <xs:attribute name="PolicyCombiningAlgId" type="xs:anyURI" use="required"/>
</xs:complexType>
>
```

L'élément <PolicySet> est du type complexe **PolicySetType**.

L'élément <PolicySet> contient les attributs et éléments suivants:

– PolicySetId [Exigé]

  Identifiant d'ensemble de politique. Il appartient au PAP de s'assurer qu'il n'y a pas deux politiques visibles pour le PDP qui aient le même identifiant. Ceci peut être fait en suivant un schéma d'URN ou d'URI prédéfini. Si l'identifiant d'ensemble de politiques est sous la forme d'un URL, il peut alors être résoluble.

– Version [Défaut 1.0]

  Numéro de version du PolicySet.

– PolicyCombiningAlgId [Exigé]

  L'identifiant de l'algorithme de combinaison de politique par lequel les composants <PolicySet>, <CombinerParameters>, <PolicyCombinerParameters> et <PolicySetCombinerParameters> doivent être combinés.

– <Description> [Facultatif]

  Description de forme libre de l'ensemble de politique.

– <PolicySetDefaults> [Facultatif]

  Ensemble des valeurs par défaut applicables à l'ensemble de politique. La portée de l'élément <PolicySetDefaults> doit être l'ensemble de politiques englobant.

– <Target> [Exigé]

  L'élément <Target> définit l'applicabilité d'un ensemble de politiques à un ensemble de demandes de décision.

  L'élément <Target> peut être déclaré par le créateur du <PolicySet> ou il peut être calculé à partir des éléments <Target> des éléments <Policy> référencés, soit comme intersection, soit comme union.

– <PolicySet> [Tout Nombre]

  Un ensemble de politiques qui est inclus dans cet ensemble de politiques.

– <Policy> [Tout Nombre]

Une politique qui est incluse dans cet ensemble de politiques.

– <PolicySetIdReference> [Tout Nombre]

Une référence à un ensemble de politiques qui doit être incluse dans cet ensemble de politiques. Si <PolicySetIdReference> est un URL, il peut alors être résoluble.

– <PolicyIdReference> [Tout Nombre]

Une référence à une politique qui doit être incluse dans cet ensemble de politiques. Si le <PolicyIdReference> est un URL, il peut alors être résoluble.

– <Obligations> [Facultatif]

Contient l'ensemble des éléments <Obligation>.

– <CombinerParameters> [Facultatif]

Contient une séquence d'éléments <CombinerParameter>.

– <PolicyCombinerParameters> [Facultatif]

Contient une séquence d'éléments <CombinerParameter> qui sont associés à un élément <Policy> ou <PolicyIdReference> particulier au sein du <PolicySet>.

– <PolicySetCombinerParameters> [Facultatif]

Contient une séquence d'éléments <CombinerParameter> qui sont associés à un élément <PolicySet> ou <PolicySetIdReference> particulier au sein du <PolicySet>.

7.4.2 Elément <Description>

L'élément <Description> contient une description de forme libre de l'élément <PolicySet>, <Policy> ou <Rule>. L'élément <Description> est du type simple **xs:string**.

```
    <xs:element name="Description" type="xs:string"/>
```

7.4.3 Elément <PolicySetDefaults>

L'élément <PolicySetDefaults> doit spécifier les valeurs par défaut qui s'appliquent à l'élément <PolicySet>.

```
<xs:element name="PolicySetDefaults" type="xacml:DefaultsType"/>
<xs:complexType name="DefaultsType">
   <xs:sequence>
          <xs:choice>
                <xs:element ref="xacml:XPathVersion" minOccurs="0"/>
          </xs:choice>
   </xs:sequence>
</xs:complexType>
```

L'élément <PolicySetDefaults> est du type complexe **DefaultsType**.

L'élément <PolicySetDefaults> contient les éléments suivants:

– <XPathVersion> [Facultatif]

Version XPath par défaut.

7.4.4 Elément <XPathVersion>

L'élément <XPathVersion> doit spécifier la version de XPath:1999 du W3C que les éléments <AttributeSelector> et les fonctions fondées sur XPath doivent utiliser dans l'ensemble de politiques ou la politique.

```
<xs:element name="XPathVersion" type="xs:anyURI"/>
```

L'URI pour XPath:1999 du W3C est "http://www.w3.org/TR/1999/Rec-xpath-19991116". L'élément <XPathVersion> est exigé si l'ensemble de politiques ou la politique XACML englobante contient des éléments <AttributeSelector> ou des fonctions fondées sur XPath.

7.4.5 Elément <Target>

L'élément <Target> identifie l'ensemble de demandes de décision que l'élément parent va évaluer. L'élément <Target> doit apparaître comme un enfant des éléments <PolicySet> et <Policy> et peut apparaître comme un enfant d'un élément <Rule>. Il contient les définitions pour les sujets, les ressources, les actions et les environnements.

L'élément <Target> doit contenir une séquence conjonctive d'éléments <Subjects>, <Resources> <Actions> et <Environments>. Pour que le parent de l'élément <Target> soit applicable à la demande de décision, il doit y avoir au moins une correspondance positive entre chaque section de l'élément <Target> et la section correspondante de l'élément <xacml-context:Request>.

```
<xs:element name="Target" type="xacml:TargetType"/>
<xs:complexType name="TargetType">
   <xs:sequence>
          <xs:element ref="xacml:Subjects" minOccurs="0"/>
          <xs:element ref="xacml:Resources" minOccurs="0"/>
          <xs:element ref="xacml:Actions" minOccurs="0"/>
          <xs:element ref="xacml:Environments" minOccurs="0"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Target> est du type complexe **TargetType**.

L'élément <Target> contient les éléments suivants:

– <Subjects> [Facultatif]

  Spécification de correspondance pour les attributs de sujet dans le contexte. Si cet élément manque, la cible doit alors correspondre à tous les sujets.

– <Resources> [Facultatif]

  Spécification de correspondance pour les attributs de ressource dans le contexte. Si cet élément manque, la cible doit alors correspondre à toutes les ressources.

– <Actions> [Facultatif]

  Spécification de correspondance pour les attributs d'action dans le contexte. Si cet élément manque, la cible doit alors correspondre à toutes les actions.

– <Environments> [Facultatif]

  Spécification de correspondance pour les attributs d'environnement dans le contexte. Si cet élément manque, la cible doit alors correspondre à tous les environnements.

7.4.6 Elément <Subjects>

L'élément <Subjects> doit contenir une séquence disjonctive des éléments <Subject>.

```
<xs:element name="Subjects" type="xacml:SubjectsType"/>
<xs:complexType name="SubjectsType">
   <xs:sequence>
          <xs:element ref="xacml:Subject" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Subjects> est du type complexe **SubjectsType**.

L'élément <Subjects> contient les éléments suivants:

– <Subject> [de un à plusieurs, Exigé]

  Comme défini au § 7.4.7.

7.4.7 Elément <Subject>

L'élément <Subject> doit contenir une séquence conjonctive d'éléments <SubjectMatch>.

```
<xs:element name="Subject" type="xacml:SubjectType"/>
<xs:complexType name="SubjectType">
   <xs:sequence>
          <xs:element ref="xacml:SubjectMatch" maxOccurs="unbounded"/>
```

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

L'élément <Subject> est du type complexe **SubjectType**.

L'élément <Subject> contient les éléments suivants:

– <SubjectMatch> [de un à plusieurs]

Séquence conjonctive de correspondances individuelles des attributs de sujet dans le contexte de la demande et des valeurs d'attribut enchâssées.

7.4.8 Elément <SubjectMatch>

L'élément <SubjectMatch> doit identifier un ensemble d'entités se rapportant au sujet en faisant correspondre les valeurs d'attribut dans un élément <xacml-context:Subject> du contexte de la demande avec la valeur d'attribut enchâssée.

```
<xs:element name="SubjectMatch" type="xacml:SubjectMatchType"/>
<xs:complexType name="SubjectMatchType">
   <xs:sequence>
         <xs:element ref="xacml:AttributeValue"/>
         <xs:choice>
               <xs:element ref="xacml:SubjectAttributeDesignator"/>
               <xs:element ref="xacml:AttributeSelector"/>
         </xs:choice>
   </xs:sequence>
   <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
</xs:complexType>
```

L'élément <SubjectMatch> est du type complexe **SubjectMatchType**.

L'élément <SubjectMatch> contient les attributs et éléments suivants:

– MatchId [Exigé]

Spécifie une fonction de correspondance. La valeur de cet attribut doit être du type **xs:anyURI** avec des valeurs légales comme exposé au § 7.6.5.

– <xacml:AttributeValue> [Exigé]

Valeur d'attribut enchâssée.

– <SubjectAttributeDesignator> [Choix exigé]

Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans un élément <Subject> du contexte de la demande.

– <AttributeSelector> [Choix exigé]

Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans le contexte de la demande. L'expression XPath devrait se résoudre en un attribut dans un élément <Subject> du contexte de la demande.

7.4.9 Elément <Resources>

L'élément <Resources> doit contenir une séquence disjonctive d'éléments <Resource>.

```
<xs:element name="Resources" type="xacml:ResourcesType"/>
<xs:complexType name="ResourcesType">
   <xs:sequence>
         <xs:element ref="xacml:Resource" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Resources> est du type complexe **ResourcesType**.

L'élément <Resources> contient les éléments suivants:

– <Resource> [de un à plusieurs, Exigé]

Voir au paragraphe 7.4.10.

7.4.10 Elément <Resource>

L'élément <Resource> doit contenir une séquence conjonctive d'éléments <ResourceMatch>.

```
<xs:element name="Resource" type="xacml:ResourceType"/>
<xs:complexType name="ResourceType">
   <xs:sequence>
          <xs:element ref="xacml:ResourceMatch" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Resource> est du type complexe **ResourceType**.

L'élément <Resource> contient les éléments suivants:

– <ResourceMatch> [de un à plusieurs]

  Une séquence conjonctive de correspondances individuelles des attributs de ressource dans le contexte de la demande et des valeurs d'attribut enchâssées.

7.4.11 Elément <ResourceMatch>

L'élément <ResourceMatch> doit identifier un ensemble d'entités se rapportant aux ressources en faisant correspondre les valeurs d'attribut dans l'élément <xacml-context:Resource> du contexte de la demande avec la valeur d'attribut enchâssée.

```
<xs:element name="ResourceMatch" type="xacml:ResourceMatchType"/>
<xs:complexType name="ResourceMatchType">
   <xs:sequence>
          <xs:element ref="xacml:AttributeValue"/>
          <xs:choice>
                 <xs:element ref="xacml:ResourceAttributeDesignator"/>
                 <xs:element ref="xacml:AttributeSelector"/>
          </xs:choice>
   </xs:sequence>
   <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
</xs:complexType>
```

L'élément <ResourceMatch> est du type complexe **ResourceMatchType**.

L'élément <ResourceMatch> contient les attributs et éléments suivants:

– MatchId [Exigé]

  Spécifie une fonction de correspondance. Les valeurs de cet attribut doivent être du type **xs:anyURI**, avec des valeurs légales comme exposé au § 7.6.5.

– <xacml:AttributeValue> [Exigé]

  Valeur d'attribut enchâssée.

– <ResourceAttributeDesignator> [Choix exigé]

  Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans l'élément <Resource> du contexte de la demande.

– <AttributeSelector> [Choix exigé]

  Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans le contexte de la demande. L'expression XPath devrait se résoudre en un attribut dans l'élément <Resource> du contexte de la demande.

7.4.12 Elément <Actions>

L'élément <Actions> doit contenir une séquence disjonctive d'éléments <Action>.

```
<xs:element name="Actions" type="xacml:ActionsType"/>
<xs:complexType name="ActionsType">
   <xs:sequence>
          <xs:element ref="xacml:Action" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Actions> est du type complexe **ActionsType**.

L'élément <Actions> contient les éléments suivants:

– <Action> [de une à plusieurs, Exigé]

Voir au § 7.4.13.

7.4.13 Elément <Action>

L'élément <Action> doit contenir une séquence conjonctive d'éléments <ActionMatch>.

```
<xs:element name="Action" type="xacml:ActionType"/>
<xs:complexType name="ActionType">
   <xs:sequence>
         <xs:element ref="xacml:ActionMatch" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Action> est du type complexe **ActionType**.

L'élément <Action> contient les éléments suivants:

– <ActionMatch> [de une à plusieurs]

Une séquence conjonctive de correspondances individuelles des attributs d'action dans le contexte de la demande et des valeurs d'attribut enchâssées, voir au § 7.4.14.

7.4.14 Elément <ActionMatch>

L'élément <ActionMatch> doit identifier un ensemble d'entités se rapportant à l'action en confrontant les valeurs d'attribut dans l'élément <xacml-context:Action> du contexte de la demande avec la valeur d'attribut enchâssée.

```
<xs:element name="ActionMatch" type="xacml:ActionMatchType"/>
<xs:complexType name="ActionMatchType">
   <xs:sequence>
         <xs:element ref="xacml:AttributeValue"/>
         <xs:choice>
               <xs:element ref="xacml:ActionAttributeDesignator"/>
               <xs:element ref="xacml:AttributeSelector"/>
         </xs:choice>
   </xs:sequence>
   <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
</xs:complexType>
```

L'élément <ActionMatch> est du type complexe **ActionMatchType**.

L'élément <ActionMatch> contient les attributs et éléments suivants:

– MatchId [Exigé]

Spécifie une fonction de correspondance. La valeur de cet attribut doit être du type **xs:anyURI**, avec des valeurs légales comme exposé au § 7.6.5.

– <xacml:AttributeValue> [Exigé]

Valeur d'attribut enchâssée.

– <ActionAttributeDesignator> [Choix exigé]

Peut être utilisé pour identifier une ou plusieurs valeurs d'*attribut* dans l'élément <Action> du *contexte* de la demande.

– <AttributeSelector> [Choix exigé]

Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans le contexte de la demande. L'expression XPath devrait se résoudre en un attribut dans l'élément <Action> du contexte.

7.4.15 Elément <Environnements>

L'élément <Environments> doit contenir une séquence disjonctive d'éléments <Environment>.

```
<xs:element name="Environments" type="xacml:EnvironmentsType"/>
<xs:complexType name="EnvironmentsType">
   <xs:sequence>
          <xs:element ref="xacml:Environment" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Environments> est du type complexe **EnvironmentsType**.

L'élément <Environments> contient les éléments suivants:

– <Environment> [de un à plusieurs, Exigé]

Voir au § 7.4.16.

7.4.16 Elément <Environment>

L'élément <Environment> doit contenir une séquence conjonctive d'éléments <EnvironmentMatch>.

```
<xs:element name="Environment" type="xacml:EnvironmentType"/>
<xs:complexType name="EnvironmentType">
   <xs:sequence>
          <xs:element ref="xacml:EnvironmentMatch" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Environment> est du type complexe **EnvironmentType**.

L'élément <Environment> contient les éléments suivants:

– <EnvironmentMatch> [de un à plusieurs]

Une séquence conjonctive de correspondances individuelles des attributs d'environnement dans le contexte de la demande et des valeurs d'attribut enchâssées.

7.4.17 Elément <EnvironmentMatch>

L'élément <EnvironmentMatch> doit identifier un environnement en confrontant les valeurs d'attribut dans l'élément <xacml-context:Environment> du contexte de la demande avec la valeur d'attribut enchâssée.

```
<xs:element name="EnvironmentMatch" type="xacml:EnvironmentMatchType"/>
<xs:complexType name="EnvironmentMatchType">
   <xs:sequence>
          <xs:element ref="xacml:AttributeValue"/>
          <xs:choice>
                 <xs:element ref="xacml:EnvironmentAttributeDesignator"/>
                 <xs:element ref="xacml:AttributeSelector"/>
          </xs:choice>
   </xs:sequence>
   <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
</xs:complexType>
```

L'élément <EnvironmentMatch> est du type complexe **EnvironmentMatchType**.

L'élément <EnvironmentMatch> contient les attributs et éléments suivants:

– MatchId [Exigé]

Spécifie une fonction de confrontation. La valeur de cet attribut doit être du type **xs:anyURI**, avec les valeurs légales exposées au § 7.6.5.

– <xacml:AttributeValue> [Exigé]

Valeur d'attribut enchâssée.

– <EnvironmentAttributeDesignator> [Choix exigé]

Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans l'élément <Environment> du contexte de la demande.

– `<AttributeSelector>` [Choix exigé]

  Peut être utilisé pour identifier une ou plusieurs valeurs d'attribut dans le contexte de la demande. L'expression XPath devrait se résoudre en un attribut dans l'élément <Environment> du contexte de la demande.

7.4.18 Elément <PolicySetIdReference>

L'élément `<PolicySetIdReference>` doit être utilisé pour référencer un élément `<PolicySet>` par un identifiant. Si `<PolicySetIdReference>` est un URL, il peut alors être résoluble en élément `<PolicySet>`. Cependant, le mécanisme de résolution d'une référence d'un ensemble de politiques en ensemble de politiques correspondant sort du domaine d'application de la présente Recommandation.

```
<xs:element name="PolicySetIdReference" type="xacml:IdReferenceType"/>
xs:complexType name="IdReferenceType">
   <xs:simpleContent>
          <xs:extension base="xs:anyURI">
                <xs:attribute name="xacml:Version" type="xacml:VersionMatchType"
                use="optional"/>
                <xs:attribute name="xacml:EarliestVersion"
type="xacml:VersionMatchType" use="optional"/>
                <xs:attribute name="xacml:LatestVersion"
type="xacml:VersionMatchType" use="optional"/>
          </xs:extension>
   </xs:simpleContent>
</xs:complexType>
```

L'élément `<PolicySetIdReference>` est du type complexe **xacml:IdReferenceType**.

IdReferenceType étend le type **xs:anyURI** avec les attributs suivants:

– `Version` [Facultatif]

  Spécifie une expression correspondante pour la version de l'ensemble de politiques référencé.

– `EarliestVersion` [Facultatif]

  Spécifie une expression correspondante pour la version acceptable la plus ancienne de l'ensemble de politiques référencé.

– `LatestVersion` [Facultatif]

  Spécifie une expression correspondante pour la version acceptable la plus récente de l'ensemble de politiques référencé.

Toute combinaison de ces attributs peut être présente dans un `<PolicySetIdReference>`. L'ensemble de politiques référencé doit correspondre pour toutes les expressions. Si aucun de ces attributs n'est présent, toute version de l'ensemble de politiques est alors acceptable. Dans le cas où on peut obtenir plus d'une version qui correspond, on devrait alors utiliser la plus récente.

7.4.19 Elément <PolicyIdReference>

L'élément `<xacml:PolicyIdReference>` doit être utilisé pour se référer à un élément `<Policy>` par l'identifiant. Si `<PolicyIdReference>` est un URL, il peut alors être résoluble à l'élément `<Policy>`. Cependant, le mécanisme de résolution d'une référence de politique en la politique correspondante sort du domaine d'application de la présente Recommandation.

```
<xs:element name="PolicyIdReference" type="xacml:IdReferenceType"/>
```

L'élément `<PolicyIdReference>` est du type complexe **xacml:IdReferenceType**.

7.4.20 Type simple VersionType

Les éléments de ce type doivent contenir le numéro de version de la politique ou de l'ensemble de politiques.

```
<xs:simpleType name="VersionType">
   <xs:restriction base="xs:string">
          <xs:pattern value="(\d+\.)*\d+"/>
   </xs:restriction>
</xs:simpleType>
```

Le numéro de version est exprimé par une séquence de chiffres décimaux, chacun séparé par un point (.). 'd+' représente une séquence de un ou plusieurs chiffres décimaux.

7.4.21 Type simple VersionMatchType

Les éléments de ce type doivent contenir une expression régulière restreinte correspondant à un numéro de version. L'expression doit correspondre aux versions d'une politique ou ensemble de politiques référencé dont l'inclusion soit acceptable dans la politique ou ensemble de politiques de référence.

```
<xs:simpleType name="VersionMatchType">
   <xs:restriction base="xs:string">
          <xs:pattern value="((\d+|\*)\.)*(\d+|\*|\+)"/>
   </xs:restriction>
</xs:simpleType>
```

Une confrontation de version est séparée par des '.', comme une chaîne de version. Un numéro représente une correspondance numérique directe. Un '*' signifie que tout chiffre seul est valide. Un '+' signifie que tout chiffre, et tous les chiffres suivants, sont valides. De cette façon, les quatre schémas suivants '1.2.3': '1.2.3', '1.*.3', '1.2.*' et '1.+' satisferaient tous à la chaîne de version.

7.4.22 Elément <Policy>

L'élément `<Policy>` est la plus petite entité qui doit être présentée au PDP pour l'évaluation.

Un élément `<Policy>` peut être évalué, auquel cas, la procédure d'évaluation définie au § 7.6.10 doit être utilisée.

Les principaux composants de cet élément sont les éléments `<Target>`, `<Rule>`, `<CombinerParameters>`, `<RuleCombinerParameters>` et `<Obligations>` et l'attribut `RuleCombiningAlgId`.

L'élément `<Target>` définit l'applicabilité de l'élément `<Policy>` à un ensemble de demandes de décision. Si l'élément `<Target>` au sein de l'élément `<Policy>` correspond au contexte de la demande, l'élément `<Policy>` peut être utilisé par le PDP pour sa prise de décision d'autorisation.

L'élément `<Policy>` inclut une séquence de choix entre les éléments `<VariableDefinition>` *(définition de variable)* et `<Rule>`.

Les règles incluses dans l'élément `<Policy>` doivent être combinées par l'algorithme spécifié par l'attribut `RuleCombiningAlgId`.

L'élément `<Obligations>` contient un ensemble d'obligations qui doivent être satisfaites par le PEP en conjonction avec la décision d'autorisation.

```
<xs:element name="Policy" type="xacml:PolicyType"/>
<xs:complexType name="PolicyType">
   <xs:sequence>
          <xs:element ref="xacml:Description" minOccurs="0"/>
          <xs:element ref="xacml:PolicyDefaults" minOccurs="0"/>
          <xs:element ref="xacml:CombinerParameters" minOccurs="0"/>
          <xs:element ref="xacml:Target"/>
          <xs:choice maxOccurs="unbounded">
                 <xs:element ref="xacml:CombinerParameters" minOccurs="0"/>
                 <xs:element ref="xacml:RuleCombinerParameters" minOccurs="0"/>
                 <xs:element ref="xacml:VariableDefinition"/>
                 <xs:element ref="xacml:Rule"/>
          </xs:choice>
          <xs:element ref="xacml:Obligations" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="PolicyId" type="xs:anyURI" use="required"/>
   <xs:attribute name="Version" type="xacml:VersionType" default="1.0"/>
   <xs:attribute name="RuleCombiningAlgId" type="xs:anyURI" use="required"/>
</xs:complexType>
```

L'élément `<Policy>` est du type complexe **PolicyType**.

L'élément `<Policy>` contient les attributs et éléments suivants:

– `PolicyId` [Exigé]

  Identifiant de politique. Il est de la responsabilité du PAP de s'assurer qu'il n'y a pas deux politiques visibles du PDP qui aient le même identifiant. Cela peut être réalisé en suivant un schéma d'URN ou d'URI prédéfini. Si l'identifiant de politique est sous la forme d'un URL, il peut alors être résoluble.

– `Version` [Par défaut 1.0]

  Le numéro de version de la politique.

– `RuleCombiningAlgId` [Exigé]

  L'identifiant de l'algorithme de combinaison de règles par lequel les composants `<Policy>`, `<CombinerParameters>` *(paramètres de combinaison)* et `<RuleCombinerParameters>` doivent être combinés.

– `<Description>` [Facultatif]

  Description de forme libre de la politique.

– `<PolicyDefaults>` [Facultatif]

  Définit un ensemble de valeurs par défaut applicable à la politique. La portée de l'élément `<PolicyDefaults>` doit être la politique qui l'englobe.

– `<CombinerParameters>` [Facultatif]

  Une séquence de paramètres à utiliser par l'algorithme de combinaison de règles.

– `<RuleCombinerParameters>` [Facultatif]

  Une séquence de paramètres à utiliser par l'algorithme de combinaison de règles.

– `<Target>` [Exigé]

  L'élément `<Target>` définit l'applicabilité d'une `<Policy>` à un ensemble de demandes de décision.

  L'élément `<Target>` peut être déclaré par le créateur de l'élément `<Policy>`, ou il peut être calculé à partir des éléments `<Target>` des éléments `<Rule>` référencés soit comme intersection soit comme union.

– `<VariableDefinition>` [Tout Nombre]

  Définitions de fonctions communes qui peuvent être référencées à partir de tout point dans une règle où on peut trouver une expression.

– `<Rule>` [Tout Nombre]

  Une séquence de règles qui doivent être combinées conformément à l'attribut `RuleCombiningAlgId`. Les règles dont les éléments `<Target>` correspondent à la demande de décision doivent être prises en considération. Les règles dont les éléments `<Target>` ne correspondent pas à la demande de décision doivent être ignorées.

– `<Obligations>` [Facultatif]

  Une séquence conjonctive d'obligations qui doivent être satisfaites par le PEP en conjonction avec la décision d'autorisation.

7.4.23 Elément <PolicyDefaults>

L'élément `<PolicyDefaults>` doit spécifier les valeurs par défaut qui s'appliquent à l'élément `<Policy>`.

```
<xs:element name="PolicyDefaults" type="xacml:DefaultsType"/>
<xs:complexType name="DefaultsType">
   <xs:sequence>
          <xs:choice>
                 <xs:element ref="xacml:XPathVersion" minOccurs="0"/>
          </xs:choice>
   </xs:sequence>
</xs:complexType>
```

L'élément `<PolicyDefaults>` est du type complexe **DefaultsType**.

L'élément `<PolicyDefaults>` contient les éléments suivants:

– `<XPathVersion>` [Facultatif]

  Version XPath par défaut.

7.4.24 Elément <CombinerParameters>

L'élément `<CombinerParameters>` porte les paramètres pour un algorithme de combinaison de politique ou de règle.

Si plusieurs éléments <CombinerParameters> surviennent au sein de la même politique ou ensemble de politiques, ils doivent être considérés comme égaux à un élément <CombinerParameters> contenant l'enchaînement de toutes les séquences de <CombinerParameters> contenues dans tous les éléments <CombinerParameters> susmentionnés, de telle sorte que l'ordre d'occurrence des éléments <CominberParameters> soit préservé dans l'enchaînement des éléments <CombinerParameter>.

Noter qu'aucun des algorithmes de combinaison spécifiés dans XACML 2.0 n'est paramétré.

```
<xs:element name="CombinerParameters" type="xacml:CombinerParametersType"/>
<xs:complexType name="CombinerParametersType">
   <xs:sequence>
          <xs:element ref="xacml:CombinerParameter" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <CombinerParameters> est du type complexe **CombinerParametersType**.

L'élément <CombinerParameters> contient les éléments suivants:

– <CombinerParameter> [Tout Nombre]

  Paramètre unique.

La prise en charge de l'élément <CombinerParameters> est facultative.

7.4.25 Elément <CombinerParameter>

L'élément <CombinerParameter> porte un seul paramètre pour un algorithme de combinaison de politiques ou de règles.

```
<xs:element name="CombinerParameter" type="xacml:CombinerParameterType"/>
<xs:complexType name="CombinerParameterType">
   <xs:sequence>
          <xs:element ref="xacml:AttributeValue"/>
   </xs:sequence>
   <xs:attribute name="ParameterName" type="xs:string" use="required"/>
</xs:complexType>
```

L'élément <CombinerParameter> est du type complexe **CombinerParameterType**.

L'élément <CombinerParameter> contient l'attribut suivant:

– ParameterName [Exigé]

  L'identifiant du paramètre.

– AttributeValue [Exigé]

  La valeur du paramètre.

La prise en charge de l'élément <CombinerParameter> est facultative.

7.4.26 Elément <RuleCombinerParameters>

L'élément <RuleCombinerParameters> porte les paramètres associés à une règle particulière au sein d'une politique pour un algorithme de combinaison de règles.

Chaque élément <RuleCombinerParameters> doit être associé à une règle contenue au sein de la même politique. Si plusieurs éléments <RuleCombinerParameters> font référence à la même règle, ils doivent être considérés comme égaux à un élément <RuleCombinerParameters> contenant l'enchaînement de toutes les séquences de <CombinerParameters> contenues dans tous les éléments <RuleCombinerParameters> susmentionnés, de telle sorte que l'ordre d'occurrence des éléments <RuleCominberParameters> soit préservé dans l'enchaînement des éléments <CombinerParameter>.

Noter qu'aucun des algorithmes de combinaison de règles spécifiés dans XACML 2.0 n'est paramétré.

```
<xs:element name="RuleCombinerParameters"
type="xacml:RuleCombinerParametersType"/>
<xs:complexType name="RuleCombinerParametersType">
   <xs:complexContent>
          <xs:extension base="xacml:CombinerParametersType">
```

```
                <xs:attribute name="RuleIdRef" type="xs:string" use="required"/>
          </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <RuleCombinerParameters> contient les éléments suivants:

– RuleIdRef [Exigé]

  L'identifiant de la <Rule> contenue dans la politique.

La prise en charge de l'élément <RuleCombinerParameters> n'est facultative que si la prise en charge des paramètres de combineur n'est pas implémentée.

7.4.27 Elément <PolicyCombinerParameters>

L'élément <PolicyCombinerParameters> porte les paramètres associés à une politique particulière au sein d'un ensemble de politiques pour un algorithme de combinaison de politiques.

Chaque élément <PolicyCombinerParameters> doit être associé à une politique contenue dans le même ensemble de politiques. Si plusieurs éléments <PolicyCombinerParameters> font référence à la même politique, ils doivent être considérés comme égaux à un élément <PolicyCombinerParameters> contenant l'enchaînement de toutes les séquences de <CombinerParameters> contenues dans tous les éléments <PolicyCombinerParameters> susmentionnés, de telle sorte que l'ordre d'occurrence des éléments <PolicyCombinerParameters> soit préservé dans l'enchaînement des éléments <CombinerParameter>.

Noter qu'aucun des algorithmes de combinaison de politiques spécifiés dans XACML 2.0 n'est paramétré.

```
<xs:element name="PolicyCombinerParameters"
type="xacml:PolicyCombinerParametersType"/>
<xs:complexType name="PolicyCombinerParametersType">
   <xs:complexContent>
          <xs:extension base="xacml:CombinerParametersType">
                <xs:attribute name="PolicyIdRef" type="xs:anyURI"
use="required"/>
          </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <PolicyCombinerParameters> est du type complexe **PolicyCombinerParametersType**.

L'élément <PolicyCombinerParameters> contient les éléments suivants:

– PolicyIdRef [Exigé]

  L'identifiant d'une <Policy> ou la valeur d'une <PolicyIdReference> contenue dans l'ensemble de politiques.

La prise en charge de l'élément <PolicyCombinerParameters> n'est facultative que si la prise en charge des paramètres de combineur n'est pas implémentée.

7.4.28 Elément <PolicySetCombinerParameters>

L'élément <PolicySetCombinerParameters> porte les paramètres associés à un ensemble de politiques particulier au sein d'un ensemble de politiques pour un algorithme de combinaison de politiques.

Chaque élément <PolicySetCombinerParameters> doit être associé à un ensemble de politiques contenu dans le même ensemble de politiques. Si plusieurs éléments <PolicySetCombinerParameters> font référence au même ensemble de politiques, ils doivent être considérés comme égaux à un élément <PolicySetCombinerParameters> contenant l'enchaînement de toutes les séquences de <CombinerParameters> contenues dans tous les éléments <PolicySetCombinerParameters> susmentionnés, de telle sorte que l'ordre d'occurrence des éléments <PolicySetCominberParameters> soit préservé dans l'enchaînement des éléments <CombinerParameter>.

Noter qu'aucun des algorithmes de combinaison de politiques spécifiés dans XACML 2.0 n'est paramétré.

```
<xs:element name="PolicySetCombinerParameters"
type="xacml:PolicySetCombinerParametersType"/>
<xs:complexType name="PolicySetCombinerParametersType">
   <xs:complexContent>
          <xs:extension base="xacml:CombinerParametersType">
```

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

L'élément <PolicySetCombinerParameters> est du type complexe **PolicySetCombinerParametersType**.

L'élément <PolicySetCombinerParameters> contient les éléments suivants:

– PolicySetIdRef [Exigé]

  Identifiant d'un <PolicySet> ou valeur d'une <PolicySetIdReference> contenue dans l'ensemble de politiques.

La prise en charge de l'élément <PolicySetCombinerParameters> n'est facultative que si la prise en charge des paramètres de combineur n'est pas implémentée.

7.4.29 Elément <Rule>

L'élément <Rule> doit définir les règles individuelles dans la politique. Les principaux composants de cet élément sont les éléments <Target> et <Condition> et l'attribut Effect.

Un élément <Rule> peut être évalué, auquel cas on doit utiliser la procédure d'évaluation définie au § 7.6.9.

```
<xs:element name="Rule" type="xacml:RuleType"/>
<xs:complexType name="RuleType">
   <xs:sequence>
          <xs:element ref="xacml:Description" minOccurs="0"/>
          <xs:element ref="xacml:Target" minOccurs="0"/>
          <xs:element ref="xacml:Condition" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="RuleId" type="xs:string" use="required"/>
   <xs:attribute name="Effect" type="xacml:EffectType" use="required"/>
</xs:complexType>
```

L'élément <Rule> est du type complexe **RuleType**.

L'élément <Rule> contient les attributs et éléments suivants:

– RuleId [Exigé]

  Chaîne identifiant cette règle.

– Effect [Exigé]

  Effet de la règle. La valeur de cet attribut est "Permit" ou "Deny".

– <Description> [Facultatif]

  Description de forme libre de la règle.

– <Target> [Facultatif]

  Identifie l'ensemble des demandes de décision que l'élément <Rule> est destiné à évaluer. Si cet élément est omis, la cible pour <Rule> doit alors être définie par l'élément <Target> de l'élément <Policy> englobant. Voir des précisions au § 7.6.6.

– <Condition> [Facultatif]

  Prédicat qui doit être satisfait pour que sa valeur de Effect soit allouée à la règle.

7.4.30 Type simple EffectType

Le type simple **EffectType** définit les valeurs admises pour l'attribut Effect de l'élément <Rule> et pour l'attribut FulfillOn de l'élément <Obligation>.

```
<xs:simpleType name="EffectType">
   <xs:restriction base="xs:string">
          <xs:enumeration value="Permit"/>
          <xs:enumeration value="Deny"/>
   </xs:restriction>
</xs:simpleType>
```

7.4.31 Elément <VariableDefinition>

L'élément <VariableDefinition> doit être utilisé pour définir une valeur qui puisse être référencée par un élément <VariableReference>. Le nom fourni pour son attribut VariableId ne doit pas apparaître dans l'attribut VariableId de tout autre élément <VariableDefinition> au sein de la politique environnante. L'élément <VariableDefinition> peut contenir un élément <VariableReference> indéfini, mais s'il le fait, un élément <VariableDefinition> correspondant doit être défini plus tard dans la politique environnante. Des éléments <VariableDefinition> peuvent être groupés ou placés à proximité de la référence dans la politique environnante. Il peut y avoir zéro, une ou plusieurs références à chaque élément <VariableDefinition>.

```
<xs:element name="VariableDefinition" type="xacml:VariableDefinitionType"/>
<xs:complexType name="VariableDefinitionType">
   <xs:sequence>
         <xs:element ref="xacml:Expression"/>
   </xs:sequence>
   <xs:attribute name="VariableId" type="xs:string" use="required"/>
</xs:complexType>
```

L'élément <VariableDefinition> est du type complexe **VariableDefinitionType**. L'élément <VariableDefinition> a les éléments et attributs suivants:

– <Expression> [Exigé]

  Tout élément du type complexe **ExpressionType**.

– VariableId [Exigé]

  Nom de la définition de variable.

7.4.32 Elément <VariableReference>

L'élément <VariableReference> est utilisé pour faire référence à une valeur définie au sein du même élément <Policy> environnant. L'élément <VariableReference> doit se référer à l'élément <VariableDefinition> par une égalité de chaîne sur la valeur de leurs attributs VariableId respectifs. Il doit y avoir une seule <VariableDefinition> dans le même élément <Policy> environnant auquel se réfère <VariableReference>. Il doit y avoir zéro, un, ou plusieurs éléments <VariableReference> qui se réfèrent au même élément <VariableDefinition>.

```
<xs:element name="VariableReference" type="xacml:VariableReferenceType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="VariableReferenceType">
   <xs:complexContent>
         <xs:extension base="xacml:ExpressionType">
               <xs:attribute name="VariableId" type="xs:string" use="required"/>
         </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <VariableReference> est du type complexe **VariableReferenceType**, qui est du type complexe **ExpressionType** et est un membre du groupe de substitution de l'élément <Expression>. L'élément <VariableReference> peut apparaître à tout endroit où survient un élément <Expression> dans ce schéma.

L'élément <VariableReference> a les attributs suivants:

– VariableId [Exigé]

  Nom utilisé pour se référer à la valeur définie dans un élément <VariableDefinition>.

7.4.33 Elément <Expression>

L'élément <Expression> n'est pas utilisé directement dans une politique. L'élément <Expression> signifie qu'un élément qui étend **ExpressionType** et est membre du groupe de substitution de l'élément <Expression> doit apparaître à sa place.

```
<xs:element name="Expression" type="xacml:ExpressionType" abstract="true"/>
<xs:complexType name="ExpressionType" abstract="true"/>
```

Les éléments suivants sont dans le groupe de substitution de l'élément <Expression>:

<Apply>, <AttributeSelector>, <AttributeValue>, <Function>, <VariableReference>, <ActionAttributeDesignator>, <ResourceAttributeDesignator>, <SubjectAttributeDesignator> et <EnvironmentAttributeDesignator>.

7.4.34 Elément <Condition>

L'élément <Condition> est une fonction booléenne sur les attributs ou fonctions d'attributs de sujet, ressource, action et environnement.

```
<xs:element name="Condition" type="xacml:ConditionType"/>
<xs:complexType name="ConditionType">
   <xs:sequence>
          <xs:element ref="xacml:Expression"/>
   </xs:sequence>
</xs:complexType>
```

<Condition> contient un élément <Expression>, avec la restriction que le type de données <Expression> retourné doit être "http://www.w3.org/2001/XMLSchema#boolean".

7.4.35 Elément <Apply>

L'élément <Apply> note l'application d'une fonction à ses arguments, et donc le codage d'un appel de fonction. L'élément <Apply> peut s'appliquer à toute combinaison des membres du groupe de substitution de l'élément <Expression>.

```
<xs:element name="Apply" type="xacml:ApplyType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="ApplyType">
   <xs:complexContent>
          <xs:extension base="xacml:ExpressionType">
                 <xs:sequence>
                        <xs:element ref="xacml:Expression" minOccurs="0"
maxOccurs="unbounded"/>
                 </xs:sequence>
                 <xs:attribute name="FunctionId" type="xs:anyURI" use="required"/>
          </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <Apply> est du type complexe **ApplyType**.

L'élément <Apply> contient les attributs et éléments suivants:

– FunctionId [Exigé]

  Identifiant de la fonction à appliquer aux arguments. Les fonctions définies en XACML sont décrites à l'Annexe A.

– <Expression> [Facultatif]

  Arguments pour la fonction, qui peuvent inclure d'autres fonctions.

7.4.36 Elément <Function>

L'élément <Function> doit être utilisé pour nommer une fonction comme un argument pour la fonction définie par l'élément <Apply> parent. Dans le cas où l'élément <Apply> parent est une fonction sac d'ordre supérieur, la fonction nommée est appliquée à chaque élément du ou des sacs identifiés dans les autres arguments de l'élément parent.

```
<xs:element name="Function" type="xacml:FunctionType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="FunctionType">
   <xs:complexContent>
          <xs:extension base="xacml:ExpressionType">
                 <xs:attribute name="FunctionId" type="xs:anyURI" use="required"/>
          </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément Function est du type complexe **FunctionType**.

L'élément Function contient les attributs suivants:

– `FunctionId` [Exigé]

  Identifiant de la fonction.

7.4.37 Type complexe AttributeDesignatorType

Le type complexe **AttributeDesignatorType** est le type pour les éléments qui identifient les attributs par le nom. Il contient les informations nécessaires pour confronter les attributs dans le contexte de la demande.

Il contient aussi des informations pour le contrôle du comportement dans le cas où aucun attribut correspondant n'est présent dans le contexte.

Les éléments de ce type ne doivent pas altérer la sémantique de confrontation des attributs nommés, mais peuvent rétrécir l'espace de recherche.

```
<xs:complexType name="AttributeDesignatorType">
   <xs:complexContent>
         <xs:extension base="xacml:ExpressionType">
               <xs:attribute name="AttributeId" type="xs:anyURI"
use="required"/>
               <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
               <xs:attribute name="Issuer" type="xs:string" use="optional"/>
               <xs:attribute name="MustBePresent" type="xs:boolean"
use="optional" default="false"/>
         </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

Un attribut nommé doit correspondre à un attribut si les valeurs de leurs attributs `AttributeId`, `DataType` et de producteur respectifs correspondent. Le `AttributeId` du désignateur d'attribut doit correspondre, par égalité d'URI, à l'`AttributeId` de l'attribut. Le `DataType` du désignateur d'attribut doit correspondre, par égalité d'URI, au `DataType` du même attribut.

Si l'attribut de producteur est présent dans le désignateur d'attribut, il doit alors correspondre, en utilisant la fonction "`urn:oasis:names:tc:xacml:1.0:function:string-equal`", au producteur du même attribut. Si le producteur n'est pas présent dans le désignateur d'attribut, la correspondance de l'attribut à l'attribut nommé doit alors être gouvernée par les seuls attributs `AttributeId` et `DataType`.

Le `<AttributeDesignatorType>` contient les attributs suivants:

– `AttributeId` [Exigé]

  Cet attribut doit spécifier le `AttributeId` avec lequel faire correspondre l'attribut.

– `DataType` [Exigé]

  Le sac retourné par l'élément `<AttributeDesignator>` doit contenir des valeurs de ce type de données.

– `Issuer` [Facultatif]

  Cet attribut, s'il est fourni, doit spécifier le `Issuer` (producteur) avec lequel faire correspondre l'attribut.

– `MustBePresent` [Facultatif]

  Cet attribut gouverne le retour par l'élément de "Indeterminate" ou d'un sac vide, dans le cas où l'attribut nommé est absent du contexte de la demande.

7.4.38 Elément <SubjectAttributeDesignator>

L'élément `<SubjectAttributeDesignator>` restitue un sac de valeurs pour un attribut de sujet catégorisé nommé d'après le contexte de la demande. Un attribut de sujet est un attribut contenu dans un élément `<Subject>` du contexte de la demande. Un sujet catégorisé est un sujet qui est identifié par un attribut de catégorie sujet particulier. Un attribut de sujet catégorisé nommé est un attribut de sujet nommé pour un sujet catégorisé particulier.

L'élément `<SubjectAttributeDesignator>` doit retourner un sac contenant toutes les valeurs d'attribut de sujet qui correspondent à l'attribut de sujet catégorisé nommé. Dans le cas où aucun attribut correspondant n'est présent dans le contexte, l'attribut `MustBePresent` commande si cet élément retourne un sac vide ou "Indeterminate".

Le **SubjectAttributeDesignatorType** étend la sémantique de correspondance du **AttributeDesignatorType** de telle sorte qu'il rétrécisse l'espace de recherche de l'attribut au sujet catégorisé spécifique de façon que la valeur de l'attribut `SubjectCategory` de cet élément corresponde, par égalité d'URI, à la valeur de l'attribut `SubjectCategory` du contexte de la demande `<Subject>`.

Si le contexte de la demande contient plusieurs sujets avec le même attribut XML `SubjectCategory`, ils doivent alors être traités comme s'ils étaient un seul sujet catégorisé.

L'élément `<SubjectAttributeDesignator>` peut apparaître dans l'élément `<SubjectMatch>` et peut être passé à l'élément `<Apply>` comme un argument.

```
<xs:element name="SubjectAttributeDesignator"
type="xacml:SubjectAttributeDesignatorType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="SubjectAttributeDesignatorType">
   <xs:complexContent>
         <xs:extension base="xacml:AttributeDesignatorType">
               <xs:attribute name="SubjectCategory" type="xs:anyURI"
use="optional" default="urn:oasis:names:tc:xacml:1.0:subject-category:access-
subject"/>
         </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément `<SubjectAttributeDesignator>` est du type **SubjectAttributeDesignatorType**. Le type complexe **SubjectAttributeDesignatorType** étend le type complexe **AttributeDesignatorType** avec un attribut `SubjectCategory`.

– `SubjectCategory` [Facultatif]

  Cet attribut doit spécifier le sujet catégorisé d'après lequel faire correspondre les attributs de sujet nommés. Si `SubjectCategory` n'est pas présent, on doit alors utiliser sa valeur par défaut de "`urn:oasis:names:tc:xacml:1.0:subject-category:access-subject`".

7.4.39 Elément <ResourceAttributeDesignator>

L'élément `<ResourceAttributeDesignator>` restitue un sac de valeurs pour un attribut de ressource nommé d'après le contexte de la demande. Un attribut de ressource est un attribut contenu dans l'élément `<Resource>` du contexte de la demande. Un attribut de ressource nommé est un attribut nommé qui correspond à un attribut de ressource. Un attribut de ressource nommé doit être considéré comme présent s'il y a au moins un attribut de ressource qui satisfait aux critères établis ci-dessous. Une valeur d'attribut de ressource est une valeur d'attribut qui est contenue dans un attribut de ressource.

L'élément `<ResourceAttributeDesignator>` doit retourner un sac contenant toutes les valeurs d'attribut de ressource qui sont satisfaites par l'attribut de ressource nommé. Dans le cas où aucun attribut correspondant n'est présent dans le contexte, l'attribut `MustBePresent` commande si cet élément retourne un sac vide ou "Indeterminate".

Un attribut de ressource nommé doit correspondre à un attribut de ressource selon la sémantique de correspondance spécifiée dans le type complexe **AttributeDesignatorType**.

L'élément `<ResourceAttributeDesignator>` peut apparaître dans l'élément `<ResourceMatch>` et peut être passé à l'élément `<Apply>` comme un argument.

```
<xs:element name="ResourceAttributeDesignator"
type="xacml:AttributeDesignatorType" substitutionGroup="xacml:Expression"/>
```

L'élément `<ResourceAttributeDesignator>` est du type complexe **AttributeDesignatorType**.

7.4.40 Elément <ActionAttributeDesignator>

L'élément `<ActionAttributeDesignator>` restitue un sac de valeurs pour un attribut d'action nommé d'après le contexte de la demande. Un attribut d'action est un attribut contenu dans l'élément `<Action>` du contexte de la demande. Un attribut d'action nommé a des critères spécifiques (décrits plus loin) selon lesquels il correspond à un attribut d'action. Un attribut d'action nommé doit être considéré comme présent s'il y a au moins un attribut d'action qui satisfait aux critères. Une valeur d'attribut d'action est une valeur d'attribut qui est contenue dans un attribut d'action.

L'élément `<ActionAttributeDesignator>` doit retourner un sac de toutes les valeurs d'attribut d'action qui sont satisfaites par l'attribut d'action nommé. Dans le cas où aucun attribut correspondant n'est présent dans le contexte, l'attribut `MustBePresent` commande si cet élément retourne un sac vide ou "Indeterminate".

Un attribut d'action nommé doit correspondre à un attribut d'action selon la sémantique de correspondance spécifiée dans le type complexe **AttributeDesignatorType**.

L'élément <ActionAttributeDesignator> peut apparaître dans l'élément <ActionMatch> et peut être passé à l'élément <Apply> comme un argument.

```
<xs:element name="ActionAttributeDesignator" type="xacml:AttributeDesignatorType"
substitutionGroup="xacml:Expression"/>
```

L'élément <ActionAttributeDesignator> est du type complexe **AttributeDesignatorType**.

7.4.41 Elément <EnvironmentAttributeDesignator>

L'élément <EnvironmentAttributeDesignator> restitue un sac de valeurs pour un attribut d'environnement nommé d'après le contexte de la demande. Un attribut d'environnement est un attribut contenu dans l'élément <Environment> du contexte de demande. Un attribut d'environnement nommé a des critères spécifiques (décrits ci-dessous) selon lesquels correspondre à un attribut d'environnement. Un attribut d'environnement nommé doit être considéré comme présent s'il y a au moins un attribut d'environnement qui satisfait à ces critères. Une valeur d'attribut d'environnement est une valeur d'attribut qui est contenue dans un attribut d'environnement.

L'élément <EnvironmentAttributeDesignator> doit s'évaluer comme un sac de toutes les valeurs d'attribut d'environnement qui sont satisfaites par l'attribut d'environnement nommé. Dans le cas où il n'y a pas d'attribut correspondant présent dans le contexte, l'attribut MustBePresent commande si cet élément retourne un sac vide ou "Indeterminate".

Un attribut d'environnement nommé doit correspondre à un attribut d'environnement selon la sémantique de correspondance spécifiée dans le type complexe **AttributeDesignatorType**.

L'élément <EnvironmentAttributeDesignator> peut être passé à l'élément <Apply> comme un argument.

```
<xs:element name="EnvironmentAttributeDesignator" type="xacml:AttributeDesignatorType"
substitutionGroup="xacml:Expression"/>
```

L'élément <EnvironmentAttributeDesignator> est du type complexe **AttributeDesignatorType**.

7.4.42 Elément <AttributeSelector>

L'élément <AttributeSelector> identifie les attributs par leur localisation dans le contexte de la demande. La prise en charge de l'élément <AttributeSelector> est facultative.

L'attribut XML RequestContextPath de l'élément <AttributeSelector> doit contenir une expression XPath légale dont le nœud de contexte est l'élément <xacml-context:Request>. L'élément AttributeSelector doit s'évaluer comme un sac de valeurs dont le type de données est spécifié par l'attribut DataType de l'élément. Si le DataType spécifié dans le AttributeSelector est un type de données de primitive (défini dans W3C Schema:2001, W3C Datatypes:2001, 3.2), la valeur lexicale retournée par l'expression XPath doit être convertie en une valeur de DataType spécifiée dans le <AttributeSelector>. Si une erreur résulte de la conversion de la valeur retournée par l'expression Xpath, comme lorsque la valeur n'est pas une instance valide du DataType, la valeur du <AttributeSelector> doit alors être "Indeterminate".

xs:string()

xs:boolean()

xs:integer()

xs:double()

xs:dateTime()

xs:date()

xs:time()

xs:hexBinary()

xs:base64Binary()

xs:anyURI()

Si le DataType spécifié dans le AttributeSelector n'est pas une des primitives DataType précédentes, le AttributeSelector doit alors retourner un sac d'instances du DataType spécifié. Si une erreur survient lors de la conversion des valeurs retournées par l'expression XPath au DataType spécifié, le résultat du AttributeSelector doit alors être "Indeterminate".

Chaque nœud choisi par l'expression XPath spécifiée doit être un nœud textuel, un nœud d'attribut, un nœud de traitement d'instruction ou un nœud de commentaire. La représentation de chaîne de la valeur de chaque nœud doit être

convertie en une valeur d'attribut du type de données spécifié, et le résultat de l'AttributeSelector est le sac de valeurs d'attribut généré à partir de tous les nœuds choisis.

Si le nœud choisi par l'expression XPath spécifiée n'est pas un de ceux dont la liste figure ci-dessus (c'est-à-dire, un nœud textuel, un nœud d'attribut, un nœud de traitement d'instruction ou un nœud de commentaire), le résultat de la politique englobante doit alors être "Indeterminate" avec une valeur de StatusCode de "urn:oasis:names:tc:xacml:1.0:status:syntax-error".

```
<xs:element name="AttributeSelector" type="xacml:AttributeSelectorType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="AttributeSelectorType">
   <xs:complexContent>
         <xs:extension base="xacml:ExpressionType">
               <xs:attribute name="RequestContextPath" type="xs:string"
use="required"/>
               <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
               <xs:attribute name="MustBePresent" type="xs:boolean"
use="optional" default="false"/>
         </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <AttributeSelector> est du type complexe **AttributeSelectorType**.

L'élément <AttributeSelector> a les attributs suivants:

– RequestContextPath [Exigé]

  Expression XPath dont le nœud de contexte est l'élément <xacml-context:Request>. Il ne doit pas y avoir de restriction sur la syntaxe de XPath.

– DataType [Exigé]

  Le sac retourné par l'élément <AttributeSelector> doit contenir des valeurs de ce type de données.

– MustBePresent [Facultatif]

  Cet attribut commande si l'élément retourne "Indeterminate" ou un sac vide dans le cas où l'expression XPath ne choisit pas de nœud.

7.4.43 Elément <AttributeValue>

L'élément <xacml:AttributeValue> doit contenir une valeur d'attribut littérale.

```
<xs:element name="AttributeValue" type="xacml:AttributeValueType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="AttributeValueType" mixed="true">
   <xs:complexContent>
         <xs:extension base="xacml:ExpressionType">
               <xs:sequence>
                     <xs:any namespace="##any" processContents="lax"
minOccurs="0" maxOccurs="unbounded"/>
               </xs:sequence>
               <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
               <xs:anyAttribute namespace="##any" processContents="lax"/>
         </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <xacml:AttributeValue> est du type complexe **AttributeValueType**.

L'élément <xacml:AttributeValue> a les attributs suivants:

– DataType [Exigé]

  Type de données de la valeur de l'attribut.

7.4.44 Elément <Obligations>

L'élément <Obligations> doit contenir un ensemble d'éléments <Obligation>.

La prise en charge de l'élément <Obligations> est facultative.

```
<xs:element name="Obligations" type="xacml:ObligationsType"/>
<xs:complexType name="ObligationsType">
   <xs:sequence>
          <xs:element ref="xacml:Obligation" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Obligations> est du type complexe **ObligationsType**.

L'élément <Obligations> contient l'élément suivant:

– <Obligation> [de un à plusieurs]

Une séquence d'obligations.

7.4.45 Elément <Obligation>

L'élément <Obligation> doit contenir un identifiant pour l'obligation et un ensemble d'attributs qui forment des arguments de l'action définie par l'obligation. L'attribut FulfillOn doit indiquer l'effet pour lequel cette obligation doit être satisfaite par le PEP.

```
<xs:element name="Obligation" type="xacml:ObligationType"/>
<xs:complexType name="ObligationType">
   <xs:sequence>
          <xs:element ref="xacml:AttributeAssignment" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="ObligationId" type="xs:anyURI" use="required"/>
   <xs:attribute name="FulfillOn" type="xacml:EffectType" use="required"/>
</xs:complexType>
```

L'élément <Obligation> est du type complexe **ObligationType**. Voir au § 7.6.14 la description de la façon dont l'ensemble d'obligations à retourner par le PDP est déterminé.

L'élément <Obligation> contient les éléments et attributs suivants:

– ObligationId [Exigé]

Identifiant d'obligation. La valeur de l'identifiant d'obligation doit être interprétée par le PEP.

– FulfillOn [Exigé]

Effet pour lequel cette obligation doit être satisfaite par le PEP.

– <AttributeAssignment> [Facultatif]

Allocation d'arguments d'obligation. Les valeurs des arguments d'obligation doivent être interprétées par le PEP.

7.4.46 Elément <AttributeAssignment>

L'élément <AttributeAssignment> est utilisé pour inclure des arguments dans les obligations. Il doit contenir un AttributeId et la valeur d'attribut correspondante, en étendant la définition de **AttributeValueType** type. L'élément <AttributeAssignment> peut être utilisé de toutes façons cohérentes avec la syntaxe de schéma, qui est une séquence d'éléments <xs:any>. La valeur spécifiée doit être comprise par le PEP, mais n'est pas spécifiée plus avant par XACML.

```
<xs:element name="AttributeAssignment" type="xacml:AttributeAssignmentType"/>
<xs:complexType name="AttributeAssignmentType" mixed="true">
   <xs:complexContent>
          <xs:extension base="xacml:AttributeValueType">
                 <xs:attribute name="AttributeId" type="xs:anyURI"
use="required"/>
          </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <AttributeAssignment> est du type complexe **AttributeAssignmentType**.

L'élément <AttributeAssignment> contient les attributs suivants:

– AttributeId [Exigé]

Identifiant de l'attribut.

7.5 Syntaxe de contexte

Les fragments de schéma dans les paragraphes qui suivent ne sont pas normatifs.

7.5.1 Elément <Request>

L'élément <Request> est un élément de niveau supérieur dans le schéma de contexte XACML. L'élément <Request> est une couche d'abstraction utilisée par le langage de politique. Pour simplifier l'expression, la présente Recommandation décrit l'évaluation de la politique en termes d'opérations sur le contexte. Cependant, un PDP conforme n'est pas obligé d'instancier réellement le contexte sous la forme d'un document XML. Mais, tout système conforme à XACML doit produire exactement les mêmes décisions d'autorisation que si toutes les entrées avaient été transformées en élément <xacml-context:Request>.

L'élément <Request> contient des éléments <Subject>, <Resource>, <Action> et <Environment>. Il peut y avoir plusieurs éléments <Subject> et, sous certaines conditions, plusieurs éléments <Resource>. Chaque élément fils contient une séquence d'éléments <xacml-context:Attribute> associée respectivement au sujet, ressource, action et environnent. Ces éléments <Attribute> peuvent faire partie de l'évaluation de politique.

```
<xs:element name="Request" type="xacml-context:RequestType"/>
<xs:complexType name="RequestType">
   <xs:sequence>
          <xs:element ref="xacml-context:Subject" maxOccurs="unbounded"/>
          <xs:element ref="xacml-context:Resource" maxOccurs="unbounded"/>
          <xs:element ref="xacml-context:Action"/>
          <xs:element ref="xacml-context:Environment"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Request> est du type complexe **RequestType**.

L'élément <Request> contient les éléments suivants:

– <Subject> [de un à plusieurs]

  Spécifie les informations sur un sujet du contexte de la demande en faisant la liste d'une séquence d'éléments <Attribute> associés au sujet. Un ou plusieurs éléments <Subject> sont admis. Un sujet est une entité associée à la demande d'accès. Par exemple, un sujet peut représenter l'utilisateur humain qui a initialisé l'application à partir de laquelle la demande a été produite; un autre sujet peut représenter le code exécutable de l'application responsable de la création de la demande; un autre sujet peut représenter la machine sur laquelle l'application a été exécutée; et un autre sujet peut représenter l'entité qui sera le receveur de la ressource. Les attributs de chacune de ces entités doivent être inclus dans des éléments <Subject> séparés.

– <Resource> [de un à plusieurs]

  Spécifie les informations sur la ou les ressources pour lesquelles l'accès est demandé en faisant la liste d'une séquence d'éléments <Attribute> associés à la ressource. Il peut inclure un élément <ResourceContent>.

– <Action> [Exigé]

  Spécifie l'action dont l'exécution est demandée sur la ressource en faisant la liste d'un ensemble d'éléments <Attribute> associés à l'action.

– <Environment> [Exigé]

  Contient un ensemble d'éléments <Attribute> pour l'environnement.

7.5.2 Elément <Subject>

L'élément <Subject> spécifie un sujet en faisant la liste d'une séquence d'éléments <Attribute> associés au sujet.

```
<xs:element name="Subject" type="xacml-context:SubjectType"/>
<xs:complexType name="SubjectType">
   <xs:sequence>
          <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="SubjectCategory" type="xs:anyURI"
default="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"/>
</xs:complexType>
```

L'élément <Subject> est du type complexe **SubjectType**.

L'élément <Subject> contient les éléments et attributs suivants:

– SubjectCategory [Facultatif]

Cet attribut indique le rôle joué par le <Subject> parent dans la formation de la demande d'accès. Si cet attribut n'est pas présent dans un élément <Subject> donné, la valeur par défaut de "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject" doit être utilisée, indiquant que l'élément <Subject> parent représente l'entité responsable ultime de l'initiative de la demande d'accès.

Si plus d'un élément <Subject> contient un attribut "urn:oasis:names:tc:xacml:2.0:subject-category" avec la même valeur, le PDP doit alors traiter le contenu de ces éléments comme s'ils étaient contenus dans le même élément <Subject>.

– <Attribute> [Tout Nombre]

Séquence d'attributs qui s'appliquent au sujet.

Normalement, un élément <Subject> contiendra un <Attribute> avec un AttributeId de "urn:oasis:names:tc:xacml:1.0:subject:subject-id", contenant l'identité du sujet.

Un élément <Subject> peut contenir des éléments <Attribute> supplémentaires.

7.5.3 Elément <Resource>

L'élément <Resource> spécifie les informations sur les ressources auxquelles l'accès est demandé, en faisant la liste d'une séquence des éléments <Attribute> associés à la ressource. Il peut inclure le contenu de la ressource.

```
<xs:element name="Resource" type="xacml-context:ResourceType"/>
<xs:complexType name="ResourceType">
   <xs:sequence>
          <xs:element ref="xacml-context:ResourceContent" minOccurs="0"/>
          <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Resource> est du type complexe **ResourceType**.

L'élément <Resource> contient les éléments suivants:

– <ResourceContent> **[Facultatif]**

Contenu de la ressource.

– <Attribute> [Tout Nombre]

Séquence des attributs de la ressource.

L'élément <Resource> peut contenir un ou plusieurs éléments <Attribute> avec un AttributeId de "urn:oasis:names:tc:xacml:2.0:resource:resource-id". Chacun de ces <Attribute> doit être une représentation absolue et pleinement résolue de l'identité de la seule ressource à laquelle l'accès est demandé. S'il y a plus d'une telle représentation absolue et pleinement résolue, et si un <Attribute> avec cet AttributeId est spécifié, un <Attribute> pour chacune de ces représentations distinctes de l'identité de la ressource doit alors être spécifié. Tous les éléments <Attribute> de cette sorte doivent se référer à la même instance unique de ressource. Un profil pour une ressource particulière peut spécifier une représentation normative unique pour des instances de la ressource; dans ce cas, tout <Attribute> avec cet AttributeId ne doit utiliser que cette représentation unique.

Un élément <Resource> peut contenir des éléments <Attribute> supplémentaires.

7.5.4 Elément <ResourceContent>

L'élément <ResourceContent> est un substitut notionnel pour le contenu de la ressource. Si une politique XACML fait référence au contenu de la ressource au moyen d'un élément <AttributeSelector>, l'élément <ResourceContent> doit alors être inclus dans la chaîne RequestContextPath.

```
<xs:complexType name="ResourceContentType" mixed="true">
   <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>
```

L'élément <ResourceContent> est du type complexe **ResourceContentType**.

L'élément <ResourceContent> permet des éléments et attributs arbitraires.

7.5.5 Elément <Action>

L'élément <Action> spécifie l'action demandée sur la ressource, en faisant la liste d'un ensemble d'éléments <Attribute> associés à l'action.

```
<xs:element name="Action" type="xacml-context:ActionType"/>
<xs:complexType name="ActionType">
   <xs:sequence>
          <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Action> est du type complexe **ActionType**.

L'élément <Action> contient les éléments suivants:

– <Attribute> [Tout Nombre]

  Liste des attributs de l'action à effectuer sur la ressource.

7.5.6 Elément <Environment>

L'élément <Environment> contient un ensemble d'attributs de l'environnement.

```
<xs:element name="Environment" type="xacml-context:EnvironmentType"/>
<xs:complexType name="EnvironmentType">
   <xs:sequence>
          <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Environment> est du type complexe **EnvironmentType**.

L'élément <Environment> contient les éléments suivants:

– <Attribute> [Tout Nombre]

  Liste des attributs d'environnement. Les attributs d'environnement sont des attributs qui ne sont associés ni avec la ressource, ni avec l'action ni aucun des sujets de la demande d'accès.

7.5.7 Elément <Attribute>

L'élément <Attribute> est l'abstraction centrale du contexte de la demande. Il contient des métadonnées d'attribut et une ou plusieurs valeurs d'attribut. Les métadonnées d'attribut comprennent l'identifiant d'attribut et le producteur de l'attribut. Les éléments <AttributeDesignator> et <AttributeSelector> dans la politique peuvent se référer à des attributs au moyen de ces métadonnées.

```
<xs:element name="Attribute" type="xacml-context:AttributeType"/>
<xs:complexType name="AttributeType">
   <xs:sequence>
          <xs:element ref="xacml-context:AttributeValue" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="AttributeId" type="xs:anyURI" use="required"/>
   <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
   <xs:attribute name="Issuer" type="xs:string" use="optional"/>
</xs:complexType>
```

L'élément <Attribute> est du type complexe **AttributeType**.

L'élément <Attribute> contient les attributs et éléments suivants:

– AttributeId [Exigé]

  Identifiant d'attribut. Un certain nombre d'identifiants sont réservés par XACML pour noter les attributs communément utilisés.

– DataType [Exigé]

  Type de données du contenu de l'élément <xacml-context:AttributeValue>. Ce doit être un type de primitive défini par la présente Recommandation ou un type (primitive ou structuré) défini dans un espace de nom déclaré dans l'élément <xacml-context>.

– Issuer [Facultatif]

  Producteur de l'attribut. Par exemple, cette valeur d'attribut peut être un x500Name qui lie à une clé publique, ou elle peut être quelque autre identifiant échangé hors bande par les parties productrice et consommatrice.

– <xacml-context:AttributeValue> [de un à plusieurs]

  Une ou plusieurs valeurs d'attribut. Chaque valeur d'attribut peut avoir des contenus vides, survenir une fois ou survenir plusieurs fois.

7.5.8 Elément <AttributeValue>

L'élément <xacml-context:AttributeValue> contient la valeur d'un attribut.

```
<xs:element name="AttributeValue" type="xacml-context:AttributeValueType"/>
<xs:complexType name="AttributeValueType" mixed="true">
   <xs:sequence>
         <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>
```

L'élément <xacml-context:AttributeValue> est du type complexe **AttributeValueType**.

Le type de données de <xacml-context:AttributeValue> doit être spécifié en utilisant l'attribut DataType de l'élément <Attribute> parent.

7.5.9 Elément <Response>

L'élément <Response> est un élément de niveau supérieur dans le schéma du contexte XACML. L'élément <Response> est une couche d'abstraction utilisée par le langage de politique. Tout système propriétaire utilisant XACML doit transformer un élément <Response> de contexte XACML dans la forme de sa décision d'autorisation.

L'élément <Response> encapsule la décision d'autorisation produite par le PDP. Il inclut une séquence d'un ou plusieurs résultats, avec un élément <Result> par ressource demandée. Plusieurs résultats peuvent être retournés par certaines implémentations, en particulier celles qui prennent en charge le profil XACML de demande de ressources multiples. La prise en charge de résultats multiples est facultative.

```
<xs:element name="Response" type="xacml-context:ResponseType"/>
<xs:complexType name="ResponseType">
   <xs:sequence>
         <xs:element ref="xacml-context:Result" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Response> est du type complexe **ResponseType**.

L'élément <Response> contient les éléments suivants:

– <Result> [de un à plusieurs]

  Résultat de décision d'autorisation.

7.5.10 Elément <Result>

L'élément <Result> représente un résultat de décision d'autorisation pour la ressource spécifiée par l'attribut ResourceId. Il peut inclure un ensemble d'obligations qui doivent être satisfaites par le PEP. Si le PEP ne comprend

pas ou ne peut pas satisfaire à une obligation, il doit alors agir comme si le PDP avait refusé l'accès à la ressource demandée.

```
<xs:complexType name="ResultType">
   <xs:sequence>
         <xs:element ref="xacml-context:Decision"/>
         <xs:element ref="xacml-context:Status" minOccurs="0"/>
         <xs:element ref="xacml:Obligations" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="ResourceId" type="xs:string" use="optional"/>
</xs:complexType>
```

L'élément `<Result>` est du type complexe **ResultType**.

L'élément `<Result>` contient les attributs et éléments suivants:

– `ResourceId` [Facultatif]

  Identifiant de la ressource demandée. Si cet attribut est omis, l'identité de ressource est alors celle qui est spécifiée par l'attribut de ressource `"urn:oasis:names:tc:xacml:1.0:resource:resource-id"` dans l'élément <Request> correspondant.

– `<Decision>` [Exigé]

  Décision d'autorisation: "Permit", "Deny", "Indeterminate" ou "NotApplicable".

– `<Status>` [Facultatif]

  Indique si des erreurs sont survenues pendant l'évaluation de la demande de décision, et facultativement, des informations sur ces erreurs. Si l'élément `<Response>` contient des éléments `<Result>` dont les éléments `<Status>` sont tous identiques, et si l'élément `<Response>` est contenu dans une enveloppe de protocole qui peut convoyer des informations d'état, les informations d'état communes peuvent alors être placées dans l'enveloppe de protocole et cet élément `<Status>` peut être omis dans tous les éléments `<Result>`.

– `<Obligations>` [Facultatif]

  Liste des obligations que le PEP doit satisfaire. Si le PEP ne comprend pas ou ne peut pas satisfaire à une obligation, il doit alors agir comme si le PDP avait refusé l'accès à la ressource demandée.

7.5.11 Elément <Decision>

L'élément `<Decision>` contient le résultat de l'évaluation de politique.

```
<xs:element name="Decision" type="xacml-context:DecisionType"/>
<xs:simpleType name="DecisionType">
   <xs:restriction base="xs:string">
         <xs:enumeration value="Permit"/>
         <xs:enumeration value="Deny"/>
         <xs:enumeration value="Indeterminate"/>
         <xs:enumeration value="NotApplicable"/>
   </xs:restriction>
</xs:simpleType>
```

L'élément `<Decision>` est du type simple **DecisionType**.

Les valeurs de l'élément `<Decision>` ont les significations suivantes:

– Permit: l'accès demandé est permis.

– Deny: l'accès demandé est refusé.

– Indeterminate: le PDP est incapable d'évaluer l'accès demandé. Une telle incapacité peut avoir les raisons suivantes: attributs manquants, erreurs du réseau pendant la restitution des politiques, division par zéro durant l'évaluation de politique, erreurs de syntaxe dans la demande de décision ou dans la politique, etc.

– NotApplicable: le PDP n'a aucune politique qui s'applique à cette demande de décision.

7.5.12 Elément <Status>

L'élément `<Status>` représente l'état du résultat de la décision d'autorisation.

```
<xs:element name="Status" type="xacml-context:StatusType"/>
<xs:complexType name="StatusType">
   <xs:sequence>
          <xs:element ref="xacml-context:StatusCode"/>
          <xs:element ref="xacml-context:StatusMessage" minOccurs="0"/>
          <xs:element ref="xacml-context:StatusDetail" minOccurs="0"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <Status> est du type complexe **StatusType**.

L'élément <Status> contient les éléments suivants:

– <StatusCode> [Exigé]

  Code d'état.

– <StatusMessage> [Facultatif]

  Message d'état qui décrit le code d'état.

– <StatusDetail> [Facultatif]

  Informations d'état supplémentaires.

7.5.13 Elément <StatusCode>

L'élément <StatusCode> contient une valeur de code d'état majeure et une séquence facultative de codes d'état mineurs.

```
<xs:element name="StatusCode" type="xacml-context:StatusCodeType"/>
<xs:complexType name="StatusCodeType">
   <xs:sequence>
          <xs:element ref="xacml-context:StatusCode" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="Value" type="xs:anyURI" use="required"/>
</xs:complexType>
```

L'élément <StatusCode> est du type complexe **StatusCodeType**.

L'élément <StatusCode> contient les attributs et éléments suivants:

– Value [Exigé]

  Voir dans le § B.8 la liste des valeurs.

– <StatusCode> [Tout Nombre]

  Code d'état mineur. Ce code d'état qualifie son code d'état parent.

7.5.14 Elément <StatusMessage>

L'élément <StatusMessage> est une description de forme libre du code d'état.

```
<xs:element name="StatusMessage" type="xs:string"/>
```

L'élément <StatusMessage> est du type **xs:string**.

7.5.15 Elément <StatusDetail>

L'élément <StatusDetail> qualifie l'élément <Status> avec des informations supplémentaires.

```
<xs:element name="StatusDetail" type="xacml-context:StatusDetailType"/>
<xs:complexType name="StatusDetailType">
   <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

L'élément <StatusDetail> est du type complexe **StatusDetailType**.

L'élément <StatusDetail> admet un contenu XML arbitraire.

L'inclusion d'un élément <StatusDetail> est facultative. Cependant, si un PDP retourne une des valeurs <StatusCode> définies par XACML suivantes et inclut un élément <StatusDetail>, les règles suivantes s'appliquent alors:

```
urn:oasis:names:tc:xacml:1.0:status:ok
```

Un PDP ne doit pas retourner un élément <StatusDetail> conjointement avec la valeur d'état "ok".

```
urn:oasis:names:tc:xacml:1.0:status:missing-attribute
```

Un PDP peut choisir de ne pas retourner d'informations <StatusDetail> ou peut choisir de retourner un élément <StatusDetail> contenant un ou plusieurs éléments <xacml-context: MissingAttributeDetail>.

```
urn:oasis:names:tc:xacml:1.0:status:syntax-error
```

Un PDP ne doit pas retourner un élément <StatusDetail> conjointement avec la valeur d'état de "syntax-error". Une erreur de syntaxe peut représenter un problème avec la politique utilisée ou avec le contexte de la demande. Le PDP peut retourner un <StatusMessage> décrivant le problème.

```
urn:oasis:names:tc:xacml:1.0:status:processing-error
```

Un PDP ne doit pas retourner d'élément <StatusDetail> en conjonction avec la valeur d'état "processing-error". Ce code d'état indique un problème interne dans le PDP. Pour des raisons de sécurité, le PDP peut choisir de ne pas retourner d'autres informations au PEP. Dans le cas d'une erreur de division par zéro ou autre erreur de calcul, le PDP peut retourner un <StatusMessage> décrivant la nature de l'erreur.

7.5.16 Elément <MissingAttributeDetail>

L'élément <MissingAttributeDetail> porte des informations sur les attributs nécessaires à l'évaluation de politique qui manquaient dans le contexte de la demande.

```
<xs:element name="MissingAttributeDetail" type="xacml-
context:MissingAttributeDetailType"/>
<xs:complexType name="MissingAttributeDetailType">
<xs:sequence>
<xs:element ref="xacml-context:AttributeValue" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="AttributeId" type="xs:anyURI" use="required"/>
<xs:attribute name="DataType" type="xs:anyURI" use="required"/>
   <xs:attribute name="Issuer" type="xs:string" use="optional"/>
</xs:complexType>
```

L'élément <MissingAttributeDetail> est du type complexe **MissingAttributeDetailType**.

L'élément <MissingAttributeDetail> contient les attributs et éléments suivants:

– AttributeValue [Facultatif]

  Valeur demandée de l'attribut manquant.

– <AttributeId> [Exigé]

  Identifiant de l'attribut manquant.

– <DataType> [Exigé]

  Type de données de l'attribut manquant.

– Issuer [Facultatif]

  Cet attribut, s'il est fourni, doit spécifier le producteur de l'attribut manquant demandé.

Si le PDP inclut des éléments <xacml-context:AttributeValue> dans l'élément <MissingAttributeDetail>, cela indique alors les valeurs acceptables pour cet attribut. Si aucun élément <xacml-context:AttributeValue> n'est inclus, cela indique alors les noms des attributs que le PDP n'a pas réussi à résoudre durant son évaluation. La liste des attributs peut être partielle ou complète. Le PDP ne peut garantir que la fourniture des valeurs ou attributs manquants sera suffisante pour satisfaire la politique.

7.6 Exigences fonctionnelles de XACML

Le présent paragraphe spécifie certaines exigences fonctionnelles qui ne sont pas directement associées à la production ou la consommation d'un élément XAML particulier.

7.6.1 Point de mise en application de politique

Le présent paragraphe décrit les exigences pour le PEP.

Une application fonctionne dans le rôle de PEP si elle garde l'accès à un ensemble de ressources et demande au PDP une décision d'autorisation. Le PEP doit respecter les décisions d'autorisation du PDP.

7.6.1.1 PEP de base

Si la décision est "Permit", le PEP doit alors permettre l'accès. Si des obligations accompagnent la décision, le PEP doit alors ne permettre l'accès que s'il les comprend, et s'il peut et va satisfaire à ces obligations.

Si la décision est "Deny", le PEP doit alors refuser l'accès. Si des obligations accompagnent la décision, le PEP ne doit alors refuser l'accès que s'il les comprend, et s'il peut et va satisfaire à ces obligations.

Si la décision est "Not Applicable", le comportement du PEP est alors indéfini.

Si la décision est "Indeterminate", le comportement du PEP est alors indéfini.

7.6.1.2 PEP enclin à refuser

Si la décision est "Permit", le PEP doit alors permettre l'accès. Si des obligations accompagnent la décision, le PEP ne doit alors permettre l'accès que s'il les comprend, et s'il peut et va satisfaire à ces obligations.

Toute autre décision doit résulter en un refus d'accès.

NOTE – D'autres actions, par exemple, consultation de PDP supplémentaires, reformulation/resoumission de la demande de décision, etc., ne sont pas interdites.

7.6.1.3 PEP enclin à permettre

Si la décision est "Deny", le PEP doit alors refuser l'accès. Si des obligations accompagnent la décision, le PEP ne doit alors refuser l'accès que s'il les comprend, et s'il peut et va satisfaire à ces obligations.

Toute autre décision doit résulter en une permission d'accès.

NOTE – D'autres actions, par exemple, consultation de PDP supplémentaires, reformulation/resoumission de la demande de décision, etc., ne sont pas interdites.

7.6.2 Evaluation d'attribut

Les attributs sont représentés dans le contexte de la demande par le gestionnaire de contexte, sans considération du fait qu'ils apparaissent ou non dans la demande de décision originale, et la politique s'y réfère par désignateurs d'attribut et sélecteurs d'attribut, de sujet, ressource, action et environnement. Un attribut nommé est le terme utilisé pour le critère que les désignateurs et sélecteurs spécifiques d'attribut sujet, ressource, action et environnement utilisent pour se référer à des attributs particuliers dans les éléments, respectivement de sujet, ressource, action et environnement du contexte de la demande.

7.6.2.1 Attributs structurés

Les éléments `<xacml:AttributeValue>` et `<xacml-context:AttributeValue>` peuvent contenir une instance d'un type de données XML structuré, par exemple `<ds:KeyInfo>`. La présente Recommandation prend en charge plusieurs manières de comparer les contenus de tels éléments.

1) Dans certains cas, de tels éléments peuvent être comparés en utilisant une des fonctions de chaîne XACML, telle que "string-regexp-match", décrite ci-dessous. Cela exige que l'élément reçoive le type de données "http://www.w3.org/2001/XMLSchema#string". Par exemple, un type de données structuré qui est en réalité un **ds:KeyInfo/KeyName** apparaîtrait dans le contexte comme:

```
<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
   &lt;ds:KeyName&gt;jhibbert-key&lt;/ds:KeyName&gt;
</AttributeValue>
```

En général, cette méthode ne sera pas adéquate, sauf si le type de données structuré est assez simple.

2) Un élément <AttributeSelector> peut être utilisé pour choisir le contenu d'un sous-élément folié du type de données structurées au moyen d'une expression XPath. Cette valeur peut alors être comparée en utilisant comme type de données de primitive une des fonctions XACML appropriées prises en charge. Cette méthode exige la prise en charge par le PDP de la caractéristique d'expression XPath facultative.

3) Un élément <AttributeSelector> peut être utilisé pour choisir tout nœud dans le type de données structuré au moyen d'une expression XPath. Ce nœud peut alors être comparé en utilisant une des fonctions fondées sur XPath décrites dans le § A.3. Cette méthode exige la prise en charge par le PDP des caractéristiques facultatives des expressions et fonctions XPath.

7.6.2.2 Sacs d'attribut

XACML définit des collections implicites de ses types de données. XACML se réfère à une collection de valeurs qui sont d'un seul type de données comme à un sac. Les sacs de types de données sont nécessaires parce que le choix des nœuds d'après un contexte de ressource XML ou demande XACML peut retourner plus d'une valeur.

L'élément <AttributeSelector> utilise une expression XPath pour spécifier le choix des données d'après une ressource XML. Le résultat d'une expression XPath est appelé un ensemble-nœud, qui contient tous les nœuds foliés provenant de la ressource XML qui correspondent au prédicat dans l'expression XPath. Sur la base de diverses fonctions d'indexation fournies dans W3C XPath:1999, il doit être implicite qu'un ensemble-nœud résultant est la collection des nœuds qui correspondent. La présente Recommandation définit aussi l'élément <AttributeDesignator> comme ayant la même méthodologie de correspondance pour les attributs dans le contexte XACML Request.

Les valeurs dans un sac ne sont pas ordonnées, et certaines de ces valeurs peuvent être dupliquées. Il ne doit pas y avoir de notion de sac contenant des sacs, ou d'un sac contenant des valeurs de différents types (c'est-à-dire que dans XACML, un sac ne doit contenir que des valeurs qui sont du même type de données).

7.6.2.3 Attributs à valeurs multiples

Si un élément <Attribute> unique dans un contexte de demande contient plusieurs éléments fils <xacml-context:AttributeValue>, le sac des valeurs résultant de l'évaluation de l'élément <Attribute> doit alors être identique au sac des valeurs qui résultent de l'évaluation d'un contexte dans lequel chaque élément <xacml-context:AttributeValue> apparaît dans un élément <Attribute> séparé, chacun portant des métadonnées identiques.

7.6.2.4 Correspondance d'attribut

Un attribut nommé inclut des critères spécifiques avec lesquels faire correspondre les attributs dans le contexte. Un attribut spécifie un AttributeId et un DataType, et un attribut nommé spécifie aussi le producteur. Un attribut nommé doit correspondre à un attribut si les valeurs de leurs attributs respectifs AttributeId, DataType et de l'attribut facultatif de producteur correspondent à leur élément particulier – sujet, ressource, action ou environnement – du contexte. Le AttributeId de l'attribut nommé doit correspondre, par égalité d'URI, à l'AttributeId de l'attribut de contexte correspondant. Le DataType de l'attribut nommé doit correspondre, par égalité d'URI, au DataType de l'attribut de contexte correspondant. Si le producteur est fourni dans l'attribut nommé, il doit alors correspondre, en utilisant la fonction urn:oasis:names:tc:xacml:1.0:function:string-equal, au producteur de l'attribut de contexte correspondant. Si le producteur n'est pas fourni dans l'attribut nommé, la correspondance de l'attribut de contexte avec l'attribut nommé doit être gouvernée par AttributeId et DataType seuls, indépendamment de la présence, absence, ou valeur réelle de Issuer (producteur) dans l'attribut de contexte correspondant. Dans le cas d'un sélecteur d'attribut, la correspondance de l'attribut avec l'attribut nommé doit être gouvernée par l'expression XPath et le DataType.

7.6.2.5 Restitution d'attribut

Le PDP doit demander les valeurs des attributs dans le contexte de la demande au gestionnaire de contexte. Le PDP doit faire référence aux attributs comme s'ils étaient dans un document de contexte de demande physique, mais le gestionnaire de contexte est responsable de l'obtention et de la fourniture des valeurs demandées par tout moyen qu'il juge approprié. Le gestionnaire de contexte doit retourner les valeurs des attributs qui correspondent au désignateur d'attribut ou au sélecteur d'attribut et les mettre sous la forme d'un sac de valeurs avec le type de données spécifié. Si aucun attribut provenant du contexte de la demande ne correspond, l'attribut doit alors être considéré comme manquant. Si l'attribut est manquant, MustBePresent commande alors si le désignateur d'attribut ou le sélecteur d'attribut retourne un sac vide ou un résultat "Indeterminate". Si MustBePresent est "False" (valeur par défaut), un attribut manquant doit avoir pour résultat un sac vide. Si MustBePresent est "True", un attribut manquant doit alors avoir pour résultat "Indeterminate". Ce résultat "Indeterminate" doit être traité en conformité avec la spécification des expressions, règles, politiques et ensembles de politiques en cours. Si le résultat est "Indeterminate", les AttributeId,

`DataType` et `Issuer` (producteur) de l'attribut peuvent être énumérés dans la décision d'autorisation. Cependant, un PDP peut choisir de ne pas retourner de telles informations pour des raisons de sécurité.

7.6.2.6 Attributs d'environnement

Si une valeur de l'un de ces attributs est fournie dans la demande de décision, le gestionnaire de contexte doit alors utiliser cette valeur. Autrement, le gestionnaire de contexte doit fournir une valeur. Dans le cas d'attributs de date et heure, la valeur fournie doit avoir la sémantique de la "date et heure qui s'appliquent à la demande de décision".

7.6.3 Evaluation d'expression

XACML spécifie des expressions en termes d'éléments dont la liste figure ci-dessous, dont les éléments `<Apply>` et `<Condition>` composent de façon récurrente de plus grandes expressions. Les expressions valides doivent être de type correct, ce qui signifie que les types de chacun des éléments contenus dans les éléments `<Apply>` et `<Condition>` doivent être en accord avec les types d'argument respectifs de la fonction qui est désignée par l'attribut `FunctionId`. Le type résultant de l'élément `<Apply>` ou `<Condition>` doit être le type résultant de la fonction, qui peut être rétréci à un type de données de primitive, ou un sac de type de données de primitive, par unification de type. XACML définit un résultat d'évaluation de "Indeterminate", qui est réputé être le résultat d'une expression non valide, ou d'une erreur de fonctionnement survenant durant l'évaluation de l'expression.

XACML définit ces éléments pour le groupe de substitution de l'élément `<Expression>`:

- `<xacml:AttributeValue>`
- `<xacml:SubjectAttributeDesignator>`
- `<xacml:ResourceAttributeDesignator>`
- `<xacml:ActionAttributeDesignator>`
- `<xacml:EnvironmentAttributeDesignator>`
- `<xacml:AttributeSelector>`
- `<xacml:Apply>`
- `<xacml:Condition>`
- `<xacml:Function>`
- `<xacml:VariableReference>`

7.6.4 Evaluation arithmétique

IEEE 754 spécifie comment évaluer les fonctions arithmétiques dans un contexte. Ces fonctions spécifient les valeurs par défaut pour les approximations, les arrondis, etc. XACML doit utiliser cette spécification pour l'évaluation de toutes les fonctions entières et doubles en s'appuyant sur le *Contexte étendu par défaut*, amélioré par la double précision:

- fanions: tous mis à 0;
- activeurs de trap: tous mis à 0 à l'exception de l'activeur de trap "division par zéro", qui doit être mis à 1;
- précision: est mis à la double précision désignée;
- arrondi: est mis à arrondi-demi-pair.

7.6.5 Evaluation de correspondance

Les éléments de correspondance d'attribut apparaissent dans l'élément `<Target>` des règles, politiques et ensembles de politiques. Ce sont:

- `<SubjectMatch>`
- `<ResourceMatch>`
- `<ActionMatch>`
- `<EnvironmentMatch>`

Ces éléments représentent respectivement des expressions booléennes sur les attributs de sujet, ressource, action et environnement. Un élément correspondant contient un attribut MatchId qui spécifie la fonction à utiliser pour effectuer l'évaluation de correspondance, un `<xacml:AttributeValue>` et un élément `<AttributeDesignator>` ou `<AttributeSelector>` qui spécifie l'attribut dans le contexte qui est à confronter à la valeur spécifiée.

L'attribut `MatchId` doit spécifier une fonction qui compare deux arguments, et retourner un type de résultat de "http://www.w3.org/2001/XMLSchema#boolean". La valeur d'attribut spécifiée dans l'élément de correspondance doit être fournie à la fonction `MatchId` comme son premier argument. Un élément du sac retourné par l'élément

<AttributeDesignator> ou <AttributeSelector> doit être fourni à la fonction MatchId comme son second argument, comme expliqué ci-dessous. Le DataType de <xacml:AttributeValue> doit correspondre au type de données du premier argument attendu par la fonction MatchId. Le DataType de l'élément <AttributeDesignator> ou <AttributeSelector> doit correspondre au type de données du second argument attendu par la fonction MatchId.

Les fonctions de la norme XACML qui satisfont aux exigences à utiliser comme valeur d'attribut MatchId sont:

```
urn:oasis:names:tc:xacml:2.0:function:-type-equal
urn:oasis:names:tc:xacml:2.0:function:-type-greater-than
urn:oasis:names:tc:xacml:2.0:function:-type-greater-than-or-equal
urn:oasis:names:tc:xacml:2.0:function:-type-less-than
urn:oasis:names:tc:xacml:2.0:function:-type-less-than-or-equal
urn:oasis:names:tc:xacml:2.0:function:-type-match
```

De plus, les fonctions qui sont strictement contenues dans une extension de XACML peuvent apparaître comme valeur pour l'attribut MatchId, et ces fonctions peuvent utiliser des types de données qui sont aussi des extensions, pour autant que la fonction d'extension retourne un résultat booléen et prenne pour entrée deux types de base uniques. La fonction utilisée comme valeur pour l'attribut MatchId devrait être facilement indexable. L'utilisation de fonctions non indexables ou complexes peut empêcher une évaluation efficace des demandes de décision.

La sémantique d'évaluation pour un élément correspondant est la suivante. Si une erreur de fonctionnement devait survenir durant l'évaluation de l'élément <AttributeDesignator> ou <AttributeSelector>, le résultat de l'expression entière doit être "Indeterminate". Si l'élément <AttributeDesignator> ou <AttributeSelector> devait être évalué comme un sac vide, le résultat de l'expression devrait alors être "False". Autrement, la fonction MatchId doit être appliquée entre <xacml:AttributeValue> et chaque élément du sac retourné de l'élément <AttributeDesignator> ou <AttributeSelector>. Si au moins une de ces applications de fonction devait être évaluée comme "True", le résultat de l'expression entière devrait être "True". Autrement, si au moins une des applications de fonction a pour résultat "Indeterminate", le résultat doit alors être "Indeterminate". Finalement, si toutes les applications de fonction s'évaluent comme "False", le résultat de l'expression entière doit alors être "False".

Il est aussi possible d'exprimer la sémantique d'un élément de correspondance de cible dans une condition. Par exemple, l'expression de correspondance de cible qui compare un "subject-name" commençant par le nom "John" peut être exprimée comme suit:

```
<SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-regexp-match">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
       John.*
    </AttributeValue>
    <SubjectAttributeDesignator
          AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
          DataType="http://www.w3.org/2001/XMLSchema#string"/>
</SubjectMatch>
```

Autrement, la même sémantique de correspondance peut être exprimée comme un élément <Apply> dans une condition en utilisant la fonction "urn:oasis:names:tc:xacml:1.0:function:any-of", comme suit:

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of">
    <Function
FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-regexp-match"/>
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
        John.*
    </AttributeValue>
    <SubjectAttributeDesignator
         AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
         DataType="http://www.w3.org/2001/XMLSchema#string"/>
</Apply>
```

7.6.6 Evaluation de cible

La valeur de cible doit être "Match" *(correspondance)* si les sujets, ressources, actions et environnements spécifiés dans la cible correspondent tous aux valeurs figurant dans le contexte de la demande. Si un des sujets, ressources, actions et environnements spécifiés dans la cible sont "Indeterminate" *(indéterminé)*, la cible doit alors être "Indeterminate". Autrement, la cible doit être "No match" *(pas de correspondance)*. Le Tableau 7-1 donne les correspondances.

Tableau 7-1/X.1142 – Tableau de correspondance de cible

Valeur de sujet	Valeur de ressource	Valeur d'action	Valeur d'environnement	Valeur de cible
"Match"	"Match"	"Match"	"Match"	"Match"
"No match"	"Match" ou "No match"	"Match" ou "No match"	"Match" ou "No match"	"No match"
"Match" ou "No match"	"No match"	"Match" ou "No match"	"Match" ou "No match"	"No match"
"Match" ou "No match"	"Match" ou "No match"	"No match"	"Match" ou "No match"	"No match"
"Match" ou "No match"	"Match" ou "No match"	"Match" ou "No match"	"No match"	"No match"
"Indeterminate"	Sans importance	Sans importance	Sans importance	"Indeterminate"
Sans importance	"Indeterminate"	Sans importance	Sans importance	"Indeterminate"
Sans importance	Sans importance	"Indeterminate"	Sans importance	"Indeterminate"
Sans importance	Sans importance	Sans importance	"Indeterminate"	"Indeterminate"

Les sujets, ressources, actions et environnements doivent correspondre aux valeurs dans le contexte de la demande si au moins un de leurs éléments `<Subject>`, `<Resource>`, `<Action>` ou `<Environment>` correspondent, respectivement, à une valeur dans le contexte de la demande. Le tableau de correspondance des sujets figure au Tableau 7-2. Les tableaux de correspondance des ressources, actions et environnements sont analogues.

Tableau 7-2/X.1142 – Correspondance de sujets

Valeurs de `<Subject>`	**Valeurs de** `<Subjects>`
Au moins une "Match"	"Match"
Aucune ne correspond et au moins une "Indeterminate"	"Indeterminate"
"No match" pour toutes	"No match"

Un sujet, ressource, action ou environnement doit correspondre à une valeur dans le contexte de la demande si la valeur de tous ses éléments `<SubjectMatch>`, `<ResourceMatch>`, `<ActionMatch>` ou `<EnvironmentMatch>`, sont respectivement "True".

Le tableau de correspondance de sujet est donné au Tableau 7-3. Les tableaux de correspondance de ressource, action et environnement sont analogues.

Tableau 7-3/X.1142 – Correspondance de sujet

Valeurs de `<SubjectMatch>`	**Valeur de** `<Subject>`
Toutes à "True"	"Match"
Pas de "False" et au moins une "Indeterminate"	"Indeterminate"
Au moins une "False"	"No match"

7.6.7 Evaluation de VariableReference

L'élément `<VariableReference>` fait référence à un seul élément `<VariableDefinition>` contenu dans le même élément `<Policy>`. Un `<VariableReference>` qui ne fait pas référence à un élément `<VariableDefinition>` particulier dans l'élément `<Policy>` en cours est appelé une référence indéfinie. Les politiques avec des références indéfinies sont invalides.

A tout endroit où survient un `<VariableReference>`, il a le même effet que si le texte de l'élément `<Expression>` défini dans l'élément `<VariableDefinition>` remplaçait l'élément `<VariableReference>`. Tout schéma d'évaluation qui préserve cette sémantique est acceptable. Par exemple, l'expression dans l'élément `<VariableDefinition>` peut sans conséquences être évaluée à une valeur particulière et mise en mémoire cache pour plusieurs références. (C'est-à-dire que la valeur d'un élément `<Expression>` reste la même pour l'évaluation de politique tout entière.). Cette caractéristique est un des avantages de XACML comme langage déclaratif.

7.6.8 Evaluation de condition

La valeur de condition doit être "True" si l'élément `<Condition>` est absent, ou s'il est évalué comme "True". Sa valeur doit être "False" si l'élément `<Condition>` est évalué comme "False". La valeur de condition doit être "Indeterminate", si l'expression contenue dans l'élément `<Condition>` est évaluée comme "Indeterminate."

7.6.9 Evaluation de règle

Une règle a une valeur qui peut être calculée en évaluant son contenu. L'évaluation de Rule implique une évaluation séparée de la cible de la règle et de la condition. Le tableau des règles figure au Tableau 7-4.

Tableau 7-4/X.1142 – Tableau de véracité de règle

Cible	Condition	Valeur de Rule
"Match"	"True"	Effect
"Match"	"False"	"NotApplicable"
"Match"	"Indeterminate"	"Indeterminate"
"No-match"	Sans importance	"NotApplicable"
"Indeterminate"	Sans importance	"Indeterminate"

Si la valeur de cible est "No-match" ou "Indeterminate", la valeur de règle doit alors être respectivement "NotApplicable" ou "Indeterminate", sans considération de la valeur de la condition. Pour ces cas, donc, il n'est pas nécessaire d'évaluer la condition.

Si la valeur de la cible est "Match" et si la valeur de la condition est "True", l'effet spécifié dans l'élément `<Rule>` englobant doit alors déterminer la valeur de la règle.

7.6.10 Evaluation de politique

La valeur d'une politique doit être déterminée par son seul contenu, considéré en relation avec le contenu du contexte de la demande. Une valeur de politique doit être déterminée par l'évaluation de la cible et des règles de la politique.

La cible de la politique doit être évaluée pour déterminer l'applicabilité de la politique. Si la cible s'évalue comme "Match", la valeur de la politique doit alors être déterminée par l'évaluation des règles de la politique, conformément à l'algorithme de combinaison de règles spécifié. Si la cible s'évalue comme "No-match", la valeur de la politique doit alors être "NotApplicable". Si la cible s'évalue comme "Indeterminate", la valeur de la politique doit alors être "Indeterminate".

Le tableau de véracité de la politique est donné au Tableau 7-5.

Tableau 7-5/X.1142 – Tableau de véracité de politique

Cible	Valeurs de règle	Valeur de politique
"Match"	Au moins une valeur de règle est son effet	Spécifié par l'algorithme de combinaison de règle
"Match"	Toutes les valeurs de règle sont "NotApplicable"	"NotApplicable"
"Match"	Au moins une valeur de règle est "Indeterminate"	Spécifié par l'algorithme de combinaison de règle
"No-match"	Sans importance	"NotApplicable"
"Indeterminate"	Sans importance	"Indeterminate"

Une valeur de règles de "Au moins une valeur de règle est son Effet" signifie que l'élément `<Rule>` est absent, ou qu'une ou plusieurs des règles contenues dans la politique sont applicables à la demande de décision (c'est-à-dire, qu'elle retourne la valeur de son "Effet"). Une valeur de règles de "Toutes les valeurs de règle sont 'NotApplicable'" doit être utilisée si aucune règle contenue dans la politique n'est applicable à la demande et si aucune règle contenue dans la politique ne retourne une valeur de "Indeterminate". Si aucune règle contenue dans la politique n'est applicable à la demande, mais qu'une ou plusieurs règles retournent une valeur de "Indeterminate", les règles doivent alors s'évaluer comme "Au moins une valeur de règle est 'Indeterminate'".

Si la valeur de cible est "No-match" ou "Indeterminate", la valeur de la politique doit alors être "NotApplicable" ou "Indeterminate", respectivement, sans considération de la valeur des règles. Donc, pour ces cas, il n'est pas nécessaire d'évaluer les règles.

Si la valeur de cible est "Match" et si la valeur de règle est "Au moins une valeur de règle est son Effet" ou "Au moins une valeur de règle est 'Indeterminate'", l'algorithme de combinaison de règles spécifié dans la politique doit alors déterminer la valeur de la politique.

Noter qu'aucun des algorithmes de combinaison de règles défini par la présente Recommandation ne prend de paramètre. Cependant, des algorithmes de combinaison non standard peuvent prendre des paramètres. Dans un tel cas, les valeurs de ces paramètres associées aux règles doivent être prises en compte lors de l'évaluation de la politique. Les paramètres et leurs types devraient être définis dans la spécification de l'algorithme de combinaison. Si l'implémentation prend en charge les paramètres de combineur et si les paramètres de combineur sont présents dans une politique, les valeurs de paramètre doivent alors être fournies à l'implémentation d'algorithme de combinaison.

7.6.11 Evaluation d'ensemble de politiques

La valeur d'un ensemble de politiques doit être déterminée par son contenu, considéré en relation avec le contenu du contexte de la demande. Une valeur d'ensemble de politiques doit être déterminée par évaluation de la cible, des politiques et ensembles de politiques de l'ensemble de politiques, conformément à l'algorithme de combinaison de politiques spécifié.

La cible de l'ensemble de politiques doit être évaluée pour déterminer l'applicabilité de l'ensemble de politiques. Si la cible s'évalue comme "Match", la valeur de l'ensemble de politiques doit alors être déterminée par évaluation des politiques et ensembles de politiques de l'ensemble de politiques, conformément à l'algorithme de combinaison de politiques spécifié. Si la cible s'évalue comme "No-match", la valeur de l'ensemble de politiques doit alors être "NotApplicable". Si la cible s'évalue comme "Indeterminate", la valeur de l'ensemble de politiques doit alors être "Indeterminate".

Le tableau de véracité d'ensemble de politiques est donné au Tableau 7-6.

Tableau 7-6/X.1142 – Tableau de véracité d'ensemble de politiques

Cible	Valeurs de politique	Valeur d'ensemble de politiques
"Match"	Au moins une valeur de politique est sa Décision	Spécifié par l'algorithme de combinaison de politiques
"Match"	Toutes les valeurs de politique sont "NotApplicable"	"NotApplicable"
"Match"	Au moins une valeur de politique est "Indeterminate"	Spécifié par l'algorithme de combinaison de politiques
"No-match"	Sans importance	"NotApplicable"
"Indeterminate"	Sans importance	"Indeterminate"

Une valeur de politiques de "Au moins une valeur de politique est sa Decision" doit être utilisée s'il n'y a aucune politique ou ensembles de politiques contenus ou référencés, ou si une ou plusieurs des politiques ou ensembles de politiques contenus dans l'ensemble de politiques ou référencés par eux sont applicables à la demande de décision (c'est-à-dire, retournent une valeur déterminée par son algorithme de combinaison). Une valeur de politiques de "Toutes les valeurs de politique sont 'NotApplicable'" doit être utilisée si aucune politique ou ensemble de politiques contenu dans l'ensemble de politiques ou référencé par lui n'est applicable à la demande et si aucune politique ou ensemble de politiques contenu dans l'ensemble de politiques ou référencé par lui ne retourne une valeur de "Indeterminate". Si aucune politique ou ensemble de politiques contenu dans l'ensemble de politiques ou référencé par lui n'est applicable à la demande mais qu'une ou plusieurs politiques ou ensembles de politiques retournent une valeur de "Indeterminate", la politique doit alors s'évaluer comme "Au moins une valeur de politique est 'Indeterminate'".

Si la valeur de cible est "No-match" ou "Indeterminate", la valeur de l'ensemble de politiques doit alors être "NotApplicable" ou "Indeterminate", respectivement, sans considération de la valeur des politiques. Donc, pour ces cas, il n'est pas nécessaire d'évaluer les politiques.

Si la valeur de cible est "Match" et si les valeurs des politiques sont "Au moins une valeur de politique est sa Decision" ou "Au moins une valeur de politique est 'Indeterminate'", l'algorithme de combinaison de politiques spécifié dans l'ensemble de politiques doit alors déterminer la valeur de l'ensemble de politiques.

Noter qu'aucun des algorithmes de combinaison de politiques définis par XACML 2.0 ne prend de paramètres. Cependant, des algorithmes de combinaison non standard peuvent prendre des paramètres. Dans de tels cas, les valeurs de ces paramètres associés aux politiques doivent être prises en compte lors de l'évaluation de l'ensemble de politiques. Les paramètres et leurs types devraient être définis dans la spécification de l'algorithme de combinaison. Si l'implémentation prend en charge les paramètres de combineur et si les paramètres de combineur sont présents dans une politique, les valeurs de paramètre doivent alors être fournies à l'implémentation d'algorithme de combinaison.

7.6.12 Ressources hiérarchiques

Il arrive souvent qu'une ressource soit organisée de façon hiérarchique (par exemple, système de fichiers, document XML). XACML offre plusieurs mécanismes facultatifs pour la prise en charge de ressources hiérarchiques telles qu'exposées dans la présente Recommandation.

7.6.13 Décision d'autorisation

En relation avec une demande de décision donnée, le PDP est défini par un algorithme de combinaison de politiques et un et/ou des ensembles de politiques. Le PDP doit retourner un contexte de réponse comme s'il avait évalué un seul ensemble de politiques consistant en cet algorithme de combinaison de politiques et l'ensemble et/ou les ensembles de politiques.

Le PDP doit évaluer l'ensemble de politiques comme spécifié aux § 7.4 et 7.6. Le PDP doit retourner un contexte de réponse, avec un élément `<Decision>` de valeur "Permit", "Deny", "Indeterminate" ou "NotApplicable".

Si le PDP ne peut pas prendre une décision, un élément `<Decision>` "Indeterminate" doit être retourné.

7.6.14 Obligations

Une politique ou ensemble de politiques peut contenir une ou plusieurs obligations. Lorsqu'une telle politique ou ensemble de politiques est évalué, une obligation ne doit être passée au niveau d'évaluation supérieur (la politique englobante ou de référence, l'ensemble de politiques ou la décision d'autorisation) que si l'effet de la politique ou ensemble de politiques en cours d'évaluation correspond à la valeur de l'attribut `FulfillOn` de l'obligation.

En conséquence de cette procédure, aucune obligation ne doit être retournée au PEP si les politiques ou ensembles de politiques d'où elles sont tirées ne sont pas évaluées, ou si le résultat de leur évaluation est "Indeterminate" ou "NotApplicable", ou si la décision résultant de l'évaluation de la politique ou ensemble de politiques ne correspond pas à la décision résultant de l'évaluation d'un ensemble de politiques englobant.

Si l'évaluation du PDP est vue comme un arbre des ensembles de politiques et politiques, dont chacun retourne "Permit" ou "Deny", l'ensemble des obligations retournées par le PDP au PEP n'inclura alors que les obligations associées aux chemins où les effets à chaque niveau d'évaluation sont les mêmes que les effets retournés par le PDP. Dans les situations où tout manque de déterminisme est inacceptable, un algorithme de combinaison déterministe, tel qu'un algorithme à priorité de refus ordonné *(ordered-deny-overrides)*, devrait être utilisé.

7.6.15 Traitement des exceptions

XACML spécifie le comportement du PDP dans les situations suivantes.

7.6.15.1 Fonctions non prises en charge

Si le PDP essaye d'évaluer un ensemble de politiques ou une politique qui contient un type ou une fonction d'élément facultatif que le PDP ne prend pas en charge, le PDP doit alors retourner une valeur `<Decision>` de "Indeterminate". Si un élément `<StatusCode>` est aussi retourné, sa valeur doit alors être "`urn:oasis:names:tc:xacml:1.0:status:syntax-error`" dans le cas d'un type d'élément non pris en charge, et "`urn:oasis:names:tc:xacml:1.0:status:processing-error`" dans le cas d'une fonction non prise en charge.

7.6.15.2 Erreurs de syntaxe et de type

Si une politique qui contient une syntaxe invalide est évaluée par le PDP XACML au moment où une demande de décision est reçue, le résultat de cette politique doit être "Indeterminate" avec une valeur de StatusCode de

```
"urn:oasis:names:tc:xacml:1.0:status:syntax-error"
```

Si une politique qui contient des types de données statiques non valides est évaluée par le PDP XACML au moment où une demande de décision est reçue, le résultat de cette politique doit être "Indeterminate" avec une valeur de StatusCode de

```
"urn:oasis:names:tc:xacml:1.0:status:processing-error"
```

7.6.15.3 Attributs manquants

L'absence d'attributs correspondants dans le contexte de la demande pour tout désignateur ou sélecteur d'attribut qui se trouve dans la politique doit résulter en un élément `<Decision>` contenant la valeur "Indeterminate". Si, dans ce cas, un code d'état est fourni, la valeur

```
"urn:oasis:names:tc:xacml:1.0:status:missing-attribute"
```

doit être utilisée, pour indiquer que plus d'informations sont nécessaires afin de rendre une décision définitive. Dans ce cas, l'élément <Status> peut faire la liste des noms et types de données de tout attribut des sujets, ressource, action ou environnements qui sont nécessaires au PDP pour affiner sa décision. Un PEP peut resoumettre un contexte de demande plus élaboré en réponse à un contenu d'élément <Decision> de "Indeterminate" sans code d'état de

```
"urn:oasis:names:tc:xacml:1.0:missing-attribute"
```

en ajoutant des valeurs d'attribut pour les noms d'attribut qui figuraient sur la liste de la réponse précédente. Lorsque le PDP retourne un contenu d'élément <Decision> de "Indeterminate", avec un code d'état de

```
"urn:oasis:names:tc:xacml:1.0:missing-attribute"
```

il ne doit faire la liste des noms et types de données d'aucun attribut de sujet, ressource, action ou environnement pour lequel des valeurs ont été fournies dans la demande d'origine. Noter que cette exigence force le PDP à retourner finalement une décision d'autorisation de "Permit", "Deny" ou "Indeterminate" avec d'autres code d'état, en réponse à des demandes affinées successivement.

7.7 Points d'extensibilité XACML

Le présent paragraphe décrit les points au sein du modèle et schéma XACML où des extensions peuvent être ajoutées. Le présent paragraphe est pour information.

7.7.1 Types d'attribut XML extensibles

Les attributs XML suivants ont des valeurs qui sont des URI. Elles peuvent être étendues par la création de nouveaux URI associés à la nouvelle sémantique pour ces attributs.

- AttributeId;
- DataType;
- FunctionId;
- MatchId;
- ObligationId;
- PolicyCombiningAlgId;
- RuleCombiningAlgId;
- StatusCode;
- SubjectCategory.

7.7.2 Attributs structurés

Les éléments <xacml:AttributeValue> et <xacml-context:AttributeValue> peuvent contenir une instance de type de données XML structurées. Ci-dessous figure la liste de quelques techniques qui requièrent des extensions XACML.

1) Pour un type de données structurées donné, une communauté d'utilisateurs XACML peut définir de nouveaux identifiants d'attribut pour chaque sous-élément folié du type de données structurées qui a un type conforme à un des types de données de primitive défini par XACML. En utilisant ces nouveaux identifiants d'attribut, les PEP ou gestionnaires de contexte utilisés par cette communauté d'utilisateurs peuvent aplatir les instances du type de données structurées en une séquence d'éléments <Attribute> individuels. Chaque élément <Attribute> de ce type peut être comparé en utilisant les fonctions définies par XACML. En utilisant cette méthode, le type de données structurées lui-même n'apparaît jamais dans un élément <xacml-context:AttributeValue>.
2) Une communauté d'utilisateurs XACML peut définir une nouvelle fonction pouvant être utilisée pour comparer une valeur du type de données structurées à quelque autre valeur. Cette méthode ne peut être utilisée que par des PDP qui prennent en charge la nouvelle fonction.

7.8 Conformité

XACML définit un certain nombre de fonctions qui ont une application assez particulière, et dont la prise en charge n'est donc pas obligatoire dans une implémentation revendiquant la conformité à la présente Recommandation.

Le présent paragraphe fait la liste des parties de la présente Recommandation qui doivent être incluses dans une implémentation de PDP qui revendique la conformité à XACML v2.0.

NOTE – "M" signifie d'implémentation obligatoire (*mandatory*). "O" (*optional*) signifie facultatif.

7.8.1 Eléments de schéma

La mise en œuvre doit prendre en charge les éléments de schéma marqués "M".

Nom d'élément	M/O
xacml-context:Action	M
xacml-context:Attribute	M
xacml-context:AttributeValue	M
xacml-context:Decision	M
xacml-context:Environment	M
xacml-context:MissingAttributeDetail	M
xacml-context:Obligations	O
xacml-context:Request	M
xacml-context:Resource	M
xacml-context:ResourceContent	O
xacml-context:Response	M
xacml-context:Result	M
xacml-context:Status	M
xacml-context:StatusCode	M
xacml-context:StatusDetail	O
xacml-context:StatusMessage	O
xacml-context:Subject	M
xacml:Action	M
xacml:ActionAttributeDesignator	M
xacml:ActionMatch	M
xacml:Actions	M
xacml:Apply	M
xacml:AttributeAssignment	O
xacml:AttributeSelector	O
xacml:AttributeValue	M
xacml:CombinerParameters	O
xacml:CombinerParameter	O
xacml:Condition	M
xacml:Description	M
xacml:Environment	M
xacml:EnvironmentMatch	M
xacml:EnvironmentAttributeDesignator	M
xacml:Environments	M
xacml:Expression	M
xacml:Function	M
xacml:Obligation	O
xacml:Obligations	O
xacml:Policy	M
xacml:PolicyCombinerParameters	O
xacml:PolicyDefaults	O
xacml:PolicyIdReference	M
xacml:PolicySet	M
xacml:PolicySetDefaults	O
xacml:PolicySetIdReference	M
xacml:Resource	M
xacml:ResourceAttributeDesignator	M

Nom d'élément	M/O
xacml:ResourceMatch	M
xacml:Resources	M
xacml:Rule	M
xacml:RuleCombinerParameters	O
xacml:Subject	M
xacml:SubjectMatch	M
xacml:Subjects	M
xacml:Target	M
xacml:VariableDefinition	M
xacml:VariableReference	M
xacml:XPathVersion	O

7.8.2 Préfixes d'identifiant

Les préfixes d'identifiant suivants sont réservés par XACML.

Identifiant
urn:oasis:names:tc:xacml:2.0
urn:oasis:names:tc:xacml:2.0:conformance-test
urn:oasis:names:tc:xacml:2.0:context
urn:oasis:names:tc:xacml:2.0:example
urn:oasis:names:tc:xacml:1.0:function
urn:oasis:names:tc:xacml:2.0:function
urn:oasis:names:tc:xacml:2.0:policy
urn:oasis:names:tc:xacml:1.0:subject
urn:oasis:names:tc:xacml:1.0:resource
urn:oasis:names:tc:xacml:1.0:action
urn:oasis:names:tc:xacml:1.0:environment
urn:oasis:names:tc:xacml:1.0:status

7.8.3 Algorithmes

L'implémentation doit inclure les algorithmes de combinaison de règle et de politique associés aux identifiants suivants qui sont marqués "M".

Algorithme	M/O
urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides	M
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:deny-overrides	M
urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides	M
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:permit-overrides	M
urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable	M
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:first-applicable	M
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:only-one-applicable	M
urn:oasis:names:tc:xacml:1.1:rule-combining-algorithm:ordered-deny-overrides	M
urn:oasis:names:tc:xacml:1.1:policy-combining-algorithm:ordered-deny-overrides	M
urn:oasis:names:tc:xacml:1.1:rule-combining-algorithm:ordered-permit-overrides	M
urn:oasis:names:tc:xacml:1.1:policy-combining-algorithm:ordered-permit-overrides	M

7.8.4 Codes d'état

La prise en charge de l'élément <StatusCode> par l'implémentation est facultative, mais si l'élément est pris en charge, les codes d'état suivants doivent alors être pris en charge et doivent être utilisés de la façon spécifiée par XACML.

Identifiant	M/O
urn:oasis:names:tc:xacml:1.0:status:missing-attribute	M
urn:oasis:names:tc:xacml:1.0:status:ok	M
urn:oasis:names:tc:xacml:1.0:status:processing-error	M
urn:oasis:names:tc:xacml:1.0:status:syntax-error	M

7.8.5 Attributs

L'implémentation doit prendre en charge les attributs associés aux identifiants suivants, comme spécifié par XACML. Si les valeurs pour ces attributs ne sont pas présentes dans la demande de décision, leurs valeurs doivent alors être fournies par le gestionnaire de contexte. Donc, à la différence de la plupart des autres attributs, leur sémantique n'est pas transparente pour le PDP.

Identifiant	M/O
urn:oasis:names:tc:xacml:1.0:environment:current-time	M
urn:oasis:names:tc:xacml:1.0:environment:current-date	M
urn:oasis:names:tc:xacml:1.0:environment:current-dateTime	M

7.8.6 Identifiants

L'implémentation doit utiliser les attributs associés aux identifiants suivants de la façon définie par XACML. Cette exigence vise principalement les implémentations de PAP ou PEP qui utilisent XACML, dans la mesure où la sémantique des attributs est transparente pour le PDP.

Identifiant	M/O
urn:oasis:names:tc:xacml:1.0:subject:authn-locality:dns-name	O
urn:oasis:names:tc:xacml:1.0:subject:authn-locality:ip-address	O
urn:oasis:names:tc:xacml:1.0:subject:authentication-method	O
urn:oasis:names:tc:xacml:1.0:subject:authentication-time	O
urn:oasis:names:tc:xacml:1.0:subject:key-info	O
urn:oasis:names:tc:xacml:1.0:subject:request-time	O
urn:oasis:names:tc:xacml:1.0:subject:session-start-time	O
urn:oasis:names:tc:xacml:1.0:subject:subject-id	O
urn:oasis:names:tc:xacml:1.0:subject:subject-id-qualifier	O
urn:oasis:names:tc:xacml:1.0:subject-category:access-subject	M
urn:oasis:names:tc:xacml:1.0:subject-category:codebase	O
urn:oasis:names:tc:xacml:1.0:subject-category:intermediary-subject	O
urn:oasis:names:tc:xacml:1.0:subject-category:recipient-subject	O
urn:oasis:names:tc:xacml:1.0:subject-category:requesting-machine	O
urn:oasis:names:tc:xacml:1.0:resource:resource-location	O
urn:oasis:names:tc:xacml:1.0:resource:resource-id	M
urn:oasis:names:tc:xacml:1.0:resource:simple-file-name	O
urn:oasis:names:tc:xacml:1.0:action:action-id	O
urn:oasis:names:tc:xacml:1.0:action:implied-action	O

7.8.7 Types de données

L'implémentation doit prendre en charge les types de données associés aux identifiants suivants marqués "M".

Type de données	M/O
http://www.w3.org/2001/XMLSchema#string	M
http://www.w3.org/2001/XMLSchema#boolean	M
http://www.w3.org/2001/XMLSchema#integer	M
http://www.w3.org/2001/XMLSchema#double	M
http://www.w3.org/2001/XMLSchema#time	M
http://www.w3.org/2001/XMLSchema#date	M
http://www.w3.org/2001/XMLSchema#dateTime	M

Type de données	M/O
urn:oasis:names:tc:xacml:2.0:data-types:dayTimeDuration	M
urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration	M
http://www.w3.org/2001/XMLSchema#anyURI	M
http://www.w3.org/2001/XMLSchema#hexBinary	M
http://www.w3.org/2001/XMLSchema#base64Binary	M
urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name	M
urn:oasis:names:tc:xacml:1.0:data-type:x500Name	M

7.8.8 Fonctions

L'implémentation doit traiter correctement les fonctions associées aux identifiants marqués d'un "M".

Fonction	M/O
urn:oasis:names:tc:xacml:1.0:function:string-equal	M
urn:oasis:names:tc:xacml:1.0:function:boolean-equal	M
urn:oasis:names:tc:xacml:1.0:function:integer-equal	M
urn:oasis:names:tc:xacml:1.0:function:double-equal	M
urn:oasis:names:tc:xacml:1.0:function:date-equal	M
urn:oasis:names:tc:xacml:1.0:function:time-equal	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-equal	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-equal	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-equal	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-equal	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-equal	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-equal	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-equal	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-equal	M
urn:oasis:names:tc:xacml:1.0:function:integer-add	M
urn:oasis:names:tc:xacml:1.0:function:double-add	M
urn:oasis:names:tc:xacml:1.0:function:integer-subtract	M
urn:oasis:names:tc:xacml:1.0:function:double-subtract	M
urn:oasis:names:tc:xacml:1.0:function:integer-multiply	M
urn:oasis:names:tc:xacml:1.0:function:double-multiply	M
urn:oasis:names:tc:xacml:1.0:function:integer-divide	M
urn:oasis:names:tc:xacml:1.0:function:double-divide	M
urn:oasis:names:tc:xacml:1.0:function:integer-mod	M
urn:oasis:names:tc:xacml:1.0:function:integer-abs	M
urn:oasis:names:tc:xacml:1.0:function:double-abs	M
urn:oasis:names:tc:xacml:1.0:function:round	M
urn:oasis:names:tc:xacml:1.0:function:floor	M
urn:oasis:names:tc:xacml:1.0:function:string-normalize-space	M
urn:oasis:names:tc:xacml:1.0:function:string-normalize-to-lower-case	M
urn:oasis:names:tc:xacml:1.0:function:double-to-integer	M
urn:oasis:names:tc:xacml:1.0:function:integer-to-double	M
urn:oasis:names:tc:xacml:1.0:function:or	M
urn:oasis:names:tc:xacml:1.0:function:and	M
urn:oasis:names:tc:xacml:1.0:function:n-of	M
urn:oasis:names:tc:xacml:1.0:function:not	M
urn:oasis:names:tc:xacml:1.0:function:integer-greater-than	M
urn:oasis:names:tc:xacml:1.0:function:integer-greater-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:integer-less-than	M
urn:oasis:names:tc:xacml:1.0:function:integer-less-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:double-greater-than	M

Fonction	M/O
urn:oasis:names:tc:xacml:1.0:function:double-greater-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:double-less-than	M
urn:oasis:names:tc:xacml:1.0:function:double-less-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-add-dayTimeDuration	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-add-yearMonthDuration	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-subtract-dayTimeDuration	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-subtract-yearMonthDuration	M
urn:oasis:names:tc:xacml:1.0:function:date-add-yearMonthDuration	M
urn:oasis:names:tc:xacml:1.0:function:date-subtract-yearMonthDuration	M
urn:oasis:names:tc:xacml:1.0:function:string-greater-than	M
urn:oasis:names:tc:xacml:1.0:function:string-greater-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:string-less-than	M
urn:oasis:names:tc:xacml:1.0:function:string-less-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:time-greater-than	M
urn:oasis:names:tc:xacml:1.0:function:time-greater-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:time-less-than	M
urn:oasis:names:tc:xacml:1.0:function:time-less-than-or-equal	M
urn:oasis:names:tc:xacml:2.0:function:time-in-range	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-greater-than	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-greater-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-less-than	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-less-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:date-greater-than	M
urn:oasis:names:tc:xacml:1.0:function:date-greater-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:date-less-than	M
urn:oasis:names:tc:xacml:1.0:function:date-less-than-or-equal	M
urn:oasis:names:tc:xacml:1.0:function:string-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:string-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:string-is-in	M
urn:oasis:names:tc:xacml:1.0:function:string-bag	M
urn:oasis:names:tc:xacml:1.0:function:boolean-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:boolean-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:boolean-is-in	M
urn:oasis:names:tc:xacml:1.0:function:boolean-bag	M
urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:integer-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:integer-is-in	M
urn:oasis:names:tc:xacml:1.0:function:integer-bag	M
urn:oasis:names:tc:xacml:1.0:function:double-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:double-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:double-is-in	M
urn:oasis:names:tc:xacml:1.0:function:double-bag	M
urn:oasis:names:tc:xacml:1.0:function:time-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:time-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:time-is-in	M
urn:oasis:names:tc:xacml:1.0:function:time-bag	M
urn:oasis:names:tc:xacml:1.0:function:date-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:date-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:date-is-in	M
urn:oasis:names:tc:xacml:1.0:function:date-bag	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-bag-size	M

Fonction	M/O
urn:oasis:names:tc:xacml:1.0:function:dateTime-is-in	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-bag	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-is-in	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-bag	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-is-in	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-bag	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-is-in	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-bag	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-is-in	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-bag	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-is-in	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-bag	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-is-in	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-bag	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-one-and-only	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-bag-size	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-is-in	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-bag	M
urn:oasis:names:tc:xacml:2.0:function:string-concatenate	M
urn:oasis:names:tc:xacml:2.0:function:uri-string-concatenate	M
urn:oasis:names:tc:xacml:1.0:function:any-of	M
urn:oasis:names:tc:xacml:1.0:function:all-of	M
urn:oasis:names:tc:xacml:1.0:function:any-of-any	M
urn:oasis:names:tc:xacml:1.0:function:all-of-any	M
urn:oasis:names:tc:xacml:1.0:function:any-of-all	M
urn:oasis:names:tc:xacml:1.0:function:all-of-all	M
urn:oasis:names:tc:xacml:1.0:function:map	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-match	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-match	M
urn:oasis:names:tc:xacml:1.0:function:string-regexp-match	M
urn:oasis:names:tc:xacml:2.0:function:anyURI-regexp-match	M
urn:oasis:names:tc:xacml:2.0:function:ipAddress-regexp-match	M
urn:oasis:names:tc:xacml:2.0:function:dnsName-regexp-match	M
urn:oasis:names:tc:xacml:2.0:function:rfc822Name-regexp-match	M
urn:oasis:names:tc:xacml:2.0:function:x500Name-regexp-match	M
urn:oasis:names:tc:xacml:1.0:function:xpath-node-count	O
urn:oasis:names:tc:xacml:1.0:function:xpath-node-equal	O
urn:oasis:names:tc:xacml:1.0:function:xpath-node-match	O
urn:oasis:names:tc:xacml:1.0:function:string-intersection	M
urn:oasis:names:tc:xacml:1.0:function:string-at-least-one-member-of	M

Fonction	M/O
urn:oasis:names:tc:xacml:1.0:function:string-union	M
urn:oasis:names:tc:xacml:1.0:function:string-subset	M
urn:oasis:names:tc:xacml:1.0:function:string-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:boolean-intersection	M
urn:oasis:names:tc:xacml:1.0:function:boolean-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:boolean-union	M
urn:oasis:names:tc:xacml:1.0:function:boolean-subset	M
urn:oasis:names:tc:xacml:1.0:function:boolean-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:integer-intersection	M
urn:oasis:names:tc:xacml:1.0:function:integer-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:integer-union	M
urn:oasis:names:tc:xacml:1.0:function:integer-subset	M
urn:oasis:names:tc:xacml:1.0:function:integer-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:double-intersection	M
urn:oasis:names:tc:xacml:1.0:function:double-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:double-union	M
urn:oasis:names:tc:xacml:1.0:function:double-subset	M
urn:oasis:names:tc:xacml:1.0:function:double-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:time-intersection	M
urn:oasis:names:tc:xacml:1.0:function:time-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:time-union	M
urn:oasis:names:tc:xacml:1.0:function:time-subset	M
urn:oasis:names:tc:xacml:1.0:function:time-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:date-intersection	M
urn:oasis:names:tc:xacml:1.0:function:date-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:date-union	M
urn:oasis:names:tc:xacml:1.0:function:date-subset	M
urn:oasis:names:tc:xacml:1.0:function:date-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-intersection	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-union	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-subset	M
urn:oasis:names:tc:xacml:1.0:function:dateTime-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-intersection	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-union	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-subset	M
urn:oasis:names:tc:xacml:1.0:function:anyURI-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-intersection	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-union	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-subset	M
urn:oasis:names:tc:xacml:1.0:function:hexBinary-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-intersection	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-union	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-subset	M
urn:oasis:names:tc:xacml:1.0:function:base64Binary-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-intersection	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-union	M
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-subset	M

Fonction	M/O
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-intersection	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-union	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-subset	M
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-intersection	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-union	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-subset	M
urn:oasis:names:tc:xacml:1.0:function:x500Name-set-equals	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-intersection	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-at-least-one-member-of	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-union	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-subset	M
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-set-equals	M

8 Rôle central et hiérarchique fondé sur le profil de commande d'accès (RBAC)

Le présent paragraphe définit un profil à utiliser en XACML pour satisfaire aux exigences pour les rôles "core" et "hierarchical" fondés sur le contrôle d'accès (RBAC, *role based access control*).

8.1 Fondements de RBAC

Le présent paragraphe est pour information.

Le présent paragraphe définit un profil à utiliser avec XACML pour satisfaire aux exigences pour les rôles "core" et "hierarchical" fondés sur le contrôle d'accès (RBAC).

NOTE – Pour des informations sur RBAC, voir [RBAC].

8.1.1 Domaine d'application

Le rôle fondé sur le contrôle d'accès permet de spécifier les politiques en termes de rôles de sujet plutôt qu'en termes strictement fondés sur les identités de sujet individuelles. Ceci est important pour la modularité et la gestion des systèmes de contrôle d'accès.

Les politiques spécifiées dans ce profil peuvent répondre à trois types de questions:

1) si un sujet a les rôles R_1 , R_2, ... R_n activés, le sujet X peut-il accéder à une ressource donnée en utilisant une action donnée?
2) le sujet X est-il autorisé à avoir le rôle R_i activé?
3) si un sujet a les rôles R_1 , R_2, ... R_n activés, cela signifie-t-il que le sujet aura les permissions associées à un rôle R' donné? C'est-à-dire, le rôle R' est il égal ou *junior* par rapport à l'un des rôles R_1 , R_2, ... R_n?

Les politiques spécifiées dans ce profil ne répondent pas à la question "Quel ensemble de rôles le sujet X a-t-il?" Cette question doit être traitée par une autorité d'activation de rôle, et non traitée directement par un PDP XACML. Une telle entité peut utiliser les politiques XACML, mais aura besoin d'informations supplémentaires.

Les politiques spécifiées dans ce profil supposent que tous les rôles pour un sujet donné ont déjà été activés au moment où une décision d'autorisation est demandée. Elles ne s'occupent pas d'un environnement dans lequel les rôles doivent être activés de façon dynamique sur la base des ressources ou actions qu'un sujet essaye d'effectuer. Pour cette raison, les politiques spécifiées dans ce profil ne traitent pas non plus de "séparation des tâches" statique ou dynamique. Un profil futur pourra traiter les exigences de ce type d'environnement.

8.1.2 Rôle

Dans la présente Recommandation, les rôles sont exprimés comme des attributs de sujet XACML. Il y a deux exceptions: dans une allocation de rôle `<PolicySet>` ou `<Policy>` et dans une `<Policy> HasPrivilegesOfRole`, le rôle apparaît comme un attribut de ressource.

Les attributs de rôle peuvent être exprimés de deux façons possibles, selon les exigences de l'environnement d'application. Dans certains environnements, il peut y avoir un plus petit nombre d'"attributs de rôle", lorsque le nom de chacun de ces attributs indique "role", et lorsque la valeur de chacun de ces attributs indique le nom du rôle tenu. Par exemple, dans le premier type d'environnement, il peut y avoir un "attribut de rôle" qui a pour `AttributeId` "`&role;`" (le présent profil recommande l'utilisation de cet identifiant). Les rôles possibles sont des valeurs pour cet attribut unique, et peuvent être "`&roles;officer`", "`&roles;manager`", et "`&roles;employee`". Cette façon d'exprimer les rôles fonctionne le mieux avec la façon qu'a XACML d'exprimer les politiques. Cette méthode d'identification des rôles est aussi la plus favorable à l'interopérabilité.

Autrement, dans des environnements d'application différents, il peut y avoir un certain nombre d'identifiants d'attribut différents, chacun indiquant un rôle distinct. Par exemple, dans ce second type d'environnement, il peut y avoir trois identifiants d'attribut: "`urn:someapp:attributes:officer-role`", "`urn:someapp:attributes:manager-role`", et "`urn:someapp:attributes:employee-role`". Dans ce cas, la valeur de l'attribut peut être vide ou elle peut contenir divers paramètres associés au rôle. Les politiques XACML peuvent tenir des rôles exprimés de cette façon, mais pas aussi naturellement que dans le premier type.

XACML prend en charge plusieurs sujets par demande d'accès, indiquant diverses entités qui peuvent être impliquées dans la formulation de la demande. Par exemple, il y a habituellement un utilisateur humain qui initialise la demande, au moins indirectement. Il y a habituellement une ou plusieurs applications ou bases de code qui génèrent la demande d'accès de niveau inférieur réelle au nom de l'utilisateur. Il y a un appareil de calcul sur lequel s'exécute l'application ou la base de code, et cet appareil peut avoir une identité, telle qu'une adresse IP. XACML identifie chacun de `Subject` ces sujets avec un attribut xml `SubjectCategory` qui indique le type de sujet décrit. Par exemple, l'utilisateur humain a un `SubjectCategory` de `&subject-category;access-subject` (c'est la catégorie par défaut); l'application qui génère la demande d'accès a un `SubjectCategory` de `&subject-category;codebase` et ainsi de suite. Dans ce profil, un attribut de rôle peut être associé à toutes les catégories de sujets impliqués dans la formulation d'une demande d'accès.

8.1.3 Politiques

La présente Recommandation spécifie quatre types de politiques.

1) **Rôle** `<PolicySet>` ou **RPS**: c'est un `<PolicySet>` qui associe les détenteurs d'un attribut et valeur d'un rôle donné à une permission `<PolicySet>` qui contient les permissions réelles associées à ce rôle. L'élément `<Target>` d'un Rôle `<PolicySet>` limite l'applicabilité du `<PolicySet>` aux sujets qui détiennent l'attribut et la valeur du rôle associé. Chaque Rôle `<PolicySet>` fait référence à une seule permission `<PolicySet>` correspondante mais ne contient ou ne fait référence à aucun autre élément `<Policy>` ou `<PolicySet>`.

2) **Permission** `<PolicySet>` ou **PPS**: c'est un `<PolicySet>` qui contient les permissions réelles associées à un rôle donné. Il contient des éléments `<Policy>` et `<Rules>` qui décrivent les ressources et actions auxquelles les sujets sont autorisés à accéder, ainsi que toutes autres conditions posées à cet accès, comme en matière d'horaire. Une permission `<PolicySet>` donnée peut aussi contenir des références à des permissions `<PolicySet>` associées à d'autres rôles qui sont juniors pour ce rôle, permettant ainsi à cette permission `<PolicySet>` d'hériter de toutes les permissions associées au rôle de la permission `<PolicySet>` référencée. L'élément `<Target>` d'une permission `<PolicySet>`, s'il est présent, ne doit pas limiter les sujets auxquels le `<PolicySet>` est applicable.

3) **Allocation de rôle** `<Policy>` ou `<PolicySet>`: c'est une `<Policy>` ou `<PolicySet>` qui définit quels rôles peuvent être activés ou alloués et à quels sujets. Elle peut aussi spécifier des restrictions sur les combinaisons de rôles ou sur le nombre total de rôles alloués ou activés pour un sujet donné. Ce type de politique est utilisé par une autorité d'activation de rôle. L'utilisation d'une allocation de rôle `<Policy>` ou `<PolicySet>` est facultative.

4) **HasPrivilegesOfRole** `<Policy>`: c'est une `<Policy>` dans une permission `<PolicySet>` qui prend en charge les demandes en posant la question de savoir si un sujet dispose des privilèges associés à un rôle donné. Si ce type de demande est pris en charge, un `HasPrivilegesOfRole` `<Policy>` doit alors être inclus dans chaque permission `<PolicySet>`. La prise en charge de ce type de `<Policy>`, et donc de la question de savoir si un sujet a les privilèges associés à un rôle donné, est facultative.

Les instances de Permission `<PolicySet>` doivent être mémorisées dans le réceptacle de politiques de telle façon qu'elles ne puissent jamais être utilisées comme politique initiale pour un PDP XACML. Les instances de Permission `<PolicySet>` doivent être joignables uniquement à travers le Rôle `<PolicySet>` correspondant. Afin de prendre en charge les rôles hiérarchiques, une permission `<PolicySet>` doit être applicable à chaque sujet. La permission `<PolicySet>` dépend de son Rôle `<PolicySet>` correspondant pour s'assurer que seuls les sujets détenteurs de l'attribut de rôle correspondant vont obtenir l'accès aux permissions dans cette permission `<PolicySet>`.

L'utilisation d'instances séparées de Rôle <PolicySet> et Permission <PolicySet> permet la prise en charge du RBAC hiérarchique, dans lequel un rôle plus *senior* peut acquérir les permissions d'un rôle plus *junior*. Une permission <PolicySet> qui ne fait pas référence à d'autres éléments Permission <PolicySet> pourrait en fait être une <Policy> XACML plutôt qu'un <PolicySet>. Exiger que ce soit un <PolicySet>, permet cependant à son rôle associé de faire partie d'une hiérarchie de rôles ultérieure sans avoir besoin d'aucun changement dans les autres politiques.

8.1.4 Permissions multirôle

Dans ce profil, il est possible d'exprimer des politiques où un utilisateur doit détenir simultanément plusieurs rôles afin d'obtenir l'accès à certaines permissions. Par exemple, changer les instructions de soins pour le patient d'un hôpital peut requérir que le Subject qui effectue l'action ait à la fois les rôles de *médecin* et de *personnel*.

Ces politiques peuvent être exprimées en utilisant un Rôle <PolicySet> où l'élément <Target> exige que le sujet ait tous les attributs de rôle nécessaires. Cela est fait en utilisant un seul élément <Subject> contenant plusieurs éléments <SubjectMatch>. La Permission <PolicySet> associée devrait spécifier les permissions associées aux sujets qui ont simultanément tous les rôles spécifiés activés.

La Permission <PolicySet> associée à une politique multirôle peut faire référence à des instances de Permission <PolicySet> associées à d'autres rôles, et elle peut donc hériter de permissions provenant d'autres rôles. Les permissions associées à un <PolicySet> multirôle donné peuvent aussi être héritées par un autre rôle si l'autre rôle inclut une référence à la Permission <PolicySet> associée à la politique multirôle dans sa propre Permission <PolicySet>.

8.2 Exemple de RBAC

Le présent paragraphe est pour information.

Le présent paragraphe présente un exemple complet des types de politiques associées à des rôles fondés sur le contrôle d'accès.

Supposons qu'une organisation utilise deux rôles, gestionnaire et employé. Dans cet exemple, ils sont exprimés comme deux valeurs séparées pour un seul attribut XACML avec l'AttributeId "&role;". Les valeurs d'attribut &role; correspondant aux deux rôles sont "&roles;employee" et "&roles;manager". Un employé a la permission de créer un ordre d'achat. Un gestionnaire a la permission de signer un ordre d'achat, plus toutes les permissions associées au rôle d'un employé. Le rôle de gestionnaire est donc senior au rôle d'employé, et le rôle d'employé est junior au rôle de gestionnaire.

Conformément à ce profil, il y aura deux instances de permission <PolicySet>: une pour le rôle de gestionnaire et une pour le rôle d'employé. La permission <PolicySet> du gestionnaire donnera à tout sujet la permission spécifique de signer un ordre d'achat et fera référence à la permission <PolicySet> de l'employé afin d'hériter de ses permissions. La permission <PolicySet> de l'employé donnera à tout sujet la permission de créer un ordre d'achat.

Conformément à ce profil, il y aura aussi deux instances de rôle <PolicySet>: une pour le rôle de gestionnaire et une pour le rôle d'employé. Le rôle <PolicySet> du gestionnaire contiendra une <Target> exigeant que le Subject détienne un attribut &role; avec une valeur de "&roles;manager". Il fera référence à la permission <PolicySet> du gestionnaire. Le rôle <PolicySet> de l'employé contiendra une <Target> exigeant que le Subject détienne un attribut &role; avec une valeur de "&roles;employee". Il fera référence à la permission <PolicySet> de l'employé.

8.2.1 Permission <PolicySet> pour le rôle de gestionnaire

La permission <PolicySet> suivante contient les permissions associées au rôle de gestionnaire. La restitution de la politique du PDP doit être établie de telle sorte que l'accès à ce <PolicySet> ne soit obtenu que par référence au rôle <PolicySet> du gestionnaire (voir au Tableau 8-1).

Tableau 8-1/X.1142 – Permission <PolicySet> pour les gestionnaires

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicySetId="PPS:manager:role"
PolicyCombiningAlgId="&policy-combine;permit-overrides">

<!-- Permissions specifically for the manager role -->
<Policy PolicyId="Permissions:specifically:for:the:manager:role"
    RuleCombiningAlgId="&rule-combine;permit-overrides">
    <!-- Permission to sign a purchase order -->
    <Rule RuleId="Permission:to:sign:a:purchase:order" Effect="Permit">
                <Target>
                      <Resources>
                            <Resource>
                                  <ResourceMatch MatchId="&function;string-equal">
                                     <AttributeValue
                                         DataType="&xml;string">purchase
order</AttributeValue>
                                  <ResourceAttributeDesignator
AttributeId="&resource;resource-id"
                                         DataType="&xml;string"/>
                            </ResourceMatch>
                            </Resource>
                      </Resources>
                      <Actions>
                            <Action>
                                  <ActionMatch MatchId="&function;string-equal">
                                    <AttributeValue
                                       DataType="&xml;string">sign</AttributeValue>
                                       ActionAttributeDesignator
                                       AttributeId="&action;action-id"
                                       DataType="&xml;string"/>
                                  </ActionMatch>
                            </Action>
                      </Actions>
                </Target>
          </Rule>
</Policy>
<!-- Include permissions associated with employee role -->
<PolicySetIdReference>PPS:employee:role</PolicySetIdReference>
</PolicySet>
```

8.2.2 Permission <PolicySet> pour le rôle de l'employé

La permission `<PolicySet>` suivante contient les permissions associées au rôle d'employé (voir au Tableau 8-2). La restitution de la politique du PDP doit être établie de telle sorte que l'accès à ce `<PolicySet>` ne soit obtenu que par référence au rôle `<PolicySet>` de l'employé ou par référence provenant du rôle `<PolicySet>` de gestionnaire le plus senior via la permission `<PolicySet>` du gestionnaire.

Tableau 8-2/X.1142 – Permission <PolicySet> pour les employés

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
   PolicySetId="PPS:employee:role"
   PolicyCombiningAlgId="&policy-combine;permit-overrides">
   <!-- Permissions specifically for the employee role -->
   <Policy PolicyId="Permissions:specifically:for:the:employee:role"
          RuleCombiningAlgId="&rule-combine;permit-overrides">
   <!-- Permission to create a purchase order -->
          <Rule RuleId="Permission:to:create:a:purchase:order" Effect="Permit">
                 <Target>
                        <Resources>
                               <Resource>
                                      <ResourceMatch MatchId="&function;string-equal">
                                        <AttributeValue
                                        DataType="&xml;string">purchase
order</AttributeValue>
                                        <ResourceAttributeDesignator
AttributeId="&resource;resource-id"
                                        DataType="&xml;string"/>
                                      </ResourceMatch>
                               </Resource>
                        </Resources>
                        <Actions>
                               <Action>
                                      <ActionMatch MatchId="&function;string-equal">
                                        <AttributeValue
DataType="&xml;string">create</AttributeValue>
                                        <ActionAttributeDesignator
AttributeId="&action;action-id"
                                        DataType="&xml;string"/>
                                      </ActionMatch>
                               </Action>
                        </Actions>
                 </Target>
          </Rule>
   </Policy>
</PolicySet>
```

8.2.3 Rôle <PolicySet> pour le rôle de gestionnaire

Le rôle `<PolicySet>` suivant n'est applicable, conformément à sa `<Target>`, qu'aux sujets qui détiennent un attribut `&role;` (voir au Tableau 8-3) avec une valeur de "`&roles;manager`". Le `<PolicySetIdReference>` pointe sur la Permission `<PolicySet>` associée au rôle de gestionnaire.

Tableau 8-3/X.1142 – Rôle <PolicySet> pour les gestionnaires

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
   PolicySetId="RPS:manager:role"
   PolicyCombiningAlgId="&policy-combine;permit-overrides">
   <Target>
          <Subjects>
                 <Subject>
                        <SubjectMatch MatchId="&function;anyURI-equal">
                               <AttributeValue
DataType="&xml;anyURI">&roles;manager</AttributeValue>
                               <SubjectAttributeDesignator AttributeId="&role;"
                                      DataType="&xml;anyURI"/>
                        </SubjectMatch>
                 </Subject>
          </Subjects>
   </Target>
<!-- Use permissions associated with the manager role -->
   <PolicySetIdReference>PPS:manager:role</PolicySetIdReference>
</PolicySet>
```

8.2.4 Rôle <PolicySet> pour le rôle *employé*

Le rôle <PolicySet> suivant n'est applicable, conformément à sa <Target>, qu'aux sujets qui détiennent un attribut &role; (voir au Tableau 8-4) avec une valeur de "&roles;employee". Le <PolicySetIdReference> pointe sur la permission <PolicySet> associée au rôle d'employé.

Tableau 8-4/X.1142 – Rôle <PolicySet> pour les employés

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
   PolicySetId="RPS:employee:role"
   PolicyCombiningAlgId="&policy-combine;permit-overrides">
   <Target>
          <Subjects>
                <Subject>
                      <SubjectMatch MatchId="&function;anyURI-equal">
                            <AttributeValue
DataType="&xml;anyURI">&roles;employee</AttributeValue>
                            <SubjectAttributeDesignator AttributeId="&role;"
DataType="&xml;anyURI"/>
                      </SubjectMatch>
                </Subject>
          </Subjects>
   </Target>
<!-- Use permissions associated with the employee role -->
<PolicySetIdReference>PPS:employee:role</PolicySetIdReference>
</PolicySet>
```

8.2.5 Politiques et demandes HasPrivilegesOfRole

Un système RBAC XACML peut choisir de prendre en charge les interrogations de la forme "Ce sujet a-t-il les privilèges du rôle X?" S'il en est ainsi, chaque permission <PolicySet> doit contenir un <Policy> HasPrivilegesOfRole. Pour la permission <PolicySet> de gestionnaire, le <Policy> HasPrivilegesOfRole ressemblerait au Tableau 8-5.

Tableau 8-5/X.1142 – Permission <PolicySet> pour gestionnaires

```
<!-- HasPrivilegesOfRole Policy for manager role -->
<Policy PolicyId="Permission:to:have:manager:role:permissions"
   RuleCombiningAlgId="&rule-combine;permit-overrides">
<!-- Permission to have manager role permissions -->
   <Rule RuleId="Permission:to:have:manager:permissions" Effect="Permit">
          <Condition>
                <Apply FunctionId="&function;and">
                      <Apply FunctionId="&function;anyURI-is-in">
                            <AttributeValue
DataType="&xml;anyURI">&roles;manager</AttributeValue>
                            <ResourceAttributeDesignator AttributeId="&role;"
DataType="&xml;anyURI"/>
                      </Apply>
                            <Apply FunctionId="&function;anyURI-is-in">
                            <AttributeValue

   DataType="&xml;anyURI">&actions;hasPrivilegesofRole</AttributeValue>
                            <ActionAttributeDesignator AttributeId="&action;action-
id"
                                  DataType="&xml;anyURI"/>
                      </Apply>
                </Apply>
          </Condition>
   </Rule>
</Policy>
```

Pour la permission <PolicySet> pour employés, la <Policy> HasPrivilegesOfRole ressemblerait au Tableau 8-6.

Tableau 8-6/X.1142 – Permission <PolicySet> pour employés

```
<!-- HasPrivilegesOfRole Policy for employee role -->
<Policy PolicyId="Permission:to:have:employee:role:permissions"
   RuleCombiningAlgId="&rule-combine;permit-overrides">
<!-- Permission to have employee role permissions -->
   <Rule RuleId="Permission:to:have:employee:permissions" Effect="Permit">
          <Condition>
                <Apply FunctionId="&function;and">
                      <Apply FunctionId="&function;anyURI-is-in">
                            <AttributeValue
DataType="&xml;anyURI">&roles;employee</AttributeValue>
                            <ResourceAttributeDesignator AttributeId="&role;"
                                  DataType="&xml;anyURI"/>
                      </Apply>
                      <Apply FunctionId="&function;anyURI-is-in">
                      <AttributeValue
DataType="&xml;anyURI">&actions;hasPrivilegesofRole
                      </AttributeValue>
                      <ActionAttributeDesignator AttributeId="&action;action-id"
                            DataType="&xml;anyURI"/>
                      </Apply>
                </Apply>
          </Condition>
   </Rule>
</Policy>
```

Une demande de savoir si le sujet Anne a les privilèges associés à `&roles;manager` ressemblerait au Tableau 8-7.

Tableau 8-7/X.1142 – Demande portant sur un sujet

```
<Request>
   <Subject>
          <Attribute AttributeId="&subject;subject-id" DataType="&xml;string">
                <AttributeValue>Anne</AttributeValue>
          </Attribute>
   </Subject>
   <Resource>
          <Attribute AttributeId="&role;" DataType="&xml;anyURI">
                <AttributeValue>&roles;manager</AttributeValue>
                </Attribute>
   </Resource>
   <Action>
          <Attribute AttributeId="&action;action-id"

   DataType="&xml;anyURI">&actions;hasPrivilegesOfRole</AttributeValue>
          </Attribute>
   </Action>
</Request>
```

La `<Request>` doit contenir les rôles directs de Anne (dans ce cas, `&roles;employee`), ou alors le gestionnaire de contexte du PDP doit être capable de les découvrir. Les politiques `HasPrivilegesOfRole` n'effectuent pas l'association des rôles aux sujets.

8.3 Attributs d'allocation et d'activation de rôle

Le présent paragraphe est pour information.

L'allocation de divers attributs de rôle aux utilisateurs et l'activation de ces attributs au sein d'une session sont en dehors du domaine d'application du PDP XACML. Il doit y avoir une ou plusieurs entités séparées, se rapportant aux autorités d'activation de rôle, qui sont mises en œuvre pour effectuer ces fonctions. Le présent profil suppose que la présence d'un attribut de rôle dans le contexte d'une demande XACML pour un utilisateur donné (`Subject`) est une allocation valide au moment de la demande de décision d'accès.

D'où viennent les attributs de rôle d'un sujet? A quoi ressemble une de ces autorités d'activation de rôle? La réponse dépend de l'implémentation, mais on peut suggérer quelques possibilités.

Dans certains cas, les attributs de rôle pourraient venir d'un service de gestion d'identités qui tient à jour les informations sur un utilisateur, y compris les rôles alloués ou permis au sujet; le service de gestion d'identités joue le rôle d'autorité d'activation de rôle. Ce service peut mémoriser les attributs de rôle statiques dans un répertoire LDAP, et

un gestionnaire de contexte de PDP peut les en restituer. Ou bien ce service peut répondre aux demandes sur les attributs de rôle d'un sujet de la part d'un gestionnaire de contexte d'un PDP, et ces demandes sont alors sous la forme d'interrogations d'attributs SAML.

Les autorités d'activation de rôle peuvent utiliser un `<Policy>` ou `<PolicySet>` d'allocation de rôle XACML pour déterminer si un sujet a l'autorisation d'avoir l'activation d'un attribut et valeur de rôle particulier. Un `<Policy>` ou `<PolicySet>` d'allocation de rôle répond à la question "Le sujet X est-il autorisé à avoir le rôle R_i activé?" Il ne répond pas à la question "A l'activation de quel ensemble de rôles le sujet X est-il autorisé?" L'autorité d'activation de rôle doit avoir quelque moyen de savoir pour quel ou quels rôles soumettre une question. Par exemple, l'autorité d'activation de rôle peut tenir à jour une liste de tous les rôles possibles, et, lorsqu'elle est interrogée sur les rôles associés à un sujet donné, faire une demande sur les politiques d'allocation de rôle pour chaque rôle candidat.

Dans le présent profil, les politiques d'allocation de rôle sont un ensemble différent des instances de Rôle `<PolicySet>` et Permission `<PolicySet>` utilisées pour déterminer les permissions d'accès associées à chaque rôle. Les politiques d'allocation de rôle ne sont utilisées que lorsque la demande XACML provient d'une autorité d'activation de rôle. Cette séparation peut être gérée de diverses façons, telles que l'utilisation de PDP différents avec des mémoires de politique différentes ou exigeant des éléments `<Request>` pour les interrogations d'activation de rôle qu'ils incluent un `<Subject>` avec une `SubjectCategory` de "`&subject-category;role-enablement-authority`".

Il n'y a pas de forme fixée pour une `<Policy>` d'allocation de rôle. L'exemple suivant (Tableau 8-8) illustre une des formes possibles. Il contient deux éléments `<Rule>` XACML. La première `<Rule>` déclare que Anne et Seth et Yassir sont autorisés à avoir les rôles "`&roles;employee`" activés entre 9 h et 17 h. La seconde `<Rule>` déclare que Steve est autorisé à avoir l'activation des rôles "`&roles;manager`", sans restriction d'horaire.

Tableau 8-8/X.1142 – Exemple d'allocation de rôle

```
<Policy xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
    PolicyId="Role:Assignment:Policy"
    RuleCombiningAlgId="&rule-combine;permit-overrides">
<!-- Employee role requirements rule -->
    <Rule RuleId="employee:role:requirements" Effect="Permit">
        <Target>
            <Subjects>
                <Subject>
                    <SubjectMatch MatchId="&function;string-equal">
                        <AttributeValue DataType="&xml;string">Seth</AttributeValue>
                        <SubjectAttributeDesignator AttributeId="&subject;subject-id"
                            DataType="&xml;string"/>
                    </SubjectMatch>
                </Subject>
                <Subject>
                    <SubjectMatch MatchId="&function;string-equal">
                        <AttributeValue DataType="&xml;string">Anne</AttributeValue>
                        <SubjectAttributeDesignator AttributeId="&subject;subject-id"
                            DataType="&xml;string"/>
                    </SubjectMatch>
                </Subject>
            </Subjects>
        <Resources>
            <Resource>
                <ResourceMatch MatchId="&function;anyURI-equal">
                    <AttributeValue
DataType="&xml;anyURI">&roles;employee</AttributeValue>
                        <ResourceAttributeDesignator AttributeId="&role;"
                            DataType="&xml;anyURI"/>
                    </ResourceMatch>
                    </Resource>
                </Resources>
                <Actions>
                <Action>
                    <ActionMatch MatchId="&function;anyURI-equal">
                        <AttributeValue DataType="&xml;anyURI">&actions;
                            enableRole</AttributeValue>
                        <ActionAttributeDesignator AttributeId="&action;action-id"
                            DataType="&xml;anyURI"/>
                    </ActionMatch>
                </Action>
            </Actions>
```

Tableau 8-8/X.1142 – Exemple d'allocation de rôle

```
        </Target>
        <Condition>
            <Apply FunctionId="&function;and">
                <Apply FunctionId="&function;time-greater-than-or-equal">
                    <Apply FunctionId="&function;time-one-and-only">
                        <EnvironmentAttributeDesignator
AttributeId="&environment;current-time"
                            DataType="&xml;time"/>
                    </Apply>
                    <AttributeValue DataType="&xml;time">9h</AttributeValue>
                </Apply>
                <Apply FunctionId="&function;time-less-than-or-equal">
                    <Apply FunctionId="&function;time-one-and-only">
                        <EnvironmentAttributeDesignator
                            AttributeId="&environment;current-time"
DataType="&xml;time"/>
                    </Apply>
                    <AttributeValueDataType="&xml;time">17h</AttributeValue>
                </Apply>
        </Apply>
    </Condition>
</Rule>
<!-- Manager role requirements rule -->
    <Rule RuleId="manager:role:requirements" Effect="Permit">
        <Target>
            <Subjects>
                <Subject>
                    <SubjectMatch MatchId="&function;string-equal">
                        <AttributeValue DataType="&xml;string">Steve</AttributeValue>
                        <SubjectAttributeDesignator AttributeId="&subject;subject-id"
                            DataType="&xml;string"/>
                    </SubjectMatch>
                </Subject>
            </Subjects>
            <Resources>
                <Resource>
                    <ResourceMatch MatchId="&function;anyURI-equal">
                        <AttributeValue
                            DataType="&xml;anyURI">&roles;:manager</AttributeValue>
                        <ResourceAttributeDesignator AttributeId="&role;"
                            DataType="&xml;anyURI"/>
                    </ResourceMatch>
                </Resource>
            </Resources>
        <Actions>
            <Action>
                <ActionMatch MatchId="&function;anyURI-equal">
                    <AttributeValue
                        DataType="&xml;anyURI">&actions;enableRole</AttributeValue>
                        <ActionAttributeDesignator AttributeId="&action;action-id"
                            DataType="&xml;anyURI"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
</Rule>
</Policy>
```

8.4 Implémentation du modèle RBAC

Le présent paragraphe est pour information.

Les paragraphes suivants décrivent comment utiliser les politiques XACML pour implémenter divers composants du modèle RBAC (voir [RBAC]).

8.4.1 RBAC central

Le RBAC central comporte les cinq éléments de données de base suivants:

- les utilisateurs sont implémentés en utilisant des sujets XACML. Toute valeur de `SubjectCategory` XACML peut être utilisée, en tant que de besoin;
- les rôles sont exprimés en utilisant un ou plusieurs attributs de sujet XACML. L'ensemble des rôles est très spécifique du domaine d'application et de politique, et il est très important de ne pas confondre des utilisations de rôles différentes. Pour cette raison, le présent profil n'essaye pas de définir d'ensemble standard de valeurs de rôle, bien qu'il recommande effectivement l'utilisation d'une valeur commune d'`AttributeId` de "`urn:oasis:names:tc:xacml:2.0:subject:role`". Il est recommandé que chaque domaine d'application ou de politique se mette d'accord sur un unique ensemble de valeurs d'`AttributeId`, de `DataType`, et de `<AttributeValue>` qui seront utilisées pour les divers rôles pertinents pour ce domaine, et les publie;
- les objets sont exprimés en utilisant les `ressources` XACML;
- les opérations sont exprimées en utilisant les `Actions` XACML;
- les permissions sont exprimées en utilisant des instances de Rôle `<PolicySet>` et Permission `<PolicySet>` XACML comme décrit dans les paragraphes précédents.

Le RBAC central exige la prise en charge d'utilisateurs multiples par rôle, de rôles multiples par utilisateur, de permissions multiples par rôle, et de rôles multiples par permission. Chacune de ces exigences peut être satisfaite comme suit par les politiques XACML sur la base du présent profil. Noter, cependant, que l'allocation réelle des rôles aux utilisateurs sort du domaine d'application du PDP XACML.

XACML permet à plusieurs sujets d'être associés à un attribut de rôle donné. Les rôles `<PolicySet>` XACML définis en termes de possession d'un `<Attribute>` et `<AttributeValue>` de rôle particulier vont s'appliquer à tout utilisateur demandeur pour lequel ces `<Attribute>` et `<AttributeValue>` de rôle sont dans le contexte de demande XACML.

XACML permet à plusieurs attributs de rôle ou valeurs d'attribut de rôle d'être associés à un `Subject` donné. Si un `Subject` a plusieurs rôles activés, toute instance de rôle `<PolicySet>` s'appliquant à l'un de ces rôles peut alors être évaluée, et les permissions dans la permission `<PolicySet>` correspondante seront autorisées. Il est même possible de définir des politiques qui exigent d'un `Subject` donné qu'il ait plusieurs attributs ou valeurs de rôle activés en même temps. Dans ce cas, les permissions associées à des exigences de rôle multiple ne s'appliqueront qu'à un `Subject` ayant tous les attributs et valeurs de rôle nécessaires au moment où un contexte de demande XACML est présenté au PDP pour évaluation.

La permission `<PolicySet>` associée à un rôle donné peut permettre l'accès à des ressources multiples en utilisant plusieurs actions. XACML dispose d'un riche ensemble de constructions pour composer des permissions, de sorte qu'il y a de nombreuses façons d'exprimer des rôles multipermission. Tout rôle A peut être associé à une permission `<PolicySet>` B en incluant un `<PolicySetIdReference>` à la permission `<PolicySet>` B dans la permission `<PolicySet>` associée au rôle A. De cette façon, le même ensemble de permissions peut être associé à plus d'un rôle.

En plus des exigences RBAC centrales, les politiques XACML qui utilisent ce profil peuvent aussi exprimer des conditions arbitraires sur l'application de permissions particulières associées à un rôle. De telles conditions peuvent inclure de limiter les permissions à une période de la journée, ou de limiter les permissions aux détenteurs de rôle qui possèdent quelque autre attribut, que ce soit ou non un attribut de rôle.

8.4.2 RBAC hiérarchique

Le RBAC hiérarchique étend le RBAC central par la capacité à définir des relations d'héritage entre rôles. Par exemple, le rôle A peut être défini comme héritant de toutes les permissions associées au rôle B. Dans ce cas, le rôle A est considéré comme senior par rapport au rôle B dans la hiérarchie de rôle. Si le rôle B hérite à son tour de permissions associées au rôle C, le rôle A héritera alors de ces permissions du fait qu'il est senior par rapport au rôle B.

Les politiques XACML qui utilisent le présent profil peuvent implémenter l'héritage de rôle en incluant une `<PolicySetIdReference>` à la permission `<PolicySet>` associée à un rôle à l'intérieur de la permission `<PolicySet>` associée à un autre rôle. Le rôle qui inclut la `<PolicySetIdReference>` héritera alors des permissions associées au rôle référencé.

Ce profil structure les politiques de telle façon que les propriétés d'héritage peuvent s'ajouter à un rôle à tout moment sans exiger de changement aux instances de `<PolicySet>` associées à tout autre rôle. Une organisation peut ne pas utiliser au départ les hiérarchies de rôles, puis décider ultérieurement d'utiliser cette fonctionnalité sans avoir à réécrire les politiques existantes.

8.5 Profil

Le présent paragraphe discute des rôles, des attributs de rôle, de l'allocation de rôle et du contrôle d'accès.

8.5.1 Rôles et attributs de rôle

Les rôles doivent être exprimés en utilisant un ou plusieurs attributs XACML. Chaque domaine d'application qui utilise le présent profil pour le contrôle d'accès fondé sur le rôle doit définir ou se mettre d'accord sur une ou plusieurs valeurs d'`AttributeId` *(identifiant d'attribut)* à utiliser pour les attributs de rôle. Chaque valeur d'`AttributeId` ainsi définie doit être associée à un ensemble de valeurs permises et leurs `DataTypes` *(types de données)*. Chaque valeur permise pour un tel `AttributeId` doit avoir une sémantique bien définie à utiliser avec la valeur correspondante dans les politiques.

Le présent profil recommande d'utiliser la valeur d'`AttributeId` "`urn:oasis:names:tc:xacml:2.0:subject:role`" pour tous les attributs de rôle. Les instances de cet attribut devraient avoir un type de données de "`http://www.w3.org/2001/XMLSchema#anyURI`".

8.5.2 Allocation ou activation de rôle

Une autorité d'activation de rôle, responsable de l'allocation des rôles aux utilisateurs et de l'activation des rôles pour leur utilisation au sein d'une session d'utilisateur, peut utiliser une `<Policy>` ou `<PolicySet>` d'allocation de rôle XACML pour déterminer quels utilisateurs sont autorisés à activer quels rôles et sous quelles conditions. Il n'y a pas de forme obligée pour une `<Policy>` ou `<PolicySet>` d'allocation de rôle. Il est recommandé que les rôles soient exprimés en attributs de ressource dans une `<Policy>` ou `<PolicySet>` d'allocation de rôle, où l'`AttributeId` est `&role;` et la `<AttributeValue>` est l'URI pour la valeur de rôle pertinente. Il est recommandé que l'action d'allouer ou d'activer un rôle soit exprimée par un Attribut d'action, où l'`AttributeId` est `&action;action-id`, le `DataType` est `&xml;anyURI`, et la `<AttributeValue>` est `&actions;enableRole`.

8.5.3 Contrôle d'accès

Le contrôle d'accès fondé sur le rôle doit être implémenté en utilisant deux types de `<PolicySet>`: rôle `<PolicySet>`, permission `<PolicySet>`. Les fonctions et exigences spécifiques de ces deux types de `<PolicySet>` sont les suivantes.

Pour chaque rôle, un rôle `<PolicySet>` doit être défini. Un tel `<PolicySet>` doit contenir un élément `<Target>` qui rende le `<PolicySet>` applicable seulement aux sujets qui ont l'attribut XACML associé à ce rôle; l'élément `<Target>` ne doit pas restreindre la `Resource`, l'`Action`, ou l'`Environment`. Chaque rôle `<PolicySet>` doit contenir un seul élément `<PolicySetIdReference>` qui fasse référence à l'unique permission `<PolicySet>` associée à ce rôle. Le rôle `<PolicySet>` ne doit contenir aucun autre élément `<Policy>`, `<PolicySet>`, `<PolicyIdReference>` ou `<PolicySetIdReference>`.

Pour chaque rôle, une permission `<PolicySet>` doit être définie. Ce `<PolicySet>` doit contenir des éléments `<Policy>` et `<Rule>` qui spécifient les types d'accès permis aux `Subjects` qui tiennent ce rôle. La `<Target>` du `<PolicySet>` et ses éléments `<PolicySet>`, `<Policy>` et `<Rule>` inclus ou référencés ne doivent pas limiter les `subjects` auxquels est applicable la permission `<PolicySet>`.

Si un rôle donné hérite des permissions d'un ou plusieurs rôles juniors, la permission `<PolicySet>` pour le rôle (senior) donné doit inclure un élément `<PolicySetIdReference>` pour chaque rôle junior. Chacun de ces `<PolicySetIdReference>` doit faire référence à la permission `<PolicySet>` associée au rôle junior duquel le rôle senior hérite.

Une permission `<PolicySet>` peut inclure une `<Policy>` `HasPrivilegesOfRole`. Une telle `<Policy>` doit avoir un élément `<Rule>` avec un effet de "Permit". Cette règle doit permettre à tout sujet d'effectuer une action avec un attribut ayant un `AttributeId` de `&action;action-id`, un `DataType` de `&xml;anyURI` et une `<AttributeValue>` ayant une valeur de `&actions;hasPrivilegesOfRole` sur une ressource ayant un attribut qui est le rôle auquel s'applique la permission `<PolicySet>` (par exemple, un `AttributeId` de `&rôle;`, un `DataType` de `&xml;anyURI` et une `<AttributeValue>` dont la valeur est l'URI de la valeur spécifique du rôle). Noter que l'attribut de rôle, qui est un attribut de sujet dans une `<Target>` de Rôle `<PolicySet>`, est traité comme un attribut de ressource dans une `<Policy>` `HasPrivilegesOfRole`.

L'organisation de tout réceptacle utilisé pour les politiques et la configuration du PDP doivent garantir que le PDP ne peut jamais utiliser une permission `<PolicySet>` comme politique initiale du PDP.

8.6 Identifiants

Le présent profil définit les identifiants d'URN suivants.

8.6.1 Identifiant de profil

L'identifiant suivant doit être utilisé comme identifiant pour ce profil lorsqu'un identifiant sous forme d'URI est nécessaire.

```
urn:oasis:names:tc:xacml:2.0:profiles:rbac:core-hierarchical
```

8.6.2 Attribut de rôle

L'identifiant suivant peut être utilisé comme `AttributeId` pour les attributs de rôle.

```
urn:oasis:names:tc:xacml:2.0:subject:role
```

8.6.3 SubjectCategory

L'identifiant suivant peut être utilisé comme `SubjectCategory` pour les attributs de sujet identifiant qu'une demande provient d'une autorité d'activation de rôle.

```
urn:oasis:names:tc:xacml:2.0:subject-category:role-enablement-authority
```

8.6.4 Valeurs d'attribut d'action

L'identifiant suivant peut être utilisé comme `<AttributeValue>` de l'attribut `&action;action-id` dans une `<Policy> HasPrivilegesOfRole`.

```
urn:oasis:names:tc:xacml:2.0:actions:hasPrivilegesOfRole
```

L'identifiant suivant peut être utilisé comme `<AttributeValue>` de l'attribut `&action;action-id` dans une `<Policy>` d'allocation de rôle.

```
urn:oasis:names:tc:xacml:2.0:actions:enableRole
```

9 Profil de ressources multiples de XACML

L'évaluation de politique effectuée par un point de décision de politique XACML, ou PDP, est défini en termes de ressource demandée unique, avec la décision d'autorisation contenue dans un seul élément `<Result>` du contexte de réponse. Un point de mise en application de politique, ou PEP, peut cependant souhaiter soumettre un seul contexte de demande pour l'accès à plusieurs ressources et souhaiter obtenir un seul contexte de réponse contenant une décision d'autorisation (élément `<Result>`) séparée pour chaque ressource demandée. Un tel contexte de demande pourrait être utilisé pour éviter d'envoyer plusieurs messages de demande de décision entre un PEP et un PDP, par exemple. Autrement, un PEP peut souhaiter soumettre un seul contexte de demande pour tous les nœuds dans une hiérarchie, et peut souhaiter obtenir une seule décision d'autorisation (élément `<Result>`) qui indique si l'accès est permis à tous les nœuds demandés. Un tel contexte de demande pourrait être utilisé lorsque le demandeur veut l'accès à tout un document XML, ou à tout un sous-arbre d'éléments dans un tel document, ou à un répertoire entier de système de fichiers avec tous ses sous-répertoires et fichiers, par exemple.

La présente Recommandation décrit trois façons par lesquelles un PEP peut demander des décisions d'autorisation pour des ressources multiples dans un seul contexte de demande, et comment le résultat de chacune de ces décisions d'autorisation est représenté dans le contexte de réponse unique qui est retourné au PEP.

La présente Recommandation décrit aussi deux façons qu'a un PEP de demander une seule décision d'autorisation en réponse à une demande pour tous les nœuds dans une hiérarchie.

La prise en charge de chacun des mécanismes décrits dans le présent profil est facultative pour les implémentations XACML conformes.

Les attributs de ressource d'utilisation courante sont abrégés comme suit:

– attribut "`resource-id`" – attribut de ressource avec un `AttributeId` de "`urn:oasis:names:tc:xacml:1.0:resource:resource-id`".

– attribut "`scope`" – attribut de ressource avec un `AttributeId` de "`urn:oasis:names:tc:xacml:2.0:resource:scope`".
Voir au § 9.3 des informations complémentaires sur cet attribut.

9.1 Demandes de ressources multiples

Le présent paragraphe est normatif, mais facultatif.

Un contexte de demande XACML unique peut représenter une demande d'accès à des ressources multiples, avec une décision d'autorisation séparée souhaitée pour chaque ressource. La syntaxe et la sémantique de telles demandes et réponses sont spécifiées dans ce paragraphe.

Les éléments `<Result>` produits en évaluant une demande d'accès à des ressources multiples doivent être identiques à ceux qui seraient produits par une série de demandes, chacune demandant l'accès à exactement une des ressources. Chacune de ces ressources est appelée une ressource individuelle. Le concept de contexte de demande qui correspond à chaque élément `<Result>` est appelé une demande de ressource individuelle. La valeur `ResourceId` dans l'élément `<Result>` doit être la `<AttributeValue>` de l'attribut "`resource-id`" dans la demande de ressource individuelle correspondante. Ce mappage de contexte de demande original contenant des demandes de décision d'autorisation multiples en demandes de ressource individuelle, et le mappage correspondant de décisions d'autorisation multiples en éléments `<Result>` multiples dans un seul contexte de réponse peut être effectué par le gestionnaire de contexte dans la présente Recommandation. Le présent profil n'exige pas que les implémentations de l'évaluation d'une demande d'accès à des ressources multiples soient conformes au modèle précédent ou que soient construites les demandes réelles de ressource individuelle. Le profil exige seulement que les éléments `<Result>` soient les mêmes que si le modèle précédent était utilisé.

Trois façons de spécifier les demandes d'accès à des ressources multiples sont décrites dans les paragraphes suivants. Chaque façon de spécifier les demandes décrit les demandes de ressources individuelles qui correspondent aux éléments `<Result>` dans le contexte de réponse.

Un contexte de demande XACML unique soumis par un PEP peut utiliser plus d'une de ces façons de demander l'accès à des ressources multiples dans des éléments `<Resource>` différents.

9.1.1 Nœuds identifiés par "scope"

Le présent paragraphe est normatif mais facultatif.

Le texte décrit l'utilisation de deux valeurs pour l'attribut de ressource "scope" pour spécifier une demande d'accès à des ressources multiples dans une hiérarchie. Cette syntaxe peut être utilisée avec toute ressource hiérarchique, que ce soit ou non un document XML:

9.1.1.1 URI de profil

Les URI suivants doivent être utilisés comme identifiants d'URI pour la fonctionnalité spécifiée dans cette partie du présent profil. Le premier identifiant doit être utilisé lorsque la fonctionnalité est prise en charge pour les ressources XML, et le second identifiant doit être utilisé lorsque la fonctionnalité est prise en charge pour des ressources qui ne sont pas des documents XML:

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:scope:xml
urn:oasis:names:tc:xacml:2.0:profile:multiple:scope:non-xml
```

9.1.1.2 Syntaxe de contexte de demande original

L'élément `<Resource>` de contexte de demande XACML original doit contenir un attribut "scope" avec une valeur de "Children" *(enfants)*, ou "Descendants". Si les ressources demandées sont dans un document XML, l'élément `<ResourceContent>` doit être présent et doit contenir tout le document XML dont les éléments demandés font partie. De plus, si les ressources demandées sont dans un document XML, l'expression XPath utilisée comme valeur de l'attribut "`resource-id`" doit s'évaluer comme un ensemble de nœuds contenant exactement un nœud.

9.1.1.3 Sémantique

Un tel contexte doit être interprété comme une demande d'accès à un ensemble de nœuds dans une hiérarchie relative au nœud unique spécifié dans l'attribut "`resource-id`". Si la valeur de l'attribut "`scope`" est "Children", chaque ressource individuelle est le nœud unique indiqué par l'attribut "`resource-id`" (ou les attributs, si la ressource unique a plusieurs identifiants normatifs) et tous ses nœuds fils immédiats. Si la valeur de l'attribut "`scope`" est "Descendants", la ressource individuelle est le nœud unique indiqué par l'attribut "`resource-id`" et tous ses nœuds descendants.

Chaque demande de ressource individuelle doit être identique au contexte de demande original avec deux exceptions: l'attribut "`scope`" ne doit pas être présent et l'élément `<Resource>` doit représenter une ressource individuelle unique. Cet élément `<Resource>` doit contenir au moins un attribut "`resource-id`", et toutes les valeurs pour de tels attributs doivent être des entités uniques, normatives de la ressource individuelle. Si l'attribut "`resource-id`" dans le contexte de la demande d'origine contenait un producteur, les attributs "`resource-id`" dans la demande de ressource

individuelle doivent contenir le même producteur. Si un élément <ResourceContent> était présent dans le contexte de la demande d'origine, le même élément <ResourceContent> doit alors être inclus dans chaque demande de ressource individuelle.

Ni XACML ni le présent profil ne spécifient comment le gestionnaire de contexte obtient les informations nécessaires pour déterminer quels nœuds sont fils ou descendants d'un nœud donné, excepté dans le cas d'un document XML, où les informations doivent être obtenues de l'élément <ResourceContent>.

9.1.2 Nœuds identifiés par XPath

Le présent paragraphe est normatif mais facultatif.

Le présent paragraphe décrit l'utilisation d'une expression XPath dans l'attribut "resource-id" attribute, conjointement avec une valeur "XPath-expression" dans l'attribut "scope" pour spécifier une demande d'accès à des nœuds multiples dans un document XML. Cette syntaxe ne doit être utilisée qu'avec des ressources qui sont des documents XML.

9.1.2.1 URI de profil

L'URI suivant doit être utilisé comme identifiant d'URI pour la fonctionnalité spécifiée dans cette partie du présent profil:

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:xpath-expression
```

9.1.2.2 Contexte de demande original

L'élément <Resource> de contexte de demande XACML original doit contenir un élément <ResourceContent> et un attribut "resource-id" avec un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression" tel que le <AttributeValue> de l'attribut "resource-id" soit une expression XPath qui s'évalue comme un ensemble de nœuds qui représentent des nœuds multiples dans l'élément <ResourceContent>. L'élément <Resource> doit contenir un attribut "scope" *(domaine d'application)* avec une valeur de "XPath-expression".

9.1.2.3 Sémantique

Un tel contexte de demande doit être interprété comme une demande d'accès aux nœuds multiples dans l'ensemble de nœuds représenté par la <AttributeValue> de l'attribut "resource-id". Chacun de ces nœuds doit représenter une ressource individuelle.

Chaque demande de ressource individuelle doit être identique au contexte de demande original avec deux exceptions: l'attribut "scope" ne doit pas être présent et la valeur d'attribut "resource-id" doit être une expression XPath qui s'évalue comme un seul nœud dans l'élément <ResourceContent>. Ce nœud doit être la ressource individuelle. Si l'attribut "resource-id" dans le contexte de la demande d'origine contenait un producteur, l'attribut "resource-id" dans la demande de ressource individuelle doit contenir le même producteur.

9.1.3 Eléments <Resource> multiples

Ce paragraphe est normatif mais facultatif.

Ce paragraphe décrit l'utilisation d'éléments <Resource> multiples dans un contexte de demande pour spécifier une demande d'accès à des ressources multiples. Cette syntaxe peut être utilisée avec toute ou toutes ressources, qu'elles soient ou non des documents XML et qu'elles soient ou non des ressources hiérarchiques.

9.1.3.1 URI de profil

L'URI suivant doit être utilisé comme identifiant d'URI pour la fonctionnalité spécifiée dans cette partie du présent profil:

```
    urn:oasis:names:tc:xacml:2.0:profile:multiple:multiple-resource-
elements
```

9.1.3.2 Contexte de demande original

Le contexte de demande XACML doit contenir des éléments <Resource> multiples.

9.1.3.3 Sémantique

Un tel contexte de demande doit être interprété comme une demande d'accès à toutes les ressources spécifiées dans les éléments <Resource> individuels. Chaque élément <Resource> doit représenter une ressource individuelle sauf si cet élément utilise les autres mécanismes décrits dans le présent profil.

Pour chaque élément <Resource>, une demande de ressource individuelle doit être créée. Cette demande de ressource individuelle doit être identique au contexte de demande original avec une exception: seul l'élément <Resource> unique doit être présent. Si un tel élément <Resource> contient un attribut "scope" ayant une valeur autre que "Immediate", la demande de ressource individuelle doit alors être traitée conformément à la partie correspondante du présent profil. Ce traitement peut impliquer de décomposer la demande de ressource individuelle unique en autres demandes de ressources individuelles avant l'évaluation par le PDP.

9.2 Demandes pour une hiérarchie entière

Le présent paragraphe est normatif mais facultatif.

Dans certains cas, une ressource est hiérarchique, mais la demande de décision d'autorisation est destinée à demander l'accès à tous les nœuds au sein de cette ressource ou à une sous-hiérarchie de nœuds complète au sein de cette ressource. Cela pourrait être le cas lorsque l'accès à un document XML est demandé pour les besoins de l'établissement d'une copie d'un document entier, ou lorsque l'accès à tout un répertoire de système de fichiers avec tous ses sous-répertoires est demandé. Un seul <Result> est désiré, indiquant si le demandeur a l'autorisation d'accès à tout l'ensemble des nœuds.

L'élément <Result> produit en évaluant une telle demande d'accès doit être identique à celui produit par le processus suivant. Une série de contextes de demande est évaluée, chacun demandant l'accès à exactement un nœud de la hiérarchie. La <Decision> dans le <Result> unique qui est retourné au PEP doit être "Permit" si et seulement si tous les éléments <Result> résultant de l'évaluation des nœuds individuels contenaient une <Decision> de "Permit". Autrement, la <Decision> dans le <Result> unique retourné au PEP doit être "Deny". Le présent profil n'exige pas que l'implémentation de l'évaluation d'une demande d'accès à une telle ressource hiérarchique se conforme au modèle précédent ou que les contextes de demande réels correspondant aux nœuds individuels dans la hiérarchie soient construits. Le présent profil exige seulement que l'élément <Result> soit le même que si le précédent modèle était utilisé.

Deux syntaxes sont spécifiées pour cette fonctionnalité dans les paragraphes qui suivent, une pour utilisation avec des ressources qui sont des documents XML, et l'autre pour utilisation avec des ressources qui ne sont pas des documents XML.

9.2.1 Ressources XML

Le présent paragraphe est normatif mais facultatif.

Le présent paragraphe décrit la syntaxe pour demander l'accès à un document XML entier, ou à un élément au sein de ce document avec tous ses sous-éléments récurrents.

9.2.1.1 URI de profil

L'URI suivant doit être utilisé comme l'identifiant pour la fonctionnalité spécifiée dans cette partie du profil:

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:entire-hierarchy:xml
```

9.2.1.2 Contexte de demande original

L'élément <Resource> dans le contexte de la demande d'origine doit contenir un attribut "scope" avec une valeur de "EntireHierarchy".

L'élément <Resource> dans le contexte de la demande d'origine doit contenir un attribut "resource-id" unique avec un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression", tel que la <AttributeValue> s'évalue comme un ensemble de nœuds qui représente exactement le nœud unique dans l'élément <ResourceContent>.

L'élément <Resource> dans le contexte de la demande d'origine peut contenir d'autres attributs.

9.2.1.3 Sémantique

Le <Result> d'une telle demande doit être équivalent à celui produit par le processus suivant. Pour chaque nœud dans la hiérarchie demandée, le gestionnaire de contexte doit créer un nouveau contexte de demande. Chacun de ces contextes de demande doit contenir un seul élément <Resource> ayant un attribut "resource-id" avec un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression" et une valeur qui soit une expression XPath qui s'évalue comme un ensemble de nœuds qui contienne exactement ce seul nœud dans l'élément <ResourceContent>. Le gestionnaire de contexte doit soumettre chacun de ces nouveaux contextes de demande au PDP pour évaluation et doit garder trace de la <Decision> dans les éléments <Result> correspondants. Si et seulement si tous les nouveaux contextes de demande s'évaluent comme "Permit", un seul <Result> contenant une <Decision> de

"Permit" doit alors être placé dans le contexte de réponse retourné au PEP. Si un des nouveaux contextes de demande s'évalue comme "Deny", "Indeterminate", ou "NotApplicable", un seul <Result> contenant une <Decision> de "Deny" doit alors être placé dans le contexte de réponse retourné au PEP.

9.2.2 Ressources non-XML

Le présent paragraphe est normatif mais facultatif.

Le présent paragraphe décrit la syntaxe pour demander l'accès à une hiérarchie entière de nœuds au sein d'une ressource hiérarchique qui n'est pas un document XML.

9.2.2.1 URI de profil

L'URI suivant doit être utilisé comme l'identifiant pour la fonctionnalité spécifiée dans cette partie du présent profil:

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:entire-hierarchy:non-xml
```

9.2.2.2 Contexte de demande original

L'élément élément <Resource> dans le contexte de la demande d'origine doit contenir un attribut "scope" avec une valeur de "EntireHierarchy".

L'élément <Resource> dans le contexte de la demande d'origine doit contenir un seul attribut "resource-id" qui représente un seul nœud dans une ressource hiérarchique.

L'élément <Resource> dans le contexte de la demande d'origine peut contenir d'autres attributs.

La représentation des nœuds dans une ressource hiérarchique spécifiée dans le profil XACML pour des ressources hiérarchiques dans la présente Recommandation peut être utilisée pour représenter l'identité du nœud unique.

9.2.2.3 Sémantique

Le <Result> d'une telle demande doit être équivalent à celui produit par les processus suivants. Pour chaque nœud dans la hiérarchie demandée, le gestionnaire de contexte doit créer un nouveau contexte de demande. Chacun de ces contextes de demande doit contenir un seul élément <Resource> ayant un attribut "resource-id" avec une valeur qui est l'identité de ce nœud unique dans la hiérarchie. Le gestionnaire de contexte doit soumettre chacun de ces nouveaux contextes de demande au PDP pour évaluation et doit garder trace de la <Decision> dans les éléments <Result> correspondants. Si et seulement si tous les nouveaux contextes de demande s'évaluent comme "Permit", un seul <Result> contenant une <Decision> de "Permit" doit alors être placé dans le contexte de réponse retourné au PEP. Si un des nouveaux contextes de demande s'évalue comme "Deny", "Indeterminate" ou "NotApplicable", un seul <Result> contenant une <Decision> de "Deny" doit alors être placé dans le contexte de réponse retourné au PEP.

Ni XACML ni le présent profil ne spécifient comment le gestionnaire de contexte obtient les informations nécessaires pour déterminer quels nœuds sont les descendants du nœud spécifié à l'origine, ou comment représenter l'identité de chacun de ces nœuds. La représentation des nœuds dans une ressource hiérarchique spécifiée dans le profil XACML pour les ressources hiérarchiques dans la présente Recommandation peut être utilisée pour représenter l'identité de chacun de ces nœuds.

9.3 Nouveaux identifiants d'attribut

L'identifiant suivant est utilisé comme AttributeId d'un attribut de ressource qui indique le domaine d'application (attribut "scope") d'une demande d'accès dans un seul élément <Resource> d'un contexte de demande.

```
urn:oasis:names:tc:xacml:2.0:resource:scope
```

L'attribut doit avoir un DataType de "http://www.w3.org/2001/XMLSchema#string".

La liste des valeurs valides pour cet attribut figure ci-dessous et au § 7.5. Une implémentation peut prendre en charge tout sous-ensemble de ces valeurs, y compris l'ensemble vide.

- "Immediate" – L'élément <Resource> se réfère à une seule ressource non hiérarchique ou à un seul nœud dans une ressource hiérarchique. C'est la valeur par défaut, si aucun attribut "scope" n'est présent. L'élément <Resource> doit être traité conformément au § 7.
- "Children" – L'élément <Resource> se réfère à des ressources multiples dans une hiérarchie. L'ensemble de ressources consiste en un seul nœud décrit par l'attribut de ressource "resource-id" et par tous les enfants immédiats de ce nœud dans la hiérarchie. L'élément <Resource> doit être traité conformément au § 9.1.1 du présent profil.

– "Descendants" – L'élément `<Resource>` se réfère à des ressources multiples dans une hiérarchie. L'ensemble de ressources consiste en un seul nœud décrit par l'attribut de ressource "`resource-id`" et tous les descendants de ce nœud dans la hiérarchie. L'élément `<Resource>` doit être traité conformément au § 9.1.1 du présent profil.

– "XPath-expression" – L'élément `<Resource>` se réfère à des ressources multiples. L'ensemble de ressources consiste en nœuds dans un ensemble de nœuds décrit par l'attribut de ressource "`resource-id`". Chacun de ces nœuds doit être contenu dans l'élément `<ResourceContent>` de l'élément `<Resource>`. L'élément `<Resource>` doit être traité conformément au § 9.1.2 du présent profil.

– "EntireHierarchy" – L'élément `<Resource>` se réfère à une seule ressource. La ressource consiste en un nœud décrit par l'attribut de ressource "resource-id" ainsi que par tous les descendants du nœud. Tous ces nœuds doivent être des nœuds dans un document XML qui est contenu dans l'élément `<ResourceContent>` de l'élément `<Resource>`. L'élément `<Resource>` doit être traité conformément au § 9.2.

9.4 Nouveaux identifiants de profil

Les valeurs d'URI suivantes doivent être utilisées comme identifiants d'URI pour la fonctionnalité spécifiée dans divers paragraphes de ce profil:

– attribut de domaine "`scope`" d'application de "children" ou "descendants" dans `<Resource>`: ressources XML

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:scope:xml
```

– attribut de domaine d'application de "children" ou "descendants" dans `<Resource>`: ressources non-XML

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:scope:non-xml
```

– expression XPath dans l'attribut "`resource-id`"

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:xpath-expression
```

– éléments multiples `<Resource>`

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:multiple-resource-
elements
```

– demandes d'une hiérarchie entière: ressources XML

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:entire-hierarchy:xml
```

– demandes d'une hiérarchie entière: ressources non-XML

```
urn:oasis:names:tc:xacml:2.0:profile:multiple:entire-hierarchy:non-
xml
```

10 Profil SAML 2.0 de XACML

Le présent paragraphe définit un profil sur la façon d'utiliser SAML 2.0 (voir la Rec. UIT-T X.1141) pour protéger, transporter et demander des instances de schéma XACML et d'autres informations nécessaires pour une implémentation XACML.

SAML est un cadre de travail fondé sur XML pour l'échange d'informations de sécurité. Ces informations de sécurité sont exprimées sous la forme d'assertions sur des sujets, où un sujet est une entité (homme ou machine) qui a une identité dans un certain domaine de sécurité. Une seule assertion pourrait contenir plusieurs déclarations internes différentes sur l'authentification, l'autorisation et les attributs. SAML définit un protocole par lequel les clients peuvent demander des assertions provenant d'autorités SAML et obtenir d'elles une réponse. Ce protocole, qui consiste en formats de messages de demande et de réponse fondés sur XML, peut se lier à de nombreux protocoles de communication et de transport sous-jacents différents; SAML définit actuellement une liaison avec SOAP sur HTTP. En créant leurs réponses, les autorités SAML peuvent utiliser diverses sources d'informations, telles que des mémoires externes de politique et d'assertions qui ont été reçues en entrée dans des demandes. SAML définit des éléments assertion, sujets, conditions, avis et déclarations.

Il y a six types d'interrogation et de déclaration qui sont utilisés dans le présent paragraphe:

1) `AttributeQuery`: demande SAML standard utilisée pour demander un ou plusieurs attributs à une autorité d'attribut;

2) `AttributeStatement`: déclaration SAML standard qui contient un ou plusieurs attributs. Cette déclaration peut être utilisée dans une réponse SAML de la part d'une autorité d'attribut, ou elle peut être utilisée dans une assertion SAML comme un format pour mémoriser des attributs dans un réceptacle d'attributs;

3) `XACMLPolicyQuery`: extension de demande SAML, définie dans le présent profil. Elle est utilisée pour demander une ou plusieurs politiques à un point d'administration de politique;

4) `XACMLPolicyStatement`: extension de déclaration SAML, définie dans le présent profil. Elle peut être utilisée dans une réponse SAML de la part d'un point d'administration de politique, ou elle peut être utilisée dans une assertion SAML comme un format pour mémoriser des politiques dans un réceptacle de politiques;

5) `XACMLAuthzDecisionQuery`: extension de demande SAML, définie dans le présent profil. Elle est utilisée par un PEP pour demander une décision d'autorisation de la part d'un PDP XACML;

6) `XACMLAuthzDecisionStatement`: extension de déclaration SAML, définie dans le présent profil. Elle peut être utilisée dans une réponse SAML de la part d'un PDP XACML. Elle peut aussi être utilisée dans une assertion SAML utilisée comme accréditif, mais ne fait pas actuellement partie du modèle d'utilisation XACML.

Le diagramme suivant (Figure 10-1) illustre le modèle d'utilisation XACML et les messages qui sont utilisés pour communiquer entre les divers composants. Tous les composants ne seront pas utilisés dans toutes les implémentations.

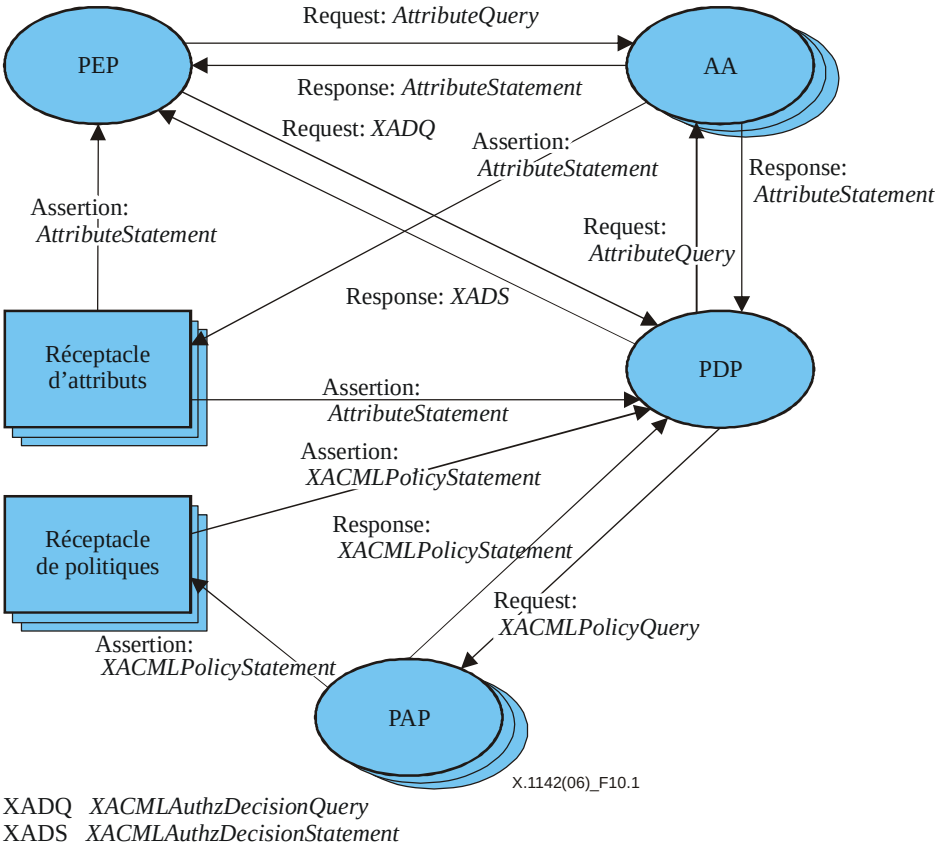

Figure 10-1/X.1142 – Modèle d'utilisation XACML

Le présent paragraphe décrit tous ces éléments de schéma d'interrogation et de déclaration, et comment les utiliser. Il décrit aussi certains autres aspects de l'utilisation de SAML avec XACML. La présente Recommandation n'exige aucun changement ou extension à XACML, mais définit des extensions à SAML.

Afin d'améliorer la lisibilité, les exemples dans ce profil supposent l'utilisation des déclarations d'entité internes XML suivantes:

```
^lt;!ENTITY saml "urn:oasis:names:tc:SAML:2.0:assertion"
^lt;!ENTITY samlp "urn:oasis:names:tc:SAML:2.0:protocol"
^lt;!ENTITY xacml "urn:oasis:names:tc:xacml:2.0:"
^lt;!ENTITY xacml-context "urn:oasis:names:tc:xacml:2.0:context:schema:os"
^lt;!ENTITY xml "http://www.w3.org/2001/XMLSchema#"
^lt;!ENTITY subject-id "urn:oasis:names:tc:xacml:1.0:subject:subject-id"
^lt;!ENTITY resource "urn:oasis:names:tc:xacml:1.0:resource:"
^lt;!ENTITY resource-id "urn:oasis:names:tc:xacml:1.0:resource:resource-id"
^lt;!ENTITY action-id "urn:oasis:names:tc:xacml:1.0:action:action-id"
^lt;!ENTITY environment "urn:oasis:names:tc:xacml:1.0:environment:"
^lt;!ENTITY current-dateTime
    "urn:oasis:names:tc:xacml:1.0:environment:current-dateTime"
```

Par exemple, "`&xml;#string`" est équivalent à http://www.w3.org/2001/XMLSchema#string. L'espace de nom associé au schéma XACML qui étend le schéma d'assertion SAML est

```
xacml-saml="urn:oasis:names:tc:xacml:2.0:saml:assertion:schema:os".
```

L'espace de nom associé au schéma XACML qui étend le schéma de protocole SAML est

```
xacml-samlp="urn:oasis:names:tc:xacml:2.0:saml:protocol:schema:os".
```

10.1 Mappage des attributs SAML et XACML

Le schéma d'assertions SAML définit une assertion d'attribut. Le schéma de protocole SAML définit un `AttributeQuery` utilisé pour demander des instances d'assertions d'attribut et une réponse contenant les instances requises. Les systèmes qui utilisent XACML peuvent utiliser des instances de ces attributs SAML de transmission et de mémorisation d'éléments SAML. Les systèmes utilisant XACML peuvent utiliser le protocole `AttributeQuery` SAML pour demander des instances d'attributs SAML. Pour être utilisé dans un contexte de demande XACML, l'attribut SAML doit être mappé dans un attribut XACML.

Une assertion d'attribut SAML est une instance `<saml:Assertion>` qui contient une ou plusieurs instances de `<saml:AttributeStatement>`, chacune d'elles pouvant contenir une ou plusieurs instances de `<saml:Attribute>`.

Pour être utilisé dans un contexte de demande XACML, chaque attribut SAML dans l'assertion d'attribut SAML doit se conformer au *profil d'attribut XACML*, espace de nom `urn:oasis:names:tc:SAML:2.0:profiles:attribute:XACML`, dans la Rec. UIT-T X.1141.

Un `<xacml-context:Attribute>` doit être construit comme suit à partir de l'élément `<saml:Attribute>` correspondant dans une assertion d'attribut SAML.

- Attribut XML d'`AttributeId` XACML
  - La valeur pleinement qualifiée de l'attribut XML `Name <saml:Attribute>` doit être utilisée.
- Attribut XML de `DataType` XACML
  - La valeur pleinement qualifiée de l'attribut XML `DataType <saml:Attribute>` doit être utilisée. Si l'attribut XML `DataType <saml:Attribute>` manque, l'attribut XML `DataType` XACML doit être http://www.w3.org/2001/XMLSchema#string.
- Attribut XML `Issuer` (de producteur) XACML
  - La valeur de chaîne de l'élément `<saml:Issuer>` provenant de l'assertion d'attribut SAML doit être utilisée.
- `<xacml-context:AttributeValue>`
  - La valeur `<saml:AttributeValue>` doit être utilisée comme valeur de l'élément `<xacml-context:AttributeValue>`.

Chaque instance de `<saml:Attribute>` est transposée en un seul élément `<xacml-context:Attribute>`. Toutes les instances de `<saml:Attribute>` dans une assertion d'attribut SAML n'ont pas besoin d'être mappées; les instances d'attributs SAML à mapper peuvent être sélectionnées par un mécanisme qui n'est pas spécifié ici. Le producteur de

l'élément <saml:Assertion> est utilisé comme producteur pour chaque élément <xacml-context:Attribute> créé.

Le <xacml-context:Attribute> créé à partir de <saml:Assertion> doit être placé dans l'élément <xacml-context:Resource>, <xacml-context:Subject>, <xacml-context:Action> ou <xacml-context:Environment> qui correspond à l'entité qui est le <saml:Subject> dans l'assertion d'attribut SAML. Par exemple, si le sujet d'assertion d'attribut SAML contient un élément <saml:NameIdentifier>, et si la valeur de ce NameIdentifier correspond à la valeur du <xacml-context:Attribute> qui a un AttributeId de &resource;resource-id, les instances de <xacml-context:Attribute> créées à partir des instances de <saml:Attribute> dans cette assertion d'attribut SAML doivent alors être placées dans l'élément <xacml-context:Resource>. Si le <xacml-context:Attribute> est placé dans un élément <xacml-context:Subject>, l'attribut XML SubjectCategory de XACML doit être aussi cohérent avec l'entité qui est le sujet de la <saml:Assertion>.

L'entité qui effectue le mappage doit s'assurer que la sémantique définie par SAML pour les éléments contenus dans <saml:Assertion> a été respectée. L'entité de mappage n'a pas besoin d'effectuer elle-même les vérifications de sémantique, mais elle doit s'assurer que les vérifications ont été faites avant que tout <xacml:Attribute> créé à partir de la <saml:Assertion> ne soit utilisé par un PDP XACML. Ces vérifications de sémantique incluent, sans s'y limiter, ce qui suit.

- Tout attribut NotBefore et NotOnOrAfter XML dans la <saml:Assertion> doit être valide par rapport à la <xacml:Request> dans laquelle le <xacml:Attribute> déduit de SAML est utilisé. Cela signifie que les valeurs d'attribut XML NotBefore et NotOnOrAfter doivent être cohérentes avec les valeurs de <xacml:Attribute> &environment;current-time, &environment;current-date et &environment:current-dateTime associées à la <xacml:Request>.
- L'entité qui effectue le mappage doit s'assurer que la sémantique définie par SAML pour tout élément <saml:AudienceRestrictionCondition> ou <saml:DoNotCacheCondition> a bien été respectée.
- Si un élément <ds:Signature> survient dans la <saml:Assertion>, l'entité qui effectue le mappage doit alors s'assurer que la signature est valide et que l'élément <Issuer> SAML est cohérent avec toute valeur <ds:X509IssuerName> dans la signature. Les lignes directrices concernant les signatures numériques dans la Rec. UIT-T X.1141 doivent être respectées.

10.2 Décision d'autorisation

SAML 2.0 définit une interrogation de décision d'autorisation AuthzDecisionQuery rudimentaire (voir la Rec. UIT-T X.1141). Une AuthzDecisionQuery SAML n'est pas capable de convoyer toutes les informations qu'un PDP XACML est capable d'accepter au titre de son contexte de demande. De même, la déclaration AuthzDecisionStatement SAML est incapable de convoyer toutes les informations contenues dans un contexte de réponse XACML.

Pour permettre à un PEP d'utiliser la syntaxe de demande et réponse SAML avec une complète prise en charge de la syntaxe de contexte de demande et de contexte de réponse XACML, la présente Recommandation définit deux extensions SAML:

- <xacml-samlp:XACMLAuthzDecisionQuery> est une interrogation SAML qui étend le schéma de protocole SAML (voir la Rec. UIT-T X.1141). Elle permet à un PEP de soumettre un contexte de demande XACML dans une demande SAML, conjointement à d'autres informations.
- <xacml-saml:XACMLAuthzDecisionStatement> est une déclaration SAML qui étend le schéma d'assertions SAML (voir la Rec. UIT-T X.1141). Elle permet à un PDP XACML de retourner un contexte de réponse XACML dans la réponse à une déclaration <XACMLAuthzDecisionStatement>, conjointement à d'autres informations. Elle permet aussi qu'un contexte de réponse XACML soir mémorisé ou transmis sous la forme d'une assertion SAML.

10.2.1 Elément <XACMLAuthzDecisionQuery>

L'élément <XACMLAuthzDecisionQuery> peut être utilisé par un PEP pour demander une décision d'autorisation de la part d'un PDP XACML. Il permet à une demande SAML de convoyer une instance XACML de contexte de demande.

```
<xs:element name="XACMLAuthzDecisionQuery" type="XACMLAuthzDecisionQueryType"/>
<xs:complexType name="XACMLAuthzDecisionQueryType">
  <xs:complexContent>
   <xs:extension base="samlp:RequestAbstractType">
    <xs:sequence>
        <xs:element ref="xacml-context:Request"/>
    </xs:sequence>
    <xs:attribute name="InputContextOnly" type="boolean" use="optional" default="false"/>
    <xs:attribute name="ReturnContext" type="boolean" use="optional" default="false"/>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

L'élément <XACMLAuthzDecisionQuery> est du type complexe **XACMLAuthzDecisionQueryType.** Cet élément offre une alternative à l'interrogation <samlp:AuthzDecisionQuery> définie par SAML qui permet à un PEP d'utiliser les pleines capacités d'un PDP XACML.

L'élément <XACMLAuthzDecisionQuery> contient les attributs et éléments XML suivants:

– InputContextOnly ["false" par défaut]

  Cet attribut XML gouverne les sources d'information que le PDP est autorisé à utiliser pour prendre sa décision d'autorisation. Si cet attribut XML est "true", la décision d'autorisation doit alors être prise seulement sur la base des informations contenues dans l'interrogation <XACMLAuthzDecisionQuery>; aucun attribut externe ne peut être utilisé. Si cet attribut XML est "false", la décision d'autorisation peut alors être prise sur la base d'attributs externes non contenus dans <XACMLAuthzDecisionQuery>.

– ReturnContext ["false" par défaut]

  Cet attribut XML permet au PEP de demander qu'un élément <xacml-context:Request> soit inclus dans la déclaration <XACMLAuthzDecisionStatement> résultant de la demande. Il commande aussi l'élément <xacml-context:Request>.

  Si cet attribut XML est "true", le PDP doit alors inclure l'élément <xacml-context:Request> dans l'élément <XACMLAuthzDecisionStatement> dans la <XACMLResponse>. Cet élément <xacml-context:Request> doit inclure tous ces attributs fournis par le PEP dans l'interrogation <XACMLAuthzDecisionQuery> qui a été utilisée dans la prise de décision d'autorisation. Le PDP peut inclure des attributs supplémentaires dans cet élément <xacml-context:Request>, tels que des attributs externes obtenus par le PDP et utilisés dans la prise de décision d'autorisation, ou d'autres attributs connus du PDP pour être utiles à la formulation par le PEP de demandes <XACMLAuthzDecisionQuery> ultérieures.

  Si cet attribut XML est "false", le PDP ne doit alors pas inclure l'élément <xacml-context:Request> dans l'élément <XACMLAuthzDecisionStatement> de la <XACMLResponse>.

– <xacml-context:Request> [Exigé]

  Un contexte de demande XACML.

10.2.2 Elément <XACMLAuthzDecisionStatement>

Le <XACMLAuthzDecisionStatement> peut être utilisé par un PDP XACML pour retourner une réponse SAML contenant un contexte de réponse XACML à un PEP en réponse à une <XACMLAuthzDecisionQuery>. Il peut aussi être utilisé dans une assertion SAML comme un format pour la mémorisation d'une décision d'autorisation dans un réceptacle.

```
<xs:element name="XACMLAuthzDecisionStatement" type="xacml-
saml:XACMLAuthzDecisionStatementType"/>
<xs:complexType name="XACMLAuthzDecisionStatementType">
  <xs:complexContent>
     <xs:extension base="saml:StatementAbstractType">
        <xs:sequence>
           <xs:element ref="xacml-context:Response"/>
              <xs:element ref="xacml-context:Request" MinOccurs="0"/>
        </xs:sequence>
     </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

L'élément <XACMLAuthzDecisionStatement> est du type complexe **XACMLAuthzDecisionStatementType**. Cet élément est une solution de remplacement de la déclaration <samlp:AuthzDecisionStatement> définie par SAML qui permet à une assertion SAML de comporter le contenu complet de la réponse provenant d'un PDP XACML.

L'élément <XACMLAuthzDecisionStatement> contient les éléments suivants:

– <xacml-context:Response> [Exigé]

  Le contexte de réponse XACML créé par le PDP XACML en réponse à la <XACMLAuthzDecisionQuery>.

– <xacml-context:Request> [Facultatif]

  Demande <xacml-context:Request> contenant les attributs XACML retournés par le PDP XACML en réponse à l'interrogation <XACMLAuthzDecisionQuery>. Cet élément doit être inclus si l'attribut XML ReturnResponse dans l'interrogation <XACMLAuthzDecisionQuery> est "true". Cet élément ne doit pas être inclus si l'attribut XML ReturnResponse dans l'interrogation <XACMLAuthzDecisionQuery> est "false".

10.3 Politiques

XACML définit deux éléments de schéma de politique: <Policy> et <PolicySet>. SAML ne définit aucun schéma de protocole ou d'assertion pour les politiques. Le présent paragraphe définit de nouvelles extensions SAML pour les éléments <XACMLPolicyQuery> et <XACMLPolicyStatement>. Des instances de ces nouveaux éléments peuvent être utilisées pour demander, transmettre, et mémoriser les instances de <Policy> et <PolicySet> XACML.

10.3.1 Elément <XACMLPolicyQuery>

L'élément <XACMLPolicyQuery> est utilisé par un PDP pour demander une ou plusieurs instances de politique ou ensemble de politiques XACML de la part d'un point d'administration de politique en ligne au titre d'une demande SAML.

```
<xs:element name="XACMLPolicyQuery" type="XACMLPolicyQueryType"/>
<xs:complexType name="XACMLPolicyQueryType">
   <complexContent>
         <xs:extension base="samlp:RequestAbstractType">
               <xs:choice minOccurs="0" maxOccurs="unbounded">
                     <xs:element ref="xacml-context:Request"/>
                     <xs:element ref="xacml:Target"/>
                     <xs:element ref="xacml:PolicySetIdReference"/>
                     <xs:element ref="xacml:PolicyIdReference"/>
               </xs:choice>
         </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <XACMLPolicyQuery> est du type complexe **XACMLPolicyQueryType**.

L'élément <XACMLPolicyQuery> contient un ou plusieurs des éléments suivants:

– <xacml-context:Request> [Tout Nombre]

  Fournit un contexte de demande XACML. Toutes les instances de politique et PolicySet d'ensemble de politique XACML applicables à cette demande doivent être retournées.

– <xacml:Target> [Tout Nombre]

  Fournit un élément <Target> XACML. Toutes les instances de politique et PolicySet d'ensemble de politique XACML applicables à cette <Target> doivent être retournées.

– <xacml:PolicySetIdReference> [Tout Nombre]

  Identifie un <PolicySet> XACML à retourner.

– <xacml:PolicyIdReference> [Tout Nombre]

  Identifie un <Policy> XACML à retourner.

10.3.2 Elément <XACMLPolicyStatement>

Le <XACMLPolicyStatement> est utilisé par un point d'administration de politique pour retourner une ou plusieurs instances de <Policy> ou <PolicySet> XACML dans une réponse SAML à une demande <XACMLPolicyQuery> SAML. La déclaration <XACMLPolicyStatement> peut aussi être utilisée dans une assertion SAML comme format pour mémoriser la déclaration <XACMLPolicyStatement> dans un réceptacle.

```
<xs:element name="XACMLPolicyStatement" type="xacml-
saml:XACMLPolicyStatementType"/>
<xs:complexType name="XACMLPolicyStatementType">
   <xs:complexContent>
          <xs:extension base="saml:StatementAbstractType">
                <xs:choice minOccurs="0" maxOccurs=unbounded">
                      <xs:element ref="xacml:Policy"/>
                      <xs:element ref="xacmlPolicySet"/>
                </xs:choice>
                </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

L'élément <XACMLPolicyStatement> est du type complexe **XACMLPolicyStatementType**.

L'élément <XACMLPolicyStatement> contient les éléments suivants. Si la <XACMLPolicyStatement> est produite en réponse à une interrogation <XACMLPolicyQuery>, et s'il n'y a pas d'autre instance de <xacml:Policy> ou <xacml:PolicySet> qui satisfasse aux spécifications de la <XACMLPolicyQuery> associée, il ne doit y avoir aucun élément dans la <XACMLPolicyStatement>.

– <xacml:Policy> [Tout Nombre]

  Instance de <xacml:Policy> qui satisfait aux spécifications de la <XACMLPolicyQuery> associée, s'il en est.

– <xacml:PolicySet> [Tout Nombre]

  Instance de <xacml:PolicySet> qui satisfait aux spécifications de la <XACMLPolicyQuery> associée, s'il en est.

10.4 Elément <saml:Assertion>

Une déclaration <XACMLAuthzDecisionStatement>, <XACMLPolicyStatement> ou <saml:AttributeStatement> standard SAML doit être encapsulée dans un élément <saml:Assertion> qui peut être signé.

La plupart des composants d'une <saml:Assertion> sont pleinement spécifiés dans la Rec. UIT-T X.1141. Les éléments et attributs XML suivants sont spécifiés plus en détail ici pour leur utilisation avec les types de déclaration SAML définis et utilisés dans le présent profil.

Excepté comme spécifié ici, le présent profil n'impose aucune exigence ou restriction sur les informations contenues dans l'élément <saml:Assertion>.

10.4.1 Elément <saml:Issuer>

L'élément <saml:Issuer> est un élément exigé pour détenir des informations sur "l'autorité SAML qui effectue la ou les revendications dans l'assertion".

Pour prendre en charge les signatures numériques de tierce partie, le présent profil n'exige pas que l'identité fournie dans l'élément <saml:Issuer> soit cohérente avec l'identité du signataire. Il appartient au consommateur d'assertions d'avoir une relation de confiance appropriée avec l'autorité qui signe la <saml:Assertion>.

Lorsqu'une <saml:AttributeAssertion> est utilisée pour construire un attribut XACML, la valeur de chaîne de l'élément <saml:Issuer> sera utilisée comme valeur de l'attribut XML de producteur XACML, de sorte que la valeur SAML devrait être spécifiée en gardant ce point en mémoire.

10.4.2 Elément <ds:Signature>

L'élément <ds:Signature> est un élément facultatif pour détenir "une signature XML qui authentifie l'assertion".

Un élément <ds:Signature> peut être utilisé dans une assertion joint à une déclaration XACML. Pour prendre en charge les signatures numériques de tierce partie, le présent profil n'exige pas que l'identité fournie dans l'élément <saml:Issuer> soit cohérente avec l'identité du signataire. Il appartient au consommateur d'assertions d'avoir une relation de confiance appropriée avec l'autorité qui signe la <saml:Assertion>.

Un consommateur d'assertions devrait vérifier toutes les signatures incluses dans l'assertion et ne devrait pas utiliser d'informations déduites de l'assertion tant que la signature n'a pas été vérifiée avec succès.

10.4.3 Elément <saml:Subject>

L'élément <saml:Subject> est un élément facultatif utilisé pour détenir "le sujet de la ou des déclarations dans l'assertion".

L'élément <saml:Subject> ne doit pas être inclus dans une assertion qui contient une <XACMLAuthzDecision> ou <XACMLPolicy>.

Dans une <saml:AttributeAssertion> qui doit être transposée en attribut XACML, l'élément <saml:Subject> doit contenir l'identité de l'entité à laquelle sont liés l'attribut et sa valeur. Pour un attribut XACML <Subject>, cette identité devrait être cohérente avec la valeur de tout attribut &subject-id; XACML qui survient dans le même élément <Subject>.

Pour un attribut XACML <Resource>, cette identité devrait être cohérente avec la valeur de tout attribut &resource-id; XACML qui survient dans le même élément <Resource>. Pour un attribut <Action> XACML, cette identité devrait être cohérente avec la valeur de tout attribut &action-id; XACML survenant dans le même élément <Action>. Pour un attribut XACML <Environment>, cette identité devrait être cohérente avec la valeur de tout attribut XACML survenant dans le même élément <Environment> et qui fournit une identité d'environnement.

10.4.4 Elément <saml:Conditions>

L'élément <saml:Conditions> est un élément facultatif qui est utilisé pour des "conditions qui doivent être prises en compte pour attester de la validité de l'assertion et/ou l'utiliser".

L'élément <saml:Conditions> devrait contenir les attributs XML NotBefore et NotOnOrAfter pour spécifier les limites de la validité de l'assertion. Si ces attributs XML sont présents, le consommateur d'assertions devrait s'assurer que les informations déduites de l'assertion ne sont utilisées par un PDP pour évaluer des politiques que lorsque la valeur de l'attribut de ressource &current-dateTime; du contexte de la demande est contenu dans la période de validité spécifiée pour l'assertion.

10.5 Elément <samlp:RequestAbstractType>

Une interrogation <XACMLAuthzDecisionQuery> ou <XACMLPolicyQuery> doit être encapsulée dans un élément <samlp:RequestAbstractType> qui peut être signé.

La plupart des composants d'un <samlp:RequestAbstractType> sont pleinement spécifiés dans la Rec. UIT-T X.1141. Pour être utilisés avec les types d'interrogation SAML définis et utilisés dans le présent profil, l'élément <saml:Issuer> et l'élément <ds:Signature> doivent être utilisés de la même manière que spécifié au paragraphe précédent. Excepté comme spécifié ici, le présent profil n'impose aucune exigence ni restriction sur les informations contenues dans l'élément <samlp:RequestAbstractType>.

10.5.1 Elément <saml:Issuer>

Voir au paragraphe 10.4.1 Elément <saml:Issuer>.

10.5.2 Elément <ds:Signature>

Voir au paragraphe 10.4.2 Elément <ds:Signature>.

10.6 Elément <samlp:Response>

Une déclaration <XACMLAuthzDecisionStatement> ou <XACMLPolicyStatement> doit être encapsulée dans un élément <samlp:Response> qui peut être signé.

La plupart des composants d'une <samlp:Response> sont pleinement spécifiés dans la Rec. UIT-T X.1141. Les éléments et attributs XML suivants sont respécifiés ici pour être utilisés avec les types de déclaration SAML définis et utilisés dans le présent profil. Excepté comme spécifié ici, le présent profil n'impose aucune exigence ou restriction sur les informations contenues dans l'élément <samlp:Response>.

10.6.1 Elément <samlp:Issuer>

Voir au paragraphe 10.4.1 Elément <saml:Issuer>.

10.6.2 Elément <ds:Signature>

Voir au paragraphe 10.4.2 Elément <ds:Signature>.

10.6.3 Elément <samlp:StatusCode>

L'élément `<samlp:StatusCode>` est un composant de l'élément `<samlp:Status>` dans la `<samlp:Response>`.

10.6.3.1 Réponse à <XACMLAuthzDecisionQuery>

Dans la réponse à une demande `<XACMLAuthzDecisionQuery>`, l'attribut XML de valeur `<samlp:StatusCode>` doit dépendre comme suit de l'élément `<xacml:StatusCode>` de l'élément `<xacml:Status>` de la décision d'autorisation:

1) `urn:oasis:names:tc:SAML:2.0:status:Success`

   Cette valeur pour l'attribut XML de valeur `<samlp:StatusCode>` doit être utilisée si et seulement si la valeur `<xacml:StatusCode>` est `urn:oasis:names:tc:xacml:1.0:status:ok`.

2) `urn:oasis:names:tc:SAML:2.0:status:Requester`

   Cette valeur pour l'attribut XML de valeur `<samlp:StatusCode>` doit être utilisée lorsque la valeur `<xacml:StatusCode>` est `urn:oasis:names:tc:xacml:1.0:status:missing-attribute` ou lorsque la valeur `<xacml:StatusCode>` est `urn:oasis:names:tc:xacml:1.0:status:syntax-error` du fait d'une erreur de syntaxe dans la demande `<xacml:Request>`.

3) `urn:oasis:names:tc:SAML:2.0:status:Responder`

   Cette valeur pour l'attribut XML de valeur `<samlp:StatusCode>` doit être utilisée lorsque la valeur `<xacml:StatusCode>` est `urn:oasis:names:tc:xacml:1.0:status:syntax-error` du fait d'une erreur de syntaxe dans une `<xacml:Policy>` ou `<xacml:PolicySet>`. Noter que toutes les erreurs de syntaxe dans les politiques ne seront pas détectées en conjonction avec le traitement d'une interrogation particulière, et donc toutes les erreurs de syntaxe de politique ne seront pas rapportées de cette façon.

4) `urn:oasis:names:tc:SAML:2.0:status:VersionMismatch`

   Cette valeur pour l'attribut XML de valeur `<samlp:StatusCode>` ne doit être utilisée que lorsque l'interface SAML au PDP ne prend pas en charge la version du message de demande SAML utilisée dans l'interrogation.

10.6.3.2 Réponse à < XACMLPolicyQuery >

Dans la réponse à une demande `<XACMLPolicyQuery>`, l'attribut XML de valeur `<samlp:StatusCode>` doit être comme spécifié dans la Rec. UIT-T X.1141.

11 Profil XML de signature numérique

Le présent paragraphe fournit un profil à utiliser avec Signature:2002 du W3C pour fournir l'authentification et la protection de l'intégrité pour les instances de schéma XACML.

Une signature numérique n'est utile pour l'authentification et la protection de l'intégrité que si les informations signées incluent une spécification de l'identité du signataire et une spécification de la période durant laquelle l'objet de données signé doit être considéré comme valide. XACML ne définit pas lui-même le format de telles informations, car XACML est destiné à utiliser d'autres normes pour les fonctions autres que la spécification et l'évaluation réelle des politiques, demandes et réponses de contrôle d'accès.

Un format approprié a été défini dans SAML. Un profil pour l'utilisation de SAML avec des instances de schéma XACML est défini au § 10. Le présent profil recommande donc l'utilisation des instances de schéma XACML dans les assertions, demandes, et réponses SAML, qui peuvent être signées numériquement comme spécifié dans la Rec. UIT-T X.1141.

11.1 Utilisation de SAML

Le présent profil recommande l'utilisation des instances de schéma XACML enchâssées dans les assertions, demandes, et réponses SAML, comme décrit au § 10. De tels objets SAML doivent être signés numériquement comme décrit au § 8.4/X.1141, *SAML et syntaxe et traitement de signature XML*.

11.2 Canonisation

Afin qu'une signature numérique soit vérifiée par un consommateur d'assertions, le flux d'octets qui a été signé doit être identique au flux d'octets qui est vérifié. Pour garantir cela, le document XML à signer doit être canonisé (voir

Canonicalization:2002 du W3C). La Rec. UIT-T X.1141 spécifie l'utilisation de la canonisation exclusive (voir Canonicalization:2002 du W3C).

11.2.1 Eléments d'espace de nom dans les objets de données XACML

Tout objet de données XACML qui est à signer doit spécifier tous les éléments d'espace de nom utilisés dans l'objet de données. Si cela n'est pas fait, l'objet de données va alors attirer les définitions d'espace de nom des ancêtres de l'objet de données qui peuvent différer d'une enveloppe à l'autre.

Lorsque la canonisation exclusive est utilisée comme méthode de canonisation ou de transformation, les espaces de nom des schémas XACML utilisés par les éléments dans l'objet de données XACML doivent alors être liés à des préfixes et inclus dans le paramètre `InclusiveNamespacesPrefixList` à http://www.w3.org/2001/10/xml-exc-c14n# (voir Canonicalization:2002 du W3C).

11.2.2 Considérations supplémentaires sur la canonisation

Des transformations supplémentaires de l'objet de données XACML doivent habituellement être effectuées afin de s'assurer que l'objet de données signé va correspondre à l'objet de données qui est vérifié. La liste de certaines de ces transformations figure ici, mais le présent profil n'essaye pas de spécifier des algorithmes pour les effectuer.

Si un objet de données XACML inclut des éléments de données qui peuvent être représentés sous plus d'une forme (tels que (TRUE, FALSE), (1,0), (true, false)), une méthode de transformation doit alors être définie et spécifiée pour normaliser ces éléments de données.

Le présent profil recommande d'appliquer les canonisations suivantes aux valeurs des types de données correspondants, qu'elles surviennent dans des valeurs d'attribut XML ou dans des attributs XACML.

1) Lorsqu'une représentation canonique d'un type de données défini par XACML est définie dans http://www.w3.org/2001/XMLSchema, la valeur du type de données doit alors être mise dans la forme canonique spécifiée en http://www.w3.org/2001/XMLSchema. Cela inclut les booléens {"true", "false"}, double, dateTime, time, date et hexBinary (majuscules).
2) http://www.w3.org/2001/XMLSchema#anyURI – utilise la forme canonique définie dans la RFC 2396 de l'IETF.
3) http://www.w3.org/2001/XMLSchema#base64Binary – retire toutes les ruptures de ligne et les espaces blancs. Retirer tous les caractères suivant la première séquence de caractères "=". La transformation Base64 (identifiant: http://www.w3.org/TR/xmldsig-core/#sec-Base-64) peut être utile pour effectuer cette canonisation.
4) `urn:oasis:names:tc:xacml:1.0:data-type:x500Name` – normalise d'abord conformément à la RFC 2253 de l'IETF. Si un RDN contient plusieurs paires de `attributeTypeAndValue`, réordonner le `AttributeValuePairs` dans ce RDN en ordre ascendant lors de la comparaison comme chaînes d'octets (voir la description au § 11.6/X.690).
5) `urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name` – normalise la partie domaine du nom en minuscules.
6) expression XPath – appliquer http://www.w3.org/2002/06/xmldsig-filter2 pour mettre l'expression XPath en forme canonique.

11.3 Schémas de signature

L'analyse de tout objet de données XACML dépend du fait d'avoir une copie exacte de tous les schémas desquels dépend l'objet de données XACML. Noter que l'inclusion d'un schéma d'URI dans les attributs d'instance de schéma XACML ne garantit pas qu'une copie exacte du schéma sera utilisée: un agresseur peut substituer un schéma bogué qui contienne l'identifiant correct. Les signatures peuvent aider à se protéger contre la substitution ou modification des schémas dont dépend un objet de données XACML. L'utilisation de signatures à cette fin est décrite dans cette section.

Dans la plupart des cas, un signataire d'objet de données devrait inclure un élément `<Reference>` pour chaque schéma dont dépend l'objet de données XACML dans l'élément `<SignedInfo>` qui contient la `<Reference>` à l'objet ou inclure l'objet de données XACML lui-même.

Dans certains cas, le signataire d'objet de données sait que tous les PDP qui vont évaluer un objet de données XACML donné auront des copies exactes de certains schémas nécessaires pour analyser l'objet de données, et il ne veut pas forcer le PDP à vérifier le résumé de message pour de tels schémas. Dans ces cas, le signataire d'objet de données peut omettre les éléments `<Reference>` pour tous les schémas dont la vérification n'est pas nécessaire.

12 Profil de ressource hiérarchique pour XACML

Il arrive souvent qu'une ressource soit organisée comme une hiérarchie. Les exemples incluent les systèmes de fichiers, les documents XML et les organisations. La présente section spécifie comment XACML peut fournir le contrôle d'accès pour une ressource qui est organisée comme une hiérarchie.

Qu'ont de particulier les ressources organisées en hiérarchie? Avant tout, les politiques sur les hiérarchies appliquent fréquemment les mêmes contrôles d'accès aux sous-arbres entiers de la hiérarchie. Etre capable d'exprimer une seule contrainte de politique qui s'applique à toute une sous-arborescence de nœuds dans la hiérarchie, plutôt que d'avoir à spécifier une contrainte distincte pour chaque nœud, accroît à la fois la facilité d'utilisation et la probabilité que la politique reflète correctement les contrôles d'accès désirés. Une autre caractéristique des ressources hiérarchiques est que l'accès à un nœud peut dépendre de la valeur d'un autre nœud. Par exemple, un patient peut ne se voir accorder l'accès au nœud "diagnostic" dans un document XML de registre médical que si le nom du patient correspond à la valeur dans le nœud "nom du patient". Lorsque c'est le cas, le nœud demandé ne peut pas travailler isolément du reste des nœuds de la hiérarchie et le PDP doit avoir accès aux valeurs des autres nœuds. Finalement, l'identité des nœuds dans une hiérarchie dépend souvent de la position du nœud dans la hiérarchie; il peut aussi y avoir de nombreuses façons de décrire l'identité d'un seul nœud. Pour que les politiques s'appliquent comme voulu aux nœuds, il faut faire attention à la cohérence des représentations de l'identité des nœuds. Autrement, un demandeur peut contourner les contrôles d'accès en demandant un nœud à l'aide d'une identité qui diffère de celle utilisée par la politique.

Une ressource organisée en hiérarchie peut être un "arbre" (une hiérarchie à une seule racine) ou une "forêt" (une hiérarchie avec des racines multiples), mais la hiérarchie ne peut pas être cyclique. Un autre terme pour ces deux types de hiérarchie est "Graphe acyclique dirigé" ou "DAG". Toutes les ressources de cette sorte sont appelées ressources hiérarchiques dans le présent profil. Un document XML est toujours structuré comme un "arbre". D'autres types de ressources hiérarchiques, telles que les fichiers dans un système de fichiers qui prend en charge des liaisons, peuvent être structurés en "forêts".

Les nœuds dans une ressource hiérarchique sont traités comme des ressources individuelles. Une décision d'autorisation qui permet l'accès à un nœud intérieur n'implique pas que l'accès soit permis à ses nœuds descendants. Une décision d'autorisation qui refuse l'accès à un nœud intérieur n'implique pas que l'accès soit refusé à ses nœuds descendants.

Il y a trois types de facilités spécifiés dans le présent profil pour traiter des ressources hiérarchiques:

- représenter l'identité d'un nœud;
- demander l'accès à un nœud;
- déclarer des politiques qui s'appliquent à un ou plusieurs nœuds.

La prise en charge de chacune de ces facilités est facultative.

Le présent paragraphe expose deux façons de représenter une ressource hiérarchique. Dans la première, la hiérarchie dont le nœud fait partie est représentée comme un document XML qui est inclus dans la demande et la ressource demandée est représentée comme un nœud dans ce document. Dans la seconde, la ressource demandée n'est pas représentée comme un nœud dans un document XML, et il n'y a pas dans la demande de représentation de la hiérarchie dont il fait partie. Noter que la ressource cible réelle n'a pas besoin dans le premier cas de faire partie d'un document XML – elle est simplement représentée de cette façon dans la demande. De même, la ressource cible dans le second cas pourrait en réalité faire partie d'un document XML, mais elle est représentée d'une autre façon dans la demande. Et donc, il n'y a pas de corrélation supposée entre la structure de la ressource telle que représentée dans la demande et la structure réelle de la ressource physique à laquelle on accède.

Les facilités de traitement des ressources représentées comme des nœuds dans les documents XML peuvent utiliser le fait que le document XML lui-même est inclus dans la demande de décision. Les expressions XPath peuvent être utilisées pour faire référence aux nœuds dans le présent document d'une façon standard, et peuvent fournir des représentations uniques pour un nœud donné dans le document. Ces facilités ne sont pas disponibles pour des ressources hiérarchiques qui ne sont pas représentées comme des documents XML. D'autres moyens doivent être fournis dans le cas de telles ressources non-XML pour déterminer la localisation du nœud demandé dans la hiérarchie. Dans certains cas, cela peut être fait en incluant la position du nœud dans la hiérarchie au titre de l'identité du nœud. Dans d'autres cas, un nœud peut avoir plus d'une identité normative, comme lorsque le nom de chemin d'un fichier dans un système de fichiers peut inclure des liaisons matérielles. Dans de tels cas, le gestionnaire de contexte du PDP XACML peut avoir besoin de fournir les identités de tous les ancêtres du nœud. Pour toutes ces raisons, les facilités de traitement avec les nœuds dans les documents XML diffèrent des facilités de traitement des nœuds dans d'autres ressources hiérarchiques.

En traitant une ressource hiérarchique, il peut être utile de demander des décisions d'autorisation pour plusieurs nœuds de la ressource dans une seule demande de décision. Le présent paragraphe peut être considéré comme étant posé sur le sommet d'un profil de ressources multiples (voir le § 9), qui à son tour est posé en couche au sommet du comportement spécifié au § 7. La fonctionnalité du présent paragraphe peut, cependant, reposer directement sur la fonctionnalité du § 7.

Le présent paragraphe sur les ressources hiérarchiques suppose que toutes les demandes d'accès à des nœuds multiples dans une ressource hiérarchique ont été résolues en demandes individuelles d'accès à un seul nœud.

12.1 Représentation de l'identité d'un nœud

Pour que les politiques XACML s'appliquent de façon cohérente aux nœuds dans une ressource hiérarchique, il est nécessaire que les nœuds dans ces ressources soient représentés d'une façon cohérente. Si une politique se réfère à un nœud en utilisant une représentation, mais qu'une demande se réfère au nœud en utilisant une représentation différente, la politique ne s'appliquera pas, et la sécurité peut en être compromise.

Les paragraphes suivants décrivent les représentations recommandées pour les nœuds dans les ressources hiérarchiques. D'autres représentations de nœuds dans une ressource donnée sont permises pour autant que tous les points d'administration de politique et tous les points de mise en application de politique qui traitent avec cette ressource se soient mis d'accord pour utiliser cette autre représentation.

12.1.1 Nœuds dans les documents XML

Le présent paragraphe est normatif mais facultatif.

L'URI suivant doit être utilisé comme l'identifiant pour la fonctionnalité spécifiée dans cette partie du présent profil:

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-id
```

L'identité d'un nœud dans une ressource qui est représentée comme une instance de document XML doit être une expression XPath qui s'évalue exactement comme ce nœud dans la copie de la ressource qui est contenue dans l'élément `<ResourceContent>` de l'élément `<Resource>` de la `<Request>`.

12.1.2 Nœuds dans des ressources qui ne sont pas des documents XML

Le présent paragraphe est normatif mais facultatif.

L'URI suivant doit être utilisé comme l'identifiant pour la fonctionnalité spécifiée dans cette partie du présent profil:

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-id
```

L'identité d'un nœud dans une ressource hiérarchique qui n'est pas représentée comme une instance de document XML doit être représentée comme un URI qui se conforme à la RFC 2396 de l'IETF. De tels URI sont de la forme suivante.

`<scheme>` ":" `<authority>` "/" `<pathname>`

Les ressources de système de fichiers doivent utiliser le schéma "file:". S'il n'est pas spécifié de `<scheme>` standard pour le type de ressource dans la RFC 2396 de l'IETF ou dans une norme qui se rapporte à un schéma d'URI enregistré, l'URI doit alors utiliser le schéma "`file`:".

La portion `<pathname>` *(nom de chemin)* de l'URI doit être de la forme

`<root name> [ "/" <node name> ]*`

La séquence des valeurs `<root name>` *(nom de racine)* et `<node name>` *(nom de nœud)* doit correspondre aux noms de composant hiérarchique individuel des ancêtres du nœud représenté le long du chemin depuis un nœud `<root>` jusqu'au nœud représenté.

La canonisation suivante doit être utilisée.

- Le codage de l'URI doit être UTF8.
- Les portions de l'URI insensibles à la casse doivent être en minuscules.
- Le codage en pourcentage de caractères doit être conforme à la RFC 2396 de l'IETF.
- La portion `<authority>` de l'URI doit être spécifiée et doit être la représentation standard de l'autorité pour le type de ressource en question. Lorsque l'`<authority>` pourrait être spécifiée en utilisant un nom de service de nom de domaine (DNS) ou une adresse numérique IPv4 ou IPv6, le nom DNS doit être utilisé.
- Les composants de la portion `<pathname>` de l'URI doivent être spécifiés en utilisant la forme canonique pour de tels composants dans l'`<authority>`.+
- Conformément à la RFC 2396 de l'IETF, le caractère séparateur entre les composants hiérarchiques de la portion `<pathname>` de l'URI doit être le caractère "/". Les séquences du caractère "/" doivent être résolues en un seul "/". Les identités de nœud ne doivent pas se terminer par le caractère "/".
- Le `<pathname>` ne doit pas contenir de liaisons symboliques.

– Toutes les valeurs de <pathname> doivent être absolues.

– S'il y a plus d'un chemin absolu, pleinement résolu d'une <root> à l'<authority> pour le nœud représenté, un attribut de ressource séparé avec l'AttributeId "urn:oasis:names:tc:xacml:1.0:resource:resource-id" et le DataType http://urn:oasis:names:tc:xacml:1.0:data-type:anyURI doit être présent dans le contexte de la demande pour chacun de ces chemins.

12.2 Demande d'accès à un nœud

Pour que les politiques XACML s'appliquent de façon cohérente aux nœuds dans une ressource hiérarchique, il est nécessaire que chaque contexte de demande qui représente une demande d'accès à un nœud de cette ressource utilise une description cohérente de l'accès à ce nœud. Si une politique se réfère à certains attributs attendus à un nœud, mais que le contexte de la demande ne contient pas ces attributs, ou si les attributs ne sont pas exprimés de la façon attendue, la politique peut ne pas s'appliquer, et la sécurité peut en être compromise.

Les paragraphes qui suivent décrivent les descriptions de contexte de demande recommandées de l'accès aux nœuds dans des ressources hiérarchiques. D'autres représentations de telles demandes sont permises pour autant que tous les points d'administration de politique et tous les points de mise en application de politique qui traitent avec cette ressource se soient mis d'accord pour utiliser ces autres représentations.

12.2.1 Nœuds dans un document XML

Le présent paragraphe est normatif mais facultatif.

L'URI suivant doit être utilisé comme l'identifiant pour la fonctionnalité spécifiée dans cette partie du présent profil:

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req
```

Les attributs avec des AttributeId de

```
"urn:oasis::names:tc:xacml:2.0:resource:resource-parent"
"urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor"
```

et:

```
"urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor-or-self"
```

sont d'implémentation facultative. Si l'utilisation dans des ressources représentées comme documents XML est prise en charge, les URI suivants doivent être utilisés comme identifiants pour la fonctionnalité qu'ils représentent:

```
"urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req:resource-parent"
"urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req:resource-ancestor"
```

et:

```
"urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req:resource-ancestor-
or-self"
```

Pour demander l'accès à une ressource représentée comme un nœud dans un document XML, l'élément <Resource> du contexte de la demande doit contenir les éléments et attributs XML suivants:

– Un élément <ResourceContent> contenant toute l'instance de document XML dont fait partie le nœud demandé.

– Un élément <Attribute> avec un AttributeId de "urn:oasis::names:tc:xacml:1.0:resource:resource-id" et un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression". La <AttributeValue> de cet <Attribute> doit être une expression XPath dont le nœud de contexte doit être le seul et unique enfant de l'élément <ResourceContent>. Cette expression XPath doit s'évaluer comme un ensemble de nœuds contenant le nœud unique de l'élément <ResourceContent> qui est le nœud auquel l'accès est demandé. Cet <Attribute> peut spécifier un producteur.

– Un élément <Attribute> avec un AttributeId de "urn:oasis::names:tc:xacml:2.0:resource:resource-parent" et un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression". La <AttributeValue> de cet <Attribute> doit être une expression XPath; le nœud de contexte pour cette expression XPath doit être le seul et unique enfant de l'élément <ResourceContent>. Cette expression XPath doit s'évaluer comme un ensemble de nœuds contenant le nœud unique de l'élément <ResourceContent> qui est le parent immédiat du nœud représenté dans l'attribut "resource-id". Cet <Attribute> peut spécifier un producteur.

– Pour chaque nœud dans l'instance de document XML qui est un ancêtre du nœud représenté par l'attribut "resource-id", un élément <Attribute> avec un AttributeId de "urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor" et un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression". La <AttributeValue> de cet <Attribute> doit être une expression XPath; le nœud de contexte pour cette expression XPath doit être le seul et unique enfant de l'élément <ResourceContent>. Cette expression XPath doit s'évaluer comme un ensemble de nœuds contenant le nœud unique de l'élément <ResourceContent> qui est l'ancêtre respectif du nœud représenté dans l'attribut "resource-id". Pour chaque attribut "resource-parent", il doit y avoir un attribut "resource-ancestor" correspondant. Cet <Attribute> peut spécifier un producteur.

– Pour chaque nœud dans l'instance de document XML qui est un ancêtre du nœud représenté par l'attribut "resource-id", et pour le nœud "resource-id" lui-même, un élément <Attribute> avec un AttributeId de "urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor-or-self" et un DataType de "urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression". La <AttributeValue> de cet <Attribute> doit être une expression XPath; le nœud de contexte pour cette expression XPath doit être le seul et unique enfant de l'élément <ResourceContent>. Cette expression XPath doit s'évaluer comme un ensemble de nœuds contenant le nœud unique dans l'élément <ResourceContent> qui est l'ancêtre respectif du nœud représenté dans l'attribut "resource-id", ou qui est le nœud "resource-id" lui-même. Pour chaque attribut "resource-parent" et "resource-id", il doit y avoir un attribut "resource-ancestor-or-self" correspondant. Cet <Attribute> peut spécifier un producteur.

Des attributs supplémentaires peuvent être inclus dans l'élément <Resource>. En particulier, l'attribut suivant peut être inclus:

– Un élément <Attribute> avec un AttributeId de "urn:oasis::names:tc:xacml:2.0:resource:document-id" et un DataType de "urn:oasis:names:tc:xacml:1.0:data-type:anyURI". La <AttributeValue> de cet <Attribute> doit être un URI qui identifie le document XML dont fait partie la ressource demandée, et dont une copie est présente dans l'élément <ResourceContent>. Cet <Attribute> peut spécifier un producteur.

12.2.2 Nœuds dans une ressource qui n'est pas un document XML

Le présent paragraphe est normatif mais facultatif.

L'URI suivant doit être utilisé comme l'identifiant pour la fonctionnalité spécifiée dans cette partie du présent paragraphe:

urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req.

Les attributs avec les AttributeId de

"urn:oasis::names:tc:xacml:2.0:resource:resource-parent",

"urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor",

et:

"urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor-ou-self"

sont d'implémentation facultative. Si l'utilisation de ressources qui ne se présentent pas comme des documents XML est prise en charge, les URI suivants doivent être utilisés comme identifiants pour la fonctionnalité qu'ils représentent:

"urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req:resource-parent",

"urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req:resource-ancestor",

et:

"urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req:resource-ancestor-or-self".

Pour demander l'accès à un nœud dans une ressource hiérarchique qui n'est pas représentée comme un document XML, l'élément <Resource> du contexte de la demande ne doit pas contenir d'élément <ResourceContent>. L'élément <Resource> du contexte de la demande doit contenir les éléments et attributs XML suivants. Noter qu'un nœud dans une ressource hiérarchique qui n'est pas représentée comme un document XML peut avoir plusieurs parents. Par

exemple, dans un système de fichiers qui prend en charge des liaisons physiques, il peut y avoir plusieurs chemins normatifs pour un seul fichier. Chacun de ces chemins peut contenir différents ensembles de parents et ancêtres.

- Pour chaque représentation normative du nœud demandé, un élément `<Attribute>` avec un `AttributeId` de "`urn:oasis::names:tc:xacml:1.0:resource:resource-id`". La `<AttributeValue>` de cet `<Attribute>` doit être une identité unique, normative, du nœud auquel l'accès est demandé. Le `DataType` de cet `<Attribute>` doit dépendre de la représentation choisie pour l'identité des nœuds dans cette ressource particulière. Cet `<Attribute>` peut spécifier un producteur.
- Pour chaque parent immédiat du nœud spécifié dans le ou les attributs "`resource-id`", et pour chaque représentation normative de ce nœud parent, un élément `<Attribute>` avec un `AttributeId` de "`urn:oasis::names:tc:xacml:2.0:resource:resource-parent`". La `<AttributeValue>` de cet `<Attribute>` doit être l'identité normative du nœud parent. Le `DataType` de cet `<Attribute>` doit dépendre de la représentation choisie pour l'identité des nœuds dans cette ressource particulière. Cet `<Attribute>` peut spécifier un producteur. Si le nœud demandé fait partie d'une forêt plutôt que d'un seul arbre, ou si le nœud parent a plus d'une représentation normative, il doit y avoir au moins une instance de cet attribut pour chaque parent sur chaque chemin des diverses racines dont le nœud demandé est un descendant, et pour chaque représentation normative de chacun de ces parents.
- Pour chaque ancêtre du nœud spécifié dans le ou les attributs "`resource-id`", et pour chaque représentation normative de ce nœud ancêtre, un élément `<Attribute>` avec un `AttributeId` de `n:oasis::names:tc:xacml:2.0:resource:resource-ancestor`". La `<AttributeValue>` de cet `<Attribute>` doit être l'identité normative du nœud ancêtre. Le `DataType` de cet `<Attribute>` doit dépendre de la représentation choisie pour l'identité des nœuds dans cette ressource particulière. Cet `<Attribute>` peut spécifier un producteur. Pour chaque attribut "`resource-parent`", il doit y avoir un attribut "`resource-ancestor`" correspondant. Si le nœud demandé fait partie d'une forêt plutôt que d'un seul arbre, ou si le nœud ancêtre a plus d'une représentation normative, il doit y avoir au moins une instance de cet attribut pour chaque ancêtre sur chaque chemin des diverses racines dont le nœud demandé est un descendant, et pour chaque représentation normative de chacun de ces ancêtres. L'ordre des valeurs pour cet attribut ne reflète pas nécessairement la position de chaque nœud ancêtre dans la hiérarchie.
- Pour chaque ancêtre du nœud spécifié dans le ou les attributs "`resource-id`", et pour chaque représentation normative de ce nœud ancêtre, et pour chaque représentation normative du nœud "`resource-id`" lui-même, un élément `<Attribute>` avec un `AttributeId` de "`urn:oasis::names:tc:xacml:2.0:resource:resource-ancestor-or-self`". La `<AttributeValue>` de cet `<Attribute>` doit être l'identité normative respective du nœud ancêtre ou du nœud "resource-id" lui-même. Le `DataType` de cet `<Attribute>` doit dépendre de la représentation choisie pour l'identité des nœuds dans cette ressource particulière. Cet `<Attribute>` peut spécifier un producteur. Pour chaque attribut "`resource-ancestor`" et "`resource-id`", il doit y avoir un attribut "`resource-ancestor-or-self`" correspondant. Si le nœud demandé fait partie d'une forêt plutôt que d'un simple arbre, ou si le nœud ancêtre a plus d'une représentation normative, il doit y avoir au moins une instance de cet attribut pour chaque ancêtre le long de chaque chemin des diverses racines desquelles le nœud demandé est un descendant, et pour chaque représentation normative de chacun de ces ancêtres. L'ordre des valeurs pour cet attribut ne reflète pas nécessairement la position de chaque nœud ancêtre dans la hiérarchie.

Des attributs supplémentaires peuvent être inclus dans l'élément `<Resource>`.

12.3 Déclaration des politiques qui s'appliquent aux nœuds

Le présent paragraphe est pour information.

Le présent paragraphe décrit diverses façons de spécifier un prédicat de politique qui puisse s'appliquer aux nœuds multiples dans une ressource hiérarchique. Cette liste ne prétend pas être exhaustive.

12.3.1 Politiques s'appliquant aux nœuds dans toute ressource hiérarchique

Le présent paragraphe est pour information.

Les attributs de ressource avec les valeurs d'AttributeId suivantes, décrites au § 12.5, peuvent être utilisés pour déclarer des politiques qui s'appliquent à un ou plusieurs nœuds dans toute ressource hiérarchique.

```
urn:oasis:names:tc:xacml:2.0:resource:resource-parent
urn:oasis:names:tc:xacml:2.0:resource:resource-ancestor
urn:oasis:names:tc:xacml:2.0:resource:resource-ancestor-or-self
```

Noter qu'un <ResourceAttributeDesignator> qui se réfère à l'attribut "resource-parent", "resource-ancestor", ou "resource-ancestor-or-self" retournera un sac de valeurs représentant, respectivement, toutes les identités normatives de tous les parents, ancêtres, ou ancêtres plus la ressource elle-même de la ressource à laquelle l'accès est demandé. La représentation des identités de ces parents, de ces ancêtres, ou d'elle-même, n'indiquera pas nécessairement le chemin de la racine de la hiérarchie respective au parent, ancêtre, ou elle-même, sauf dans la représentation recommandée au § 12.2.2.

Les fonctions de sac XACML standards et d'ordre supérieur peuvent être utilisées pour déclarer des politiques qui s'appliquent à un ou plusieurs nœuds dans toute ressource hiérarchique. Les nœuds utilisés comme arguments pour ces fonctions peuvent être spécifiés en utilisant un <ResourceAttributeDesignator> avec la valeur d'AttributeId de "resource-parent", "resource-ancestor" ou "resource-ancestor-or-self".

12.3.2 Politiques s'appliquant seulement aux nœuds dans les documents XML

Le présent paragraphe est pour information.

Pour les ressources hiérarchiques qui sont représentées comme des instances de document XML, la fonction suivante peut être utilisée pour déclarer des prédicats de politique qui s'appliquent à un ou plusieurs nœuds dans cette ressource.

```
urn:oasis:names:tc:xacml:2.0:function:xpath-node-match
```

L'élément standard XACML <AttributeSelector> peut être utilisé dans des politiques pour se référer à toute une ressource, ou à des portions d'une ressource, représentée comme un document XML et contenue dans l'élément <ResourceContent> d'un contexte de demande.

Les fonctions de sac XACML standards et d'ordre supérieur peuvent être utilisées pour déclarer des politiques qui s'appliquent à un ou plusieurs nœuds dans une ressource représentée comme un document XML. Les nœuds utilisés comme arguments pour ces fonctions peuvent être spécifiés en utilisant un <AttributeSelector> qui choisit une portion de l'élément <ResourceContent> de l'élément <Resource>.

12.3.3 Politiques ne s'appliquant qu'aux nœuds dans des ressources non-XML

Le présent paragraphe est pour information.

Pour les ressources hiérarchiques qui ne sont pas représentées comme des instances de document XML, et où est utilisée la représentation d'URI des nœuds comme spécifié dans le présent profil, les fonctions suivantes peuvent être utilisées pour déclarer les politiques qui s'appliquent à un ou plusieurs nœuds dans cette ressource.

```
urn:oasis:names:tc:xacml:1.0:function:anyURI-equal
urn:oasis:names:tc:xacml:1.0:function:regexp-uri-match
```

12.4 Nouveau DataType: xpath-expression

Le présent paragraphe est normatif mais facultatif.

La valeur suivante pour la valeur d'attribut XML DataType peut être prise en charge pour une utilisation avec des ressources hiérarchiques représentées comme des documents XML. La prise en charge de ce DataType est exigée afin de se conformer au § 12.1.1.

Le DataType représenté par l'URI suivant représente une expression XPath. Les valeurs d'attribut qui ont ce DataType doivent être des chaînes qui sont à interpréter comme des expressions XPath. Le résultat de l'évaluation d'un tel attribut doit être l'ensemble de nœuds qui résulte de l'évaluation de l'expression XPath. Si la chaîne n'est pas une expression XPath valide, le résultat de l'évaluation de l'attribut doit être indéterminé.

```
urn:oasis:names:tc:xacml:2.0:data-type:xpath-expression
```

12.5 Nouveaux identifiants d'attribut

Le présent paragraphe est normatif mais facultatif.

12.5.1 document-id

L'identifiant suivant indique l'identité du document XML qui représente la hiérarchie dont fait partie la ressource demandée et dont une copie est présente dans l'élément <ResourceContent>. Chaque fois qu'est demandé l'accès à un nœud dans une ressource représentée comme un document XML, une ou plusieurs instances d'un attribut avec un AttributeId peuvent être fournies dans l'élément <Resource> du contexte de la demande. Le DataType de ces attributs doit être "urn:oasis:names:tc:xacml:1.0:data-type:anyURI".

```
urn:oasis:names:tc:xacml:2.0:resource:document-id
```

12.5.2 resource-parent

L'identifiant suivant indique une identité normative d'un nœud parent dans l'arbre ou la forêt dont fait partie le nœud demandé. Chaque fois qu'est demandé l'accès à un nœud dans une ressource hiérarchique, une instance d'un attribut avec cet AttributeId doit être fournie dans l'élément `<Resource>` du contexte de la demande pour chaque représentation normative de chaque nœud qui est un parent du nœud demandé.

```
urn:oasis:names:tc:xacml:2.0:resource:resource-parent
```

12.5.3 resource-ancestor

L'identifiant suivant indique une identité normative d'un nœud ancêtre dans l'arbre ou la forêt dont fait partie le nœud demandé. Chaque fois qu'est demandé l'accès à un nœud dans une ressource hiérarchique, une instance d'un attribut avec cet `AttributeId` doit être fournie dans l'élément `<Resource>` du contexte de la demande pour chaque représentation normative de chaque nœud qui est un ancêtre du nœud demandé.

```
urn:oasis:names:tc:xacml:2.0:resource:resource-ancestor
```

12.5.4 resource-ancestor-or-self

L'identifiant suivant indique une identité normative d'un nœud ancêtre dans l'arbre ou la forêt dont fait partie le nœud demandé, ou une identité normative du nœud demandé lui-même. Chaque fois qu'est demandé l'accès à un nœud dans une ressource hiérarchique, une instance d'un attribut avec cet `AttributeId` doit être fournie dans l'élément `<Resource>` du contexte de la demande pour chaque représentation normative de chaque nœud qui est un ancêtre du nœud demandé, et pour chaque représentation normative du nœud demandé lui-même.

```
urn:oasis:names:tc:xacml:2.0:resource:resource-ancestor-or-self
```

12.6 Identifiants de nouveau profil

Les valeurs d'URI suivantes doivent être utilisées comme identifiants pour la fonctionnalité spécifiée dans divers paragraphes du présent profil:

Paragraphe 12.1.1: nœuds dans les documents XML

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-id
```

Paragraphe 12.1.2: nœuds dans des ressources qui ne sont pas des documents XML

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-id
```

Paragraphe 12.2.1: nœuds dans un document XML

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req
```

La prise en charge des attributs "`resource-parent`", "`resource-ancestor`" et "`resource-ancestor-or-self`" est facultative dans ce paragraphe, et ils ont donc des identifiants séparés:

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req:resource-parent
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req:resource-ancestor
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:xml-node-req:resource-ancestor-
or-self
```

Paragraphe 12.2.2: nœuds dans une ressource qui n'est pas un document XML

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req
```

La prise en charge des attributs "`resource-parent`", "`resource-ancestor`" et "`resource-ancestor-or-self`" est facultative dans ce paragraphe, et ils ont donc des identifiants séparés:

```
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req:resource-parent
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req:resource-
ancestor
urn:oasis:names:tc:xacml:2.0:profile:hierarchical:non-xml-node-req:resource-
ancestor-or-self
```

13 Profil de politique de confidentialité

Les deux obligations qui pèsent sur un gardien de données sont de s'assurer que l'utilisation des données personnelles est limitée à l'accomplissement de l'objet de leur collecte, ou d'autres objets qui ne sont pas incompatibles avec celui-ci, et d'empêcher la révélation de données personnelles sauf avec le consentement du sujet des données ou par autorité de justice. Le présent paragraphe fournit un profil pour les attributs standards et un élément `<Rule>` standard pour la mise en application de ces obligations, qui se rapportent aux objectifs pour lesquels des informations personnelles identifiables sont collectées et utilisées.

13.1 Attributs standards

Le présent profil définit deux attributs.

```
urn:oasis:names:tc:xacml:2.0:resource:purpose
```

Cet attribut, du type "http://www.w3.org/2001/XMLSchema#string", indique l'objet pour lequel la ressource de données a été collectée. Le propriétaire de la ressource devrait être informé et consentir à l'utilisation de la ressource pour cet objet. La valeur d'attribut peut être une expression régulière. La politique de confidentialité du gardien devrait définir la sémantique de toutes les valeurs disponibles.

```
urn:oasis:names:tc:xacml:2.0:action:purpose
```

Cet attribut, du type "http://www.w3.org/2001/XMLSchema#string", indique l'objet pour lequel l'accès aux ressources de données est demandé. Les objets d'action devraient être organisés de façon hiérarchique, auquel cas la valeur doit représenter un nœud dans la hiérarchie.

13.2 Règles standards: objectif de correspondance

Cette règle doit être utilisée avec l'algorithme de combinaison de règles "`urn:oasis:names:tc:xacml:2.0:rule-combining-algorithm:deny-overrides`". Il stipule que l'accès doit être refusé sauf si l'objet pour lequel l'accès est demandé correspond, par une correspondance d'expression régulière, à l'objet pour lequel la ressource de données a été collectée.

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os" RuleId="
urn:oasis:names:tc:xacml:2.0:matching-purpose"
Effect="Permit">
       <Condition FunctionId="urn:oasis:names:tc:xacml:2.0:function:regexp-
string-match">
              <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:2.0:resource:purpose"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
              <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:2.0:action:purpose"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
       </Condition>
</Rule>
```

Annexe A

Types de données et fonctions

A.1 Introduction

La présente annexe spécifie les types de données et fonctions utilisés dans XACML pour créer des prédicats pour les correspondances de conditions et de cibles.

La présente Recommandation décrit les types de données et sacs de primitives. Les fonctions standards sont nommées et leur sémantique de fonctionnement est décrite.

NOTE – Voir [IEEE 754] et [RBAC] pour la représentation de chaînes d'information de valeurs numériques.

A.2 Types de données

Bien que les instances XML représentent tous les types de données comme des chaînes, un PDP XACML doit raisonner sur des types de données qui, alors qu'ils sont représentés comme des chaînes, ne sont pas que des chaînes. Des types tels que chaînes, booléens, entiers et doubles doivent être convertis de leur représentation de chaîne XML en valeurs qui peuvent être comparées dans leur domaine de discours, comme les nombres. Les types de primitives de données suivants sont spécifiés pour être utilisés avec XACML et avoir des représentations de données explicites:

- http://www.w3.org/2001/XMLSchema#string *(chaîne)*
- http://www.w3.org/2001/XMLSchema#boolean *(booléen)*
- http://www.w3.org/2001/XMLSchema#integer *(entier)*
- http://www.w3.org/2001/XMLSchema#double *(double)*
- http://www.w3.org/2001/XMLSchema#time *(heure)*
- http://www.w3.org/2001/XMLSchema#date *(date)*
- http://www.w3.org/2001/XMLSchema#dateTime *(date et heure)*
- http://www.w3.org/2001/XMLSchema#anyURI *(tout URI)*
- http://www.w3.org/2001/XMLSchema#hexBinary *(binaire hexadécimal)*
- http://www.w3.org/2001/XMLSchema#base64Binary *(binaire base 64)*
- urn:oasis:names:tc:xacml:2.0:data-type:dayTimeDuration *(durée en heure)*
- urn:oasis:names:tc:xacml:2.0:data-type:yearMonthDuration *(durée en mois/an)*
- urn:oasis:names:tc:xacml:1.0:data-type:x500Name *(nom X.500)*
- urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name *(nom de la RFC 822)*
- urn:oasis:names:tc:xacml:2.0:data-type:ipAddress *(adresse IP)*
- urn:oasis:names:tc:xacml:2.0:data-type:dnsName *(nom DNS)*

Pour améliorer l'interopérabilité, il est recommandé que toutes les références horaires soient en UTC.

Un PDP XACML doit être capable de convertir des représentations de chaîne en divers types de données de primitive.

NOTE – Pour les entiers et les doubles, XACML doit utiliser les conversions décrites dans [IEEE 754].

XACML définit six types de données; ce sont:

```
"urn:oasis:names:tc:xacml:2.0:data-type:dayTimeDuration"
"urn:oasis:names:tc:xacml:2.0:data-type:yearMonthDuration"
"urn:oasis:names:tc:xacml:1.0:data-type:x500Name"
"urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name"
"urn:oasis:names:tc:xacml:2.0:data-type:ipAddress"
"urn:oasis:names:tc:xacml:2.0:data-type:dnsName"
```

Ces types représentent des identifiants pour les sujets ou ressources et apparaissent dans plusieurs applications standards, telles que TLS/SSL et la messagerie électronique.

A.2.1 Durée en jours et heures

Le type de primitive "urn:oasis:names:tc:xacml:2.0:data-type:dayTimeDuration" est défini comme restriction du type **xs:duration**, ne retenant que les composants de signe, d'année et de mois.

```
<xs:simpleType name='urn:oasis:names:tc:xacml:2.0:data-
type:yearMonthDuration'>
    <xs:restriction base='xs:duration'>
        <xsd:pattern value="[-]?P\p{Nd}+(Y(\p{Nd}+M)?|M)"/>
    </xs:restriction>
</xs:simpleType>
```

La valeur d'une durée yearMonthDuration est en unités de mois et est:

```
('value of the year component' * 12) + ('value of the month component')
```

Si le composant de signe est "-", la valeur résultante est négative; autrement, la valeur résultante est positive.

A.2.2 Durée en année et mois

Le type de primitive "urn:oasis:names:tc:xacml:2.0:data-type:yearMonthDuration" est défini comme restriction du type **xs:duration**, ne retenant que les composants de signe, de jour, d'heure, de minute et de seconde.

```
<xs:simpleType name='urn:oasis:names:tc:xacml:2.0:data-
type:dayTimeDuration'>
    <xs:restriction base='xs:duration'>
         <xsd:pattern value="[-
]?P(\p{Nd}D(T(\p{Nd}+(H(\p{Nd}+(M(\p{Nd}+(\.\p{Nd}*)?S
            |\.\p{Nd}+S)?|(\.\p{Nd}*)?S)|(\.\p{Nd}*)?S)?|M(\p{Nd}+
            (\.\p{Nd}*)?S|\.\p{Nd}+S)?|(\.\p{Nd}*)?S)|\.\p{Nd}+S))?
            |T(\p{Nd}+(H(\p{Nd}+(M(\p{Nd}+(\.\p{Nd}*)?S|\.\p{Nd}+S)?

|(\.\p{Nd}*)?S)|(\.\p{Nd}*)?S)?|M(\p{Nd}+(\.\p{Nd}*)?S|\.\p{Nd}+S)?
            |(\.\p{Nd}*)?S)|\.\p{Nd}+S))"/>
    </xs:restriction>
</xs:simpleType>
```

La valeur d'une durée dayTimeDuration est en unités de secondes et est:

```
('value of the day component' * 24) +
('value of the hour component' * 60) +
('value of the minute component' * 60) +
('value of the second component')
```

Si le composant de signe est "-", la valeur résultante est négative; autrement, la valeur résultante est positive.

A.2.3 Nom de répertoire X.500

Le type de primitive "urn:oasis:names:tc:xacml:1.0:data-type:x500Name" représente un nom distinctif de X.520. La syntaxe valide pour un tel nom est décrite dans la RFC 2253 de l'IETF.

A.2.4 Nom de la RFC 822 de l'IETF

Le type de primitive "urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name" représente une adresse de messagerie électronique. La syntaxe valide pour un tel nom est décrite dans le § 4.1.2 de la RFC 2821 de l'IETF.

A.2.5 Adresse IP

Le type de primitive "urn:oasis:names:tc:xacml:2.0:data-type:ipAddress" représente une adresse de réseau IPv4 ou IPv6, avec un gabarit facultatif et un accès ou gamme d'accès facultatif. La syntaxe doit être:

```
ipAddress = address [ "/" mask ] [ ":" [ portrange ] ]
```

Pour une adresse IPv4, l'adresse et le gabarit sont formatés conformément à la syntaxe pour un "host" au § 3.2 de la RFC 2396 de l'IETF.

Pour une adresse IPv6, l'adresse et le gabarit sont formatés conformément à la syntaxe pour une "ipv6reference" dans la RFC 2732 de l'IETF. (Noter qu'une adresse ou gabarit IPv6, dans cette syntaxe, est comprise entre des guillemets "[" "]" littéraux.)

A.2.6 Nom DNS

Le type de primitive "urn:oasis:names:tc:xacml:2.0:data-type:dnsName" représente un nom d'hôte du service de nom de domaine (DNS), avec un accès ou gamme d'accès facultatif. La syntaxe doit être:

```
dnsName = hostname [ ":" portrange ]
```

Le hostname *(nom d'hôte)* est formaté conformément au § 3.2 de la RFC 2396 de l'IETF, sauf qu'un caractère générique "*" peut être utilisé dans le composant le plus à gauche du nom d'hôte pour indiquer "tout sous-domaine" sous le domaine spécifié à sa droite.

Pour les deux types de données "urn:oasis:names:tc:xacml:2.0:data-type:ipAddress" et "urn:oasis:names:tc:xacml:2.0:data-type:dnsName", la syntaxe d'accès ou gamme d'accès doit être

```
portrange = portnumber | "-"portnumber | portnumber"-"[portnumber]
```

où "portnumber" *(numéro d'accès)* est un numéro d'accès décimal. Si le numéro d'accès est de la forme "-x", où "x" est un numéro d'accès, la gamme est alors constituée de tous les accès numérotés "x" et en dessous. Si le numéro d'accès est de la forme "x-", la gamme est alors constituée de tous les accès numérotés "x" et au-dessus.

A.3 Fonctions

XACML spécifie les fonctions suivantes. Si un argument d'une de ces fonctions s'évalue comme "Indeterminate", la fonction doit alors être mise à "Indeterminate".

A.3.1 Prédicats d'égalité

Les fonctions suivantes sont les fonctions d'égalité pour les divers types de primitive. Chaque fonction pour un type de données particulier suit une convention standard spécifiée pour ce type de données.

```
urn:oasis:names:tc:xacml:1.0:function:string-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si la valeur de ses deux arguments est de longueur égale et si chaque chaîne est déterminée égale octet par octet conformément à la fonction "integer-equal". Autrement, elle doit retourner "False".

```
urn:oasis:names:tc:xacml:1.0:function:boolean-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#boolean" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les arguments sont égaux. Autrement, elle doit retourner "False".

```
urn:oasis:names:tc:xacml:1.0:function:integer-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#integer" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean".

NOTE 1 – Voir [IEEE 754] pour des informations sur les évaluations entières sur les entiers.

```
urn:oasis:names:tc:xacml:1.0:function:double-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#double" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean".

NOTE 2 – Voir [IEEE 754] sur la façon d'évaluer les doubles.

```
urn:oasis:names:tc:xacml:1.0:function:date-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#date" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les valeurs des deux arguments sont égales. Si l'un des arguments manque d'une zone horaire explicite, une valeur de zone horaire doit être fournie par l'implémentation.

```
urn:oasis:names:tc:xacml:1.0:function:time-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#time" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les valeurs des deux arguments sont égales. Si l'un des arguments manque d'une zone horaire explicite, une valeur de zone horaire doit être fournie par l'implémentation.

```
urn:oasis:names:tc:xacml:1.0:function:dateTime-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#dateTime" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les valeurs des deux arguments sont égales. Si l'un des arguments manque d'une zone horaire explicite, une valeur de zone horaire doit être fournie par l'implémentation.

```
urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-equal
```

Cette fonction doit prendre deux arguments de type de données "`urn:oasis:names:tc:xacml:2.0:datatypes:dayTimeDuration` " et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les valeurs des deux arguments sont égales. Noter que la représentation lexicale de chaque argument doit être convertie en une valeur exprimée en fraction de seconde.

```
urn:oasis:names:tc:xacml:1.0:function:yearMonthDuration-equal
```

Cette fonction doit prendre deux arguments de type de données "`urn:oasis:names:tc:xacml:2.0:datatypes:yearMonthDuration` " et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les valeurs des deux arguments sont égales. Noter que la représentation lexicale de chaque argument doit être convertie en une valeur exprimée en mois entiers.

```
urn:oasis:names:tc:xacml:1.0:function:anyURI-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#anyURI" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si et seulement si les valeurs des deux arguments sont égales codet par codet.

```
urn:oasis:names:tc:xacml:1.0:function:x500Name-equal
```

Cette fonction doit prendre deux arguments de "`urn:oasis:names:tc:xacml:1.0:data-type:x500Name`" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si chaque nom distinctif relatif (RDN) correspond dans les deux arguments. Autrement, elle doit retourner "False". Deux RDN doivent être dits correspondre si et seulement si le résultat des opérations suivantes est "True".

1) Normaliser les deux arguments conformément à la RFC 2253 de l'IETF.
2) Si un RDN contient plusieurs paires d'`attributeTypeAndValue`, réordonner l'`AttributeValuePairs` dans ce RDN en ordre ascendant lors de la comparaison à une chaîne d'octets (décrite dans le § 11.6/X.690).
3) Comparer les RDN en utilisant les règles du § 4.1.2.4 de la RFC 3280 de l'IETF.

```
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-equal
```

Cette fonction doit prendre deux arguments de type de données "`urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name`" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si les deux arguments sont égaux. Autrement, elle doit retourner "False". Un nom de la RFC 822 de l'IETF consiste en une partie locale suivie par "@" suivi par une partie domaine. La partie locale est sensible à la casse, alors que la partie domaine (qui est habituellement un nom d'hôte DNS) n'est pas sensible à la casse. Effectuer les opérations suivantes:

1) normaliser la partie domaine de chaque argument en minuscules;
2) comparer les expressions en appliquant la fonction "`urn:oasis:names:tc:xacml:1.0:function:string-equal`" aux arguments normalisés.

```
urn:oasis:names:tc:xacml:1.0:function:hexBinary-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#hexBinary" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si les séquences d'octets représentées par la valeur des deux arguments ont une longueur égale et sont égales dans une comparaison conjonctive, point par point, en utilisant la fonction "`urn:oasis:names:tc:xacml:1.0:function:integer-equal`". Autrement, elle doit retourner "False". La conversion de la représentation de chaîne à une séquence d'octets doit être comme spécifié au § 3.2.15 de Datatypes:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:base64Binary-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#base64Binary" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si les séquences d'octets représentées par la valeur des deux arguments ont une longueur égale et sont égales dans une comparaison conjonctive, point par point, en utilisant la fonction "`urn:oasis:names:tc:xacml:1.0:function:integer-equal`". Autrement, elle doit retourner "False". La conversion de la représentation de chaîne à une séquence d'octets doit être comme spécifié au § 3.2.16 de Datatypes:2001 du W3C.

A.3.2 Fonctions arithmétiques

Toutes les fonctions suivantes doivent prendre deux arguments du type de données spécifié, entier ou double, et doivent retourner un élément de type de données, respectivement entier ou double. Cependant, les fonctions "add" *(addition)* peuvent prendre plus de deux arguments. Dans une expression qui contient une de ces fonctions, si un des arguments est "Indeterminate", l'expression doit alors s'évaluer comme "Indeterminate". Dans le cas de fonctions de division, si le diviseur est zéro, la fonction doit alors s'évaluer comme "Indeterminate".

NOTE – Chaque évaluation de fonction devrait procéder comme spécifié par ses contreparties logiques dans [IEEE 754].

```
urn:oasis:names:tc:xacml:1.0:function:integer-add
```

Cette fonction peut avoir deux ou plusieurs arguments.

```
urn:oasis:names:tc:xacml:1.0:function:double-add
```

Cette fonction peut avoir deux ou plusieurs arguments.

```
urn:oasis:names:tc:xacml:1.0:function:integer-subtract
urn:oasis:names:tc:xacml:1.0:function:double-subtract
urn:oasis:names:tc:xacml:1.0:function:integer-multiply
urn:oasis:names:tc:xacml:1.0:function:double-multiply
urn:oasis:names:tc:xacml:1.0:function:integer-divide
urn:oasis:names:tc:xacml:1.0:function:double-divide
urn:oasis:names:tc:xacml:1.0:function:integer-mod
```

Les fonctions suivantes doivent prendre un seul argument du type de données spécifié. Les fonctions d'arrondi et de plancher doivent prendre un seul argument de type de données "http://www.w3.org/2001/XMLSchema#double" et retourner une valeur du type de données "http://www.w3.org/2001/XMLSchema#double".

```
urn:oasis:names:tc:xacml:1.0:function:integer-abs
urn:oasis:names:tc:xacml:1.0:function:double-abs
urn:oasis:names:tc:xacml:1.0:function:round
urn:oasis:names:tc:xacml:1.0:function:floor
```

A.3.3 Fonctions de conversion de chaîne

Les fonctions suivantes convertissent les valeurs des types de primitive du type de données "http://www.w3.org/2001/XMLSchema#string".

```
urn:oasis:names:tc:xacml:1.0:function:string-normalize-space
```

Cette fonction doit prendre un argument du type de données "http://www.w3.org/2001/XMLSchema#string" et doit normaliser la valeur en effaçant tous les caractères espace blanc devant et derrière.

```
urn:oasis:names:tc:xacml:1.0:function:string-normalize-to-lower-case
```

Cette fonction doit prendre un argument du type de données "http://www.w3.org/2001/XMLSchema#string" et doit normaliser la valeur en convertissant chaque caractère majuscule en son équivalent minuscule.

A.3.4 Fonctions de conversion numérique de type de données

Les fonctions suivantes convertissent le type de données "http://www.w3.org/2001/XMLSchema#integer" et les types de primitive "http://www.w3.org/2001/XMLSchema#double".

```
urn:oasis:names:tc:xacml:1.0:function:double-to-integer
```

Cette fonction doit prendre un argument de type de données "http://www.w3.org/2001/XMLSchema#double" et doit tronquer sa valeur numérique à un nombre entier et retourner un élément de type de données "http://www.w3.org/2001/XMLSchema#integer".

```
urn:oasis:names:tc:xacml:1.0:function:integer-to-double
```

Cette fonction doit prendre un argument de type de données "http://www.w3.org/2001/XMLSchema#integer" et doit promouvoir sa valeur en un élément de type de données "http://www.w3.org/2001/XMLSchema#double" avec la même valeur numérique.

A.3.5 Fonctions logiques

Le présent paragraphe contient la spécification des fonctions logiques qui opèrent sur des arguments de type de données "http://www.w3.org/2001/XMLSchema#boolean".

```
urn:oasis:names:tc:xacml:1.0:function:or
```

Cette fonction doit retourner "False" si elle n'a pas d'arguments et doit retourner "True" si au moins un de ses arguments s'évalue comme "True". L'ordre d'évaluation doit être du premier argument au dernier. L'évaluation doit s'arrêter avec un résultat de "True" si un argument s'évalue comme "True", laissant le reste des arguments non évalués.

```
urn:oasis:names:tc:xacml:1.0:function:and
```

Cette fonction doit retourner "True" si elle n'a pas d'arguments et doit retourner "False" si un de ses arguments s'évalue comme "False". L'ordre d'évaluation doit être du premier argument au dernier. L'évaluation doit s'arrêter avec un résultat de "False" si un argument s'évalue comme "False", laissant le reste des arguments non évalués.

```
urn:oasis:names:tc:xacml:1.0:function:n-of
```

Le premier argument de cette fonction doit être un type de données de http://www.w3.org/2001/XMLSchema#integer. Les arguments restants doivent être du type de données http://www.w3.org/2001/XMLSchema#boolean. Le premier argument spécifie le nombre minimum des arguments restants qui doivent s'évaluer comme "True" pour que l'expression soit considérée "True". Si le premier argument est 0, le résultat doit être "True". Si le nombre d'arguments après le premier est inférieur à la valeur du premier argument, l'expression doit alors avoir un résultat "Indeterminate". L'ordre d'évaluation doit être: d'abord évaluer la valeur entière, puis évaluer chaque argument suivant. L'évaluation doit s'arrêter et retourner "True" si le nombre d'arguments spécifié s'évalue comme "True". L'évaluation des arguments doit s'arrêter s'il est déterminé que l'évaluation des arguments restants ne satisfera pas l'exigence.

```
urn:oasis:names:tc:xacml:1.0:function:not
```

Cette fonction doit prendre un argument du type de données "http://www.w3.org/2001/XMLSchema#boolean". Si l'argument s'évalue comme "True", le résultat de l'expression doit alors être "False". Si l'argument s'évalue comme "False", le résultat de l'expression doit alors être "True".

NOTE – Lors de l'évaluation de and, or ou n-of, il peut n'être pas nécessaire d'essayer une évaluation complète de chaque argument afin de déterminer si l'évaluation de l'argument résultera en "Indeterminate". L'analyse de l'argument concernant la disponibilité de ses attributs, ou une autre analyse concernant les erreurs, telles que "diviser par zéro", peut rendre l'argument libre d'erreur. De tels arguments survenant dans l'expression dans une position où l'évaluation a déjà été déclarée s'arrêter, n'ont pas besoin d'être traités.

A.3.6 Fonctions de comparaison numérique

Ces fonctions forment un ensemble minimal pour la comparaison de deux nombres, donnant un résultat booléen.

NOTE – Ces fonctions devraient se conformer aux règles édictées dans [IEEE 754].

```
urn:oasis:names:tc:xacml:1.0:function:integer-greater-than
urn:oasis:names:tc:xacml:1.0:function:integer-greater-than-or-equal
urn:oasis:names:tc:xacml:1.0:function:integer-less-than
urn:oasis:names:tc:xacml:1.0:function:integer-less-than-or-equal
urn:oasis:names:tc:xacml:1.0:function:double-greater-than
urn:oasis:names:tc:xacml:1.0:function:double-greater-than-or-equal
urn:oasis:names:tc:xacml:1.0:function:double-less-than
urn:oasis:names:tc:xacml:1.0:function:double-less-than-or-equal
```

A.3.7 Fonctions d'arithmétique de date et d'heure

Ces fonctions effectuent des opérations arithmétiques avec la date et l'heure.

```
urn:oasis:names:tc:xacml:1.0:function:dateTime-add-dayTimeDuration
```

Cette fonction doit prendre deux arguments, le premier doit être du type de données "http://www.w3.org/2001/XMLSchema#dateTime" et le second doit être du type de données "`urn:oasis:names:tc:xacml:2.0:data-types:dayTimeDuration`". Elle doit retourner un résultat de "http://www.w3.org/2001/XMLSchema#dateTime". Cette fonction doit retourner la valeur en ajoutant le second argument au premier argument conformément à l'Appendice E de DataTypes:2001 du W3C.

urn:oasis:names:tc:xacml:1.0:function:dateTime-add-yearMonthDuration

Cette fonction doit prendre deux arguments, le premier doit être un "http://www.w3.org/2001/XMLSchema#dateTime" et le second doit être un "urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration". Elle doit retourner un résultat de "http://www.w3.org/2001/XMLSchema#dateTime". Cette fonction doit retourner la valeur en ajoutant le second argument au premier argument conformément à l'Appendice E de DataTypes:2001 du W3C.

urn:oasis:names:tc:xacml:1.0:function:dateTime-subtract-dayTimeDuration

Cette fonction doit prendre deux arguments, le premier doit être un "http://www.w3.org/2001/XMLSchema#dateTime" et le second doit être un "urn:oasis:names:tc:xacml:2.0:data-types:dayTimeDuration". Elle doit retourner un résultat de "http://www.w3.org/2001/XMLSchema#dateTime". Si le second argument est une durée positive, cette fonction doit alors retourner la valeur en ajoutant la durée négative correspondante, conformément à l'Appendice E de DataTypes:2001 du W3C. Si le second argument est une durée négative, le résultat doit alors être comme si la fonction "urn:oasis:names:tc:xacml:1.0:function:dateTime-add-dayTimeDuration" avait été appliquée à la durée positive correspondante.

urn:oasis:names:tc:xacml:1.0:function:dateTime-subtract-yearMonthDuration

Cette fonction doit prendre deux arguments, le premier doit être un "http://www.w3.org/2001/XMLSchema#dateTime" et le second doit être un "urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration". Elle doit retourner un résultat de "http://www.w3.org/2001/XMLSchema#dateTime". Si le second argument est une durée positive, cette fonction doit alors retourner la valeur en ajoutant la durée négative correspondante, conformément à l'Appendice E de DataTypes:2001 du W3C. Si le second argument est une durée négative, le résultat doit alors être comme si la fonction "urn:oasis:names:tc:xacml:1.0:function:dateTime-add-yearMonthDuration" avait été appliquée à la durée positive correspondante.

urn:oasis:names:tc:xacml:1.0:function:date-add-yearMonthDuration

Cette fonction doit prendre deux arguments, le premier doit être un "http://www.w3.org/2001/XMLSchema#date" et le second doit être un "urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration". Elle doit retourner un résultat de "http://www.w3.org/2001/XMLSchema#date". Cette fonction doit retourner la valeur en ajoutant le second argument au premier argument conformément à l'Appendice E de DataTypes:2001 du W3C.

urn:oasis:names:tc:xacml:1.0:function:date-subtract-yearMonthDuration

Cette fonction doit prendre deux arguments, le premier doit être un "http://www.w3.org/2001/XMLSchema#date" et le second doit être un "urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration". Elle doit retourner un résultat de "http://www.w3.org/2001/XMLSchema#date". Si le second argument est une durée positive, cette fonction doit alors retourner la valeur en ajoutant la durée négative correspondante, conformément à l'Appendice E de DataTypes:2001 du W3C. Si le second argument est une durée négative, le résultat doit alors être comme si la fonction "urn:oasis:names:tc:xacml:1.0:function:date-add-yearMonthDuration" avait été appliquée à la durée positive correspondante.

A.3.8 Fonctions de comparaison non numérique

Ces fonctions effectuent des opérations de comparaison sur deux arguments de types non numériques.

urn:oasis:names:tc:xacml:1.0:function:string-greater-than

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si les arguments sont comparés octet par octet et, après un préfixe initial des octets correspondants provenant des deux arguments qui sont considérés égaux par "urn:oasis:names:tc:xacml:1.0:function:integer-equal", la comparaison octet par octet suivante est telle que l'octet provenant du premier argument est plus grand que l'octet provenant du second argument en utilisant la fonction "urn:oasis:names:tc:xacml:2.0:function:integer-greater-then". Autrement, elle doit retourner "False".

urn:oasis:names:tc:xacml:1.0:function:string-greater-than-or-equal

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner un résultat comme s'il était évalué avec la fonction logique "urn:oasis:names:tc:xacml:1.0:function:or" avec deux arguments contenant les fonctions "urn:oasis:names:tc:xacml:1.0:function:string-greater-than" et "urn:oasis:names:tc:xacml:1.0:function:string-equal" contenant les arguments d'origine.

urn:oasis:names:tc:xacml:1.0:function:string-less-than

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si les arguments sont comparés octet par octet et, après un préfixe initial des octets correspondants provenant des deux arguments qui sont considérés égaux par "urn:oasis:names:tc:xacml:1.0:function:integer-equal", la comparaison octet par octet suivante est telle que l'octet provenant du premier argument est inférieur à l'octet provenant du second argument en utilisant la fonction "urn:oasis:names:tc:xacml:1.0:function:integer-less-than". Autrement, elle doit retourner "False".

urn:oasis:names:tc:xacml:1.0:function:string-less-than-or-equal

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner un résultat comme s'il était évalué avec la fonction logique "urn:oasis:names:tc:xacml:1.0:function:or" avec deux arguments contenant les fonctions "urn:oasis:names:tc:xacml:1.0:function:string-less-than" et "urn:oasis:names:tc:xacml:1.0:function:string-equal" contenant les arguments d'origine.

urn:oasis:names:tc:xacml:1.0:function:time-greater-than

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#time" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est supérieur au second argument selon la relation d'ordre spécifiée pour http://www.w3.org/2001/XMLSchema#time (W3C Signature:2002, § 3.2.8). Autrement, elle doit retourner "False".

NOTE 1 – Il n'est pas approprié de comparer une heure qui inclut une valeur de zone horaire avec une qui n'en inclut pas. Dans un tel cas, la fonction time-in-range devrait être utilisée.

urn:oasis:names:tc:xacml:1.0:function:time-greater-than-or-equal

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#time" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est supérieur ou égal au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#time" (W3C Datatypes:2001, § 3.2.8). Autrement, elle doit retourner "False".

NOTE 2 – Il n'est pas approprié de comparer une heure qui inclut une valeur de zone horaire avec une qui n'en inclut pas. Dans un tel cas, la fonction time-in-range devrait être utilisée.

urn:oasis:names:tc:xacml:1.0:function:time-less-than

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#time" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est inférieur au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#time" (W3C Datatypes:2001, § 3.2.8). Autrement, elle doit retourner "False".

NOTE 3 – Il n'est pas approprié de comparer une heure qui inclut une valeur de zone horaire avec une qui n'en inclut pas. Dans un tel cas, la fonction time-in-range devrait être utilisée.

urn:oasis:names:tc:xacml:1.0:function:time-less-than

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#time" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est inférieur ou égal au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#time". Autrement, elle doit retourner "False".

NOTE 4 – Il n'est pas approprié de comparer une heure qui inclut une valeur de zone horaire avec une qui n'en inclut pas. Dans un tel cas, la fonction time-in-range devrait être utilisée.

urn:oasis:names:tc:xacml:1.0:function:time-in-range

Cette fonction doit prendre trois arguments de type de données "http://www.w3.org/2001/XMLSchema#time" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si le premier argument tombe dans la gamme définie inclusivement par les second et troisième arguments. Autrement, elle doit retourner "False". Indépendamment de sa valeur, le troisième argument doit être interprété comme une heure qui est égale, ou plus tard de moins de vingt-quatre heures, au second argument. Si aucune zone horaire n'est fournie pour le premier argument, elle doit utiliser la zone horaire par défaut au gestionnaire de contexte. Si aucune zone horaire n'est fournie pour le second ou le troisième argument, ils doivent alors utiliser la zone horaire provenant du premier argument.

urn:oasis:names:tc:xacml:1.0:function:dateTime-greater-than

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#dateTime" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le

premier argument est supérieur au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#dateTime" par W3C Datatype:2001, § 3.2.7. Autrement, elle doit retourner "False".

NOTE 5 – Si une valeur dateTime n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:dateTime-greater-than-or-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#dateTime" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est supérieur ou égal au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#dateTime" par W3C Datatype:2001, § 3.2.7. Autrement, elle doit retourner "False".

NOTE 6 – Si une valeur dateTime n'inclut pas de valeur de zone horaire, un valeur de zone horaire implicite doit être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:dateTime-less-than
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#dateTime" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est inférieur au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#dateTime" par W3C Datatype:2001, § 3.2.7. Autrement, elle doit retourner "False".

NOTE 7 – Si une valeur dateTime n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:dateTime-less-than-or-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema# dateTime" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est inférieur ou égal au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#dateTime" par W3C Datatypes:2001, § 3.2.7. Autrement, elle doit retourner "False".

NOTE 8 – Si une valeur dateTime n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:date-greater-than
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#date" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est supérieur au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#date". Autrement, elle doit retourner "False".

NOTE 9 – Si une `dataTime` (valeur de date) n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit alors être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:date-greater-than-or-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#date" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est supérieur ou égal au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#date". Autrement, elle doit retourner "False".

NOTE 10 – Si une valeur de date n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit alors être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:date-less-than
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#date" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est inférieur au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#date". Autrement, elle doit retourner "False".

NOTE 11 – Si une valeur de date n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit alors être allouée, comme décrit dans Datatype:2001 du W3C.

```
urn:oasis:names:tc:xacml:1.0:function:date-less-than-or-equal
```

Cette fonction doit prendre deux arguments de type de données "http://www.w3.org/2001/XMLSchema#date" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier

argument est inférieur ou égal au second argument selon la relation d'ordre spécifiée pour "http://www.w3.org/2001/XMLSchema#date". Autrement, elle doit retourner "False".

NOTE 12 – Si une valeur de date n'inclut pas de valeur de zone horaire, une valeur de zone horaire implicite doit alors être allouée, comme décrit dans Datatype:2001 du W3C.

A.3.9 Fonctions de chaîne

Les fonctions suivantes fonctionnent sur les chaînes et les URI.

`urn:oasis:names:tc:xacml:2.0:function:string-concatenate`

Cette fonction doit prendre deux arguments ou plus de type de données "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#string". Le résultat doit être la concaténation, dans l'ordre, des arguments.

`urn:oasis:names:tc:xacml:2.0:function:url-string-concatenate`

Cette fonction doit prendre un argument de type de données "http://www.w3.org/2001/XMLSchema#anyURI" et un ou plusieurs arguments du type "http://www.w3.org/2001/XMLSchema#string", et doit retourner un "http://www.w3.org/2001/XMLSchema#anyURI". Le résultat doit être l'URI construit en ajoutant, dans l'ordre, les arguments "string" de l'argument "anyURI".

A.3.10 Fonctions de sac

Ces fonctions opèrent sur un sac de valeurs 'type', où type est un des types de données de primitive. Des conditions supplémentaires définies pour chacune des fonctions ci-dessous amèneront l'expression à s'évaluer comme "Indeterminate".

`urn:oasis:names:tc:xacml:1.0:function:type-one-and-only`

Cette fonction doit prendre un sac de valeurs 'type' comme argument et doit retourner une valeur de '-type'. Elle doit retourner la seule valeur dans le sac. Si le sac n'a pas une et unique valeur, l'expression doit alors s'évaluer comme "Indeterminate".

`urn:oasis:names:tc:xacml:1.0:function:type-bag-size`

Cette fonction doit prendre un sac de valeurs 'type' comme argument et doit retourner un "http://www.w3.org/2001/XMLSchema#integer" indiquant le nombre de valeurs dans le sac.

`urn:oasis:names:tc:xacml:1.0:function:type-is-in`

Cette fonction doit prendre un argument de 'type' comme premier argument et un sac de valeurs type comme second argument et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit s'évaluer comme "True" si et seulement si le premier argument correspond par "`urn:oasis:names:tc:xacml:x.x:function:type-equal`" à toute valeur dans le sac. Autrement, elle doit retourner "False".

`urn:oasis:names:tc:xacml:1.0:function:type-bag`

Cette fonction doit prendre tout nombre d'arguments de 'type' et retourner un sac de valeurs 'type' contenant les valeurs des arguments. Une application de cette fonction à zéro argument doit produire un sac vide du type de données spécifié.

A.3.11 Fonctions d'ensemble

Ces fonctions opèrent sur des sacs imitant des ensembles en éliminant les éléments dupliqués d'un sac.

`urn:oasis:names:tc:xacml:1.0:function:type-intersection`

Cette fonction doit prendre deux arguments qui sont tous deux un sac de valeurs 'type'. Elle doit retourner un sac de valeurs 'type' tel qu'il contienne seulement les éléments qui sont communs aux deux sacs, ce qui est déterminé par "`urn:oasis:names:tc:xacml:x.x:function:type-equal`". Aucune duplication, comme déterminé par "`urn:oasis:names:tc:xacml:x.x:function:type-equal`", ne doit exister dans le résultat.

`urn:oasis:names:tc:xacml:1.0:function:type-at-least-one-member-of`

Cette fonction doit prendre deux arguments qui sont tous deux un sac de valeurs 'type'. Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit s'évaluer comme "True" si et seulement si au moins

un élément du premier argument est contenu dans le second argument comme déterminé par "urn:oasis:names:tc:xacml:x.x:function:type-is-in".

urn:oasis:names:tc:xacml:1.0:function:type-union

Cette fonction doit prendre deux arguments qui sont tous deux un sac de valeurs 'type'. L'expression doit retourner un sac de ‘type' tel qu'il contienne tous les éléments des deux sacs. Aucune duplication, comme déterminé par "urn:oasis:names:tc:xacml:x.x:function:type-equal", ne doit exister dans le résultat.

urn:oasis:names:tc:xacml:1.0:function:type-subset

Cette fonction doit prendre deux arguments qui sont tous deux un sac de valeurs 'type'. Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument est un sous-ensemble du second argument. Chaque argument doit être considéré comme ayant ses duplications supprimées, comme déterminé par "urn:oasis:names:tc:xacml:x.x:function:type-equal", avant le calcul du sous-ensemble.

urn:oasis:names:tc:xacml:1.0:function:type-set-equals

Cette fonction doit prendre deux arguments qui sont tous deux un sac de valeurs 'type'. Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner le résultat de l'application de "urn:oasis:names:tc:xacml:1.0:function:and" à l'application de "urn:oasis:names:tc:xacml:x.x:function:type-subset" aux premier et second arguments et de l'application de "urn:oasis:names:tc:xacml:x.x:function:type-subset" aux second et premier arguments.

A.3.12 Fonctions de sac d'ordre supérieur

Le présent paragraphe expose les fonctions qui dans XACML effectuent les opérations sur les sacs de telle sorte que ces fonctions puissent s'appliquer aux sacs en général.

Un langage fonctionnel d'objet général pourrait être bénéfique pour la spécification de la sémantique de ces fonctions (voir à l'Appendice III un exemple informatif de l'utilisation d'un langage fonctionnel).

1) urn:oasis:names:tc:xacml:1.0:function:any-of

   Cette fonction applique une fonction booléenne entre une valeur de primitive spécifique et un sac de valeurs, et doit retourner "True" si et seulement si le prédicat est "True" pour au moins un élément du sac.

   Cette fonction doit prendre trois arguments. Le premier argument doit être un élément <xacml:Function> qui nomme une fonction booléenne prenant deux arguments de types de primitive. Le second argument doit être une valeur de type de données de primitive. Le troisième argument doit être un sac de type de données de primitive. L'expression doit être évaluée comme si la fonction nommée dans l'argument <xacml:Function> s'appliquait au second argument et à chaque élément du troisième argument (le sac) et les résultats sont combinés avec "urn:oasis:names:tc:xacml:1.0:function:or".

2) urn:oasis:names:tc:xacml:1.0:function:all-of

   Cette fonction applique une fonction booléenne entre une valeur de primitive spécifique et un sac de valeurs, et retourne "True" si et seulement si le prédicat est "True" pour chaque élément du sac.

   Cette fonction doit prendre trois arguments. Le premier argument doit être un élément <xacml:Function> qui nomme une fonction booléenne prenant deux arguments de type de primitive. Le second argument doit être une valeur de type de données de primitive. Le troisième argument doit être un sac d'un type de données de primitive. L'expression doit être évaluée comme si la fonction nommée dans l'argument <xacml:Function> s'appliquait au second argument et à chaque élément du troisième argument (le sac) et les résultats sont combinés en utilisant "urn:oasis:names:tc:xacml:1.0:function:and".

3) urn:oasis:names:tc:xacml:1.0:function:any-of-any

   Cette fonction applique une fonction booléenne entre chaque élément d'un sac de valeurs et chaque élément d'un autre sac de valeurs, et retourne "True" si et seulement si le prédicat est "True" pour au moins une comparaison.

   Cette fonction doit prendre trois arguments. Le premier argument doit être un élément <xacml:Function> qui nomme une fonction booléenne prenant deux arguments de types de primitive. Le second argument doit être un sac d'un type de données de primitive. Le troisième argument doit être un sac d'un type de données de primitive. L'expression doit être évaluée comme si la fonction nommée

dans l'argument <xacml:Function> s'appliquait entre chaque élément du second argument et chaque élément du troisième argument et que les résultats étaient combinés en utilisant "urn:oasis:names:tc:xacml:1.0:function:or". La sémantique est que le résultat de l'expression doit être "True" si et seulement si le prédicat appliqué est "True" pour au moins une comparaison d'éléments provenant des deux sacs.

4) urn:oasis:names:tc:xacml:1.0:function:all-of-any

Cette fonction applique une fonction booléenne entre les éléments de deux sacs. L'expression doit être "True" si et seulement si le prédicat fourni est 'True' entre chaque élément du premier sac et tout élément du second sac.

Cette fonction doit prendre trois arguments. Le premier argument doit être un élément <xacml:Function> qui nomme une fonction booléenne prenant deux arguments de types de primitive. Le second argument doit être un sac d'un type de données de primitive. Le troisième argument doit être un sac d'un type de données de primitive. L'expression doit être évaluée comme si la fonction "urn:oasis:names:tc:xacml:1.0:function:any-of" avait été appliquée à chaque valeur du premier sac et à tout le second sac en utilisant l'élément xacml:Function fourni, et que les résultats étaient alors combinés en utilisant "urn:oasis:names:tc:xacml:1.0:function:and".

5) urn:oasis:names:tc:xacml:1.0:function:any-of-all

Cette fonction applique une fonction booléenne entre les éléments de deux sacs. L'expression doit être "True" si et seulement si le prédicat fourni est "True" entre chaque élément du second sac et chaque élément du premier sac.

Cette fonction doit prendre trois arguments. Le premier argument doit être un élément <xacml:Function> qui nomme une fonction booléenne prenant deux arguments de types de primitive. Le second argument doit être un sac d'un type de données de primitive. Le troisième argument doit être un sac d'un type de données de primitive. L'expression doit être évaluée comme si la fonction "urn:oasis:names:tc:xacml:1.0:function:any-of" avait été appliquée à chaque valeur du second sac et à la totalité du premier sac en utilisant l'élément xacml:Function fourni, et si les résultats étaient alors combinés en utilisant "urn:oasis:names:tc:xacml:1.0:function:and".

6) urn:oasis:names:tc:xacml:1.0:function:all-of-all

Cette fonction applique une fonction booléenne entre les éléments de deux sacs. L'expression doit être "True" si et seulement si le prédicat fourni est "True" collectivement entre chaque élément du premier sac par rapport à tous les éléments du second sac.

Cette fonction doit prendre trois arguments. Le premier argument doit être un élément <xacml:Function> qui nomme une fonction booléenne qui prend deux arguments de types de primitive. Le second argument doit être un sac d'un type de données de primitive. Le troisième argument doit être un sac d'un type de données de primitive. L'expression s'évalue comme si la fonction nommée dans l'élément <xacml:Function> s'appliquait entre chaque élément du second argument et chaque élément du troisième argument et que les résultats étaient combinés en utilisant "urn:oasis:names:tc:xacml:1.0:function:and". La sémantique est que le résultat de l'expression est "True" si et seulement si le prédicat appliqué est "True" pour tous les éléments du premier sac par rapport à tous les éléments du second sac.

7) urn:oasis:names:tc:xacml:1.0:function:map

Cette fonction convertit un sac de valeurs en un autre sac de valeurs.

Cette fonction doit prendre deux arguments. La première fonction doit être un élément <xacml:Function> qui nomme une fonction prenant un seul argument d'un type de données de primitive et retourne une valeur d'un type de données de primitive. Le second argument doit être un sac d'un type de données de primitive. L'expression doit être évaluée comme si la fonction nommée dans l'élément <xacml:Function> s'appliquait à chaque élément dans le sac, résultant en un sac de la valeur convertie. Le résultat doit être un sac du type de données de primitive qui est retourné par la fonction nommée dans l'élément <xacml:Function>.

A.3.13 Fonctions fondées sur les expressions régulières

Ces fonctions opèrent sur divers types utilisant des expressions régulières et s'évaluent à "http://www.w3.org/2001/XMLSchema#boolean".

urn:oasis:names:tc:xacml:1.0:function:string-regexp-match

Cette fonction décide d'une confrontation d'expression régulière. Elle doit prendre deux arguments de "http://www.w3.org/2001/XMLSchema#string" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean".

Le premier argument doit être une expression régulière et le second argument doit être une chaîne générale. La fonction doit retourner "True" si et seulement si le second argument correspond à la valeur dans le schéma d'expression régulière dans le premier argument. La syntaxe de l'expression régulière doit être celle définie à l'Appendice F de Datatypes:2001 du W3C, augmenté des expressions et règles de schéma supplémentaires suivantes:

- X?? ne correspond pas plus d'une fois à X
- X*? correspond à X n'importe quel nombre de fois, y compris zéro
- X+? correspond à X au moins une fois
- X{n}? correspond à X exactement n fois
- X(n,}? correspond à X au moins n fois
- X{n,m}? correspond à X au moins n fois, mais pas plus de m fois

Lorsqu'une de ces expressions de schéma supplémentaires est utilisée, l'expression régulière doit correspondre à la plus courte sous-chaîne possible du premier argument qui est cohérent avec le schéma.

Sauf pour ces expressions supplémentaires, l'expression régulière doit correspondre à la plus longue sous-chaîne possible du premier argument qui est cohérent avec le schéma.

Excepté lorsque c'est explicitement fixé au début ou à la fin de la chaîne en utilisant, respectivement, les caractères de schéma "^" et "$", le schéma est considéré comme correspondant s'il correspond à toute sous-chaîne du second argument.

Toute implémentation conforme doit prendre en charge les schémas d'expression régulière spécifiés. Une implémentation conforme peut prendre en charge des schémas d'expression régulière supplémentaires.

```
urn:oasis:names:tc:xacml:2.0:function:anyURI-regexp-match
```

Cette fonction décide d'une confrontation d'expression régulière. Elle doit prendre deux arguments; le premier est du type "http://www.w3.org/2001/XMLSchema#string" et le second est du type "http://www.w3.org/2001/XMLSchema#anyURI". Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Le premier argument doit être une expression régulière et le second argument doit être un URI. La fonction doit convertir le second argument en type "http://www.w3.org/2001/XMLSchema#string", puis appliquer "`urn:oasis:names:tc:xacml:1.0:function:string-regexp-match`".

```
urn:oasis:names:tc:xacml:2.0:function:ipAddress-regexp-match
```

Cette fonction décide d'une confrontation d'expression régulière. Elle doit prendre deux arguments; le premier est du type "http://www.w3.org/2001/XMLSchema#string" et le second est du type "`urn:oasis:names:tc:xacml:2.0:data-type:ipAddress`". Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Le premier argument doit être une expression régulière et le second argument doit être une adresse IPv4 ou IPv6. La fonction doit convertir le second argument au type "http://www.w3.org/2001/XMLSchema#string", puis appliquer "`urn:oasis:names:tc:xacml:1.0:function:string-regexp-match`".

```
urn:oasis:names:tc:xacml:2.0:function:dnsName-regexp-match
```

Cette fonction décide d'une confrontation d'expression régulière. Elle doit prendre deux arguments; le premier est du type "http://www.w3.org/2001/XMLSchema#string" et le second est du type "`urn:oasis:names:tc:xacml:2.0:data-type:dnsName`". Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Le premier argument doit être une expression régulière et le second argument doit être un nom DNS. La fonction doit convertir le second argument en type "http://www.w3.org/2001/XMLSchema#string", puis appliquer "`urn:oasis:names:tc:xacml:1.0:function:string-regexp-match`".

```
urn:oasis:names:tc:xacml:2.0:function:rfc822Name-regexp-match
```

Cette fonction décide d'une confrontation d'expression régulière. Elle doit prendre deux arguments; le premier est du type "http://www.w3.org/2001/XMLSchema#string" et le second est du type "`urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name`". Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Le premier argument doit être une expression régulière et le second argument doit être un nom de la RFC 822. La fonction doit convertir le second argument en type "http://www.w3.org/2001/XMLSchema#string", puis appliquer "`urn:oasis:names:tc:xacml:1.0:function:string-regexp-match`".

```
urn:oasis:names:tc:xacml:2.0:function:x500Name-regexp-match
```

Cette fonction décide d'une confrontation d'expression régulière. Elle doit prendre deux arguments; le premier est du type "http://www.w3.org/2001/XMLSchema#string" et le second est du type "urn:oasis:names:tc:xacml:1.0:data-type:x500Name". Elle doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Le premier argument doit être une expression régulière et le second argument doit être un nom de répertoire X.500. La fonction doit convertir le second argument au type "http://www.w3.org/2001/XMLSchema#string", puis appliquer "urn:oasis:names:tc:xacml:1.0:function:string-regexp-match".

A.3.14 Fonctions spéciales de correspondance

Ces fonctions opèrent sur divers types et s'évaluent comme "http://www.w3.org/2001/XMLSchema#boolean" sur la base de l'algorithme de correspondance standard spécifié.

```
urn:oasis:names:tc:xacml:1.0:function:x500Name-match
```

Cette fonction doit prendre deux arguments de "urn:oasis:names:tc:xacml:2.0:data-type:x500Name" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Elle doit retourner "True" si et seulement si le premier argument correspond à une séquence terminale de noms RDN provenant du second argument lors d'une comparaison utilisant x500Name-equal.

```
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-match
```

Cette fonction doit prendre deux arguments, le premier est du type de données "http://www.w3.org/2001/XMLSchema#string" et le second est du type de données "urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name" et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Cette fonction doit s'évaluer comme "True" si le premier argument correspond au second argument conformément au § 3.2.1 de Datatypes:2001 du W3C.

Un nom de la RFC 822 de l'IETF consiste en une partie locale suivie de "@" suivi d'une partie domaine. La partie locale est sensible à la casse alors que la partie domaine (qui est habituellement un nom DNS) n'est pas sensible à la casse.

Le second argument contient un nom rfc822Name complet. Le premier argument est un nom rfc822Name complet ou partiel utilisé pour choisir les valeurs appropriées dans le second argument comme suit.

Pour correspondre à une adresse particulière dans le second argument, le premier argument doit spécifier l'adresse de messagerie complète à satisfaire. Pour correspondre à toute adresse dans un domaine particulier dans le second argument, le premier argument doit spécifier seulement un nom de domaine (habituellement, un nom DNS). Pour correspondre à n'importe quelle adresse dans un domaine particulier dans le second argument, le premier argument doit spécifier la partie de domaine désirée avec un "." en tête.

A.3.15 Fonctions fondées sur XPath

Le présent paragraphe spécifie des fonctions qui prennent des expressions XPath pour arguments. Une expression XPath s'évalue comme un ensemble de nœuds, qui est un ensemble de nœuds XML qui satisfont l'expression. Un nœud ou ensemble de nœuds n'est pas dans le système de type de données formel de XACML. Toutes les comparaisons ou autres opérations sur les ensembles de nœuds sont effectuées isolément de la fonction particulière spécifiée. C'est-à-dire que les expressions XPath dans ces fonctions sont restreintes au contexte de demande XACML. L'élément <xacml-context:Request> est le nœud de contexte pour chaque expression XPath. Les fonctions suivantes sont définies:

```
urn:oasis:names:tc:xacml:1.0:function:xpath-node-count
```

Cette fonction doit prendre un "http://www.w3.org/2001/XMLSchema#string" comme argument, qui doit être interprété comme une expression XPath, et s'évalue comme un "http://www.w3.org/2001/XMLSchema#integer". La valeur retournée de la fonction doit être le compte des nœuds qui au sein de l'ensemble de nœuds satisfont à l'expression XPath en question.

```
urn:oasis:names:tc:xacml:1.0:function:xpath-node-equal
```

Cette fonction doit prendre deux arguments "http://www.w3.org/2001/XMLSchema#string", qui doivent être interprétés comme des expressions XPath, et doivent retourner un "http://www.w3.org/2001/XMLSchema#boolean". La fonction doit retourner "True" si l'un des nœuds XML au sein de l'ensemble de nœuds confrontés par le premier argument est égal à l'un des nœuds XML dans l'ensemble de nœuds confrontés par le second argument. Deux nœuds sont considérés égaux s'ils ont la même identité.

```
urn:oasis:names:tc:xacml:1.0:function:xpath-node-match
```

Cette fonction doit prendre deux arguments "http://www.w3.org/2001/XMLSchema#string", qui doivent être interprétés comme des expressions XPath et doit retourner un "http://www.w3.org/2001/XMLSchema#boolean". Cette fonction doit s'évaluer comme "True" si une des deux conditions suivantes est satisfaite:

1) tout nœud XML dans l'ensemble de nœuds confrontés par le premier argument est égal à tout nœud XML dans l'ensemble de nœuds confrontés par le second argument;

2) tout nœud d'attribut et d'élément en dessous de l'un des nœuds XML de l'ensemble de nœuds confrontés par le premier argument est égal à l'un des nœuds XML dans l'ensemble des nœuds confrontés par le second argument. Deux nœuds sont considérés égaux s'ils ont la même identité.

NOTE – La première condition est équivalente à "xpath-node-equal", et garantit que "xpath-node-equal" est un cas particulier de "xpath-node-match".

A.3.16 Fonctions d'extension et types de primitive

Les fonctions et les types de primitive sont spécifiés par des identifiants de chaîne qui permettent l'introduction de fonctions en plus de celles spécifiées par XACML. Cette approche permet d'étendre le module XACML avec des fonctions spéciales et des types de données de primitive particuliers.

Afin de préserver l'intégrité de la stratégie d'évaluation XACML, le résultat d'une fonction d'extension ne doit dépendre que des valeurs de ses arguments. Des paramètres globaux et cachés ne doivent pas affecter l'évaluation d'une expression. Les fonctions ne doivent pas avoir d'effets secondaires, car l'ordre d'évaluation ne peut pas être garanti de façon standard.

Annexe B

Identifiants XACML

La présente annexe définit des identifiants standards pour les entités couramment utilisées.

B.1 Espaces de nom XACML

Deux espaces de nom XACML sont actuellement définis.

Les politiques sont définies en utilisant cet identifiant.

```
urn:oasis:names:tc:xacml:2.0:policy:schema:os
```

Les demandes et les contextes de réponse sont définis en utilisant cet identifiant.

```
urn:oasis:names:tc:xacml:2.0:context:schema:os
```

B.2 Catégories de sujet d'accès

Cet identifiant indique l'entité système qui a initié la demande d'accès. C'est-à-dire l'entité initiale dans une chaîne de demande. Si la catégorie du sujet n'est pas spécifiée, la valeur par défaut est la suivante:

```
urn:oasis:names:tc:xacml:1.0:subject-category:access-subject
```

Cet identifiant indique l'entité système qui va recevoir les résultats de la demande (utilisé lorsqu'il est distinct du sujet d'accès).

```
urn:oasis:names:tc:xacml:1.0:subject-category:recipient-subject
```

Cet identifiant indique une entité système à travers laquelle a été passée la demande d'accès. Il peut y en avoir plus d'une. Aucun moyen n'est fourni pour spécifier l'ordre dans lequel le message a été passé.

```
urn:oasis:names:tc:xacml:1.0:subject-category:intermediary-subject
```

Cet identifiant indique une entité système associée à une base de code locale ou distante qui a généré la demande. Les attributs de sujet correspondants peuvent inclure l'URL d'où il a été chargé et/ou l'identité du signataire du code. Il peut y en avoir plus d'un. Aucun moyen n'est fourni pour spécifier l'ordre dans lequel ils ont traité la demande.

```
urn:oasis:names:tc:xacml:1.0:subject-category:codebase
```

Cet identifiant indique une entité système associée à l'ordinateur qui a initialisé la demande d'***accès***. Ce pourrait, par exemple, être une identité IPSEC.

```
urn:oasis:names:tc:xacml:1.0:subject-category:requesting-machine
```

B.3 Types de données

Les identifiants suivants indiquent les types de données qui sont définis au § A.2.

```
urn:oasis:names:tc:xacml:2.0:data-types:dayTimeDuration
urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration
urn:oasis:names:tc:xacml:1.0:data-type:x500Name.
urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name
urn:oasis:names:tc:xacml:2.0:data-type:ipAddress
urn:oasis:names:tc:xacml:2.0:data-type:dnsName
```

Les identifiants de type de données suivants sont définis par DataTypes2001 du W3C.

```
http://www.w3.org/2001/XMLSchema#string
http://www.w3.org/2001/XMLSchema#boolean
http://www.w3.org/2001/XMLSchema#integer
http://www.w3.org/2001/XMLSchema#double
http://www.w3.org/2001/XMLSchema#time
http://www.w3.org/2001/XMLSchema#date
```

```
http://www.w3.org/2001/XMLSchema#dateTime
http://www.w3.org/2001/XMLSchema#anyURI
http://www.w3.org/2001/XMLSchema#hexBinary
http://www.w3.org/2001/XMLSchema#base64Binary
```

B.4 Attributs de sujet

Ces identifiants indiquent les attributs d'un sujet. Lorsqu'ils sont utilisés, ils doivent apparaître au sein d'un élément `<Subject>` du contexte de la demande. Il doit y être accédé au moyen d'un élément `<SubjectAttributeDesignator>` ou d'un élément `<AttributeSelector>` pointant sur un élément `<Subject>` du contexte de la demande.

Au plus un de chacun de ces attributs est associé à chaque sujet. Chaque attribut associé à l'authentification incluse dans un seul élément `<Subject>` se rapporte au même événement d'authentification.

Cet identifiant indique le nom du sujet. Le format par défaut est "http://www.w3.org/2001/XMLSchema#string". Pour indiquer d'autres formats, utiliser les attributs `DataType` dont la liste figure au § B.3.

```
urn:oasis:names:tc:xacml:1.0:subject:subject-id
```

Cet identifiant indique la catégorie du sujet. "access-subject" est la valeur par défaut.

```
urn:oasis:names:tc:xacml:1.0:subject-category
```

Cet identifiant indique le domaine de sécurité du sujet. Il identifie l'administrateur et la politique que gère l'espace de nom dans lequel l'identifiant de sujet est administré.

```
urn:oasis:names:tc:xacml:1.0:subject:subject-id-qualifier
```

Cet identifiant indique une clé publique utilisée pour confirmer l'identité du sujet.

```
urn:oasis:names:tc:xacml:1.0:subject:key-info
```

Cet identifiant indique l'heure à laquelle le sujet a été authentifié.

```
urn:oasis:names:tc:xacml:1.0:subject:authentication-time
```

Cet identifiant indique la méthode utilisée pour authentifier le sujet.

```
urn:oasis:names:tc:xacml:1.0:subject:authn-locality:authentication-method
```

Cet identifiant indique l'heure à laquelle le sujet a initié la demande d'accès, selon le PEP.

```
urn:oasis:names:tc:xacml:1.0:subject:request-time
```

Cet identifiant indique l'heure à laquelle a commencé la session en cours du sujet, selon le PEP.

```
urn:oasis:names:tc:xacml:1.0:subject:session-start-time
```

Les identifiants suivants indiquent la localisation à laquelle les accréditifs d'authentification ont été activés. Ils sont destinés à prendre en charge les entités correspondantes provenant de la déclaration d'authentification SAML.

Cet identifiant indique que la localisation est exprimée comme une adresse IP.

```
urn:oasis:names:tc:xacml:1.0:subject:authn-locality:ip-address
```

L'attribut correspondant doit être du type de données "http://www.w3.org/2001/XMLSchema#string".

Cet identifiant indique que la localisation est exprimée comme un nom DNS.

```
urn:oasis:names:tc:xacml:1.0:subject:authn-locality:dns-name
```

L'attribut correspondant doit être du type de données "http://www.w3.org/2001/XMLSchema#string".

Lorsqu'un attribut convenable est déjà défini en LDAP, l'identifiant XACML doit être formé en ajoutant le nom de l'attribut à l'URI de la RFC LDAP de l'IETF (voir la RFC 2256 de l'IETF). Par exemple, le nom de l'attribut pour le `userPassword` *(mot de passe d'utilisateur)* défini dans la RFC 2256 de l'IETF doit être:

```
http://www.ietf.org/rfc/rfc2256.txt#userPassword
```

B.5 Attributs de ressource

Ces identifiants indiquent les attributs de la ressource. Les attributs correspondants peuvent apparaître dans l'élément <Resource> du contexte de la demande et on y accède au moyen d'un élément <ResourceAttributeDesignator>, ou par un élément <AttributeSelector> qui pointe sur l'élément <Resource> du contexte de la demande.

Cet attribut identifie la ressource à laquelle l'accès est demandé. Si un élément <xacml-context:ResourceContent> est fourni, la ressource à laquelle l'accès est demandé doit alors être tout ou partie de la ressource fournie dans l'élément <xacml-context:ResourceContent>.

```
urn:oasis:names:tc:xacml:1.0:resource:resource-id
```

Cet attribut identifie l'espace de nom de l'élément supérieur du contenu de l'élément <xacml-context:ResourceContent>. Dans le cas où le contenu de la ressource est fourni dans le contexte de la demande et où l'espace de nom de la ressource est défini dans la ressource, le PDP doit confirmer que l'espace de nom défini par cet attribut est le même que celui défini dans la ressource. Le type de l'attribut correspondant doit être "http://www.w3.org/2001/XMLSchema#anyURI".

```
urn:oasis:names:tc:xacml:2.0:resource:target-namespace
```

B.6 Attributs d'action

Ces identifiants indiquent les attributs de l'action demandée. Lorsqu'ils sont utilisés, ils doivent apparaître au sein de l'élément <Action> du contexte de la demande. On doit y accéder au moyen d'un élément <ActionAttributeDesignator>, ou d'un élément <AttributeSelector> qui pointe sur l'élément <Action> du contexte de la demande.

Cet attribut identifie l'action pour laquelle l'accès est demandé.

```
urn:oasis:names:tc:xacml:1.0:action:action-id
```

Lorsque l'action est implicite, la valeur de l'attribut action-id doit être

```
urn:oasis:names:tc:xacml:1.0:action:implied-action
```

Cet attribut identifie l'espace de nom dans lequel l'attribut action-id est défini.

```
urn:oasis:names:tc:xacml:1.0:action:action-namespace
```

B.7 Attributs d'environnement

Ces identifiants indiquent les attributs de l'environnement dans lequel la demande de décision va être évaluée. Lorsqu'ils sont utilisés dans la demande de décision, ils doivent apparaître dans l'élément <Environment> du contexte de la demande. On doit y accéder au moyen d'un élément <EnvironmentAttributeDesignator>, ou d'un élément <AttributeSelector> qui pointe sur l'élément <Environment> du contexte de la demande.

Cet identifiant indique l'heure en cours chez le gestionnaire de contexte. En pratique, c'est l'heure à laquelle a été créé le contexte de la demande. Pour cette raison, si ces identifiants apparaissent dans plusieurs endroits dans une <Policy> ou <PolicySet>, la même valeur doit alors être allouée à chaque occurrence dans la procédure d'évaluation, indépendamment du temps qui s'est écoulé entre le traitement de ces occurrences.

```
urn:oasis:names:tc:xacml:1.0:environment:current-time
```

L'attribut correspondant doit être du type de données "http://www.w3.org/2001/XMLSchema#time".

```
urn:oasis:names:tc:xacml:1.0:environment:current-date
```

L'attribut correspondant doit être du type de données "http://www.w3.org/2001/XMLSchema#date".

```
urn:oasis:names:tc:xacml:1.0:environment:current-dateTime
```

L'attribut correspondant doit être du type de données "http://www.w3.org/2001/XMLSchema#dateTime".

B.8 Codes d'état

Les valeurs de code d'état suivantes sont définies.

Cet identifiant indique le succès.

```
urn:oasis:names:tc:xacml:1.0:status:ok
```

Cet identifiant indique que tous les attributs nécessaires pour prendre une décision de politique n'étaient pas disponibles.

```
urn:oasis:names:tc:xacml:1.0:status:missing-attribute
```

Cet identifiant indique que certaines valeurs d'attribut contenaient une erreur de syntaxe, comme une lettre dans un champ numérique.

```
urn:oasis:names:tc:xacml:1.0:status:syntax-error
```

Cet identifiant indique qu'une erreur est survenue durant l'évaluation de politique. Une division par zéro, par exemple.

```
urn:oasis:names:tc:xacml:1.0:status:processing-error
```

B.9 Algorithmes de combinaison

L'algorithme de combinaison de règles deny-overrides a la valeur suivante pour l'attribut `ruleCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides
```

L'algorithme de combinaison de politiques deny-overrides a la valeur suivante pour l'attribut `policyCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:deny-overrides
```

L'algorithme de combinaison de règles permit-overrides a la valeur suivante pour l'attribut `ruleCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides
```

L'algorithme de combinaison de politiques permit-overrides a la valeur suivante pour l'attribut `policyCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:permit-overrides
```

L'algorithme de combinaison de règles first-applicable a la valeur suivante pour l'attribut `ruleCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable
```

L'algorithme de combinaison de règles first-applicable a la valeur suivante pour l'attribut `policyCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:first-applicable
```

L'algorithme de combinaison de politiques only-one-applicable-policy *(une seule politique est applicable)* a la valeur suivante pour l'attribut `policyCombiningAlgId`:

```
urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:only-one-applicable
```

L'algorithme de combinaison de règles ordered-deny-overrides *(priorité de refus ordonné)* a la valeur suivante pour l'attribut ruleCombiningAlgId:

```
urn:oasis:names:tc:xacml:1.1:rule-combining-algorithm:ordered-deny-overrides
```

L'algorithme de combinaison de politiques ordered-deny-overrides a la valeur suivante pour l'attribut policyCombiningAlgId:

```
urn:oasis:names:tc:xacml:1.1:policy-combining-algorithm:ordered-deny-overrides
```

L'algorithme de combinaison de règles ordered-permit-overrides *(priorité d'acceptation ordonnée)* a la valeur suivante pour l'attribut ruleCombiningAlgId:

```
urn:oasis:names:tc:xacml:1.1:rule-combining-algorithm:ordered-permit-overrides
```

L'algorithme de combinaison de politiques ordered-permit-overrides a la valeur suivante pour l'attribut policyCombiningAlgId:

```
urn:oasis:names:tc:xacml:1.1:policy-combining-algorithm:ordered-permit-
overrides
```

Annexe C

Algorithmes de combinaison

La présente annexe contient une description des algorithmes de combinaison de règles et de politiques spécifiés par XACML.

C.1 Deny-overrides

C.1.1 Le présent paragraphe définit l'algorithme de combinaison de règles "Deny-overrides" *(priorité au refus)* d'une politique.

Dans tout l'ensemble des règles de la politique, si une règle évalue à "Deny" *(refus)*, le résultat de la combinaison de règles doit être "Deny". Si une règle évalue à "Permit" et si toutes les autres règles évaluent à "NotApplicable", le résultat de la combinaison de règles doit alors être "Permit". En d'autres termes, "Deny" a la préséance, indépendamment du résultat de l'évaluation de toute autre règle dans la combinaison. Si toutes les règles se trouvent être "NotApplicable" pour la demande de décision, la combinaison de règles doit alors s'évaluer comme "NotApplicable".

Si une erreur survient lors de l'évaluation de la cible ou de la condition d'une règle qui contient une valeur d'effet de "Deny", l'évaluation doit continuer d'évaluer les règles suivantes, en cherchant un résultat de "Deny". Si aucune autre règle n'évalue à "Deny", la combinaison doit s'évaluer comme "Indeterminate", avec l'état d'erreur approprié.

Si au moins une règle évalue à "Permit", si toutes les autres règles qui n'ont pas d'erreur d'évaluation évaluent à "Permit" ou "NotApplicable" et si toutes les règles qui ont des erreurs d'évaluation contiennent des effets de "Permit", le résultat de la combinaison doit alors être "Permit".

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de règles.

```
Decision denyOverridesRuleCombiningAlgorithm(Rule rule[])
{
   Boolean atLeastOneError  = false;
   Boolean potentialDeny    = false;
   Boolean atLeastOnePermit = false;
   for( i=0 ; i < lengthOf(rules) ; i++ )
   {
          Decision decision = evaluate(rule[i]);
          if (decision == Deny)
          {
                return Deny;
          }
          if (decision == Permit)
          {
                atLeastOnePermit = true;
                continue;
          }
          if (decision == NotApplicable)
          {
                continue;
          }
          if (decision == Indeterminate)
          {
                atLeastOneError = true;

                if (effect(rule[i]) == Deny)
                {
                      potentialDeny = true;
                }
                continue;
          }
   }
   if (potentialDeny)
   {
          return Indeterminate;
   }
   if (atLeastOnePermit)
```

```
    {
          return Permit;
    }
    if (atLeastOneError)
    {
          return Indeterminate;
    }
    return NotApplicable;
}
```

C.1.2 Le présent paragraphe définit l'algorithme de combinaison de politiques "Deny-overrides" d'un ensemble de politiques.

Dans tout l'ensemble des politiques dans l'ensemble de politiques, si une politique évalue à "Deny", le résultat de la combinaison de politiques doit être "Deny". En d'autres termes, "Deny" a la préséance, indépendamment du résultat de l'évaluation d'une ou des autres politiques dans l'ensemble de politiques. Si toutes les politiques se trouvent être "NotApplicable" à la demande de décision, l'ensemble de politiques doit alors s'évaluer comme "NotApplicable".

Si une erreur survient pendant l'évaluation de la cible d'une politique, ou si une référence à une politique est considérée comme invalide ou si l'évaluation de politique a un résultat de "Indeterminate", l'ensemble de politiques doit alors s'évaluer comme "Deny".

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de politiques.

```
Decision denyOverridesPolicyCombiningAlgorithm(Policy policy[])
{
    Boolean atLeastOnePermit = false;
    for( i=0 ; i < lengthOf(policy) ; i++ )
    {
          Decision decision = evaluate(policy[i]);
          if (decision == Deny)
          {
                return Deny;
          }
          if (decision == Permit)
          {
                atLeastOnePermit = true;
                continue;
          }
          if (decision == NotApplicable)
          {
                continue;
          }
          if (decision == Indeterminate)
          {
                return Deny;
          }
    }
    if (atLeastOnePermit)
    {
          return Permit;
    }
    return NotApplicable;
}
```

Les obligations des politiques individuelles doivent être combinées comme décrit au § 7.6.14.

C.2 Ordered-deny-overrides

L'alinéa suivant définit l'algorithme de combinaison de règles "Ordered-deny-overrides" d'une politique.

Le comportement de cet algorithme est identique à celui de l'algorithme de combinaison de règles Deny-overrides avec une exception. L'ordre dans lequel la collection de règles est évaluée doit respecter l'ordre figurant dans la politique.

L'alinéa suivant définit l'algorithme de combinaison de politiques "Ordered-deny-overrides" d'un ensemble de politiques.

Le comportement de cet algorithme est identique à celui de l'algorithme de combinaison de politiques Deny-overrides avec une exception. L'ordre dans lequel la collection de politiques est évaluée doit respecter l'ordre figurant dans l'ensemble de politiques.

C.3 Permit-overrides

C.3.1 Le présent paragraphe définit l'algorithme de combinaison de règles "Permit-overrides" d'une politique.

Dans tout l'ensemble de règles de la politique, si une règle évalue à "Permit", le résultat de la combinaison de règles doit être "Permit". Si une règle évalue à "Deny" et si toutes les autres règles évaluent à "NotApplicable", la politique doit alors s'évaluer comme "Deny". En d'autres termes, "Permit" prend la préséance, indépendamment du résultat de l'évaluation de toutes les autres règles dans la politique. Si toutes les règles se trouvent être "NotApplicable" à la demande de décision, la politique doit alors s'évaluer comme "NotApplicable".

Si une erreur survient lors de l'évaluation de la cible ou de la condition d'une règle qui contient un effet de "Permit", l'évaluation doit alors continuer en cherchant un résultat de "Permit". Si aucune autre règle ne s'évalue comme "Permit", la politique doit alors s'évaluer comme "Indeterminate", avec l'état d'erreur approprié.

Si au moins une règle s'évalue comme "Deny", si toutes les autres règles qui n'ont pas d'erreur d'évaluation s'évaluent comme "Deny" ou "NotApplicable" et si toutes les règles qui ont des erreurs d'évaluation contiennent une valeur d'effet de "Deny", la politique doit alors s'évaluer comme "Deny".

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de règles.

```
Decision permitOverridesRuleCombiningAlgorithm(Rule rule[])
{
   Boolean atLeastOneError  = false;
   Boolean potentialPermit  = false;
   Boolean atLeastOneDeny   = false;
   for( i=0 ; i < lengthOf(rule) ; i++ )
   {
          Decision decision = evaluate(rule[i]);
          if (decision == Deny)
          {
                 atLeastOneDeny = true;
                 continue;
          }
          if (decision == Permit)
          {
                 return Permit;
          }
          if (decision == NotApplicable)
          {
                 continue;
          }
          if (decision == Indeterminate)
          {
                 atLeastOneError = true;

                 if (effect(rule[i]) == Permit)
                 {
                        potentialPermit = true;
                 }
                 continue;
          }
   }
   if (potentialPermit)
   {
          return Indeterminate;
   }
   if (atLeastOneDeny)
   {
          return Deny;
   }
   if (atLeastOneError)
   {
          return Indeterminate;
   }
   return NotApplicable;
}
```

C.3.2 Le présent paragraphe définit l'algorithme de combinaison de politiques "Permit-overrides" d'un ensemble de politiques.

Dans tout l'ensemble de politiques dans l'ensemble de politiques, si une politique s'évalue comme "Permit", le résultat de la combinaison de politiques doit être "Permit". En d'autres termes, "Permit" a la préséance, indépendamment du résultat de l'évaluation des autres politiques dans l'ensemble de politiques. Si toutes les politiques se trouvent être "NotApplicable" à la demande de décision, l'ensemble de politiques doit alors s'évaluer comme "NotApplicable".

Si une erreur survient lors de l'évaluation de la cible d'une politique, si une référence à une politique est considérée comme invalide ou si l'évaluation de politique a un résultat "Indeterminate", l'ensemble de politiques doit alors s'évaluer comme "Indeterminate", avec l'état d'erreur approprié, pourvu qu'aucune autre politique ne s'évalue comme "Permit" ou "Deny".

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de politique.

```
Decision permitOverridesPolicyCombiningAlgorithm(Policy policy[])
{
   Boolean atLeastOneError = false;
   Boolean atLeastOneDeny  = false;
   for( i=0 ; i < lengthOf(policy) ; i++ )
   {
          Decision decision = evaluate(policy[i]);
          if (decision == Deny)
          {
                atLeastOneDeny = true;
                continue;
          }
          if (decision == Permit)
          {
                return Permit;
          }
          if (decision == NotApplicable)
          {
                continue;
          }
          if (decision == Indeterminate)
          {
                atLeastOneError = true;
                continue;
          }
   }
   if (atLeastOneDeny)
   {
          return Deny;
   }
   if (atLeastOneError)
   {
          return Indeterminate;
   }
   return NotApplicable;
}
```

Les obligations des politiques individuelles doivent être combinées comme décrit au § 7.6.14.

C.4 Ordered-permit-overrides

L'alinéa suivant définit l'algorithme de combinaison de règles "Ordered-permit-overrides" d'une politique.

Le comportement de cet algorithme est identique à celui de l'algorithme de combinaison de règles Permit-overrides avec une exception. L'ordre dans lequel la collection de règles est évaluée doit respecter l'ordre tel qu'il figure dans la politique.

L'alinéa suivant définit l'algorithme de combinaison de politiques "Ordered-permit-overrides" d'un ensemble de politiques.

Le comportement de cet algorithme est identique à celui de l'algorithme de combinaison de politiques Permit-overrides avec une exception. L'ordre dans lequel la collection de politiques est évaluée doit respecter l'ordre tel qu'il figure dans l'ensemble de politiques.

C.5 First-applicable

C.5.1 Le présent paragraphe définit l'algorithme de combinaison de règles "First-Applicable " d'une politique.

Chaque règle doit être évaluée dans l'ordre dans lequel elle figure dans la politique. Pour une règle particulière, si la cible correspond et si la condition est évaluée comme "True", l'évaluation de la politique doit alors s'arrêter et l'effet correspondant de la règle doit être le résultat de l'évaluation de la politique (c'est-à-dire, "Permit" ou "Deny"). Pour une règle particulière choisie dans l'évaluation, si la cible est évaluée à "False" ou si la condition est évaluée à "False", la prochaine règle dans l'ordre doit alors être évaluée. Si aucune autre règle n'existe à la suite, la politique doit alors s'évaluer comme "NotApplicable".

Si une erreur survient lors de l'évaluation de la cible ou de la condition d'une règle, l'évaluation doit alors s'arrêter, et la politique doit s'évaluer comme "Indeterminate", avec l'état d'erreur approprié.

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de règles.

```
Decision firstApplicableEffectRuleCombiningAlgorithm(Rule rule[])
{
   for( i = 0 ; i < lengthOf(rule) ; i++ )
   {
          Decision decision = evaluate(rule[i]);
          if (decision == Deny)
          {
                return Deny;
          }
          if (decision == Permit)
          {
                return Permit;
          }
          if (decision == NotApplicable)
          {
                continue;
          }
          if (decision == Indeterminate)
          {
                return Indeterminate;
          }
   }
   return NotApplicable;
}
```

C.5.2 Le présent paragraphe définit l'algorithme de combinaison de politiques "First-applicable" d'un ensemble de politiques.

Chaque politique est évaluée dans l'ordre dans lequel elle apparaît dans l'ensemble de politiques. Pour une politique particulière, si la cible s'évalue comme "True" et si la politique s'évalue à une valeur déterminée de "Permit" ou "Deny", l'évaluation doit alors s'arrêter et l'ensemble de politiques doit s'évaluer comme la valeur d'effet de cette politique. Pour une politique particulière, si la cible s'évalue comme "False", ou si la politique s'évalue comme "NotApplicable", la politique suivante dans l'ordre doit alors être évaluée. S'il n'existe pas d'autre politique à la suite, l'ensemble de politiques doit alors s'évaluer comme "NotApplicable".

Si une erreur devait survenir lors de l'évaluation de la cible, ou lors de l'évaluation d'une politique spécifique, la référence à cette politique est considérée comme invalide, ou la politique elle-même s'évalue comme "Indeterminate", l'évaluation de l'algorithme de combinaison de politiques doit s'arrêter, et l'ensemble de politiques doit s'évaluer comme "Indeterminate" avec un état d'erreur approprié.

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de politiques.

```
Decision firstApplicableEffectPolicyCombiningAlgorithm(Policy policy[])
{
    for( i = 0 ; i < lengthOf(policy) ; i++ )
    {
        Decision decision = evaluate(policy[i]);
        if(decision == Deny)
        {
            return Deny;
        }
        if(decision == Permit)
        {
            return Permit;
        }
        if (decision == NotApplicable)
        {
            continue;
        }
        if (decision == Indeterminate)
        {
            return Indeterminate;
        }
    }
    return NotApplicable;
}
```

Les obligations des politiques individuelles doivent être combinées comme décrit au § 7.6.14.

C.6 Only-one-applicable

Le présent paragraphe définit l'algorithme de combinaison de politiques "Only-one-applicable" d'un ensemble de politiques.

Dans tout l'ensemble de politiques de l'ensemble de politiques, si aucune politique n'est considérée comme applicable au titre de sa cible, le résultat de l'algorithme de combinaison de politiques doit être "NotApplicable". Si plus d'une politique est considérée comme applicable au titre de sa cible, le résultat de l'algorithme de combinaison de politiques doit être "Indeterminate".

Si une seule politique est considérée applicable par l'évaluation de sa cible, le résultat de l'algorithme de combinaison de politiques doit alors être le résultat de l'évaluation de la politique.

Si une erreur survient lors de l'évaluation de la cible d'une politique, ou si une référence à une politique est considérée comme invalide ou si l'évaluation de politique résulte en "Indeterminate", l'ensemble de politiques doit alors s'évaluer comme "Indeterminate", avec l'état d'erreur approprié.

Le pseudo-code suivant représente la stratégie d'évaluation de cet algorithme de combinaison de politiques.

```
Decision onlyOneApplicablePolicyPolicyCombiningAlogrithm(Policy policy[])
{
  Boolean          atLeastOne     = false;
  Policy           selectedPolicy = null;
  ApplicableResult appResult;

  for ( i = 0; i < lengthOf(policy) ; i++ )
  {
     appResult = isApplicable(policy[I]);

     if ( appResult == Indeterminate )
     {
         return Indeterminate;
     }
```

```
    if( appResult == Applicable )
    {
        if ( atLeastOne )
        {
            return Indeterminate;
        }
        else
        {
            atLeastOne     = true;
            selectedPolicy = policy[i];
        }
    }
    if ( appResult == NotApplicable )
    {
        continue;
    }
  }
  if ( atLeastOne )
  {
     return evaluate(selectedPolicy);
  }
  else
  {
     return NotApplicable;
  }
}
```

Annexe D

Schéma XACML

D.1 Schéma de contexte XACML

Le présent paragraphe donne le schéma de contexte XACML.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:import namespace="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
   schemaLocation="http://docs.oasis-open.org/xacml/access_control-xacml-2.0-policy-
schema-os.xsd"/>
   <!--   -->
   <xs:element name="Request" type="xacml-context:RequestType"/>
   <xs:complexType name="RequestType">
          <xs:sequence>
                 <xs:element ref="xacml-context:Subject" maxOccurs="unbounded"/>
                 <xs:element ref="xacml-context:Resource" maxOccurs="unbounded"/>
                 <xs:element ref="xacml-context:Action"/>
                 <xs:element ref="xacml-context:Environment"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="Response" type="xacml-context:ResponseType"/>
   <xs:complexType name="ResponseType">
          <xs:sequence>
                 <xs:element ref="xacml-context:Result" maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="Subject" type="xacml-context:SubjectType"/>
   <xs:complexType name="SubjectType">
          <xs:sequence>
                 <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="SubjectCategory" type="xs:anyURI"
          default="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="Resource" type="xacml-context:ResourceType"/>
   <xs:complexType name="ResourceType">
          <xs:sequence>
                 <xs:element ref="xacml-context:ResourceContent" minOccurs="0"/>
                 <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="ResourceContent" type="xacml-context:ResourceContentType"/>
   <xs:complexType name="ResourceContentType" mixed="true">
          <xs:sequence>
                 <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:anyAttribute namespace="##any" processContents="lax"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="Action" type="xacml-context:ActionType"/>
   <xs:complexType name="ActionType">
          <xs:sequence>
```

```
                <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="Environment" type="xacml-context:EnvironmentType"/>
   <xs:complexType name="EnvironmentType">
          <xs:sequence>
                <xs:element ref="xacml-context:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="Attribute" type="xacml-context:AttributeType"/>
   <xs:complexType name="AttributeType">
          <xs:sequence>
                <xs:element ref="xacml-context:AttributeValue"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="AttributeId" type="xs:anyURI" use="required"/>
          <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
          <xs:attribute name="Issuer" type="xs:string" use="optional"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="AttributeValue" type="xacml-context:AttributeValueType"/>
   <xs:complexType name="AttributeValueType" mixed="true">
          <xs:sequence>
                <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:anyAttribute namespace="##any" processContents="lax"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="Result" type="xacml-context:ResultType"/>
   <xs:complexType name="ResultType">
          <xs:sequence>
                <xs:element ref="xacml-context:Decision"/>
                <xs:element ref="xacml-context:Status" minOccurs="0"/>
                <xs:element ref="xacml:Obligations" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="ResourceId" type="xs:string" use="optional"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="Decision" type="xacml-context:DecisionType"/>
   <xs:simpleType name="DecisionType">
          <xs:restriction base="xs:string">
                <xs:enumeration value="Permit"/>
                <xs:enumeration value="Deny"/>
                <xs:enumeration value="Indeterminate"/>
                <xs:enumeration value="NotApplicable"/>
          </xs:restriction>
   </xs:simpleType>
   <!-- -->
   <xs:element name="Status" type="xacml-context:StatusType"/>
   <xs:complexType name="StatusType">
          <xs:sequence>
                <xs:element ref="xacml-context:StatusCode"/>
                <xs:element ref="xacml-context:StatusMessage" minOccurs="0"/>
                <xs:element ref="xacml-context:StatusDetail" minOccurs="0"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="StatusCode" type="xacml-context:StatusCodeType"/>
   <xs:complexType name="StatusCodeType">
          <xs:sequence>
                <xs:element ref="xacml-context:StatusCode" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Value" type="xs:anyURI" use="required"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="StatusMessage" type="xs:string"/>
```

```
   <!-- -->
   <xs:element name="StatusDetail" type="xacml-context:StatusDetailType"/>
   <xs:complexType name="StatusDetailType">
          <xs:sequence>
                 <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="MissingAttributeDetail" type="xacml-
context:MissingAttributeDetailType"/>
   <xs:complexType name="MissingAttributeDetailType">
          <xs:sequence>
                 <xs:element ref="xacml-context:AttributeValue" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="AttributeId" type="xs:anyURI" use="required"/>
          <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
          <xs:attribute name="Issuer" type="xs:string" use="optional"/>
   </xs:complexType>
   <!-- -->
</xs:schema>
```

D.2 Schéma de politique

Le présent paragraphe fournit le schéma de politique XACML.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <!-- -->
   <xs:element name="PolicySet" type="xacml:PolicySetType"/>
   <xs:complexType name="PolicySetType">
          <xs:sequence>
                 <xs:element ref="xacml:Description" minOccurs="0"/>
                 <xs:element ref="xacml:PolicySetDefaults" minOccurs="0"/>
                 <xs:element ref="xacml:Target"/>
                 <xs:choice minOccurs="0" maxOccurs="unbounded">
                        <xs:element ref="xacml:PolicySet"/>
                        <xs:element ref="xacml:Policy"/>
                        <xs:element ref="xacml:PolicySetIdReference"/>
                        <xs:element ref="xacml:PolicyIdReference"/>
                        <xs:element ref="xacml:CombinerParameters"/>
                        <xs:element ref="xacml:PolicyCombinerParameters"/>
                        <xs:element ref="xacml:PolicySetCombinerParameters"/>
                 </xs:choice>
                 <xs:element ref="xacml:Obligations" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="PolicySetId" type="xs:anyURI" use="required"/>
          <xs:attribute name="Version" type="xacml:VersionType" default="1.0"/>
          <xs:attribute name="PolicyCombiningAlgId" type="xs:anyURI" use="required"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="CombinerParameters" type="xacml:CombinerParametersType"/>
   <xs:complexType name="CombinerParametersType">
          <xs:sequence>
                 <xs:element ref="xacml:CombinerParameter" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="CombinerParameter" type="xacml:CombinerParameterType"/>
   <xs:complexType name="CombinerParameterType">
          <xs:sequence>
                 <xs:element ref="xacml:AttributeValue"/>
          </xs:sequence>
          <xs:attribute name="ParameterName" type="xs:string" use="required"/>
   </xs:complexType>
```

```
   <!-- -->
   <xs:element name="RuleCombinerParameters"
type="xacml:RuleCombinerParametersType"/>
   <xs:complexType name="RuleCombinerParametersType">
          <xs:complexContent>
                 <xs:extension base="xacml:CombinerParametersType">
                        <xs:attribute name="RuleIdRef" type="xs:string"
use="required"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="PolicyCombinerParameters"
type="xacml:PolicyCombinerParametersType"/>
   <xs:complexType name="PolicyCombinerParametersType">
          <xs:complexContent>
                 <xs:extension base="xacml:CombinerParametersType">
                        <xs:attribute name="PolicyIdRef" type="xs:anyURI"
use="required"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="PolicySetCombinerParameters"
type="xacml:PolicySetCombinerParametersType"/>
   <xs:complexType name="PolicySetCombinerParametersType">
          <xs:complexContent>
                 <xs:extension base="xacml:CombinerParametersType">
                        <xs:attribute name="PolicySetIdRef" type="xs:anyURI"
use="required"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="PolicySetIdReference" type="xacml:IdReferenceType"/>
   <xs:element name="PolicyIdReference" type="xacml:IdReferenceType"/>
   <!-- -->
   <xs:element name="PolicySetDefaults" type="xacml:DefaultsType"/>
   <xs:element name="PolicyDefaults" type="xacml:DefaultsType"/>
   <xs:complexType name="DefaultsType">
          <xs:sequence>
                 <xs:choice>
                        <xs:element ref="xacml:XPathVersion"/>
                 </xs:choice>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="XPathVersion" type="xs:anyURI"/>
   <!-- -->
   <xs:complexType name="IdReferenceType">
          <xs:simpleContent>
                 <xs:extension base="xs:anyURI">
                        <xs:attribute name="Version" type="xacml:VersionMatchType"
use="optional"/>
                        <xs:attribute name="EarliestVersion"
type="xacml:VersionMatchType" use="optional"/>
                        <xs:attribute name="LatestVersion"
type="xacml:VersionMatchType" use="optional"/>
                 </xs:extension>
          </xs:simpleContent>
   </xs:complexType>
   <!-- -->
   <xs:simpleType name="VersionType">
          <xs:restriction base="xs:string">
                 <xs:pattern value="(\d+\.)*\d+"/>
          </xs:restriction>
   </xs:simpleType>
   <!-- -->
   <xs:simpleType name="VersionMatchType">
          <xs:restriction base="xs:string">
                 <xs:pattern value="((\d+|\*)\.)*(\d+|\*|\+)"/>
```

```
      </xs:restriction>
</xs:simpleType>
<!-- -->
<xs:element name="Policy" type="xacml:PolicyType"/>
<xs:complexType name="PolicyType">
      <xs:sequence>
            <xs:element ref="xacml:Description" minOccurs="0"/>
            <xs:element ref="xacml:PolicyDefaults" minOccurs="0"/>
            <xs:element ref="xacml:CombinerParameters" minOccurs="0"/>
            <xs:element ref="xacml:Target"/>
            <xs:choice maxOccurs="unbounded">
                  <xs:element ref="xacml:CombinerParameters" minOccurs="0"/>
                  <xs:element ref="xacml:RuleCombinerParameters" minOccurs="0"/>
                  <xs:element ref="xacml:VariableDefinition"/>
                  <xs:element ref="xacml:Rule"/>
            </xs:choice>
            <xs:element ref="xacml:Obligations" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="PolicyId" type="xs:anyURI" use="required"/>
      <xs:attribute name="Version" type="xacml:VersionType" default="1.0"/>
      <xs:attribute name="RuleCombiningAlgId" type="xs:anyURI" use="required"/>
</xs:complexType>
<!-- -->
<xs:element name="Description" type="xs:string"/>
<!-- -->
<xs:element name="Rule" type="xacml:RuleType"/>
<xs:complexType name="RuleType">
      <xs:sequence>
            <xs:element ref="xacml:Description" minOccurs="0"/>
            <xs:element ref="xacml:Target" minOccurs="0"/>
            <xs:element ref="xacml:Condition" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="RuleId" type="xs:string" use="required"/>
      <xs:attribute name="Effect" type="xacml:EffectType" use="required"/>
</xs:complexType>
<!-- -->
<xs:simpleType name="EffectType">
      <xs:restriction base="xs:string">
            <xs:enumeration value="Permit"/>
            <xs:enumeration value="Deny"/>
      </xs:restriction>
</xs:simpleType>
<!-- -->
<xs:element name="Target" type="xacml:TargetType"/>
<xs:complexType name="TargetType">
      <xs:sequence>
            <xs:element ref="xacml:Subjects" minOccurs="0"/>
            <xs:element ref="xacml:Resources" minOccurs="0"/>
            <xs:element ref="xacml:Actions" minOccurs="0"/>
            <xs:element ref="xacml:Environments" minOccurs="0"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Subjects" type="xacml:SubjectsType"/>
<xs:complexType name="SubjectsType">
      <xs:sequence>
            <xs:element ref="xacml:Subject" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Subject" type="xacml:SubjectType"/>
<xs:complexType name="SubjectType">
      <xs:sequence>
            <xs:element ref="xacml:SubjectMatch" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Resources" type="xacml:ResourcesType"/>
<xs:complexType name="ResourcesType">
      <xs:sequence>
            <xs:element ref="xacml:Resource" maxOccurs="unbounded"/>
```

```
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Resource" type="xacml:ResourceType"/>
<xs:complexType name="ResourceType">
      <xs:sequence>
            <xs:element ref="xacml:ResourceMatch" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Actions" type="xacml:ActionsType"/>
<xs:complexType name="ActionsType">
      <xs:sequence>
            <xs:element ref="xacml:Action" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Action" type="xacml:ActionType"/>
<xs:complexType name="ActionType">
      <xs:sequence>
            <xs:element ref="xacml:ActionMatch" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Environments" type="xacml:EnvironmentsType"/>
<xs:complexType name="EnvironmentsType">
      <xs:sequence>
            <xs:element ref="xacml:Environment" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="Environment" type="xacml:EnvironmentType"/>
<xs:complexType name="EnvironmentType">
      <xs:sequence>
            <xs:element ref="xacml:EnvironmentMatch" maxOccurs="unbounded"/>
      </xs:sequence>
</xs:complexType>
<!-- -->
<xs:element name="SubjectMatch" type="xacml:SubjectMatchType"/>
<xs:complexType name="SubjectMatchType">
      <xs:sequence>
            <xs:element ref="xacml:AttributeValue"/>
            <xs:choice>
                  <xs:element ref="xacml:SubjectAttributeDesignator"/>
                  <xs:element ref="xacml:AttributeSelector"/>
            </xs:choice>
      </xs:sequence>
      <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
</xs:complexType>
<!-- -->
<xs:element name="ResourceMatch" type="xacml:ResourceMatchType"/>
<xs:complexType name="ResourceMatchType">
      <xs:sequence>
            <xs:element ref="xacml:AttributeValue"/>
            <xs:choice>
                  <xs:element ref="xacml:ResourceAttributeDesignator"/>
                  <xs:element ref="xacml:AttributeSelector"/>
            </xs:choice>
      </xs:sequence>
      <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
</xs:complexType>
<!-- -->
<xs:element name="ActionMatch" type="xacml:ActionMatchType"/>
<xs:complexType name="ActionMatchType">
      <xs:sequence>
            <xs:element ref="xacml:AttributeValue"/>
            <xs:choice>
                  <xs:element ref="xacml:ActionAttributeDesignator"/>
                  <xs:element ref="xacml:AttributeSelector"/>
            </xs:choice>
      </xs:sequence>
```

```
          <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="EnvironmentMatch" type="xacml:EnvironmentMatchType"/>
   <xs:complexType name="EnvironmentMatchType">
          <xs:sequence>
                <xs:element ref="xacml:AttributeValue"/>
                <xs:choice>
                      <xs:element ref="xacml:EnvironmentAttributeDesignator"/>
                      <xs:element ref="xacml:AttributeSelector"/>
                </xs:choice>
          </xs:sequence>
          <xs:attribute name="MatchId" type="xs:anyURI" use="required"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="VariableDefinition" type="xacml:VariableDefinitionType"/>
   <xs:complexType name="VariableDefinitionType">
          <xs:sequence>
                <xs:element ref="xacml:Expression"/>
          </xs:sequence>
          <xs:attribute name="VariableId" type="xs:string" use="required"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="Expression" type="xacml:ExpressionType" abstract="true"/>
   <xs:complexType name="ExpressionType" abstract="true"/>
   <!-- -->
   <xs:element name="VariableReference"
   type="xacml:VariableReferenceType" substitutionGroup="xacml:Expression"/>
   <xs:complexType name="VariableReferenceType">
          <xs:complexContent>
                <xs:extension base="xacml:ExpressionType">
                      <xs:attribute name="VariableId" type="xs:string"
use="required"/>
                </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="AttributeSelector"
   type="xacml:AttributeSelectorType" substitutionGroup="xacml:Expression"/>
   <xs:complexType name="AttributeSelectorType">
          <xs:complexContent>
                <xs:extension base="xacml:ExpressionType">
                      <xs:attribute name="RequestContextPath" type="xs:string"
use="required"/>
                      <xs:attribute name="DataType" type="xs:anyURI"
use="required"/>
                      <xs:attribute name="MustBePresent" type="xs:boolean"
use="optional" default="false"/>
                </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="ResourceAttributeDesignator"
   type="xacml:AttributeDesignatorType" substitutionGroup="xacml:Expression"/>
   <xs:element name="ActionAttributeDesignator"
   type="xacml:AttributeDesignatorType" substitutionGroup="xacml:Expression"/>
   <xs:element name="EnvironmentAttributeDesignator"
   type="xacml:AttributeDesignatorType" substitutionGroup="xacml:Expression"/>
   <!-- -->
   <xs:complexType name="AttributeDesignatorType">
          <xs:complexContent>
                <xs:extension base="xacml:ExpressionType">
                      <xs:attribute name="AttributeId" type="xs:anyURI"
use="required"/>
                      <xs:attribute name="DataType" type="xs:anyURI"
use="required"/>
                      <xs:attribute name="Issuer" type="xs:string" use="optional"/>
                      <xs:attribute name="MustBePresent" type="xs:boolean"
use="optional" default="false"/>
                </xs:extension>
          </xs:complexContent>
```

```
   </xs:complexType>
   <!-- -->
   <xs:element name="SubjectAttributeDesignator"
   type="xacml:SubjectAttributeDesignatorType" substitutionGroup="xacml:Expression"/>
   <xs:complexType name="SubjectAttributeDesignatorType">
          <xs:complexContent>
                 <xs:extension base="xacml:AttributeDesignatorType">
                        <xs:attribute name="SubjectCategory" type="xs:anyURI"
                        use="optional" default="urn:oasis:names:tc:xacml:1.0:subject-
category:access-subject"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="AttributeValue" type="xacml:AttributeValueType"
          substitutionGroup="xacml:Expression"/>
   <xs:complexType name="AttributeValueType" mixed="true">
          <xs:complexContent mixed="true">
                 <xs:extension base="xacml:ExpressionType">
                        <xs:sequence>
                               <xs:any namespace="##any" processContents="lax"
minOccurs="0"
                                      maxOccurs="unbounded"/>
                        </xs:sequence>
                        <xs:attribute name="DataType" type="xs:anyURI"
use="required"/>
                        <xs:anyAttribute namespace="##any" processContents="lax"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="Function" type="xacml:FunctionType"
substitutionGroup="xacml:Expression"/>
   <xs:complexType name="FunctionType">
          <xs:complexContent>
                 <xs:extension base="xacml:ExpressionType">
                        <xs:attribute name="FunctionId" type="xs:anyURI"
use="required"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="Condition" type="xacml:ConditionType"/>
   <xs:complexType name="ConditionType">
          <xs:sequence>
                 <xs:element ref="xacml:Expression"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
   <xs:element name="Apply" type="xacml:ApplyType"
substitutionGroup="xacml:Expression"/>
   <xs:complexType name="ApplyType">
          <xs:complexContent>
                 <xs:extension base="xacml:ExpressionType">
                        <xs:sequence>
                               <xs:element ref="xacml:Expression" minOccurs="0"
maxOccurs="unbounded"/>
                        </xs:sequence>
                        <xs:attribute name="FunctionId" type="xs:anyURI"
use="required"/>
                 </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
   <xs:element name="Obligations" type="xacml:ObligationsType"/>
   <xs:complexType name="ObligationsType">
          <xs:sequence>
                 <xs:element ref="xacml:Obligation" maxOccurs="unbounded"/>
          </xs:sequence>
   </xs:complexType>
   <!-- -->
```

```
   <xs:element name="Obligation" type="xacml:ObligationType"/>
   <xs:complexType name="ObligationType">
          <xs:sequence>
                <xs:element ref="xacml:AttributeAssignment" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:attribute name="ObligationId" type="xs:anyURI" use="required"/>
          <xs:attribute name="FulfillOn" type="xacml:EffectType" use="required"/>
   </xs:complexType>
   <!-- -->
   <xs:element name="AttributeAssignment" type="xacml:AttributeAssignmentType"/>
   <xs:complexType name="AttributeAssignmentType" mixed="true">
          <xs:complexContent mixed="true">
                <xs:extension base="xacml:AttributeValueType">
                      <xs:attribute name="AttributeId" type="xs:anyURI"
use="required"/>
                </xs:extension>
          </xs:complexContent>
   </xs:complexType>
   <!-- -->
</xs:schema>
```

D.3 Schéma de protocole SAML XACML

Le présent paragraphe fournit le schéma de protocole SAML de XACML.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
    targetNamespace="urn:oasis:xacml:2.0:saml:protocol:schema:os"
    xmlns:xs="http://www.23.org/2001/XMLSchema"
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
    xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
    elementFormDefault="unqualified"
    attributeFormDefault="unqualified"
    blockDefault="substitution"
    version="2.0">
  <xs:import namespace="urn:oasis:names:tc:SAML:2.0:assertion"
      schemaLocation="http://www.oasis-
open.org/committees/tc_home.php?wg_abbrev=security"/>
  <xs:import namespace="urn:oasis:names:tc:SAML:2.0:protocol"
      schemaLocation="http://www.oasis-
open.org/committees/tc_home.php?wg_abbrev=security"/>
  <xs:import namespace="urn:oasis:names:tc:xacml:2.0:context:schema:os"
      schemaLocation="http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-
context-schema-os.xsd"/>
  <xs:import namespace="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
      schemaLocation="http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-
policy-schema-os.xsd"/>
  <xs:annotation>
    <xs:documentation>
        Document identifier: access_control-xacml-2.0-saml-protocol-schema-os.xsd
        Location: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-
protocol-schema-os.xsd
    </xs:documentation>
  </xs:annotation>
  <!--    -->
  <xs:element name="XACMLAuthzDecisionQuery"
           type="XACMLAuthzDecisionQueryType"/>
  <xs:complexType name="XACMLAuthzDecisionQueryType">
    <xs:complexContent>
      <xs:extension base="samlp:RequestAbstractType">
        <xs:sequence>
          <xs:element ref="xacml-context:Request"/>
        </xs:sequence>
        <xs:attribute name="InputContextOnly"
                      type="boolean"
                      use="optional"
```

```
                      default="false"/>
        <xs:attribute name="ReturnContext"
                      type="boolean"
                      use="optional"
                      default="false"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <!--    -->
  <xs:element name="XACMLPolicyQuery"
           type="XACMLPolicyQueryType"/>
  <xs:complexType name="XACMLPolicyQueryType">
    <xs:complexContent>
      <xs:extension base="samlp:RequestAbstractType">
        <xs:choice minOccurs="0" maxOccurs="unbounded">>
          <xs:element ref="xacml-context:Request"/>
          <xs:element ref="xacml:Target"/>
          <xs:element ref="xacml:PolicySetIdReference"/>
          <xs:element ref="xacml:PolicyIdReference"/>
        </xs:choice>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</schema>
```

D.4 Schéma d'assertion SAML XACML

Le présent paragraphe fournit le schéma d'assertions SAML de XACML.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
    targetNamespace="urn:oasis:xacml:2.0:saml:assertion:schema:os"
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:xs="http://www.23.org/2001/XMLSchema"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
    xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
    elementFormDefault="unqualified"
    attributeFormDefault="unqualified"
    blockDefault="substitution"
    version="2.0">
  <xs:import namespace="urn:oasis:names:tc:SAML:2.0:assertion"
      schemaLocation="http://www.oasis-
open.org/committees/tc_home.php?wg_abbrev=security"/>
  <xs:import namespace="urn:oasis:names:tc:SAML:2.0:protocol"
      schemaLocation="http://www.oasis-
open.org/committees/tc_home.php?wg_abbrev=security"/>
  <xs:import namespace="urn:oasis:names:tc:xacml:2.0:context:schema:os"
      schemaLocation="http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-
context-schema-os.xsd"/>
  <xs:import namespace="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
      schemaLocation="http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-
policy-schema-os.xsd"/>
  <xs:annotation>
    <xs:documentation>
        Document identifier: access_control-xacml-2.0-saml-assertion-schema-cd-02.xsd
        Location: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-
assertion-schema-cd-os.xsd
    </xs:documentation>
  </xs:annotation>
  <!--    -->
  <xs:element name="XACMLAuthzDecisionStatement"
           type="XACMLAuthzDecisionStatementType"/>
  <xs:complexType name="XACMLAuthzDecisionStatementType">
    <xs:complexContent>
      <xs:extension base="samlp:StatementAbstractType">
        <xs:sequence>
          <xs:element ref="xacml-context:Response"/>
          <xs:element ref="xacml-context:Request"  MinOccurs="0"/>
```

```
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <!--    -->
  <xs:element name="XACMLPolicyStatement"
           type="XACMLPolicyStatementType"/>
  <xs:complexType name="XACMLPolicyStatementType">
    <xs:complexContent>
      <xs:extension base="samlp:StatementAbstractType">
        <xs:choice minOccurs="0" maxOccurs="unbounded">>
          <xs:element ref="xacml:Policy"/>
          <xs:element ref="xacml:PolicySet"/>
        </xs:choice>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</schema>
```

Appendice I

Considérations sur la sécurité

Le présent appendice identifie les scénarios possibles de compromission de la sécurité et de la confidentialité qui devraient être pris en considération lors de l'implémentation de systèmes fondés sur XACML. Il appartient au développeur de décider si ces scénarios de compromission présentent un caractère pratique dans leur environnement et de choisir les sauvegardes appropriées.

I.1 Modèle de menace

Il est supposé ici que l'adversaire a accès au canal de communication entre les acteurs XACML et est capable d'interpréter, insérer, supprimer et modifier des messages ou parties de messages.

De plus, un acteur peut utiliser de façon malveillante des informations provenant d'un message précédent dans des transactions ultérieures. On supposera de plus que la fiabilité des règles et des politiques correspond à celle des acteurs qui les créent et les utilisent. Et donc il incombe à chaque acteur d'établir la confiance appropriée envers les autres acteurs sur lesquels elle repose. Les mécanismes d'établissement de la confiance sont en dehors du domaine d'application de la présente Recommandation.

Les messages qui sont transmis entre les acteurs dans le modèle XACML sont susceptibles d'être attaqués par des tiers malveillants. D'autres points de vulnérabilité incluent le PEP, le PDP et le PAP. Alors que certaines de ces entités ne sont pas strictement dans le domaine d'application de la présente Recommandation, leur compromission pourrait conduire à la compromission du contrôle d'accès mis en application par le PEP.

Il vaut de noter que d'autres composants d'un système distribué peuvent être compromis, comme un système d'exploitation et le système de noms de domaine (DNS, *domain name system*) qui sont en dehors du domaine de discussion des modèles de menace. La compromission de ces composants peut aussi conduire à une violation de la politique.

Les paragraphes qui suivent précisent les scénarios spécifiques de compromission qui sont pertinents pour un système XACML.

I.1.1 Divulgation non autorisée

XACML ne spécifie aucun mécanisme inhérent pour protéger la confidentialité des messages échangés entre les acteurs. Donc, un adversaire pourrait observer les messages en transit. Dans certaines politiques de sécurité, la divulgation de ces informations est une violation. La divulgation des attributs ou des types de demande de décisions qu'un sujet soumet peut être une atteinte à la politique de confidentialité. Dans le secteur commercial, les conséquences d'une divulgation non autorisée de données personnelles peuvent aller d'ennuis pour le détenteur à l'emprisonnement et de grosses amendes dans le cas de données médicales ou financières.

La divulgation non autorisée est traitée par des sauvegardes de confidentialité.

I.1.2 Répétition de message

Une attaque en répétition de message est celle où l'adversaire enregistre et repasse des messages légitimes entre des acteurs XACML. Cette attaque peut conduire à des dénis de service, à l'utilisation d'informations périmées ou à la substitution d'identité.

La prévention des attaques en répétition requiert l'utilisation de sauvegardes de la fraîcheur du message.

Noter que le chiffrement du message n'atténue pas les attaques en répétition car le message est simplement repassé et n'a pas à être compris de l'adversaire.

I.1.3 Insertion de message

Une attaque d'insertion de message est celle dans laquelle l'adversaire insère des messages dans la séquence des messages entre les acteurs XACML.

La solution à une attaque d'insertion de message est d'utiliser l'authentification mutuelle et des sauvegardes de l'intégrité des séquences de message entre les acteurs. Il faut noter qu'utiliser seulement l'authentification mutuelle SSL n'est pas suffisant. Cela prouve seulement que l'autre partie est celle identifiée par le sujet du certificat X.509. Pour être efficace, il est nécessaire de confirmer que le sujet du certificat est autorisé à envoyer le message.

I.1.4 Suppression de message

Une attaque de suppression de message est celle où l'adversaire supprime les messages dans la séquence de messages entre les acteurs XACML. La suppression de message peut conduire à un déni de service. Cependant, un système XACML conçu de façon appropriée ne devrait pas prendre une décision d'autorisation incorrecte par suite d'une attaque de suppression de message.

La solution à une attaque de suppression de message est d'utiliser les sauvegardes d'intégrité de séquence de messages entre les acteurs.

I.1.5 Modification de message

Si un adversaire peut intercepter un message et changer son contenu, il peut alors être capable d'altérer une décision d'autorisation. Une sauvegarde d'intégrité de message peut empêcher la réussite d'une attaque de modification de message.

I.1.6 Résultats non applicables

Un résultat de "NotApplicable" signifie que le PDP n'a pas pu localiser une politique dont la cible corresponde aux informations contenues dans la demande de décision. En général, il est fortement recommandé qu'une politique d'effet "Deny" soit utilisée, de sorte que lorsqu'un PDP a retourné "NotApplicable", un résultat de "Deny" soit plutôt retourné.

Cependant, dans certains modèles de sécurité, tels que ceux qu'on trouve dans de nombreux serveurs de la toile, une décision d'autorisation de "NotApplicable" est traitée comme équivalente à "Permit". Il y a des considérations de sécurité particulières qui doivent être prises en compte pour que ce comportement soit sûr. Elles sont expliquées dans les alinéas qui suivent.

Si "NotApplicable" est traité comme "Permit", il est vital que les algorithmes de correspondance utilisés par la politique pour faire correspondre les éléments dans la demande de décision soient étroitement alignés sur la syntaxe des données utilisée par les applications qui vont soumettre la demande de décision. Un échec de correspondance aura pour résultat un "NotApplicable" et il sera traité comme "Permit". Ainsi un échec inattendu à établir la correspondance peut permettre des accès non désirés.

Les appareils de réponse http du commerce permettent que diverses syntaxes soient traitées de façon équivalente. Le caractère "%" peut être utilisé pour représenter des caractères par la valeur hexadécimale. Le chemin d'URL "/../" fournit plusieurs façons de spécifier la même valeur. Plusieurs ensembles de caractères peuvent être permis et, dans certains cas, le même caractère imprimé peut être représenté par différentes valeurs binaires. Si l'algorithme de correspondance utilisé par la politique n'est pas assez sophistiqué pour saisir ces différences, des accès non désirés peuvent être permis.

Traiter "NotApplicable" comme "Permit" ne peut être sûr que dans un environnement clos où toutes les applications qui formulent une demande de décision ont la garantie d'utiliser la syntaxe exacte attendue par les politiques. Dans un environnement plus ouvert, où les demandes de décision peuvent être reçues de la part d'applications qui utilisent toutes les syntaxes légales, il est fortement recommandé que "NotApplicable" ne soit pas traité comme "Permit" à moins que des règles de correspondance n'aient été très soigneusement conçues pour correspondre à toutes les entrées applicables possibles, indépendamment des variations de syntaxe ou de type. Un PEP doit refuser l'accès s'il ne reçoit pas une décision d'autorisation "Permit" explicite.

I.1.7 Règles négatives

Une règle négative est fondée sur un prédicat qui n'est pas "True". Si elles ne sont pas utilisées avec grand soin, les règles négatives peuvent conduire à une violation de politique, et donc certaines autorités recommandent de ne pas les utiliser. Cependant, des règles négatives peuvent être extrêmement efficaces dans certains cas, aussi XACML a choisi de les inclure. A tout le moins, il est recommandé de les utiliser avec grand soin et de les éviter si possible.

Une utilisation courante des règles négatives est pour refuser l'accès à un individu ou sous-groupe lorsque son adhésion à un groupe plus étendu lui permettrait autrement l'accès. Par exemple, on peut vouloir écrire une règle qui permette à tous les vice-présidents de voir les données financières non publiées, excepté Joe, qui n'est qu'un vice-président honorifique et peut être indiscret dans sa communication. Si on a le contrôle complet sur l'administration des attributs de sujet, une approche supérieure serait de définir "Vice-Président" et "Vice-Président honorifique" comme des groupes distincts et définir alors des règles en conséquence. Cependant, dans certains environnements, cette approche peut n'être pas faisable. (Il vaut la peine de noter en passant que, généralement parlant, se référer à des individus dans les règles n'est pas très cohérent. En général, on préfère des attributs partagés.)

Si on n'y veille pas, des règles négatives peuvent conduire à des violations de politique dans deux cas courants, à savoir quand des attributs sont supprimés et quand le groupe de base change. Un exemple d'attributs supprimés serait si nous avons une politique disant que l'accès devrait être permis, sauf si le sujet présente un risque d'insolvabilité. S'il est possible que l'attribut d'être un risque d'insolvabilité soit inconnu du PDP pour une raison quelconque, il peut en résulter un accès non autorisé. Dans certains environnements, le sujet peut être capable de supprimer la publication d'attributs par l'application de contrôles de confidentialité, ou le serveur ou réceptacle qui contient les informations peut être indisponible pour des raisons accidentelles ou intentionnelles.

Un exemple de changement de groupe de base serait celui où la politique du département d'ingénierie permet à tous de changer le code source du logiciel, sauf les secrétaires. Supposons maintenant que le département doit fusionner avec un autre département d'ingénierie et qu'on ait l'intention de maintenir la même politique. Cependant, le nouveau département inclut aussi des individus identifiés comme assistants administratifs, qui devraient être traités de la même façon que les secrétaires. A moins de changer la politique, on va involontairement leur permettre de changer le code source logiciel. Des problèmes de ce type sont faciles à éviter lorsqu'un seul individu administre toutes les politiques, mais lorsque l'administration est distribuée, comme le permet XACML, ce type de situation doit être explicitement empêché.

I.2 Sauvegardes

I.2.1 Authentification

L'authentification donne les moyens à une partie à une transaction de déterminer l'identité de l'autre partie à la transaction. L'authentification peut être dans une seule direction, ou elle peut être bilatérale.

Etant donné la nature sensible des systèmes de contrôle d'accès, il est important qu'un PEP authentifie l'identité du PDP auquel il envoie des demandes de décision. Autrement, il y a un risque qu'un adversaire puisse fournir des décisions d'autorisation fausses ou non valides, conduisant à une violation de politique.

Il est également important qu'un PDP authentifie l'identité du PEP et s'assure du niveau de confiance pour déterminer quelles données sensibles, s'il en est, devraient être passées. On devrait garder présent à l'esprit que même de simples réponses "Permit" ou "Deny" peuvent être exploitées si un adversaire a la possibilité de faire des demandes sans limite à un PDP.

De nombreuses techniques différentes peuvent être utilisées pour fournir l'authentification, comme un code colocalisé, un réseau privé, un VPN ou des signatures numériques. L'authentification peut aussi être effectuée au titre du protocole de communication utilisé pour échanger les contextes. Dans ce cas, l'authentification peut être effectuée au niveau du message ou au niveau de la session.

I.2.2 Administration de politique

Si le contenu des politiques est exposé en dehors du système de contrôle d'accès, des sujets potentiels peuvent utiliser ces informations pour déterminer comment acquérir un accès non autorisé.

Pour empêcher cette menace, le réceptacle utilisé pour la mémorisation des politiques peut lui-même exiger un contrôle d'accès. De plus, l'élément `<Status>` ne devrait être utilisé pour retourner les valeurs des attributs manquants que lorsque l'exposition des identités de ces attributs ne compromet pas la sécurité.

I.2.3 Confidentialité

Les mécanismes de confidentialité garantissent que le contenu d'un message ne peut être lu que par les receveurs prévus et par personne d'autre qui pourrait rencontrer le message pendant son transit. Il y a deux domaines où la confidentialité doit être examinée: l'une est la confidentialité durant la transmission; l'autre est la confidentialité dans un élément `<Policy>`.

I.2.3.1 Confidentialité de la communication

Dans certains environnements, il est réputé de bonne pratique de traiter toutes les données au sein d'un système de contrôle d'accès comme confidentielles. Dans d'autres environnements, les politiques peuvent être librement accessibles à la distribution, l'inspection et l'étude. L'idée sur laquelle repose une politique de secret de l'information est de rendre plus difficile à l'adversaire la connaissance des étapes qui seraient suffisantes pour obtenir un accès non autorisé. Quelle que soit l'approche choisie, la sécurité du système de contrôle d'accès ne devrait pas dépendre du secret de la politique.

Toutes les considérations de sécurité qui se rapportent à la transmission ou l'échange des éléments `<Policy>` de XACML sont en dehors du domaine d'application de la norme XACML. Alors qu'il est souvent important de s'assurer que l'intégrité et la confidentialité des éléments `<Policy>` est conservée lorsqu'ils sont échangés entre deux parties, le soin est laissé aux développeurs de déterminer les mécanismes appropriés pour leur environnement.

La confidentialité des communications peut être fournie par des mécanismes de confidentialité, tels que SSL. Utiliser un schéma point à point comme SSL peut conduire à d'autres faiblesses lorsque l'un des points d'extrémité est compromis.

I.2.3.2 Confidentialité de niveau de déclaration

Dans certains cas, une implémentation peut vouloir chiffrer seulement des parties d'un élément `<Policy>` XACML.

Encryption:2002 du W3C peut être utilisé pour chiffrer tout ou partie d'un document XML. Encryption:2002 du W3C est recommandé pour une utilisation avec XACML.

Il va sans dire que si un réceptacle est utilisé pour faciliter la communication de texte en clair (c'est-à-dire, non chiffré) de la politique entre le PAP et le PDP, on devrait utiliser un réceptacle de confiance pour mémoriser ces données sensibles.

I.2.4 Intégrité de la politique

La politique XACML, utilisée par le PDP pour évaluer le contexte de la demande, est le cœur du système. Donc, le maintient de son intégrité est essentiel. La maintenance de l'intégrité de la politique revêt deux aspects. L'un est de s'assurer que les éléments `<Policy>` n'ont pas été altérés depuis leur création par le PAP. L'autre est de s'assurer que des éléments `<Policy>` n'ont pas été insérés ou supprimés de l'ensemble des politiques.

Dans de nombreux cas, les deux aspects peuvent être couverts en s'assurant de l'intégrité des acteurs et en mettant en œuvre des mécanismes de niveau session pour sécuriser la communication entre les acteurs. Le choix des mécanismes appropriés relève de l'implémentation. Cependant, lorsque la politique est distribuée entre les organisations pour être appliquée à un stade ultérieur, ou lorsque la politique voyage avec la ressource protégée, il serait utile de signer la politique. Dans des cas semblables, il est recommandé d'utiliser la norme de syntaxe et de traitement de signature XML du W3C avec XACML.

Les signatures numériques ne devraient être utilisées que pour protéger l'intégrité des déclarations. Les signatures numériques ne devraient pas être utilisées comme méthode de choix ou d'évaluation de politique. C'est-à-dire que le PDP ne devrait pas demander une politique sur la base de la personne qui l'a signée ou du fait qu'elle ait été signée ou non (un tel critère de choix serait par lui-même une question de politique). Cependant, le PDP doit vérifier que la clé utilisée pour signer la politique est contrôlée par le producteur prétendu de la politique. Les moyens de le faire dépendent de la technique spécifique de signature choisie et sortent du domaine d'application de la présente Recommandation.

I.2.5 Identifiants de politique

Comme les politiques peuvent être référencées par leurs identifiants, il est de la responsabilité du PAP de s'assurer qu'ils sont uniques. La confusion entre les identifiants pourrait conduire à une mauvaise identification de la politique applicable. La présente Recommandation ne dit rien sur la question de savoir si un PAP doit générer un nouvel identifiant lorsqu'une politique est modifiée ou s'il doit utiliser le même identifiant dans la politique modifiée. Ceci est une affaire de pratique administrative. Cependant, il faut faire attention dans les deux cas. Si l'identifiant est réutilisé, il y a un risque que d'autres politiques ou ensembles de politiques qui y font référence en soient affectés. A l'inverse, si un nouvel identifiant est utilisé, ces autres politiques peuvent continuer à utiliser la politique précédente, tant qu'elle n'est pas supprimée. Dans les deux cas, le résultat peut n'être ce que voulait l'administrateur de la politique.

I.2.6 Modèle de confiance

Les discussions sur l'authentification, les sauvegardes d'intégrité et de confidentialité supposent nécessairement un modèle de confiance sous-jacent: comment un acteur peut en venir à croire qu'une clé donnée est associée de façon univoque à un acteur identifié, spécifique, de telle sorte que la clé puisse être utilisée pour chiffrer des données pour cet acteur ou vérifier des signatures (ou autres structures liées à l'intégrité) provenant de cet acteur? Il existe de nombreux types différents de modèles de confiance, qui incluent des hiérarchies strictes, des autorités réparties, la toile, le pontage et ainsi de suite.

Il vaut la peine de considérer les relations entre les divers acteurs du système de contrôle d'accès en termes d'existence ou non d'interdépendance.

- Aucune des entités du système d'autorisation ne dépend du PEP. Elles peuvent en collecter les données, par exemple les données d'authentification, mais elles ont elles-mêmes la charge de les vérifier.
- Le fonctionnement correct du système dépend de la capacité du PEP à mettre réellement en application des décisions de politique.
- Le PEP dépend du PDP pour évaluer correctement les politiques. Ceci implique en retour que le PDP soit approvisionné avec les entrées correctes. En dehors de cela, le PDP ne dépend pas du PEP.
- Le PDP dépend du PAP pour fournir les politiques appropriées. Le PAP ne dépend pas d'autres composants.

I.2.7 Confidentialité

Il est important de savoir que toute transaction qui survient dans le champ d'un contrôle d'accès peut révéler des informations privées sur les acteurs. Par exemple, si une politique XACML déclare que certaines données ne peuvent être lues que par des sujets ayant le statut de "Membre privilégié", toute transaction dans laquelle un sujet a un accès autorisé à ces données laisse échapper pour un adversaire l'information sur le statut du sujet. Les considérations de confidentialité peuvent donc conduire à des exigences de chiffrement et/ou de contrôle d'accès entourant la mise en application des instances de politique XACML elles-mêmes: des canaux dont la confidentialité est protégée pour les messages de demande/réponse de protocole, la protection des attributs de sujet dans le stockage et le transit, et ainsi de suite.

La sélection et l'utilisation de mécanismes de confidentialité appropriés à un environnement donné sortent du domaine d'application de XACML. La décision concernant comment et quand déployer de tels mécanismes relève du développeur associé à l'environnement.

Appendice II

Exemples pour XACML

Le présent appendice contient deux exemples d'utilisation de XACML à des fins d'illustration. Le premier exemple est relativement simple pour illustrer l'utilisation d'une cible, d'un contexte, de fonctions de correspondance et d'attributs de sujet. Le second exemple illustre en plus l'utilisation de l'algorithme de combinaison de règles, de conditions et d'obligations.

II.1 Exemple un

Le présent paragraphe contient le premier exemple.

II.1.1 Exemple de politique

Supposons qu'une société appelée Medi Corp (identifiée par son nom de domaine: med.example.com) a une politique de contrôle d'accès qui déclare, en français:

Tout utilisateur avec un nom de messagerie dans l'espace de nom "med.example.com" a l'autorisation d'effectuer toute action sur toute ressource.

Une politique XACML consiste en informations d'en-tête, un texte facultatif de description de la politique, une **cible**, une ou plusieurs **règles** et un ensemble facultatif d'**obligations**.

```
[a02]  <?xml version="1.0" encoding="UTF-8"?>
[a03]  <Policy
[a04]  xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
[a05]  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
[a06]  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-policy-schema-
os.xsd"
[a07]  PolicyId="urn:oasis:names:tc:example:SimplePolicy1"
[a08]  RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-
overrides">
[a09]   <Description>
[a10]    Medi Corp access control policy
[a11]   </Description>
[a12]   <Target/>
[a13]   <Rule
[a14]   RuleId= "urn:oasis:names:tc:xacml:2.0:example:SimpleRule1"
[a15]   Effect="Permit">
[a16]    <Description>
[a17]     Any subject with an e-mail name in the med.example.com domain
[a18]     can perform any action on any resource.
[a19]    </Description>
[a20]    <Target>
[a21]     <Subjects>
[a22]      <Subject>
[a23]       <SubjectMatch
[a24]       MatchId="urn:oasis:names:tc:xacml:1.0:function:rfc822Name-
match">
[a25]        <AttributeValue
[a26]        DataType="http://www.w3.org/2001/XMLSchema#string">
[a27]         med.example.com
[a28]        </AttributeValue>
[a29]        <SubjectAttributeDesignator
[a30]        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
[a31]        DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name"/>
[a32]       </SubjectMatch>
[a33]      </Subject>
[a34]     </Subjects>
[a35]    </Target>
[a36]   </Rule>
[a37]  </Policy>
```

[a02] est une étiquette de document XML standard qui indique quelle version de XML est utilisée et quel est le codage de caractères.

[a03] introduit la politique XACML elle-même.

[a04] – [a05] sont les déclarations d'espace de nom XML.

[a06] donne un URN pour le schéma de politiques XACML.

[a07] alloue un nom à cette instance de politique. Le nom d'une politique doit être unique pour un PDP donné de sorte qu'il n'y ait pas d'ambiguïté si une politique est référencée à partir d'une autre politique. L'attribut de `version` est omis, et il prend donc sa valeur par défaut de "1.0".

[a08] spécifie l'algorithme qui sera utilisé pour résoudre le résultat des diverses règles qui peuvent être dans la politique. L'algorithme de combinaison de règles deny-overrides spécifié ici dit que, si une règle évalue à "Deny", la politique doit alors retourner "Deny". Si une règle évalue à "Permit", la politique doit alors retourner "Permit". L'algorithme de combinaison de règles, qui est entièrement décrit à l'Annexe C, dit aussi ce qu'il convient de faire en cas d'erreur lors de l'évaluation d'une règle et ce qu'il faut faire avec les règles qui ne s'appliquent pas à une demande de décision particulière.

[a09] – [a11] fournit une description textuelle de la politique. Cette description est facultative.

[a12] décrit les demandes de décision auxquelles cette politique s'applique. Si le sujet, la ressource, l'action et l'environnement dans une demande de décision ne correspondent pas aux valeurs spécifiées dans la cible de la politique, le reste de la politique n'a pas besoin d'être évalué. Cette section de cible est utile pour créer un indice d'un ensemble de politiques. Dans cet exemple simple, la section cible dit que la politique est applicable à toute demande de décision.

[a13] introduit la seule et unique règle dans cette politique simple.

[a14] spécifie l'identifiant pour cette règle. Comme pour une politique, chaque règle doit avoir un identifiant unique (au moins unique pour tout PDP qui va utiliser la politique).

[a15] dit quel effet a cette règle si elle évalue comme "True". Les règles peuvent avoir un effet de "Permit" ou de "Deny". Dans ce cas, si la règle est satisfaite, elle évaluera à "Permit", signifiant que, pour autant que cette règle soit concernée, l'accès demandé devrait être permis. Si une règle évalue comme "False", elle retournera alors un résultat de "NotApplicable". Si une erreur survient lors de l'évaluation de la règle, la règle retournera un résultat de "Indeterminate". Comme mentionné ci-dessus, l'algorithme de combinaison de règles pour la politique spécifie comment les diverses valeurs de règles sont combinées en une seule valeur de politique.

[a16] – [a19] fournit une description textuelle de cette règle. Cette description est facultative.

[a20] introduit la cible de la règle. Comme décrit ci-dessus pour la cible d'une politique, la cible d'une règle décrit les demandes de décision auxquelles cette règle s'applique. Si le sujet, ressource, action et environnement dans une demande de décision ne correspond pas aux valeurs spécifiées dans la cible de la règle, le reste de la règle n'a alors pas besoin d'être évalué, et une valeur de "NotApplicable" est retournée à l'évaluation de la règle.

La cible de règle est semblable à la cible de la politique elle-même, mais avec une importante différence. [a23]- [a32] énonce une valeur spécifique que le sujet doit satisfaire dans la demande de décision. L'élément `<SubjectMatch>` spécifie une fonction de correspondance dans l'attribut `MatchId`, une valeur littérale de "med.example.com" et un pointeur sur un attribut spécifique du sujet dans le contexte de la demande au moyen de l'élément `<SubjectAttributeDesignator>`. La fonction de correspondance sera utilisée pour comparer la valeur littérale avec la valeur de l'attribut de sujet . La règle ne sera applicable à une demande de décision particulière que si la confrontation retourne "True". Si la confrontation retourne "False", cette règle retournera alors une valeur de "NotApplicable".

[a36] ferme la règle. Dans cette règle, tout le travail est fait dans l'élément `<Target>`. Dans des règles plus complexes, `<Target>` peut être suivi par un élément `<Condition>` (qui pourrait aussi être un ensemble de conditions qui seraient à la fois ET et OU).

[a37] ferme la politique. Comme mentionné ci-dessus, cette politique a seulement une règle, mais des politiques plus complexes peuvent avoir un nombre quelconque de règles.

II.1.2 Exemple de contexte de demande

Examinons une demande de décision hypothétique qui pourrait être soumise à un PDP qui exécute la politique ci-dessus. En français, la demande d'accès qui génère la demande de décision peut se déclarer comme suit:

Bart Simpson, avec le nom de messagerie "bs@simpsons.com", veut lire son dossier médical chez Medi Corp.

En XACML, les informations de la demande de décision sont formatées en une déclaration de contexte de demande qui ressemble à ce qui suit:

```
[a38]  <?xml version="1.0" encoding="UTF-8"?>
[a39]  <Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
[a40]  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
http://docs.oasis-
open.org/xacml/access_control-xacml-2.0-context-schema-os.xsd">
[a41]   <Subject>
[a42]    <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name">
[a43]     <AttributeValue>
[a44]      bs@simpsons.com
[a45]     </AttributeValue>
[a46]    </Attribute>
[a47]   </Subject>
[a48]   <Resource>
[a49]    <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#anyURI">
[a50]     <AttributeValue>
[a51]      file://example/med/record/patient/BartSimpson
[a52]     </AttributeValue>
[a53]    </Attribute>
[a54]   </Resource>
[a55]   <Action>
[a56]    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-
id"
DataType="http://www.w3.org/2001/XMLSchema#string">
[a57]     <AttributeValue>
[a58]      read
[a59]     </AttributeValue>
[a60]    </Attribute>
[a61]   </Action>
[a62]   <Environment/>
[a63]  </Request>
```

[a38] – [a40] contient les informations d'en-tête pour le contexte de la demande, qui sont utilisées de la même façon que l'en-tête pour la politique décrite ci-dessus.

L'élément `<Subject>` contient un ou plusieurs attributs de l'entité qui fait la demande d'accès. Il peut y avoir plusieurs sujets, et chaque sujet peut avoir plusieurs attributs. Dans ce cas, en [a41] – [a47], il n'y a qu'un seul sujet, et le sujet n'a qu'un seul attribut: l'identité du sujet, exprimée comme un nom de messagerie électronique, est "bs@simpsons.com". Dans cet exemple, `subject-category` l'attribut de catégorie de sujet est omis. Donc, il adopte sa valeur par défaut de "access-subject".

L'élément `<Resource>` contient un ou plusieurs attributs de la ressource à laquelle le sujet ou les sujets ont demandé l'accès. Il ne peut y avoir qu'une `<Resource>` par demande de décision . Les lignes [a48] – [a54] contiennent le seul attribut de la ressource à laquelle Bart Simpson a demandé l'accès: la ressource identifiée par son fichier d'URI, qui est "**file**:`//medico/record/patient/BartSimpson`".

L'élément `<Action>` contient un ou plusieurs attributs de l'action que le sujet ou les sujets souhaitent effectuer sur la ressource. Il ne peut y avoir qu'une action par demande de décision. [a55] – [a61] décrit l'identité de l'action que Bart Simpson souhaite entreprendre, qui est "read" *(lire)*.

L'élément `<Environment>`, [a62], est vide.

[a63] ferme le contexte de la demande. Un contexte de demande plus complexe pourrait contenir des attributs non associés avec le sujet, la ressource ou l'action. Ils auraient été placés dans un élément `<Environment>` facultatif à la suite de l'élément `<Action>`.

Le PDP qui traite ce contexte de demande loge la politique dans son réceptacle de politique. Il compare le sujet, la ressource, l'action et l'environnement dans le contexte de la demande avec les sujets, ressources, actions et environnements dans la cible de la politique. Comme la cible de la politique est vide, la politique satisfait à ce contexte.

Le PDP compare ensuite les sujet, ressource, action et environnement dans le contexte de la demande avec la cible de la seule règle dans cette politique. La ressource demandée satisfait l'élément `<Target>` et l'action demandée satisfait

l'élément <Target>, mais l'attribut de "subject-id" (l'identifiant de sujet) demandeur ne satisfait pas à "med.example.com".

II.1.3 Exemple de contexte de réponse

Comme résultat de l'évaluation de la politique, il n'y a pas dans cette politique de règle qui retourne un résultat "Permit" pour cette demande. L'algorithme de combinaison de règle pour la politique spécifie que dans ce cas un résultat de "NotApplicable" devrait être retourné. Le contexte de réponse ressemble à ce qui suit:

```
[a64]  <?xml version="1.0" encoding="UTF-8"?>
[a65]  <Response xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
http://docs.oasis-open.org/xacml/xacml-core-2.0-context-schema-os.xsd">
[a66]   <Result>
[a67]    <Decision>NotApplicable</Decision>
[a68]   </Result>
[a69]  </Response>
```

[a64] – [a65] contient la même sorte d'informations d'en-tête pour la réponse que ce qui a été décrit ci-dessus pour une politique.

L'élément <Result> dans les lignes [a66] – [a68] contient le résultat de l'évaluation de demande de décision par rapport à la politique. Dans ce cas, le résultat est "NotApplicable". Une politique peut retourner "Permit", "Deny", "NotApplicable" ou "Indeterminate". Donc, le PEP est prié de refuser l'accès.

[a69] ferme le contexte de réponse.

II.2 Exemple deux

Le présent paragraphe contient un exemple de document XML, un exemple de contexte de demande et un exemple de règles XACML. Le document XML est un dossier médical. Quatre règles distinctes sont définies. Cela illustre un algorithme de combinaison de règles, les conditions et obligations.

II.2.1 Exemple d'instance de dossier médical

Ce qui suit est une instance d'un dossier médical auquel peut s'appliquer l'exemple des règles XACML. Le schéma <record> est défini dans l'espace de nom enregistré administré par Medi Corp.

```
[a70]  <?xml version="1.0" encoding="UTF-8"?>
[a71]   <record xmlns="urn:example:med:schemas:record"
[a72]   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
[a73]    <patient>
[a74]     <patientName>
[a75]      <first>Bartholomew</first>
[a76]      <last>Simpson</last>
[a77]     </patientName>
[a78]     <patientContact>
[a79]      <street>27 Shelbyville Road</street>
[a80]      <city>Springfield</city>
[a81]      <state>MA</state>
[a82]      <zip>12345</zip>
[a83]      <phone>555.123.4567</phone>
[a84]      <fax/>
[a85]      <email/>
[a86]     </patientContact>
[a87]     <patientDoB>1992-03-21</patientDoB>
[a88]     <patientGender>male</patientGender>
[a89]     <patient-number>555555</patient-number>
[a90]    </patient>
[a91]    <parentGuardian>
[a92]     <parentGuardianId>HS001</parentGuardianId>
[a93]     <parentGuardianName>
[a94]      <first>Homer</first>
[a95]      <last>Simpson</last>
[a96]     </parentGuardianName>
[a97]     <parentGuardianContact>
[a98]      <street>27 Shelbyville Road</street>
[a99]      <city>Springfield</city>
[a100]     <state>MA</state>
```

```
[a101]     <zip>12345</zip>
[a102]     <phone>555.123.4567</phone>
[a103]     <fax/>
[a104]     <email>homers@aol.com</email>
[a105]    </parentGuardianContact>
[a106]   </parentGuardian>
[a107]   <primaryCarePhysician>
[a108]    <physicianName>
[a109]     <first>Julius</first>
[a110]     <last>Hibbert</last>
[a111]    </physicianName>
[a112]    <physicianContact>
[a113]     <street>1 First St</street>
[a114]     <city>Springfield</city>
[a115]     <state>MA</state>
[a116]     <zip>12345</zip>
[a117]     <phone>555.123.9012</phone>
[a118]     <fax>555.123.9013</fax>
[a119]     <email/>
[a120]    </physicianContact>
[a121]     <registrationID>ABC123</registrationID>
[a122]    </primaryCarePhysician>
[a123]    <insurer>
[a124]     <name>Blue Cross</name>
[a125]     <street>1234 Main St</street>
[a126]     <city>Springfield</city>
[a127]     <state>MA</state>
[a128]     <zip>12345</zip>
[a129]     <phone>555.123.5678</phone>
[a130]     <fax>555.123.5679</fax>
[a131]     <email/>
[a132]    </insurer>
[a133]   <medical>
[a134]    <treatment>
[a135]     <drug>
[a136]      <name>methylphenidate hydrochloride</name>
[a137]      <dailyDosage>30mgs</dailyDosage>
[a138]      <startDate>1999-01-12</startDate>
[a139]     </drug>
[a140]     <comment>
[a141]      patient exhibits side-effects of skin coloration and carpal
degeneration
[a142]     </comment>
[a143]    </treatment>
[a144]    <result>
[a145]     <test>blood pressure</test>
[a146]     <value>120/80</value>
[a147]     <date>2001-06-09</date>
[a148]     <performedBy>Nurse Betty</performedBy>
[a149]    </result>
[a150]   </medical>
[a151]  </record>
```

II.2.2 Exemple de contexte de demande

L'exemple suivant illustre un contexte de demande auquel l'exemple de règles peut être applicable. Il représente une demande du médecin Julius Hibbert de lire la date de naissance du patient dans le dossier de Bartholomew Simpson.

```
[a152] <?xml version="1.0" encoding="UTF-8"?>
[a153] <Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=" urn:oasis:names:tc:xacml:2.0:context:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-context-schema-
os.xsd">
[a154]  <Subject>
[a155]   <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject-
category"
DataType="http://www.w3.org/2001/XMLSchema#anyURI">
[a156]    <AttributeValue>urn:oasis:names:tc:xacml:1.0:subject-
category:access-subject</AttributeValue>
[a157]   </Attribute>
```

```
[a158]   <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
 DataType="http://www.w3.org/2001/XMLSchema#string"
Issuer="med.example.com">
[a159]    <AttributeValue>CN=Julius Hibbert</AttributeValue>
[a160]   </Attribute>
[a161]   <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:name-
format"
DataType="http://www.w3.org/2001/XMLSchema#anyURI"
Issuer="med.example.com">
[a162]    <AttributeValue>
[a163]     urn:oasis:names:tc:xacml:1.0:datatype:x500name
[a164]    </AttributeValue>
[a165]   </Attribute>
[a166]   <Attribute
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:role"
DataType="http://www.w3.org/2001/XMLSchema#string"
Issuer="med.example.com">
[a167]    <AttributeValue>physician</AttributeValue>
[a168]   </Attribute>
[a169]   <Attribute
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:physician-id"
DataType="http://www.w3.org/2001/XMLSchema#string"
Issuer="med.example.com">
[a170]    <AttributeValue>jh1234</AttributeValue>
[a171]   </Attribute>
[a172]  </Subject>
[a173]  <Resource>
[a174]   <ResourceContent>
[a175]    <md:record xmlns:md="urn:example:med:schemas:record"
xsi:schemaLocation="urn:example:med:schemas:record
http:www.med.example.com/schemas/record.xsd">
[a176]     <md:patient>
[a177]      <md:patientDoB>1992-03-21</md:patientDoB>
[a178]      <md:patient-number>555555</md:patient-number>
[a179]     </md:patient>
[a180]    </md:record>
[a181]   </ResourceContent>
[a182]   <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string">
[a183]    <AttributeValue>
[a184]     //med.example.com/records/bart-simpson.xml#
[a185] xmlns(md=:Resource/ResourceContent/xpointer
[a186] (/md:record/md:patient/md:patientDoB)
[a187]    </AttributeValue>
[a188]   </Attribute>
[a189] </Resource>
[a190]  <Action>
[a191]   <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-
id"
DataType="http://www.w3.org/2001/XMLSchema#string">
[a192]    <AttributeValue>read</AttributeValue>
[a193]   </Attribute>
[a194]  </Action>
[a195]  <Environment/>
[a196] </Request>
```

[a152] – [a153] Déclarations d'espace de nom standard.

[a154] – [a172] les attributs de sujet sont placés dans l'élément `<Subject>` de l'élément `<Request>`. Chaque attribut consiste en métadonnées d'attribut et une valeur d'attribut. Un seul sujet est impliqué dans cette demande.

[a155] – [a157] Chaque élément `<Subject>` a un attribut `SubjectCategory`. La valeur de cet attribut décrit le rôle que le sujet en cause joue en faisant la demande de décision. La valeur de "`access-subject`" note l'identité pour laquelle la demande a été produite.

[a158] – [a160] Attribut `subject-id` du sujet.

[a161] – [a165] Format du `subject-id`.

[a166] – [a168] Attribut de `role` du sujet.

[a169] – [a171] Attribut `physician-id` du sujet.

[a173] – [a189] Les attributs de ressource sont placés dans l'élément `<Resource>` de l'élément `<Request>`. Chaque attribut consiste en métadonnées d'attribut et une valeur d'attribut.

[a174] – [a181] Contenu de la ressource. L'instance de ressource XML, à tout ou partie de laquelle on peut demander l'accès, est placée ici.

[a182] – [a188] L'identifiant de l'instance de la ressource à laquelle l'accès est demandé, qui est une expression XPath dans l'élément `<ResourceContent>` qui choisit les données auxquelles accéder.

[a190] – [a194] Les attributs d'action sont placés dans l'élément `<Action>` de l'élément `<Request>`.

[a192] Identifiant d'action.

[a195] L'élément `<Environment>` vide.

II.2.3 Exemple de règles de langage clair

Les règles de langage clair suivantes sont à mettre en application:

1) règle 1: une personne, identifiée par son numéro de patient, peut lire tout dossier pour lequel il est le patient désigné;
2) règle 2: une personne peut lire tout dossier pour lequel il est le parent ou gardien désigné, et pour lequel le patient est âgé de moins de 16 ans;
3) règle 3: un médecin peut écrire sur tout élément médical pour lequel il est le médecin de soin principal désigné, pourvu qu'un message électronique soit envoyé au patient;
4) règle 4: un administrateur n'a pas la permission de lire ou écrire sur les éléments médicaux d'un dossier de patient.

Ces règles peuvent être écrites par différents PAP fonctionnant de façon indépendante, ou par un seule PAP.

II.2.4 Exemple d'instances de règle XACML

II.2.4.1 Règle 1

La règle 1 illustre une règle simple avec un seul élément `<Condition>`. Elle illustre aussi l'utilisation de l'élément `<VariableDefinition>` pour définir une fonction qui peut être utilisée tout au long de la politique.

L'instance de `<Rule>` XACML suivante exprime la règle 1:

```
[a197] <?xml version="1.0" encoding="UTF-8"?>
[a198] <Policy
[a199] xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
[a200] xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=" urn:oasis:names:tc:xacml:2.0:policy:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-context-schema-
os.xsd"
[a201] xmlns:md="http://www.med.example.com/schemas/record.xsd"
[a202] PolicyId="urn:oasis:names:tc:xacml:2.0:example:policyid:1"
[a203] RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
algorithm:deny-overrides">
[a204]  <PolicyDefaults>
[a205]        <XPathVersion>http://www.w3.org/TR/1999/Rec-xpath-
19991116</XPathVersion>
[a206]  </PolicyDefaults>
[a207]  <Target/>
[a208]  <VariableDefinition VariableId="17590034">
[a209]   <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
equal">
[a210]    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
one-and-only">
[a211]     <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:patient-number"
[a212]     DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a213]    </Apply>
[a214]    <Apply
```

```
[a215]    FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-
only">
[a216]     <AttributeSelector
[a217]     RequestContextPath="//xacml-context:Resource/xacml-
context:ResourceContent/md:record/md:patient/md:patient-number/text()"
[a218]     DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a219]    </Apply>
[a220]   </Apply>
[a221]  </VariableDefinition>
[a222]  <Rule
[a223]  RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:1"
[a224]  Effect="Permit">
[a225]   <Description>
[a226]     A person may read any medical record in the
[a227]     http://www.med.example.com/schemas/record.xsd namespace
[a228]     for which he or she is the designated patient
[a229]   </Description>
[a230]   <Target>
[a231]    <Resources>
[a232]     <Resource>
[a233]      <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a234]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a235]         urn:example:med:schemas:record
[a236]       </AttributeValue>
[a237]       <ResourceAttributeDesignator AttributeId=
[a238]       "urn:oasis:names:tc:xacml:2.0:resource:target-namespace"
[a239]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a240]      </ResourceMatch>
[a241]      <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:xpath-node-match">
[a242]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a243]         /md:record
[a244]       </AttributeValue>
[a245]       <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:xpath"
[a246]        DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a247]      </ResourceMatch>
[a248]     </Resource>
[a249]    </Resources>
[a250]    <Actions>
[a251]     <Action>
[a252]      <ActionMatch
[a253]       MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a254]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a255]          read
[a256]       </AttributeValue>
[a257]       <ActionAttributeDesignator
[a258]        AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
[a259]        DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a260]      </ActionMatch>
[a261]     </Action>
[a262]    </Actions>
[a263]   </Target>
[a264]   <Condition>
[a265]    <VariableReference VariableId="17590034"/>
[a266]   </Condition>
[a267]  </Rule>
[a268] </Policy>
```

[a199] – [a201] Déclarations d'espace XML.

[a205] Les expressions XPath dans la politique sont à interpréter conformément à XPath:1999 du W3C.

[a208] – [a221] Elément `<VariableDefinition>`. Il définit une fonction qui évalue la véracité de la déclaration: l'attribut numéro de patient du sujet est égal au `patient-number` (numéro de patient) dans la ressource.

[a209] L'attribut `FunctionId` nomme la fonction à utiliser pour la comparaison. Dans ce cas, la comparaison est faite avec la fonction "`urn:oasis:names:tc:xacml:1.0:function:string-equal`"; cette fonction prend deux arguments de type "http://www.w3.org/2001/XMLSchema#string".

[a210] Le premier argument de définition de variable est une fonction spécifiée par l'attribut `FunctionId`. Comme `urn:oasis:names:tc:xacml:1.0:function:string-equal` prend des arguments du type "http://www.w3.org/2001/XMLSchema#string" et que `SubjectAttributeDesignator` choisit un sac de type "http://www.w3.org/2001/XMLSchema#string", "`urn:oasis:names:tc:xacml:1.0:function:string-one-and-only`" est utilisé. Cette fonction garantit que son argument s'évalue comme un sac contenant exactement une valeur.

[a211] Le `SubjectAttributeDesignator` choisit un sac de valeurs pour l'attribut `patient-number` du sujet dans le contexte de la demande.

[a215] Le second argument de la définition de variable est une fonction spécifiée par l'attribut `FunctionId`. Comme "`urn:oasis:names:tc:xacml:1.0:function:string-equal`" prend des arguments du type "http://www.w3.org/2001/XMLSchema#string" et que `AttributeSelector` choisit un sac de type "http://www.w3.org/2001/XMLSchema#string", "`urn:oasis:names:tc:xacml:1.0:function:string-one-et-only`" est utilisé. Cette fonction garantit que son argument s'évalue comme un sac contenant exactement une valeur.

[a216] L'élément `<AttributeSelector>` choisit un sac de valeurs provenant du contexte de la demande en utilisant une expression XPath de forme libre. Dans ce cas, il choisit la valeur du `patient-number` dans la ressource. Noter que les préfixes d'espace de nom dans l'expression XPath sont résolus avec les déclarations d'espace de nom standard XML.

[a223] Identifiant de règle.

[a224] Déclaration d'effet de règle. Lorsqu'une règle évalue à 'True', elle émet la valeur de l'attribut d'effet. Cette valeur est alors combinée avec les valeurs d'effet des autres règles conformément à l'algorithme de combinaison de règles.

[a225] – [a229] Description de forme libre de la règle.

[a230] – [a263] Une cible de règle définit un ensemble de demandes de décision que la règle est destinée à évaluer. Dans cet exemple, les éléments `<Subjects>` et `<Environments>` sont omis.

[a231] – [a249] L'élément `<Resources>` contient une séquence disjonctive d'éléments `<Resource>`. Dans cet exemple, il n'y en a qu'un.

[a232] – [a248] L'élément `<Resource>` englobe la séquence conjonctive d'éléments `ResourceMatch`. Dans cet exemple, Il y en a deux.

[a233] – [a240] Le premier élément `<ResourceMatch>` compare ses premier et second éléments fils conformément à la fonction de correspondance. Une correspondance est positive si la valeur du premier argument satisfait une des valeurs choisies par le second argument. Cette confrontation compare l'espace de nom cible du document demandé avec la valeur de "`urn:example:med:schemas:record`".

[a233] L'attribut `MatchId` nomme la fonction de correspondance.

[a235] Valeur littérale d'attribut à satisfaire.

[a237] – [a239] L'élément `<ResourceAttributeDesignator>` choisit l'espace de nom cible provenant de la ressource contenue dans le contexte de la demande. Le nom de l'attribut est spécifié par le `AttributeId`.

[a241] – [a247] Le second élément `<ResourceMatch>`. Cette confrontation compare le résultat de deux expressions XPath. La seconde expression XPath est le chemin de localisation pour l'élément XML demandé et la première expression XPath est la valeur littérale "`/md:record`". La fonction "xpath-node-match" évalue à "True" si l'élément XML demandé est inférieur à l'élément "`/md:record`".

[a250] – [a262] L'élément `<Actions>` contient une séquence disjonctive d'éléments `<Action>`. Dans ce cas, il y a un seul élément `<Action>`.

[a251] – [a261] L'élément `<Action>` contient une séquence conjonctive d'éléments `<ActionMatch>`. Dans ce cas, il y a un seul élément `<ActionMatch>`.

[a252] – [a260] L'élément `<ActionMatch>` compare ses premier et second éléments fils conformément à la fonction de correspondance. La confrontation est positive si la valeur du premier argument satisfait une des valeurs choisies par le second argument. Dans ce cas, la valeur de l'attribut d'action `action-id` dans le contexte de la demande est comparée à la valeur littérale "`read`".

[a264] – [a266] L'élément `<Condition>`. Une condition doit évaluer à "True" pour que la règle soit applicable. Cette condition contient une référence à une définition de variable qui est définie ailleurs dans la politique.

II.2.4.2 Règle 2

La règle 2 illustre l'utilisation d'une fonction mathématique, c'est-à-dire, l'élément `<Apply>` avec un `FunctionId` de "`urn:oasis:names:tc:xacml:1.0:function:date-add-yearMonthDuration`" pour calculer la date du seizième anniversaire du patient. Elle illustre aussi l'utilisation des expressions de prédicat, avec le `FunctionId` de "`urn:oasis:names:tc:xacml:1.0:function:and`". Cet exemple a une fonction enchâssée dans l'élément `<Condition>` et une autre référencée dans un élément `<VariableDefinition>`.

```
[a269] <?xml version="1.0" encoding="UTF-8"?>
[a270] <Policy
[a271] xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
[a272] xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-policy-schema-os.xsd"
[a273] xmlns:xf="urn:oasis:names:tc:xacml:2.0:data-types"
[a274] xmlns:md="http:www.med.example.com/schemas/record.xsd"
[a275] PolicyId="urn:oasis:names:tc:xacml:2.0:example:policyid:2"
RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-
overrides">
[a276]  <PolicyDefaults>
[a277]   <XPathVersion>http://www.w3.org/TR/1999/Rec-xpath-19991116</XPathVersion>
[a278]  </PolicyDefaults>
[a279]  <Target/>
[a280]  <VariableDefinition VariableId="17590035">
[a281]   <Apply FunctionId="urn:oasis:names:tc:xacml:2.0:function:date-less-or-
equal">
[a282]    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:date-one-and-
only">
[a283]     <EnvironmentAttributeDesignator
[a284]     AttributeId= "urn:oasis:names:tc:xacml:1.0:environment:current-date"
[a285]     DataType="http://www.w3.org/2001/XMLSchema#date"/>
[a286]    </Apply>
[a287]    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:date-add-
yearMonthDuration">
[a288]     <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:date-one-and-
only">
[a289]      <AttributeSelector RequestContextPath=
[a290]      "//md:record/md:patient/md:patientDoB/text()"
[a291]      DataType="http://www.w3.org/2001/XMLSchema#date"/>
[a292]     </Apply>
[a293]     <AttributeValue
[a294]     DataType="urn:oasis:names:tc:xacml:2.0:data-types:yearMonthDuration">
[a295]      <xf:dt-yearMonthDuration>
[a296]       P16Y
[a297]      </xf:dt-yearMonthDuration>
[a298]     </AttributeValue>
[a299]    </Apply>
[a300]   </Apply>
[a301]  </VariableDefinition>
[a302]  <Rule
[a303]  RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:2"
[a304]  Effect="Permit">
[a305]   <Description>
[a306]    A person may read any medical record in the
[a307]    http://www.med.example.com/records.xsd namespace
[a308]    for which he or she is the designated parent or guardian,
[a309]    and for which the patient is under 16 years of age
[a310]   </Description>
[a311]   <Target>
[a312]    <Resources>
[a313]     <Resource>
[a314]      <ResourceMatch
[a315]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a316]       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
[a317]          http://www.med.example.com/schemas/record.xsd
```

```
[a318]        </AttributeValue>
[a319]       <ResourceAttributeDesignator AttributeId=
"urn:oasis:names:tc:xacml:2.0:resource:target-namespace"
[a320]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a321]      </ResourceMatch>
[a322]      <ResourceMatch
[a323]      MatchId="urn:oasis:names:tc:xacml:1.0:function:xpath-node-match">
[a324]       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
[a325]         /md:record
[a326]       </AttributeValue>
[a327]       <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:xpath"
[a328]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a329]      </ResourceMatch>
[a330]     </Resource>
[a331]    </Resources>
[a332]    <Actions>
[a333]     <Action>
[a334]      <ActionMatch
[a335]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a336]       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
[a337]         read
[a338]       </AttributeValue>
[a339]       <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
[a340]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a341]     </ActionMatch>
[a342]     </Action>
[a343]    </Actions>
[a344]   </Target>
[a345]   <Condition>
[a346]   <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
[a347]    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a348]     <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-
and-only">
[a349]      <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:
[a350] parent-guardian-id"
[a351]      DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a352]     </Apply>
[a353]     <Apply
[a354]     FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
[a355]      <AttributeSelector
[a356]      RequestContextPath="//xacml-context:Resource/xacml-
context:ResourceContent/md:record/md:parentGuardian/md:parentGuardianId/text()"
[a357]      DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a358]      </Apply>
[a359]     </Apply>
[a360]     <VariableReference VariableId="17590035"/>
[a361]    </Apply>
[a362]   </Condition>
[a363]  </Rule>
[a364] </Policy>
```

[a280] – [a301] L'élément `<VariableDefinition>` contient une partie de la condition (c'est-à-dire, le patient est-il un mineur de 16 ans?). Le patient est un mineur de 16 ans si la date actuelle est inférieure à la date calculée en ajoutant 16 à l'année de naissance du patient.

[a281] – [a300] "`urn:oasis:names:tc:xacml:1.0:function:date-less-or-equal`" est utilisé pour calculer la différence de deux arguments de date.

[a282] – [a286] Le premier argument de date utilise "`urn:oasis:names:tc:xacml:1.0:function:date-one-and-only`" pour s'assurer que le sac de valeurs choisi par son argument contient exactement une valeur de type "http://www.w3.org/2001/XMLSchema#date".

[a284] La date actuelle est évaluée en choisissant l'attribut d'environnement "`urn:oasis:names:tc:xacml:1.0:environment:current-date`".

[a287] – [a299] Le second argument de date utilise "`urn:oasis:names:tc:xacml:1.0:function:date-add-yearMonthDuration`" pour calculer la date du seizième anniversaire du patient en ajoutant 16 ans à l'année de

naissance du patient. Le premier de ses arguments est du type "http://www.w3.org/2001/XMLSchema#date" et le second est du type "urn:oasis:names:tc:xacml:2.0:datatypes:yearMonthDuration".

[a289] L'élément <AttributeSelector> choisit la date de naissance du patient en prenant l'expression XPath sur le contenu de la ressource.

[a293] – [a298] Durée en an et mois de 16 ans.

[a311] – [a344] Déclaration de règle et cible de règle. Voir la règle 1 au § II.4.2.4.1 pour les explications détaillées de ces éléments.

[a345] – [a362] L'élément <Condition>. La condition doit s'évaluer comme "True" pour que la règle soit applicable. Cette condition évalue la véracité de la déclaration: le demandeur est le parent ou gardien désigné et le patient est un mineur de 16 ans. Elle contient un élément <Apply> enchâssé et un élément <VariableDefinition> référencé.

[a346] La condition utilise la fonction "urn:oasis:names:tc:xacml:1.0:function:and". C'est une fonction booléenne qui prend un ou plusieurs arguments booléens (2 dans ce cas) et effectue l'opération logique "ET" pour calculer la vraie valeur de l'expression.

[a347] – [a359] La première partie de la condition est évaluée (c'est-à-dire, le demandeur est-il le parent ou gardien désigné?). La fonction est "urn:oasis:names:tc:xacml:1.0:function:string-equal" et elle prend deux arguments de type "http://www.w3.org/2001/XMLSchema#string".

[a348] désigne le premier argument. Comme "urn:oasis:names:tc:xacml:1.0:function:string-equal" prend des arguments de type "http://www.w3.org/2001/XMLSchema#string", "urn:oasis:names:tc:xacml:1.0:function:string-one-and-only" est utilisé pour s'assurer que l'attribut de sujet "urn:oasis:names:tc:xacml:2.0:example:attribute:parent-guardian-id" dans le contexte de la demande contient exactement une valeur.

[a353] désigne le second argument. La valeur de l'attribut de sujet "urn:oasis:names:tc:xacml:2.0:example:attribute:parent-guardian-id" est choisie d'après le contexte de la demande en utilisant l'élément <SubjectAttributeDesignator>.

[a354] Comme ci-dessus, "urn:oasis:names:tc:xacml:1.0:function:string-one-and-only" est utilisé pour s'assurer que le sac de valeurs choisi par son argument contient exactement une valeur de type "http://www.w3.org/2001/XMLSchema#string".

[a355] Le second argument choisit la valeur de l'élément <md:parentGuardianId> d'après le contenu de la ressource en utilisant l'élément <AttributeSelector>. Cet élément contient une expression XPath de forme libre, qui pointe dans le contexte de la demande. Noter que tous les préfixes d'espace de nom dans l'expression XPath sont résolus avec des déclarations d'espace de nom standard. Le AttributeSelector évalue le sac de valeurs de type "http://www.w3.org/2001/XMLSchema#string".

[a360] fait référence à l'élément <VariableDefinition>, où la seconde partie de la condition est définie.

II.2.4.3 Règle 3

La règle 3 illustre l'utilisation d'une obligation. La syntaxe de l'élément <Rule> de XACML n'inclut pas d'élément convenable pour porter une obligation, et donc la règle 3 doit être formatée comme un élément <Policy>.

```
[a365] <?xml version="1.0" encoding="UTF-8"?>
[a366] <Policy
[a367]  xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
[a368]  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
[a369]  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-policy-schema-
os.xsd"
[a370] xmlns:md="http:www.med.example.com/schemas/record.xsd"
[a371]  PolicyId="urn:oasis:names:tc:xacml:2.0:example:policyid:3"
[a372]  RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
algorithm:deny-overrides">
[a373]  <Description>
[a374]   Policy for any medical record in the
[a375]   http://www.med.example.com/schemas/record.xsd namespace
[a376]  </Description>
[a377]  <PolicyDefaults>
[a378]   <XPathVersion>http://www.w3.org/TR/1999/Rec-xpath-
19991116</XPathVersion>
```

```
[a379]  </PolicyDefaults>
[a380]  <Target>
[a381]   <Resources>
[a382]    <Resource>
[a383]     <ResourceMatch
[a384]     MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a385]      <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a386]       urn:example:med:schemas:record
[a387]      </AttributeValue>
[a388]      <ResourceAttributeDesignator AttributeId=
[a389]      "urn:oasis:names:tc:xacml:1.0:resource:target-namespace"
[a390]      DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a391]     </ResourceMatch>
[a392]    </Resource>
[a393]   </Resources>
[a394]  </Target>
[a395]  <Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:3"
[a396]  Effect="Permit">
[a397]   <Description>
[a398]    A physician may write any medical element in a record
[a399]    for which he or she is the designated primary care
[a400]    physician, provided an email is sent to the patient
[a401]   </Description>
[a402]   <Target>
[a403]    <Subjects>
[a404]     <Subject>
[a405]      <SubjectMatch
[a406]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a407]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a408]         physician
[a409]        </AttributeValue>
[a410]        <SubjectAttributeDesignator AttributeId=
"urn:oasis:names:tc:xacml:2.0:example:attribute:role"
[a411]        DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a412]      </SubjectMatch>
[a413]     </Subject>
[a414]    </Subjects>
[a415]    <Resources>
[a416]     <Resource>
[a417]      <ResourceMatch
[a418]      MatchId="urn:oasis:names:tc:xacml:1.0:function:xpath-node-
match">
[a419]       <AttributeValue
[a420]        DataType="http://www.w3.org/2001/XMLSchema#string">
[a421]          /md:record/md:medical
[a422]        </AttributeValue>
[a423]       <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:xpath"
[a424]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a425]      </ResourceMatch>
[a426]     </Resource>
[a427]    </Resources>
[a428]    <Actions>
[a429]     <Action>
[a430]      <ActionMatch
[a431]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a432]       <AttributeValue
[a433]       DataType="http://www.w3.org/2001/XMLSchema#string">
[a434]         write
[a435]       </AttributeValue>
[a436]       <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
[a437]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a438]      </ActionMatch>
[a439]     </Action>
[a440]    </Actions>
[a441]   </Target>
[a442]   <Condition>
```

```
[a443]   <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
equal">
[a444]    <Apply
[a445]    FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-
only">
[a446]     <SubjectAttributeDesignator
[a447]     AttributeId="urn:oasis:names:tc:xacml:2.0:example:
attribute:physician-id"
[a448]     DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a449]    </Apply>
[a450]    <Apply
[a451]    FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-
only">
[a452]     <AttributeSelector RequestContextPath=
[a453]     "//xacml-context:Resource/xacml-
context:ResourceContent/md:record/md:primaryCarePhysician/md:registrationID
/text()"
[a454]     DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a455]    </Apply>
[a456]     </Apply>
[a457]   </Condition>
[a458]  </Rule>
[a459]  <Obligations>
[a460]   <Obligation
ObligationId="urn:oasis:names:tc:xacml:example:obligation:email"
[a461]   FulfillOn="Permit">
[a462]    <AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:mailto"
[a463]    DataType="http://www.w3.org/2001/XMLSchema#string">
[a464]     &lt;AttributeSelector RequestContextPath=
[a465]     "//md:/record/md:patient/md:patientContact/md:email"
[a466]     DataType="http://www.w3.org/2001/XMLSchema#string"/&gt ;
[a467]    </AttributeAssignment>
[a468]    <AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
[a469]    DataType="http://www.w3.org/2001/XMLSchema#string">
[a470]      Your medical record has been accessed by:
[a471]    </AttributeAssignment>
[a472]    <AttributeAssignment
AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
[a473]    DataType="http://www.w3.org/2001/XMLSchema#string">
[a474]     &lt;SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
[a475]     DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
[a476]    </AttributeAssignment>
[a477]   </Obligation>
[a478]  </Obligations>
[a479] </Policy>
```

[a366] – [a372] L'élément `<Policy>` inclut des déclarations d'espace de nom standard ainsi que des paramètres spécifiques de politique, tels que `PolicyId` et `RuleCombiningAlgId`.

[a371] Identifiant de politique. Ce paramètre permet que la politique soit référencée par un ensemble de politiques.

[a372] L'algorithme de combinaison de règles identifie l'algorithme pour combiner les résultats de l'évaluation de règle.

[a373] – [a376] Description de forme libre de la politique.

[a379] – [a394] Cible de la politique. La cible de la politique définit un ensemble de demandes de décision applicables. La structure de l'élément `<Target>` dans `<Policy>` est identique à la structure de l'élément `<Target>` dans `<Rule>`. Dans ce cas, la cible de la politique est l'ensemble de toutes les ressources XML conformes à l'espace de nom "`urn:example:med:schemas:record`".

[a395] Seul élément `<Rule>` inclus dans cette `<Policy>`. Deux paramètres sont spécifiés dans l'en-tête de règle: `RuleId` et `Effect`.

[a402] – [a441] La cible de règle restreint encore la cible de la politique.

[a405] – [a412] L'élément `<SubjectMatch>` fait viser la règle sur les sujets dont l'attribut de sujet "`urn:oasis:names:tc:xacml:2.0:example:attribute:role`" est égal à "`physician`".

[a417] – [a425] L'élément `<ResourceMatch>` fait viser la règle sur les ressources qui satisfont l'expression XPath "`/md:record/md:medical`".

[a430] – [a438] L'élément `<ActionMatch>` fait viser la règle sur les actions dont l'attribut d'action "`urn:oasis:names:tc:xacml:1.0:action:action-id`" est égal à "write".

[a442] – [a457] L'élément `<Condition>`. Pour que la règle soit applicable à la demande de décision, la condition doit évaluer à "True". Cette condition compare la valeur de l'attribut de sujet "`urn:oasis:names:tc:xacml:2.0:example:attribute:physician-id`" avec la valeur de l'élément `<registrationId>` dans le dossier médical auquel on accède.

[a459] – [a478] L'élément `<Obligations>`. Les obligations sont un ensemble d'opérations qui doivent être effectuées par le PEP en conjonction avec une décision d'autorisation. Une obligation peut être associée à une décision d'autorisation "Permit" ou "Deny". L'élément contient une seule obligation.

[a460] – [a477] L'élément `<Obligation>` consiste en l'attribut `ObligationId`, la valeur de la décision d'autorisation pour laquelle elle doit être satisfaite, et un ensemble d'allocations d'attributs. Le PDP ne résout pas les allocations d'attribut. C'est la tâche du PEP.

[a460] L'attribut `ObligationId` identifie l'obligation. Dans ce cas, le PEP est obligé d'envoyer un message électronique.

[a461] L'attribut `FulfillOn` définit la valeur de la décision d'autorisation pour laquelle cette obligation doit être remplie. Dans ce cas, lorsque l'accès est permis.

[a462] – [a467] Le premier paramètre indique où le PEP trouvera l'adresse de messagerie électronique dans la ressource.

[a468] – [a471] Le second paramètre contient le texte littéral pour le corps du message électronique.

[a472] – [a476] Le troisième paramètre indique où le PEP trouvera plus de texte pour le corps du message électronique dans la ressource.

II.2.4.4 Règle 4

La règle 4 illustre l'utilisation de la valeur `Effect` (d'effet) "Deny" et une `<Rule>` sans élément `<Condition>`.

```
[a480] <?xml version="1.0" encoding="UTF-8"?>
[a481] <Policy
[a482] xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
[a483] xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-policy-schema-
os.xsd"
[a484] xmlns:md="http:www.med.example.com/schemas/record.xsd"
[a485] PolicyId="urn:oasis:names:tc:xacml:2.0:example:policyid:4"
[a486] RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
algorithm:deny-overrides">
[a487]  <PolicyDefaults>
[a488]   <XPathVersion>http://www.w3.org/TR/1999/Rec-xpath-
19991116</XPathVersion>
[a489]  </PolicyDefaults>
[a490]  <Target/>
[a491]  <Rule
[a492]  RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:4"
[a493]  Effect="Deny">
[a494]   <Description>
[a495]    An Administrator shall not be permitted to read or write
[a496]    medical elements of a patient record in the
[a497]    http://www.med.example.com/records.xsd namespace.
[a498]   </Description>
[a499]   <Target>
[a500]    <Subjects>
[a501]     <Subject>
[a502]      <SubjectMatch
[a503]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a504]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a505]        administrator
[a506]       </AttributeValue>
```

```
[a507]       <SubjectAttributeDesignator AttributeId=
[a508]       "urn:oasis:names:tc:xacml:2.0:example:attribute:role"
[a509]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a510]      </SubjectMatch>
[a511]     </Subject>
[a512]    </Subjects>
[a513]    <Resources>
[a514]     <Resource>
[a515]      <ResourceMatch
[a516]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a517]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a518]        urn:example:med:schemas:record
[a519]       </AttributeValue>
[a520]       <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:target-namespace"
[a521]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a522]      </ResourceMatch>
[a523]      <ResourceMatch
[a524]      MatchId="urn:oasis:names:tc:xacml:1.0:function:xpath-node-
match">
[a525]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a526]        /md:record/md:medical
[a527]       </AttributeValue>
[a528]       <ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:xpath"
[a529]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a530]      </ResourceMatch>
[a531]     </Resource>
[a532]    </Resources>
[a533]    <Actions>
[a534]     <Action>
[a535]      <ActionMatch
[a536]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a537]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a538]        read
[a539]       </AttributeValue>
[a540]       <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
[a541]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a542]      </ActionMatch>
[a543]     </Action>
[a544]     <Action>
[a545]      <ActionMatch
[a546]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a547]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a548]        write
[a549]       </AttributeValue>
[a550]       <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
[a551]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a552]      </ActionMatch>
[a553]     </Action>
[a554]    </Actions>
[a555]   </Target>
[a556]  </Rule>
[a557] </Policy>
```

[a492] – [a493] La déclaration d'élément `<Rule>`.

[a493] Effet de la règle. Chaque règle qui évalue à "True" émet l'effet de la règle comme sa valeur. Cet `Effect` de règle est "Deny" qui signifie que selon cette règle, l'accès doit être refusé quand elle évalue à "True".

[a494] – [a498] Description de forme libre de la règle.

[a499] – [a555] Cible de la règle. La cible de la règle définit l'ensemble de demandes de décision qui sont applicables à la règle.

[a502] – [a510] L'élément <SubjectMatch> vise la règle aux sujets dont l'attribut de sujet "urn:oasis:names:tc:xacml:2.0:example:attribute:role" est égal à "administrator".

[a513] – [a532] L'élément <Resources> contient un élément <Resource>, qui (à son tour) contient deux éléments <ResourceMatch>. La cible correspond si la ressource identifiée par le contexte de la demande satisfait aux deux critères de correspondance de la ressource.

[a515] – [a522] Le premier élément <ResourceMatch> vise la règle aux ressources dont l'attribut de ressource "urn:oasis:names:tc:xacml:2.0:resource:target-namespace" est égal à "urn:example:med:schemas:record".

[a523] – [a530] Le second élément <ResourceMatch> vise la règle aux éléments XML qui satisfont à l'expression XPath "/md:record/md:medical".

[a533] – [a554] L'élément <Actions> contient deux éléments <Action>, dont chacun contient un élément <ActionMatch>. La cible correspond si l'action identifiée dans le contexte de la demande satisfait chacun des critères de correspondance d'action.

[a535] – [a552] Les éléments <ActionMatch> visent la règle aux actions dont l'attribut d'action "urn:oasis:names:tc:xacml:1.0:action:action-id" est égal à "read" ou "write".

Cette règle n'a pas d'élément <Condition>.

II.2.4.5 Exemple d'ensemble de politique

Le présent paragraphe utilise les exemples des paragraphes précédents pour illustrer le processus de combinaison de politiques. La politique qui gouverne l'accès en lecture aux éléments médicaux d'un dossier est formée de chacune des quatre règles décrites au § 4.2.3. En langage clair la règle combinée est:

- le demandeur est le patient; ou
- le demandeur est le parent ou gardien et le patient a moins de 16 ans; ou
- le demandeur est le médecin de soin principal et une notification est envoyée au patient; et
- le demandeur n'est pas un administrateur.

L'ensemble de politiques suivant illustre les politiques combinées. La politique 3 est incluse par référence et la politique 2 est explicitement incluse.

```
[a558] <?xml version="1.0" encoding="UTF-8"?>
[a559]  <PolicySet
[a560]  xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
[a561]  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
http://docs.oasis-open.org/xacml/access_control-xacml-2.0-policy-schema-
os.xsd"
[a562]  PolicySetId=
[a563]  "urn:oasis:names:tc:xacml:2.0:example:policysetid:1"
[a564]  PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:
[a565]  policy-combining-algorithm:deny-overrides">
[a566]   <Description>
[a567]    Example policy set.
[a568]   </Description>
[a569]   <Target>
[a570]    <Resources>
[a571]     <Resource>
[a572]      <ResourceMatch
[a573]      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
[a574]       <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
[a575]        urn:example:med:schema:records
[a576]       </AttributeValue>
[a577]       <ResourceAttributeDesignator AttributeId=
[a578]       "urn:oasis:names:tc:xacml:1.0:resource:target-namespace"
[a579]       DataType="http://www.w3.org/2001/XMLSchema#string"/>
[a580]      </ResourceMatch>
[a581]     </Resource>
[a582]    </Resources>
[a583]   </Target>
[a584]  <PolicyIdReference>
```

```
[a585]   urn:oasis:names:tc:xacml:2.0:example:policyid:3
[a586]  </PolicyIdReference>
[a587]  <Policy
[a588]  PolicyId="urn:oasis:names:tc:xacml:2.0:example:policyid:2"
[a589]  RuleCombiningAlgId=
[a590] "urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-
overrides">
[a591]   <Target/>
[a592]   <Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:1"
[a593]   Effect="Permit">
[a594]   </Rule>
[a595]   <Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:2"
[a596]   Effect="Permit">
[a597]   </Rule>
[a598]   <Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:ruleid:4"
[a599]   Effect="Deny">
[a600]   </Rule>
[a601]  </Policy>
[a602] </PolicySet>
```

[a559] – [a565] La déclaration d'élément `<PolicySet>`. Les déclarations d'espace de nom XML standard sont incluses.

[a562] L'attribut `PolicySetId` est utilisé pour identifier cet ensemble de politiques pour l'inclusion possible dans un autre ensemble de politiques.

[a564] L'identifiant d'algorithme de combinaison de politique. Les politiques et ensembles de politiques dans cet ensemble de politiques sont combinés conformément à l'algorithme de combinaison de politiques spécifié lorsque la décision d'autorisation est calculée.

[a566] – [a568] Description de forme libre d'ensemble de politiques.

[a569] – [a583] L'élément `<Target>` d'ensemble de politiques définit l'ensemble des demandes de décision qui sont applicables à cet élément `<PolicySet>`.

[a584] `PolicyIdReference` inclut une politique par identifiant.

[a588] La `Policy 2` (politique 2) est explicitement incluse dans cet ensemble de politiques. Les règles dans `Policy 2` sont omises pour la clarté du texte.

Appendice III

Exemple de description de fonctions sac d'ordre supérieur

III.1 Exemple de fonctions sac d'ordre supérieur

Le présent appendice décrit les fonctions dans XACML qui effectuent des opérations sur les sacs telles que des fonctions puissent s'appliquer aux sacs en général.

A titre d'exemple, un langage fonctionnel généraliste appelé Haskell (voir [Haskell]) est utilisé pour spécifier la sémantique de ces fonctions. Bien que la description en anglais soit adéquate, une spécification appropriée de la sémantique est utile.

Pour résumer, dans la notation Haskell suivante une définition de fonction prend la forme de clauses qui sont appliquées à des schémas de structures, à savoir des listes. Le symbole "[]" note la liste vide, alors que l'expression "(x:xs)" s'assortit à un argument d'une liste non vide où "x" représente le premier élément de la liste, et "xs" est le reste de la liste, qui peut être une liste vide. On utilise la notion Haskell de liste, qui est une collection ordonnée d'éléments, pour modéliser les sacs de valeurs XACML.

Une définition Haskell simple de la fonction familière "urn:oasis:names:tc:xacml:1.0:function:and" qui prend une liste de valeurs de type booléen se définit comme suit:

```
and:: [Bool]    -> Bool
and []          = True
and (x:xs)      = x && (et xs)
```

La première ligne de définition notée par un "::" décrit formellement le type de données de la fonction, qui prend une liste de booléens, notés par "[Bool]", et retourne un booléen, noté "Bool". La seconde ligne de définition est une clause qui déclare que la fonction "and" appliquée à la liste vide est "True". La troisième ligne de définition est une clause qui déclare que pour une liste non vide, telle que le premier élément soit "x", qui est une valeur de type de données Bool, la fonction "and" appliquée à x doit être combinée, en utilisant la fonction de conjonction logique, qui est notée par le symbole infixe "&&", le résultat d'une application récursive de la fonction "and" au reste de la liste. Bien sûr, une application de la fonction "and" est "True" si et seulement si la liste à laquelle elle est appliquée est vide ou si chaque élément de la liste est "True". Par exemple, l'évaluation des expressions Haskell suivantes,

(and []), (and [True]), (and [True,True]), (and [True,True,False])

évalue à "True", "True", "True" et "False", respectivement.

1) `urn:oasis:names:tc:xacml:1.0:function:any-of`

En Haskell, la sémantique de cette opération est la suivante:

```
any_of:: ( a -> b -> Bool )        -> a -> [b] -> Bool
any_of   f   a   []                              = False
any_of   f   a   (x:xs)                          = (f  a  x) || (any_of  f  a  xs)
```

Dans la notation ci-dessus, "f" est la fonction à appliquer, "a" est la valeur de primitive, et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

Par exemple, l'expression suivante doit retourner "True":

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of">
      <Function
            FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-   equal"/>
      <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            Paul
      </AttributeValue>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                  John
            </AttributeValue>
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                  Paul
            </AttributeValue>
            <AttributeValue
                  DataType="http://www.w3.org/2001/XMLSchema#string">George
            </AttributeValue>
```

```
            <AttributeValue
                  DataType="http://www.w3.org/2001/XMLSchema#string">
                  Ringo
            </AttributeValue>
      </Apply>
</Apply>
```

Cette expression est "True" parce que le premier argument est égal à au moins un des éléments du sac, conformément à la fonction.

2) `urn:oasis:names:tc:xacml:1.0:function:all-of`

En Haskell, la sémantique de cette opération est la suivante:

```
all_of:: ( a -> b -> Bool )  -> a -> [b] -> Bool
all_of   f   a   []                          = True
all_of   f   a   (x:xs)                      = (f  a  x) && (all_of  f  a  xs)
```

Dans la notation ci-dessus, "f" est la fonction à appliquer, "a" est la valeur de primitive, et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

Par exemple, l'expression suivante doit s'évaluer comme "True":

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:all-of">
      <Function FunctionId="urn:oasis:names:tc:xacml:2.0:function:integer-
      greater"/>
      <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#integer">10</AttributeValue>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#integer">9</AttributeValue>
            <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#integer">3</AttributeValue>
            <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#integer">4</AttributeValue>
            <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#integer">2</AttributeValue>
      </Apply>
</Apply>
```

Cette expression est "True" parce que le premier argument (10) est supérieur à tous les éléments du sac (9, 3, 4 et 2).

3) `urn:oasis:names:tc:xacml:1.0:function:any-of-any`

En Haskell, tirant parti de la fonction "any_of" définie ci-dessus, la sémantique de la fonction "any_of_any" est la suivante:

```
any_of_any :: ( a -> b -> Bool )     -> [a]-> [b] -> Bool
any_of_any  f  []          ys                          = False
any_of_any  f  (x:xs)      ys                          = (any_of  f  x  ys) || (any_of_any  f  xs  ys)
```

Dans la notation ci-dessus, "f" est la fonction à appliquer et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

Par exemple, l'expression suivante doit s'évaluer comme "True":

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of-any">
      <Function FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
equal"/>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Ringo</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Mary</AttributeValue>
      </Apply>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">John</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Paul</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">George</AttributeValue>
```

```
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Ringo</AttributeValue>
      </Apply>
</Apply>
```

Cette expression est "True" parce que au moins le premier élément du premier sac, à savoir "Ringo", est égal à au moins un des éléments du second sac.

4) `urn:oasis:names:tc:xacml:1.0:function:all-of-any`

En Haskell, tirant parti de la fonction "any_of" définie ci-dessus en Haskell, la sémantique de la fonction "all_of_any" est la suivante:

all_of_any:: (a -> b -> Bool) -> [a]-> [b] -> Bool

all_of_any f [] ys = True

all_of_any f (x:xs) ys = (any_of f x ys) && (all_of_any f xs ys)

Dans la notation ci-dessus, "f" est la fonction à appliquer et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

ar exemple, l'expression suivante doit s'évaluer comme "True":

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:all-of-any">
      <Function FunctionId="urn:oasis:names:tc:xacml:2.0:function:integer-
greater"/>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">10</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">20</AttributeValue>
      </Apply>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">1</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">3</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">5</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">19</AttributeValue>
      </Apply>
</Apply>
```

Cette expression est "True" parce que chacun des éléments du premier sac est supérieur à au moins un des éléments du second sac.

5) `urn:oasis:names:tc:xacml:1.0:function:any-of-all`

En Haskell, tirant parti de la fonction "all_of" définie ci-dessus en Haskell, la sémantique de la fonction "any_of_all" est la suivante:

any_of_all:: (a -> b -> Bool) -> [a]-> [b] -> Bool

any_of_all f [] ys = False

any_of_all f (x:xs) ys = (all_of f x ys) || (any_of_all f xs ys)

Dans la notation ci-dessus, "f" est le nom de la fonction à appliquer et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

Par exemple, l'expression suivante doit s'évaluer comme "True":

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:any-of-all">
      <Function FunctionId="urn:oasis:names:tc:xacml:2.0:function:integer-
greater"/>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">3</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">5</AttributeValue>
      </Apply>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">1</AttributeValue>
```

```
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">2</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">3</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">4</AttributeValue>
      </Apply>
</Apply>
```

Cette expression est "True" parce que, pour toutes les valeurs dans le second sac, il y a une valeur dans le premier sac qui est supérieure.

6) `urn:oasis:names:tc:xacml:1.0:function:all-of-all`

En Haskell, tirant parti de la fonction "all_of" définie ci-dessus en Haskell, la sémantique de la fonction "all_of_all" est la suivante:

```
all_of_all:: ( a -> b -> Bool )      -> [a] -> [b] -> Bool
all_of_all   f  []          ys                    = True
all_of_all   f  (x:xs)      ys                    = (all_of  f  x  ys) && (all_of_all  f  xs  ys)
```

Dans la notation ci-dessus, "f" est la fonction à appliquer et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

Par exemple, l'expression suivante doit s'évaluer comme "True":

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:all-of-all">
      <Function FunctionId="urn:oasis:names:tc:xacml:2.0:function:integer-
greater"/>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">6</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">5</AttributeValue>
      </Apply>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">1</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">2</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">3</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">4</AttributeValue>
      </Apply>
</Apply>
```

Cette expression est "True" parce que tous les éléments du premier sac, "5" et "6", sont chacun plus grands que toutes les valeurs entières "1", "2", "3", "4" du second sac.

7) `urn:oasis:names:tc:xacml:1.0:function:map`

En Haskell, cette fonction se définit comme suit:

```
map:: (a -> b)  -> [a] -> [b]
map  f  []          = []
map  f  (x:xs)  =  (f x): (map  f   xs)
```

Dans la notation ci-dessus, "f" est la fonction à appliquer et "(x:xs)" représente le premier élément de la liste comme "x" et le reste de la liste comme "xs".

Par exemple, l'expression suivante,

```
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:map">
      <Function FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-
normalize-to-lower-case">
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Hello</AttributeValue>
            <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">World!</AttributeValue>
      </Apply>
</Apply>
```

évalue un sac contenant "hello" et "world!".

BIBLIOGRAPHIE

[Haskell] THOMPSON (S.): Haskell: The Craft of Functional Programming (2nd Edition), (L'art de la programmation fonctionnelle (deuxième édition)), *Addison Wesley*, ISBN 0-201-34275-8, 1996.

[IEEE 754] IEEE 754-1985, *Binary Floating-Point Arithmetic*, (Arithmétique binaire à virgule flottante) ISBN 1-5593-7653-8, IEEE Product No. SH10116-TBR.

[RBAC] ANSI INCITS 359-2004, *Information technology – Role Based Access Control*, (Contrôle d'accès fondé sur le rôle), http://csrc.nist.gov/rbac/.

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	Gestion des télécommunications y compris le RGT et maintenance des réseaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	**Réseaux de données, communication entre systèmes ouverts et sécurité**
Série Y	Infrastructure mondiale de l'information, protocole Internet et réseaux de prochaine génération
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Imprimé en Suisse
Genève, 2007