TECHNICAL SPECIFICATION

ISO/TS 22600-3

First edition
2009-12-01

Health informatics — Privilege management and access control —

Part 3: Implementations

Informatique de santé — Gestion de privilèges et contrôle d'accès —

Partie 3: Mises en œuvre

Reference number
ISO/TS 22600-3:2009(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

In other circumstances, particularly when there is an urgent market requirement for such documents, a technical committee may decide to publish other types of document:

— an ISO Publicly Available Specification (ISO/PAS) represents an agreement between technical experts in an ISO working group and is accepted for publication if it is approved by more than 50 % of the members of the parent committee casting a vote;

— an ISO Technical Specification (ISO/TS) represents an agreement between the members of a technical committee and is accepted for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/PAS or ISO/TS is reviewed after three years in order to decide whether it will be confirmed for a further three years, revised to become an International Standard, or withdrawn. If the ISO/PAS or ISO/TS is confirmed, it is reviewed again after a further three years, at which time it must either be transformed into an International Standard or be withdrawn.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/TS 22600-3 was prepared by Technical Committee ISO/TC 215, *Health informatics*.

ISO/TS 22600 consists of the following parts, under the general title *Health informatics — Privilege management and access control*:

— *Part 1: Overview and policy management*

— *Part 2: Formal models*

— *Part 3: Implementations*

Introduction

A common situation today is that hospitals are supported by several vendors providing different applications, which are not able to communicate authentication and authorization since they have their own way to handle these functions. In an integrated scenario, one has to spend a huge amount of money to get users and organizational information mapped before starting the communication. Resources are required for development and maintenance of security functions, which grow exponentially with the number of applications.

On the other hand, if one looks at authorization from the healthcare-organization point of view, the need for a flexible bridging model becomes obvious from the fact that organizations change continuously. Units are closed down, opened and merged.

The situation becomes even more complex when communications over security policy domain boundaries are necessary. The policy differences of these domains then have to be bridged via system changes and/or through *policy agreements* between the parties.

Another complexity is found in functional and structural roles when it comes to users. A user can be authorized to operate under multiple functional and structural roles. While a user can concurrently hold multiple structural roles, the user can only perform under a single functional role for a given information request. The policy will define the relationship among the multiple functional roles that can be held by a user, including relationships of grouping, hierarchy, composite structural roles, functional roles taken in sequence, and roles that must not be combined. For example: if a general practitioner is also a psychiatrist, the policy can specify that the psychiatrist can be grouped with other similarly privileged structural roles, that the psychiatrist inherits the privileges of the General Practitioner, that a new composite structural role is created, or that the two structural roles cannot be combined. An example of a restriction of functional roles taken in sequence would be a conflict of duties which can be restricted by law (e.g. a requester for reimbursement cannot also be the signer for the same reimbursement). Moreover, different responsibilities can be identified in the healthcare organization regarding the role and activities of the users. Moving from country to country or from one healthcare establishment to another, different types or levels of authorization can be applied to similar types of users, both for execution of particular functions and for access to information.

Another important issue today is to improve the quality of care by using Information Technology (IT), without interfering with the respect for the privacy of the patient. To allow physicians to have more adequate information about patients, you need to have something like a “virtual electronic healthcare record” which makes it possible to keep track of all the activities belonging to one patient, regardless of where and by whom they have been documented. With such an approach, we need to have a generic model or a specific agreement between the parties for authorization.

Besides the need for support of a diversity of roles and responsibilities, which are typical in any type of large organization, additional critical aspects can be identified, such as ethical and legal aspects in the healthcare scenario due to the particular type of information that is managed.

The need for restrictive authorization is already high today but is going to dramatically increase over the next couple of years. The reason for this is the increase of exchange of information between applications, in order to fulfil physicians' demands on having access to more and more patient-related information to ensure the quality and efficiency of patient treatment.

The situation with respect to healthcare and its communication and application security services has changed during the last decade. The reasons are, for example:

— moving from mainframe-based proprietary legacy systems to distributed systems running in local environments;

— more data are stored in the information systems and are therefore also more valuable to the users;

— patients are more ambulant and in need of their medical information at different locations.

From this, it follows that advanced security is required in communication and use of health information due to the sensitivity of the person-related information and its corresponding personal and social impact. Those security services concern both communication and application security. Regarding the application security services, such as authentication, integrity, confidentiality, availability, accountability (including traceability and non-repudiation) as well as the notary's services, the first service mentioned, authentication, is of crucial importance for most of the other services. This is also valid for application security, such as access control, integrity, confidentiality, availability, accountability, audibility and the notary's services.

The implementation of a Technical Specification like this is very complex since the involved parties already have systems in operation and are not immediately willing to update these to newer versions or new systems. It is therefore very important that a policy agreement is written between the parties that states that they intend to move towards ISO/TS 22600 for any changes in the systems they are going to make.

The policy agreement must also contain defined differences in the security systems and the agreed solutions on how to overcome the differences. For example, the authentication service, privileges and duties of a requesting party at the responding site have to be managed according to the agreed policy written down in the agreement. For that reason, the information and service requester, as well as the information and service provider on the one hand and the information and services requested and provided on the other hand, have to be grouped and classified properly. Based on that classification, claimant mechanisms, target sensitivity mechanisms and policy specification and management mechanisms can be implemented. With such an all parties underwritten policy agreement, the communication and information exchange can start with the existing systems if the parties do not see any risks. If there are risks which are of such importance that they have to be eliminated before the information exchange starts, they must also be recorded in the policy agreement together with an action plan for how these risks must be removed. The policy agreement must also contain a time plan for this work and an agreement on how it must be financed.

The documentation process is a very important part of a platform for the policy agreement.

ISO/TS 22600 consists of the following parts.

Part 1: Overview and policy management: describes the scenarios and the critical parameters in the cross-border information exchange. It also gives examples of necessary documentation methods as the basis for the policy agreement.

Part 2: Formal models: describes and explains, in a more detailed manner, the architectures and underlying models for the privileges and privilege management which are necessary for secure information sharing, plus examples of policy agreement templates.

Part 3: Implementations: describes examples of implementable specifications of application security services and infrastructural services using different specification languages.

ISO/TS 22600 introduces principles and specifies services needed for managing privileges and access control. Cryptographic protocols are outside the scope of ISO/TS 22600.

ISO/TS 22600 is strongly related to other corresponding International Standards and specifications, such as ISO 17090, ISO/TS 21091 and ISO/TS 21298.

ISO/TS 22600 is meant to be read in conjunction with its complete set of associated standards.

The distributed architecture of shared care information systems with the trend to personal health supporting Service Oriented Architecture (SOA) is increasingly based on networks. Due to their user friendliness, the use of standardized user interfaces, tools and protocols, and therefore their platform independence, the number of really open information systems based on corporate networks and virtual private networks has been rapidly growing during the last couple of years.

This part of ISO/TS 22600 is intended to support the needs of healthcare information sharing across unaffiliated providers of healthcare, healthcare organizations, health insurance companies, their patients, staff members and trading partners.

This part of ISO/TS 22600 is intended to support enquiries from both individuals and application systems.

ISO/TS 22600 defines methods for managing authorization and access control to data and/or functions. It allows policy bridging. It is based on a conceptual model where local authorization manager servers and a cross-border directory server can assist access control in various applications (software components). This directory server provides information on rules for access to various application functions based on roles and other attributes of the individual user. The granted access will be based on the following aspects:

— the authenticated identification of the user;

— the rules for access connected with a specific information object;

— the rules regarding authorization attributes linked to the user provided by the authorization manager;

— the functions of the specific application.

This part of ISO/TS 22600 is used in a perspective ranging from a local situation to a regional or national situation. One of the key points in these perspectives is to have organizational criteria combined with authorization profiles, agreed upon from both the requesting and the delivering side in a written policy agreement.

This part of ISO/TS 22600 supports collaboration between several authorization managers that can operate over organizational and policy borders.

The collaboration is defined in a *policy agreement*, signed by all of the involved organizations, which constitutes the basic platform for the operation.

A documentation format is proposed as a platform for the policy agreement, which makes it possible to obtain comparable documentation from all parties involved in the information exchange.

Based on the aforementioned unified process, a three-dimensional architectural reference model has been derived for defining the constraint models needed. The dimensions of the Generic Component Model used are the domain axis, the decomposition/composition axis and the axis describing the views on a system and its components. For it to be future-proof, sustainable, flexible, portable and scalable, only the constraining process and the resulting security-related meta-models are presented. The instantiation and implementation, e.g. the specification of mechanisms and encoding definitions, is a long-term process, dedicated to other standards and projects or the vendor/provider community, respectively.

After summarizing the ISO/TS 22600-2 basics, the different ways of representing different levels of maturity with different levels of interoperability below the ideal situation of a semantically valid one are discussed.

For those different environments and levels, this part of ISO/TS 22600 introduces examples for specializing and implementing the formal high-level models for architectural components based on ISO/IEC 10746 and defined in ISO/TS 22600-2. These examples and related services are grouped in different annexes.

The specifications are provided using derivates of eXtensible Markup Language (XML), especially SAML (Security Assertion Markup Language) and XACML (eXtensible Access Control Markup Language) specified by OASIS. Additional specifications are also presented in the traditional ASN.1 syntax.

ISO/TS 22600 has been harmonized in essential parts with ASTM E2595-07.

Health informatics — Privilege management and access control —

Part 3: Implementations

1 Scope

This part of ISO/TS 22600 instantiates requirements for repositories for access control policies and requirements for privilege management infrastructures for health informatics. It provides implementation examples of the formal models specified in ISO/TS 22600-2.

This part of ISO/TS 22600 is strongly related to other ISO/TC 215 documents, such as ISO 17090, ISO 22857 and ISO/TS 21091. It is also related to ISO/TS 21298.

This part of ISO/TS 22600 excludes platform-specific and implementation details. It does not specify technical communication security services, authentication techniques and protocols that have been established in other standards such as, for example, ISO 7498-2, ISO/IEC 10745 (ITU-T X.803), ISO/IEC TR 13594 (ITU-T X.802), ISO/IEC 10181-1 (ITU-T X.810), ISO/IEC 9594-8 Authentication framework (equivalent to ITU-T X.509), ISO/IEC 9796, ISO/IEC 9797 and ISO/IEC 9798.

ISO/TS 22600 defines privilege management and access control services required for the communication and use of distributed health information over domain and security borders. ISO/TS 22600 introduces principles and specifies services needed for managing privileges and access control. It specifies the necessary component-based concepts and is intended to support their technical implementation. It does not specify the use of these concepts in particular clinical process pathways nor does it address the safety concerns, if any, associated with their use.

While ISO/TS 22600-1 is a narrative introducing the problem of policy bridging in the context of inter-organizational communication and cooperation, ISO/TS 22600-2 defines a generic development process for analysing, designing, implementing and semantically deploying health information systems. The security services needed due to legal, social, organizational, user-related, functional and technological requirements have to be embedded in the advanced and sustainable system architecture meeting the paradigms for semantic interoperability.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 8601:2004, *Data elements and interchange formats — Information interchange — Representation of dates and times*

ISO/IEC 9594-8:2001, *Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks* (also available as ITU-T X.509: 2000)

ISO/IEC 10181-3:1996, *Information technology — Open Systems Interconnection — Security frameworks for open systems: Access control framework* (also available as ITU-T X.812: 1995)

ISO/TS 21298:2008, *Health informatics — Functional and structural roles*

ASTM E2595-07, *Standard Guide for Privilege Management Infrastructure*

ASTM E1762-07, *Standard Guide for Electronic Authentication of Health Care Information*

ASTM E1986-98. *Standard Guide for Information Access Privileges to Health Information*

ASTM E2212-02a, *Standard Practice for Healthcare Certificate Policy*

OASIS, *eXtensible Access Control Markup Language* (XACML) v2.0, February 2005

OASIS, *XACML Profile for Role Based Access Control* (RBAC): Committee Draft 01 (normative, 13 February 2004)

OASIS, *Security Assertion Markup Language* (SAML), Version 2.0, March 2005

OASIS 200306, *Service Provisioning Markup Language* (SPML), V1.0, October 2003

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
access control
means of ensuring that the resources of a data processing system can be accessed only by authorized entities in authorized ways

[ISO/IEC 2382-8:1998]

3.2
access control decision function
ADF
specialized function that makes access control decisions by applying access control policy rules to a requested action

3.3
access control enforcement function
AEF
specialized function that is part of the access path between a requester and a protected resource that enforces the decisions made by the ADF

3.4
access control information
any information used for access control purposes, including contextual information

3.5
accountability
property that ensures that the actions of an entity may be traced uniquely to the entity

[ISO 7498-2:1989]

3.6
asymmetric cryptographic algorithm
algorithm for performing encipherment or the corresponding decipherment in which the keys used for encipherment and decipherment differ

[ISO/IEC 10181-1:1996]

3.7
attribute authority
AA
authority which assigns privileges by issuing attribute certificates

[ISO/IEC 9594-8:2001]

3.8
attribute authority revocation list
AARL
revocation list containing a list of references to attribute certificates issued to AAs that are no longer considered valid by the certificate-issuing authority

3.9
attribute certificate
data structure, digitally signed by an attribute authority, that binds some attribute values with identification about its holder

[ISO/IEC 9594-8:2001]

3.10
attribute certificate revocation list
ACRL
revocation list containing a list of references to attribute certificates that are no longer considered valid by the certificate-issuing authority

3.11
authentication
process of reliably identifying security subjects by securely associating an identifier and its authenticator

NOTE See also **data origin authentication** (3.50).

[ISO 7498-2:1989]

3.12
authentication token
information conveyed during a strong authentication exchange, which can be used to authenticate its sender

3.13
authority
entity, which is responsible for the issuance of certificates

NOTE Two types of authority are defined in this part of ISO/TS 22600: certification authority, which issues public key certificates, and attribute authority, which issues attribute certificates.

3.14
authority certificate
certificate issued to a certification authority or an attribute authority

NOTE Adapted from ISO/IEC 9594-8:2001.

3.15
authority revocation list
ARL
revocation list containing a list of public key certificates issued to authorities, which are no longer considered valid by the certificate issuer

3.16
authorization
granting of privileges, which includes the granting of access based on access privileges or conveyance of privileges from one entity that holds higher privileges, to another entity holding lower privileges

NOTE Adapted from ISO 7498-2:1989.

3.17
authorization credential
signed assertion of a user's permission attributes

3.18
availability
property of being accessible and usable upon demand by an authorized entity

[ISO 7498-2:1989]

3.19
base CRL
CRL that is used as the foundation in the generation of a dCRL

3.20
business partner agreement
document used to demarcate the legal, ethical and practical responsibilities between subscribers to a PMI and between cooperating PMI implementations

3.21
CA certificate
certificate for one CA issued by another CA

3.22
certificate
public key certificate

3.23
certificate distribution
act of publishing certificates and transferring certificates to security subjects

3.24
certificate holder
entity that is named as the subject of a valid certificate

3.25
certificate management
procedures relating to certificates: certificate generation, certificate distribution, certificate archiving and revocation

3.26
certificate policy
named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements

EXAMPLE A particular certificate policy might indicate the applicability of a type of certificate to the authentication of electronic data interchange transactions for the trading of goods within a given price range.

3.27
certificate revocation
act of removing any reliable link between a certificate and its certificate holder because the certificate is not trusted any more, whereas it is unexpired

3.28
certificate revocation list
CRL
assigned list indicating a set of certificates that are no longer considered valid by the certificate issuer

NOTE In addition to the generic term CRL, some specific CRL types are defined for CRLs that cover particular scopes. A published list of the suspended and revoked certificates exists (digitally signed by the CA).

3.29
certificate serial number
integer value, unique within the issuing authority, which is unambiguously associated with a certificate issued by that CA

3.30
certificate suspension list
CSL
published list of the suspended certificates (digitally signed by the CA)

3.31
certificate user
entity that needs to know, with certainty, the public key of another entity

3.32
certificate using system
implementation of those functions defined in this Directory Specification that are used by a certificate user

3.33
certificate validation
process of ensuring that a certificate was valid at a given time, possibly including the construction and processing of a certification path, and ensuring that all certificates in that path were valid (i.e. were not expired or revoked) at that given time

3.34
certificate verification
verification that a certificate is authentic

3.35

3.35.1
certification authority
CA
certificate issuer; an authority trusted by one or more relying parties to create and assign certificates; optionally the certification authority may create the relying parties' keys

NOTE Adapted from ISO/IEC 9594-8:2001.

3.35.2
certification authority
entity that issues certificates by signing certificate data with its private signing key.

NOTE Authority in the CA term does not imply any government authorization; it only means that it is trusted. Certificate issuer may be a better term but CA is used very broadly.

3.36
certification authority revocation list
CARL
revocation list containing a list of public key certificates issued to certification authorities, which are no longer considered valid by the certificate issuer

3.37
certification path
ordered sequence of certificates of objects in the DIT which, together with the public key of the initial object in the path, can be processed to obtain that of the final object in the path

3.38
ciphertext
data produced through the use of encipherment

NOTE The semantic content of the resulting data is not available.

[ISO 7498-2:1989]

3.39
claimant
entity requesting that a sensitive service be performed or provided by a verifier, based on the claimant's privileges as identified in their attribute certificate or subject directory attributes extension of their public key certificate

3.40
confidentiality
property indicating that information is not made available or disclosed to unauthorized individuals, entities or processes

[ISO 7498-2:1989]

3.41
consent
special policy which defines an agreement between an entity playing the role of the subject of an act and an entity acting

3.42
credential
information describing the security attributes (identity or privilege or both) of a principal, which is a prerequisite for the entitlement of, or the eligibility for, a role

NOTE Credentials are claimed through authentication or delegation and used by access control.

3.43
CRL distribution point
directory entry or other distribution source for CRLs

NOTE A CRL distributed through a CRL distribution point may contain revocation entries for only a subset of the full set of certificates issued by one CA or may contain revocation entries for multiple CAs.

3.44
cryptography
discipline which embodies principles, means, and methods for the transformation of data in order to hide its information content, prevent its undetected modification and/or prevent its unauthorized use

[ISO 7498-2:1989]

3.45
cryptographic algorithm
cipher
method for the transformation of data in order to hide its information content, prevent its undetected modification and/or prevent its unauthorized use

[ISO 7498-2:1989]

3.46
cryptographic system
cryptosystem
collection of transformations from plaintext into ciphertext and vice versa, the particular transformation(s) to be used being selected by keys

NOTE The transformations are normally defined by a mathematical algorithm.

3.47
data confidentiality
service that can be used to provide for protection of data from unauthorized disclosure

NOTE The data confidentiality service is supported by the authentication framework. It can be used to protect against data interception.

3.48
data integrity
property that data has not been altered or destroyed in an unauthorized manner

[ISO 7498-2:1989]

3.49
data origin authentication
corroboration that the source of data received is as claimed

[ISO 7498-2:1989]

3.50
decipherment
decryption
process of obtaining, from a ciphertext, the original corresponding data

NOTE 1 A ciphertext may be enciphered a second time, in which case a single decipherment does not produce the original plaintext.

NOTE 2 Adapted from ISO/IEC 2382-8:1998.

3.51
delegation
conveyance of privilege from one entity that holds such a privilege to another entity

3.52
delegation path
ordered sequence of certificates which, together with authentication of a privilege asserter's identity, can be processed to verify the authenticity of a privilege asserter's privilege

3.53
delta CRL
dCRL
partial revocation list that only contains entries for certificates that have had their revocation status changed since the issuance of the referenced base CRL

3.54
digital signature
data appended to, or a cryptographic transformation (see cryptography) of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery, e.g. by the recipient

[ISO 7498-2:1989]

3.55
encipherment
encryption
cryptographic transformation of data (see cryptography) to produce ciphertext

[ISO 7498-2:1989]

3.56
end entity
certificate subject that uses its private key for purposes other than signing certificates or an entity that is a relying party

3.57
end-entity attribute certificate revocation list
EARL
revocation list containing a list of attribute certificates that are no longer considered valid by the certificate issuer and that were issued to certificate holders that were not also AAs

3.58
end-entity public key certificate revocation list
EPRL
revocation list containing a list of public key certificates, issued to subjects that are not also CAs, which are no longer considered valid by the certificate issuer

3.59
environmental variables
those aspects of policy required for an authorization decision that are not contained within static structures, but are available through some local means to a privilege verifier (e.g. time of day or current account balance)

3.60
full CRL
complete revocation list that contains entries for all certificates that have been revoked for the given scope

3.61
functional role
roles bound to an act and that can be assigned to be performed during an act

NOTE 1 Functional roles correspond to the RIM participation.

NOTE 2 Adapted from ISO/TS 21298:2008.

NOTE 3 See also **structural role** (3.105).

3.62
hash function
(mathematical) function which maps values from a large (possibly very large) domain into a smaller range

NOTE A “good” hash function is such that the results of applying the function to a (large) set of values in the domain will be evenly distributed (and apparently at random) over the range.

3.63
holder
entity to whom some privilege has been delegated either directly from the source of authority or indirectly through another attribute authority

3.64
identification
performance of tests to enable a data processing system to recognize entities

[ISO/IEC 2382-8:1998]

3.65
identifier
piece of information used to claim an identity, before a potential corroboration by a corresponding authenticator

[EN 13608-1:2007]

3.66
indirect CRL
iCRL
revocation list that at least contains revocation information about certificates issued by authorities other than that which issued this CRL

3.67
integrity
proof that the message content has not altered, deliberately or accidentally in any way during transmission

[ISO 7498-2:1989]

3.68
key
sequence of symbols that controls the operations of encipherment and decipherment

[ISO 7498-2:1989]

3.69
key agreement
method for negotiating a key value online without transferring the key, even in an encrypted form, e.g. the Diffie-Hellman technique

NOTE See ISO/IEC 11770-1 for more information on key agreement mechanisms.

3.70
key management
generation, storage, distribution, deletion, archiving and application of keys in accordance with a security policy

[ISO 7498-2:1989]

3.71
lightweight directory access protocol
LDAP
standard access protocol for directories allowing public or controlled access to certificates and other information needed in a PKI

3.72
non-repudiation
service providing proof of the integrity and origin of data (both in an unforgeable relationship) which can be verified by any party

3.73
object identifier
OID
unique alphanumeric/numeric identifier registered under the ISO registration standard to reference a specific object or object class

NOTE The object identifier is a name for a certificate policy that is recorded in a field of each certificate issued in conformance with the policy.

3.74
object method
action that can be invoked on a resource

EXAMPLE A file system may have read, write and execute object methods.

3.75
one-way function
(mathematical) function f which is easy to compute, but which, for a general value y in the range, it is computationally difficult to find a value x in the domain such that $f(x) = y$

NOTE There may be a few values y for which finding x is not computationally difficult.

3.76
permission
approval for performing an operation on one or more RBAC protected objects

3.77
policy
set of legal, political, organizational, functional and technical obligations or omissions for communication and cooperation

3.78
policy agreement
written agreement where all involved parties commit themselves to a specified set of policies

3.79
policy decision point
PDP
system entity that evaluates an applicable policy and renders an authorization decision

NOTE 1 This term is defined differently in RFC 3198 [45].

NOTE 2 This term corresponds to “Access Decision Function” (ADF) in ISO/IEC 10181-3:1996.

3.80
policy enforcement point
PEP
system entity that performs access control, by making decision requests and enforcing authorization decisions

NOTE 1 This term is defined in a joint effort by the IETF Policy Framework Working Group and the Distributed Management Task Force (DMTF)/Common Information Model (CIM) in RFC 3198.

NOTE 2 This term corresponds to “Access Enforcement Function” (AEF) in ISO/IEC 10181-3:1996.

3.81
policy mapping
recognizing that, when a CA in one domain certifies a CA in another domain, a particular certificate policy in the second domain may be considered by the authority of the first domain to be equivalent to (but not necessarily identical to in all respects) a particular certificate policy in the first domain

3.82
principal
actor able to realise specific scenarios (user, organization, system, device, application, component, object)

3.83
private key
key that is used with an asymmetric cryptographic algorithm and whose possession is restricted (usually to only one entity)

[ISO/IEC 10181-1:1996]

3.84
privilege
capacity assigned to an entity by an authority according to the entity's attribute

3.85
privilege asserter
privilege holder using their attribute certificate or public key certificate to assert privilege

3.86
privilege management infrastructure
PMI
infrastructure able to support the management of privileges in support of a comprehensive authorization service and in relationship with a public key infrastructure

3.87
privilege policy
policy that outlines conditions for privilege verifiers to provide/perform sensitive services to/for qualified privilege asserters

NOTE Privilege policy relates attributes associated with the service, as well as attributes associated with privilege asserters.

3.88
privilege verifier
entity verifying certificates against a privilege policy

3.89
public key
key that is used with an asymmetric cryptographic algorithm and that can be made publicly available

[ISO/IEC 10181-1:1996]

3.90
public key certificate
X.509 public key certificates (PKCs) [X.509], binding an identity and a public key

NOTE The identity may be used to support identity-based access control decisions after the client proves that it has access to the private key that corresponds to the public key contained in the PKC.

3.91
public key infrastructure
PKI
infrastructure used in the relation between a key holder and a relying party that allows a relying party to use a certificate relating to the key holder for at least one application using a public-key-dependent security service

NOTE PKI includes a certification authority, a certificate data structure, means for the relying party to obtain current information on the revocation status of the certificate, a certification policy and methods to validate the certification practice.

3.92
relying party
recipient of a certificate who acts in reliance on that certificate and/or digital signature verified using that certificate

3.93
role
set of competences and/or performances that are associated with a task

NOTE For managing role relationships between the entities, structural and functional roles can be defined.

[ISO/TS 21298:2008].

3.94
role assignment certificate
certificate that contains the role attribute, assigning one or more roles to the certificate holder

3.95
role certificate
certificate that assigns privileges to a role rather than directly to individuals

NOTE Individuals assigned to that role, through an attribute certificate or public key certificate with a subject directory attributes extension containing that assignment, are indirectly assigned the privileges contained in the role certificate.

3.96
role specification certificate
certificate that contains the assignment of privileges to a role

3.97
sensitivity
characteristic of a resource that implies its value or importance

3.98
security
combination of availability, confidentiality, integrity and accountability

3.99

3.99.1
security policy
plan or course of action adopted for providing computer security

[ISO/IEC 2382-8:1998]

3.99.2
security policy
set of rules laid down by the security authority governing the use and provision of security services and facilities

3.100
security service
service, provided by a layer of communicating open systems, which ensures adequate security of the systems or of data transfers

[ISO 7498-2:1989]

3.101
simple authentication
authentication by means of simple password arrangements

3.102
source of authority
SoA
attribute authority that a privilege verifier for a particular resource trusts as the ultimate authority to assign a set of privileges, or a special type of attribute authority upon which a verifier endows unlimited privilege

NOTE The verifier trusts the source of authority to delegate that privilege to certificate holders, some of which may further delegate that privilege to other certificate holders.

3.103
strong authentication
authentication by means of cryptographically derived credentials

3.104
structural role
structural roles specify relations between entities in the sense of competence (RIM roles), often reflecting organizational or structural relations (hierarchies)

[ISO/TS 21298:2008]

NOTE See also **functional role** (3.61).

3.105
target
resource being accessed by a claimant

NOTE Its sensitivity is modelled in this part of ISO/TS 22600 as a collection of attributes, represented as either ASN.1 attributes or XML elements.

3.106
trust
an entity can be said to "trust" a second entity when it (the first entity) makes the assumption that the second entity will behave exactly as the first entity expects

NOTE This trust may apply only for some specific function. The key role of trust in this framework is to describe the relationship between an authenticating entity and an authority; an entity shall be certain that it can trust the authority to create only valid and reliable certificates.

3.107
third party
party other than the data originator, or data recipient, required to perform a security function as part of a communication protocol

3.108
trusted third party
TTP
third party which is considered to be trusted for purposes of a security protocol

[EN 13608-1:2007]

NOTE This term is used in many ISO/IEC standards and other documents describing mainly the services of a CA. The concept is, however, broader and includes services like time stamping and possibly escrowing. TTPs provide basic services, infrastructural services and value-added services.

3.109
verifier
entity responsible for performing or providing a sensitive service for/to qualified claimants

NOTE The verifier enforces the privilege policy. When validating certification paths, a verifier is a type of relying party.

4 Abbreviations

This list of abbreviations includes all the abbreviations used in the three parts of ISO/TS 22600.

AA	Attribute Authority
AARL	Attribute Authority Revocation List
ACI	Access Control Information
ACRL	Attribute Certificate Revocation List
ADF	Access Decision Function
ADI	Access Control Decision Information
AEF	Access Enforcement Function
ANSI	American National Standards Institute
ARL	Authority Revocation List
CA	Certification Authority
CARL	Certification Authority Revocation List
CIM	Common Information Model
CORBA	Common Object Request Broker Architecture
CRL	Certificate Revocation List
dCRL	Delta Certificate Revocation List
DAP	Directory Access Protocol
DEA	Drug Enforcement Administration
DIB	Directory Information Base
DIT	Directory Information Tree
DMTF	Distributed Management Task Force
DSA	Directory System Agent
DTD	Data Type Definition
DUA	Directory User Agent
EARL	End-entity Attribute Certificate Revocation List
EHR	Electronic Health Record
EPRL	End-entity Public Key Certificate Revocation List
HL7	Health Level Seven
iCRL	Indirect Certificate Revocation List
IETF	Internet Engineering Task Force
IT	Information Technology

LDAP	Lightweight Directory Access Protocol
OASIS	Organization for the Advancement of Structured Information Standards
OCSP	Online Certificate Status Protocol
OMG	Object Management Group
PA	Privilege Allocator
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PKC	Public Key Certificate
PKCS	Public Key Cryptosystem
PKI	Public Key Infrastructure
PMI	Privilege Management Infrastructure
PPS	Permission Policy Set
RA	Registration Authority
RBAC	Role-Based Access Control
RPS	Role Policy Set
S/MIME	Secure Multipurpose Internet Mail Extensions
SAML	Security Assertion Markup Language
SOA	Service-Oriented Architecture
SoA	Source of Authority
SPML	Service Provisioning Markup Language
TTP	Trusted Third Party
UDDI	Universal Description, Discovery and Integration
UHID	Universal Healthcare Identifier
UML	Unified Modeling Language
URI	Uniform Resource Identifier
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

5 Structures and services for privilege management and access control

Privilege management and access control are ruled by policies, which should be formally expressed for enabling interoperability. In reference to the Generic Component Model, the base class structure of policies has been defined in ISO/TS 22600-2 (see Figure 1). As mentioned in ISO/TS 22600-2, policies can be represented differently. For example, OASIS WS-policy provides a general-purpose model and syntax to describe and communicate the policies of a web service. It specifies a set of common message policy assertions within a policy and attachment mechanisms for using policy expressions with existing XML service technologies. This part of ISO/TS 22600 does not enforce a policy representation language.

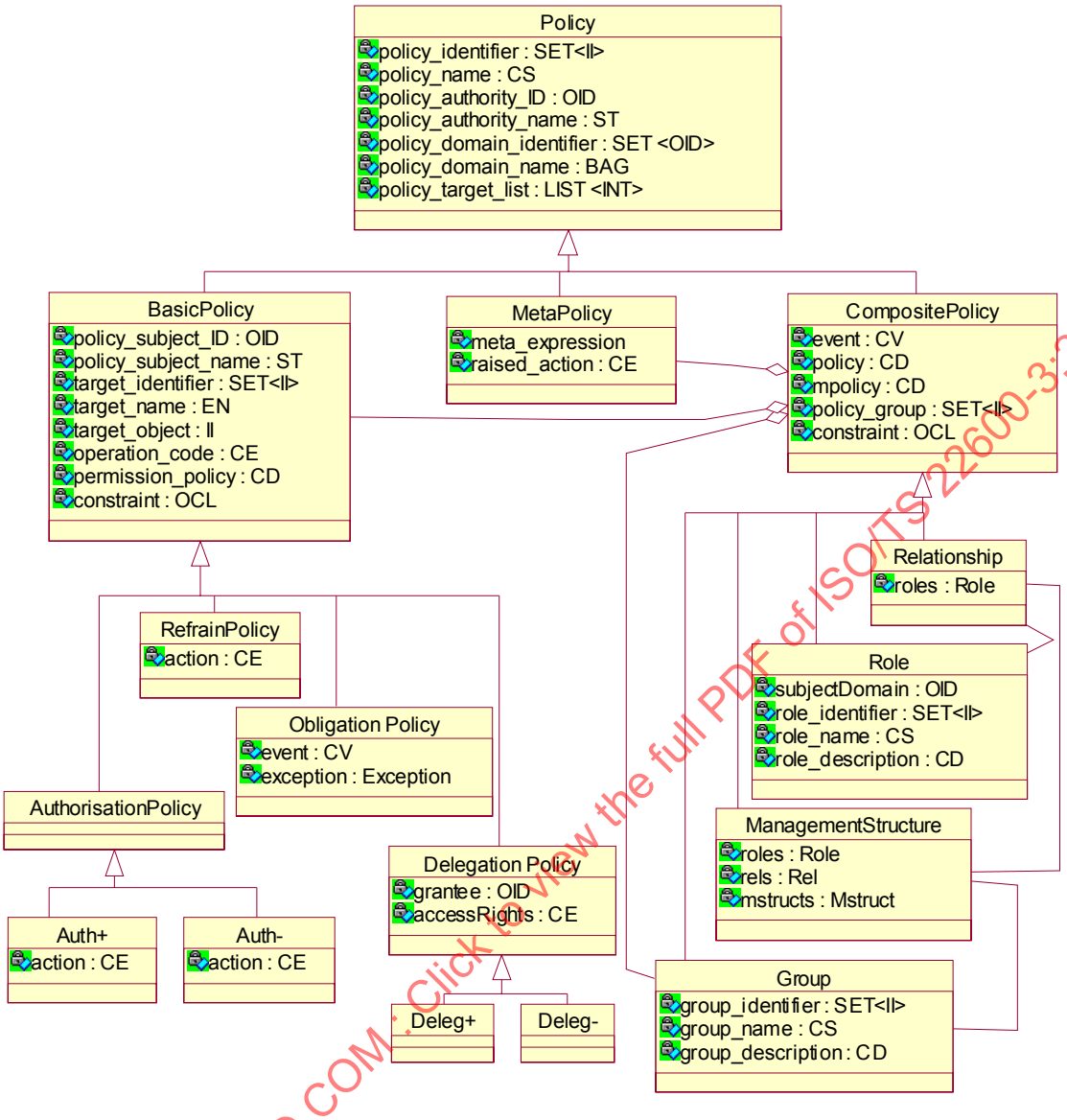

Figure 1 — Policy base class diagram

Privilege management and access control are based on a series of infrastructural services directly or indirectly related to security and privacy services. In that context, a series of PKI services and trusted third party services have to be mentioned, such as ID management, role management, privilege management, policy management, object management, etc. Additionally, services such as audits, LDAPs, intrusion detection, etc., have to be discussed. Instantiations including the services' mechanisms or encoding definitions are dynamically changing or informative issues deferred to the annexes. This part of ISO/TS 22600 has been harmonized as much as possible with ASTM E2595-07.

Regarding the specific requirements and conditions of healthcare, the underlying security model shall consider the whole spectrum of security services and mechanisms that can be accomplished by secure micro-domains.

a) Interaction between security domains

Separate security domains in the domain model can exchange privilege information by agreement of the parties concerned. This interaction between security domains shall be coordinated on both a technical and documentary level. The creating and exchange of privilege sets should take the organizational structure into consideration.

b) Technical basis

Exchanges of privilege information shall be examined to ensure that the meaning of privileges is consistent between security domains. This can be accomplished by creating a standard set of privileges. The standard set of privileges may include a mutually defined mapping of equivalent privileges between the domains. The equivalence of the exchanged privileges shall be reviewed on a technical basis to ensure that the intended security implications are achieved.

c) Administrative basis

1) Privilege information exchanged between security domains may involve separate administrative entities (for example, distinct business partners or companies). An agreement as to the exchange of privileges and their use shall be documented, typically in a "business partner agreement". The use of a business partner agreement is required to distinguish the legal, ethical, and practical responsibilities between business partners and that may extend between other cooperating PMI implementations. An equivalent procedure is performed in a PKI through the use of a certificate practices statement and certificate policies. An alternative procedure uses "policy assertions" in WS-policy.

2) Multiple security domains may exist within a single company or organization. An agreement documenting responsibilities between such domains should also be set forth in a business partner agreement or a memorandum of understanding (MOU). The document should be periodically reviewed to ensure that privileges extended across security domains exist only as long as required to meet the needs of the enterprise.

d) Organizational considerations

Privilege information exchanged between security domains should be structured to reflect organizational considerations. Establishing a security domain that encompasses an organizational objective (for example, accounting or human resources) is an essential element of a coherent approach. The resulting standard set of privileges suitable for inter-domain exchange, coupled with other environmental factors (e.g. threats), will then, as a result, be highly cohesive. That is, the privilege set provides privileges to a subset of the organization (for example, accounting) without extending privileges required in an unrelated subset of the organization (for example, human resources). In addition, a cohesive privilege set provides all privileges that are required to meet a specific objective.

This part of ISO/TS 22600 is restricted to those issues presented in the shaded boxes in Figure 2.

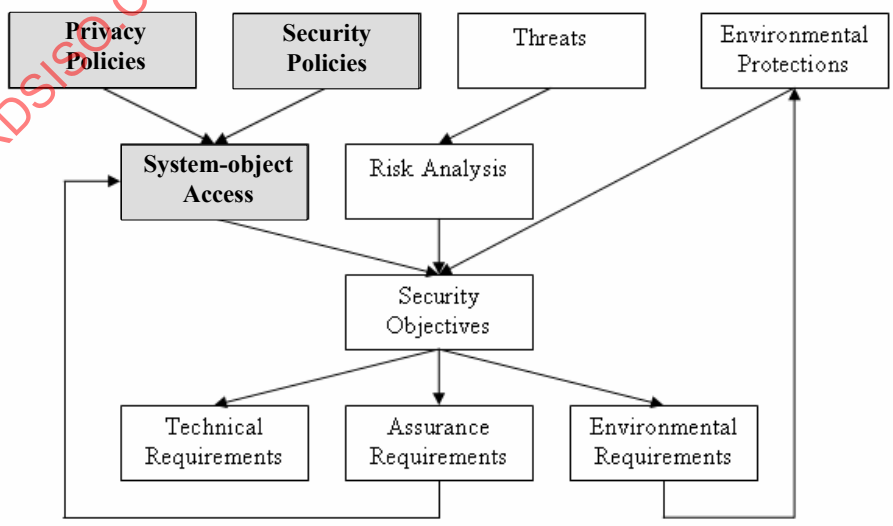

Figure 2 — Security issues that this part of ISO/TS 22600 deals with

All management services include creation, naming, definition, grouping/classification, assignment, maintenance, correlation, synchronization and deactivation, etc.

Based on the given policy model, privilege management and access control have to be policy-driven as expressed in ISO/TS 22600-1 and ISO/TS 22600-2, as well as in Figure 3 ([2, 3] in the Bibliography).

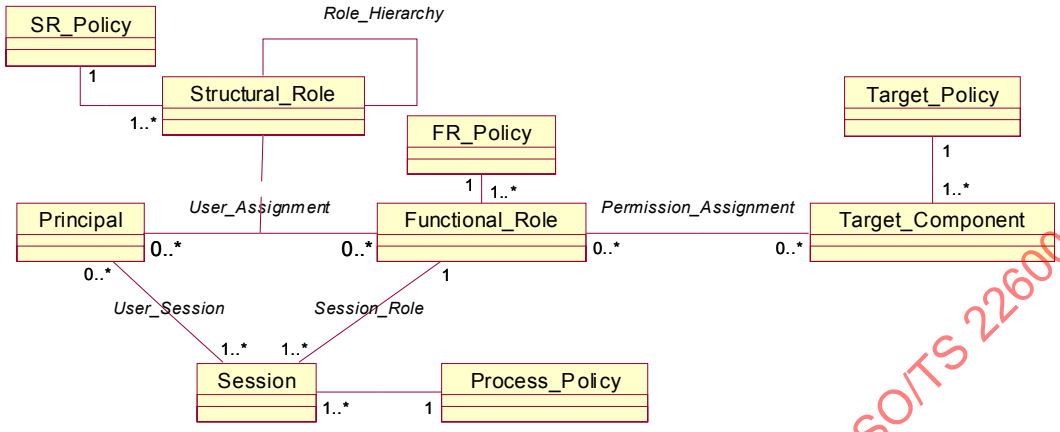

Figure 3 — Policy-driven RBAC schema

6 Interpretation of ISO/TS 22600-2 formal models in healthcare settings

The formal models introduced in ISO/TS 22600-2 are based on the Generic Component Model ([3, 4] in the Bibliography). The Generic Component Model describes an architectural approach consisting of formal representations of concepts and their relationships, both representing knowledge and derived from references using constraint modelling. To support the understanding of the models presented in this part of ISO/TS 22600, a short summary of the basic principles follows, combined with some refinements or profiles of the models introduced in ISO/TS 22600-2.

To understand, communicate and change the reality according to our social, environmental, organizational or technical business objectives, the reality must be observed, described and interpreted properly in a closed cycle using models ([5, 6, 7, 8] in the Bibliography). A model is a representation of something helpful in thinking about the real world, without having to deal with every detail of reality. A purpose of models is to create knowledge. An outcome of developing mathematical models is that it helps model builders and decision makers understand the relationships between important variables in a business situation. On the other hand, the description and especially the interpretation of real systems are based on knowledge. Knowledge is a combination of instincts, ideas, rules and procedures that guide actions and decisions. It is used to transform data into information that is useful in a situation. Knowledge helps users interpret and act on information. Building terms implies knowledge. Therefore, the classification of term sets deals with the ordering of knowledge. A model for the classification of terms consists of items and their instances (semantic concepts, terms), relationships between terms of a terminology and the classification for explicit presentation of that relation ([9] in the Bibliography).

7 Concept representation for health information systems

7.1 Overview

A concept is a formal model. It shall be uniquely identifiable, and accepted by experts and users, as well as independent. A concept as a knowledge component can be specialized and generalized as components can. It provides a coherent description of domain entities, which can be identified and independently used by domain users for recording information ([8] in the Bibliography). The sum of concepts is called “ontology”. An ontology provides the formalization of the domain knowledge. Knowledge representation is frequently

provided in two of many possible ways: through rules (production rules, if/then rules) or frames. Production rules focus on the logic of making inferences. Frames are object-oriented approaches that focus on objects' important characteristics. A frame consists of slots identifying attributes for the particular kind of entity. The data in a frame can be used to identify which aspects of a situation are pertinent, to organize the data, and to identify exceptions. Any of the slots in a frame could have default values or references to other frames. In other words, a concept is a coherent description of domain entities (components), which are separately identified and usable for recording information. Knowledge consists of concepts that can be composed or decomposed through generalization or specialization, respectively, representing relationships between them. The following are considered: analysis, conceptualization, design, implementation and maintenance of information systems in their combination of work practices, information, people, and information technologies organized to accomplish goals in an organization.

Like knowledge representation in the world of objects or components, concepts are also structured (organized) in slots (see frame definition). This has been done with all existing health concept representations such as Archetypes, Arden Syntax MLMs, OCL, but also with security policy representations using formal expression means, such as first order logics or predicate logics or formal languages such as SAML, XACML, etc.

7.2 Domain languages

For expressing and sharing knowledge, the underlying concepts must be expressed properly, deploying common languages, domain-specific languages, formal languages, and formal models. The terms and knowledge applied can be summarized in terminologies and ontologies. A meta-thesaurus defines the presentation of domain knowledge as shown in Table 1. Domain-specific concepts result from business requirements. Therefore, any development framework for advanced information systems has to start with processes and methods for developing requirements and enterprise architecture, i.e. the creation of a business model. A business model collection of related architectures or blueprints of, by and for domain experts is aimed toward capturing the business essence, but not the ICT perspective. In the next phase, a methodology has to be provided to develop, deploy, test, maintain and integrate applications. Providing the complete development framework, the Generic Component Model describes any business model through the Enterprise View. The model's three-dimensional architecture allows for knowledge representation by concepts and their relationships including generalization and specialization. Alternatively, Archetypes have been established for healthcare business concept modelling.

Table 1 — Knowledge presentation through a meta-thesaurus

Concepts Synonymous terms are clustered into a concept. Properties are attached to concepts, e.g. — unique identifier, — definition.
Relations Concepts are related to other concepts. Properties are attached to relations, e.g. — type of relationship, — source.

Different levels of concept and rule representation as different ways for knowledge representation provide different levels of interoperability. At the highest level of a sustainable architecture following the Generic Component Model and the formal models derived in ISO/TS 22600-2 in regard to privilege management and access control, autonomous semantic interoperability is provided. Depending on the expression means used for concept representation covering structural and functional information in a processable or manually interpretable way, the interoperability level moves down. While the first allows for policy negotiation, the latter is based on attribute assignments provided by administrators or other system users (e.g. patients). In that

context, the interrelationship of concept representation in different domains using different domain languages has to be managed. The following are briefly discussed: different policy presentation, assignment and implementation means.

7.3 OCL constraint modelling

OCL is a standard extension to UML that allows for querying model elements, constraining them (at modelling time) and defining query operations ([5, 11] in the Bibliography). It enables the integration in the architectural process. Defining business rules is a challenge to be met in the General Component Model paradigm. OCL is constraining models with specific behaviour. OCL expressions consist of three parts: package context (optional), expression context (mandatory), and one or more expressions. OCL constraints are organized in expressions as shown in Table 2.

Table 2 — OCL constraint modelling

package <packagePath>	package context
context <contextualInstanceName>:<modelElement>	expression context
<expressionType><expressonName>:	expression
<expressionBody>	
<expressionType><expressionName>:	expression
<expressionBody>	
…	
endpackage	

7.4 Other constraint representations

7.4.1 General

According to the Generic Component Model, healthcare and its supporting information systems deal with other domains beside medicine and biology. In that context, finance, technology, legislation and security, etc. have to be mentioned. Regarding the latter, legal and policy concepts have to be modelled. A policy covers all implications on health and health information systems, such as legal, social, organizational, psychological, functional and technical.

Managing the security policy may include some or all of the following steps: writing, reviewing, testing, approving, issuing, combining, analysing, modifying, withdrawing, retrieving and enforcing the policy. The complete policy applicable to a particular decision request may be composed of a number of individual rules or policies. For instance, in a personal privacy application, the subject of care of the personal information may define certain aspects of disclosure policy, whereas the enterprise that is the custodian of the information may define certain other aspects. In order to render an authorization decision, it must be possible to combine the two separate policies to form the single policy applicable to the request.

OASIS Security Assertion Markup Language (SAML) defines security services assigned to entities in a header-body-reference structure using XML. For formally modelling policies and ruling access control, eXtensible Access Control Markup Language (XACML) has been developed at OASIS with the XML meta-language.

7.4.2 eXtensible Access Control Markup Language

eXtensible Access Control Markup Language (XACML) defines three top-level policy elements: `<Rule>`, `<Policy>` and `<PolicySet>` ([12] in the Bibliography). The `<Rule>` element contains a Boolean expression that can be evaluated in isolation. However, authorization decisions (e.g. by a PDP) should not be performed without reference to the corresponding policy. XACML offers rule-combining algorithms allowing for,

for example, deny-overrides (ordered and unordered), permit-overrides (ordered and unordered), first- or only-one-applicable.

The `<Policy>` element contains a set of `<Rule>` elements and a specified procedure for combining the results of their evaluation. It serves as a basis for authorization decisions by a PDP.

The `<PolicySet>` element contains a set of `<Policy>` or other `<PolicySet>` elements and a specified procedure for combining the results of their evaluation. It is the standard means for combining separate ***policies*** into a single combined ***policy***.

The main components of a rule are: a target; an effect; and a condition. The target defines the set of: resources; subjects; actions; and environment.

Therefore, a policy comprises: the main components target; a rule-combining algorithm-identifier; a set of rules; and obligations.

The presented structure and function partially reflect the component architecture for policies according to ISO/TS 22600-2, simplifying the defined policy subcomponents, however. For details, please refer to the XACML definition ([12] in the Bibliography).

7.4.3 Web Services Description Language

Web Services Description Language is an XML grammar for describing web services, defining information regarding the interface for all publicly available functions, data types for all messages, bindings to the transport protocol used, as well as addresses to locate specified services. It represents a contract between the service requester and the service provider in a platform- and language-independent way.

7.4.4 Business Process Execution Language

Business Process Execution Language is an XML-based language for describing a business process in a distributed cooperating environment. It formalizes the components and exceptions to be managed. The BPEL4WS process model is layered on top of the service model defined by the Web Services Policy Language (WSPL).

7.4.5 WS-policy

WSPL provides a flexible and extensible XML grammar for expressing the capabilities, requirements and general characteristics of entities in a Web Services-based system, summarized as domain-specific Web Service Policy information. It defines a collection of policy alternatives, where each of them is a collection of policy assertions, such as authentication scheme, transport protocol selection, privacy policy, QoS characteristics, etc. Compatible policies have to agree on vocabulary and semantics. Web Services Policy Attachment (WS-PolicyAttachment) defines how to associate policies with the subjects to which they apply, using WSPL descriptors.

7.4.6 Web Services Policy Language

WSPL, as a strict subset of the OASIS eXtensible Access Control Markup Language (XACML) standard, supports policy bridging. The merged policies can be based on fine-grained attributes, such as time of day, cost or network. WSPL supports the Boolean operators exclusive-OR and AND. A PolicySet represents the policies of a particular service, while each policy containing a sequence of rules represents a single aspect of the service. The rules represent an acceptable set of attributes expressed by predicates. It supports the decision between sets of attributes regarding the acceptability.

7.4.7 Domain-Independent Web Services Policy Assertion Language

Domain-Independent Web Services Policy Assertion Language (DIPAL) concerns the deployment choices (policies) rather than the business logic and interfaces of the service, using domain-specific information.

7.4.8 Security Assertion Markup Language

Security Assertion Markup Language (SAML) provides a grammar for wrapping security and identity information and exchanging them across domain boundaries. It allows business entities to make assertions (claims) regarding the identity, attributes and entitlements of a subject (an entity that is often a human user) to other entities.

SAML protocols define the structure of request/response pairs to obtain assertions and manage identity for this purpose, establishing assertion query/request protocols, authentication request protocols, name/identifier management protocols, single login and single logout protocols.

SAML bindings define the ways in which the SAML protocols are used with the standard messaging and communication protocols to specify, for example, SAML SOAP binding, SAML PAOS (reverse SOAP) binding, HTTP redirect binding or HTTP post binding.

SAML profiles combine selected assertions, protocols and bindings, and define how to handle defined use-cases. Here, web single sign-on (SSO) profiles, enhanced client and proxy (ECP) profiles, single logout profiles, assertion query/request profiles, SAML and attribute profiles should be mentioned.

SAML Authentication Context defines types and strength of existing authentication methods (e.g. Internet Protocol Password, Kerberos, Public Key – X.509, Smartcard PKI, SSL-TLS-based client authentication).

SAML metadata enable SAML actors to define their preferences and configurations.

8 Consent

8.1 Overview

While a policy provides a statement about legal, organizational, social, functional, etc., implications to be met, thereby providing the rules for constraining related concepts of the other domain, a consent is a special policy agreed between entities inclusively expressing approval or acceptance of the statements contained. Therefore, in many jurisdictions, legislation requires that entities involved be informed in detail about the concerns and their consequences to provide a legally and ethically acceptable basis for such approval (informed consent). Consent shall be formulated according to the established policy model and expressed through a formal policy representation language.

8.2 Patient consent

For meeting the aforementioned requirements, a patient consent has to be defined in a technology-challenging way, expressing fine-grained and detailed concepts and relationships concerning data, functionalities and services. Thus, consent may cover relationships to all entities occurring in health, such as data and information, documents, functions, systems, persons, organizations, applications, etc., i.e. principals as well as their actions/products.

8.3 Patient consent management

Patient consent management has to meet all policy principles, including definition, negotiation, harmonization, assignment, revocation, etc.

9 Emergency access

Emergency has to be provided according to the needs meeting all basic principles, such as legal constraints, the need-to-know-principle, etc., fixed in the emergency access policy. An important rule for emergency access policies is the complete logging of all actions performed.

10 Refinement of the control model

The generality of this model makes the names of its parts appear somewhat abstract; however, with suitable interpretation, it can be applied to all the situations introduced and discussed in this part of ISO/TS 22600.

10.1 Use of push or pull

The control model uses a verifier that acquires access control information to make an access control decision. The claimant can provide the information (for example, in a token) along with the request to the verifier (push) or the verifier can get the required information from a trusted source (pull). In deciding whether to use a push or pull model, several factors should be considered; see 10.1.1 and 10.1.2.

10.1.1 Push

Tokens should have a short time to live.

Tokens shall be validated against an authentication service.

Token delivery should be encrypted.

Tokens should include a nonce or a unique key within the encrypted token.

10.1.2 Pull

Authentication services shall be accessible.

Authentication service ACI repository shall be accessible.

A trusted communication path to the ACI repository is required.

Tokens holding authentication information shall be sufficiently secure for the environment to guard against the possibility of replay attacks.

11 Refinement of the delegation model

Restrictions on the delegation may be established by the SoA, claimant or target and include the following.

a) Delegation level — For example, the claimant in the delegate role may not be allowed to further delegate privileges.

b) Delegation context — For example, delegate my “assigned-radiologist” privilege only for a given patient identity and only for a given set of X-ray images and only for a specified period of time.

c) Delegate set — For example, no restrictions on number of levels of delegation, but all delegates shall be from a specified set of claimants.

d) Reference restriction — The privileges to use an object under specified circumstances are passed as part of the object reference to the recipient. For example, in privilege delegation, the initiating principal's access control information (that is, its security attributes) may be delegated to further objects in the chain to give the recipient the privileges to act on the initiating principal's behalf under specified circumstances.

e) Improper delegation — Restrictions that prevent a delegate from assigning privileges to inappropriate delegates, for example, a clinician assigning privileges to order medications to administrative staff.

Annex A
(informative)

Privilege management infrastructure

A.1 Use of this annex in harmonization with ASTM E2595-07

Existing standards, including ANSI X9.45 [46], ISO/IEC 9594-8, IETF RFC 3280 X.509 [47], OASIS SPML, SAML, WS-* and XACML, define a number of mechanisms that can be used to construct a healthcare-specific PMI specification. This would include the following features.

Privileges needed to access a target are conveyed in a claimant's authorization credential. The claimant's authorization credential may be an authorization certificate compliant with ISO/IEC 9594-8 (a particular form of attribute certificate) or a policy set description compliant with XACML or other referenced authorization standards.

The sensitivity or other properties of the target being accessed may be held in a local database or in a signed data structure. This guide does not define a standard way to represent this information, since this is a local matter. It does provide guidance on how such information might be represented and manipulated using common mechanisms such as ASN.1 and XML. For a given target object, there may be multiple operations that may be performed; each such operation may have a different set of sensitivity attributes.

The privilege policy may be held centrally or locally, or may be conveyed as a signed data structure. Different operations on a target may be subject to different privilege policies. This guide defines several standard policies, and applications may define additional policies.

In the document authorization paradigm, co-signature requirements may be associated with a user or document, such that the signed document is considered to be authorized only if all necessary signatures are attached.

Users may delegate privileges to other users.

Users may be assigned to roles that convey permissions.

Some authorizations may be sufficiently dynamic that it is not feasible to place them in an enterprise authorization infrastructure (that is, the cost of maintenance is too high, given the short lifetime or rapid frequency of change of the privileges or constraints). Such authorizations may be kept in a local authorization server's database and accessed as environmental variables.

The remaining sections of this part of ISO/TS 22600 discuss mechanisms to convey privilege, sensitivity and policy information in a distributed PMI.

A.2 Privilege management infrastructure framework

A.2.1 PMI services

The privilege management infrastructure (PMI) framework establishes relationships between components of an abstract role system to the components of the underlying security infrastructure.

The control model of ISO/TS 22600-1 has been extended to the security-distributed or service-oriented architecture according to Figure A.1. The verifier is shown with its component PDP/PEP. In the security SOA, infrastructure security authentication and authorization is provided to applications as a service. Figure A.1 abstracts the access control models of ISO/IEC 10181-3 (the PDP/PEP terminology follows OASIS XACML).

If the external (to the target) verifier is removed, then we have the traditional control model. Various implementations are achieved by functional allocations of service between application-level and SOA levels. Access control information is used to inform the Verifier PDP of additional conditions affecting the decision.

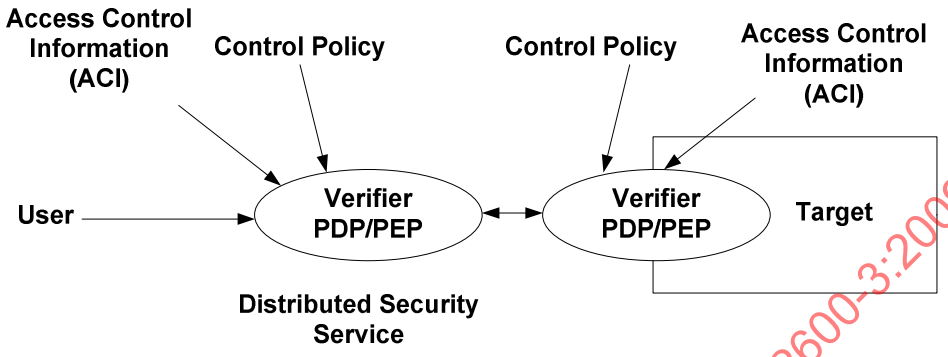

Figure A.1 — Extended control model

A.2.2 Access control information

ISO access control information (ACI) includes the following.

a) Initiator ACI, User ACI:

- individual access control identities;
- identifier of hierarchical group in which membership is asserted, for example, organizational position;
- identifier of functional group in which membership is asserted, for example, membership of a project or task group;
- role that may be taken;
- sensitivity markings to which access is allowed;
- integrity markings to which access is allowed;
- a target access control identity and the actions allowed on the target that is a capability;
- security attributes of delegates;
- location, for example, sign-on workstation.

b) Target ACI:

- target access control identities;
- individual initiator access control identities and the actions on the target allowed or denied them;
- hierarchical group membership access control identities and the actions on the target allowed or denied them;
- functional group membership access control identities and the actions on the target allowed or denied them;
- role access control identities and the actions on the target allowed or denied them;
- authorities and the actions authorized for them;

— sensitivity markings;

— integrity markings.

c) Action ACI:

1) ACI associated with operating zoning action (data ACI), for example:

— sensitivity markings,

— integrity markings,

— originator identity, and

— subject of care identity[1)].

2) ACI associated with the action as a whole, for example:

— initiator ACI.

3) Permitted initiator and target pairs:

— permitted targets,

— permitted initiators (claimants),

— allowed class of operations (for example, read, write), and

— required integrity level.

d) Contextual ACI:

— time periods;

— route (an access may be granted only if the route is being used to specific characteristics);

— location (an access may be granted only if two initiators are specified: as specific in-systems — workstations or terminals — or specific physical locations);

— system status (an access may be granted only for a particular ACI when the system has a particular status, for example during a disaster recovery);

— strength of authentication (an access may only be granted when authentication mechanisms of at least a given strength are used);

— other access currently active for this or other initiators.

There are two types of high-level healthcare role supported by the infrastructure: structural roles and functional roles. Structural roles reflect the structural aspects of relationships between entities. Structural roles describe prerequisites, feasibilities or competences for acts. Functional roles reflect functional aspects of relationships between entities. Functional roles are bound to the realization/performance of acts.

Possible examples for structural roles of healthcare professionals are

— medical director,

— director of clinic,

1) This is termed “Owner identity” in ASTM E2595-07.

— head of the department,

— senior physician,

— resident physician,

— physician,

— medical assistant,

— trainee,

— head nurse,

— nurse, and

— medical student.

Possible examples for functional roles of healthcare professionals are

— caring doctor (responsible doctor),

— member of diagnostic team,

— member of therapeutic team,

— consulting doctor,

— admitting doctor,

— family doctor, and

— function-specific nurse.

A detailed description of structural and functional roles is presented in the following subclauses.

A.2.3 Governance, risk and compliance

An effective security governance framework involves establishing chains of responsibility, authority and communication to empower people to control the system effectively. Governance is very important for the security services, as managing the security policy and implementation is vital to the integrity of the environment.

Governance in the service creation scenario involves monitoring compliance of the security services with the security policies, monitoring compliance with governance structures in place, and monitoring the overall security effectiveness of the environment.

Security compliance management measures the performance of the security implementation relative to the measures defined by the security policy. These can be realized based on reporting on the system behaviour using audit information and comparing that behaviour with configured policies in systems. When these are viewed in the context of business-defined policies, they can provide an overarching view of where a business stands in implementation and enforcement of intended policies.

A.2.4 Trust management

From a business viewpoint, trust management includes the liability and legal aspects around the services. It also includes protection messages that the service provider can implement for sensitive data.

At a technology level, trust management may include the following.

— The protocols for the service consumer to contact the service provider. For example, this may require a simple object access protocol (SOAP) message carried on HTTPS.

— The security token and its contents that need to be included in a WS-security message. For example, a SAML assertion carrying role-based information is required.

A.2.5 Identity management

The following enterprise identity management needs are identified:

— to provide an enterprise identity management service/capability to support the various business lines within the enterprise;

— to identify all persons of interest uniquely; this includes persons that have customer, contractual and/or employee roles;

— to facilitate the sharing of information between internal lines of business and with external partners.

A provisioning policy, for example, is defined to automatically create user accounts (an LDAP directory or a database, with an account identifier *joe* for Joe Smith) in the enterprise repository (account identifier *joesmith* for Joe Smith) and in the entity identification service (EIS) system (an application-level account with *joey1234* as the identifier for the same person Joe Smith).

This provisioning policy can be extended to cross the enterprise boundaries so that the user is also created in the registry used for the external service consumer (account identifier *homejoe* for Joe Smith). This provisioning policy may include several workflow activities, for example, getting user management approvals.

As part of these provisioning policies, identifier and password policies have to be taken into account. Identifier policies define how the different attributes for the different accounts are created, based on the user identity information and the company security rules.

In this example, an identifier policy may be defined to create the accounts on a system using the first letter of the first name and the letters of the family name for a user (for example, *jsmith* for Joey Smith). This can also apply to other attributes, such as an e-mail address.

Password policies can be used to enforce the way passwords for the different user accounts are created and managed. For example, it can be decided to define a policy requiring a minimum length, the inclusion of numeric and special characters, and an expiration date so that users need to change their passwords every three months. This enforces security, as password weakness is known to be a common risk for the systems.

Federation policies are another foundation layer of the scenario. They allow the validation, mapping and exchange of the different security tokens that are used between the domains or systems.

Enterprises shall provide a capability that provides access on a least-privilege basis, with a need-to-know for protected health information that is based upon users' roles in the organization and the tasks they are assigned.

Organizations therefore require an integrated approach to security that provides the infrastructure to meet the following needs:

— to consistently authenticate users across enterprise and extranet/federated boundaries;

— to consistently authorize/grant users permissions to protected information assets;

— to robustly enforce access by authenticated and authorized users to protected information assets;

— to audit access to and use of sensitive information and functions;

— to meet applicable guidelines and mandates for information security.

Users may need to use a user name and password to authenticate to their portal or a strong authentication (for example, *homejoe*, *jsmith* or *joe*).

An SAML assertion may be required at the service component level to authenticate a user accessing the service through an external consumer, while a user name token can be enough for an internal application. The user name token provided may be different from the one used to authenticate to the local portal (for example, *joesmith*). Finally, on the application, the user credential is validated and mapped to the local account identity.

In the case of the direct exposure scenario, the internal service consumers have to use the trust service to exchange the identity information they have regarding the user for a valid consumable credential. For example, the local identity *jsmith* or *joe* is exchanged for a credential generated for user *joey1234*. The application then validates the credential. The trusted security token service (STS) needs to be configured so that the security tokens are correctly mapped to ones suitable for the receiving entities in this scenario.

A.3 Privilege management infrastructure services expressed in ASN.1

A.3.1 X.509-based certificate specifications

A.3.1.1 General

The X.509 role-based PMI model uses the attribute certificates (ACs) described in ISO/IEC 9594-8. An example of the use of an X.509 role-based PMI is the Permis project ([13] in the Bibliography). ACs are issued by attribute authorities (AAs). The AC is bound to the identity using the holder field of an X.509 identity certificate. This coupling permits separate management of PMI and PKI ([13] in the Bibliography). Decoupling PMI and PKI allows sensitive access control information to remain private, while identity certificates can be managed by a third party. At least three types of ACs are used in an X.509 role-based PMI: role-specification ACs, role-assignment ACs, and policy ACs. All ACs are digitally signed by the AA and are therefore tamper resistant.

Role-specification ACs hold the permission assignments granted to each role. Role-assignment ACs hold the roles assigned to each identity. Policy ACs indicate the root of the PMI trust and contain a pointer to a policy file as an attribute value. ACs are typically stored in a lightweight directory access protocol (LDAP)-enabled directory service. The verifier finds all role-assignment ACs granted to a user and validates the digital signatures and that a certificate has not been revoked. The verifier also finds the role-assignment ACs for each identified user role. This process can be optimized in several ways while keeping the overall strategy.

The X.509 role-based PMI can use a policy language such as Ponder, Keynote or XACML. Use of a domain-wide authorization policy by the verifier provides a secure, centrally managed approach.

The local SoA for a security domain creates the domain-wide authorization policy. Privilege allocators (PAs) use the policies signed by the SoA, possibly from a different security domain, to generate and digitally authenticate policy ACs. The SoA or the AA uses the PA to sign and publish role-assignment ACs to an LDAP directory. The resulting role-assignment ACs are used as the verifier to make access control decisions.

The following implementable specifications, which are available in existing standards or are de facto standards, have been copied into the document for convenience, adapting the current practice in similar specifications. Therefore, they are expressed in ASN.1 notation.

This part of ISO/TS 22600 provides specifications developed independently of the formal models established in ISO/TS 22600-2.

A.3.1.2 Authentication certificates

The authentication certificate follows the X.509V3 specification:

```
Certificate    ::=  SIGNED SEQUENCE
    {
    version                  [0]  Version DEFAULT v1,
    serialNumber                  CertificateSerialNumber,
    signature                     AlgorithmIdentifier,
    issuer                        Name,
    validity                      Validity,
    subject                       Name,
    subjectPublicKeyInfo          SubjectPublicKeyInfo,
    issuerUniqueIdentifier   [1]  IMPLICIT UniqueIdentifier OPTIONAL,
    subjectUniqueIdentifier  [2]  IMPLICIT UniqueIdentifier OPTIONAL
    extensions               [3]  Extensions MANDATORY
    }
```

version is the version of the encoded certificate. The certificate version SHALL be v3.

There are several ways for binding key-related ID certificates to key-less attribute certificates: the monolithic approach, the autonomic approach, and the approach of chained signatures.

In the monolithic approach, the attribute certificate is part of the ID certificate.

In the autonomic approach, some relevant information in the ID certificate is referred to bind with the attribute certificate.

In the binding approach using chained signatures, the ID certification authority's signature is referred to bind with the attribute certificate. ISO 17090 fixed the first approach.

A.3.1.3 Attribute certificates

A.3.1.3.1 General

Claimant privileges are conveyed as attributes, in either a public key certificate (in the **subjectDirectoryAttributes** extension) or (more frequently) in an attribute certificate.

The syntax of an attribute certificate is specified in X.509:

```
AttributeCertificate ::= SIGNED {AttributeCertificateInfo}
AttributeCertificateInfo ::= SEQUENCE
    {
    version              AttCertVersion DEFAULT v1,
    holder               Holder,
    issuer               AttCertIssuer,
    signature            AlgorithmIdentifier,
    serialNumber             CertificateSerialNumber,
    attrCertValidityPeriod   AttCertValidityPeriod,
    attributes           SEQUENCE OF Attribute,
    issuerUniqueID           UniqueIdentifier OPTIONAL,
    extensions           Extensions  OPTIONAL
    }
AttCertVersion::= INTEGER {v1(0), v2(1) }
```

```
Holder   ::=  SEQUENCE
              {
              baseCertificateID      [0] IssuerSerial   OPTIONAL,
              -- the issuer and serial number of the holder's Public Key Certificate
              entityName         [1] GeneralNames     OPTIONAL,
              -- the name of the entity or role
              objectDigestInfo [2] ObjectDigestInfo  OPTIONAL
              -- if present, version must be v2
   --at least one of baseCertificateID, entityName or objectDigestInfo must be present--}
   ObjectDigestInfo ::= SEQUENCE
         {
         digestedObjectType ENUMERATED {
              publicKey    (0),
              publicKeyCert    (1),
              otherObjectTypes   (2) },
         otherObjectTypeID   OBJECT IDENTIFIER OPTIONAL,
         digestAlgorithm          AlgorithmIdentifier,
         objectDigest             BIT STRING }
   AttCertIssuer ::= CHOICE
    {
     v1Form GeneralNames, -- v1 or v2
     v2Form [0] V2Form  -- v2 only
    }
   V2Form ::= SEQUENCE
    {
     issuerName      GeneralNames OPTIONAL,
     baseCertificateID     [0]  IssuerSerial OPTIONAL,
     objectDigestInfo      [1]  ObjectDigestInfo OPTIONAL
    }
-- At least one component must be present
     ( WITH COMPONENTS { ..., issuerName PRESENT } |
      WITH COMPONENTS { ..., baseCertificateID PRESENT } |
      WITH COMPONENTS { ..., objectDigestInfo PRESENT } )
   IssuerSerial ::= SEQUENCE {
       issuer       GeneralNames,
       serial            CertificateSerialNumber,
       issuerUID  UniqueIdentifier OPTIONAL }
   CertificateSerialNumber ::= INTEGER
   UniqueIdentifier ::= BIT STRING
   Attribute ::= CLASS
    {
      &id                OBJECT IDENTIFIER UNIQUE,
      &singleValued BOOLEAN DEFAULT FALSE,
      &Syntax }
   Attribute ::= SEQUENCE
    {
    attrType      ATTRIBUTE.&id ({SupportedAttrs}),
    attrValues    ATTRIBUTE.&Syntax ({SupportedAttrs} {@attrType}) }
   AttCertValidityPeriod ::= SEQUENCE
    {
    notBefore     GeneralizedTime,
    notAfter      GeneralizedTime }
```

The components of the attribute certificate are used as follows.

The **version** number differentiates between different versions of the attribute certificate. If **objectDigestInfo** is present or if **issuer** is identified with **baseCertificateID**, **version** must be **v2**.

The **holder** field conveys the identity of the attribute certificate holder. In this part of ISO/TS 22600, use of the issuer name and serial number of a specific public key certificate is required; use of the general name(s) is optional; and use of the object digest is prohibited. There is a risk with the use of **GeneralNames** by itself to identify the certificate holder, in that there is insufficient binding of a name to a public key to enable the authentication process of the certificate holder's identity to be bound to the use of an attribute certificate. Also, some of the options in **GeneralNames** (e.g. **IPAddress**) are inappropriate for use in naming an attribute certificate holder which is a role rather than an individual entity. General name forms should be restricted to distinguished name, RFC 822 [48] (email) address, and (for role names) object identifiers.

The **issuer** field conveys the identity of the AA which issued the certificate. Use of the issuer name and serial number of a specific public key certificate is required, and use of the general name(s) is optional.

The **signature** identifies the cryptographic algorithm used to digitally sign the attribute certificate.

The **serialNumber** is the serial number that uniquely identifies the attribute certificate within the scope of its issuer.

The **attrCertValidityPeriod** field conveys the time period during which the attribute certificate is considered valid, expressed in **GeneralizedTime** format.

The **attributes** field contains the attributes associated with the certificate holder which are being certified (e.g. the privileges).

The **issuerUniqueID** may be used to identify the issuer of the attribute certificate in instances where the issuer name is not sufficient.

The **extensions** field allows addition of new fields to the attribute certificate. Standard extensions from ISO/IEC 9594-8 are described in Annex B.

A.3.1.3.2 Confidentiality of attribute certificates

In some applications, it may be desirable to protect the contents of attribute certificates from entities other than the certificate holder and the relying party (which uses the certificate). This can be done by using the certificate holder's public key certificate(s) to establish an authenticated, encrypted path to the relying party (e.g. using SSL). This path can then be used to send the attribute certificates confidentially.

A.3.1.3.3 Attribute certificate paths

Just as with public key certificates, there may be a requirement to convey an attribute certificate path (e.g. within an application protocol to assert privileges). The following ASN.1 data type can be used to represent an attribute certificate path:

```
AttributeCertificationPath ::= SEQUENCE
  {
    attributeCertificate        AttributeCertificate,
    acPath                      SEQUENCE OF ACPathData OPTIONAL }
ACPathData ::= SEQUENCE
    {
    certificate          [0] Certificate OPTIONAL,
    attributeCertificate    [1] AttributeCertificate OPTIONAL }
```

A.3.1.4 Role certificates

A user's attribute certificate may contain a reference to another attribute certificate which contains additional privileges. This provides an efficient mechanism for implementing privileged roles.

The following specifications are possible:

— any number of roles can be defined by any AA;

— the role itself and the members of a role can be defined and administered separately, by separate AAs;

— the privileges assigned to a given role may be placed into one or more attribute certificates;

— a member of a role may be assigned only a subset of the privileges associated with a role, if desired;

— role membership may be delegated;

— roles and membership may be assigned any suitable lifetime.

An entity is assigned an attribute certificate containing an attribute asserting that the entity occupies a certain role. That certificate may have an extension pointing to another attribute certificate which defines the role (i.e. this role certificate specifies the role of the certificate holder and contains a list of privileges assigned to that role). The issuer of the entity certificate may be independent of the issuer of the role certificate and these may be administered (e.g. expired, revoked, etc.) entirely separately.

Not all forms of **GeneralName** are appropriate for use as role names. The most useful choices are object identifiers and distinguished names.

A.3.1.5 Credentials

Credential is a prerequisite for the entitlement of, or the eligibility for, a role. Credentials are related to their environment (they are localized).

Credentials are typically matched by type (e.g. “physician”) or by type and issuer (e.g. “physician licensed in Virginia”).

```
Credential ::= SEQUENCE
    {
    credType          OBJECT IDENTIFIER,
    issuer          GeneralName OPTIONAL,
    identifier          UTF8String }
credentials ATTRIBUTE ::=
    {
    &id             id-credentials,
    &SEQUENCE OF Credential }
```

If the issuer name is absent, then the issuer name from the enclosing attribute or public key certificate is used. If the certificate issuer name is absent, the credential issuer name must be present. (Note that a certificate may explicitly have more than one credential, from more than one issuer, in order to minimize the number of AAs in a system.)

A.3.2 Clearance

The clearance attribute is compared with target security labels. This provides a coarse-grained authorization and access control mechanism.

```
Clearance ::= SEQUENCE
{
    policyId              OBJECT IDENTIFIER,
classList             ClassList DEFAULT {unclassified},
securityCategories    SET OF SecurityCategory OPTIONAL }
ClassList ::= BIT STRING
    {
    unmarked     (0),
    unclassified (1),
    restricted   (2),
    confidential (3),
    secret       (4),
    topSecret    (5)}
SECURITY-CATEGORY ::= TYPE-IDENTIFIER
SecurityCategory ::= SEQUENCE
    {
    type         [0]  SECURITY-CATEGORY.&id({Categories}),
    value    [1]  SECURITY-CATEGORY.&Syntax({Categories}{@type}) }
```

The security policy identifier identifies the semantics of the label (allowable classification levels and security categories). The policy identifier must be the same for both the clearance and the target's security label for a comparison to be possible. The class list includes a list of hierarchical classification levels. The levels in the clearance are compared against the level in a target's security label; at least one level in the clearance must be greater than or equal to the level in the label. The security categories contain other, non-hierarchical information. The value of a category in the clearance must "dominate" that in the corresponding category in the label; the meaning of "dominance" is specified when the category is defined. For example, in military applications, a label may contain a set of codewords or compartments. To dominate a label, a clearance must contain all of the codewords/compartments in the label (and possibly additional ones as well). More details on security labels can be found in A.3.4.3 and A.9.17.

A.3.3 Claimant mechanisms

Current approaches to management of user privileges include

a) centralized storage of privileges (e.g. in an LDAP directory), where they can easily be retrieved,

b) storage of privileges in the **subjectDirectoryAttributes** extension in a user's public key certificate, and

c) storage of privileges in attribute certificates.

The last two approaches represent the mechanism of storing user privileges in digitally signed credentials.

Option a) is easy to implement, but requires the privilege server to be continuously available. The other two options allow the authorization information to be conveyed in certificates, eliminating the need for an online server. There are several reasons why it may be preferable for a separate attribute authority (AA) to place privileges in attribute certificates rather than in a public key certificate. These include

d) different lifetimes for privileges versus public keys,

e) separation of duties (the CA is not the entity responsible for privilege management),

f) need-to-know (it may not be desirable for all entities to know all privileges of a claimant), and

g) the principle of least privilege (the user only receives/presents the minimum privileges needed to perform a particular operation).

A.3.4 Target sensitivity mechanisms

A.3.4.1 General

Management of target attributes (e.g. access control lists and security labels) has traditionally been done on a per-system basis, and there has been little standardization of the representation of this information. This part of ISO/TS 22600 does not dictate how such information is represented, but does give suggestions based on several document syntaxes (ASN.1 or XML).

A.3.4.2 Signed data encapsulation

Attributes and other sensitivity information may be bound to the digest of the target using the **SignedData** construct. In particular, the use of detached signatures (with the object conveyed separately from the signature structure) would be appropriate. Sensitivity information would be carried as signed attributes, with the originator of the information being the signer.

The types of authorization information which can be attached to a target include

— access control information (ACI), as described in ISO/IEC 10181-3,

— co-signature requirements, as described in A.9.19, and

— descriptive information about the document (e.g. document type), described in A.11.3.

A.3.4.3 Security labels

The security label syntax is taken from RFC 2634 [49].

```
ESSSecurityLabel ::= SET
      {
      security-policy-identifier SecurityPolicyIdentifier,
      security-classification    SecurityClassification OPTIONAL,
      privacy-mark               ESSPrivacyMark OPTIONAL,
      security-categories        SecurityCategories OPTIONAL }
id-aa-securityLabel OBJECT IDENTIFIER ::= { iso(1) member-body(2)
  us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 2}
SecurityPolicyIdentifier ::= OBJECT IDENTIFIER
SecurityClassification ::= INTEGER {
      unmarked (0),
      unclassified (1),
      restricted (2),
      confidential (3),
      secret (4),
      top-secret (5) } (0..ub-integer-options)
ub-integer-options INTEGER ::= 256
ESSPrivacyMark ::= CHOICE
      {
      pString      PrintableString SIZE (1..ub-privacy-mark-length),
      utf8String   UTF8String SIZE (1..MAX) }
ub-privacy-mark-length INTEGER ::= 128
SecurityCategories ::= SET SIZE (1..ub-security-categories) OF
   SecurityCategory
ub-security-categories INTEGER ::= 64
-- see A.3.2 for definition of SecurityCategory
```

A security policy is a set of criteria for the provision of security services. The security-policy-identifier is used to identify the security policy in force to which the security label relates. It indicates the semantics of the other security label components.

This specification defines the use of the Security Classification field exactly as it is specified in RFC 2634 [49] and ITU-T X.411 [50].

If present, a security classification may have one of a hierarchical list of values. The basic security-classification hierarchy is defined in this Recommendation, but the use of these values is defined by the security policy in force. Additional values of security classification, and their position in the hierarchy, may also be defined by a security policy as a local matter or by bilateral agreement. The basic security-classification hierarchy is, in ascending order: unmarked, unclassified, restricted, confidential, secret, top-secret.

This means that the security policy in force (identified by the security-policy-identifier) defines the **SecurityClassification** integer values and their meanings. An organization can develop its own security policy that defines the **SecurityClassification** INTEGER values and their meanings. However, the general interpretation is that the values of 0 through 5 are reserved for the “basic hierarchy” values of unmarked, unclassified, restricted, confidential secret, and top-secret.

There is no universal definition of the rules for using these “basic hierarchy” values. Each organization (or group of organizations) will define a security policy which documents how the “basic hierarchy” values are used (if at all) and how access control is enforced (if at all) within their domain. Therefore, the security-classification value MUST be accompanied by a security-policy-identifier value to define the rules for its use. For example, a company's “secret” classification may convey a different meaning from the US Government “secret” classification. In summary, a security policy SHOULD NOT use integers 0 through 5 for other than their X.411 meanings, and SHOULD instead use other values in a hierarchical fashion.

Note that the set of valid security-classification values SHALL be hierarchical, but these values do not necessarily need to be in ascending numerical order. Furthermore, the values do not need to be contiguous. For example, in the Defense Message System security policy, the security-classification value of 11 indicates sensitive-but-unclassified and 5 indicates top-secret. The hierarchy of sensitivity ranks top-secret as more sensitive than sensitive-but-unclassified, even though the numerical value of top-secret is less than sensitive-but-unclassified. (Of course, if security-classification values are both hierarchical and in ascending order, a casual reader of the security policy is more likely to understand it.)

An example of a security policy that does not use any of the X.411 values might be:

10 – anyone

15 – Morgan Corporation and its contractors

20 – Morgan Corporation employees

25 – Morgan Corporation board of directors

An example of a security policy that uses part of the X.411 hierarchy might be:

0 – unmarked

1 – unclassified, can be read by everyone

2 – restricted to Timberwolf Productions staff

6 – can only be read to Timberwolf Productions executives

A possible set of hierarchical security levels might be administrative (unclassified), clinical, sensitive (AIDS/mental health information), and may be anonymized.

If present, the privacy-mark is not used for access control. The content of the privacy-mark may be defined by the security policy in force (identified by the security-policy-identifier) which may define a list of values to be used. Alternatively, the value may be determined by the originator of the security label.

If present, the security categories provide further granularity for the sensitivity of the message. The security policy in force (identified by the security-policy-identifier) is used to indicate the syntaxes that are allowed to be present in the security categories. Alternatively, the security categories and their values may be defined by bilateral agreement.

There might be many labels for the same object with different evaluation schemes, e.g. requiring all permissions or only some (or even one) to be met for access permission.

A.3.5 Access control framework

This subclause defines an access control information (ACI) attribute which can be used to indicate to a recipient (or trusted third party) which entities may read the target's contents.

```
ACCESS-CONTROL-SCHEME ::= TYPE-IDENTIFIER

AccessControlScheme ::= INSTANCE OF ACCESS-CONTROL-SCHEME

astmScheme ACCESS-CONTROL-SCHEME ::=
    { ASTMScheme IDENTIFIED BY id-astmScheme }

ASTMScheme ::= SEQUENCE OF ASTMEntry

ASTMEntry ::= SEQUENCE
    {
    who                 Requester,
    constraints         Constraints OPTIONAL }

Requester ::= CHOICE
    {
    role                [0]  OBJECT IDENTIFIER,
    individual          [1]  GeneralName,          -- from X9.55
    group               [2]  GeneralName,
    organizationalUnit  [3]  GeneralName }

Constraints ::= SEQUENCE OF Constraint

CONSTRAINT ::= TYPE-IDENTIFIER

Constraint ::= INSTANCE OF CONSTRAINT

caseID CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-caseID }

encounterID CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-encounterID }

claimEventID CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-claimEventID }

planRegistration CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-planRegistration }

encounterRegistration                                    CONSTRAINT                    ::=
    { OCTET STRING IDENTIFIED BY id-encounterRegistration }

admission CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-admission }

diagnosis CONSTRAINT ::= { Code IDENTIFIED BY id-code }

disease CONSTRAINT ::= { Code IDENTIFIED BY id-disease }

disorder CONSTRAINT ::= { Code IDENTIFIED BY id-disorder }
```

department CONSTRAINT ::= { GeneralName IDENTIFIED BY id-department }

testID CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-testID }

resultID CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-resultID }

procedureID CONSTRAINT ::= { OCTET STRING IDENTIFIED BY id-procedureID }

specimenType CONSTRAINT ::= { Code IDENTIFIED BY id-specimenType }

shift CONSTRAINT ::= { INTEGER IDENTIFIED BY id-shift }

workgroup CONSTRAINT ::= { GeneralName IDENTIFIED BY id-workgroup }

training CONSTRAINT ::= { BOOLEAN IDENTIFIED BY id-disorder }

therapeuticAgent CONSTRAINT ::= { Code IDENTIFIED BY id-therapeuticAgent }

diagnosticAgent CONSTRAINT ::= { Code IDENTIFIED BY id-diagnosticAgent }

disidentified CONSTRAINT ::= { BOOLEAN IDENTIFIED BY id-disidentified }

```
Code ::= SEQUENCE
    {
    codeSet         PrintableString,
    codeValue       IA5String }
```

Access is allowed to the target if the requester matches an entry in the Who list (by name, role, group, or organizational unit) and if the requester's attribute certificate matches all of the constraints contained in the target's ACI attribute. These constraints would be contained in the constraints attribute in the requester's certificate. This attribute has syntax **constraints**.

A.4 Privilege management infrastructure services expressed in XML

A.4.1 XACML-based role assignment

The Organization for the Advancement of Structured Information Standards (OASIS) standards group developed eXtensible Access Control Markup Language (XACML) as a language to express and evaluate access decisions. The XACML technical specification includes a profile for RBAC using XACML that complies with the ANSI RBAC standard.

For the convenience of the reader, terms found in the NIST core RBAC document [14] are compared with the terms in the XACML profile and the terms in this guide in Table A.1.

Table A.1 — RBAC core functionality mapping

Core element	XACML profile	ASTM PMI
Users	XACML Subjects	Claimants
Roles	XACML Subject Attributes	Roles
Objects	XACML Resources	Objects
Operations	XACML Actions	Operations
Permissions	XACML Role <PolicySet and Permission <PolicySet>	Permissions

The XACML RBAC profile also supports hierarchical RBAC, allowing inheritance between roles. Additional XACML policies are provided to support system and review functions described in the ANSI RBAC standard. Specifically, the Role PolicySet (RPS) associates holders of a given role attribute with a Permission PolicySet. The Permission PolicySet (PPS) describes the permissions associated with a specific role. The RPS and PPS replace the role assignment and role specification ACs in the X.509-based role model.

The XACML role-based PMI features a rich and extensible policy language integrated throughout the design. The concept of structural versus functional roles is supported using a two-tiered system comprising role attributes. That is, users can have roles assigned to them in the request context. An entity separate from the policy decision point can use an XACML role assignment policy or PolicySet to enable attributes within the user session.

A.4.2 Further PMI services

In A.5, further PMI services expressed in XML are introduced by examples.

A.5 Examples for privilege management and access control scenarios and services

A.5.1 Organization-based privilege assignment

In the case of organization-based privilege assignment, the structural role defined according to ISO/TS 21298 reflecting the relationship between an individual and an organizational entity establishes certain privileges. The privilege is based on the membership of an organization. This is valid for any entity type, i.e. persons but also systems, components, devices. The assigned privileges may apply within the organization, but also in external organizations (for example, membership to authorities).

A.5.2 Workflow-based privilege assignment

Privileges can be launched in accordance with a workflow for enabling the realization of a process. The assignment may be established for the entire workflow, for certain activities or even for certain transactions. This approach follows the assignment according to the functional role of an entity according to ISO/TS 21298, reflecting the relationship between an individual entity and an act.

A.5.3 Policy-based privilege management

Privileges can be assigned according to constraints in time, environmental conditions, functions, and operations, expressed in legislation, regulation, specification, etc.

A.5.4 Role and permission assignment

Constraints defining privileges and duties are normally specified in policies. In the case of simple policies and interrelations, privileges and duties can be transferred into roles as done in simple RBAC implementations, e.g. provided within the HL7 RBAC standard ([15] in the Bibliography). Such a simplified approach is connected with the problem of the growing numbers of defined roles according to new policy definitions. Furthermore, essential aspects such as context and other variables (see ISO/TS 22600-2) cannot be considered properly. For more details see ISO/TS 21298.

A.5.5 Authorization specification

The attribute authority (AA) and certification authority (CA) are logically (and, in many cases, physically) completely independent. The creation and maintenance of “identity” can (and often should) be separated from the PMI. Thus the entire PKI, including CAs, may be existing and operational prior to the establishment of the PMI. The CA, although it is the source of authority for identity within its domain, is not automatically the source of authority for privilege. The CA, therefore, will not necessarily itself be an AA and, by logical implication, will not necessarily be responsible for the decision as to what other entities will be able to function as AAs (e.g. by including such a designation in their identity certificates).

The source of authority (SoA) is the entity that is trusted by a privilege verifier as the entity with ultimate responsibility for assignment of a set of privileges. A resource may limit the SoA authority by trusting certain SoAs for specific functions (e.g. one for read privileges and a different one for write privileges). An SoA is itself an AA as it issues certificates to other entities in which privileges are assigned to those entities. An SoA is analogous to a “root CA” or “trust anchor” in the PKI, in that a privilege verifier trusts certificates signed by the SoA. In some environments, there is a need for CAs to have tight control over the entities that can act as SoAs. This framework provides a mechanism for supporting that requirement. In other environments, that control is not needed and mechanisms for determining the entities that can act as SoAs in such environments may be outside the scope of this part of ISO/TS 22600.

This framework is flexible and can satisfy the requirements of many types of environments.

When attribute certificates point to public key certificates for their issuers and holders, the PKI is used to authenticate holders (privilege asserters) and verify the digital signatures of the issuers.

A.5.6 Certificate-based privilege management

The following strategies for managing privileges based on certificates are described.

A.5.6.1 Privilege in attribute certificates

Entities may acquire privileges in two ways.

— An AA may unilaterally assign privilege to an entity through the creation of an attribute certificate (perhaps totally on its own initiative, or at the request of some third party). This certificate may be stored in a publicly accessible repository and may subsequently be processed by one or more privilege verifiers to make an authorization decision. All of this may occur without the entity's knowledge or explicit action.

— Alternatively, an entity may request a privilege of some AA. Once created, this certificate may be returned (only) to the requesting entity, which explicitly supplies it when requesting access to some protected resource.

Note that, in both procedures, the AA must perform its due diligence to ensure that the entity should really be assigned this privilege. This may involve some out-of-band mechanisms, analogous to the certification of an identity/key-pair binding by a CA.

The attribute-certificate-based PMI is suitable in environments where any one of the following is true.

— A different entity is responsible for assigning a particular privilege to a holder than for issuing public key certificates to the same subject.

— There are a number of privilege attributes to be assigned to a holder, from a variety of authorities.

— The lifetime of a privilege differs from that of the holder's public key certificate validity (generally the lifetime of privileges is much shorter).

— The privilege is valid only during certain intervals of time which are asynchronous with that user's public key validity or validity of other privileges.

A.5.6.2 Privilege in public key certificates

In some environments, privileges are associated with the subject through the practices of a CA. Such privileges may be put directly into public key certificates (thereby re-using much of an already-established infrastructure), rather than issuing attribute certificates. In such cases, the privilege is included in the **subjectDirectoryAttributes** extension of the public key certificate.

This mechanism is suitable in environments where one or more of the following are true.

— The same physical entity is acting both as a CA and as an AA.

— The lifetime of the privilege is aligned with that of the public key included in the certificate.

— Delegation of privilege is not permitted.

— Delegation is permitted, but for any one delegation, all privileges in the certificate (in the **subjectDirectoryAttributes** extension) have the same delegation parameters and all extensions relevant to delegation apply equally to all privileges in the certificate.

A.5.7 Privilege management infrastructure

The object may be a resource being protected, for example in an access control application. The resource being protected is referred to as the object. This type of object has methods which may be invoked (for example, the object may be a firewall which has an “Allow Entry” object method, or the object may be a file in a file system which has Read, Write, and Execute object methods). Another type of object in this model may be an object that was signed in a non-repudiation application.

The privilege asserter is the entity that holds a particular privilege and asserts its privileges for a particular context of use.

The privilege verifier is the entity that makes the determination as to whether or not asserted privileges are sufficient for the given context of use.

The pass/fail determination made by the privilege verifier is dependent upon four things:

— privilege of the privilege asserter;

— privilege policy in place;

— current environment variables, if relevant;

— sensitivity of the object method, if relevant.

The privilege of a privilege holder reflects the degree of trust placed in that holder, by the certificate issuer, that the privilege holder will adhere to those aspects of policy which are not enforced by technical means. This privilege is encapsulated in the privilege holder's attribute certificate(s) (or **subjectDirectoryAttributes** extension of its public key certificate), which may be presented to the privilege verifier in the invocation request, or may be distributed by some other means, such as via the directory. Codifying privilege is done through the use of the **Attribute** construct, containing an **AttributeType** and a **SET OF AttributeValue**. Some attribute types used to specify privilege may have very simple syntax, such as a single **INTEGER** or an **OCTET STRING**. Others may have more complex syntaxes. An example is provided in Annex D.

The privilege policy specifies the degree of privilege which is considered sufficient for a given object method's sensitivity or context of use. The privilege policy must be protected for integrity and authenticity. A number of possibilities exist for conveying policy. At one extreme is the idea that policy is not really conveyed at all, but is simply defined and only ever kept locally in the privilege verifier's environment. At the other extreme is the idea that some policies are “universal” and should be conveyed to, and known by, every entity in the system. Between these extremes are many shades of variation. Schema components for storing privilege policy information in the directory are defined in this part of ISO/TS 22600.

Privilege policy specifies the threshold for acceptance for a given set of privileges. That is, it defines precisely when a privilege verifier should conclude that a presented set of privileges is “sufficient” in order that it may grant access (to the requested object, resource, application, etc.) to the privilege asserter.

Syntax for the definition of privilege policy is not standardized in this part of ISO/TS 22600. Annex D contains a couple of examples of syntaxes that could be used for this purpose. However, these are examples only. Any syntax may be used for this purpose, including clear text. Regardless of the syntax used to define the privilege policy, each instance of privilege policy must be uniquely identified. Object identifiers are used for this purpose.

PrivilegePolicy ::= OBJECT IDENTIFIER

The environment variables, if relevant, capture those aspects of policy required for the pass/fail determination (e.g. time of day or current account balance) which are available through some local means to the privilege verifier. Representation of environment variables is entirely a local matter.

The object method sensitivity, if relevant, may reflect attributes of the document or request to be processed, such as the monetary value of a funds transfer that it purports to authorize, or the confidentiality of a document's content. The object method's sensitivity may be explicitly encoded in an associated security label or in an attribute certificate held by the object method, or it may be implicitly encapsulated in the structure and contents of the associated data object. It may be encoded in one of a number of different ways. For instance, it may be encoded outside the scope of PMI in the X.411 label associated with a document, in the fields of an EDIFACT interchange, or hard-coded in the privilege verifier's application. Alternatively, it may be done within the PMI, in an attribute certificate associated with the object method. For some contexts of use, no object method sensitivity is used.

There is not necessarily any binding relationship between a privilege verifier and any particular AA. Just as privilege holders may have attribute certificates issued to them by many different AAs, privilege verifiers may accept certificates issued by numerous AAs, which need not be hierarchically related to one another, to grant access to a particular resource.

The attribute certificate framework can be used to manage privileges of various types and for a number of purposes. The terms used in this specification, such as privilege asserter, privilege verifier, etc., are independent of the particular application or use.

A.5.8 Permission constraints

Permission constraints are recorded as part of the role-engineering process. For example, a constrained permission occurs when only one role is allowed to perform a particular task at any given time. A constraint occurs for a permission when its definition is tied to cardinality. In this case, the constraint on the permission would be the cardinality specification. Examples of permission constraints include

— Head Nurse on a hospital floor (cardinality of 1),

— Chief of Staff (cardinality of 1),

— Laboratory Technician versus Laboratory Technician Supervisor (separation of duties),

— provider's access to a remote hospital which is not his/her primary workplace (location), and

— shift-dependent access (time-dependency).

The discussion above relies on the fact that the functional role name is arbitrary and only has meaning as a collection of ANSI INCITS compliant permissions. Furthermore, the defined roles are necessarily specific to the organization defining them and hence do not by themselves provide any interoperability with business partners, other than through a business partner agreement. To achieve true interoperability, organizations will require standardization of the healthcare permissions and a means to advertise and assert standard privileges.

A.5.9 Separating security from business logic

In designing a privilege management infrastructure framework, care must be taken to distinguish between the enforcement of security policy and business logic. The use of the security framework to enforce business logic tends to compromise system security, leads to poor design, and does not support clear security roles. It is

therefore advantageous to separate and isolate security and business application codes, whenever possible. This separation is essential for security engineers to obtain a clear definition of security boundaries, ensure that the security code is not affected by changes in business logic, and in support certification.

Separating security from business logic can be difficult. In general, access to a resource within business logic deals with interaction upon a set of resources that the user has the authority to access. Alternatively, security logic can be identified using the criteria below.

Considerations for use:

— involves a change of security state (e.g. confidential to non-confidential);

— involves confidentiality, integrity or availability of data;

— involves the concepts of least privilege, need-to-know, or separation of duties;

— involves regulatory requirements.

A.5.10 Policy enforcement point guidelines

The design of a privilege management infrastructure can influence placement of the policy enforcement point. In a distributed privilege management infrastructure, a PEP can be placed at the application level or at the enterprise level. The following factors are important in deciding the proper placement:

— availability of the PEP;

— reduction of the number of PEPs;

— ability to centrally manage;

— PEP lifecycle maintenance;

— physical security of the PEP;

— the PEP should be on the network;

— the PEP should be located as close to the application as possible.

Co-location of the PEP with applications will decrease latency. In these cases, a local policy store should be available to allow the policy engine to work during extranet outages.

A.6 LDAP-enabled directory service versus policy engines

A.6.1 General

The choice of claimant mechanism can have an effect on performance and complexity of the PMI framework. LDAP-based mechanisms are typically fast but provide simple data, for example, whether the claimant is a member of a specific role. Alternatively, policy engines can factor several independent elements into the authorization decision process. Considerations are listed in A.6.2 to A.6.4.

A.6.2 LDAP

Use of an LDAP-enabled directory service is suggested when speed is required and the decision can be based on possession of a certain role, especially for structural roles.

Considerations for use:

— data retrieved consists of single strings;

— there is a sensitivity to processing overhead;

— there is a need for documented interface;

— there is no need for evaluating policies.

A.6.3 Policy engine

Policy engines offer flexibility but reduced speed compared with LDAP lookup when implemented in general-purpose software engines. Special-purpose hardware-based policy engines, on the other hand, offer advantages over LDAP when evaluating complex rules, constraints, and combining rules from multiple policy points. Note that caching of decisions (within their validity period) from a policy engine keyed by a hash of the request attributes can improve performance significantly in an environment in which many accesses are being made in a given period of time and many of these involve the same request attributes.

Considerations for use:

— data retrieved consists of complex or multiple elements;

— requirement for the evaluation of multiple elements;

— need to support complex policy languages (for example, XACML);

— ability to tolerate slower response time.

Use of an LDAP-enabled directory service is suggested when speed is required and the decision can be based on possession of a certain role, especially structural roles.

Policy engines are appropriate when evaluating participation in a workflow, especially in functional roles (A.11.1). The privilege management infrastructure framework may also use a combination of LDAP and policy engine ([16] in the Bibliography).

A.6.4 Policy decision point (PDP)

Policy decision points (PDPs) can be implemented in many ways depending on the model appropriate to the infrastructure (see SAML 2.0 profile of XACML). The PDP is responsible for deciding if an access control request should be allowed or denied. The decision is based on policies available to the PDP from one or more policy stores. Additional ACI can be used by the PDP in making the access control decision. The decision is returned to the requester directly to the PEP or through an intermediary such as a context handler.

Considerations for use:

— place PDP such that it can be centrally managed;

— use XACML to convey access control requests and decisions;

— consider the security of the PDP and inter-process communications;

— consider if the PDP is located on the application (local decisions) or network (global decisions) or both;

— consider mechanisms to provide assurance of the decision provided to the PEP;

— place PDP on hardware-assisted accelerator to improve performance.

Many design options are possible that increase security and flexibility of the PDP support infrastructure ([17] in the Bibliography).

A.7 Identity management systems

Several functional areas within the PMI require a secure identity management subsystem (IdM). IdM is responsible for securely providing identity information and identity management functions. Identity management functions include:

— administration functions (user provisioning, password management);

— identity data control functions (metadata, identity content);

— access (authenticate requests, confidentiality);

— lifecycle management (configuration, patches, disaster recovery);

— backup, audit, logging, and reporting functions.

IdM shall provide a secure access mechanism for the request and delivery of identity information, typically using mutual authentication. Access to the IdM can be viewed as a service accessible throughout an enterprise. Organizationally, the IdM subsystem can be deployed in various ways:

— authoritative source integral to PMI security framework;

— an administrative function available to the PMI as needed;

— integral to a special-purpose IdM product solution.

Having an IdM as a service provides several advantages over a local special-purpose IdM. By their nature, IdM services provide a network interface using standard protocols that provide flexibility in changes to enterprise architecture. An IdM service can be centrally managed, allowing consistent enforcement of security and business policies.

A.8 Audit

The major purpose of the audit subsystem is to provide accountability of actions taken by agents on the network. Audit is not instrumental in the use of privileges to allow or disallow access to protected resources. However, the audit subsystem should interface with the PMI so that its correct operation can be verified. For a discussion of the use of audit in healthcare applications, see RFC 3881 [51].

An audit supports the SOA architecture and provides assurance for authentication and authorization services.

Accountability is the concept that individual persons or entities can be held responsible for specified actions, such as obtaining informed consent or breaching confidentiality ([18] in the Bibliography). Accountability is achieved through the implementation of a pervasive technical audit service. An audit provides a record of potential insecurities irrefutably traceable back to the originator of the action. A security audit provides not only accountability, but a means to assess damage done to a system by malicious action or accident. A security audit generated by the actions of other security services provides a check on their proper operation. In a distributed system, centralized audit collection and processing also provides a method to obtain near-real-time misuse detection and alerts. To be effective, security audit must be on.

A security audit trail provides a journal of security-related events collected for potential use in intrusion detection or security audits or both. An audit is a pervasive function of the healthcare system providing essential accountability features. An audit also provides assurance of the correct operation of the system's security features by monitoring user and system access to data and resources. An audit is generated as a by-product of the security controls in place, i.e. authentication, access and authorization (privileging), and upon occurrence of specific security-relevant events (for example, modifying a file). An audit acts as a deterrent to (unauthorized) user activities and, as such, users should know that their actions are being

monitored (usually part of a log-on banner). An audit also provides a means to assess the degree of harm caused should a break-in occur.

In a distributed architecture involving diverse commercial off-the-shelf (COTS) products, each product produces audit trails in a proprietary format. Even the events recorded may be different from product to product (for example, use of “grant” option makes sense in a database but not in an operating system). System audit trails may be character-based or binary. COTS audit trails often require specialized audit tools for review and processing. Audit trails may be stored in the file system or in database tables and so forth. Audit-analysing systems shall be able to harmonize and account for these differences.

In distributed systems, audit is produced at multiple locations on multiple components, making review and analysis difficult. Accordingly, in such a system, it is very desirable to consolidate and forward low-level audit from various audit-producing sources to a central audit server. There the audit can be reformatted to a single-composite format and automatically processed by a tool. Several such COTS tools are available, providing for a distributed audit capability for collecting, forwarding, processing and reporting audit events originating from diverse sources. Since the amount of audit produced may be considerable, a single centralized audit server is a practical way to manage workflow without affecting the response time of operational systems. Audit processing may be both real time and batch.

An automated audit tool provides the means of identifying events at different levels of security, performing automatic profiling, reporting and alerting, and a facility to store, sort and search for potential insecurities. Automated tools manage audit collection across host- and network-based audit systems. The placement of the audit tool, agents and components (including real-time network monitoring and intrusion detection) is considered to maximize the effectiveness of the audit system.

The Department of Health and Human Services' (DHHS) final rule for security and electronic signature standards identifies audit requirements, including alarms and event reporting (45CFR Parts 160, 162, and 164).

Products built or procured for Veterans' Health Administration use must incorporate audit controls or else must be integrated within the existing audit framework.

Pre-configured reports are prepared, based upon selected criteria to document security-critical events and to provide reports, graphs and statistical summaries of system security activity. A system for maintaining audit records may be a file or database. To maintain records for all users, the system should have the capability to select “all users” as a configurable portion of the event.

Audit records are reviewed by examination of the audit trail. Consolidation of audit records, when more than one source is involved at a central “audit server”, facilitates review by providing an automated means to examine the (typically) large amount of audit generated from these events. Continuous monitoring of audit records should be a part of the operation phase of the system development life cycle ([19] in the Bibliography).

The security architecture should support the establishment of auditing capabilities on an application, facility or national basis. To meet the requirement for a persistent retention capability, the audit function will include long-term archival and storage facilities. This requirement specifies the minimum length of time (five years) for which the archive shall be retained. Organizations should establish policies and procedures for log management consistent with accepted standards ([20] in the Bibliography).

Patient consent can act as the trigger of this audit record. Collection of disclosures made under this requirement requires that the audit configuration for this event be “mandatory”. The security architecture supports the centralized collection, processing, and reporting of disclosures of patient information. Storage of events recording certain disclosure under the provisions of jurisdictional privacy legislation may require a longer period of storage than simple security audit.

Intrusion-detection systems should be an integral part of the distributed architecture. Commercially available intrusion-detection systems provide alarm capabilities to permit rapid notification of specified intrusions. Intrusions can be categorized into two main classes, misuse and anomaly intrusions. Misuse intrusions are well-defined attacks on known weak points of a system. They can be detected by watching for certain actions being performed on certain objects. Anomaly intrusions are based on observations of deviations from normal

system usage patterns. They are detected by building up a profile of the system being monitored and detecting significant deviations from this profile.

Adherence to industry standards facilitates a robust audit subsystem. Industry standard profiling groups, such as Integrating the Healthcare Enterprise (IHE), publish profiles that describe how to use established standards to share healthcare information better in the clinical setting. IHE has published several integration profiles addressing security and privacy considerations, including the Audit Trail and Node Authentication (ATNA), Basic Patient Privacy Consents (BPPC), Document Digital Signature (DSG), Enterprise User Authentication (EUA), and Cross-enterprise User Assertion (XUA). These integration profiles describe security measures that, together with the security policy and procedures, provide patient information confidentiality, data integrity, and user accountability ([21] in the Bibliography). The IHE ATNA profile provides for a secure audit trail and is consistent with DICOM Supplement 95: Audit Trail Messages ([22] in the Bibliography).

A.9 Additional PMI services

A.9.1 Audit allocation

An audit is allocated to end systems: workstations, information servers, gateways and relay systems including security servers (domain controllers, proxy servers, etc.). A host-based audit monitors activities involving data. A network-based audit, such as at a gateway, records information on packets received. An audit is also allocated to security management activities for configuring, processing and reporting audit information and may provide basic data for processing by an intrusion detection system.

A.9.2 Intrusion detection

Intrusions can be categorized into two main classes, misuse and anomaly intrusions. Misuse intrusions are well-defined attacks on known weak points of a system. They can be detected by watching for certain actions being performed on certain objects. Anomaly intrusions are based on observations of deviations from normal system usage patterns. They are detected by building up a profile of the system being monitored and detecting significant deviations from this profile.

An Intrusion Detection System (IDS) may also perform its own system monitoring. It may keep aggregate statistics that give a system-usage profile. These statistics can be derived from a variety of sources, such as CPU usage, disk I/O, memory usage, activities by users, number of attempted logins, etc. These statistics must be continually updated to reflect the current system state. They are correlated with an internal model that will allow the IDS to determine if a series of actions constitutes a potential intrusion. This model may describe a set of intrusion scenarios or possibly encode the profile of a clean system.

A host-based intrusion-detection system is software that monitors a system or application's log files. It responds with an alarm or a countermeasure when a user attempts to gain access to unauthorized data, files or services.

A network-based intrusion-detection system monitors network traffic and responds with an alarm when it identifies a traffic pattern that it deems to be either a scanning attempt or a denial of service or other attack.

Network-based intrusion-detection systems need to be placed at security control points such as security gateways. Host-based intrusion detection systems require intrusion-detection software installed on the protected host. The architecture allocates intrusion detection to relay systems and end-systems.

A.9.3 Claimant mechanisms

Possible approaches to management of user privileges in a service-oriented PMI architecture include

— storage of privileges in a digitally signed credential,

— centralized storage of privileges (for example, in an LDAP-enabled service), and

— assertions by an AA.

Storage of privileges in a digitally signed certificate can be based on a public key certificate, an attribute certificate or an XACML attribute. The advantages to each approach are summarized in ISO/IEC 9594-8. Public key certificates may store privileges as noncritical extensions as described in ASTM E2212. The use of an identity certificate in this way tightly binds authentication to authorization and arguably provides resilience to network failure. However, there are several disadvantages to this approach.

Invalidating or changing any privilege owned by the certificate holder would require revoking and reissuing the certificate. Revocation of the certificate typically involves listing the certificate identification number on a certificate revocation list (CRL). As a result, the identity holder is unable to use their card (or other token) until it is reissued. Since identity certificates are typically in the possession of the holder, the reissuing process is unwieldy. Note that use of the identity certificate to store privileges does not provide resilience to network failure, since the CRL must be consulted to validate the certificate. Accordingly, identity certificates are more appropriate for the slowly changing “structural” roles rather than the highly dynamic “functional” roles.

One alternative is the use of a separate digitally signed certificate, called an attribute certificate, designed to hold user privileges. Additional infrastructure is required to issue and manage this second type of digitally signed certificate. However, there are several advantages to this approach.

The attribute certificate is issued and signed by an attribute authority separate from the certificate authority that manages identity certificates. However, authorization and authentication are still tightly bound by placing the user's identity certificate serial number in the holder field of the attribute certificate. Revocation of the attribute can be accomplished using an attribute certificate revocation list (ACRL). However, there is no reason why a user should take physical possession of an attribute certificate. Thus, the attribute certificates themselves can be stored in an LDAP-enabled service. Revocation of the certificate, therefore, involves replacement of the earlier attribute certificate with the current attribute certificate. There is no penalty in frequent changes of privileges held by the attribute certificate seen in the previous mechanism.

There is an additional benefit to the additional infrastructure required to support attribute certificates as pointed out in [23] in the Bibliography. Certificate authorities issuing identity certificates are typically managed by a trusted entity outside the enterprise. Adding infrastructure within the enterprise to support attribute certificates helps keep sensitive privilege information from outsiders. Attribute certificates are typically only needed by the verifier, so there is no need to expose the privileges held by a user to entities outside the enterprise.

A.9.4 LDAP lookup (role lookup)

Role lookup can be supported by keeping role information in an LDAP structure. LDAP provides a fast search and read functionality required in quickly collecting role information required in determining privileges associated with a user. The use of LDAP for role lookup is typically supported by web application servers. Alternatively, LDAP can be called from applications running independently of application servers. The preferred approach is to deploy a PDP that performs the role lookup in responding to an access control request. In any case, LDAP role lookup is beneficial because it provides a mechanism that separates the role information from the application code requesting the information.

Role information can be provided from an assertion (for example, an SAML assertion) reducing the need for LDAP lookup.

A.9.5 Certificates

A.9.5.1 Overview

As discussed above, certificates provide assurance by binding an identity to a certificate through the use of a cryptographic key. Certificates can be used by services to establish trust relationships throughput the PMI. Trusted certificates can be used to provide assurance over assertions made by trusted services of identity or privilege to a relying party. Identity certificates can provide role information. However, this information should be static in nature (that is, structural roles) because changing role information in an identity certificate necessitates cancellation and re-issuance of the certificate. Attribute certificates offer a better way to pass non-static privilege and role (that is, functional role) information. Certificates also provide means for confidentiality and non-repudiation.

A.9.5.2 Attribute certificates

A user's attribute certificate may contain a reference to another attribute certificate which contains additional privileges. This provides an efficient mechanism for implementing privileged roles.

Many environments which have authorization requirements require the use of role-based privileges (typically in conjunction with identity-based privileges) for some aspect of their operation. Thus, a claimant may present something to the verifier, demonstrating only that the claimant has a particular role (for example, “licensed healthcare provider” or “file clerk”). The verifier may know *a priori*, or may have to discover, by some other means, the privileges associated with the asserted role in order to make a pass/fail authorization decision.

Considerations for use:

— lack of reliable communication system;

— desire not to confirm information with issuing authority;

— slow changing or relatively static roles;

— appropriateness of use to support functional role.

The following are all possible.

— Any number of roles can be defined by any AA.

— The role itself and the members of a role can be defined and administered separately by separate AAs.

— The privileges assigned to a given role may be placed into one or more attribute certificates.

— A member of a role may be assigned only a subset of the privileges associated with a role, if desired.

— Role membership may be delegated.

— Roles and membership may be assigned any suitable lifetime.

An entity is assigned an attribute certificate containing an attribute asserting that the entity occupies a certain role. That certificate may have an extension pointing to another attribute certificate which defines the role (that is, this role certificate specifies the role of the certificate holder and contains a list of privileges assigned to that role). The issuer of the attribute certificate may be independent of the issuer of the role certificate and these may be administered (for example, expired, revoked, etc.) entirely separately.

Not all forms of **GeneralName** are appropriate for use as role names. The most useful choices are object identifiers and distinguished names.

A.9.6 Medical credentials

One common type of privilege is the user credential. These credentials are issued by a trusted authority and include an identification string. Examples include licensing of medical professionals by state boards and assignment of Drug Enforcement Agency (DEA) numbers. A credential includes a type, an issuer name, and an identifier. Geographically structured issuer names can be useful to indicate the state and other locality information. Credentials are typically matched by type (for example, “physician”) or type and issuer (for example, “physician licensed in Virginia”).

If the credential issuer name is absent, then the issuer name from the enclosing attribute or public key certificate is used. If the certificate issuer name is absent, the credential issuer name must be present. (Note that a certificate may explicitly reflect more than one credential, from more than one issuer, to minimize the number of attribute certificate authorities (AAs) in a system.)

Considerations for use follow.

— Consider the use of ASTM E2212 descriptions of the use of noncritical X.509 fields to describe a user's medical credentials in an identity certificate.

— Consider the use of medical credentials (current/non-current, location of applicability) as an additional constraint on granting authorizations to clinicians to healthcare information.

— Alternatively, clinician medical credentials could be considered for inclusion as part of the security provisioning of user attributes in a privilege management infrastructure.

A.9.7 SAML assertions

SAML assertions can be used to transmit security information from an asserting party to a relying party. The assertion can be made on behalf of a subject during authentication, or in response to a request from another SAML entity. Assertions can be constructed from privilege or role information stored either in a central store or from signed certificate information.

Considerations for use:

— flexibility in a federated environment;

— acquire additional information on the claimant;

— consideration of network availability and reliability;

— existence of SAML service;

— support for intra-enterprise assertions;

— compatibility with SOAP and WS security;

— whether one needs simultaneous support for variety of authentication mechanisms (for example, PKI Credentials, Kerberos tokens, biometrics, etc.).

A.9.8 Target sensitivity mechanisms

Management of target attributes (for example, access control lists and sensitivity labels) has traditionally been done on a per-system basis, and there has been little standardization of the representation of this information. This guide does not dictate how such information is represented, but it does give suggestions based on several document syntaxes (ASN.1 or XML).

A.9.9 Signed data encapsulation

Attributes and other sensitivity information may be bound to the digest of the target using the **SignedData** construct. In particular, the use of detached signatures (with the object conveyed separately from the signature structure) would be appropriate. Sensitivity information would be carried as signed attributes, with the originator of the information being the signer.

The types of authorization information that can be attached to a target include:

a) access control information (ACI), as described in ISO/IEC 10181-3;

b) co-signature requirements, as described in A.9.19;

c) descriptive information about the document (for example, document type), described in A.11.3.

A.9.10 Use of XML

eXtensible Markup Language (XML) provides a software- and hardware-independent tool for transmitting information. It is expected that many documents will be expressed using XML. The structure for such a document is defined in a document-type definition (DTD) or an XML schema.

Privileges may be defined as XML elements in which the name of the element represents the privilege identifier. Alternatively, an XML element can be associated with one or more XML attributes that represent the privilege identifier(s).

Privileges can be grouped into useful sets using XML. For example, a set of privileges encoded into XML can be associated with a uniform resource identifier (URI) such as:

“urn:application_name:attribute:privilege_set_name”

Alternatively, privilege sets can be associated within a schema definition addressed by a unique namespace such as:

xmlns: privilege_set_name=“http://www.astm.org/:privilege_sets/ privilege_set_name/”

XML privilege sets can be used by the verifier to associate the claimant to its scope of authority. Claimants can be associated with standard groups or roles before evaluation by the verifier. Alternatively, the verifier can associate the claimant to one or more privilege sets through a database or an LDAP-enabled directory service. The verifier can also look up the requester group or role association in an external XML document using XPath.

A verifier using a privilege policy may act directly on the XML elements (for example, by comparing attributes in an authorization certificate to elements in the document). One example of an XML-based policy that can be used to verify privileges is eXtensible Access Control Markup Language (XACML). Subclauses A.9.11 to A.9.19 discuss the comparison rules in detail. Generally, single-valued attributes will be compared with a single (complete) element, while multi-valued attributes will be compared with a collection of elements in a model group.

A.9.11 Access control framework

This subclause defines an access control information (ACI) attribute which can be used to indicate to a recipient (or trusted third party) which entities may read the target's contents.

Access is allowed to the target if the requester matches an entry in the *who* list (by name, role, group or organizational unit) and if the requester's attribute certificate matches all of the constraints contained in the target's ACI attribute. The constraints associated with the requester's attribute certificate would be contained in the **constraints** attribute in the requester's certificate.

A.9.12 Policy specification mechanisms

Although this guide does not dictate how privilege policies are represented within an end system, XACML provides a standard approach to authorization policies. Several scenarios are evident.

Two entities may need to determine whether their authorization policies are compatible, especially in a web services environment. If their policies are compatible, the entities need to determine specific policy variable values that are acceptable to both. In this case, an XACMLAuthzAssertion, defined in the "XACML Profile for Web Services (WS-XACML)" may be used. Such an assertion may be included in a WS-policy instance or provided as independent metadata.

An XACMLAuthzAssertion may also be used in a web services environment by a service provider to publish an authorization policy for retrieval by potential clients. Publishing authorization policies is not appropriate for all environments, but publishing certain aspects of an authorization policy may be useful even where publication of an entire policy would be a security problem.

Managing a security policy may include some or all of the following steps: writing, reviewing, testing, approving, issuing, combining, analysing, modifying, withdrawing, retrieving and enforcing policy.

The complete ***policy*** applicable to a particular ***decision request*** may be composed of a number of individual ***rules*** or ***policies***. For instance, in a personal privacy application, the subject of care of the personal information may define certain aspects of disclosure ***policy***, whereas the enterprise that is the custodian of the information may define certain other aspects. In order to render an ***authorization decision***, it must be possible to combine the two separate ***policies*** to form the single ***policy*** applicable to the request.

XACML defines three top-level policy elements: `<Rule>`, `<Policy>` and `<PolicySet>`. The `<Rule>` element contains a Boolean expression that can be evaluated in isolation, but that is not intended to be accessed in isolation by a ***PDP***. So, it is not intended to form the basis of an ***authorization decision*** by itself. It is intended to exist in isolation only within an XACML ***PAP***, where it may form the basic unit of management, and be re-used in multiple ***policies***.

The `<Policy>` element contains a set of `<Rule>` elements and a specified procedure for combining the results of their evaluation. It is the basic unit of ***policy*** used by the ***PDP***, and so it is intended to form the basis of an ***authorization decision***.

The `<PolicySet>` element contains a set of `<Policy>` or other `<PolicySet>` elements and a specified procedure for combining the results of their evaluation. It is the standard means for combining separate ***policies*** into a single combined ***policy***.

XACML defines a number of combining algorithms that can be identified by a `RuleCombiningAlgId` or `PolicyCombiningAlgId` attribute of the `<Policy>` or `<PolicySet>` elements, respectively. The ***rule-combining algorithm*** defines a procedure for arriving at an ***authorization decision***, given the individual results of evaluation of a set of ***rules***.

Standards combining algorithms are defined for

— deny-overrides (ordered and unordered),

— permit-overrides (ordered and unordered),

— first-applicable, and

— only-one-applicable.

A ***rule*** can be evaluated on the basis of its contents. The main components of a ***rule*** are

— a target,

— an effect, and

— a condition.

The ***target*** defines the set of

— resources,

— subjects,

— actions, and

— environment.

The `<Condition>` element may further refine the applicability established by the ***target***. If the ***rule*** is intended to apply to all entities of a particular data-type, then the corresponding entity is omitted from the ***target***. An XACML ***PDP*** verifies that the matches defined by the ***target*** are satisfied by the ***subjects, resource, action*** and ***environment attributes*** in the request ***context***. ***Target*** definitions are discrete, in order that applicable ***rules*** may be efficiently identified by the ***PDP***. The `<Target>` element may be absent from a `<Rule>`. In this case, the ***target*** of the `<Rule>` is the same as that of the parent `<Policy>` element.

The following questions arise: how should a name that identifies a set of ***subjects*** or ***resources*** be interpreted by the ***PDP***, whether it appears in a ***policy*** or a request ***context***? Are they intended to represent just the node explicitly identified by the name, or are they intended to represent the entire sub-tree subordinate to that node?

The ***effect*** of the ***rule*** indicates the rule-writer's intended consequence of a "True" evaluation for the ***rule***. Two values are allowed: "Permit" and "Deny".

Condition represents a Boolean expression that refines the applicability of the ***rule*** beyond the ***predicates*** implied by its ***target***. Therefore, it may be absent.

A.9.13 Transferring XACML policies

Policies may need to be transferred from one entity to another in a PMI. Some of the situations in which this is required are the following.

— A PDP evaluates a policy that references other policies by name. The other policies shall be fetched from a policy administration point (PAP) when required for evaluation.

— A PDP may need to obtain its "root" policy from the enterprise policy administration point, as part of a configuration.

— A resource may be transferred between security domains, and the source domain may transfer a policy for protection of the resource that the destination domain is responsible for enforcing.

— Multiple sites may need to use common policies, even though their PDPs are local for performance reasons. These policies need to be transferred from the central PAP to each site's PDP.

While XACML defines a policy language, it is designed to be one component in an overall authorization system. It relies on other components to provide mechanisms for verifying that policy instances were issued by a trusted PAP, for protecting the integrity and confidentiality of instances of policies, and for protocols used to query for and respond with policy instances. XACML has been integrated with OASIS Security Assertion Markup Language (SAML) Version 2.0 as one way of providing these necessary functions. SAML may be

used with XACML to protect ACI attributes as well as policies. Figure A.2 illustrates the integration of SAML and XACML.

As shown in Figure A.2, when the enforcement point requires an authorization decision, a request is made of the PDP ①. The PDP evaluates the request against its available policies and attributes and produces an authorization decision ② that is returned to the PEP. The PEP may obtain attributes from online AAs ③ or from attribute repositories ④ into which AAs have previously stored attributes ⑤. The PDP may obtain attributes from online AAs ⑥ or from attribute repositories ⑦.

The authorization decision of the PDP is based on policies returned from the PAP ⑧ or retrieved from the online policy repository ⑨. The policy repository serves as a cache of policies previously stored by a PAP ⑩.

The XACMLPolicyQuery is an SAML query defined in this profile that may be used to request policies from a PAP, either by name or by applicability to a certain request. A corresponding XACMLPolicyStatement is returned in an SAML response. The XACMLPolicyStatement may be digitally signed and may be associated with issuer and validity period information, among other things.

Several policies are defined. They may be used individually or in combination.

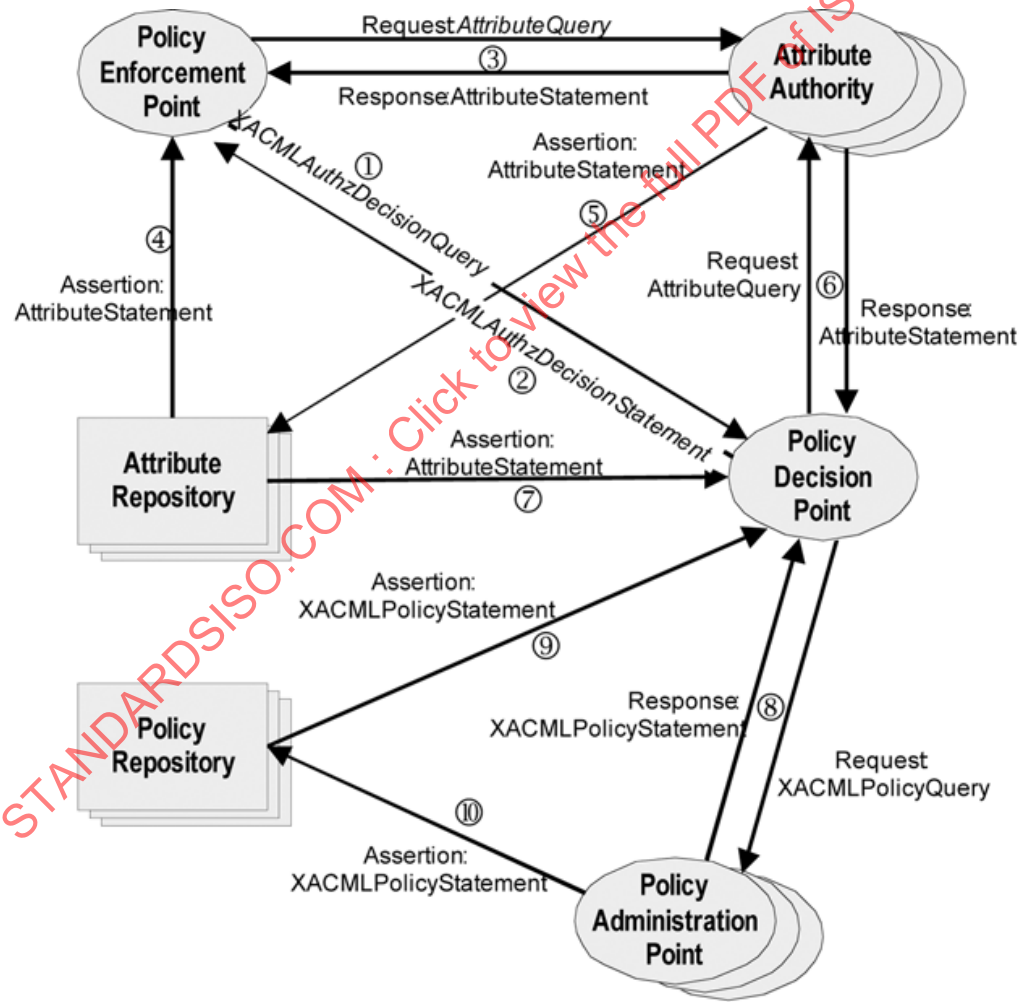

Figure A.2 — Using SAML 2.0 to transport XACML ([12] in the Bibliography)

A.9.14 Mapping between ANS.1 Types and XML Elements

Mapping between standard ASN.1 types and XML elements is done as follows (where possible, this maps to ongoing work on XML schemas).

a) An ASN.1 Boolean maps to an XML element content or attribute with the following (case-insensitive) values: For TRUE: true, yes or 1. For FALSE: false, no or 0. Only equality matching is allowed.

b) An ASN.1 Integer maps to an XML element content or attribute consisting of solely numeric characters, with an optional sign character (+ or –) in front.

c) An ASN.1 Real maps to an XML element or attribute which uses the ASN.1 value notation for a real number.

d) ASN.1 Bit Strings and Octet Strings map to any XML element content (#PCDATA). As the XML schema work evolves, this should map to an XML binary object; such objects may be encoded in base64 for transport, with the encoding indicated either in the schema or as an attribute of the element.

e) ASN.1 Enumerated map to XML attributes of type NMTOKENS (a list of strings), where each string is the identifier of one of the enumerated values.

NOTE 1 XML schemas will likely allow such information to be carried as elements, as well as attributes.

f) ASN.1 Character Strings map to XML element content (#PCDATA).

g) ASN.1 Object Identifiers do not have an equivalent in XML.

NOTE 2 XML uses URIs for some of the same things. A URI namespace or protocol of type OID: could be defined to convey object identifiers in XML.

h) ASN.1 Times (UTC and generalized time) map to XML elements with ISO 8601 syntax.

i) An ASN.1 Sequence maps to a content model group with the comma connector, e.g.:

<!ELEMENT x (foo,bar)>

j) An ASN.1 Choice maps to a content model group with | connector, e.g.:

<!ELEMENT x (foo|bar)>

k) An ASN.1 Set does not map to any XML construct, although it does map to the SGML content group with & connector.

l) An ASN.1 “sequence of” maps to a repeating component using the XML regular expression syntax (* for zero or more occurrences, + for one or more), e.g.:

<!ELEMENT y (foo*)>

m) An ASN.1 “set of” does not map to any XML construct.

n) ASN.1 optional fields map to optional XML components using the ? notation, e.g.: <!ELEMENT z (foo?)>

AAs should ensure that policies are internally consistent, e.g. the same attribute type should not appear in two logically contradictory clauses. Policies should be signed by an AA; they may be conveyed in a claimant's authorization certificate, or as separate objects.

A.9.15 Patient consent

Patient consent is a specific policy. It shall be expressed according to the established policy model, using policy representation languages (for example, XACML).

A.9.16 Credential matching

The credential attributes have been defined in ISO/TS 22600-2:2006, Clause 11, as well as 5.6 and 5.8. They are issued by a trusted authority and include an identification string. Examples include licensing of medical professionals by state boards and assignment of DEA numbers. A credential includes a type, an issuer name, and an identifier.

To match a credential policy, the claimant's certificates shall, in combination, contain a matching credential for each entry in the credential list. To match an entry, the credential shall have the same credential type, and, if the entry has an issuer name, the credential (or enclosing certificate) shall have the same issuer name.

A.9.17 Security label matching

Security label matching compares the initiator's clearance with the target's security label. All of the following must be true for authorization to be granted.

— The security policy identifiers shall be identical.

— The classification level of the initiator shall be greater than or equal to that of the target (that is, there shall be at least one value in the classification list of the clearance greater than or equal to the classification of the target).

— For each security category in the target label, there shall be a security category of the same type in the initiator's clearance, and the initiator's classification level shall dominate that of the target.

A.9.18 General assertion matching

A privilege policy consists of one of the following:

— *ppPredicate:* an assertion about a specific attribute;

— *and* relation: a list of constituent policies, all of which shall be true for this policy to be true;

— *or* relation: a list of simpler policies, at least one of which shall be true for this policy to be true;

— *not* function: a single policy, which shall be false for this policy to be true; or

— *orderedPPE:* a list of simpler policies, which are verified in the order specified.

Predicates may be as follows:

— *single value assertion:* a single attribute value in a target document (or context variable) is compared with an attribute value in the assertion;

— *set value assertion:* the entire set of attribute values in a target document (or context variable) is compared with the set of values in the assertion;

— *present:* the attribute shall be present in the document;

— *approximateMatch:* the asserted value(s) match the value(s) in the document, using some locally defined matching algorithms (for example, phonetic matches or approximate arithmetic matches); or

— *extensibleMatch:* the asserted value(s) match the value(s) in the document using a matching rule defined using the X.500 MATCHING-RULE macro.