NISTIR 7874

Guidelines for Access Control System Evaluation Metrics

Vincent C. Hu
Karen Scarfone

http://dx.doi.org/10.6028/NIST.IR.7874

National Institute of Standards and Technology
U.S. Department of Commerce

NISTIR 7874

Guidelines for Access Control System Evaluation Metrics

Vincent C. Hu
Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg

Karen Scarfone
Scarfone Cybersecurity

http://dx.doi.org/10.6028/NIST.IR.7874

September 2012

U.S. Department of Commerce
Rebecca Blank, Acting Secretary

National Institute of Standards and Technology
Patrick D. Gallagher, Under Secretary of Commerce for Standards and Technology and Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in Federal information systems.

Authority

This publication has been developed by NIST to further its statutory responsibilities under the Federal Information Security Management Act (FISMA), Public Law (P.L.) 107-347. NIST is responsible for developing information security standards and guidelines, including minimum requirements for Federal information systems, but such standards and guidelines shall not apply to national security systems without the express approval of appropriate Federal officials exercising policy authority over such systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), *Securing Agency Information Systems*, as analyzed in Circular A-130, Appendix IV: *Analysis of Key Sections*. Supplemental information is provided in Circular A-130, Appendix III, *Security of Federal Automated Information Resources*.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Interagency Report 7874
48 pages (Sep. 2012)

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by Federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, Federal agencies may wish to closely follow the development of these new publications by NIST.

Organizations are encouraged to review all draft publications during public comment periods and provide feedback to NIST. All NIST publications, other than the ones noted above, are available at http://csrc.nist.gov/publications.

Comments on this publication may be submitted to:

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930), Gaithersburg, MD 20899-8930

Acknowledgments

The authors, Vincent C. Hu of the National Institute of Standards and Technology (NIST) and Karen Scarfone of Scarfone Cybersecurity, wish to thank their colleagues who reviewed drafts of this document, including David F. Ferraiolo, D. Rick Kuhn and Shirley Radack of NIST. The authors also gratefully acknowledge and appreciate the comments and contributions made by government agencies, private organizations, and individuals in providing direction and assistance in the development of this document.

Abstract

Nearly all applications include some form of access control (AC). AC is concerned with determining the allowed activities of legitimate users, mediating every attempt by a user to access a resource in the system. AC systems come with a wide variety of features and administrative capabilities, and their operational impact can be significant. In particular, this impact can pertain to administrative and user productivity, as well as to the organization's ability to perform its mission. Therefore, it is reasonable to use quality metrics to verify the mechanical properties of AC systems. This document discusses the administration, enforcement, performance, and support properties of AC mechanisms that are embedded in each AC system. Because of the rigorous nature of the metrics and the knowledge needed to gather them, these metrics are intended to be used by AC experts who are evaluating the highest security AC systems.

Keywords

access control; access control system evaluation; information security; security metrics

Trademark Information

Table of Contents

List of Appendices

List of Figures

List of Tables

Executive Summary

Adequate security of information and information systems is a fundamental management responsibility. Nearly all applications include some form of access control (AC). AC is concerned with determining the allowed activities of legitimate users, mediating every attempt by a user to access a resource in the system. AC is concerned with how authorizations are structured; in some systems, complete access is granted after successful authentication of the user, but most systems require more sophisticated and complex control.

There are three primary abstractions used in AC system planning: AC policies, models, and mechanisms. AC policies are high-level requirements that specify how access is managed and who may access information under what circumstances. At a high level, AC policies are enforced through a mechanism that translates a user's access request, often in terms of a structure that a system provides. AC models bridge the gap in abstraction between policy and mechanism. Rather than attempting to evaluate and analyze AC systems exclusively at the mechanism level, AC models are usually written to describe the security properties of an AC system.

AC systems come with a wide variety of features and administrative capabilities, and their operational impact can be significant. In particular, this impact can pertain to administrative and user productivity, as well as to the organization's ability to perform its mission. Therefore, it is reasonable to use quality metrics to verify the mechanical properties of AC systems. Since 1) administration is the main consideration of cost, 2) enforcement capabilities are the requirements for AC applications, 3) performance is a major factor for AC usability, and 4) support functions allow an AC system to utilize and connect to related technologies so as to enable more efficient integration with network and host service functions, this publication provides metrics for the evaluation of AC systems based on the features of administration, enforcement, performance, and support of AC properties. Because of the rigorous nature of the metrics and the knowledge needed to gather them, these metrics are intended to be used by AC experts who are evaluating the highest security AC systems.

1. Introduction

1.1 Purpose and Scope

The purpose of this document is to provide Federal agencies with background information on access control (AC) properties, and to help access control experts improve their evaluation of the highest security AC systems. This document discusses the administration, enforcement, performance, and support properties of AC mechanisms that are embedded in each AC system. (Even though this document covers most of the essential AC properties, the listed properties are not necessarily complete.)

This document extends the information in NIST IR 7316, *Assessment of Access Control Systems* [NISTIR 7316], which demonstrates the fundamental concepts of policy, models, and mechanisms of AC systems. Readers of this document should first read [NISTIR 7316].

1.2 Audience

This document assumes that readers are AC experts who also have basic operating system, database, networking, and security expertise. Because of the constantly changing nature of the information technology (IT) industry, readers are strongly encouraged to take advantage of other resources (including those listed in this document) for more current and detailed information.

1.3 Document Structure

This document is divided into the following sections and appendixes:

- Section 2 gives a brief overview of AC and explores the common AC primitives, which are widely used AC conventions used as definitions throughout this document.
- Section 3 introduces the basic abstractions of AC: policies, models, and mechanisms from the perspectives of concept components, XACML [XACML] architecture, policy ontology, and responsible principals.
- The focus of this document is presented in Section 4, which discusses the properties of AC systems, as well as their related components introduced in Section 3 for the AC system evaluation metrics.
- Section 5 is devoted to metric element selection examples.
- Section 6 is the conclusion of the document.
- Appendix A provides additional information on currently available AC languages.
- Appendix B defines the acronyms used in this document.
- Appendix C lists the references for the document.

2. Access Control Primitives

Access control (AC) systems come with a wide variety of features and administrative capabilities, each with their individual (and often proprietary) attributes, functions, and methods for configuring a class of AC policies. Instead of being individually managed, permissions of practical AC mechanisms are organized in terms of and derived from a set of policy specific user attributes, providing a strategy for organizing, managing, and reviewing permission data, and controlling the access requests of subjects. AC primitives are the common operands for AC attributes and functions of AC mechanisms. Abstractly, AC mechanisms apply a set of rules to system states for the purpose of allowing or denying a specified combination of primitives. The rule set is composed according to the AC policy, such that the final process of any AC is the decision making for an access request that matches a set of AC primitives.

AC primitives include subject, object, action, capability, and privileges, which are established through administrative or system assignments. These AC primitives are defined as follows:

Subject: An active entity, generally in the form of a person, process, or device that causes information to flow among objects (see below) or changes the system state. Technically, it is a process/domain pair [NCSC88].

Object: A passive entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are records, blocks, pages, segments, files, directories, directory trees, process, and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, and network nodes [NCSC88].

Action: An active process invoked by a subject. The subject is permitted to perform the action on the object. An action refers to a specific operation applied to an object, such as read and write, and the object requires protection.

Capability: A capability is associated with a subject and specifies the subject's actions. A capability list corresponds to a row of the access control matrix. Each entry in the list is a capability—a pair < set of actions, object>. In most capability-based AC implementations, presenting the appropriate capability guarantees access, so it can be useful to think of capabilities as being similar to tickets. However, unlike tickets, in some systems capabilities can be copied, and there may be the potential for the possessor of a capability to give a copy to someone else (this capacity is itself often represented as a "right"). Clearly, the protection system must ensure that capabilities are not forged or improperly changed and must control how they propagate. Capability-based systems often use special capability architectures [SUMM97].

Privileges: Privileges reduce the access space from a space where any authenticated subject can access all information to a space where specific users can only perform specific actions on specific objects. An individual privilege can be thought of as a primitive tuple of the form (*s*, *a*, *o*) where *s* is a subject belonging to the set of system subjects (*S*), *a* is an action belonging to the set of actions (*A*), and *o* is an object belonging to the set of system objects (*O*). With respect to a privilege (*s*, *a*, *o*), (*a*, *o*) is said to be a capability of *s*, and (*s*, *a*) is said to be an access entry to *o*. While the concept of privilege transcends products and models, representations of privileges (e.g., rules, relations, matrices, access control lists and capability lists) vary considerably. The definition of privilege can be used to compare and normalize different (non-discretionary) AC schemes.

In summary, AC primitives are the fundamental elements of AC policy, thus they are the building blocks of AC rules, which are then enforced by an AC mechanism. To avoid overloading of terms and definitions, the relations between the AC primitives, rules, and mechanism are illustrated in Section 3.

3. Access Control Concepts

The concept of AC can be derived from the processes of constructing AC systems; elements for the construction include high level abstraction (policy, model, and mechanism) and AC primitives. The Extensible Access Control Markup Language (XACML) [XACML], provides the architecture and language scheme for standardizing the construction of rule-based AC policies (RuBAC). In addition, AC policy ontology is used to describe the relations between the high-level abstraction and primitives as described in this section. The AC policy ontology not only illustrates the concept but also unifies the definitions and terms used throughout this document. Note that to focus on the essential ideas, the policy ontology presented is for an AC system serving a single host system; this can be extended by adding application layers and linking to other similar AC policy ontologies to cover a larger scope of operation environment, such as a privilege management system, which encompasses networking, federating, and authentication functions.

Section 3.1 explains the general implementations for AC systems, Section 3.2 introduces the XACML architecture for AC, and Section 3.3 illustrates the AC policy ontology. Finally, Section 3.4 explains the concept of responsible principals.

3.1 AC Implementations

AC *policies* are high-level requirements that specify how access is managed and who, under what circumstances, may access what information. While AC policies can be application-specific and thus taken into consideration by the application vendor, policies are just as likely to pertain to subject actions within the context of an organizational unit or across organizational boundaries. For instance, policies may pertain to object usage within or across organizational units or may be based on need-to-know, competence, authority, obligation, or conflict-of-interest factors. Such policies may span multiple computing platforms and applications. To enforce policies, organizations are required to codify their internal privacy and security policies into machine-enforceable algorithms or AC policy languages to govern the exchange of data within their organizations.

AC *models* are formal presentations of the security policies enforced by AC systems, and are useful for proving theoretical limitations of systems. AC models bridge the rather wide gap in abstraction between policy and mechanism. Access control mechanisms can be designed to adhere to the properties of the model. On the other extreme, a security model will allow for the expression and enforcement of a wide variety of policies and policy classes [FKC03, HFF01]. In general, an AC model can be defined as:

- A formalized computing algorithm or math representation
- A well-recognized formal concept, e.g., levels in Multi-Level Security (MLS), Conflicts of Interests (COI) classes in Chinese Wall [SA99], or roles in Role-Based Access Control (RBAC) [FKC03]
- Formally-defined properties, e.g., confidentiality protection in MLS, or Separation of Duties (SoD) in RBAC

Benefits of using AC models to represent AC policies include embedding properties through formalization, communicating more effectively when describing the behavior of an AC system, and reducing AC rule specification effort.

At a high level, AC policies are enforced through a *mechanism* that translates a user's access request, often in terms of a structure that a system provides. There is a wide variety of structures; for example, a

simple table lookup can be performed to grant or deny access. Although no well-accepted standard yet exists for determining their policy support, some AC mechanisms are direct implementations of formal AC policy concepts [NISTIR7316]. Access control mechanisms can be designed to adhere to the properties of the model by machine implementation using protocols, architecture, or formal languages such as program code.

In many documents and presentations, the concepts of AC policy, model, and mechanism are often overloaded or interchangeably used, which causes confusion in interpreting their roles in AC systems. As they are applied to other computing systems, it is necessary to formally define these implementations in such a way that each implementation represents a different perspective of an AC system. Figure 1 shows the relations and mappings of AC policy, model, and mechanism; an AC system should be initiated by an AC policy, and then formalized with a formal model if there is one that can be applied. AC policy is then physically implemented by hardware or software mechanisms based on the policy directly or the model indirectly. The mapping relations from a policy to either models or mechanisms are one-to-many, as well as from a model to mechanisms, because there are multiple ways to implement or interpret the mapped ones from the application's point of view.

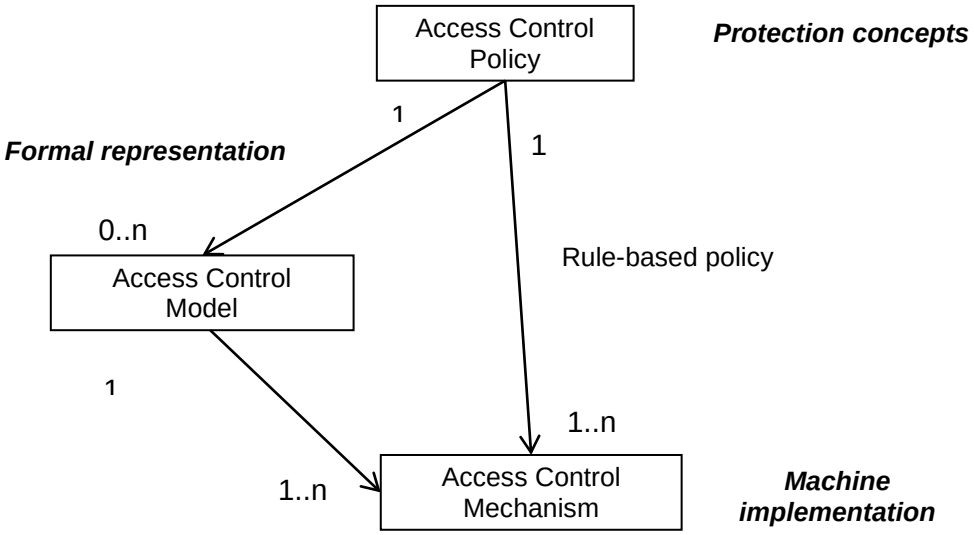

Figure 1 – Mapping of AC policy, model, and mechanism of AC systems

3.2 XACML Architecture

The Organization for the Advancement of Structured Information Standards (OASIS) standard XACML [XACML] is an important authorization-related standard. XACML is an XML-based [XML] general-purpose language used to describe policies, requests, and responses for AC policies. XACML provides a flexible and system-independent representation of access rules that vary in granularities, allowing the combination of policies for different authoritative domains into one policy set for making AC decisions in a widely distributed system environment. The five basic elements of XACML policies are *PolicySet*, *Policy*, *Rule*, *Target*, and *Condition* [XACML]. A policy set is simply a container that holds other policies or policy sets. A policy is expressed through a set of rules.

With multiple policy sets, policies, and rules, XACML must have a way to reconcile conflicting rules. A collection of combining algorithms serves this function. Each algorithm defines a different way to combine multiple decisions into a single decision. Both policy combining algorithms and rule combining algorithms are provided. Seven standard combining algorithms are provided but user-defined combining algorithms are also allowed [SUN].

Despite the language scheme, XACML proposes five basic processing entities in its system schema; each entity handles a different stage in processing a user's access request. As shown in Figure 2, the basic entities are as follows:

- **Policy Decision Point (PDP):** Makes the access decisions by evaluating the applicable policy. PDP implements the decision procedures according to the XACML specification.

- **Policy Administration Point (PAP):** Provides a user interface for creating, testing, and debugging XACML policies, and storing these policies in the appropriate repository.

- **Policy Enforcement Point (PEP):** Performs AC by making decision requests made by the PDP and enforcing authorization decisions.

- **Policy Information Point (PIP):** Serves as the source of attribute values, or the data required for policy evaluation to provide the information needed by the PDP to make the decisions.

- **Policy Retrieval Point (PRP):** Where the policies are stored and fetched by the PDP.

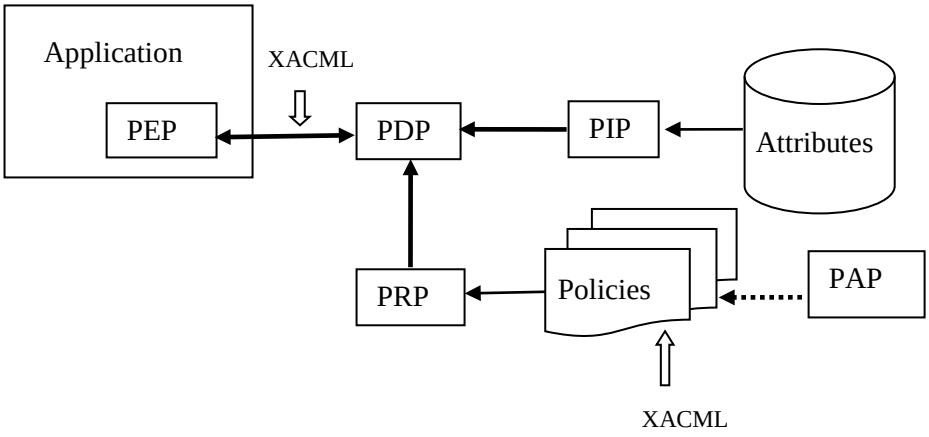

Figure 2 – XACML Architecture

The flexibility and expressiveness of XACML make it complex to work directly with some AC mechanisms. For example, specifying attribute relations in XACML calls for completely specified relations for each and every directly or indirectly related attribute, thus producing a highly verbose document even if the actual policy rules are trivial. (In general, AC policies expressed in an abstract language are difficult for AC policy administrators to create and maintain [UXACML].)

3.3 AC Policy Ontology

In most of the current presentations or documents, AC terms (as in Section 2 and 3.1) for describing AC primitives and implementations are used ambiguously; some documents even treat them as interchangeable concepts. For example, instead of discretionary access control (DAC) "rules", DAC "model" or "policy" is used. In some cases, authors tend to use the word "based" to describe either policy, model, or rule sets without clearly specifying which term they meant, such as Role-Based AC or Policy-Based AC. The policy ontology shown in Figure 3 is an attempt to mediate the terms, primitives, and their relations; thereafter, unambiguous definitions will be used in the rest of this document.

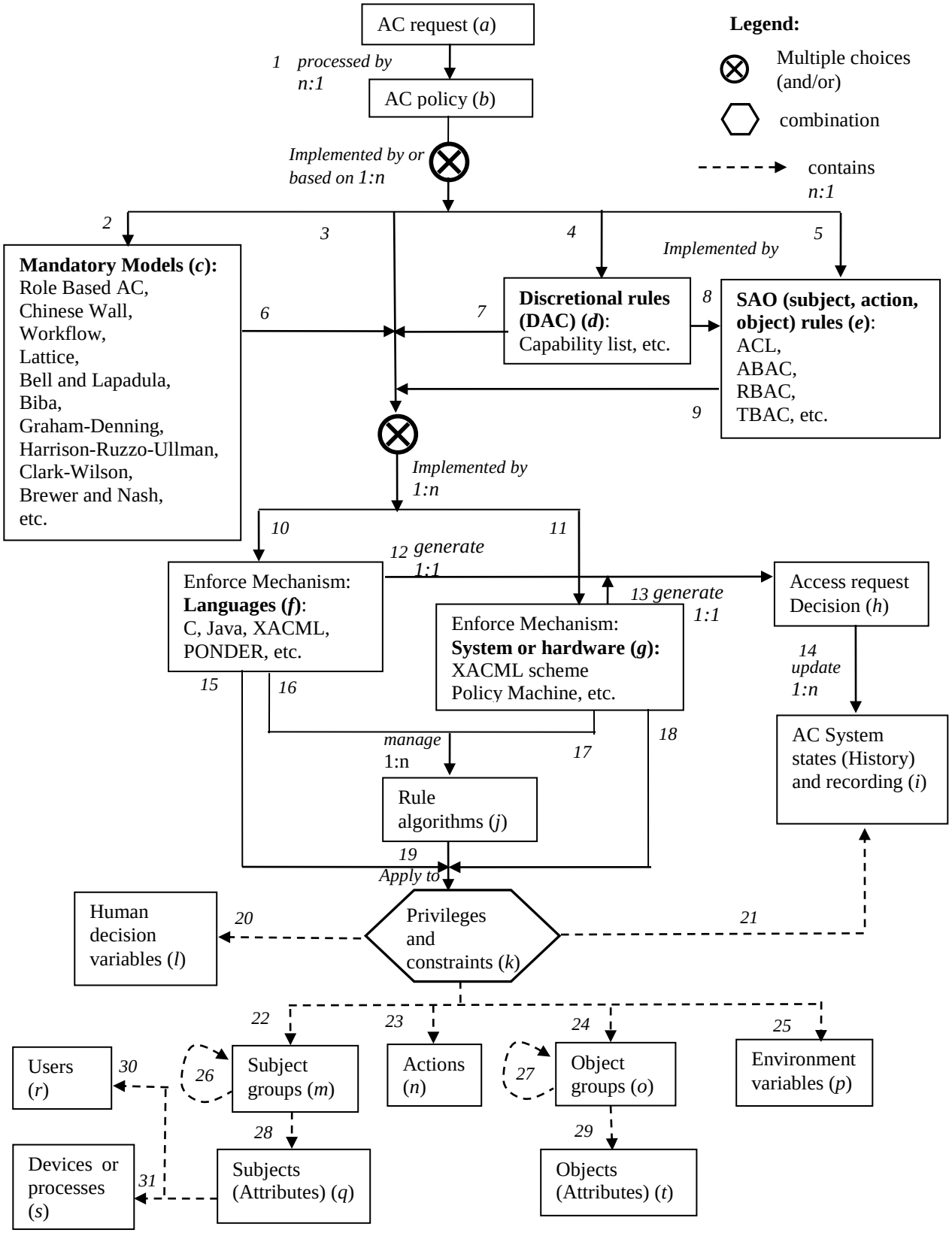

Figure 3 – AC policy ontology

Figure 3 demonstrates the sequence of events within an AC policy ontology.

First, an AC request (*a*) is processed (*1*) by an AC policy (*b*), which can be implemented by (or based on) (*2*, *4*, *5*) mandatory models (*c*), discretional rules (*d*), or SAO (subject, action, object) rules (*e*). These will be either implemented (*6*, *7*, *8*, *9*, *10*, *11*) by language, or system (*f*) and/or hardware mechanism (*g*). AC policy can also be directly implemented (*3*) by languages or system/hardware mechanism without explicit rules or embedded models. Access request decisions (*h*) are the outputs (*12*, *13*) from both mechanisms. Note that DAC rules can be implemented by (*8*) SAO (subject, action, object) rules (*e*). At the same time, an access decision updates (*14*) the AC system states/log (or history for historical or dynamic types of AC models/rules) (*i*).

Language and system/hardware AC mechanisms manage (*15*, *18*) privileges and constraints (*k*) directly or through rule algorithms (*j*) indirectly *(16*, *17*, *18*). Privileges and constraints primitives are the combination of (*20*, *21*, *22*, *23*, *24*, *25*) AC system states (history), human decision variables (*l*), subject groups (*m*), actions (*n*), object groups (*o*), and environment variables (*p*). Subject groups contain (*26*, *28*) other subject groups or subjects (*q*), which contain (*30*, *31*) users (*r*), devices or processes (*s*). Object groups contain (*27*, *29*) other object groups or objects (*t*).

The primitives and relations in the policy ontology show how one primitive element can be broken down into multiple next level elements in cascaded orders. However, the ontology does not comprehensively include all the detailed and special primitive elements that are facilitated for special applications that stem from basic AC functions.

3.4 Responsible Principals

Responsible principals—principals who are responsible for or have impacts on the AC properties of an organization's information system—are identified below:

- **Organization CIO (Chief Information Officer) (*OC*):** oversee the establishment of information systems from the cost, service, and security perspectives of the organization's policy
- **AC policy authors (*PA*):** define or design security policies for the organization's information system according to business practices and security requirements
- **AC system implementers (*SI*):** install, configure and/or implement the AC system in accordance with the PA's design
- **AC system administrators, (operators, or maintainers) (*SA*):** facilitate building, networking, deploying, administrating, and maintaining the AC system
- **Authentication system managers (*ASM*):** responsible for connecting authentication or other service functions for the AC system
- **AC system users (*SU*):** access information through the AC system

4. Evaluation Metrics for Access Control Systems

The ability of an organization to enforce its access policies determines the degree to which its data may be protected and shared among its user community. The focus on sharing and protecting information is becoming increasingly acute for many organizations. Unfortunately, when it comes to AC systems, one size does not fit all. Some AC capabilities are packaged as part of an overall product offering, and others are provided as an add-on feature for managing access configurations within or across architectural abstractions. Although the particular AC mechanism that is included within an application or operating system is rarely scrutinized by an organization in selecting an application, its operational impact can be significant, affecting administrative and user productivity and even the organization's ability to perform its mission. Even for medium-sized enterprises, the number of subjects can be significant; the number of systems that need to be configured for AC can be in the hundreds, and the number of objects that need to be protected can be in the tens of millions. If a single permission is incorrectly configured, a user will either be ineffective in performing his/her duties or will be given unintended access to information and systems, which could result in undermining the security posture of the organization. Further, the quality of administrative capabilities has an impact on administrative cost, user downtime between administrative events, and the abilities of users to perform their duties, as well as the overall security posture of the enterprise. Currently no well-accepted metrics exist for measuring the quality of an AC system.

This section presents properties for quality metrics of AC systems based on the configurable features and limitations of the implemented mechanism using concepts and primitives described in the previous sections. The metrics, which build on those originally defined in [NISTIR7316], can be used when considering and comparing the properties for current configuration or future expansion of an AC system. The properties to be evaluated are divided into four categories according to an organization's operational needs:

- **Administration:** properties that in general impact the cost, efficiency, and performance of an AC system's administration. See Section 4.1.

- **Enforcement:** properties of the mechanisms or algorithms that the AC system uses to enforce the embedded AC models and rules. These properties affect the efficiency of rendering AC decisions. See Section 4.2.

- **Performance:** properties that impact performance in addition to the enforcement of the AC system's processes. See Section 4.3.

- **Support:** properties that are not essential but that can increase the usability and portability of an AC system. See Section 4.4.

The information provided for each function in these four categories follows the same format. For each function, the subsection first lists the responsible principals (see Section 3.4), who are the people who should be responsible or who have the best knowledge of the specific property; the required policy ontology elements (see Section 3.3), which list the concept components that support the property; and the related XACML architecture entities (see Section 3.2). The rest of each subsection lists metric items to evaluate, along with supporting descriptions.

4.1 Administration Properties

This section lists the following functions to be considered for administration properties:

- Auditing

- Privileges/capabilities discovery
- Ease of privilege assignments
- Syntactic and semantic support for specifying AC rules
- Policy management
- Delegation of administrative capabilities
- Flexibilities of configuration into existing systems
- The horizontal scope (across platforms and applications) of control
- The vertical scope (between application, DBMS, and OS) of control

4.1.1 Auditing

Responsible principals: ***OC*** PA SI ***SA*** ASM SU	
Required policy ontology elements: a b c d e ***f g h i*** j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 ***12 13 14*** 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: ***Application PEP*** PDP PRP PIP PAP N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system log system failure?	Log for source of errors records when the AC system fails to make grant decisions.
□ Does the AC system log denied access requests?	Log for attempted policy violations records the denied user request with respect to the AC policies involved.
□ Does the AC system log granted access requests?	Log for access tracking records the granted capabilities of a subject. Because objects can be renamed, copied, and given away, tracking the dissemination and retention of access is difficult or impossible to achieve through privilege expressions alone.
□ Does the AC system provide additional log functions required by the organization?	Customize the audit information-providing capabilities for managing log data (e.g., set the maximum size of audit logs).

4.1.2 Privileges/capabilities discovery

Responsible principals: OC PA ***SI SA*** ASM SU	
Required policy ontology elements: a b c d e ***f g h i j k l m n o p q r s t*** 1 2 3 4 5 6 7 8 9 10 11 12 13 ***14 15 16 17 18 19*** 20 ***21 22 23 24 25 26 27 28 29 30 31*** N/A	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
Does the system provide query/display for □ capabilities discovery? □ privileges discovery? □ constrained capabilities discovery? □ constrained privileges discovery?	• Discover (by query, for example) the capabilities for a given subject/subject groups from assigned privileges or constraints. • Discover the subjects/subject groups for a given capability from assigned privileges or constraints. • Discover the objects/object groups with capabilities for a given subject/subject groups from assigned privileges or constraints.

Does the system provide query/display for human decision variables discovery for □ capabilities? □ privileges? □ constrained capabilities? □ constrained privileges?	• Discover the human decision variables for a given subject/subject group from assigned privileges or constraints. • Discover the human decision variables for a given capability from assigned privileges or constraints. • Discover the human decision variables for a given object/object group from assigned privileges or constraints.
Does the system provide query/display for AC system states discovery for □ capabilities? □ privileges? □ constrained capabilities? □ constrained privileges?	• Discover the system states for a given subject/subject group from assigned privileges or constraints. • Discover the system states for a given capability from assigned privileges or constraints. • Discover the system states for a given object/object group from assigned privileges or constraints.
Does the system provide query/display for environment variables discovery for □ capabilities? □ privileges? □ constrained capabilities? □ constrained privileges?	• Discover the environment variables for a given subject/subject group from assigned privileges or constraints. • Discover the environment variables for a given capability from assigned privileges or constraints. • Discover the environment variables for a given object/object group from assigned privileges or constraints.
□ Does the system provide graphic display?	• The above discovery features are available by What You See Is What You Get (WYSIWYG) display capability.

4.1.3 Ease of privilege assignments

Responsible principals: OC PA ***SI*** ***SA*** ASM SU	
Required policy ontology elements: a b c d e f g h i ***j*** k l ***m*** n ***o*** p ***q*** r s ***t*** 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 ***20*** 21 ***22*** 23 24 25 ***26*** ***27*** 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
How many steps are required for - assigning a privilege? - changing a privilege? - removing a privilege? - assigning a capability to a subject/subject group? - changing a capability for a subject/subject group? - removing a capability for a subject/subject group?	The steps required for assigning, changing, and removing subjects, privileges, or capabilities within the system are crucial to the usability of an AC system. The more steps that are required to perform these updates, the more mistakes that can be made due to either human or system errors. Fewer steps requires less turnaround time for man-machine interaction. For example, in RBAC, individual capabilities are assigned to roles (subject groups) and subjects are made members of roles, thereby acquiring the roles' capabilities. Roles are globally created for various job functions in an organization, and subjects are assigned roles based on their responsibilities and qualifications. Subjects can be granted new capabilities as new applications and systems are incorporated, and capabilities can be easily revoked from roles (and as such revoked from subjects) as needed. If a subject moves to a new function in the organization, the subject can simply be assigned to the new role(s) corresponding to the function and removed from old roles, whereas in the absence of the role abstraction (e.g., identity-based AC), the subject's old capabilities would need to be identified locally and revoked, and new capabilities would have to be granted. Although RBAC offers administrative benefits in assigning/revoking subjects and capabilities, RBAC would not measure as well as other mechanisms in assigning/revoking AC entries to/from objects.
How many steps are required for - assigning subject groups and group relations? - assigning object groups and group relations?	*The steps required for assigning subject group relations*: A critical capability of an AC system is to allow an AC administrator to specify relations between AC attributes of the subjects. With this capability, an AC system is able to maintain hierarchical orders of the attributes of the subjects related to the privileges. The expression of privilege inheritance relations is essential for many popular AC models such as Bell-La Padula and Biba of MLS and Hierarchical Role Based Access Control (HRBAC) [HRBAC] as well as constraint policies such as SoD. *The steps required for assigning, changing, and removing objects' AC entries into the system*: As with the subject group assignments, the objects can be assigned to object groups that might be arranged by secrecy levels of the objects (as in the multilevel security policy). The object groups can also be organized according to the business function; thus, the assigning, changing, and removing operations are in line with the secrecy or business functions of the organization.
How many steps are required for - assigning privilege inheritance?	*The steps required for creating, changing, and removing AC privileges*: AC policies consist of a number of AC privileges made up by AC rules. AC rules are composed through the user interface provided by a mechanism such as AC languages or interactive graphic tools. The interfaces may be efficient in describing particular kinds of rule algorithms while clumsy in describing others when the number of relations required to create AC rules is compared for the policies. For example, some mechanisms allow inheritance of access privileges; if subject *x* is assigned to subject group *A*, which inherits access privileges from group *B*, then *x* automatically inherits all the privileges of *B*. The only required relation for this inheritance assignment is the assigned relation of *x* and *A*. Other mechanisms without the semantic of *A* to *B* inheritance may be required to specify the additional relationship of *x* and *B*.

4.1.4 Syntactic and semantic support for specifying AC rules

Responsible principals: OC ***PA*** ***SI*** SA ASM SU	
Required policy ontology elements: a b c d e ***f*** ***g*** h i ***j*** k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 ***15*** ***16*** 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of logical expression for rule specification? □ Is the AC system capable of programming logic for rule specification?	Privileges and constraints can be specified in complex expressions with Boolean logic relations such as AND, OR, <, =, >, and * (wildcards) between AC primitives. Logic operators provide AC policy authors with efficient ways to specify complex semantics for the intended policy rules. Beyond logic operators, AC policies can be specified by the support of a specific programming language, which supports syntactic as well as semantic grammar required for specifying AC policies. However, current AC languages are designed with a specific application or architecture in mind, so they are not universally applicable for all AC models or mechanisms. Appendix A lists a summary of some AC languages.

4.1.5 Policy management

Responsible principals: OC ***PA*** SI ***SA*** ASM SU	
Required policy ontology elements: a ***b*** c d e f g h i j k l m n o p q r s t 1 ***2*** ***3*** ***4*** ***5*** 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: ***Application*** ***PEP*** ***PDP*** ***PRP*** PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide policy identification for multiple policies?	Query and modify meta-policy information to a policy repository; for example, query a policy identifier for current enforced policies and their relations (e.g., combination effect) with other policies (if the policy is a combination of more than one policy)
□ Does the AC system allow policy expiration assignment?	Lifecycle management, such as expiration date of a policy
□ Does the AC system allow policy target assignment?	Policy targets, such as specific divisions of an organization
□ Does the AC system allow policy target event assignment?	Policy activation, deactivation, and trigger events
□ Does the AC system provide policy combination logic?	Aggregate or deconflict algorithms for combinations for policies
□ Does the AC system require policy distribution approval?	Policy distribution approval
□ Does the AC system provide policy source authorization?	Management of policy authoritative sources
□ Does the AC system provide policy deployment or activation verification?	Ability to verify policy deployment and activation compliance
□ Does the AC system provide policy impact analysis?	For even more robust management, some organizations may want to predict the consequence of activation policies, or analyze the impact when an AC policy is modified or when other policies are combined with the current policy.
□ Does the AC system allow runtime policy rule change?	Some AC systems provide the capability to change access rules during operation. Such a capability allows the extent of policy changes/evolution through equipment/software upgrades, for example, when equipment /software is an object of the AC system.

4.1.6 Delegation of administrative capabilities

Responsible principals: OC ***PA*** SI ***SA*** ***ASM*** SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP PAP N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system allow policy administration delegation?	It may be necessary or convenient for AC system administrators to delegate their privileges to other administrators. Thus, there is a need to consider how easy and secure it is for the AC system to allow policy administration delegation.

4.1.7 Flexibilities of configuration into existing systems

Responsible principals: ***OC*** PA SI SA ASM SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: Application PEP PDP PRP PIP PAP ***N/A***	
Metric Items to Evaluate	**Description**
□ Is the AC mechanism enforced by the operating system? □ Is the AC mechanism enforced by a microkernel?	Some AC systems are integrated within the operating system, while others can be an add-on as a microkernel of an operating system. A microkernel is an approach to operating system design emphasizing small modules that implement the basic features of the system kernel and that is flexibly configured. An AC mechanism using a microkernel-based approach impacts the performance of a system; it has cleaner separation of mechanism and policy specified in the security architecture, enabling a richer set of security policies to be supported, but switching from one policy to another is not an easy task.
□ Is the AC enforced by applications?	An AC mechanism as an application has the advantages of flexibility and ease of installation and uninstallation; however, the AC mechanism suffers from lack of reliability because it can be compromised by attacks.
□ Is the AC enforced by a client/server communication protocol?	Client/server configuration of an AC system is more flexible and secure when compared with the application type of configuration; however, it requires extra hardware, and thus management costs, which include the system communication overhead between clients and server.

4.1.8 The horizontal scope (across platforms and applications) of control

Responsible principals: ***OC PA SI SA ASM SU***	
Required policy ontology elements: a b c d e ***f g*** h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 ***15 16 17*** 18 19 20 ***21*** 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application ***PEP*** PDP PRP ***PIP PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system support only a single host? □ Does the AC system support multiple hosts via network? □ Does the AC system support virtual communities?	Depending on the architecture design, the operational coverage of an AC mechanism may be limited to the scope of platforms, applications, or enterprise environments. Example scopes are single host (as most of the current systems are), distributed network, or virtual community such as cloud or grid systems. Consideration for this feature also applies if the covered scope is under the same AC policy or multiple AC policies for each system unit that can be incorporated into single AC management.

4.1.9 The vertical scope (between application, DBMS, and OS) of control

Responsible principals: OC ***PA*** SI ***SA*** ***ASM*** SU	
Required policy ontology elements: a b c d e ***f*** ***g*** h i j ***k*** l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 ***15*** ***16*** ***17*** ***18*** ***19*** 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application ***PEP*** PDP PRP ***PIP*** ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the scope of data control cover applications? □ Does the scope of data control cover files? □ Does the scope of data control cover database records? □ Does the scope of data control cover the fields of database records? □ Does the scope of data control cover network devices?	The regulation scope of an AC mechanism may be extended from the core OS to the higher system layers; then the extension can be configured through an Application Programming Interface (API) to incorporate the existing ACs of applications, database management systems (DBMS), networks, etc. The vertical scope of the AC system allows different levels of granularity for AC policy and administration.

4.2 Enforcement Properties

This section lists the following functions to be considered for the enforcement properties:

- Policy combination, composition, and constraint
- Bypass
- Separation of Duty (SoD)
- Safety (confinements and constraints)
- Conflict resolution or prevention
- Operational/situational awareness
- Granularity of control
- Expression (policy/model) properties
- Adaptable to the implementation and evolution of AC policies

4.2.1 Policy combination, composition, and constraint

Responsible principals: OC ***PA*** ***SI*** SA ASM SU	
Required policy ontology elements: a b ***c*** ***d*** ***e*** ***f*** ***g*** h i j k l m n o p q r s t 1 2 3 4 5 ***6*** ***7*** ***8*** ***9*** ***10*** ***11*** ***12*** 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP ***PRP*** PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of combining policy rules of different policies? □ Is the AC system capable of combining different policy models? □ Is the AC system capable of combining AC models with rules?	AC policies may be implemented as an application, which allows the implementation by the methods of combination, composition, and constraint of different AC model/rules. In addition to the basic operations, an AC system might analyze the semantic or grammatical expression of AC model/rules and generate a new rule or policy that encapsulates the results of the combination, composition, and constraints logics. For example, it may be convenient or necessary to be able to combine rules for Separation of Duty and Workflow policies.

4.2.2 Bypass

Responsible principals: ***OC*** ***PA*** SI SA ASM ***SU***	
Required policy ontology elements: a b c d e f g h i ***j*** ***k*** ***l*** m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 ***20*** ***21*** 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP PAP N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of bypassing policy rules for critical AC decisions? □ Is the risk for bypassing policy rules tolerable if the AC system is able to bypass policy rules?	If some or all of the AC decision and enforcement will be done at the application level, is the risk of bypassing the mechanism[1] commensurate with the risk tolerance of the enterprise? From the XACML scheme's view, does the AC system allow the application to make access decisions by ignoring the PEP or PDP?

[1] For example, bypass through authorized or unauthorized privileged actions in the operating system.

4.2.3 Least privilege principle support

Responsible principals: ***OC PA SI*** SA ASM ***SU***	
Required policy ontology elements: a b c d e f g h i ***j k*** l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 ***19*** 20 ***21 22*** 23 ***24 25*** 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of enforcing the least privilege principle? □ Does the AC system allow specifying least privilege via constraints? □ Does the AC system allow specifying least privilege via other specifications?	Every subject and process should have the least set of privileges needed to perform the task at hand. The implementation of this principle has the effect of limiting damage that can result from system error or malicious events. In addition to an AC mechanism's reference mediation function, there are two other basic functions: a function to create subjects and associate these subjects with their users, and a function to associate a subject with a subset of attributes. Regardless of its implementation, and the type of attributes that are deployed, reference mediation of an AC system constrains the subject's requests to the capabilities that are associated with a subject's attributes. Although a number of AC mechanisms associate a subject with each and every attribute, in order for an AC mechanism to support the principle of least privilege, constraints must be placed on the attributes that are associated with a subject to further reduce the permissible capabilities. The organization-specific least privilege policy is described by specifying the AC rules, and the AC systems provide various specifying methods, which achieve different degrees of granularity, flexibility, and scope, and different groupings of the controlled objects for the least privilege policies.

4.2.4 Separation of Duty (SoD)

Responsible principals: ***OC PA SI*** SA ASM SU	
Required policy ontology elements: a b ***c d e*** f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of specifying Static SoD rules? □ Is the AC system capable of specifying Dynamic SoD rules? □ Is the AC system capable of specifying Historical SoD rules?	One of the most basic AC policies is to prevent information from unintended access such that object accesses are only permitted to the subjects that are duty-related to the objects. Or, in some business environment, accesses of an organization's objects are controlled to avoid conflict of interests, and therefore restricted to a limited number of subjects. SoD policies have had wide application in business, industry and government and include three basic policy types: Static SoD (SSoD, for example, RBAC), Dynamic SoD (DSoD, for example, Chinese Wall Policy), and Historical SoD (for example, Clark-Wilson) policy. Different AC mechanisms provide different degrees of support for SoD requirements. In general, AC mechanisms allow the attribute assignment of subjects or objects that are more flexible and efficient in supporting SoD than those that do not have the attribute assignment functions. The SoD feature can be measured by counting the number of different types of SoD (such as SSoD, DSoD, and historical SoD) a system can support, as well as the required steps to separate group A and B subjects to the group X and Y objects (see Section 4.1.3).

4.2.5 Safety (confinements and constraints)

Responsible principals: OC ***PA*** ***SI*** SA ASM SU	
Required policy ontology elements: a b c d e f g h i ***j*** k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide safety check capabilities to prevent leaking of permissions?	Safety is an important feature of an AC system, and is needed to ensure that the AC configuration (e.g., AC model) will not result in the leakage of permissions to an unauthorized principal. Thus, a configuration is said to be safe if no privilege can be escalated to an unauthorized or unintended principal. Safety is fundamental to ensuring that the most basic of AC policies can be enforced. Safety is achieved either through the use of restricted AC models that can be proven in general for that model, or via expressions called constraints that describe the safety requirements of any configuration [JT01]. To enforce safety, the AC implementation should include a mechanism for preventing leakage of privileges through either constraints or confinement.[2] The capability of safety enforcement of an AC mechanism can be measured by the number of different types of safety constraints (restrict model or constraint expression such as different types of SoDs) a mechanism can support, as well as how many operational steps are required to build a particular kind of safety constraint. Even though the general safety computation is proven undecidable [HRU76], practical mechanisms exist for achieving the safety requirement, such as safety confinement (constraints) built into the mechanism.

4.2.6 Conflict resolution or prevention

Responsible principals: OC ***PA*** ***SI*** SA ASM SU	
Required policy ontology elements: a b c d e f g h i ***j*** ***k*** l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of preventing policy rule conflicts? □ Is the AC system capable of resolving conflict policy rules? □ Is the AC system capable of preventing policy conflicts (if multiple policies enforcement is available)? □ Is the AC system capable of resolving policy conflicts (if multiple policies enforcement is available)?	Policy rule conflicts appear when the specifications of two or more access rules result in the conflict decision of granting a subject's access request by either direct or indirect access assignments. Policy rule conflicts can also be a result of the deadlock of access rules specification. Deadlock can be defined as: a *rule r* has a dependency on other *rule(s)*, which eventually depend back on *r* itself such that the subject's request will never reach a decision because of the cyclic referencing. In addition to policy rules, when multiple policies are evoked for granting permission, conflicts of policy may occur between policy *X* and policy *Y*. To support conflict resolution, an AC system may provide automatic conflict identification with suggested corrections.

[2] Definition and discussion of "confinement" and "constraints" is beyond the scope of this report.

4.2.7 Operational/situational awareness

Responsible principals: OC ***PA*** ***SI*** ***SA*** ASM SU	
Required policy ontology elements: a b c d e ***f*** ***g*** ***h*** ***i*** ***j*** k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 ***12*** ***13*** 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: ***Application*** ***PEP*** ***PDP*** PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of specifying and enforcing operational/situational awareness control?	Some AC systems allow the PDP to take into account operational/situational factors in making AC decisions. This sort of capability needs to be supported not only by the PDP but by the privilege management application as well. For example, an operational/situational awareness decision is based on a policy such as reaching a threshold value of some environment variable.

4.2.8 Granularity of control

Responsible principals: ***OC*** ***PA*** SI ***SA*** ASM SU	
Required policy ontology elements: a b c d e f g h i j k l m n ***o*** p q r s ***t*** 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 ***27*** 28 ***29*** 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP PRP ***PIP*** ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system allow configuring the granularity of controlled objects (based on the organization's requirements and system architecture)?	This involves the granularity of objects (e.g., data field) an AC system can control. For example, this feature enables the privacy control for the same information with different classification of the data fields in the record. In addition to data record fields, some AC systems are required to control or manage endpoint system components such as servers, workstations, routers, switches, guards, mobile devices, firewalls, email, antivirus, databases, web applications, STU-III (Secure Telephone Unit), STE (Secure Telephone Equipment), or cross domain AC systems.

4.2.9 Expression (policy/model) properties

Responsible principals: OC ***PA*** ***SI*** ***SA*** ASM SU	
Required policy ontology elements: a b ***c*** ***d*** ***e*** f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: Application PEP PDP ***PRP*** PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system support existing AC standards such as those listed in Appendix A?	Existing AC standards such as XACML can provide useful guidance in terms of usage and implementation.
□ Does the AC system support AC rule specification languages (such as those listed in Appendix A)?	Supporting AC languages provides specification syntaxes and schemes to express AC rules.
□ Does the AC system support rule composing using templates of standard AC formal models/mechanisms?	The chosen AC model/mechanism should support the enterprise's AC policy or policies that must be followed, such as MAC, DAC, RBAC, Workflow, Chinese Wall, etc. including the interface to express the model/mechanism.
□ Does the AC system support policy or rule combinations?	Support of policy composition, constraint, and combination; for example, in RBAC and Organizational Units, the Teller role can only access certain resources in his/her Branch, but may be able to access other resources in a different operational unit.

4.2.10 Adaptable to the implementation and evolution of AC policies

Responsible principals: OC ***PA*** ***SI*** ***SA*** ASM SU	
Required policy ontology elements: a b ***c*** ***d*** ***e*** f g h ***i*** ***j*** k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: ***Application*** ***PEP*** PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of handling the evolution of the organization's future AC policy changes? □ Is the AC system capable of dynamically interposing AC rules based on the system states?	AC policies can be as diverse as the applications that rely upon them, and are heavily dependent on the needs of a particular environment. For instance, the mandatory labeling policy for the military, the commercial integrity policy for banking, and the confidentiality policy for healthcare institutions have each modeled their unique policies to meet their own internal control needs and external regulatory requirements. Although these tailored policy models are successful in the specific situations for which they were developed, notations that easily express one collection of AC policies may be awkward (or incapable) in another venue. An example of this situation would be when a company's documents are under MAC policy control at the development stage. When the development is finished, the documents that are available for use by employees could then be required to be controlled by a role-based policy. Most existing commercial technologies used to provide security to systems are restricted to a single policy model, rather than permitting a variety of models to be used. For instance, Linux applies a DAC policy, and it is difficult to implement RBAC policy (among others) in such a system. An AC mechanism that supports the implementing and evolving (combining a policy with new ones, or extending the current policy model) policies can be evaluated by the degree or the number of different well-known policies the mechanism can implement or evolve, such as the support of most of the commonly implemented policies like DAC, MAC and Chinese Wall. Another point of view for adaptability is to consider a computer system abstractly as a state machine performing state transitions, such that a system is considered flexible if the security policy can interpose atomically on any operation performed by the system, allowing the operation to proceed, denying the operation, or even injecting operations of its own. In such a system, the access request decision of the security policy relies on the knowledge of the entire current system state, which includes the history of the system. Because it is possible to interpose on all access requests, it is possible to modify the existing security policy and to revoke any previously granted access. The suggested approach for the needed flexibility is to identify the system state that is potentially security relevant and to control operations that affect or are affected by that state. The degree of flexibility in such a system will depend upon the completeness of both the set of controlled operations and the current system state that is available to the security policy, as well as the granularity of the controlled operations.

4.3 Performance Properties

This section lists the following functions to be considered for the Performance properties:

- Response time
- Policy repository and retrieval
- Policy distribution
- Integrated with authentication function

4.3.1 Response time

Responsible principals: OC ***PA*** ***SI*** ***SA*** ASM ***SU***	
Required policy ontology elements: ***a*** ***b*** ***c*** ***d*** ***e*** ***f*** ***g*** ***h*** ***i*** ***j*** ***k*** ***l*** ***m*** ***n*** ***o*** ***p*** ***q*** ***r*** ***s*** ***t*** ***1*** ***2*** ***3*** ***4*** ***5*** ***6*** ***7*** ***8*** ***9*** ***10*** ***11*** ***12*** ***13*** ***14*** ***15*** ***16*** ***17*** ***18*** ***19*** ***20*** ***21*** ***22*** ***23*** ***24*** ***25*** ***26*** ***27*** ***28*** ***29*** ***30*** ***31*** N/A	
Applicable XACML architecture components: ***Application*** ***PEP*** ***PDP*** ***PRP*** ***PIP*** PAP N/A	
Metric Items to Evaluate	**Description**
□ Does the response time of granting an access request meet the organization's requirement? □ Does the response time for the maximum number of access requests in an expected timeframe meet the organization's requirement? □ Does the response time for activating and revoking AC rules meet the organization's requirement?	Will the AC system be able to process subject requests for access within a time that is consistent with the operational needs of the enterprise? This can be evaluated in part by the complexity of the decision-making algorithm, by modeling, by prototyping, by hardware, and by network. The performance of AC enforcement includes the number of operations required for an AC system to grant a subject's access request, and for the system to check the safety (if available) of an access request. The measurement can be achieved by the computational complexities calculation according to the system model. Note that performance measures are only critical for a system that hosts a large number of subjects. Administrators should evaluate the number of subjects in the worst case to decide if performance needs to be considered.

4.3.2 Policy repository and retrieval

Responsible principals: ***OC*** PA SI ***SA*** ***ASM*** SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: Application PEP PDP ***PRP*** ***PIP*** ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC policy retrieval and deposit meet the organization's safety, operation and cost requirements?	An AC system may store and retrieve AC policies in different forms of repositories. Policies can be stored and retrieved in local, global, federated, or subscribed (e.g., grid and cloud environment) repositories. Also, some AC policies might require connecting to multiple repositories simultaneously. It is important to balance the cost of hardware and software with efficiency based on the organization's requirements.

4.3.3 Policy distribution

Responsible principals: ***OC*** PA SI ***SA*** ***ASM*** SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: Application PEP PDP ***PRP*** ***PIP*** ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide an AC policy distribution mechanism?	A finished policy can be distributed to designated hosting domains including a local policy repository, a global policy repository, or across federations. An AC system may support distribution by manually or automatically pushing to the endpoints. Another consideration is what mechanism(s) for endpoint policy distribution and retrieval the AC system supports: for example, Security Content Automation Protocol (SCAP), Active Directory, LDAP, or proprietary.

4.3.4 Integrated with authentication function

Responsible principals: ***OC*** ***PA*** SI SA ***ASM*** SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p ***q*** r s ***t*** 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 N/A	
Applicable XACML architecture components: ***Application*** PEP PDP PRP ***PIP*** PAP N/A	
Metric Items to Evaluate	**Description**
□ Can the AC system be integrated with or support identification authentication systems (or functions)?	As attributes of subjects and objects can be associated to the identification of users and objects (for example, to achieve SSO), and such attributes are specified in privileges and constraints, for some AC systems it is efficient or required to trust the subject and object attributes provided by the identification system through a secure connection.

4.4 Support Properties

This section lists the following functions to be considered for support properties:

- Policy import and export
- OS compatibility
- Policy source management
- User interfaces and API
- Verification and compliance function support

4.4.1 Policy import and export

Responsible principals: ***OC*** PA SI ***SA*** ASM SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP PAP N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide functions to convert, import, or export external AC policies?	Some AC systems are capable of translating one scheme of AC rule specification to another through import and export file processes. For example, translate from system command to AC rules (e.g., a Simple Network Management Protocol (SNMP) command to a XACML rule), or if the AC mechanism is supported by an AC language, translate a policy from one structured language into another structured language (e.g., from OWL to XACML). Further, the AC system may provide the selection of different forms for exporting as well as provide a reference from a translated policy back to the original source from which it was derived (e.g., an XACML rule back to the original English language rule from which it was translated).

4.4.2 OS compatibility

Responsible principals: OC PA SI ***SA*** ASM SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP ***PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Is the AC system capable of supporting a different OS beside the one used by the intended host(s)?	It is important to check what operating system an AC system is compatible with before using. Some organizations may require an AC system to be portable or flexible to support different platforms hosted by the organization's network environment.

4.4.3 Policy source management

Responsible principals: OC PA ***SI SA ASM*** SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: ***Application*** PEP PDP PRP ***PIP PAP*** N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide a function to maintain or manage the source and destination of AC policies?	For the integrity of the resource, some AC systems, especially those that handle multiple AC policies, are required to manage the authoritative policy source(s). Management functions include identifying authoritative sources of record (creator), identifying authoritative sources of reference (distributor), establishing authoritative source authority, delegating authoritative source authority, updating existing authoritative source authority, terminating authoritative source authority, and maintaining an enterprise authoritative source authority. It is also required to use secure communication channel functions if information exchange among these management functions is necessary.

4.4.4 User interfaces and API

Responsible principals: OC PA ***SI SA*** ASM SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP PAP N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide a GUI or an API for AC policy management and authoring?	For the efficiency of administrating and composing AC policies, it is desirable for an AC system to include a GUI (Graphic User Interface) or an API (Application Interface). Advanced GUI features allow management of AC policies through graphically represented assignment and relations between privileges and constraints. An API provides an interface for AC system developers to add or modify privileges and constraints through more efficient functions.

4.4.5 Verification and compliance function support

Responsible principals: OC ***PA*** SI ***SA*** ASM SU	
Required policy ontology elements: a b c d e f g h i j k l m n o p q r s t 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 ***N/A***	
Applicable XACML architecture components: ***Application*** PEP PDP PRP PIP PAP N/A	
Metric Items to Evaluate	**Description**
□ Does the AC system provide a function to verify or test the AC rules against the intended AC properties from the AC policy authors?	As with any software application, AC rules can be too complex for policy authors to verify if the final rules comply with the intention of the AC policy author. It is also important to ensure there are no leaked privileges due to the syntactic and semantic errors when AC rules were composed or combined. Verification and testing techniques based on the model verification and software testing are usually applied for checking the conformance of AC rules against AC policies [VHU et al, ACPT]. In addition to static verification and testing, dynamic compliance functions such as monitoring the current system states, reporting specific access, alerting privilege conflicts, and remediation of conflict rules are examples of support functions.

5. Metric Element Selection Examples

This section presents examples of using properties items from Section 4 to form evaluation metrics based on the configurable features and limitations of the implemented mechanisms of AC systems. As AC policies reflect the operational requirements of an organization, the quality metric should be evaluated based on the specific needs for the AC policy. For example, some policies regulate only one single host; for these, the element of horizontal scope (Section 4.1.8) should be excluded from the metric for evaluation. Or, when only one kind of AC policy will be applied for the organization, then the adaptability to the implementation and evolution (Section 4.2.10) should not be under consideration. When multiple metric elements are selected for evaluation, one should weigh each of the elements based on the criticality for the organization's mission.

Questions or statements from the metric items should match the organization's requirements for the AC system. Selected AC metric items are categorized as *Critical* if they are necessary for the system, *Optional* if they are desirable but not essential (e.g., improve performance), and *Supplemental* if they will not affect the normal AC operation, but might be required for extension or future services. The sample scenarios below may vary based on the decisions of IT policy makers and technical staff from the different perspectives of the company's resources. Thus some metric items may be placed in different categories than the sample cases. In all, the purpose of the demonstrations is to provide examples of how to use the evaluation metrics in Section 4 for composing AC system requirements from various application environments.

5.1 Example A

Company A decides to set up a publicly accessible web service. The main purpose of the web service is for paid customers to access instructional documents based on their membership status: VIP, Premier, General, and Trial. In addition to the membership status (roles), access to the documents is limited by additional rules; for example, if a customer is also a company employee or has a company-issued coupon, then the customer is allowed access to documents that are one level above the regular access level. Further, the architecture of the AC system for company A's server will be operated in a single host system. The AC policy for the service can be modeled by RBAC model and RuBAC rules. AC metric items from Section 4 that meet the requirements are:

Subsection	Metric Items to Evaluate
Critical Metric Items	
4.2.1 (Policy combination, composition, and constraint)	□ Is the AC system capable of combining policy rules of different policies? □ Is the AC system capable of combining different policy models?
4.2.9 (Expression (policy/model) properties)	□ Does the AC system support rule composing using templates of standard AC formal models/mechanisms? □ Does the AC system support policy or rule combinations?
4.3.1 (Response time)	□ Does the response time of granting an access request meet the organization's requirement? □ Does the response time for the maximum number of access requests in an expected timeframe meet the organization's requirement? □ Does the response time for activating and revoking AC rules meet the organization's requirement?

Optional Metric Items	
4.1.1 (Auditing)	□ Does the AC system log system failure? □ Does the AC system log denied access requests? □ Does the AC system log granted access requests? □ Does the AC system provide additional log functions required by the organization?
4.1.2 (Privileges/capabilities discovery)	Does the system provide query/display for □ capabilities discovery? □ privileges discovery? □ constrained capabilities discovery? □ constrained privileges discovery?
4.1.5 (Policy management)	□ Does the AC system allow policy expiration assignment? □ Does the AC system provide policy combination logic? □ Does the AC system allow runtime policy rule change?
4.2.5 (Safety (confinements and constraints))	□ Does the AC system provide safety check capabilities to prevent leaking of permissions?
4.3.4 (Integrated with authentication function)	□ Can the AC system be integrated with or support identification authentication systems (or functions)?
Supplemental Metric Items	
4.3.2 (Policy repository and retrieval)	□ Does the AC policy retrieval and deposit meet the organization's safety, operation and cost requirements?
4.4.5 (Verification and compliance function support)	□ Does the AC system provide a function to verify or test the AC rules against the intended AC properties from the AC policy authors?

According to the above properties, the AC system of Company A's web service needs to consider the metric items as in Table 1.

Table 1 – Metric items required for AC system of Company A's web service

	Critical metric items	**Optional metric items**	**Supplemental metric items**
Responsible principals	*OC, PA, SA, SI*	*ASM*	
Required policy ontology elements	*c, d, e, f, g, j, 6, 7, 8, 9, 10, 11, 12*	*b, h, i, m, n, o, p, q, r, s, t, 2, 3, 4, 5, 13, 14, 15, 16, 17, 18, 19, 20, 21,22, 23, 24, 25, 26, 27, 28, 29, 30*	*N/A*
Applicable XACML architecture components	*PRP, PAP*	*Application, PEP, PDP, PIP*	*Application, PIP*

From Table 1, establishing the AC system for the web service system, the company's budget and technical planners are able to identify and distinguish the summary and differences (between critical, optional, and supplemental properties) of the required resources, thus they decided the appropriate design of the system from the technical and cost effective points of view.

5.2 Example B

Company B is considering building an AC system for its Internal Business Server (IBS) system. The IBS is physically networked by multiple host systems that belong to different departments of the company. Each host system is running under different operating systems and applications serving business functions

for the department. In addition, each department has different AC policies for managing the required information accesses for its users. Table 2 lists the OS and AC policy used for each individual host (i.e., department) of the IBS system.

Table 2 – IBS hosts

HOST	OS	Local AC model/rule sets	IBS shareable resources
Accounting Department	MS Windows	RBAC	Selected files in the system
Human Resources Department	MS Windows	RuBAC	Selected records in the database
Contracting Department	Mac OS X	Chinese Wall	Selected files in the system
Technical Support Department	Linux	RuBAC	Whole system

To keep the cost and management resources under budget, the company's CIO is planning to purchase an AC system, which is capable of managing cross department access requests, and which is able to be integrated with the company's current authentication system (to achieve SSO). To meet the AC requirements that accommodate each individual host system and its policy, Section 4 provides properties that can be placed into three categories: Critical, Optional, and Supplemental.

Subsection	Metric Items to Evaluate
Critical Metric Items	
4.1.5 (Policy management)	□ Does the AC system provide policy identification for multiple policies? □ Does the AC system allow policy expiration assignment? □ Does the AC system allow policy target assignment? □ Does the AC system provide policy combination logic? □ Does the AC system require policy distribution approval?
4.1.7 (Flexibilities of configuration into existing systems)	□ Is the AC mechanism enforced by the operating system? □ Is the AC enforced by applications? □ Is the AC enforced by a client/server communication protocol?
4.1.8 (The horizontal scope (across platforms and applications) of control)	□ Does the AC system support only a single host? □ Does the AC system support multiple hosts via network? □ Does the AC system support virtual communities?
4.1.9 (The vertical scope (between application, DBMS, and OS) of control)	□ Does the scope of data control cover applications? □ Does the scope of data control cover files? □ Does the scope of data control cover database records? □ Does the scope of data control cover the fields of database records? □ Does the scope of data control cover network devices?
4.2.1 (Policy combination, composition, and constraint)	□ Is the AC system capable of combining policy rules of different policies? □ Is the AC system capable of combining different policy models?
4.2.6 (Conflict resolution or prevention)	□ Is the AC system capable of preventing policy rule conflicts? □ Is the AC system capable of resolving conflict policy rules? □ Is the AC system capable of preventing policy conflicts (if multiple policies enforcement is available)? □ Is the AC system capable of resolving policy conflicts (if multiple policies enforcement is available)?

4.2.9 (Expression (policy/model) properties)	□ Does the AC system support existing AC standards such as those listed in Appendix A? □ Does the AC system support AC rule specification languages (such as those listed in Appendix A)? □ Does the AC system support rule composing using templates of standard AC formal models/mechanisms? □ Does the AC system support policy or rule combinations?
4.3.2 (Policy repository and retrieval)	□ Does the AC policy retrieval and deposit meet the organization's safety, operation and cost requirements?
4.3.4 (Integrated with authentication function)	□ Can the AC system be integrated with or support identification authentication systems (or functions)?
4.4.2 (OS compatibility)	□ Is the AC system capable of supporting a different OS beside the one used by the intended host(s)?
Optional Metric Items	
4.1.1 (Auditing)	□ Does the AC system log system failure? □ Does the AC system log denied access requests? □ Does the AC system log granted access requests? □ Does the AC system provide additional log functions required by the organization?
4.1.2 (Privileges/capabilities discovery)	Does the system provide query/display for □ capabilities discovery? □ privileges discovery? □ constrained capabilities discovery? □ constrained privileges discovery? Does the system provide query/display for human decision variables discovery for □ capabilities? □ privileges? □ constrained capabilities? □ constrained privileges? Does the system provide query/display for AC system states discovery for □ capabilities? □ privileges? □ constrained capabilities? □ constrained privileges? Does the system provide query/display for environment variables discovery for □ capabilities? □ privileges? □ constrained capabilities? □ constrained privileges? Does the system provide graphic display?
4.1.3 (Ease of privilege assignments)	How many steps are required for □ assigning a privilege? □ changing a privilege? □ removing a privilege? □ assigning a capability to a subject/subject group? □ changing a capability for a subject/subject group? □ removing a capability for a subject/subject group? □ assigning subject groups and group relations? □ assigning object groups and group relations? □ assigning privilege inheritance?

4.1.5 (Policy management)	□ Does the AC system allow policy target assignment? □ Does the AC system provide policy source authorization? □ Does the AC system provide policy deployment or activation verification? □ Does the AC system provide policy impact analysis? □ Does the AC system allow runtime policy rule change?
4.2.10 (Adaptable to the implementation and evolution of AC policies)	□ Is the AC system capable of handling the evolution of the organization's future AC policy changes?
4.3.1 (Response time)	□ Does the response time of granting an access request meet the organization's requirement? □ Does the response time for the maximum number of access requests in an expected timeframe meet the organization's requirement? □ Does the response time for activating and revoking AC rules meet the organization's requirement?
4.4.3 (Policy source management)	□ Does the AC system provide a function to maintain or manage the source and destination of AC policies?
4.4.4 (User interfaces and API)	□ Does the AC system provide a GUI or an API for AC policy management and authoring?
4.4.5 (Verification and compliance function support)	□ Does the AC system provide a function to verify or test the AC rules against the intended AC properties from the AC policy authors?
Supplemental Metric Items	
4.1.4 (Syntactic and semantic support for specifying AC rules)	□ Is the AC system capable of logical expression for rule specification? □ Is the AC system capable of programming logic for rule specification?
4.2.2 (Bypass)	□ Is the AC system capable of bypassing policy rules for critical AC decisions? □ Is the risk for bypassing policy rules tolerable if the AC system is able to bypass policy rules?
4.2.3 (Least privilege principle support)	□ Is the AC system capable of enforcing the least privilege principle? □ Does the AC system allow specifying least privilege via constraints? □ Does the AC system allow specifying least privilege via other specifications?
4.2.4 (Separation of Duty (SoD))	□ Is the AC system capable of specifying Static SoD rules? □ Is the AC system capable of specifying Dynamic SoD rules? □ Is the AC system capable of specifying Historical SoD rules?
4.2.5 (Safety (confinements and constraints))	□ Does the AC system provide safety check capabilities to prevent leaking of permissions?
4.2.7 (Operational/situational awareness)	□ Is the AC system capable of specifying and enforcing operational/situational awareness control?
4.2.8 (Granularity of control)	□ Does the AC system allow configuring the granularity of controlled objects (based on the organization's requirements and system architecture)?

According to the above properties, the AC system of IBS needs to examine the metric items as listed in Table 3.

Table 3 – Metric items required for IBS AC system

Work items	Critical metric items	Optional metric items	Supplemental metric items
Responsible principals	*OC, PA, SA, SI, ASM*		*SU*
Required policy ontology elements	*b, c, d, e, f, g, j, q, t, 6, 7, 8, 9, 10, 11, 14, 15, 16, 17, 18*	*h, i, k, l, m, n, o, p, q, r, s, t, 13, 14, 15, 16, 17, 18, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30*	*h, k, o, 12, 15, 21, 22, 23, 27, 28, 29*
Applicable XACML architecture components	*Application, PEP, PDP, PRP PAP, PIP*		

The IBS's budget and technical planners are able to identify and distinguish the summary and differences (between critical, optional, and supplemental properties) of the required resources from Table 3, thus decided the appropriate design of the system from the technical point of view.

5.3 Example C

Research Institute C is researching the group assignment (or attribute relation assignment) feature of AC; group assignment allows the specification of inheritance relations among subject groups, i.e. subject in group *x* can inherit privileges from group *y* if an *inheritance relation* is assigned between them [VDS]. The goal of the research is to compare how the mechanisms ACL, XACML and Policy Machine (PM) [PM] support this feature with respect to some essential properties selected from different categories of metric items in Section 4. Tables 4 to 7 list the properties in column titles, the compared AC mechanisms in row titles, and the supporting metric items from the column properties for the AC mechanism in intersected cells.

Table 4 – Admin properties support for group relation assignment

Admin Properties vs AC mechanism	4.1.2 Privileges/capabilities discovery	4.1.3 Ease of privilege assignments
ACL	□ Does the system provide query/display for capabilities discovery?	□ How many steps are required for assigning a capability to a subject/subject group?
XACML	No support	No support
PM	□ Does the system provide query/display for capabilities discovery? □ Does the system provide query/display for privileges discovery?	□ How many steps are required for assigning a capability to a subject/subject group?

Table 5 – Enforcement properties support for group relation assignment

Enforcement Properties vs AC mechanism	4.2.3 Least Privilege principle support	4.2.4 Separation of Duty	4.2.6 Safety (Confinements and Constraints)
ACL	No support	No support	□ Is the AC system capable of preventing policy rule conflicts?
XACML	No support	No support	No support
PM	□ Does the AC system	□ Is the AC system	No support

	allow specifying least privilege via constraints?	capable of specifying Static SoD rules?	

Table 6 – Performance properties support for group relation assignment

Performance Properties vs AC mechanism	4.3.1 Response time
ACL	□ Does the response time of granting an access request meet the organization's requirement? □ Does the response time for the maximum number of access requests in an expected timeframe meet the organization's requirement? □ Does the response time for activating and revoking AC rules meet the organization's requirement?
XACML	□ Does the response time of an access request meet the organization's operation requirement? □ Does the response time for activating and revoking AC rules meet the organization's requirement? □ Does the response time for safety check meet the organization's operation requirement?
PM	□ Does the response time of an access request meet the organization's operation requirement? □ Does the response time for activating and revoking AC rules meet the organization's requirement? □ Does the response time for safety check meet the organization's operation requirement?

Table 7 – Support properties support for group relation assignment

Support Properties	4.4.4 User Interfaces and API	4.4.5 Verification and compliance function support
ACL	No support	No support
XACML	No support	No support
PM	□ Does the AC system provide a GUI or an API for AC policy management and authoring?	No support

The result from the comparison tables demonstrates that PM is more adept in group relation assignment AC feature than the others. It also provides information about what functions are required for the three mechanisms in order to support this AC feature.

6. Conclusion

This document explains general AC concepts from the perspective of XACML, policy-model-mechanism, and AC ontology in terms of AC primitives. In addition, a list of principals, who are responsible for different AC system tasks, is included. Starting with the explanations of the concept components for an AC system, the main focus of the document is to illustrate a list of AC quality metrics based on four categories of AC properties: administration, enforcement, performance, and support. Each of the property categories is further expanded by available metric items, descriptions, responsible principals, required policy ontology elements, and applicable XACML architecture components. From the AC concepts introduced earlier, the metric items are rendered in either questions or statements that can be used as performance criteria when an AC system is evaluated.

In general, this document covers most of the AC system properties, except those relating to the application and management aspects (e.g., privilege management, privacy support), which are built upon the fundamental properties. Such topics should be covered separately in dedicated documents.

Although only fundamental AC properties are discussed in this document, many extended, combined, and various functions are possible. Tradeoffs and limitations are involved with all AC mechanisms when considering the selection of properties, so it is the organization's responsibility to determine the best-fit AC properties (thus, mechanisms) that work for their business functions and requirements. Note that proper selection of metrics depends not only on the consideration of administration cost, but also on the flexibility of the mechanism used.

Appendix A—Access Control Languages

There are many different languages to express preferences, obligations, and constraints for privacy, identity management, and intellectual property rights. All these languages serve the specific purpose they were created for. However, users want to have a seamless experience, developers want ease of implementation and community groups want respect for user rights. The current languages can't satisfy that concert of requirements. These challenges require the combination of existing ways to manage the flow of information with new techniques to manage policy interactions.

This Appendix provides a non-exhaustive list of policy languages and frameworks, which mainly referred from the PLING [PLING]. In Table 8, each heading bar (shaded gray) shows the title of the language or platform of the system, Description give a short description of the language or platform, Function explains the purpose or service the language or platform system provide, Provider/Specification list the producer and the reference of the system, and Format/Schema lists the output of the system generate.

Table 8 – Review of Policy Languages and Frameworks

ACEL (Autonomic Computing Expression Language)	
Description	A language for creating an expression, parsing it, preparing input for it, and evaluating it. ACEL was originally developed as a part of the Autonomic Computing Policy Language to describe conditions when a policy should be applied to a managed system.
Function	Policies for policy management for autonomic computing
Provider/Specification	IBM / Autonomic Computing Expression Language
Format/Schema	XML
ACPT (Access Control Policy Tool)	
Description	A tool for verifying and testing the design and properties of AC policies; it provides templates for composing and combing popular AC modes and straight AC rules.
Function	Composing, combining, verification, testing AC models and rules
Provider/Specification	NIST / NIST ACPT website
Format/Schema	GUI and XACML
AIR (Accountability in RDF)	
Description	A policy language, represented in Turtle + N3-like quoting, which employs dependency tracking to provide automated explanation generation for policy decisions. It is integrated with the Tabulator extension and has a customized interface for exploring explanations.
Function	Privacy and accountability policies
Provider/Specification	Decentralized Information Group / AIR Policy Language
Format/Schema	Turtle
APPEL (A P3P Preference Exchange Language)	
Description	A language for describing collections of preferences regarding P3P policies between P3P agents.
Function	Privacy preferences
Provider/Specification	W3C P3P Specification Working Group / A P3P Preference Exchange Language 1.0 (APPEL1.0)
Format/Schema	XML
Common Policy	
Description	A framework for authorization policies controlling access to application-specific data.
Function	Access control

Provider/Specification	IETF Geopriv Working Group / RFC4745
Format/Schema	XML
Cross-Origin Resource Sharing	
Description	Mechanism to express policies that enable client-side cross-site requests.
Function	Cross-site AC policies
Provider/Specification	Web Application Formats (WAF) Working Group / Cross-Origin Resource sharing
Format/Schema	HTTP headers, XML Processing Instructions
GeoXACML (Geospatial eXtensible Access Control Markup Language)	
Description	Extends XACML 2.0 by defining the data type "Geometry" and geo-specific functions to declare and enforce access restrictions based on geometric and topological characteristics of the protected objects. GeoXACML supports GML2 and GML3 encoding of geometries.
Function	Access control
Provider/Specification	Open Geospatial Consortium, Inc (OGC) / GeoXACML Implementation Specification
Format/Schema	XML
MPEG-21 Part 5: Rights Expression Language	
Description	ISO/IEC 21000-5:2004 does not give any permission, including permissions about who is legally or technically allowed to create Rights Expressions. It does not specify the security measures of trusted systems, propose specific applications, or describe the details of the systems required for accounting (monetary transactions, state transactions, and so on). It also does not specify if or when Rights Expressions shall be consulted. However, ISO/IEC 21000-5:2004 does define an authorization model to specify whether the semantics of a set of Rights Expressions permit a given Principal to perform a given Right upon a given optional Resource during a given time interval based on a given authorization context and a given trust root.
Function	Digital rights
Provider/Specification	MPEG-21 Working Group / ISO/IEC 21000-5:2004
Format/Schema	XML
P3P (Platform for Privacy Preference Project)	
Description	Enables websites to express their privacy practices in a standard format that can be retrieved automatically and interpreted by user agents.
Function	Privacy
Provider/Specification	W3C P3P Specification Working Group/P3P 1.0 Recommendation
Format/Schema	XML
PERMIS (PrivilEge and Role Management Infrastructure Standards Validation)	
Description	A language for specifying role based authorization control policies for distributed systems.
Function	Role-based authorization control policies
Provider/Specification	Department of Computing, University of Kent, UK / PERMIS
Format/Schema	XML
PM (Policy Machine)	
Description	Provides an infrastructure for central management of AC. PM provides an attribute based universal mechanism for enforcing multiple AC modes and rules through assigning relations between AC attributes and resource containers.
Function	Access control
Provider/Specification	NIST / NIST CSRC web site
Format/Schema	Attribute mapping GUI
Ponder	
Description	A language for specifying management and security policies for distributed systems.

Function	Management and security policies including access control
Provider/Specification	Policy Group, Department of Computing, Imperial College London, UK / Ponder Version 2.3 (pdf)
Format/Schema	EBNF, compiles into XML
Ponder2	
Description	A significant re-design and re-implementation of the Ponder framework for policy-based management. This revised version re-focusses the target application domain of the framework to self-management. The specification language used by the framework draws on SmallTalk syntax and is called PonderTalk.
Function	Management and security policies including access control for self-managing systems.
Provider/Specification	Policy Group, Department of Computing, Imperial College London, UK / PonderTalk
Format/Schema	EBNF, compiles into XML
PRIME Languages	
Description	A privacy-aware AC policy language and a data handling policy language, comprehensive of privacy obligation policies. This R&D work is in progress. The aim has primarily been to deal with privacy management both at the user and enterprise/organizational sides. PRIME R&D work factors in "privacy elements" into policies, including users' preferences and organizational privacy constraints and automates policy decision and enforcement steps. PRIME recognizes that different types of policies and languages are required in the privacy management space, given its complexity and variety of needs and requirements.
Function	Privacy-aware access control
Provider/Specification	Privacy and Identity Management for Europe (PRIME) project / PRIME
Format/Schema	XML
Protune (PROvisional TrUst Negotiation)	
Description	A policy framework meant to support the creation of policies and advanced policy enforcement points, supporting not only traditional AC but also trust negotiation (to automate security checks and privacy-aware information release) and second generation explanation facilities (to improve user awareness about and control on policies).
Function	Privacy-aware AC and trust management.
Provider/Specification	Naples University, Italy / The PROTUNE Project
Format/Schema	EBNF
Rei	
Description	Describes positive and negative permissions and obligations of entities in the policy domain. A distinguishing feature is that it includes specifications for speech acts, policy analysis and conflict resolution. The speech acts included are delegation, revocation, request and cancel and are used for remote policy management. The two kinds of specifications included for policy analysis are use-case management and what-if analysis. As Rei is geared towards distributed environments, it also includes conflict resolution specifications like modality preferences or priority assignments between policies or between individual rules of a policy.
Function	AC
Provider/Specification	UMBC ebiquity research group / Rei Ontology Specification Ver 2.0
Format/Schema	RDF, DAML+OIL and OWL.
RuleML (Rule Markup Language)	
Description	Related to SWRL, although it was produced by a group without the Artificial Intelligence perspective of SWRL; it implements what is known as Horn Logic. It uses the Prolog reasoning engine in most of its versions, and so operates under the Closed World Assumption (CWA).
Function	Web service
Provider/Specification	The Rule Markup Initiative / TheRule Markup Initiative

Format/Schema	XML
SAML (Security Assertion Markup Language)	
Description	An XML-based framework for communicating user. authentication, entitlement, and attribute information
Function	Authentication
Provider/Specification	OASIS / OASIS Security Service(SAML) TC
Format/Schema	XML
SCA (Service Component Architecture) Policy Framework	
Description	Allows policies and policy subjects specified using WS-Policy, as well as with other policy languages to be associated with SCA components.
Function	Web services
Provider/Specification	Open Service Oriented Architecture/ What is SCA
Format/Schema	XML
SWRL (Semantic Web Rule Language)	
Description	In some respects, SWRL resembles Rule Markup Language (RuleML) with the addition of Semantic in the form of the OWL Description Logic (DL) ontology language. However, from a logic standpoint it is less powerful, as it is based on the subset of Prolog known as Datalog. Whereas RuleML uses full Prolog as its inference language.
Function	Web services
Provider/Specification	W3C / SWRL: A Semantic Web Rule Language Combining OWL and RuleML
Format/Schema	OWL
Web Services Policy Framework	
Description	A general purpose model and corresponding syntax to describe the policies of entities in a Web services-based system
Function	Web services
Provider/Specification	W3C Web Services Policy Working Group / WS Policy 1.5 W3C Recommendation
Format/Schema	XML
Web Services Policy Language (WSPL) - WS-XACML	
Description	A profile of XACML for use in a Web Services policy context, it is a popular contender for specifying policies about web services. The syntax is a strict subset of XACML; it is suitable for specifying a wide range of policies, including authorization, quality-of-service, quality-of protection, reliable messaging, privacy, and application-specific service options.
Function	Access control
Provider/Specification	OASIS Extensible Access Control Markup Language TC / V1 Dec 2006
Format/Schema	XML
WIQA-PL (Web Information Quality Assessment Policy Language)	
Description	Allows the description of policies about the quality of information available on the web to be accessed by capturing measures of the quality of the information.
Function	Information quality policies
Provider/Specification	Freie Universität Berlin / Web Information Quality Assessment Policy Language (WIQA-PL)
Format/Schema	RDF
XACML (eXtensible Access Control Markup Language)	
Description	Enables the expression of well-established ideas in the field of access-control policy using an extension language of XML.
Function	Access control
Provider/Specification	OASIS / OASIS eXtensible Access Control Markup Language (XACML) TC

Format/Schema	XML
XrML (eXtensible rights Management Language)	
Description	The XrML language, a Right Expression Language (REL) is one component of an Enterprise Rights Management (ERM) System. Other components are a companion Data Dictionary and system elements that create and interpret REL instances. XrML is used in Windows Media Player to maintain the intellectual property rights of downloaded media. XrML per se is used in proprietary products for a limited number of file formats, and its use entails royalty fees.
Function	Digital rights
Provider/Specification	ContentGuard /XrML
Format/Schema	REL

Appendix B—Acronyms and Abbreviations

Selected acronyms and abbreviations used in the guide are defined below.

AC	Access Control
ACEL	Autonomic Computing Expression Language
ACL	Access Control List
ACPT	Access Control Policy Tool
AIR	Accountability in RDF
API	Application Programming Interface
APPEL	A P3P Preference Exchange Language
ASM	Authentication System Manager
CIO	Chief Information Officer
COI	Conflict-of-Interest
CWA	Closed World Assumption
DAC	Discretionary Access Control
DBMS	Database Management System
DL	Description Logic
DSoD	Dynamic Separation of Duty
EBNF	Extended Backus Naur Form
ERM	Enterprise Rights Management
FISMA	Federal Information Security Management Act
GUI	Graphical User Interface
HRBAC	Hierarchical Role Based Access Control
HTTP	Hypertext Transfer Protocol
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IR	Interagency Report
ISO	International Organization for Standardization
IT	Information Technology
ITL	Information Technology Laboratory
LDAP	Lightweight Directory Access Protocol
MAC	Mandatory Access Control
MLS	Multi-Level Security
MPEG	Moving Picture Experts Group
NIST	National Institute of Standards and Technology
OASIS	Organization for the Advancement of Structured Information Standards
OC	Organization CIO
OGC	Open Geospatial Consortium
OMB	Office of Management and Budget
OWL	Web Ontology Language
P3P	Platform for Privacy Preferences Project
PA	Access Control Policy Author
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PERMIS	PrivilEge and Role Management Infrastructure Standards
PIP	Policy Information Point
PM	Policy Machine
PP	Protection Profile
PRIME	Privacy and Identity Management for Europe

PRP	Policy Retrieval Point
RBAC	Role-Based Access Control
RDF	Resource Description Framework
REL	Rights Expression Language
RuBAC	Rule-Based Access Control
RuleML	Rule Markup Language
SA	Access Control System Administrator
SAML	Security Assertion Markup Language
SAO	Subject, Action, Object
SCA	Service Component Architecture
SCAP	Security Content Automation Protocol
SI	Access Control System Implementer
SNMP	Simple Network Management Protocol
SoD	Separation of Duty
SP	Special Publication
SSO	Single Sign-On
SSoD	Static Separation of Duty
SU	Access Control System User
SWRL	Semantic Web Rule Language
Turtle	Terse RDF Triple Language
UMBC	University of Maryland, Baltimore County
W3C	World Wide Web Consortium
WAF	Web Application Formats
WIQA-PL	Web Information Quality Assessment Policy Language
WYSIWYG	What You See Is What You Get
XACML	Extensible Access Control Markup Language
XML	Extensible Markup Language
XrML	Extensible Rights Management Language

Appendix C—References

[ACPT] Access Control Policy Tool – ACPT, http://csrc.nist.gov/groups/SNS/acpt/index.html.

[FKC03] Ferraiolo D., Kuhn D., and Chandramouli R., "Role-Based Access Control", *Artech House, Computer Security Series*, 2003.

[HFF01] Hu V., Frincke D., and Ferraiolo D., "The Policy Machine For Security Policy Management", *Proceedings ICCS Conference*, San Francisco, May 2001.

[HRU76] Harrison M. A., Ruzzo W. L., and Ullman J. D., "Protection in Operating Systems", *Communications of the ACM*, Volume 19, 1976.

[JT01] Jaeger T. and Tidswell J. E., "Practical Safety in Flexible Access Control Models", *ACM Transactions on Information and System Security*, Vol. 4, No. 2, May 2001, pages 158-190.

[NCSC88] National Computer Security Center (NCSC), "Glossary of Computer Security Terms", *Report NSCD-TG-004*, Fort Meade, Md.: NCSC, 1988.

[NISTIR7316] Hu V., Ferraiolo D., and Kuhn D., "Assessment of Access Control Systems", *NIST IR 7316*, http://csrc.nist.gov/publications/nistir/7316/NISTIR-7316.pdf, 2006.

[PLING] The World Wide Web Consortium (W3C) (http://www.w3.org/) Policy Languages Interest Group – PLING, http://www.w3.org/Policy/pling/wiki/PolicyLangReview, 2011.

[PM] Ferraiolo D., Gavrila S., Hu V., and Kuhn D., "Composing and Combining Policies under the Policy Machine", *Proceeding ACM SACMAT*, Stockholm, Sweden, 2005.

[SA99] Sobel A. E. K. and Alves-Foss J., "A Trace-Based Model of the Chinese Wall Security Policy", page 131, Proceedings, Volume 1, *22nd National Information Systems Security Conference*, Arlington, Virginia, October 18-21, 1999.

[SAML] OASIS, "Security Assertion Markup Language (SAML)". http://www.oasis-open.org/committees/security/, 2005.

[SUMM97] Summers R. C., "Secure Computing Threats and Safeguard", *McGraw-Hill*, 1997.

[SUN] Sun's XACML implementation, http://sunxacml.sourceforge.net/.

[UXACML] Lorch M. et al, "First Experiences Using XACML for Access Control in Distributed Systems," ACM Workshop on XML Security, Fairfax, Virginia, 2003.

[VDS] Hu V., Ferraiolo D., and Gavrila S, "Specification of Attribute Relations for Access Control Policies and Constraints Using Policy Machine", *Proceeding p32-35, the "Sixth International Conference on Information Assurance and Security" (IAS 2010)*, Atlanta, August 23-25, 2010.

[VHU et al] Hu V., Kuhn D., Xie T., and Hwang J., "Model Checking for Verification of Mandatory Access Control Models and Properties", *Int'l Journal of Software Engineering and Knowledge Engineering (IJSEKE) regular issue*, Vol. 21, No. 1, 2011.

[XACML] OASIS, “eXtensible Access Control Markup Language (XACML)”, http://www.oasis-open.org/committees/xacml/.

[XML] M. Naedele. “Standards for XML and web services security”, *Computer*, 36(4):96–98, http://www.tik.ee.ethz.ch/~naedele/Computer03.pdf.