

**National Security Agency
Information Assurance Directorate**



**Net-Centric Enterprise Services (NCES) Profile of eXtensible
Access Control Markup Language (XACML) for Role Based
Access Control (RBAC)**

08 APRIL 2008

**Prepared for the
Defense Information Systems Agency (DISA)**

**By the
National Security Agency (NSA)
9800 Savage Road
Fort George G. Meade, MD 20755**

Table of Contents

1. Scope.....	5
2. General Description.....	5
2.1 Relationship to XACML.....	5
2.2 Goals.....	6
2.3 Position in the Taxonomy.....	7
2.4 Dependencies on other profiles.....	8
2.5 Development Methodology.....	8
3. Profile Scenario.....	9
3.1 Assumptions.....	11
3.2 Modeling tailored constraints in security policies.....	11
3.3 XACML Terminology.....	11
4. Definitions.....	12
5. Profile Requirements.....	12
5.1 Access Control Policies.....	13
5.1.1 PolicySet.....	13
5.1.2 Policy.....	13
5.1.3 Rules.....	14
5.1.4 Target.....	14
5.1.5 Condition.....	15
5.1.6 Combining Algorithms.....	15
5.2 Attributes.....	16
5.3 Policy Decision Point Behavior.....	16
5.3.1 Multiple Subjects.....	16
5.3.2 Interpretation of “NotApplicable”.....	16
5.3.3 Obligations.....	16
6. Conformance.....	16

7. Way Ahead	17
8. Normative References	17
8.1 Informative References	18
9. Abbreviations and Acronyms	19
10. Security considerations	19
10.1 Subject Authentication and Trust	19
10.2 Perimeter Based Security and Policy Decisions	20
10.3 Digital Signing	20
10.4 Confidentiality.....	20
10.5 Impacts on Existing Policies and Processes	20
11. Scope considerations.....	21
12. Other considerations	21
12.1 Relationship to XCCDF	21
12.2 Relationship to NIST SP 800-95	21
12.3 Challenges in Enterprise-level Access Control Across the DoD	21
12.4 Attribute Provisioning Authorities	22
Annex A: Profile Table	23
Annex B: Annex on Role-Based Access Control	44
B.1 Relationship with DCID 6/3	44
B.2 Relationship to Core RBAC	44
B.3 Identity and Attributes	46
B.4 Roles and Groups	47
B.5 Expressing RBAC with XACML Constructs	47
B.5.1 Roles	47
B.5.1.1 Roles as Operational Functions	48
B.5.1.2 Global definitions of roles	48
B.5.2 Objects	49

B.5.3 Operations..... 49

B.5.4 Permissions..... 49

B.5.5 Multi-Role Permissions..... 49

B.6 Transitioning from Roles to Attributes..... 49

B.7 Attribute-Based Access Control and the Limitations of RBAC..... 50

B.8 Tenets of Attribute-Based Access Control..... 51

B.9 Scope Considerations..... 51

1. SCOPE

This profile has been created to establish standard means to express policies and functions within the eXtensible Access Control Markup Language (XACML) construct. The scope of this profile is strictly limited to the definition and evaluation of XACML policy objects. The XACML standard prescribes many policy and functional requirements for compliant decision points. Therefore, implementations conforming to this profile are expected to conform to the requirements for XACML as well as the requirements defined in this document. In addition, implementations that conform to this profile are also expected to support related functionality that is subject to other information assurance (IA) profiles.

This profile specifies implementation options and functional selections for XACML policy definition and evaluation such that conformant implementations will satisfy the conformance requirements for XACML-based authorization services.

2. GENERAL DESCRIPTION

This profile was created to provide guidance on the representation of authorization policies in XACML policy language. This base policy language is used to describe general access control requirements, and has standard extension points for defining new functions, data types, combining logic. Annexes to this document contain specific rules associated with established policy models, such as Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC).

2.1 Relationship to XACML

Figure 1 illustrates the generic XACML policy framework. XACML provides a general-purpose mechanism for expressing an organization's access control policies. At its core, XACML is an access-control policy language that allows standard specification of rules about who can do what and when. XACML provides for fine-grained control of activities based on common types of various criteria, such as entity attributes, authentication mechanisms, and protocol employed. XACML defines a vocabulary for expressing these policies as XML constructs. Each policy defines individual rules as the basic unit of management. Each rule evaluates some combination of characteristics of the requester, the requested resource, the desired action and the current environment. XACML also defines a number of rule-combining algorithms that govern how individual rules within a single policy are evaluated together. XACML also defines a method by which a policy may reference other policies. In addition, as a policy language and evaluation model, the XACML specification complements an access control model, such as RBAC and ABAC in several ways. XACML complements RBAC by establishing a basis for evaluating a wide range of characteristics as part of an access control decision. XACML complements ABAC by establishing a policy language which can express the various tenets of a policy model.

Since the primary focus of this architecture is to provide authorization for the invocation of services, the XACML diagram is tailored to that discussion. Therefore, the Policy Enforcement Point (PEP) is depicted between a service Consumer and a service Provider. The general framework, however, is equally valid for other authorization needs. The primary difference would be the location of the PEP and the Policy Decision Point (PDP).

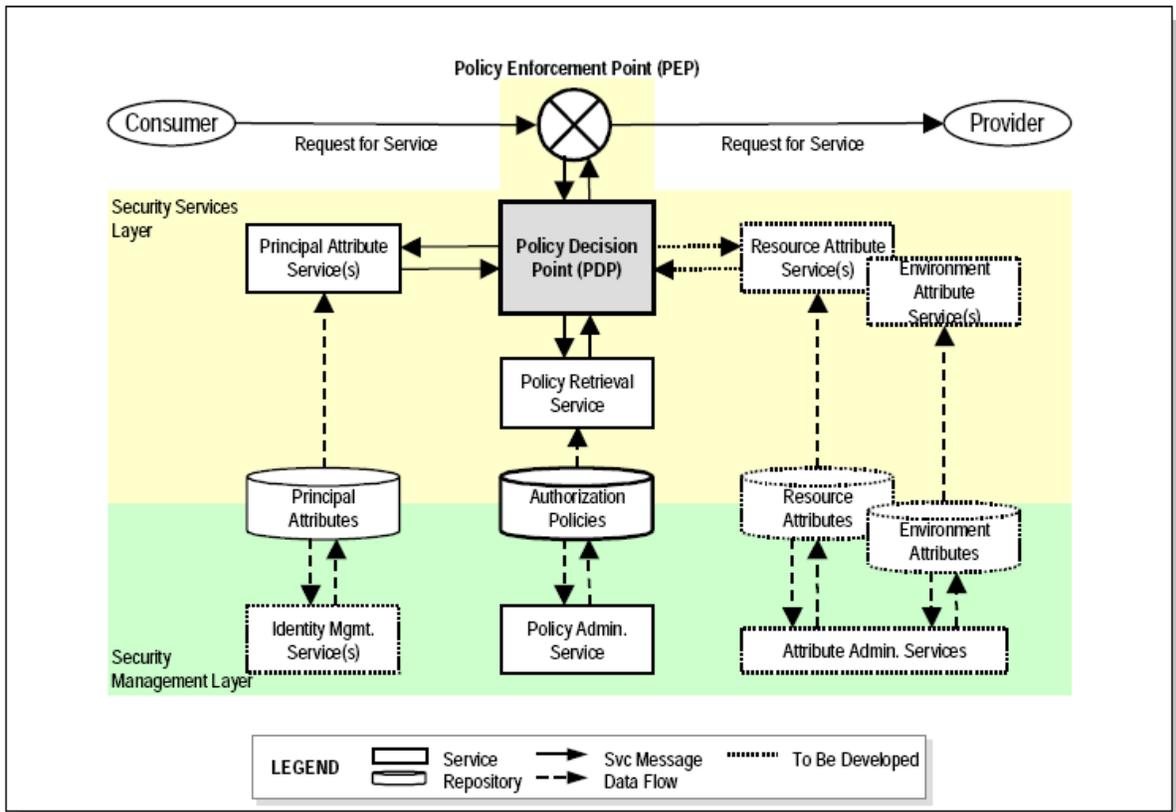


Figure 1 – XACML Policy Framework

XACML also defines an XML based request/response protocol by which access requests are made and the appropriate response determined. Figure 1 also represents the XACML defined authorization decision flow based upon this model. Conceptually, XACML defines two components that work together during the request/response of the authorization decision process: the policy enforcement point (PEP), which enforces a policy decision and the policy decision point (PDP), which evaluates a request, determines which policies to apply and then evaluates those policies to grant the appropriate authorization. This provides for a much richer, more easily understood, and highly extensible approach to authorization. Therefore, profiles are needed to select standard representations for certain access control approaches from all the possible options afforded by XACML to achieve balance between flexibility and interoperability.

2.2 Goals

The goal of this profile is to build a foundation for flexible policy definition for Web Services protected within a SOA environment within the XACML framework.

This document defines no new protocols. Rather, it consolidates and applies many aspects of work accomplished by other standards bodies, particularly OASIS, ANSI and WS-I. Figure 2 illustrates the relationship of this profile to the Web Service standards stack.

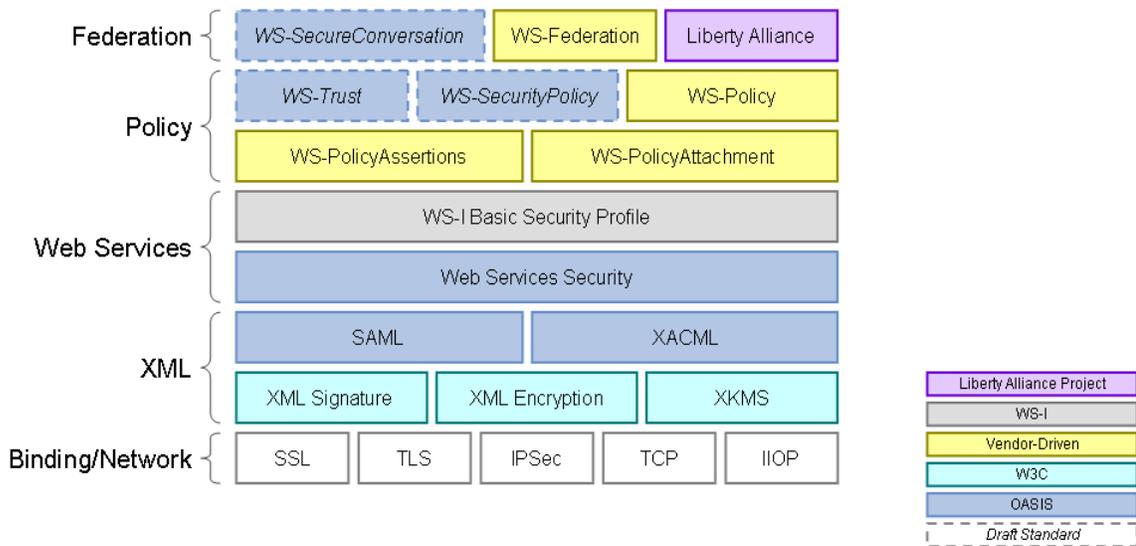


Figure 2 – Relationship of profile to Web Service Standard Stack

2.3 Position in the Taxonomy

This specification is part of a taxonomy of profiles to augment existing web services (WS) standards to meet NCES Information Assurance (IA) requirements. The objectives of these profiles are to:

- Refine WS standards requirements to improve interoperability
- Identify known gaps in the standards without necessarily taking actions on these gaps
- Address known WS security vulnerabilities where possible through interface profiles
- Harmonize standards profiles with existing security architecture efforts, in particular NCES.

The top-level taxonomy of major interface types is defined by the following three categories:



Figure 3 – Top-Level Taxonomy of Major Interface Types

Specifically, this profile is part of the series of Information Element Profiles (i.e., F-profiles) that are described in the IA profiles taxonomy. The position of this profile within the F-profiles taxonomy is illustrated in Figure 4.

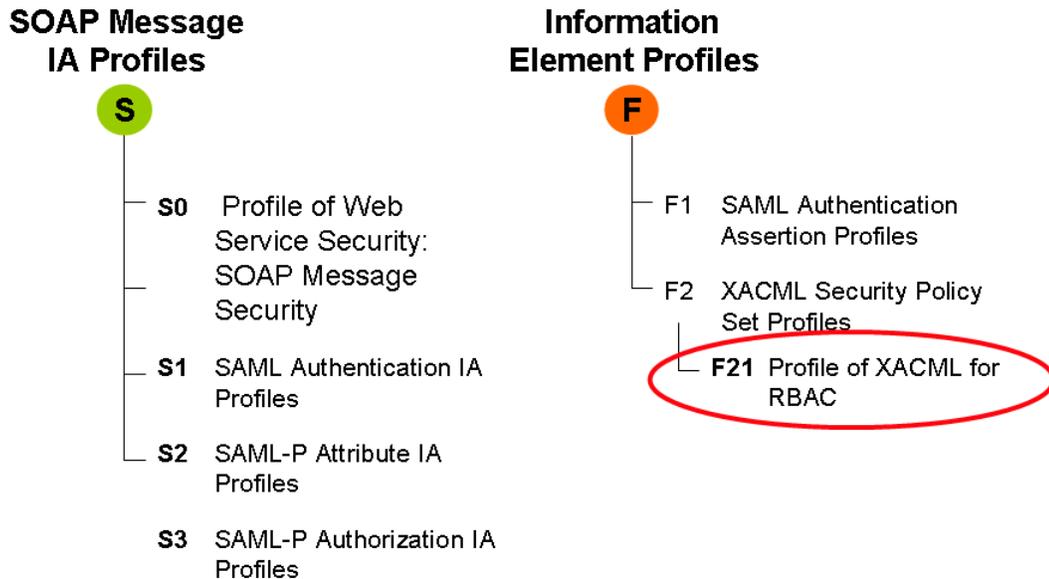


Figure 4 – Information Element Profiles

2.4 Dependencies on other profiles

This section contains information related to the dependencies between this profile and other profiles in the NCES IA profiles taxonomy. As discussed in the security considerations section, this profile is dependent upon the “NCES Profile of Web Service Security: SOAP Message Security (WSSE)” profile to express the rules for exchanging messages which contain policies compliant with this profile. Examples of future profiles which are closely related to this profile include the “Profile for SAML Attribute Assertions.”

2.5 Development Methodology

This profile first examined web services and security standards as well as typical security scenarios in a WS implementation of SOA. From this examination, expressing policy for controlling access to a web service and its operations was selected as the target scenario. Next, identification and examination of existing documents from NCES, OASIS, W3C, NIST and IETF Standards were evaluated as the basic building blocks for the profile. These documents include:

- NCES Service Security Design & Interface Specifications
- XACMLv1.0
- XACMLv2.0
- XML-Signature and Syntax Processing
- XML-Encryption and Syntax Processing
- Web Services Addressing (WS-Addressing) Specification
- NIST documents were used as reference for specifying security requirements

In particular, we investigated specific aspects for this profile (e.g. expression of policy). As a result, we were able to identify a set of alternatives to support secure interoperability. In addition, the selected security scenario required a harmonization of existing standards. Therefore, options and solutions to address standards interoperability issues were identified and validated against the architecture concepts of the scenario. Several IA and interoperability issues identified highlighted interdependencies with, or cascading requirements for, other profiles. These are noted in Section 2.4.

3. PROFILE SCENARIO

As mentioned in Section 2.1, existing industry and community standards such as XACML essentially provide building blocks with which to resolve secure interoperability concerns. Standards represent consensus on needs, requirements and capabilities. These standards may originate from several sources such as International Standards (e.g., IETF), US National Standards (e.g., NIST), Federal or State Regulations (e.g., DoD Directives) and Consortia Specifications (e.g., W3C, OASIS) which identify the level of consensus backing the standard. This consensus forms a basis upon which multiple technologies can provide generalized functionality. However, a standard does not detail how to adopt, adapt and tailor this functionality to best fit the needs of a particular enterprise. Furthermore, individual standards cannot consider how to integrate the capabilities of a set of standards into the best solution for an enterprise. Architecture establishes the context and perspectives for which solutions are developed to address the needs of an enterprise. Therefore, architecture guides the evaluation of the problem space, identification of risks and eventual development of a solution.

The scenario for this profile concentrates on the evaluation of policies within a Web Services environment, such as the NCES architecture. This scenario highlights the critical architecture aspects of a Web Service based service-oriented system that cannot be captured exclusively through the specification of applicable Web Service standards.

The NCES Service Security is comprised of infrastructure-level components that:

- Prevent unauthorized users from accessing Web Services
- Enable enterprise security access policy to be set and enforced
- Provide developers a mechanism to protect deployed service components
- Clearly articulate the business processing rules necessary to enforce access to protected enterprise service components
- Leverage existing industry standards and specifications from standards bodies such as OASIS and the W3C

Figure 5 below illustrates the use of XACML policies with the NCES Security Service architecture.

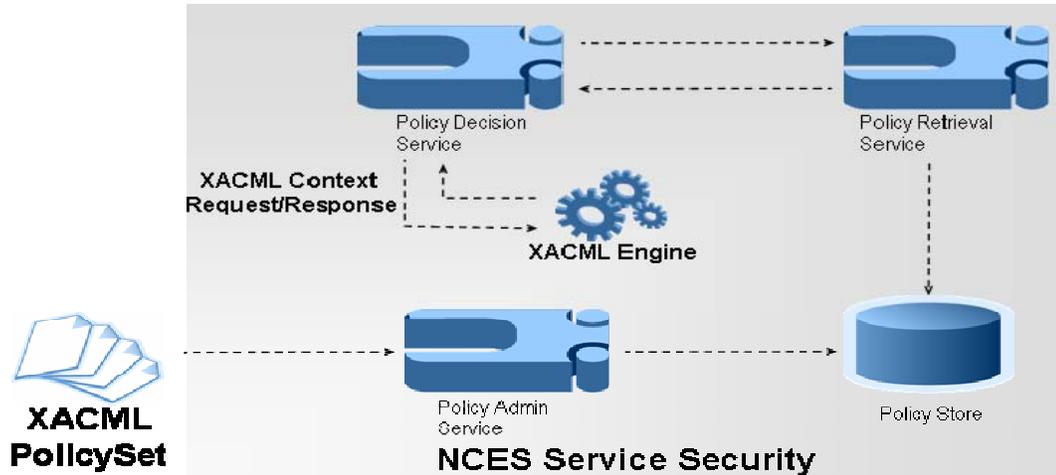
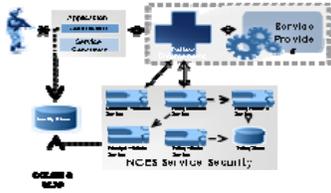


Figure 5 – Profile Scenario

This profile specifies how to construct XACML policies that are specified in terms of subject and resource characteristics instead of individual subject and resource identifiers. This is a natural evolution of identity based access control to improve the scalability and manageability of access control systems. Rules in the above model represent authorization business logic that governs the ability of subjects to perform one or more actions on one or more resources. Note that a resource (e.g., a Web service) usually has multiple actions (e.g., service operations defined in WSDL), although the same action may be applicable to multiple resources. For example, the “getInventory” operation could be defined for multiple services.

Each XACML message exchange consists of a request and response pair to evaluate an appropriate set of XACML policies so to determine whether a particular user should be allowed to perform a given action on a designated resource. The Policy Enforcement Point (PEP) forms an access control request expressed exclusively in attributes, which are characteristics of the relevant Subject, Resource, Action, or Environment. Within this scenario, any roles assigned to the user attempting to access a resource would be expressed as Subject attributes. The PEP sends the Request to a Policy Decision Point (PDP), which may have access to additional information about the subject making the request, the resource acted upon, and the action being requested. It is important to understand that the PEP may include the attributes within the request, the PDP may request attributes from an Attribute Service, or the PDP may validate attributes forwarded from the PEP before passing the request to the XACML engine. As specified by XACML, a PDP retrieves all policies which match the Request (typically via examination of the <Target>). Within the NCES Security Service architecture, the PDP is implemented as a web service entitled “Policy Decision Service” that consumes messages with an XACML message as the payload. The PDS relies on a trusted Policy Retrieval Service to retrieve the correct policies from a policy store. As part of the PDS logic, an XACML engine evaluates the Request against the Policy to authorize access to service operations.

The PDS evaluates the request and responds to the PEP with one of four values that specify whether the access should be allowed. These values are: Permit, Deny, and Not Applicable.¹

3.1 Assumptions

This profile prescribes policies related to a SOA, in which a set of network-accessible operations and associated resources are abstracted as a “service.”

This profile is limited to the evaluation of policies by an XACML Policy Decision Point.

Suitable attributes used within policy rules will be developed by each policy domain.

Attribute definition will neither be defined nor deliberately constrained by system design or standardization processes.

Certain attributes may be defined globally, and therefore will be recognized across the enterprise environment.

Attributes may also be defined locally, and therefore have a limited applicability and scope that depends on the nature of the environment for which they are intended. For example, an agency might create and assign functional roles based on its management structure. These characteristics could be considered by policy sets affecting data objects within that agency’s policy domain. However, other agencies are not obligated to consider characteristics contained in policy sets maintained by other agencies.

A Public Key Infrastructure (PKI) is in place so that users are identified by a PKI certificate.

This profile assumes the SIPRNet meets DoD policy for protecting classified data. Thus, SIPRNet provides confidentiality via underlying communications security. For communications on the NIPRNet, confidentiality MAY be addressed via underlying communications security, message level security or some combination thereof.

It is the responsibility of a data owner to encrypt data. Future enhancements to this profile will address specific types of data that should be encrypted by NCEs. For example, policy and HIPAA attributes are two elements that should be encrypted.

3.2 Modeling tailored constraints in security policies

In a traditional security domain, resources and services are often protected by a uniform set of security rules that are not granular enough to meet specific application needs. Under an SOA, service provider requirements may vary in terms of how they need to be protected. For example, one service may require X.509 certificate-based authentication whereas another service may only need username / password authentication. Furthermore, because clients that access a resource may or may not be from the local domain, different “strengths” of authentication and access control may be required. Consequently, security policies must be expressive and flexible enough to be tailored according to a variety of parameters (e.g., principal attributes).

3.3 XACML Terminology

Attribute: The term “attribute” refers to an XACML <Attribute>. An XACML <Attribute> is an element in XACML having among its components an attribute name identifier, a data type identifier, and an attribute value. Each <Attribute> is associated either with one of the subjects (Subject Attribute), the protected resource (Resource Attribute), the action to be taken on the resource (Action Attribute), or the environment of the Request (Environment Attribute).

¹ The XACML language also defines a fourth return value “*Indeterminate*.”

Attributes are referenced by one of the following: <SubjectAttributeDesignator>, <ResourceAttributeDesignator>, <ActionAttributeDesignator>, or <EnvironmentAttributeDesignator>.

Role: a job function within the context of an organization that has associated semantics regarding the authority and responsibility conferred on the user assigned to the role [ANSI-RBAC].

PDP: Policy Decision Point: An entity that evaluates an access request against one or more policies to produce an access decision.

PEP: Policy Enforcement Point: An entity that enforces access control for one or more resources. When a resource access is attempted, a PEP sends an access request describing the attempted access to a PDP. The PDP returns an access decision that the PEP then enforces.

Policy: a set of rules indicating which subjects are permitted to access which resources using which actions under which conditions.

4. DEFINITIONS

This section establishes the conventions and definitions used in this profile, including specific terminology, general terminology, and the profile classification scheme.

In the context of the body of this profile, the requirements language of the Internet Engineering Task Force will be used. In this document, the keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL are to be interpreted as described in IETF RFC 2119 [RFC2119].

The following general definitions apply to this profile:

Base Standards: References to the base standard or base standards in this profile imply the underlying standards cited as normative references in clause 2.1.

Basic Requirement: A protocol element, function, procedural element, or other identifiable feature specified in the base standards for which support is required by all implementations conforming to this profile.

Functional Group: A specification of one or more related protocol elements, functions, procedural elements or other identifiable features specified in the base standards that together support a significant optional area of functionality in this profile.

5. PROFILE REQUIREMENTS

XACML standardizes the representation of access control policies as XML constructs that can be evaluated as part of an access control decision flow.

This profile provides a basic application of the XACML standard to the typical policy requirements scenarios encountered in a Web services implementation service-oriented architecture (SOA). The following sections address specific aspects of XACML policy expression, such as expression of subjects, roles, permissions, and resources as well as several policy evaluation rules.

This approach builds the scalability and manageability of access control systems required for a web services SOA from the introduction of indirection via specifications and interfaces. Further, this profile supports the concept of least privilege which requires restricting a user within a domain to only the minimum set of privileges necessary to perform a function. Therefore, this

profile REQUIRES access control policies be defined in two complementary stages. The first stage selects the appropriate attributes of the subject, resource and environment to be considered within the access control decision flow. The second stage defines rules according to appropriate combinations of those characteristics.

Implementations of this profile MUST support the normative portions of the [XACML] base standard.

5.1 Access Control Policies

Policies MUST be defined using `<PolicySet>`, `<Policy>` and `<Rules>`, which must be in terms of characteristics of the protected resource (including a resource identifier), actions which may be performed upon the resource (e.g. invoke the web service operation), and characteristics of the principal which may perform the action up the resource (e.g. the clearance granted). Access control policies SHOULD NOT be defined directly through specific subject identifiers. However, policies MAY be based on one or more characteristics, or combination thereof, of the protected resource and one or more characteristic of the subject.

The actual XACML exchange format of the policy may vary depending on the actual query. For instance, if a XACML query is concerned with a specific action on a specific resource, then only that part of the policy concerning the requested action may be returned.

As the NCES architecture addresses authorization within a Service Oriented Architecture (SOA), as implemented through Web Service technologies; the following sub-sections outline a standard set of guidelines for constructing policy to control access to a Web Service.

5.1.1 PolicySet

Generally when using XACML for clarity and granularity sake, a `<PolicySet>` SHOULD be comprised of one or more policies that are applicable to the same resource, and all policies pertaining to the resource SHOULD be contained within the `<PolicySet>`.

- There should be one (1) `<PolicySet>` per Web service. The `<Target>` of the `<PolicySet>` MUST identify the Web Service as the protected resource using values from the associated WSDL for the Web Service (see section 5.1.4).
- Each `<PolicySet>` MUST have an associated `<PolicySetId>`
- The `<Target>` of the `<PolicySet>` SHOULD define applicability in terms of `<Resources>` and SHOULD apply to `<AnySubject>` and `<AnyAction>`.
- The value of **PolicyCombiningAlgID** SHOULD be “Deny-overrides” as defined by the XACML standard.
- Each `<PolicySet>` SHOULD contain a `<Description>` that describes the applicability of the policy in English language.

5.1.2 Policy

The following approach to constructing `<Policy>` is defined within the context of this profile:

- There should be one (1) Policy per Web service operation. The `<Target>` of the `<Policy>` MUST identify the Web Service as the protected resource and the operation to be invoked using values from the associated WSDL for the Web Service (see section 5.1.4).
- One `<Policy>` per Web service operation.
 - Each `<Policy>` MUST have an associated `<PolicyId>`.

- Each *<Policy>* MUST contain two (2) rules with the following **Effect**:
 - **Permit** expressed as Subject Attributes to access the operation.
 - **Deny** (default) if Permit fails.
- The value of **RuleCombiningAlgID** SHOULD be “Deny-overrides” as defined by the XACML standard.
- Each *<Policy>* SHOULD contain a *<Description>* that describes the applicability of the policy in English language.
- The *<Target>* of the *<Policy>* SHOULD define applicability in terms of *<Resources>* and SHOULD apply to *<AnySubject>*.

5.1.3 Rules

The following approach to constructing *<Rule>*s is defined within the context of this profile:

- Each *<Rule>* SHOULD contain a *<Description>* that describes the rule in English language terms.
- Each *<Rule>* SHOULD contain one or more *<Condition>* elements that express the predicates that MUST be satisfied for the rule to be assigned its **Effect** value.
- If it applies to an operation on a Web Service, the *<Rule>* SHOULD NOT define a *<Target>*.

A negative rule is one that is based on a predicate not being "True". If not used with care, negative rules can lead to a policy violation. Therefore, negative and positive rules SHOULD NOT be used together within a single policy.

5.1.4 Target

XACML *<Target>*s MUST be expressed in terms of resource, action and environment attributes.

- The value for **MatchId** MUST be one of:
urn:oasis:names:tc:xacml:1.0:function:-anyURI-equal
urn:oasis:names:tc:xacml:2.0:function:-anyURI-equal
- **Resource Match:**
 - If the protected resource is a Web Service, the *<ResourceMatch>* SHOULD be constructed from the WSDL that defines the functional interface of the service. This SHOULD include the namespace and the service name of the service.
 - The *<ResourceAttributeDesignator>* should match the following attribute within a request context: urn:oasis:names:tc:xacml:1.0:resource:resource-id
- **Action Match:**
 - If the protected resource is a Web Service, the *<ActionMatch>* SHOULD reference a particular operation defined for the Web Service. This should be constructed from the WSDL that defines the operation of the service.
 - The *<ActionAttributeDesignator>* should match the following attribute within a request context: urn:oasis:names:tc:xacml:1.0:resource:action-id

5.1.5 Condition

An XACML *<Condition>* SHOULD be expressed in terms of subject attributes.

- The *<SubjectAttributeDesignator>* should identify the **SubjectCategory**, with the default:
urn:oasis:names:tc:xacml:1.0:subject-category:access-subject assumed.
- The **functionId** for the *<Apply>* element SHOULD be a function specified as mandatory by XACML base specification.

5.1.6 Combining Algorithms

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

The following non-normative XACML policy file illustrates the policy construction guidelines stipulated by this profile.

```
<?xml version="1.0" encoding="UTF-8"?>
<PolicySet PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:deny-overrides" PolicySetId="c2b3405f-65d7-4977-ab72-
bf1ce73861fa" xmlns="urn:oasis:names:tc:xacml:1.0:policy">
  <Description>This PolicySet governs access for any subject attempting to do any action on the WeatherService</Description>
  <Target>
    <Subject s>
      <AnySubject/>
    </Subjects>
    <Resources>
      <Resource>
        <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue
Data Type="http://www.w3.org/2001/XMLSchema#anyURI">urn:geography:weather:0.3:service:WeatherConditions#WeatherService</AttributeValue>
          <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
Data Type="http://www.w3.org/2001/XMLSchema#anyURI"/>
          </ResourceMatch>
        </Resource>
      </Resources>
    <Actions>
      <AnyAction/>
    </Actions>
  </Target>
  <Policy PolicyId="1676c5c7-f25f-4f90-a540-e78d00a2eb42" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-
overrides">
    <Description>This Policy governs access for any subject attempting to invoke getCurrentWeath on the WeatherService</Description>
    <Target>
      <Subjects>
        <AnySubject/>
      </Subjects>
      <Resources>
        <Resource>
          <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
            <AttributeValue
Data Type="http://www.w3.org/2001/XMLSchema#anyURI">:geography:weather:0.3:service:WeatherConditions#WeatherService </AttributeValue>
            <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
Data Type="http://www.w3.org/2001/XMLSchema#anyURI"/>
            </ResourceMatch>
          </Resource>
        </Resources>
        <Actions>
          <Action>
            <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue Data Type="http://www.w3.org/2001/XMLSchema#string">currentWeatherAtLocation</AttributeValue>
              <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
Data Type="http://www.w3.org/2001/XMLSchema#string"/>
            </ActionMatch>
          </Action>
        </Actions>
      </Target>
    </Policy>
  </PolicySet>
```

```

        </ActionMatch>
    </Action>
</Actions>
</Target>
<Rule Effect="Permit" RuleId="2676c5c7-f25f-4f90-a540-e78d00a2eb42">
    <Description>This rule states that any subject may invoke currentWeatherAtLocation on the WeatherService service if they are a member of the
military</Description>
    <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-at-least-one-member-of">
        <SubjectAttributeDesignator AttributeId="role" DataType="http://www.w3.org/2001/XMLSchema#string"
SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"/>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Army</AttributeValue>
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Navy</AttributeValue>
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">Air Force</AttributeValue>
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">USMC</AttributeValue>
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">DOD</AttributeValue>
        </Apply>
    </Condition>
</Rule>
<Rule Effect="Deny" RuleId="DenyRule"/>
</Policy>
</PolicySet>

```

Figure 6 – Example XACML Policy

5.2 Attributes

XACML provides a standard way to reference the attributes defined in the LDAP series of specifications [LDAP-1, LDAP-2]. This is intended to encourage implementers to use standard attribute identifiers for some common subject attributes.

There are instances in which a particular attribute may have multiple values. In fact, common techniques for communicating attributes (e.g. SAML, LDAP and XPath) all support the notion of multiple attribute values. Therefore, multiple values MAY be associated with an attribute. This SHOULD not provoke an error condition at a Policy Decision Point.

5.3 Policy Decision Point Behavior

5.3.1 Multiple Subjects

Users are represented as XACML Subjects. Any of the XACML SubjectCategory values may be used. XACML supports multiple subjects per access request, recognizing that PDP MAY choose to consider any of the entities involved with a single request. This provides support for both the point to point and brokered trust models necessary to security within a SOA environment.

5.3.2 Interpretation of “NotApplicable”

A result of “NotApplicable” means that the PDP could not locate a policy whose target matched the information in the decision request. In general, this profile RECOMMENDS that the PEP enforce a “Deny” effect when a PDP returns a “NotApplicable.”

5.3.3 Obligations

<Obligations> elements associated with a policy are returned to the PEP for enforcement. Therefore, an XACML PEP that conforms to this profile MUST deny access to a protected resource unless it understands and can discharge all of the <Obligations> elements associated with the applicable policy.

6. CONFORMANCE

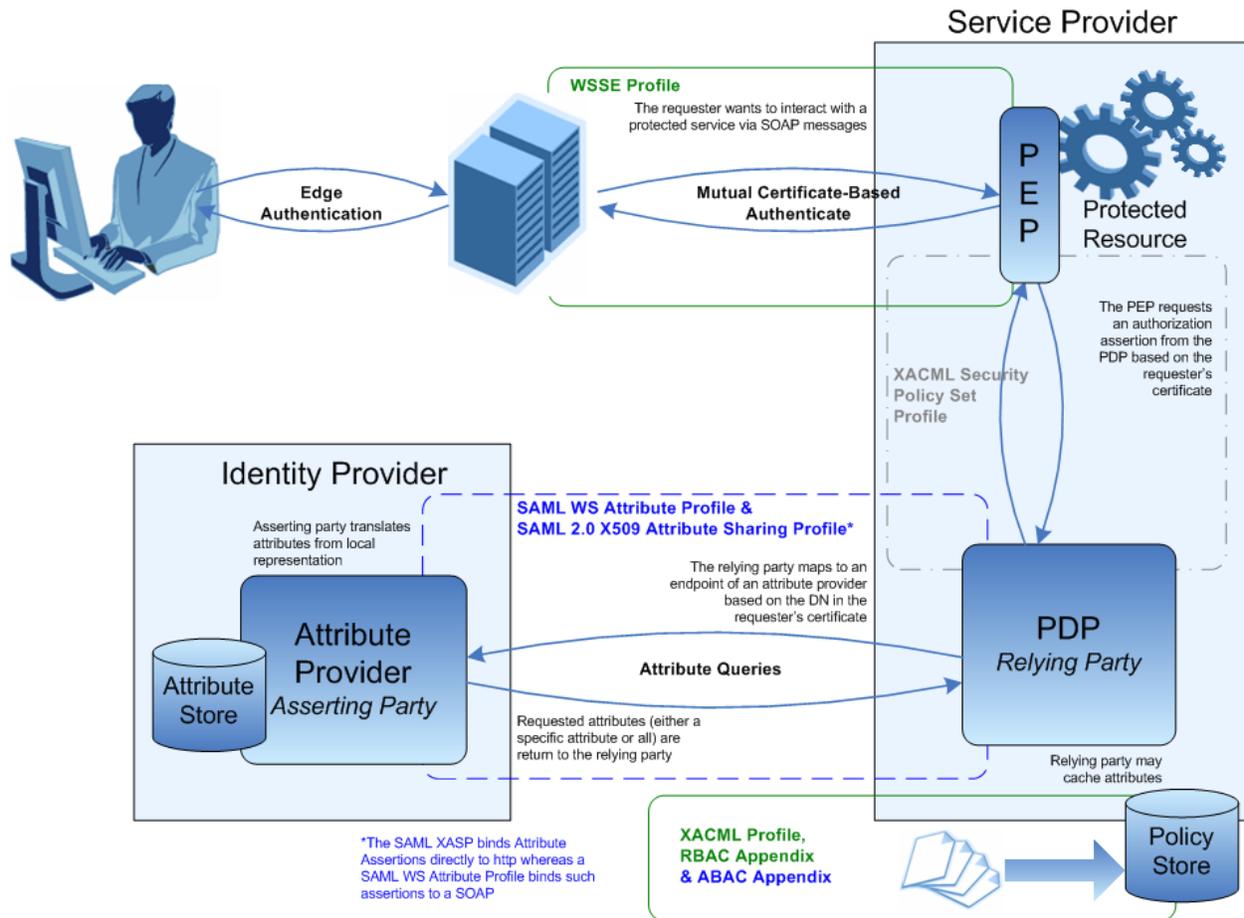
This profile enumerates several functions, elements, and so forth that have special applications; therefore, they are not required to be implemented in an implementation that claims to conform to the OASIS standard.

Additional conformance tests specific to this profile are still under development.

7. WAY AHEAD

This section will contain next steps and future directions of this profile as prioritized by the community. It is recommended that the prioritization of these future directions be determined by the risk assessments.

The following diagram highlights several areas of on-going and future work which closely relate to this profile.



- Profile for exchanging SAML Assertions via SOAP messaging
- Profile for attaching assertions (policy and attribute) to SOAP messages
- Relationships to SAML X.509 Attribute Sharing Profile and the WS-I SAML Token Profile.
- Exchanging and retrieving policies

8. NORMATIVE REFERENCES

The following references form part of the basic foundation of this profile and must be supported by all implementations as qualified by the detailed profile requirements in section 5.

- [RFC2119] *Key Words for use in RFCs to Indicate Requirement Levels*, IETF RFC 2119, March 1997, <http://www.ietf.org/rfc/rfc2119.txt>.
- [XACML] *OASIS eXtensible Access Control Markup Language (XACML) Version 2.0*, OASIS Standard, 1 February 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf.

8.1 Informative References

The following references provide additional information on related concepts that contribute to a fuller understanding of the context of this profile or the rationale for its requirements.

- [ANSI-RBAC] *Information Technology – Role-Based Access Control International Committee for Information Technology Standards, ANSI/INCITS 359*, 3 February 2004. <http://csrc.nist.gov/rbac/>.
- [INTRO] *A Brief Introduction to XACML*, OASIS XACML TC, 14 March 2003, http://oasis-open.org/committees/download.php/2713/brief_introduction_to_xacml.html.
- [NCESarch] *Net-Centric Enterprise Services (NCES) Security Core Enterprise Services (CES) Architecture*, Version 0.4 (Pilot), 2004 March 26.
- [NCESintf] *Net-Centric Enterprise Services (NCES) Service Security Interface Specification*, nces-soaf-ss-interfacespec-2005v01-WD-01, 2005 May 23.
- [NCESspec] *Net-Centric Enterprise Services (NCES) Service Security Design Specification*, nces-soaf-ss-designspec-2005v01-WD-01, 2005 May 23.
- [NCESvuln] *Global Information Grid (GIG) Network Centric Enterprise Services (NCES) Preliminary Vulnerability Assessment Report*, Draft Version 0.8, 2004 October 6.
- [WS-vuln] *Web Services Security Vulnerability Assessment*, Report # I333-020R-2004, 2004 September 15.
- [NIST-RBAC] *Proposed NIST Standard for Role Based Access Control*, D. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, R. Chandramouli, *ACM Transaction on Information and System Security*, Vol.4, No.3, August 2001, pages 224-274, <http://csrc.nist.gov/rbac/rbacSTD-ACM.pdf>.
- [RBAC] *Core and Hierarchical Role-Based Access Control (RBAC) profile of XACML v2.0*, OASIS Standard, 1 February 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf.
- [XACML-RBAC] *XACML profile for Role Based Access Control (RBAC)*, OASIS Access Control TC Committee Draft version 01, 13 February 2004, <http://docs.oasis-open.org/xacml/cd-xacml-rbac-profile-01.pdf>.

9. ABBREVIATIONS AND ACRONYMS

The following abbreviations and acronyms are applicable to this profile.

DISN	Defense Information Systems Network
GES	GIG Enterprise Services
GIG	Global Information Grid
IBAC	Identity-Based Access Control
LBAC	List-Based Access Control
MAC	Mandatory Access Control
NCES	Network Centric Enterprise Services
NIPRNET	Non-classified IP Router Network
PKI	Public Key Infrastructure
RBAC	Role-Based Access Control
SAML	Security Assertion Markup Language
SIPRNET	Secret IP Router Network
SOAP	Simple Object Access Protocol
WS	Web Services
XACML	eXtensible Access Control Markup Language
XML	Extensible Markup Language

In addition, the following abbreviations are used for classification of support levels of base standard features (see Annex A).

m	Mandatory Support
o	Optional Support
c	Conditional Support
i	Out of Scope
–	Not Applicable
r	Required Use
x	Prohibited Use

10. SECURITY CONSIDERATIONS

Expression of authorization policy within the XACML framework provides a flexible and secure way of authorizing that a user is allowed to access restricted resources. However, the messages used to support authorization decisions based on XACML policies are themselves vulnerable to tampering, and must be protected in order prevent a third party from corrupting the process.

10.1 Subject Authentication and Trust

It should be reiterated that ABAC is not concerned with authenticating the subject of a policy. It is assumed that a user requesting access to a resource according to the rules established in a XACML-compliant policy is authenticated at gateway (portal or otherwise) before the authorization process begins.

Rules and policies produced in accordance with this profile are only as reliable as the subjects that create and use them. It is incumbent on each subject to establish appropriate trust in the other subjects on which it relies. Authentication and integrity in both transit and storage should be included. Some requirements would include—

- XACML messages **MUST** be authenticated and protected with respect to integrity and replay detection.
- XACML policies **MUST** be signed by the issuer of the policy.

10.2 Perimeter Based Security and Policy Decisions

The shift towards service-oriented system collaboration and composition also brings fundamental changes to the approach used to define security architectures. Most security solutions that exist today are based on the assumption that both clients and servers are located on the same physical (for example, local LAN) or logical (for example, VPN) network. However, the application of the XACML policy framework to a service oriented environment provides for manipulation and evaluation of policies via services. Services are inherently location independent and not necessarily even bound to a physical location. The network addresses or access points of services can change over time as services are relocated during normal system evolution or for fail-over reasons during system maintenance. Therefore, in a SOA environment, the focus on perimeter-based security models must be augmented with a service-level view of security. The emphasis is placed not on physical ownership and control but on network identities, trust, and authorized access to resources by both users and other principals.

10.3 Digital Signing

To ensure that policy rules are not manipulated by intermediate entities as they are in transit within the environment, they should be digitally signed at the point of creation, by the policy author. Furthermore, if policies are to be exchanged via SOAP messaging, these messages **MUST** be signed according to the rules outlined in the SOAP Message Security specification.

The benefits of signing are two-fold. First, a signature allows the relying party to verify the origin of the message is a trusted source. Second, the message hash included in the signature also ensures that a trusted message was not tampered or modified en route, both during transport and while being processed at an intermediary. This can protect the integrity of assertions, and the entire ABAC process, even if certain entities within the system are compromised.

10.4 Confidentiality

This profile does not mandate mechanisms to protect the confidentiality of XACML policies exchanged between different subjects. XACML policies are expected to be considered sensitive in the target implementation environment. Confidentiality in both transit and storage should be addressed by the policy owner. Therefore:

- Messages which contain XACML policies **SHOULD** be encrypted to protect confidentiality.
- Restricting access to the Policy Store and the Policy Decision Points within the target implementation environment is **RECOMMENDED**. Protection from denial of service attacks and requirements for otherwise achieving a high-availability service should be considered.

10.5 Impacts on Existing Policies and Processes

Current C&A policies generally require identification of system boundaries, whereas in an SOA based network trust relationships are established more dynamically. One possible solution is to define the C&A boundaries at the Web Service interfaces.

11. SCOPE CONSIDERATIONS

The following items are considered out of scope for this profile:

- Issues and considerations related to user authentication.
- Mechanisms for trust establishment, which may be addressed by future profiles within the taxonomy.
- Mechanisms for confidentiality protection.
- Provisioning of attributes (see Section 12.4).

12. OTHER CONSIDERATIONS

12.1 Relationship to XCCDF

The XCCDF specification defines a means for expressing security benchmarks in a way that foster development of interoperable tools and content designed to permit the same document in multiple roles. XCCDF will be used predominantly at the infrastructure level, which is the major audience of these profiles.

The XCCDF specification has the possibility to work well with XACML profiles. Moving forward, the XCCDF specification should be treated as either a framework or a template upon which to base all initial profiles. Each profile will remain flexible according to each scenario. XCCDF also profiles the framework to discuss profile requirements within the context of the Vulnerability Assessments. Finally, XCCDF may fit well in determining how secure policies are in XACML.

12.2 Relationship to NIST SP 800-95

The recent NIST publication, SP 800-95, Guide to Secure Web Services “describes how to implement security mechanisms in Web services. It also discusses how to make Web services robust against the attacks to which they are subject, [and summarizes] security techniques for Web Services.” Therefore, future versions of this profile document will evaluate the requirements contained within the NIST SP 800-95 and align these requirements accordingly.

12.3 Challenges in Enterprise-level Access Control Across the DoD

The Department of Defense (DoD) has many different organizations, all serving different functions. It has become increasingly difficult to formulate a common set of roles for its users. Therefore, this profile defines a standard expression for an arbitrary attribute and suggests several approaches with which a policy domain MAY define the attributes it recognizes. To implement an XACML-based authorization service for a service-oriented architecture, it will be necessary for responsible operational authorities to define the sets of attributes which govern access to the operations of a service.

In addition, the growing popularity of distributed computing and the enhanced exposure of service oriented architecture initiatives such as net-centricity it has become apparent that access control systems should now be able to support sharing of resources with unknown users. Therefore, this profile suggests a migration path toward representing subject roles as attributes so to support authorization for all users within the enterprise.

The end result is an attribute-based access control (ABAC) model. An ABAC implementation provides a holistic profile that extends beyond the subject attributes. An ABAC implementation is also capable of supporting resource attributes and environmental attributes as factors in an access control decision. The ABAC approach can be leveraged to implement other access control models, such as: mandatory access control (MAC), discretionary access control (DAC), identity-based access control (IBAC), lattice-based access control (LBAC), solely, or in combination with RBAC as users see fit and requirements dictate

12.4 Attribute Provisioning Authorities

This profile defines rules for the expression of access control policies in the XACML language in support of the transition to an ABAC approach. This approach controls access to resources through the evaluation of attributes provisioned to various entities. However, this profile does not address the attribute provisioning process through which attributes are assigned to entities. A policy compliant to this profile assumes that all the attributes for a given subject, resource, and environment that will be evaluated in a policy have already been assigned and enabled by the time an authorization decision is requested.

ANNEX A: PROFILE TABLE

The tables in this appendix outline the requirements and restrictions on the individual elements, their values and their arguments as defined by the base standard. As consistent with the hierarchical nature of XML, the classification of information elements is relative to that of the containing information element, if any.

Where the children of a element are not individually specified, then each shall be considered to have the classification of that parent element.

Where the range of values to be supported for an element is not specified, then all values defined in the applicable base standard shall be supported.

To specify the support level of arguments, results and other protocol features for this profile, standardized terminology is used to classify each element – according to static and dynamic behavior.

Static Classifications

Static classifications describe the level of requirements for implementations to support the capability to use a particular feature of the standard. The following classifications are used in this profile to specify static conformance requirements:

Mandatory Support (m): The element or feature shall be supported. An implementation shall be able to generate the element and/or receive the element and perform all associated procedures (i.e., implying the ability to handle both the syntax and semantics of the element) as relevant, as specified in the appropriate base standard. Where support for origination (generation) and reception are not distinguished, both capabilities shall be assumed.

Optional Support (o): An implementation is not required to support the element. If support is claimed, the element shall be treated as if it were specified as mandatory support. If support for origination is not claimed, the element is not generated. If support for reception is not claimed, an implementation may ignore the element on delivery, but will not treat it as an error.

Conditional Support (c): The element shall be supported only under the conditions specified in the profile. If these conditions are met, the element shall be treated as if it were specified as mandatory support. If these conditions are not met, the element shall be treated as if it were specified as optional support (unless otherwise stated).

Out of Scope (i): The element is outside the scope of the profile and will not be the subject of a conformance test, or the element is not applicable in the particular context in which this classification is used.

Dynamic Behavior

Dynamic classifications describe the level of requirements for the behavior of implementations with respect to a particular feature. Dynamic conformance requirements are normally as specified in the applicable base standard. In some cases, however, it is necessary to specify additional dynamic conformance requirements in this profile. These are specified using a second classification code for particular elements. If no dynamic classification code is applied to an element, the required behavior is as specified in the applicable base standard. The following classifications are used in this profile to specify dynamic conformance requirements:

Required Use (r): The element shall always be present. An implementation shall ensure that the element is always generated or otherwise used, as appropriate. Absence of the element on reception shall result in termination or rejection of the communication with an appropriate error indication as specified in the base standards.

Prohibited Use (x): The element shall not be originated by an implementation claiming conformance to this profile. If the element is received it may be treated as a protocol violation unless otherwise stated.

When the requirements of this profile deviate from the requirements of XACML, the rationale column provides motivating information for this difference. This rationale traces to the context and discussion contained within the main body of this profile document.

Global Base Rationale Notes:

A1—Base classification is as per [XACML2] Section 10 schema.

A2—Base classification is as per [XACML2] Section 5 requirements.

A3—Base classification is as per [XACML2] Section 6 requirements.

A4—All enumerated values must be supported to support the element semantics.

A5—Classification is assumed as per selection of [RBAC] as base standard, based on the described model and basic scenario.

A6—Base classification is out of scope because the element is no longer part of the cited base standards.

B1 – No change from the requirement defined in the base standard/protocol.

B2 – Profile classification is out of scope because the element is no longer part of the cited base standards.

Global References and Remarks Notes:

C1 – This element is not present in v1.0 or v1.1 of the XACML specification.

C2 – This element is no longer part of the XACML specification.

Table Column Descriptions:

Ref – Unique identifier assigned to this profile requirement to facilitate references to specific requirements in other profiles

XML Element – Namespace and name of element from base standard

Base Standard Usage – Allowable usage of XML element according to base standard requirements

Base Standard Rationale – Reasoning for expected usage of XML element according to base standard

Profile Usage – Allowable usage of XML element according to the requirements of this profile

Reference and Remarks – Additional information related to this element

Namespaces Prefixes – all elements are defined by the XACML specification suite

Table A.1 Basic Requirements

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	<PolicySet>	m	Global note A1	m	See Table A.2
2	<Request>	m	Global note A1; note based on [XACML] clause 6.1	i	See Table A.6/1
3	<Response>	m	Global note A1; note based on [XACML] clause 6.9	i	See Table A.6/2
4	Data-types	o		o	See Table A.7
5	Attribute-types	o		o	See Table A.8.1
6	Functions	o		o	See Table A.9
7	Functional Requirements	m	Base classification is as per [XACML] Section 7 requirements.	m	See Table A.13

Table A.2 <PolicySet>

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	PolicySetId=	mr	Global note A2	mr	
2	Version=	o	Assumed base “o” because default is present and versioning is not strictly necessary.	o	Global note C1
3	PolicyCombiningAlgId=	mr	Global note A2	mr	See Table A.12
4	<Description>	m	Global note A1	m	
5	<PolicySetDefaults>	o		m	
5.1	<XPathVersion>	c ²	Global note A2	m	
6	<Target> ³	mr	Global note A2	mr	See Table A.3
7	Set of <PolicySet>	m	Global note A1	x m	Recursive; see Table A.2
8	Set of <PolicySetIdReference>	m	Global note A1	mr	
8.1	Version=	o		i	Global note C1
8.2	EarliestVersion=	o		i	Global note C1
8.3	LatestVersion=	o		i	Global note C1
9	Set of <Policy>	m	Global note A1	m	
9.1	PolicyId=	mr	Global note A2	mr	

² If the Policy Set contains <AttributeSelector> elements or XPath-based functions, then “mr,” otherwise “o.”

³ MAY be either *declared* or *calculated*.

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
9.2	Version=	o	Assumed base "o" because default is present and versioning is not strictly necessary.	m	Global note C1
9.3	RuleCombiningAlgID=	mr	Global note A2	mr	See Table A.12
9.4	<Description>	m	Global note A1	m	
9.5	<PolicyDefaults>	o		m	
9.5.1	<XPathVersion>	c ²	Global note A2	m	
9.6	<Target> ³	mr	Global note A2	c	See Table A.3
9.7	Set of <VariableDefinition> ⁴	m	Global note A1	i	Global note C1
9.7.1	VariableId	mr	Global note A2	i	
9.7.2	<Expression>	mr	Global note A2	i	See Table A.4
9.8	Set of <Rule>	m	Global note A1	m	
9.8.1	RuleId=	mr	Global note A2	mr	
9.8.2	Effect=	mr	Global note A2	mr	
9.8.3	<Description>	m	Global note A1	m	
9.8.4	<Target> ⁵	m	Global note A1	c	See Table A.3
9.8.5	<Condition>	m	Global note A1	m	XACML v1.0 and v1.1 hard-wire this directly to Table A.4/1
9.8.5.1	<Expression> ⁶	mr	Global note A2	mr	See Table A.4

⁴ If <VariableDefinition> is supported, then use of the <VariableReference> element shall also be supported throughout the various elements and attributes of the <Policy> element.

⁵ If absent, the DEFAULT value is that of <Target> element of the parent <Policy>.

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
9.9	<Obligations>	o		i	See Table A.5
9.10	<CombinerParameters>	o		i	Global note C1
9.10.1	Set of <CombinerParameter>	o		i	
9.10.1.1	ParameterName=	mr	Global note A2	i	
9.10.1.2	AttributeValue=	mr	Global note A2	i	
9.11	<RuleCombinerParameters>	c ⁷	Global note A2	i	Global note C1
9.11.1	RuleIdRef=	mr	Global note A2	i	
10	Set of <PolicyIdReference>	m	Global note A1	m	
10.1	Version=	o		o	Global note C1
10.2	EarliestVersion=	o		o	Global note C1
10.3	LatestVersion=	o		o	Global note C1
11	<Obligations>	o		o	See Table A.5
12	<CombinerParameters>	o		o	Global note C1
12.1	Set of <CombinerParameter>	o		o	
12.1.1	ParameterName=	mr	Global note A2	mr	
12.1.2	AttributeValue=	mr	Global note A2	mr	
13	<PolicyCombinerParameters>	c ⁷	Global note A2	o	Global note C1
13.1	PolicyIdRef=	mr	Global note A2	mr	
14	<PolicySetCombinerParameters>	c ⁷	Global note A2	o	Global note C1
14.1	PolicySetIdRef=	o	Global note A2	mr	

⁶ Returned data-type must be Boolean.

⁷ If <CombinerParameters> is supported, then “mr,” otherwise “o.”

Table A.3 Target Specification Elements, <Target>

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	<Subjects>	m	Global note A1	m	Base was "mr" in XACML v1.0 and v1.1
1.1	Set of <Subject>	mr	Global note A2	x mr	Base was only "m" in XACML v1.0 and v1.1
1.1.1	Set of <SubjectMatch>	mr	Global note A2	mr	
1.1.1.1	MatchId=	mr	Global note A2	mr	
1.1.1.2	<xacml:AttributeValue>	mr	Global note A2	mr	
1.1.1.3	<SubjectAttributeDesignator>	m	Global note A2	mr	
1.1.1.4	<AttributeSelector>	o	Global note A2	c	
1.2	<AnySubject>	i	Global note A6	i	Note C2 (was "m")
2	<Resources>	m	Global note A1	m	Base was "mr" in XACML v1.0 and v1.1
2.1.1	Set of <ResourceMatch>	mr	Global note A2	mr	
2.1.1.1	MatchId=	mr	Global note A2	mr	
2.1.1.2	<xacml:AttributeValue>	mr	Global note A2	mr	
2.1.1.3	<ResourceAttributeDesignator>	m	Global note A2	mr	
2.1.1.4	<AttributeSelector>	o	Global note A2	c	
2.2	<AnyResource>	i	Global note A6	i	Note C2 (was "m")
3	<Actions>	m	Global note A1	m	Was "mr" in XACML v1.0 and v1.1
3.1.1.1	MatchId=	mr	Global note A2	mr	
3.1.1.2	<xacml:AttributeValue>	mr	Global note A2	mr	

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
3.1.1.3	<ActionAttributeDesignator>	m	Global note A2	mr	
3.1.1.4	<AttributeSelector>	o	Global note A2	c	
3.2	<AnyAction>	i	Global note A6	i	Note C2 (was "m")
4	<Environments>	m	Global note A1	m	Global note C1
4.1	Set of <Environment>	mr	Global note A2	mr	
4.1.1	Set of <EnvironmentMatch>	mr	Global note A2	mr	
4.1.1.1	MatchId=	mr	Global note A2	mr	
4.1.1.2	<xacml:AttributeValue>	mr	Global note A2	mr	
4.1.1.3	<EnvironmentAttributeDesignator>	m	Global note A2	m	
4.1.1.4	<AttributeSelector>	o	Global note A2	o	

Table A.4 Value Expression, <Expression>

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	<Apply>	m	Global note A1	i	XACML v1.0 and v1.1 collapse this element with the <Expression> element.
1.1	FunctionId=	mr	Global note A2	i	

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1.2	<Expression>	m	Global note A1	i	Recursive; see Table A.4; Element name varies in XACML v1.0 and v1.1. The <Expression> element is not used directly, but is an abstract complex type consisting of one of the subelements that comprise its substitution group. At least one of the substitution groups SHOULD be supported.
2	<AttributeSelector>	o	Global note A2	i	
2.1	RequestContextPath=	mr	Global note A2	i	
2.2	DataType=	mr	Global note A2	i	
2.3	MustBePresent=	o		i	
3	<AttributeValue>	m	Global note A1	m	
3.1	DataType=	mr	Global note A2	mr	
4	<Function>	m	Global note A1	m	

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
4.1	FunctionId=	mr	Global note A2	mr	
5	<VariableReference>	m	Base classification “c ⁸ ” as per [XACML] Section 5 requirements upgraded to “m” per [XACML] Section 10 schema	i	
5.1	VariableId=	mr	Global note A2	i	
6	<ActionAttributeDesignator>	m	Global note A1	m	
6.1	AttributeId=	mr	Global note A2	mr	
6.2	DataType=	mr	Global note A2	mr	
6.3	Issuer=	o		o	Need to determine if the issue is sufficient to differentiate attributes
6.4	MustBePresent=	o		m	
7	<ResourceAttributeDesignator>	m	Global note A1	m	
7.1	AttributeId=	mr	Global note A2	mr	
7.2	DataType=	mr	Global note A2	mr	
7.3	Issuer=	o		o	Need to determine if the issue is sufficient to differentiate attributes
7.4	MustBePresent=	o		m	

⁸ If <VariableDefinition> is supported, then “mr,” otherwise “o.”

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
8	<SubjectAttributeDesignator>	o		m	Difficult to specify the options for section 8 to extreme detail relevant to the role attribute without excluding future support for other subject attributes that do not represent roles.
8.1	AttributeId=	mr	Global note A2	mr	
8.2	DataType=	mr	Global note A2	mr	
8.3	Issuer=	o		o	Need to determine if the issue is sufficient to differentiate attributes
8.4	MustBePresent=	o		m	
8.5	SubjectCategory=	m	Default is specified.	m	Support for multiple subject categories enables brokered trust
9	<EnvironmentAttributeDesignator>	m	Global note A1	i	
9.1	AttributeId=	mr	Global note A2	i	
9.2	DataType=	mr	Global note A2	i	
9.3	Issuer=	o		i	Need to determine if the issue is sufficient to differentiate attributes
9.4	MustBePresent=	o		i	

Table A.5 <Obligations>

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	Set of <Obligation>	mr	Global note A2	i	
1.1	ObligationId=	mr	Global note A2	i	
1.2	FulfillOn=	mr	Global note A2	i	
1.3	<AttributeAssignment>	o		i	At least one value required in XACML v1.0 and v1.1; relaxed by errata 001.
1.3.1	AttributId=	mr	Global note A2	i	

Table A.6 <Request> and <Response>

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	<Request>				
1.1	Set of <Subject>	mr ⁹	Global note A3	i	
1.1.1	SubjectCategory=	o		i	
1.1.2	Set of <Attribute>	m	Global note A1	i	See Table A.8 & Table A.8.1/1
1.2	Set of <Resource>	mr ⁹	Global note A3	i	
1.2.1	<ResourceContent>	o		i	Arbitrary content

⁹ At least one instance of the tag shall be present.

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1.2.2	Set of <Attribute>	m	Global note A1	i	See Table A.8 & Table A.8.1/2
1.3	<Action>	mr	Global note A3	i	
1.3.1	Set of <Attribute>	m	Global note A1	i	See Table A.8 & Table A.8.1/3
1.4	<Environment>	mr	Global note A3	i	"o" in XACML v1.0 and v1.1
1.4.1	Set of <Attribute>	m	Global note A1	i	See Table A.8 & Table A.8.1/4
2	<Response>				
2.1	Set of <Result>	mr ⁹	Global note A3	i	
2.1.1	ResourceId=	o		i	
2.1.2	<Decision>	mr	Global note A3	i	
2.1.2.1	Permit	m	Global notes A3 and A4	i	
2.1.2.2	Deny	m	Global notes A3 and A4	i	
2.1.2.3	Indeterminate	m	Global notes A3 and A4	i	
2.1.2.4	NotApplicable	m	Global notes A3 and A4	i	
2.1.3	<Status>	m	Global note A1	i	"mr" in XACML v1.0 and v1.1

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
2.1.3.1	<StatusCode>	mr	Global notes A3 Note that [XACML] clause 10.2.4 of schema states that support is optional ("o"), but that seems to conflict with "[Required]" in 6.12.	i	
2.1.3.1.1	Value=	mr	Global note A3	i	See Table A.11
2.1.3.1.2	Set of <StatusCode>	o		i	Recursive; see table A.6/2.1.3.1; Value= attribute contains minor status codes; see Table A.11
2.1.3.2	<StatusMessage>	o		i	a string
2.1.3.3	<StatusDetail>	o		i	arbitrary XML content possibly containing the listed sub-elements
2.1.3.3.1	Set of <MissingAttributeDetail>	m	Global note A1	i	Global note C1; a set of <Attribute> is used instead.
2.1.3.3.1.1	AttributeValue=	m	Global note A1	i	
2.1.3.3.1.2	<AttributeId>	mr	Global note A3	i	
2.1.3.3.1.3	<DataType>	mr	Global note A3	i	
2.1.3.3.1.4	Issuer=	o		i	
2.1.4	<Obligations>	o		i	See Table A.5

Table A.7 Data-types

No change from the base requirements of XACML.

Table A.8 Attribute Syntax, <Attribute>

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	AttributeId=	mr	Global note A2	i	
2	DataType=	mr	Global note A2	i	
3	Issuer=	o		i	Need to determine if the issuer is sufficient to differentiate attributes
4	MustBePresent=	o		i	
5	Set of < AttributeValue>	mr	Global note A3	mr	

Table A.8.1 Attribute-types

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	Subject Attributes				
1.1	subject-id	o		c	
1.2	subject-category	o		m	
1.3	subject-id-qualifier	o		o	
1.4	key-info	o		o	
1.5	authentication-time	o		o	
1.6	authentication-method	o		m	
1.7	request-time	o		m	

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1.8	session-start-time	o		m	
1.9	ip-address	o		o	
1.10	dns-name	o		o	
1.11	userPassword	o		x	
2	Resource Attributes				
2.1	resource-id	m	Global note A1	m	
2.2	target-namespace	o		m	
2.3	resource-location	o		i	Note that this element is cited in [XACML] Section 10 schema, but otherwise is not defined in the specification.
2.4	simple-file-name	o	Although not defined in [XACML] Appendix B, this element is still cited as "o" in [XACML] Section 10 schema	o	Note C2 (was "o," see schema)
2.5	resource-content	i	Global note A6	i	Note C2 (was also absent from v1.1 schema)
2.6	xpath	i	Global note A6	i	Note C2 (was also absent from v1.1 schema)
2.7	ufs-path	i	Global note A6	i	Note C2 (was also absent from v1.1 schema)

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
2.8	scope	i	Global note A6	i	Note C2 (was "o," see schema)
2.8.1	Immediate	i	Global note A6	i	
2.8.2	Children	i	Global note A6	i	
2.8.3	Descendants	i	Global note A6	i	
3	Action Attributes				
3.1	action-id	o		m	"m" in XACML schema v1.1
3.1.1	implied-action	m		m	"m" in XACML schema v1.1
3.2	action-namespace	o		m	
4	Environment Attributes				
4.1	current-time	m	Global note A1	m	
4.2	current-date	m	Global note A1	m	
4.3	current-dateTime	m	Global note A1	m	
5	Extension Attributes				
5.1	role	o	This is specified as MAY in clause 6.2 of [RBAC].	m	Subject attribute added by [RBAC] May also be used as a resource identifier in future REA profiles
5.2	hasPrivilegesOfRole	o	This is specified as MAY in clause 6.4 of [RBAC].	l	Action attribute added by [RBAC]
5.3	enableRole	o	This is specified as MAY in clause 6.4 of [RBAC].	i	SHOULD be used in future REA profiles
5.4	(any)	o		o	

Table A.9 Functions

No change from the base requirements of XACML.

Table A.10 Access Subject Categories

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	access-subject	m	Base classification as per [XACML] appendix B requirements; because this is the default subject category, it must be implicitly supported. See also global note A1.	m	
2	recipient-subject	o		o	
3	intermediary-subject	o		c	
4	codebase	o		o	
5	requesting-machine	o		o	
6	Extensions				
6.1	role-enablement-authority	o	This is specified as MAY in clause 6.3 of [RBAC].	o	Added by [RBAC] Absent from the CD RBAC specification for XACML v1.0 and v1.1.
6.2	(any)	o		o	

Table A.11 Status Codes

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	ok	m	Global note A1	m	
2	missing-attribute	m	Global note A1	c	
3	syntax-error	m	Global note A1	m	
4	processing-error	m	Global note A1	m	
5	Extensions				
5.1	(any)	o		o	

Table A.12 Combining Algorithms

No change from base requirements defined in XACML.

Table A.13 Functional Requirements

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1	Policy Enforcement Requirements	m	Base classification as per XACML Section 7 requirements	m	
1.1	base PEP	o		x	
1.2	deny-biased PEP	o		mr	

Ref	XML Element	Base Standard Usage	Base Standard Rationale	Profile Usage	Reference and Remarks
1.3	permit-biased PEP	o		x	
2	Multi-valued Attributes	m	As per XACML clause 7.2.3	m	
3	Nested PolicySets	m		c	See Table A.2/7
4	Remote PolicySet References	m		m	See Table A.2/8
5	Remote Policy References	m		m	See Table A.2/10
6	Access Control Models				
6.1	Mandatory Access Control Policies (MAC)	o		o	
6.1.1	Basic MAC	o		o	
6.1.2	Partition Rules Based Access Control (PRBAC)	o		o	
6.2	Discretionary Access Control Policies (DAC)	m	Global note A5	m	
6.2.1	Identity-based Policies (IBAC/LBAC)	o		l	
6.2.2	Role-based Policies (RBAC)	m	Global note A5	m	See Annex B.
6.2.3	Local Rules Based Access Control (LRBAC)	o		l	

Table A.14 Obligation Types

No change on base requirements defined in XACML.

ANNEX B: ANNEX ON ROLE-BASED ACCESS CONTROL

This annex describes how to represent RBAC policies within the XACML framework by following the approach specified in core and hierarchical RBAC profile of XACML 2.0, dated 1 February 2005. This approach anticipates future adoption of XACML 2.0 and therefore facilitates the transition to XACML 2.0 compliant policies. Further, this approach supports the decision to restrict the current scope of this profile to Core RBAC support without precluding future enhancements from addressing Multi-Role Permissions, Role Enablement, and Hierarchical RBAC support.

In addition, this annex is structured in a modular format to allow for future enhancements to provisional guidance on policy definitions in the following scenarios:

- Multi-Role Permissions - The expression of policies where a user must hold several roles simultaneously in or to gain access to certain permissions.
- Role Enablement – The assignment of various role attributes to subjects and the enabling of those attributes within a session
- Hierarchical RBAC support – The ability to define inheritance relations between roles.

However, RBAC (when operating independently) limits the indirection that can be expressed in a policy to one attribute type only, namely roles. It precludes the consideration of any other quality or characteristic that can be ascribed to subjects, including identifiers. An XACML system considers these characteristics directly, rather than attempting to falsely represent them as roles.

All roles are assumed to be static.

B.1 Relationship with DCID 6/3

Under typical implementations, users and permissions are grouped by roles, and those role-permission sets determine user accesses. Also, RBAC implementations inherently can only provide some level of Discretionary Access Control (DAC). DCID 6/3 defines DAC as a means of restricting access to objects (e.g., files, data entities) based on the identity and need-to-know of subjects (e.g., users, processes) and/or groups to which the object belongs. Mandatory Access Control (MAC), on the other hand, is defined within DCID 6/3 as a means of restricting access to objects based on the sensitivity of the information contained in the objects and the formal authorization (i.e., clearance, formal access approvals, and need-to-know) of subjects to access information of such sensitivity.

RBAC will govern Discretionary Access Control (DAC) decisions for actions that can be performed on one or more resources. With DAC, the owner of a resource controls access to a resource by restricting a subject's access to a resource. When DAC permissions are managed correctly, only those subjects specified by the resource owner may perform some combination of actions on a resource (e.g. invoke a web service operation).

Although, provision of MAC is beyond the scope of the profile, it is compatible with the XACML approach to RBAC defined in this profile.

B.2 Relationship to Core RBAC

For access control of Web Service invocations, we hereby describe a simple RBAC policy model that is consistent with the Core RBAC Reference Model defined in the ANSI standard.

Figure B.1 illustrates Core RBAC as represented in the *NCES Security Service Design Specification*. [ANSI-RBAC] describes the necessities for implementing the elements of the RBAC model. Essentially, RBAC is best described as a policy model for access control in which permitted actions on resources are identified via roles rather than individual identities. In this model: 1) subjects are assigned roles, 2) subject roles are defined according to job functions within the context of an organization 3) permissions are the ability to perform some action on a resource, possibly under certain specified conditions, 4) permissions are granted strictly based on rules that use role values, and 5) rules are evaluated using the assigned roles of the subject attempting access. In addition, RBAC introduces the notion of a session within a computing environment into the policy model. Through this, RBAC acknowledges that the assignment of roles to subjects is distinct from the activation of selected roles for a subject's session from that subject's assigned set of roles. Therefore, it is more appropriate to recognize that authorization policies should be defined in terms of a subject's activated roles. This profile specifies how to express these RBAC concepts as an XACML compliant construct. NCES and the XACML Technical Committee (TC) have adopted slightly different terms for key RBAC Concepts. In particular,

- Resource = Object
- Action = Operation
- Subject = User

The terms selected to represent key RBAC concepts as illustrated below.

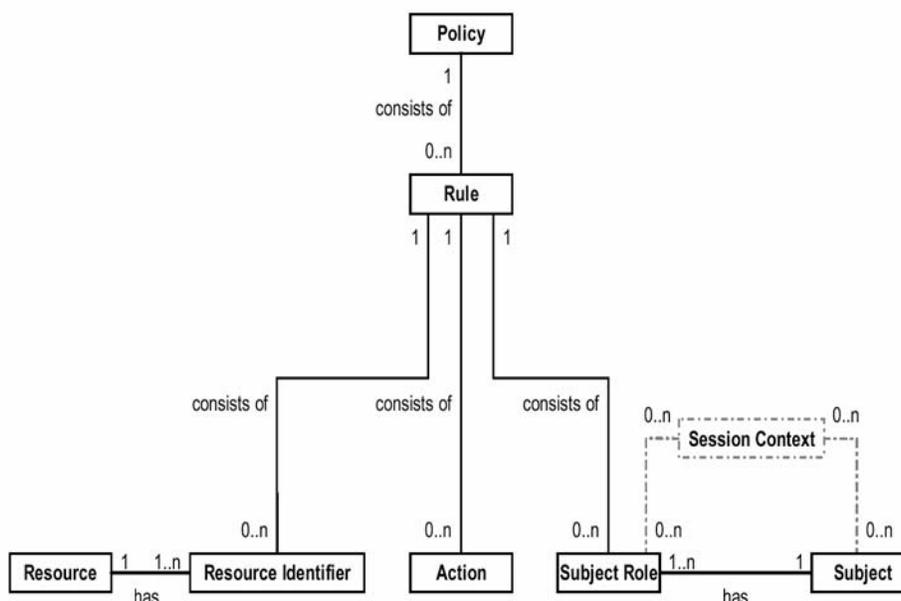


Figure B.1 – Core RBAC Policy Model

Therefore, there are two types of policies outlined in this profile:

- Permission Policies – Policy Sets to manage the permissions associated with a particular role
- Role Policies – Policy Sets to manage the assignment of roles to a particular set of subjects

Figure B.2 depicts the specific application of this architecture concept to XACML RBAC policies. Each request will identify the user requesting access as the subject of the request and provide a `<SubjectAttribute>` with the unique name from the PKI certificate used for authentication. The PEP must represent the resource requested and the action requested for that resource to the PDP. The PDP requests subject attributes which match the unique name to capture the activated roles assigned to the subject. The PDP retrieves policies according to the resource and action requested. This will result in the evaluation of a Policy Set containing both a Role `<PolicySet>` and a Permission `<PolicySet>`. The PDP reaches a decision according to the evaluation of the `<Target>` and `<Rules>` expressed in the Role Policy Set (and corresponding Permission `<PolicySet>` as per the requirements of this profile). Finally, this decision is returned to the PEP to enforce.

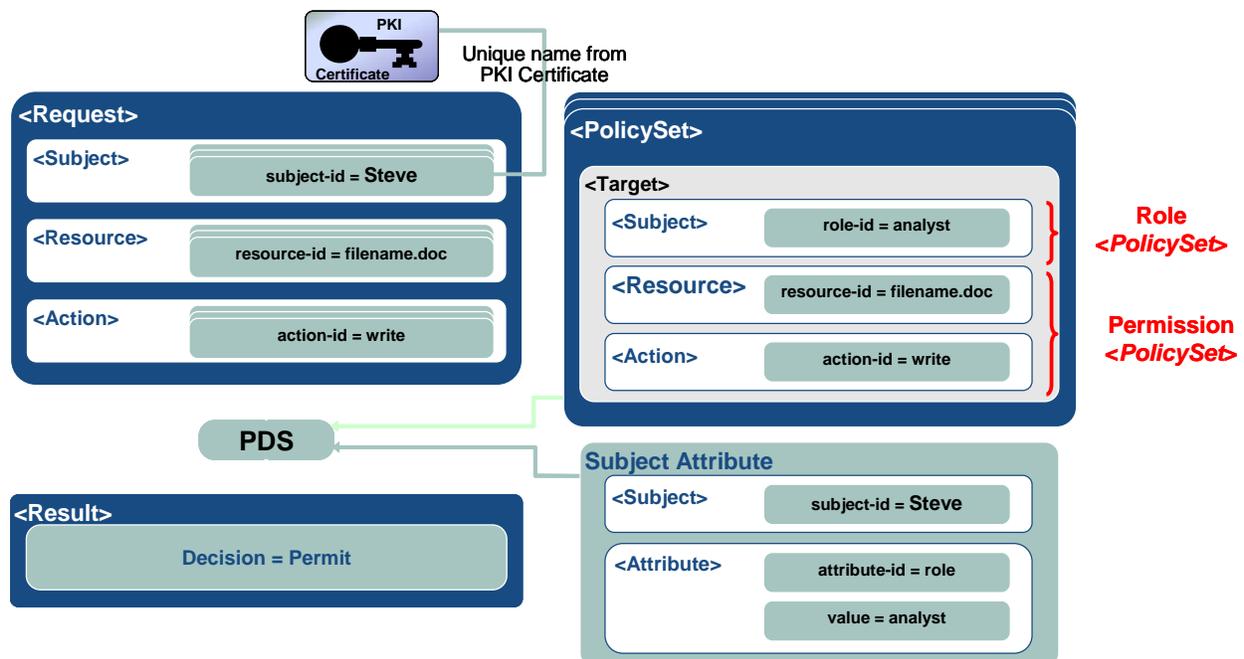


Figure B.2 – Evaluation of RBAC Policies expressed as XACML

This profile **REQUIRES** that permissions associated with a given role be expressed within a `<PolicySet>`. The Permission `<PolicySet>` contains `<Policy>` elements and `<Rules>` that describe the resource actions that subjects are permitted to access.

B.3 Identity and Attributes

Within the context of authorization and access control policy, it is important to understand the difference between identity and identifier. The traditional definition of

identity is the set of attribute values (i.e., characteristics) by which an entity is recognizable and that, within the scope of an identity manager's responsibility, is sufficient to distinguish that entity from any other entity and to distinguish the identity from any other identity. An attribute is a quality or characteristic inherent in or ascribed to an entity. As entities, both subjects and resources have attributes. Consequently, identifier is defined as a sufficiently unambiguous reference to the identity of a subject (e.g., name, employee number) or resource (e.g., file name, fully qualified distinguished name), controlled by an identity manager.

A number of potential attribute types can be implemented in an ABAC model in addition to traditional functional roles. These include attributes based on organization, hierarchical clearances, nonhierarchical clearances, national affiliation, communities of interest, and so forth. The attribute-based access (ABAC) model is a holistic approach that allows the system to take all, or any subset of, attributes into consideration when vetting a request for access to the system.

B.4 Roles and Groups

Groups are a collection of subjects whereas roles are representations of particular job functions. Therefore, RBAC allows for indirect privilege management that aligns privileges with job functions. It is a very useful policy model when controlling access to shared resources. Further, roles are associated with privileges, and have some associated semantics regarding the authority and responsibility. A role is assigned to a subject, whereas in groups, subjects are added.

This model forces thinking of role as an attribute and therefore introduces some subject-role assignment responsibility. However, any other relevant subject attributes must be discovered and considered prior to adding subject to an ACL. Therefore, RBAC cannot satisfy the unanticipated consumer. Further, RBAC is not useful when access to resources must be controlled based on individual subjects rather than job functions.

An attribute-based approach does not require specific role definitions; instead, policies are based on binary conditions associated with the attributes in an individual's profile, the system resource constraints, and environmental conditions (such as threat levels, network conditions, network security classifications, etc.). In addition to user attributes, ABAC allows for resource and environmental attributes to be included in the user profile and, thereby, provides greater functionality, flexibility, extensibility, and scalability, especially in the context of the NCES.

B.5 Expressing RBAC with XACML Constructs

In this profile, a role attribute MAY be associated with any of the categories of subjects involved in making an access request. Roles MUST be expressed as one or more Subject Attributes. In the RBAC model, permissions are assigned solely on the basis of business functions of the requestor. Even with common representation for roles, the set of valid roles remains very policy-domain specific. Therefore this profile does not attempt to define any standard set of roles.

B.5.1 Roles

XACML supports two different expressions of roles as Subject Attributes. The first expresses roles with a small set of attributes; the name of each attribute within the set indicates "role" semantics. The values of each attribute indicate the name of the role

held. The second method expresses roles as individual attributes, with the attribute value remaining empty or containing various parameters for the roles.

This profile RECOMMENDS that the use of a common AttributeId value of role. Figure B.3 illustrates this AttributeId definition as part of an XACML policy.

```
- <Target>
- <Subjects>
- <Subject>
- <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
  <AttributeValue
    DataType="http://www.w3.org/2001/XMLSchema#string">urn:nces:Administrator</AttributeValue>
  <SubjectAttributeDesignator DataType="http://www.w3.org/2001/XMLSchema#string" AttributeId="role" />
  </SubjectMatch>
</Subject>
</Subjects>
</Target>
```

Figure B.3 – Example of AttributeId role

It is RECOMMENDED that roles be defined as values for subject attributes assigned this attribute identifier. This profile RECOMMENDS that potential attribute values for be expressed as fully qualified URIs. Figure B.4 illustrates this expression of attribute values as part of an XACML policy. Definition of a minimum set of values for the role attribute is out of scope for this profile. However, policy domains SHOULD agree upon and publish a unique set of values for this attribute.

```
- <Target>
- <Subjects>
- <Subject>
- <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
  <AttributeValue
    DataType="http://www.w3.org/2001/XMLSchema#string">urn:nces:Administrator</AttributeValue>
  <SubjectAttributeDesignator DataType="http://www.w3.org/2001/XMLSchema#string" AttributeId="role" />
  </SubjectMatch>
</Subject>
</Subjects>
</Target>
```

Figure B.4 – Example of a Role Expressed as a Fully Qualified URI Value

B.5.1.1 Roles as Operational Functions

Role definitions are based on operational function. However, different organizations focus on different missions or sets of information. Therefore, their business functions, and their access control policies are defined differently. RBAC policies are defined according to the roles defined by a particular organization. Therefore, when attempting to access resources across organizational boundaries, the access control mechanisms as well as the role definitions must align across organizations. For example, several commonly defined roles are: Administrator, Civil Servant, and Contractor. However, each of these functions will likely have an agency-specific meaning, or may allow access to features of a system such as write access, modify privilege, OS access levels, or ability to access acquisition sensitive files.

B.5.1.2 Global definitions of roles

The concept of promoting a common schema for roles or attributes is rooted in the logic that centralization is the only way to ensure true interoperability. To add to the complexity, the introduction of multiple agency partners suggests that there may be multiple identity stores that

are authoritative. Within these multiple identity stores, some stores will be authoritative for a certain set of attributes or roles, while other stores will be authoritative for a different set of roles and attributes. The retrieval of attributes becomes an issue because the system will need to retrieve attributes from their authoritative sources. There are several outstanding issues related to obtaining role or attribute information from multiple directory sources. There is a need to determine which data sources are authoritative across the enterprise. Establishing new authoritative data sources will introduce additional challenges.

B.5.2 Objects

Objects SHOULD be expressed as XACML *Resources*.

B.5.3 Operations

Operations SHOULD be expressed as XACML *Actions*.

B.5.4 Permissions

Permissions SHOULD be expressed with XACML Policies which comply with the rules contained Section 5.1 of this profile.

B.5.5 Multi-Role Permissions

In multi-role permissions a user must simultaneously hold more than one role in order to gain access to all or part of the system. Under the XACML Profile, it is possible to express policies in order to satisfy these conditions. For example, changing financial information for a program may require that the user <Subject> performing the action have both the employee and the project manager role. These policies may be expressed using a Role <PolicySet> where the <Target> element requires the <Subject> to have all necessary role attributes. This can be accomplished using a single <Subject> element containing a set of <SubjectMatch> elements, one for each role. The associated permission <PolicySet> should specify the permissions associated with Subjects who simultaneously have all the specified roles enabled.

B.6 Transitioning from Roles to Attributes

As defined by the ANSI RBAC Standard, a role is a job function within the context of an organization, with some associated semantics regarding the authority and responsibility conferred on the subjects to whom the role is assigned. Therefore, under traditional RBAC, subjects are assigned roles and rules are evaluated using the assigned roles of the subject attempting access.

In comparison, an attribute-based policy model formulates the rules which govern access through an extensible notion of subject, resource, and other attributes. This model treats the concept of a role as any other subject attribute. Therefore, subject roles are understood from the identifiers, groups, roles, and any number of other attributes associated with a particular subject. This approach does not build a role as a predicate over an attribute or over an aggregated set of attributes that a user exhibits in relation to his assigned business functions. Rather, this approach assigns attributes to users to characterize the business functions assigned to them. The rules defined in a policy contain the predicates to be evaluated in order to grant access.

When transitioning from roles to attributes, the existing investment in RBAC models can be leveraged and made more extensible through the attribute model by building and extending the facets of the existing roles in the XACML profile as attributes of the <Subject>, <Resource>, and <Environment>.

B.7 Attribute-Based Access Control and the Limitations of RBAC

RBAC is a system in which users are granted access to resources based on their individual roles. Traditionally, roles are grouped in a hierarchical model in which some roles contain greater access levels than do other roles (Parent-Child). These roles can be assigned to individuals or groups, but generally are not always flexible enough to accommodate every role without developing an extremely complex RBAC rule set and may require a large number of roles to be defined up front.

Using the XACML framework, an ABAC model can be implemented to determine access. The ABAC model is a holistic approach that allows the system to take all, or any subset of, attributes into consideration when vetting a request for access to the system. The ABAC approach does not require specific role definitions; instead, policies are based on binary conditions associated with the attributes in an individual's profiles, the system resource constraints, and environmental conditions (such as threat levels, network conditions, network security classifications, etc.).

When basing system authorization on this holistic model, the attributes that compose the individual's XACML profile become the description for that principal. These attributes can include all the aforementioned RBAC role types listed above as well as additional characteristics such as National Affiliation and Community of Interest Affiliation. However, the extensibility of the XACML profile allows us to grow beyond that. XACML can leverage existing security models and be used to enhance said models to add multiple levels of security controls.

Users will be enabled to have multiple roles, as opposed to single roles prescribed under a traditional RBAC model, without having to manage discrete accounts assigned to each role. An XACML profile can contain attributes for multiple roles across multiple systems. This model also provides the requisite level of flexibility and scalability to support a system-of-systems implementation and is extremely useful in an extended enterprise, a cross-domain solution, or a federated system architecture.

An inherent limitation in RBAC is the single dimension of roles. Finer grained access control policies often involve multiple user and resource attributes. As the number of attributes grows, the number of discreet roles and permissions needed to encode these attributes will grow exponentially, thereby making access control difficult to manage under traditional RBAC implementations. Further, RBAC requires centralized management and is not well suited for a highly distributed environment, such as NCES, especially when the user and resources are transacting across multiple security domains. By implementing XACML-based RBAC, resource and environmental attributes can extend the profile and allow for broader security implementations, such as MAC, to be enabled to complement RBAC.

The ABAC model, brings many advantages over traditional identity- or role-based models and allows for more flexible and more robust representation of complex, fine-grained access control semantics. That is especially suitable for the dynamic Service Oriented Architectures and Web services environment envisioned under the NCES. Management of security information is spread over a number of Attribute and Policy Authorities and can be extended across organizational boundaries. That is particularly suitable for the large-scale, cross-domain information sharing required under the NCES. ABAC also reduces the overall system complexity, allowing different system components (user directory, service registry, policy server, etc.) to be focused on, and optimized for, their respective administrative tasks. To realize the full potential of the ABAC, the entire attribute management process needs to be considered, such as attribute provisioning, binding, discovery, auditing, and reporting.

B.8 Tenets of Attribute-Based Access Control

Attribute-Based Access Control (ABAC) is best described as a policy model that allows for authorization policy applicability and the associated rules that govern access to be formulated based on an extensible notion of subject, resource, and other attributes. ABAC is essentially defined by three basic tenets:

I. An extensible notion of subject attributes encompassing identifiers, groups, roles, and any number of additional subject attribute types.

This is necessary to encompass all relevant characteristics of subjects, not only identifiers and roles. As a result, ABAC policy rules can encompass both Identity-Based Access Control (IBAC) and RBAC rules, whereas RBAC rules cannot encompass IBAC rules. Moreover, as potential characteristics are not limited to identity and role, ABAC surpasses the capabilities of RBAC. The extensible nature of ABAC allows it to potentially encompass any policy rule.

II. The use of attributes in policy rules where attributes are compared with fixed values or with each other, in accordance with the appropriate security business logic(s).

It is important to understand the significance of permitting the comparison of attributes. In essence, attributes in policy rules are variables. As was discussed, IBAC and RBAC both enable the comparison of the current subject's identifier(s) or role(s) to predefined or fixed values such as an ACL, or a defined list of role values.

In addition to enabling the comparison of subject attributes to fixed values, ABAC allows for comparison between subject and resource attributes. This essentially permits the use of variables on both sides of the rule and allows for the incorporation of virtually any resource metadata elements into policy rules.

III. The use of resource attributes when specifying the applicability of a policy.

In addition to policy rules, every policy must specify to which action(s) and resource(s) it is applicable. In earlier policy examples, the applicability of the policy was [ACTION (Read) on RESOURCE (File #7)], meaning the policy is only applicable when a subject is attempting to read a specific file (i.e., File #7). In the previous policy example, the applicability of the policy was [ACTION (Destroy) on RESOURCE (Any)], which means the policy is applicable when a subject is attempting to destroy any resource. In this example, the wildcard keyword "any" takes the place of a specific resource identifier. This "one or all" approach for specifying policy applicability is limited. Under ABAC, policies can be defined that are applicable to classes of resources versus individually named resources.

B.9 Scope Considerations

The following items are considered out of scope for this Annex:

- This profile does not address static or dynamic "Separation of Duty." Policies conforming to this profile do not address roles that are enabled dynamically based on the resource or actions a subject is attempting to perform.
- As discussed in Section B.5, this profile does not attempt to define any standard set of roles.