# SAML and XACML Overview

Prepared by
Abbie Barbir, abbieb@nortel.com
Nortel
Canada
April 25, 2006

# Acknowledgements

Some slides are provided by

> Eve Maler, Sun Microsystems

> Hal Lockhart, BEA

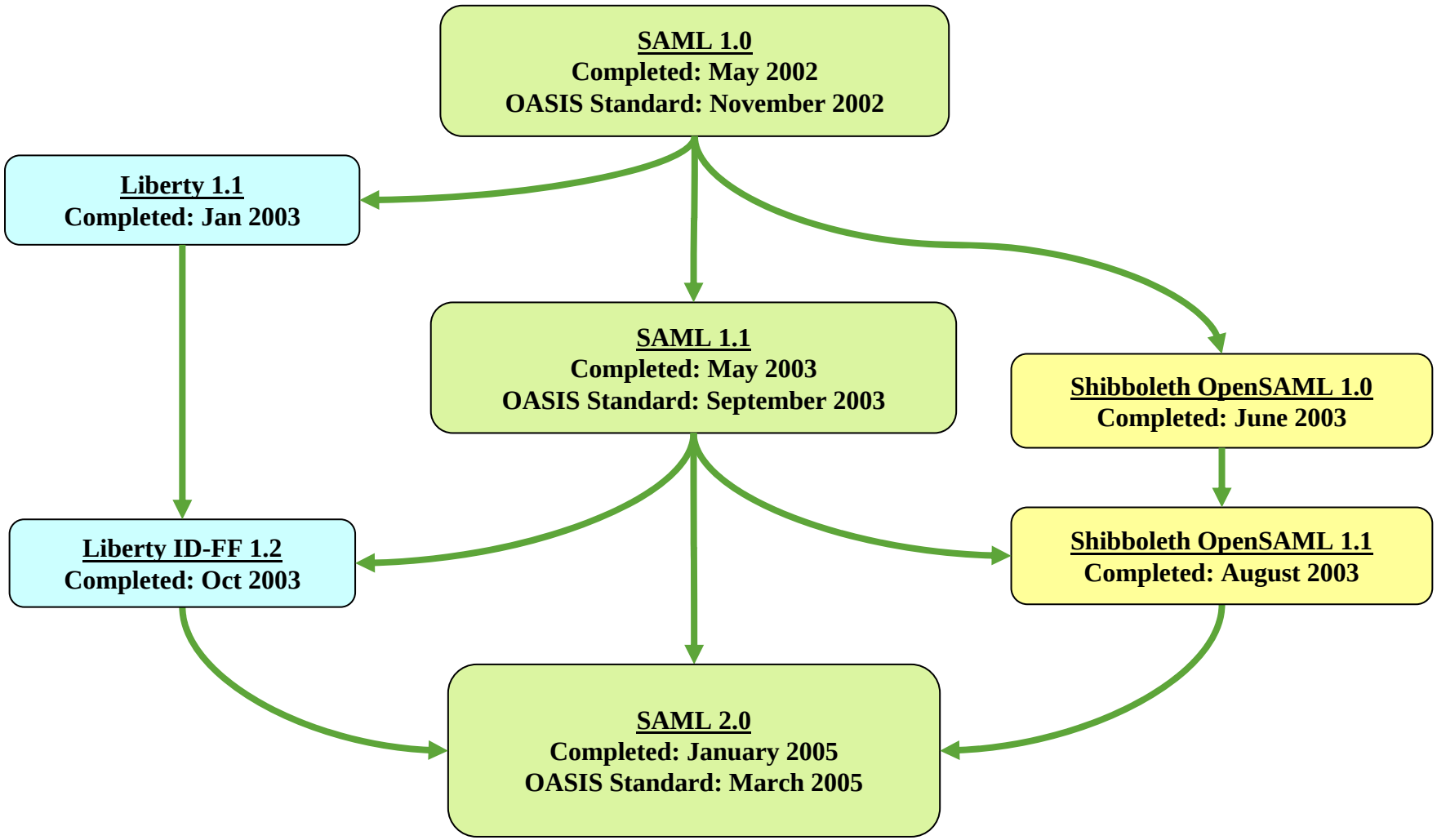
# Agenda

- SAML History and Overview
- SAML 2.0 Features
- Status in ITU-T
- XACML History and Overview
- XACML 2.0 Features
- Status in ITU-T

# SAML Overview and History

- **SAML: Security Assertion Markup Language**
  - A framework for the exchange of security-related information between trusting parties
  - The key standard for federated identity systems
  - Supports many real-world business scenarios
  - Widely used today for cross-domain single sign-on
- **OASIS Security Services Technical Committee (SSTC)**
  - SSTC manages SAML development

# SAML Timeline

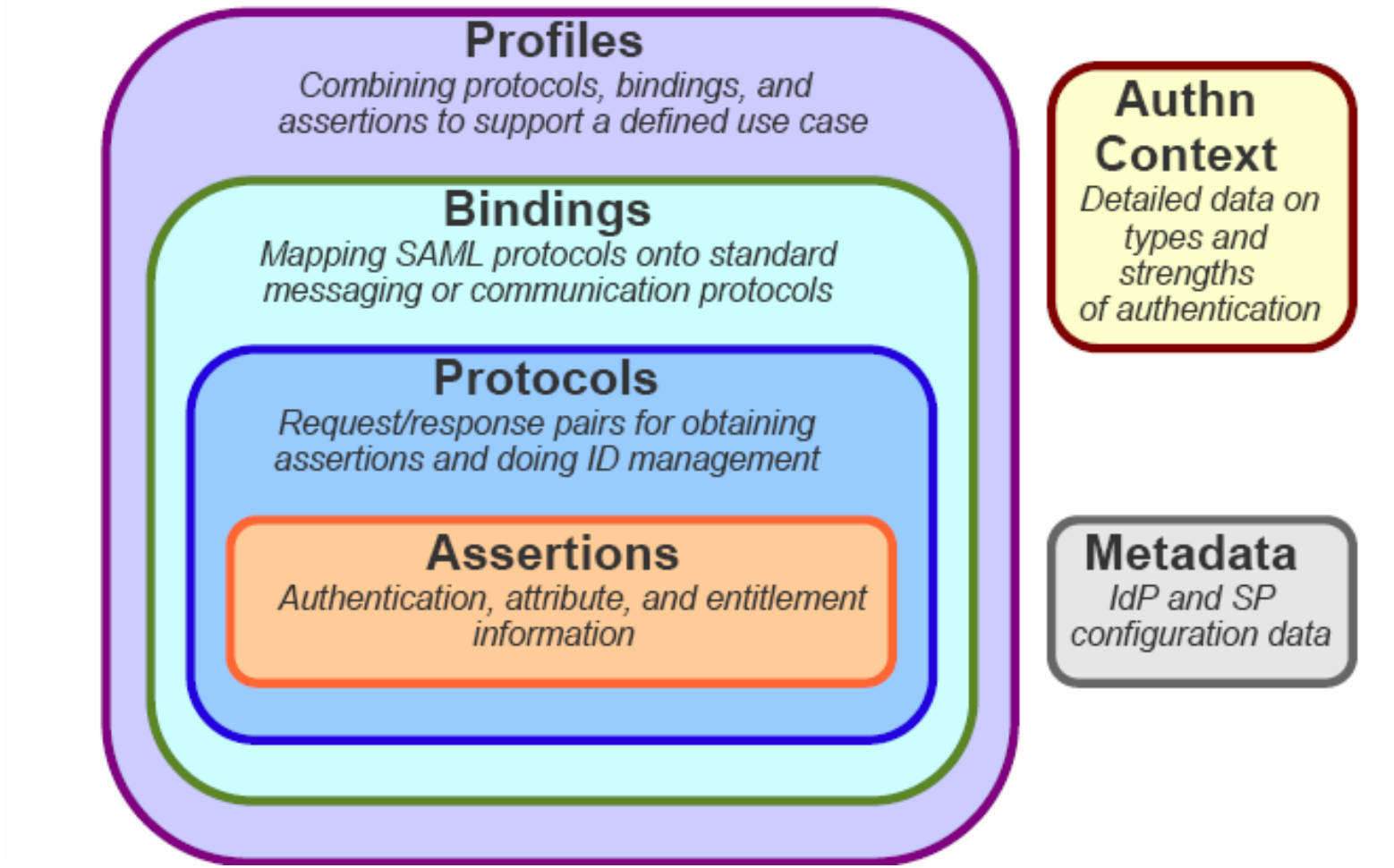


# SAML 2.0 Specification Suite

- **Conformance Requirements**
  - Required “Operational Modes” for SAML implementations
- **Assertions and Protocols**
  - The “Core” specification
- **Bindings**
  - Maps SAML messages onto common communications protocols
- **Profiles**
  - “How-to’s” for using SAML to solve specific business problems

- **Metadata**
  - Configuration data for establishing agreements between SAML entities
- **Authentication Context**
  - Detailed descriptions of user authentication mechanisms
- **Security and Privacy Considerations**
  - Security and privacy analysis of SAML 2.0
- **Glossary**
  - Terms used in SAML 2.0

# SAML Concepts



# Terms and concepts 1

## Subjects

- **Entity** (system entity): An active element in computer/network system
- **Principal**: An entity whose identity can be authenticated
- **Subject:** A principal in the context of a security domain

## Identities

- **Identity**: The essence of an entity, often described by one's characteristics, traits, and preferences
  - **Anonymity**: Having an identity that is unknown or concealed
- **Identifier**: A data object that uniquely refers to a particular entity
  - **Pseudonym**: A privacy-preserving identifier
- **Federated identity**: Existence of an agreement between providers on a set of identifiers and/or attributes to use to refer to a principal
  - **Account linkage**: Relating a principal's accounts at two different providers so that they can communicate about the principal

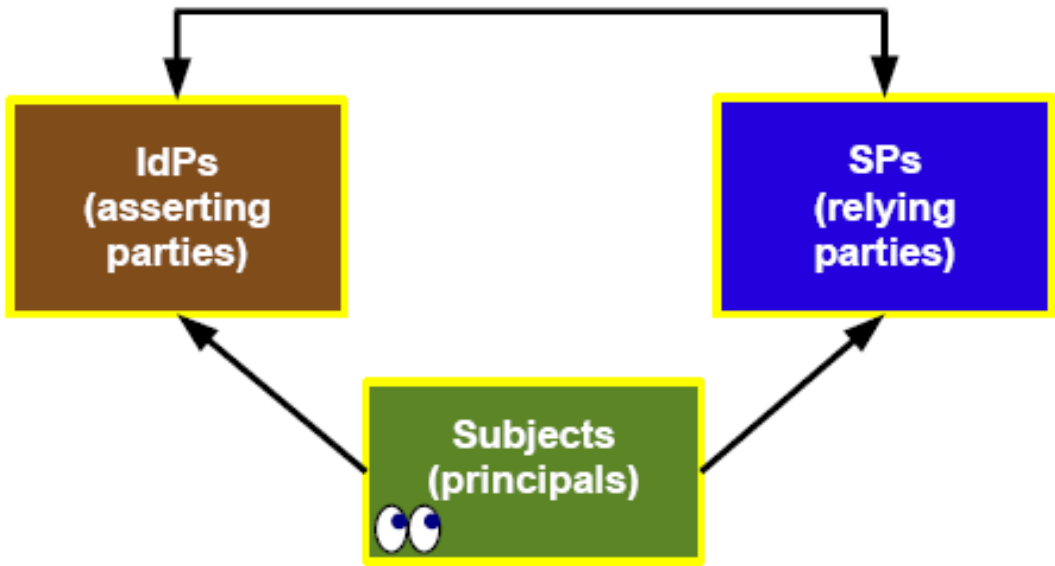
# Terms and concepts 2

## More Entities

- **Asserting party (SAML authority):** An entity that produces SAML assertions
  - **Identity provider**: An entity that creates, maintains, and manages identity information for principals and provides principal authentication to other service providers
- **Relying party:** An entity that decides to take an action based on information from another system entity
  - **Service provider**: An entity that provides services to principals or other entities

# How these entities interrelate

- Most of the SAML and ID-FF use cases are eyeballoriented
- But some backchannel (SOAP and other) communication takes place in service of this



# SAML assertions

- **Assertion is a declarations of fact**
  - **according to someone**
- **SAML assertions contain one or more “statement” about “subject” (human or program):**
  - Authentication statement: “Joe authenticated with a password at 9:00am”
  - Attribute statement (which itself can contain multiple attributes)
    - Joe is a manager with a $500 spending limit
  - Authorization decision statement (now deprecated)
  - You can extend SAML to make your own kinds of assertions and statements
- Assertions can be digitally signed

# Example: Common Assertion Portions

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   Version="2.0“ IssueInstant="2005-01-31T12:00:00Z">
   <saml:Issuer>
         www.acompany.com
   </saml:Issuer>
   <saml:Subject>
         <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
               format:emailAddress">
               j.doe@company.com
         </saml:NameID>
   </saml:Subject>
   <saml:Conditions
         NotBefore="2005-01-31T12:00:00Z"
         NotOnOrAfter="2005-01-31T12:00:00Z">
   </saml:Conditions>
... statements go here ...
</saml:Assertion>
```

# Example: Authentication Statement

```
<saml:Assertion ... common info goes here ... >
   ... and here ...
<saml: AuthnStatement
   AuthnInstant="2005-01-31T12:00:00Z"
   SessionIndex="67775277772">
   <saml:AuthnContext>
         <saml:AuthnContextClassRef>
               urn:oasis:names:tc:SAML:2.0:ac:classes:
               PasswordProtectedTransport
         </saml:AuthnContextClassRef>
   </saml:AuthnContext>
</saml:AuthnStatement>
```

# Authentication context classes

- ✓ Internet Protocol
- ✓ Internet Protocol Password
- ✓ Kerberos
- ✓ Mobile One Factor Unregistered
- ✓ Mobile Two Factor Unregistered
- ✓ Mobile One Factor Contract
- ✓ Mobile Two Factor Contract
- ✓ Password
- ✓ Password Protected Transport
- ✓ Previous Session
- ✓ Public Key – X.509
- ✓ Public Key – PGP
- ✓ Public Key – SPKI
- ✓ Public Key – XML Signature
- ✓ Smartcard
- ✓ Smartcard PKI
- ✓ Software PKI
- ✓ Telephony
- ✓ Nomadic Telephony
- ✓ Personalized Telephony
- ✓ Authenticated Telephony
- ✓ Secure Remote Password
- ✓ SSL/TLS Cert-Based Client Authn
- ✓ Time Sync Token
- ✓ Unspecified

# Example of an attribute statement

```
<saml:Assertion ... common info goes here ... >
   ... and here ...
   <saml:AttributeStatement>
         <saml:Attribute NameFormat="http://smithco.com">
               Name="PaidStatus"
               <saml:AttributeValue> PaidUp </saml:AttributeValue>
         </saml:Attribute>
         <saml:Attribute NameFormat="http://smithco.com">
               Name="CreditLimit"
               <saml:AttributeValue xsi:type="smithco:type">
                     <smithco:amount currency="USD">
                           500.00
                     </my:amount>
               </saml:AttributeValue>
         </saml:Attribute>
   </saml:AttributeStatement>
</saml:Assertion>
```

# Artifacts

- A small, fixed-size, structured data object pointing to a typically larger, variably sized SAML protocol message
  - can be embedded in URLs / conveyed in HTTP messages
- Allows for “pulling” SAML messages as opposed to “push”
- SAML defines one artifact format but you can roll your own

# Protocols

- **Assertion query and request**
  - Query for assertion based on simple reference, subject-matching, or statement type
- **Authentication request**
  - SP requests a fresh authn assertion that adheres to various requirements (specified by means of Authentication Context)
- **Artifact resolution (“meta-protocol”)**
  - Dereferences an artifact to get a protocol message
- **Name identifier management**
  - IdPs and SPs inform each other of changes to their mutual understanding of what a principal’s name is
- **Name identifier mapping**
  - Privacy-preserving way for two SPs to refer to the same principal
- **Single logout**
  - Signals to all SPs using the same session to drop the session

# Bindings

- **SOAP**
  - Basic way for IdPs and SPs to send SAML protocol messages
- **Reverse SOAP (PAOS)**
  - Multi-stage SOAP/HTTP exchange that allows an HTTP client to send an HTTP request containing a SOAP response
- **HTTP redirect**
  - Method to send SAML messages by means of HTTP 302
- **HTTP POST**
  - Method to send SAML messages in base64-encoded HTML form control
- **HTTP artifact**
  - Way to transport an artifact using HTTP in two ways
    - URL query string and
    - HTML form control
- **URI**
  - How to retrieve a SAML message by resolving a URI

# Profiles

- **Web browser SSO**
  - SSO using standard browsers to multiple SPs: profiles Authn Request protocol and HTTP Redirect, POST, and artifact bindings
- **Enhanced client and proxy (ECP)**
  - SSO using ECPs: profiles Authn Request protocol and SOAP and PAOS bindings
- **IdP discovery**
  - One way for SPs to learn the IdPs used by a principal
- **Single logout**
- **Name identifier management**
  - Profiles the NIM protocol with SOAP, HTTP redirect, HTTP POST, and HTTP artifact bindings
- **Artifact resolution**
- **Assertion query/request**

# SAML Status in ITU-T

> Currently X.websec-1

> In Q9/17

> Text is stable and reviewed

# Agenda

> XACML History and Overview

> XACML 2.0 Features

> Status in ITU-T

# XACML History

- First Meeting – 21 May 2001
- Requirements from: Healthcare, DRM, Registry, Financial, Online Web, XML Docs, Fed Gov, Workflow, Java, Policy Analysis, WebDAV
- XACML 1.0 - OASIS Standard – 6 February 2003
- XACML 1.1 – Committee Specification – 7 August 2003
- XACML 2.0 – OASIS Standard – 1 February 2005
- **XACML TC Charter**
  - Define a core XML schema for representing authorization and entitlement policies
  - Target - any object - referenced using XML
  - Fine grained control, characteristics - access requestor, protocol, classes of activities, and content introspection
  - Consistent with and building upon SAML

**Technologies and procedures intended to implement organizational policy in spite of human efforts to the contrary**

# XACML Objectives

- Ability to locate policies in distributed environment
- Ability to federate administration of policies about the same resource
- Base decisions on wide range of inputs
  - Multiple subjects, resource properties
- Decision expressions of unlimited complexity
- Ability to do policy-based delegation
- Usable in many different environments
  - Types of Resources, Subjects, Actions
  - Policy location and combination

**Policy Examples**

- “Primary physician can have any of her patients’ medical records sent to a specialist in the same practice.”
- “Salespeople can create orders, but if the total cost is greater that $1M, a supervisor must approve”

# General Characteristics

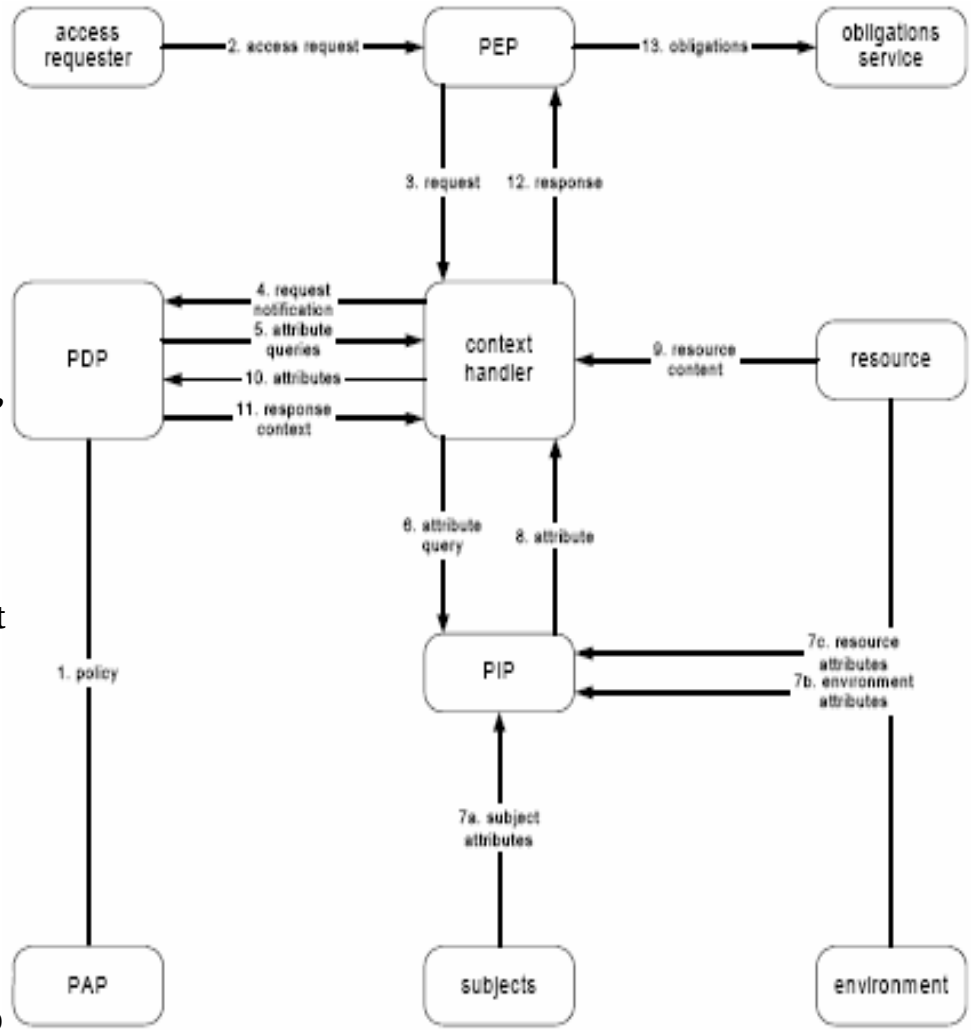
- Defined using XML Schema
- Strongly typed language
- Extensible in multiple dimensions
- Borrows from many other specifications
- Features requiring XPath are optional
- Obligation feature optional
- Language is very “wordy”
  - Many long URLs
- Expect it to be generated by programs
- Complex enough that there is more than one way to do most things

# Generic RBAC functionality

- **RBE (Rule Based Engine):** Central policy decision point,
- **PEP (Policy Enforcement Point):** Resource specific authorization decision request/response handling and policy defined obligations execution,
- **PAP (Policy Authority Point)** or Policy DB:  policy storage (distributed)
- **PIP (Policy Information Point):** Supply external policy context and attributes to RBE: subject credentials and attributes verification
- **RIP (Resource Information Point):** Provides resource context.
- **AA (Attribute Authority):** Manages user attributes

# XACML Data Flow Model

1. **PAP: policies/sets → PDP**
2. **Access Requestor sends request to PEP**
3. **PEP sends request to context handler in its native request format, optionally including attributes of the subjects, resource, action and environment**
4. **Context handler constructs an XACML request context and sends it to the PDP.**
5. **PDP requests any additional subject, resource, action and environment attributes from the context handler**
6. **Context handler requests attributes from PIP**
7. **PIP obtains the requested attributes.**
8. **PIP returns requested attributes to the context handler**
9. **Optionally, the context handler includes the resource in the context**
10. **Context handler sends requested attributes and (optionally) the resource to the PDP. PDP evaluates the policy**
11. **PDP returns response context (including the authorization decision) to the context handler.**
12. **Context handler translates response context to the native response format of the PEP. Context handler returns the response to the PEP.**
13. **PEP fulfills the obligations.**



# Novel XACML Features

- Large Scale Environment
  - Subjects, Resources, Attributes, etc. not necessarily exist or be known at Policy Creation time
  - Multiple Administrators - potentially conflicting policy results
  - Combining algorithms
- Request centric
  - Use any information available at access request time
  - Zero, one or more Subjects
  - No invented concepts (privilege, role, etc.)
- Dynamically bound to request
  - Not limited to Resource binding
  - Only tell what policies apply in context of Request

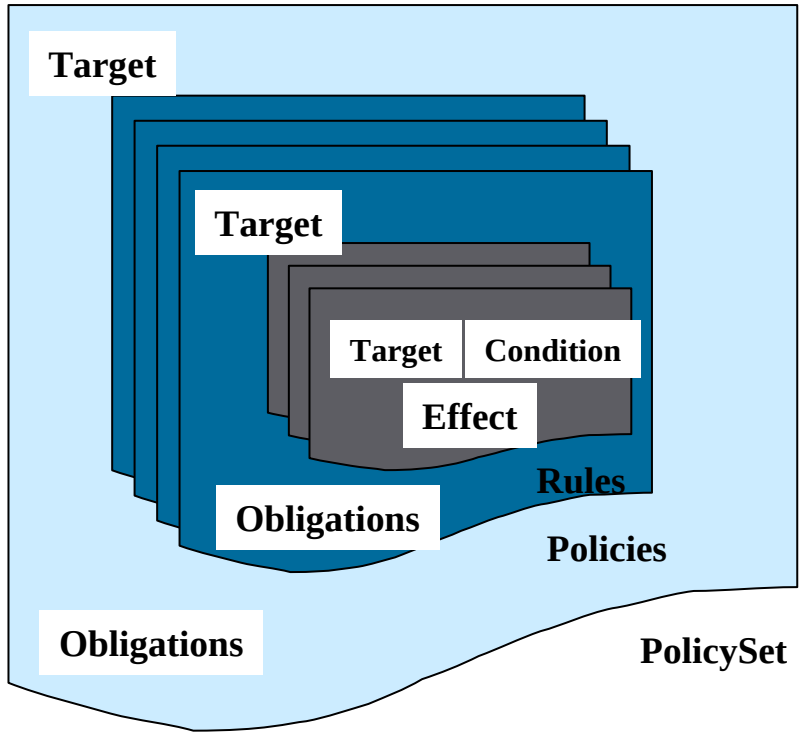
# XACML Concepts 1

- Policy & PolicySet – combining of applicable policies using CombiningAlgorithm
- Target – Rapidly index to find applicable Policies or Rules
- Conditions – Complex boolean expression with many operands, arithmetic & string functions
- Effect – “Permit” or “Deny”
- Obligations – Other required actions
- Request and Response Contexts – Input and Output
- Bag – unordered list which may contain duplicates

# XACML Concepts 2

**Rule**

- Smallest unit of administration, cannot be evaluated alone
- Elements
  - Description – documentation
  - Target – select applicable policies
  - Condition – boolean decision function
  - Effect – either “Permit” or “Deny”
- Results
  - If condition is true, return Effect value
  - If not, return NotApplicable
  - If error or missing data return Indeterminate
    - Plus status code




**Target**

- Find policies that apply to a request
- Enables dynamic binding
- Allow complex Conditions
- Attributes of Subjects, Resources, Actions and Environments
- Matches against value, using match function
  - Regular expression
  - RFC822 (email) name
  - X.500 name
  - User defined
- Attributes specified by Id or XPath expression
- Normally use Subject or Resource, not both

**Condition**

- Boolean function to decide if Effect applies
- Inputs come from Request Context
- Values can be primitive, complex or bags
- Can be specified by id or XPath expression
- Fourteen primitive types
- Rich array of typed functions defined
- Functions for dealing with bags
- Allowed to quit when result is known
- Side effects not permitted

# Data types and Functions

**Data Types**

- From XML Schema
  - String, boolean
  - Integer, double
  - Time, date
  - dateTime
  - anyURI
  - hexBinary
  - base64Binary
- From Xquery (Stand alone now)
  - dayTimeDuration
  - yearMonthDuration
- Unique to XACML
  - rfc822Name
  - x500Name

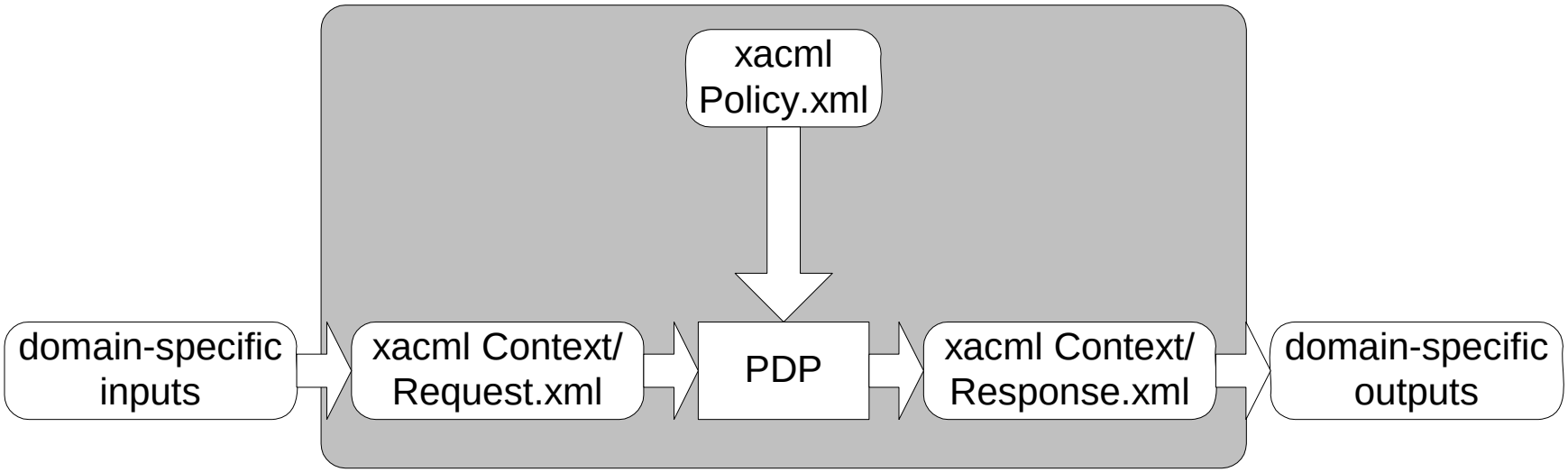
**Functions**

- Equality predicates
- Arithmetic functions
- String conversion functions
- Numeric type conversion functions
- Logical functions
- Arithmetic comparison functions
- Date and time arithmetic functions
- Non-numeric comparison functions
- Bag functions
- Set functions
- Higher-order bag functions
- Special match functions
- XPath-based functions
- Extension functions and primitive types

# Policies and Policy Sets

- Policy
  - Smallest element PDP can evaluate
  - Contains: Description, Defaults, Target, Rules, Obligations, Rule Combining Algorithm
- Policy Set
  - Allows Policies and Policy Sets to be combined
  - Use not required
  - Contains: Description, Defaults, Target, Policies, Policy Sets, Policy References, Policy Set References, Obligations, Policy Combining Algorithm
- Combining Algorithms: Deny-overrides, Permit-overrides, First-applicable, Only-one-applicable

# Request and Response Context



# Request and Response Context

- Request Context
  - Attributes of:
    - Subjects – requester, intermediary, recipient, etc.
    - Resource – name, can be hierarchical
    - Resource Content – specific to resource type, e.g. XML document
    - Action – e.g. Read
    - Environment – other, e.g. time of request
- Response Context
  - Resource ID
  - Decision
  - Status (error values)
  - Obligations

# XACML Core Specification 1

- Develops policy expression for generic RBAC used by PDP
  - Define a simple Request/Response messages format.
- Defines policy format for access control based on “Subject-Resource-Action” triad attributes.
  - Defines format for policy and request/response messages.
- Decision request sent in a message provides context for policy-based decision.
- Complete policy applicable to a particular decision request can be composed of a number of individual rules or policies
- Policies can be combined to form a single policy applicable to the request.

# XACML Core Specification 2

- Defines three top-level policy elements:
- <Rule>, <Policy> and <PolicySet>
- <Rule>
  - The <Rule> element contains a Boolean expression that can be evaluated in isolation
    - Not intended to be accessed in isolation by a PDP.
    - Not intended to form the basis of an authorization decision on its own
    - Exist in isolation only within an XACML PAP
  - May form the basic unit of management
    - Can be re-used in multiple policies.
- The <Policy> element contains a set of <Rule> elements and a particular procedure for combining the results of their evaluation.
- Basic unit of policy used by the PDP
  - Form the basis of an authorization decision

# XACML Core Specification 3

- <PolicySet> element contains a set of <Policy> or other <PolicySet> elements
  - Contains a specified procedure for combining the results of their evaluation
    - Standard means for combining separate policies into a single combined policy
    - Defines Rule and Policy combining algorithms that describe procedures for arriving at an authorization decision based on results of evaluation of a set of rules or policies:
      - Deny-overrides,
      - Permit-overrides,
      - First applicable,
      - Only-one-applicable

# XACML Core Specification 4

- Authorization decision, requires that the attributes of many different types to be compared or computed
  - XACML includes a number of built-in functions and a method of adding non-standard functions
  - Functions may be nested to build arbitrarily complex expressions
- Achieved with the <Apply> element.
  - Has an XML attribute called FunctionId
    - Identifies function to be applied to element  contents
    - Each standard function is defined for specific argument data-type combinations, (return data-type specified)

# XACML Profiles

- Digital Signature
  - Integrity protection of Policies
- Hierarchical Resources
  - Using XACML to protect files, directory entries, web pages
- Privacy
  - Determine “purpose” of access
- RBAC
  - Support ANSI RBAC Profile with XACML
- SAML Integration
  - XACML-based decision request
  - Fetch applicable policies
  - Attribute alignment

# XACML Uptake

- Three open source implementations available
  - See OASIS website
- Product Statements
  - Astrogrid, BEA Systems, CapeClear, CA, Entrust, IBM, Jericho, Layer 7, Parthenon Computing, PSS Systems, Starbourne, Sun Microsystems, Xtradyne
- Standards references
  - OASIS ebXML reference implementation
  - Open GIS Consortium
  - XRI Data Interchange – interest
  - UDDI – interest
  - Global Grid Forum – joint work
  - PRISM (Publication Metatadata) – interest
  - ASTM – Healthcare Informatics PMI

# XACML Status in ITU-T

- Currently X.websec-2
- In Q9/17
- Text is stable and reviewed