Introduction to XACML

Session RTA13

Tim Hahn – IBM Tivoli Security

Tivoli. software

e business software

Intent of this Session

- Provide an introduction to XACML for RACF administrators
- To help in understanding XACML, show how several RACF constructs could be expressed using XACML

What are We going to talk about?

- XACML Introduction
- RACF Review
- RACF Resources and Permissions in XACML
- Beyond this Introduction

XACML Introduction

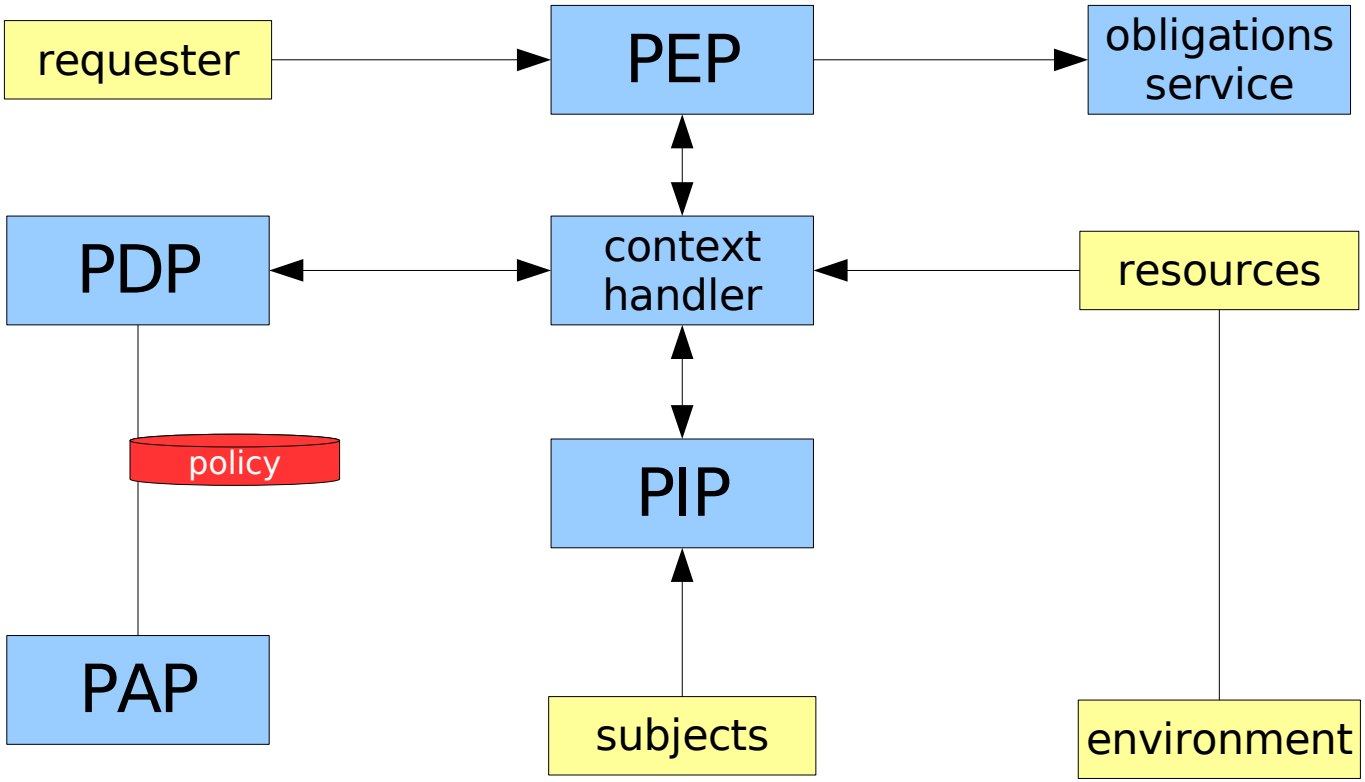
What is XACML?

- eXtensible Access Control Markup Language
- A XML-based markup language for specifying access policy
- An XACML document represents a tree of policy specifications
- The tree/hierarchy structure is used to determine whether a policy or rule is applicable
- Includes an expression language
- allows complex calculations to be used to determine if a policy or rule applies
- Calculations are based on user information, environment information, and resource information
  - All information is treated as attributes

Terminology

- Policy Enforcement Point (PEP) – the system entity that performs access control
- Policy Decision Point (PDP) – the system entity that evaluates applicable policy and renders an authorization decision
- Policy Information Point (PIP) – the system entity that acts as a source of attribute values
- Policy Administration Point (PAP) – the system entity that creates a policy or policy set

The Big Picture



Where could XACML be used?

- Between PAP and PDP
  - a format for expressing policy information
  - could be in administration requests/responses
  - could be in persistent form
- As a policy storage format
  - an access control database could be stored in XACML form
- As a policy interchange format
  - two communicating policy decision points may exchange policy information using XACML
  - a request may contain XACML policy information as a part of the handling policy for the information in the request

XACML Request/Response Contexts

- A formulation of information sent between PEP and PDP
- A Request Contains
  - Subject
    - Attributes
  - Resource
    - Attributes
  - Action
    - Attributes
  - Environment (attributes)
    - Attributes
- A Response Contains
  - Result (one or more), each of which contain
    - Decision
    - Status
    - Obligations

An XACML document – the Policy

- An XML document written according to the XACML XML schema
- Expresses a set of access control policies (i.e. rules)
- The data model representing an XACML document defines
  - PolicySet
  - Policy
  - Rule
  - Target
  - Obligation
  - Condition
  - Effect

PolicySet

- Represents a set of policy statements
- Top level structure in an XACML document
- Model:
  - Recursive (may contain additional PolicySets within it)
  - contains 0 or more Policies
  - contains a single Target
  - contains 0 or more Obligations
  - contains a Policy/PolicySet Combining Algorithm
- Example

```
<PolicySet ...>
  <Description>This is a Policy Set</Description>
  <Target>
  ...
  </Target>
  <Policy ...>
  ...
  </Policy>
</PolicySet>
```

Policy

- Represents a policy statement which is a set of rules
- A Target may be specified by the policy or by the rules
- Model:

  contains a single Target

  contains 0 or more Rules

  contains 0 or more Obligations

  contains a Rule Combining Algorithm

- Example

```
<Policy ...>
  <Description>This is a Policy</Description>
  <Target>
  ...
  </Target>
  <Rule ... Effect=”Permit”>
  ...
  </Rule>
</Policy>
```

Rule and Effect

- Represents a rule for defining who can access what
- Must be contained within a Policy

  A Rule only has applicability based on the Target specification
- Effect is an attribute of the Rule
- Model:

  contains an Effect

  contains a 0 or 1 Condition

  contains 0 or 1 Target
- Example

```
<Rule ... Effect=”Deny”>
  <Description>Disallow access for the target if the condition is true</Description>
  <Target>
  ...
  </Target>
  <Condition>
  ...
  </Condition>
</Rule>
```

Target

- Represents a combination of subject, resource, action and attributes to which a rule applies
- May be specified at the PolicySet, Policy, or Rule level

  Target is what determines if a PolicySet, Policy or Rule is applicable!
- Model:

  contains 0 or more Subjects

  contains 0 or more Resources

  contains 0 or more Actions

  contains 0 or more Attributes (Environment data)
- Example

```
<Target>
  <Description>Defines a subject/resource/action combination</Description>
  <Resources>
  ... myFavoriteResourceName ...
  </Resources>
  <Actions>
  ... read ...
  </Actions>
</Target>
```

Obligations and Conditions

- Obligations are additional operations which the PEP should perform when enforcing the authorization decision
  - contain a FulfillOn statement which must be fulfilled by the PEP
  - Optional AttributeAssignment arguments to be interpreted by the PEP
  - Obligations are expected to be fulfilled by the PEP – and if unable to do so, the PEP is expected to “Deny” access
- Conditions are additional computations used to determine whether a rule applies
  - must evaluate to True, False, or Indeterminate
  - If present, a Rule will only have its Effect if the Condition evaluates to True
  - False or Indeterminate result in the Rule being marked either NotApplicable or Indeterminate

Extensibility

- the XACML language allows for very complex policy expressions to be written
- Attributes, Data-types, and Functions are defined using URI identifiers

  These are used in declaring Subjects, Targets, Resources, Environment, Conditions, and Obligations

  new types and functions may be added without requiring an update to the XACML schema
- Over 200 Functions are already listed in the mandatory-to-implement list

Review

- XACML is used or can be used in a number of ways
  - as a policy storage format
  - as a policy interchange format
- XACML documents have a structure
  - PolicySet
  - Policy
  - Rules
- A PDP receives a request from a PEP containing Subject, Resource, Action, and Environment data
- A PDP uses the PolicySet and based on matching to the supplied information, produces a Result for the PEP
- The result contains a Decision (Permit, Deny, Indeterminate, or NotApplicable)

RACF Review

RACF Entities

- Users
  - Information about users on the system
- Groups
  - collections of users
- Resources
  - represent things to which access controls can be applied

RACF Users

- Information about a user on the system
- Multiple pieces of information, organized into segments
- Information examples: userid, omvs UID, TSO Initial program, region size
- Attributes represent extra capabilities
- Attribute examples: SPECIAL, AUDITOR, OPERATIONS

RACF Groups

- collections of users defined to RACF
- can be structured into a group hierarchy using SUPGROUP during group creation
- Users are CONNECTed to groups to become members of those groups
- Group level user attributes: SPECIAL, AUDITOR, OPERATIONS

  Allows users to perform various administration work but only to a subset of the entities defined in the RACF database

RACF Resources

- represent things to which access controls can be applied
- resource profiles represent resources on the system

  two types: dataset and general
- Access is granted to resources by setting up resource profiles and PERMITing access to those resource profiles
- An attribute of a resource profile is the UACC – universal access. UACC applies when no PERMIT specification applies
- Resource profile names may contain wildcards

  known as generics

  %, *, **
- For any given resource, multiple access rules (resource profiles) may apply to the resource

Access to Resources in RACF

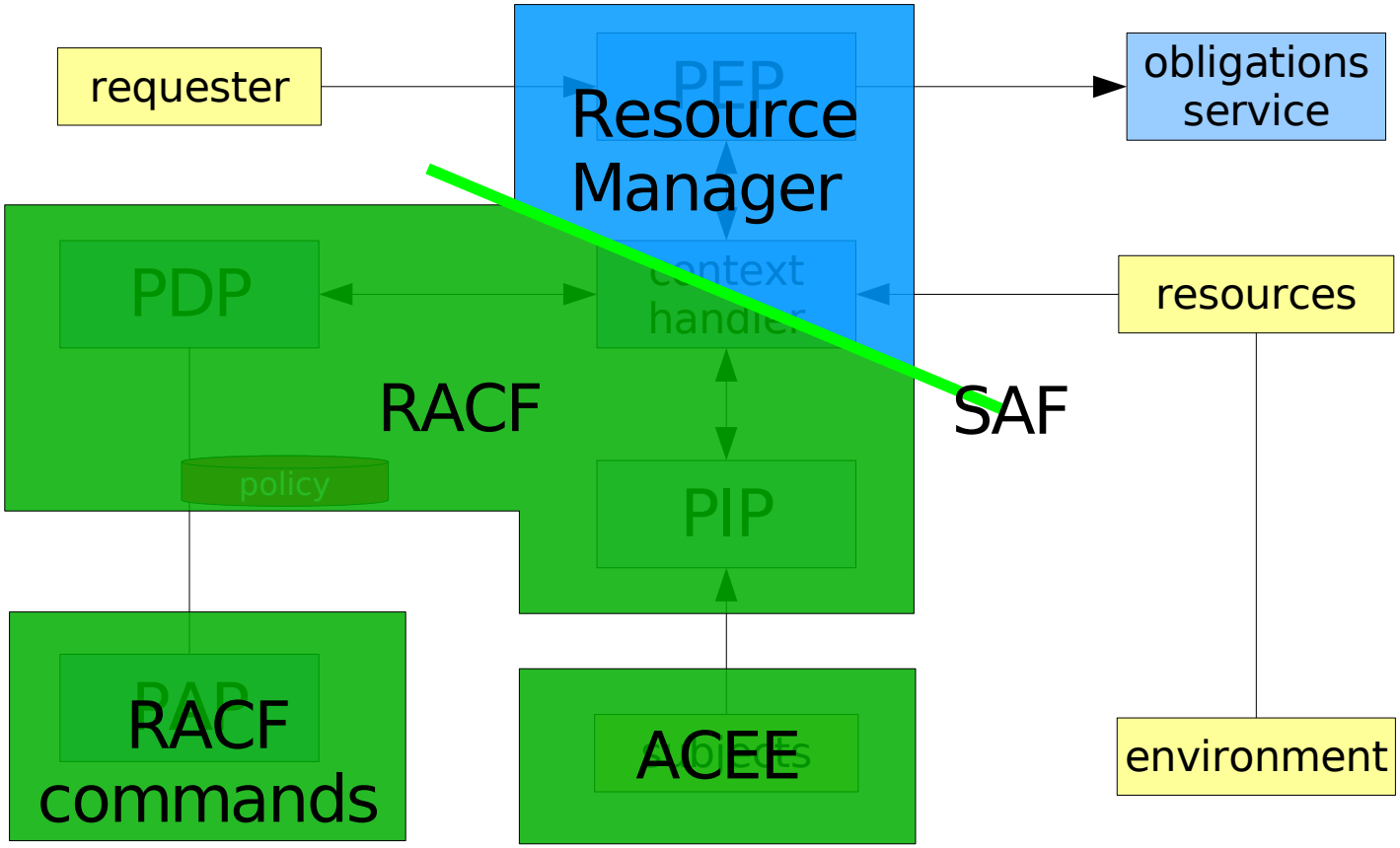
- A user is granted access to a resource by several means
  - A user may be the resource owner
  - A user may be granted access to a resource directly
  - A user may obtain access to the resource by being in a group which has been granted access to the resource
  - A user may obtain access based on settings in RACF's Global Access Checking Table
  - access may be conditional on several factors, including being based on what program is being used
- If the user is neither a resource owner or a member of a group which is the resource owner, then resource profiles are used to determine access

RACF Access Checking based on Resource Profiles

- When access to a resource (dataset or general resource) is attempted
  - RACF finds resource profiles that may apply
  - RACF orders resource profiles from most-specific to least-specific (relative to the resource in question) to determine the resource profile which best applies
  - RACF then evaluates the user's access to the resource based on this resource profile
- A user's granted authority to the resource must be greater than or equal to the authority required for access to be permitted
  - NONE, EXECUTE
  - READ, UPDATE, CONTROL
  - ALTER

RACF Resource and Permissions in XACML

The Big Picture – How RACF applies to the XACML-defined entities



Disclaimer

- The following does not express or imply that RACF supports the information which follows
- The alignment of RACF information with XACML constructs is purely my own fabrication

  This is done only to help in understanding XACML mark-up by showing analogies to existing constructs
- This alignment is meant to help you better understand the concepts involved in working with XACML policies
- This alignment may help XACML-knowledgeable people better understand RACF constructs
- Other alignments may be possible – there is often more than one way to express the same rules when using XACML

Aligning RACF and XACML terms

- Resources in RACF align with Resources in XACML
- Users, Groups in RACF align with Subjects in XACML
- Authority in RACF align with Actions in XACML
- Permissions in RACF align with Rules in XACML
- WHEN() type clauses align with Conditions in XACML
- RACF Dataset and General Resource profiles align with Policies (sets of Rules) in XACML
- The combined set of all defined Dataset and General Resource profiles aligns with a PolicySet in XACML

Examine a single RACF Resource Profile

- RACF General Resource Profile:

```
rlist facility (BPX.DAEMON) authuser

CLASS      NAME
-----      ----
FACILITY   BPX.DAEMON

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    IBMUSER         NONE               READ    NO

USER      ACCESS    ACCESS COUNT
----      ------    ------ -----
IBMUSER   ALTER        000000
GLDSRV    READ         000000

   ID     ACCESS  ACCESS COUNT  CLASS                    ENTITY  NAME
-------- -------  ------------ -------- ---------------------------------------
NO ENTRIES IN CONDITIONAL ACCESS LIST
```

Examine a single RACF Resource Profile

- Analogous XACML Policy:

```
<Policy PolicyId="BPX.DAEMON">
      <Description>Represents example BPX.DAEMON RACF resource profile</Description>
      <Target>
            <Resources>
                  ... <AttributeValue ...>BPX.DAEMON</AttributeValue> ...
            </Resources>
      </Target>
      <Rule Effect="Permit">
            <Target>
                  <Subjects>
                        ... <AttributeValue ...>IBMUSER</AttributeValue> ...
                  </Subjects>
                  <Actions>
                        ... <AttributeValue ...>ALTER</AttributeValue> ...
                  </Actions>
            </Target>
      </Rule>
      <Rule Effect="Permit">
            <Target>
                  <Subjects>
                        ... <AttributeValue ...>GLDSRV</AttributeValue> ...
                  </Subjects>
                  <Actions>
                        ... <AttributeValue ... >READ</AttributeValue> ...
                  </Actions>
            </Target>
      </Rule>
</Policy>
```

Understand Combining multiple Resource Profiles

- RACF SEARCH order:

```
SEARCH FILTER(FOO.**) CLASS(DATASET) GENERIC

FOO.SUB2.* (G)
FOO.*.** (G)

SEARCH FILTER(FOO.**) CLASS(DATASET)

FOO.SUB1
FOO.SUB2.FILE1
FOO.SUB3.FILE1
FOO.SUB3.FILE2
FOO.SUB4
FOO.SUB5
FOO.SUB6
```

Understand Combining multiple Resource Profiles

- Analogous XACML Combining Order:

```
<PolicySet
 PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:first-
  applicable">
    <Policy PolicyId="foo.sub3.file1"></Policy>
    <Policy PolicyId="foo.sub3.file2"></Policy>
    <Policy PolicyId="foo.sub2.file1"></Policy>
    <Policy PolicyId="foo.sub1"></Policy>
    <Policy PolicyId="foo.sub4"></Policy>
    <Policy PolicyId="foo.sub5"></Policy>
    <Policy PolicyId="foo.sub6"></Policy>
    <Policy PolicyId="foo.sub2.*"></Policy>
    <Policy PolicyId="foo.*.**"></Policy>
</PolicySet>
```

Adding Conditions to represent WHEN() constructs

- RACF Resource Profile containing a WHEN() clause:

```
rlist facility (BPX.DAEMON) authuser

CLASS      NAME
-----      ----
FACILITY   BPX.DAEMON

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    IBMUSER         NONE               READ    NO

USER      ACCESS    ACCESS COUNT
----      ------    ------ -----
IBMUSER   ALTER        000000
GLDSRV    READ         000000

   ID     ACCESS  ACCESS COUNT  CLASS                    ENTITY  NAME
-------- -------  ------------ -------- ---------------------------------------
USER1    READ        000000    CONSOLE  CONS0001
```

Adding Conditions to represent WHEN() constructs

- Analogous XACML Condition construct:

```
<Policy PolicyId="BPX.DAEMON">
      <Description>Represents example BPX.DAEMON RACF resource profile</Description>
      <Target><Resources> ... <AttributeValue ...>BPX.DAEMON</AttributeValue> ... </Resources></Target>
      <Rule Effect="Permit"> ... </Rule>
      <Rule Effect="Permit"> ... </Rule>
      <Rule Effect="Permit">
            <Target>
                  <Subjects>
                        ... <AttributeValue ...>USER1</AttributeValue> ...
                  </Subjects>
                  <Actions>
                        ... <AttributeValue ...>READ</AttributeValue> ...
                  </Actions>
            </Target>
            <Condition>
                  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal>
                  ... <EnvironmentAttributeDesignator
                        AttributeId="connectedFromConsoleName"
                        DataType="http://www.w3.org/2001/XMLSchema#string" .../>
                  ... <AttributeValue ...>CONS0001</AttributeValue> ...
                  </Apply>
            </Condition>
      </Rule
</Policy>
```

Review

- RACF Dataset and General Resource profiles can be expressed using XACML constructs
- and, appropriately constrained, vice versa
- For each resource profile
  - define a Rule
  - determine the users and groups permitted, and define Subject information
  - determine the resource name (possibly generic), and define the Resource information
  - determine any WHEN() type clauses and define Condition information
- Build a PolicySet using proper CombiningRules to match the RACF SEARCH order

Beyond this Introduction

XACML allows for many constructs

- We have only discussed a very constrained sub-set of the XACML language
- PDPs which act on arbitrary (anything expressible in XACML) XACML markup do exist
- XACML will be used used to express business-level access policy statements
- ... we have only scratched the surface of everything XACML can represent

Useful Links

- IBM Products employing XACML
  - WebSphere DataPower XS40 and XI50
    http://www.ibm.com/software/integration/datapower/xs40/features/
  - WebSphere Service Registry and Repository v6.0
    http://www.ibm.com/software/integration/wsrr/index.html
- OASIS – XACML
  - Committee page
    http://www.oasis-open.org/committees/xacml
  - eXtensible Access Control Markup Language (XACML) 2.0
    http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf
- RACF Documentation
  - RACF Security Administrator's Guide – SA22-7683
    http://publibz.boulder.ibm.com/epubs/pdf/ichza751.pdf
  - RACF Command Language Reference – SA22-7687
    http://publibz.boulder.ibm.com/epubs/pdf/ichza451.pdf
- My e-mail: mailto:hahnt@us.ibm.com