PURDUE UNIVERSITY

Department of Computer Sciences

XACML Function Annotations

Prathima Rao

Dan Lin

Elisa Bertino

XACML

- OASIS standard for specifying access control policies in enterprise systems.
  - XML based.
- Need for efficient policy management.
  - Several analysis techniques
    - Similarity analysis
      - Relation between sets of permitted(denied) requests of policies.
    - Policy Ratification[1]
      - conflict detection, dominance check, coverage check etc.

Motivation

- Policy analysis techniques often need to abstract a policy as a Boolean expression.
- Example :

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy PolicyId="Bill-Policy" RuleCombiningAlgId="permit-overrides">
<Rule RuleId="R1" Effect="Permit">
<Condition>
<Apply FunctionId="xacml:1.0:function:string-at-least-one-member-of">
        <SubjectAttributeDesignator AttributeId="E-Mail"
        Datatype="#string"/>
        <Apply FunctionId="xacml:1.0:function:string-bag">
                <AttributeValue Datatype="#string"> .gov
                </AttributeValue>
                 <AttributeValue Datatype="#string"> .edu
                </AttributeValue>
        </Apply>
</Apply>
</Condition></Rule></Policy>
```

E-Mail == .gov ||
E-Mail == .edu

Motivation

- XACML uses functions to express conditions on request attributes.
  - Not straightforward to abstract these functions into boolean expressions.
  - Need to know the behavioral semantics of these functions.
  - Standard XACML functions
    - Manually map each function to a boolean expression
  - User-defined XACML functions
    - Unknown semantics

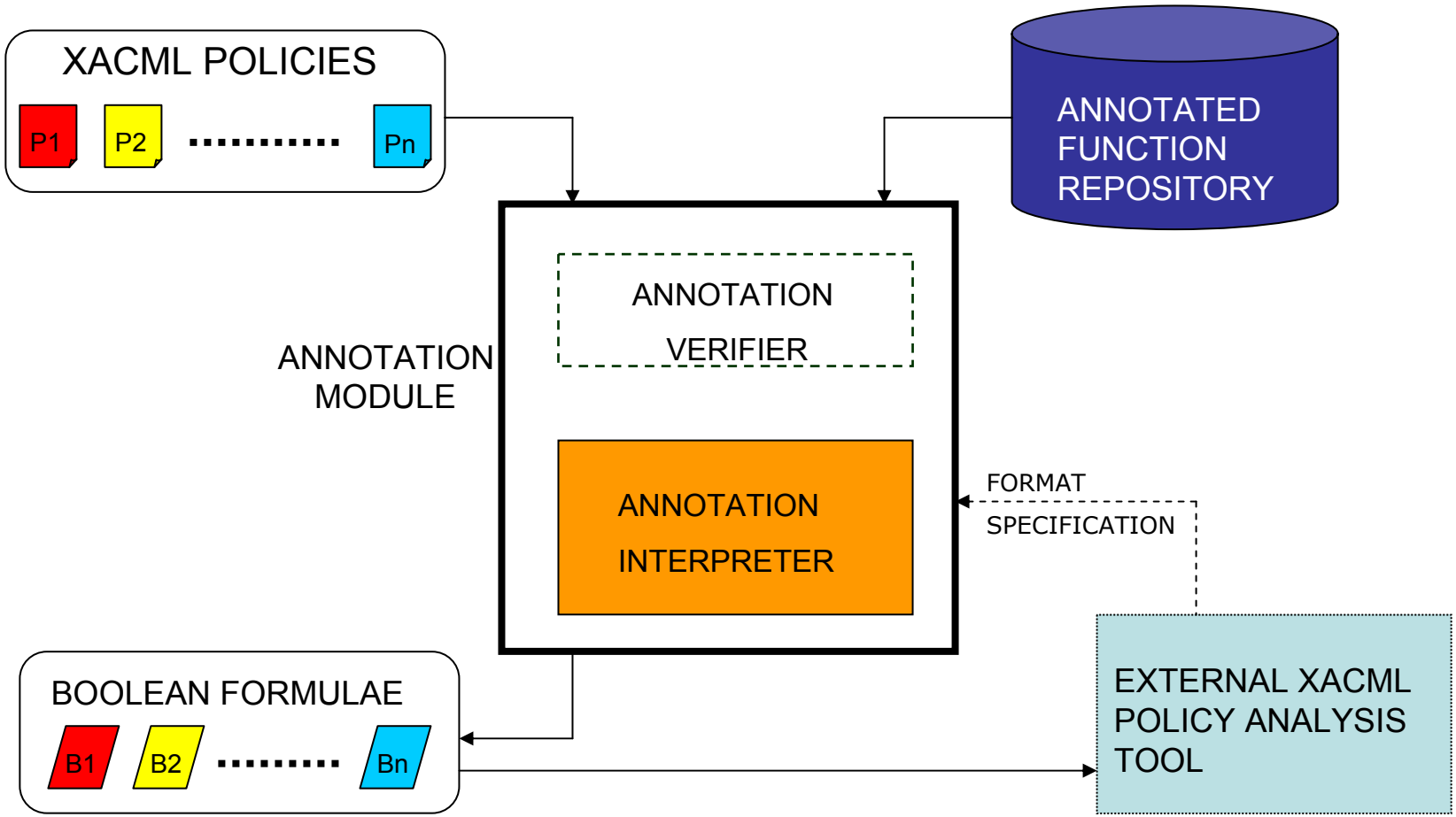
Need for a technique to explicitly convey the function semantics.

Annotation Syntax

- <Annotation></Annotation>
  - AnnotationId attribute ~ FunctionId
- <Operand></Operand>
  - Denotes function parameter
  - Datatype and Value attributes
- <Uni-Operator></Uni-Operator>
  - Unary operators -, ~ ..etc.
- <Bi-Operator></Bi-Operator>
  - Binary operators including arithmetic (+, -, %, ..), logical (||, &), set (intersect, union, belongsto..).
- <Uni-Term></Uni-Term>
  - Term with one operand
- <Bi-Term></Bi-Term>
  - Term with two operands
- <Boolean-Form></Boolean-Form>
  - Boolean formula

```
<Annotation AnnotationId =
"string-at-least-one-member-of">
<Bi-Term>
   <Operand Datatype="#string-
bag" value = "&Param_1" />
   <Bi-Operator> belongsto </Bi-
Operator>
   <Operand Datatype="#string-
bag" value = "&Param_2" />
</Bi-Term>
</Annotation>
```

Annotation Framework



Annotation Example

INPUT XACML POLICY

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy PolicyId="Bill-Policy"
RuleCombiningAlgId="permit-overrides">
<Rule RuleId="R1" Effect="Permit">
<Condition>
<Apply FunctionId="xacml:1.0:function:string-
at-least-one-member-of">
        <SubjectAttributeDesignator
AttributeId="E-Mail"
        Datatype="#string"/>
        <Apply
FunctionId="xacml:1.0:function:string-bag">
                <AttributeValue
Datatype="#string"> .gov
        </AttributeValue>
                <AttributeValue
Datatype="#string"> .edu
        </AttributeValue>
        </Apply>
</Apply>
</Condition></Rule></Policy>
```

ANNOTATION REPOSITORY

```
<Ann
"strin
<Bi-T
    <
bag"
    <
Oper
    <
bag"
</Bi
</An
```

```
<Annotation AnnotationId =
"string-at-least-one-member-of">
<Bi-Term>
   <Operand Datatype="#string-
bag" value = {"E-Mail"} />
   <Bi-Operator> belongsto </Bi-
Operator>
   <Operand Datatype="#string-
bag" value = {".gov", ".edu"} />
</Bi-Term>
</Annotation>
```

```
<Boolean-Form>
E-Mail == .gov || E-Mail == .edu
</Boolean-Form>
```

Annotation Consistency Verification

- Policy analysis can be negatively influenced by incorrect annotations.
  - Need to verify consistency between annotation and associated functions.
  - Translate annotations to JML [2] post-conditions and use existing tools (JACK[3]) to perform verification.
    - JML is a behavioral annotation language for Java methods and classes.
  - Use tools like WHY[4] .

Conclusions

- Proposed an annotation framework for XACML policies.
  - Enhance policy documentation
  - Supplement policy analysis
- Annotation syntax can express several categories of Boolean expressions that can be handled by state of the art policy analysis techniques.
- Future Work
  - Extend support for XPATH functions

References

- [1] Policy Ratification, Dakshi Agrawal, James Giles, Kang-Wong Lee, Jorge Lobo, POLICY 2005.
- [2] JML: A notation for detailed design, Gary T. Leavens, Albert L. Baker and Clyde Ruby, Specifications of Businesses and Systems, 1999.
- [3] Jack : Java applet correctness kit, L. Burdy and A. Requet, GDC 2002.
- [4] The Why Certification Tool, J. C. Filliatre, http://why.lri.fr/.

Thank you !