

Access Control Policy Tool (ACPT)

ACPT: Access Control Policy Tool

Access Control Policy

Presently Policy authoring are hand crafted by administrators, and difficult to check for correctness, we need tool for:

- Composing policy by structure framework
- Detecting conflicts in policy rules
- Efficient testing of implementation
- Policy code generation

Outline

Access Control Policy Tool (ACPT) Overview

Approaches

- Model specification and composition
- Property verification
- Policy testing
- XACML generation

Related work

Future work

ACPT Overview - Functions

Composition

Allows specification of policy combinations, rules and properties through model and rule templates.

Verification

Allows testing and verification of policies against specified properties and reports problems that may lead to security holes.

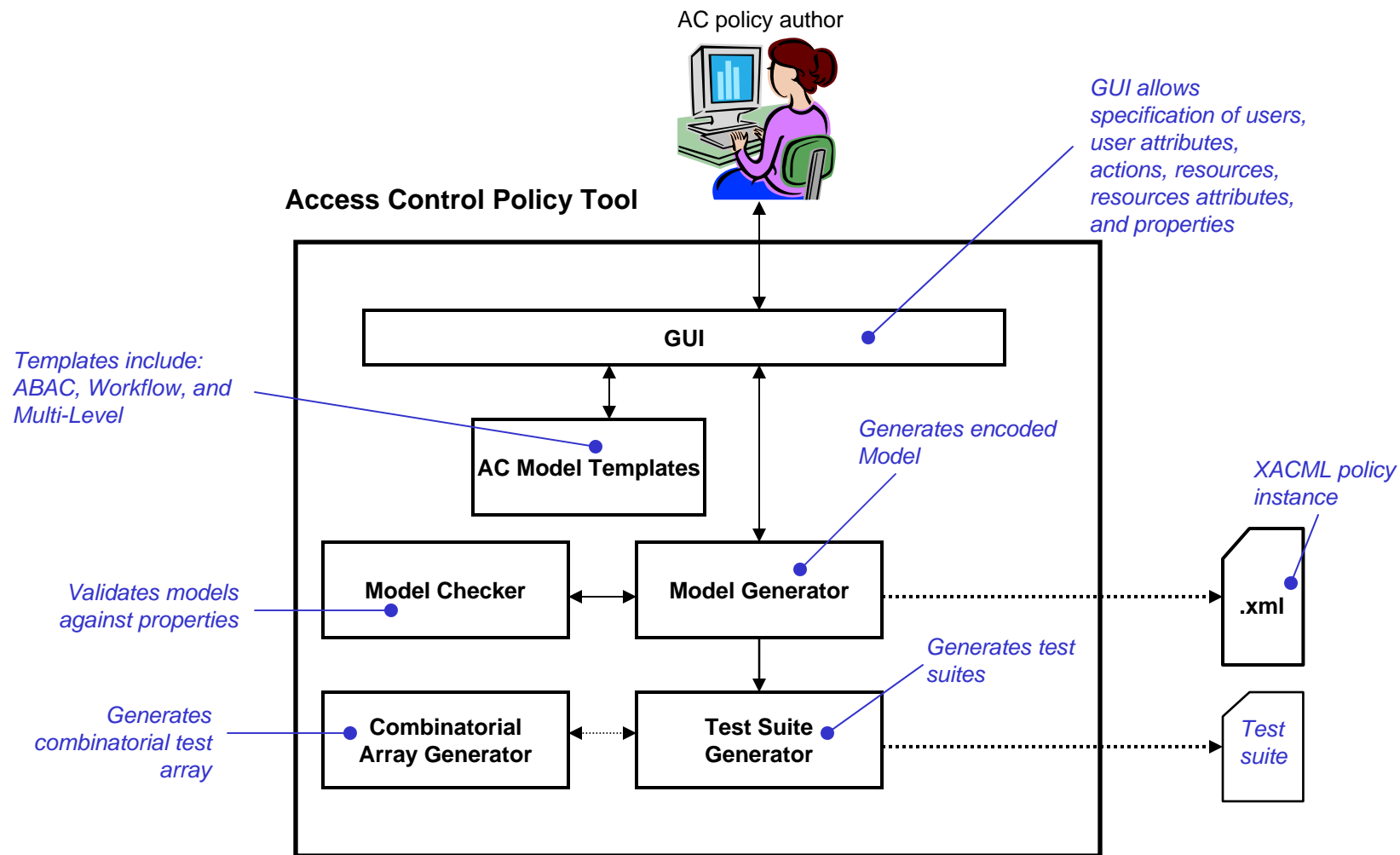
Testing

Generates efficient test suites (by applying NIST's combinatorial testing technology) for testing of access control implementation, test suites can be applied to any access control implementation.

Policy

XACML policy generation.

ACPT Overview - Architecture



ACPT Overview



Approaches: AC Model Specification and Composition

Allow to conveniently specify mandatory AC models (as well as AC rules) through pre-defined model templates

- Allow to create various models by specifying attribute values e.g., role subjects, resources, and actions for RBAC, user and resources ranks for MLS.
- Combine different AC models or rules into a composed one e.g., combine RBAC with multi-level models.
- allow to configure model priority for combining models or rules.

Approaches: AC Model Specification and Composition - Example

The screenshot displays the ACPT - DNI_demo.xml application window. The interface includes a menu bar with 'File' and 'Edit', and a tabbed interface with 'Subject' selected. The 'Properties' section is divided into two panes: 'Attributes' and 'Attribute Values'. The 'Attributes' pane lists four attributes: 'Government_Category;String', 'CFR_Part_23_Training;String', 'Assurance_Level;String', and 'Remote_Access;Boolean'. The 'Attribute Values' pane shows 'Federal' and 'State'. Both panes have 'Add', 'Update', and 'Delete' buttons. A 'Log' window at the bottom shows the following text: 'Param:Government_Category;String value:1', 'Param:Assurance_Level;String value:3', 'Param:read;Boolean value:1', and 'DONE!'.

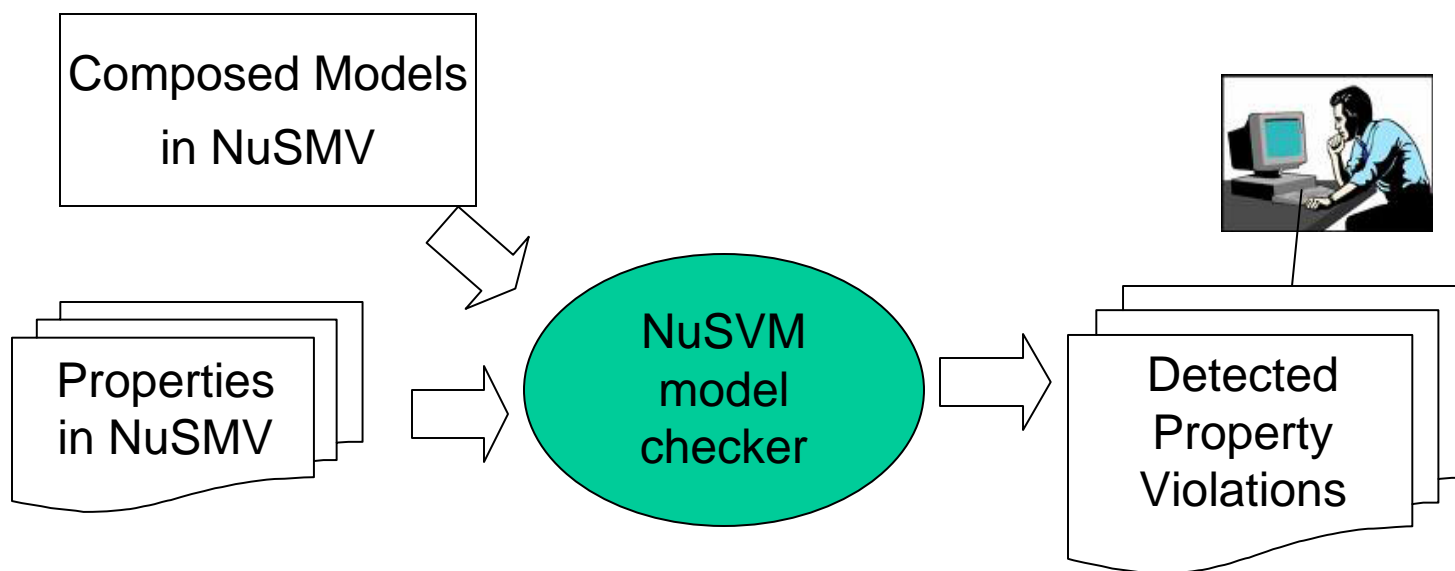
Approaches: Property Verification

Conflicts among policy entities and their complexity may leak unauthorized or prohibit authorized access privileges.

- Convert composed models and user-specified properties to input models and properties for a model checker (e.g., NuSMV).
- Verify models against specified properties, and report detected property violations.

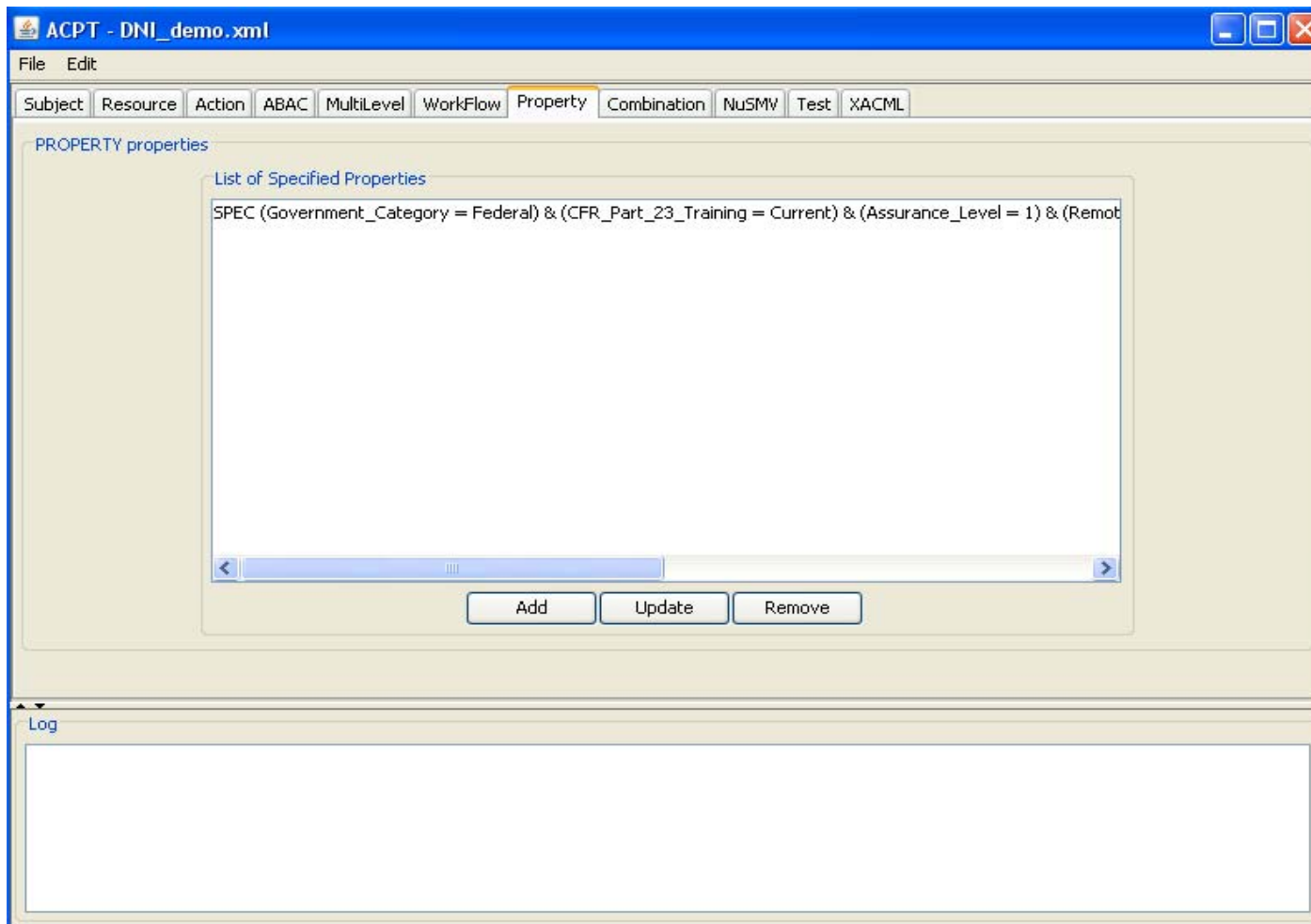
Approaches: Property Verification cont.

ACPT uses the NuSMV model checker, a well-structured, flexible, and efficient tool (supporting CTL and LTL model checking)



Approaches: Property Verification - Example

Property specification in ACPT



Approaches: Property Verification – Example cont.

Test the property against Policy A **combined** with Policy B. Combined policies has the priorities of the combined rules. This slide shows the combination of policies, where Policy B has higher priority than policy A

The screenshot displays the 'ACPT - DNI_demo.xml' application window. The 'Combination' tab is active, showing two lists of policy models. The left list, 'Policy Models', contains 'ABAC#Policy A' and 'ABAC#Policy B'. The right list, 'Selected Policy Models (first-applicable combining)', contains 'ABAC#Policy B' and 'ABAC#Policy A'. A 'Select-->' button is positioned between the lists. Below the lists are 'Remove', 'Up', and 'Down' buttons. A 'Log' window at the bottom shows the following text:

```
run NuSMV verification....  
NuSMV file is created  
NuSMV file : E:\V work\Data\project\access control\AC-testing\JeeHyun\v17\acpt\results\nu-src-  
run NuSMV verification....
```

Approaches: Property Verification – Example cont.

Test the property against Policy B, the result return *true*.

The screenshot shows the ACPT - DNI_demo.xml application window. The 'Test' tab is active, and the 'Commands' section shows 'ABAC#Policy B' selected for verification. A 'Results' window is open, displaying the output of the NuSMV verification process. The output includes version information, contact details, and the specific property being verified. The result of the verification is 'is true', which is circled in red.

```
File: results\nu-out--1516904739.txt

*** This is NuSMV 2.4.3 (compiled on Tue May 22 14:08:54 UTC 2007)
*** For more information on NuSMV see <http://nusmv.irst.itc.it>
*** or email to <nusmv-users@irst.itc.it>.
*** Please report bugs to <nusmv@irst.itc.it>.

*** This version of NuSMV is linked to the MiniSat SAT solver.
*** See http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat
*** Copyright (c) 2003-2005, Niklas Een, Niklas Sorensson

-- specification AG (((((Government_Category = Federal &
CFR_Part_23_Training = Current) & Assurance_Level = 1) &
Remote_Access = True) & Privacy_Category = ISE) & read = True) ->
AF (decision = Deny | decision = Non-applicable)) IN ABAC_Policy_B
is true
```

Approaches: Property Verification – Example cont.

Test the property against Policy A, the result return *false* with counterexample.

ACPT - DNI_demo.xml

File Edit

Subject Resource Action ABAC MultiLevel WorkFlow Property Combination NuSMV Test XACML

Commands

Select policies to be merged (without any order among policies)

ABAC#Policy A ABAC#Policy B

NUSMV Verification for Merged Policies

Combined Policies (Precedence based on first-applicable combination)

Default deny rule for each combined policy

NUSMV Verification for Combined Policies

Log

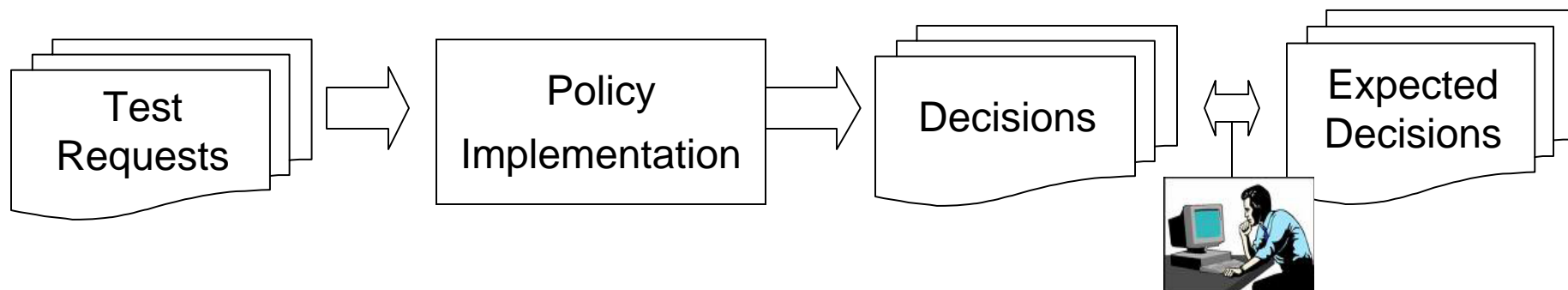
Results

File: results\nu-out--984463566.txt

```
-- specification AG (((((Government_Category = Federal &
CFR_Part_23_Training = Current) & Assurance_Level = 1) &
Remote_Access = True) & Privacy_Category = ISE) & read = True)
-> AF (decision = Deny | decision = Non-applicable)) IN
ABAC_Policy_A is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
-> State: 1.1 <-
  Government_Category = Federal
  CFR_Part_23_Training = Current
  Assurance_Level = 1
  Remote_Access = True
  Privacy_Category = ISE
  read = True
  ABAC_Policy_A.decision = Pending
-> Input: 1.2 <-
-- Loop starts here
-> State: 1.2 <-
  ABAC_Policy_A.decision = Permit
-> Input: 1.3 <-
-> State: 1.3 <-
```

Close

Approaches: Policy Testing



Assure correct policy implementations by

- Test Generation: Generate *test requests*.
- Test Execution: Evaluate test requests (against policy implementations) and produce their decisions.
- Test-Result Evaluation: Check if the decisions are consistent with expected decisions (from properties or manual inspection, etc.).
 - If inconsistent, implementation faults are revealed.

Approaches: Policy Testing – Combinatorial Testing

Exhaustive testing is impractical (esp. for large number of AC entities).

Generating efficient and effective test suites (from AC models) using Combinatorial Array Generation Technology.

Generated test suites can be applied to any access control implementations in practice to find implementation faults

Approaches: Policy Testing - Combinatorial Test cont.

Collect domain variables in AC models and generate **efficient** test suite automatically to detect faults using NIST combinatorial testing tool (ACTS)

- inputs: a domain of variables
- outputs: t-way covering arrays as tests

For example, domain of variables:

2 subjects: Faculty and Student

2 actions: write and view

2 resources: grades and records

Given the domain, 4 and 8 tests are generated for 2-way and 3-way interactions, respectively

<Faculty, grades, write>, <Faculty, records, view >, ...

Approaches: Policy Testing - Combinatorial Test cont.

- Combinatorial tests based on 2-way interactions

	SUBJECTS	RESOURCES	ACTIONS
1	Faculty	grades	write
2	Faculty	records	view
3	Student	grades	view
4	Student	records	write

- Combinatorial tests based on 3-way interactions (being exhaustive tests)

	SUBJECTS	RESOURCES	ACTIONS
1	Faculty	grades	write
2	Faculty	grades	view
3	Faculty	records	write
4	Faculty	records	view
5	Student	grades	write
6	Student	grades	view
7	Student	records	write
8	Student	records	view

Approaches: Policy Testing – Example

Test cases generation:

The screenshot shows the ACPT - DNI_demo.xml application interface. The main window has a menu bar (File, Edit) and a toolbar with tabs for Subject, Resource, Action, ABAC, MultiLevel, WorkFlow, Property, Combination, NuSMV, Test, and XACML. The 'Test' tab is active, showing a 'Commands' section with three sub-sections: 'Subject Attributes' (Government_Category;String, CFR_Part_23_Training;String, Assurance_Level;String, Remote_Access;Boolean), 'Resource Attributes' (Privacy_Category;String), and 'Action Attributes' (read;Boolean). Below these is a 'Select t Combinations (t-way):' dropdown set to '4' and a checked checkbox for 'Default deny rule for each combined policy'. A 'Test Generation' button is at the bottom.

A secondary window titled 'File: results\test--193574773.txt' displays the generated test cases:

```
1: (Government_Category = Federal) & (CFR_Part_23_Training = Current) & (Assurance_Level = 2) & (Remote_Access = True) & (Privacy_Category = ISE) & (read = True) -> decision = Permit  
2: (Government_Category = Federal) & (CFR_Part_23_Training = Current) & (Assurance_Level = 2) & (Remote_Access = False) & (Privacy_Category = SLT) & (read = False) -> decision = Deny  
3: (Government_Category = Federal) & (CFR_Part_23_Training = Expired_None) & (Assurance_Level = 2) & (Remote_Access = True) & (Privacy_Category = SLT) & (read = True) -> decision = Deny  
4: (Government_Category = Federal) & (CFR_Part_23_Training = Expired_None) & (Assurance_Level = 2) & (Remote_Access = False) & (Privacy_Category = ISE) & (read = False) -> decision = Deny  
5: (Government_Category = State) & (CFR_Part_23_Training =
```

A 'Log' window at the bottom left shows the execution process:

```
run NuSMV verification....  
NuSMV file is created  
NuSMV file : E:\V work\Data\project\access control\AC-te  
run NuSMV verification....
```

A 'Close' button is visible at the bottom of the results window.

Approaches: XACML Generation

Generate XACML policy based on the verified (combined or individual) models and rules.

Approaches: XACML Generation – Example

XACML generation:

The screenshot displays two windows from the ACPT - DNI_demo.xml application. The background window shows the 'XACML' tab with a 'Commands' area containing the following text:

```
Combined Policies (Precedence based on first-  
combination)  
1. ABAC#Policy B  
2. ABAC#Policy A
```

Below the commands, there is a checked checkbox labeled 'Default deny rule for each combined p' and a button labeled 'XACML Ge'. The foreground window, titled 'File: results\xacml-1195886153.xml', displays the generated XACML XML code:

```
</SubjectMatch>  
</Subject>  
<Subject>  
  <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:  
    <AttributeValue DataType="http://www.w3.org/2001/XM  
    <SubjectAttributeDesignator SubjectCategory="urn:oa  
  </SubjectMatch>  
</Subject>  
<Subject>  
  <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:  
    <AttributeValue DataType="http://www.w3.org/2001/XM  
    <SubjectAttributeDesignator SubjectCategory="urn:oa  
  </SubjectMatch>  
</Subject>  
</Subjects>  
<Resources>  
  <Resource>  
    <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1  
      <AttributeValue DataType="http://www.w3.org/2001  
      <ResourceAttributeDesignator AttributeId="Privac  
    </ResourceMatch>
```

At the bottom of the foreground window is a 'Close' button. The background window also has a 'Log' section at the bottom with the following text:

```
run NuSMV verification....  
It would take time for generating Test oracles...  
. .  
Test oracle creation is finished....
```

Approaches: XACML Generation – Example cont.

```

<PolicySet PolicySetId="n" PolicyCombiningAlgId="First-Applicable">
  <Target/>
  <Policy PolicyId="RBAC_school" RuleCombinationAlgId="First-Applicable">
    <Target/>
    <Rule RuleId="1" Effect="Deny">
      <Target>
        <Subjects><Subject> Student </Subject>
          <Subject> Secretary </Subject></Subjects>
        <Resources><Resource> Grades </Resource></Resources>
        <Actions><Action> Change </Action></Actions>
      </Target>
    </Rule>
    <Rule RuleId="2" Effect="Permit">
      <Target>
        <Subjects><Subject> Professor </Subject>
          <Subject> Lecturer </Subject>
          <Subject> Secretary </Subject></Subjects>
        <Resources><Resource> Grades </Resource>
          <Resource> Records </Resource></Resources>
        <Actions><Action> Change </Action>
          <Action> Read </Action></Actions>
      </Target>
    </Rule>
  </Policy>
  <Policy PolicyId="ABAC_school" RuleCombinationAlgId="First-Applicable">
    <Target/>
    <Rule RuleId="3" Effect="Permit">
      <Target>
        <Subjects><Subject> Jim </Subject></Subjects>
        <Resources><Resource> Records </Resource></Resources>
        <Actions><Action> Change </Action>
          <Action> Read </Action></Actions>
      </Target>
    </Rule>
  </Policy>
</PolicySet>

```

Rule 1:
A student or secretary
can not change grades.

Rule 2:
A professor, lecturer,
or secretary can
change grades or records.

Rule 3:
Jim can change grades or
records.

RBAC_school
policy

Policy rules

Related Work: Compare with Commercial AC Tools

A commercial AC policy management tool does not have all the following capabilities that NIST ACPT has:

- **AC model templates** for specifying models/policies: ABAC, Multi-Level, and Workflow.
- **Composition of multiple AC models** into a composed one, e.g., combine RBAC with MLS models.
- **AC property verification** to detect faults in models/policies. Some have only limited SOD (Separation of Duty) check.
- **Test-suite generation** for testing AC implementations in real operation environment to detect faults in implementations.

Future Work

- Available soon after final Alpha test.
- Enhance capabilities:
 - White-box model/properties verification to verify coverage and confinement of AC rules.
 - Additional AC policy templates including dynamic and historical access control models.
 - API or mechanism for acquiring or consuming information about users, attributes, resources, etc.

Questions?

vhu@nist.gov