

Access Control Policy Tool (ACPT), an assurance tool that combines symbolic model checking with combinatorial coverage

*Access Control Policy Tool (ACPT), an assurance tool that combines symbolic model checking with combinatorial coverage*

## Access Control Policy

Presently access control policies are hand crafted by administrators, and difficult to check for correctness. We need a 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.

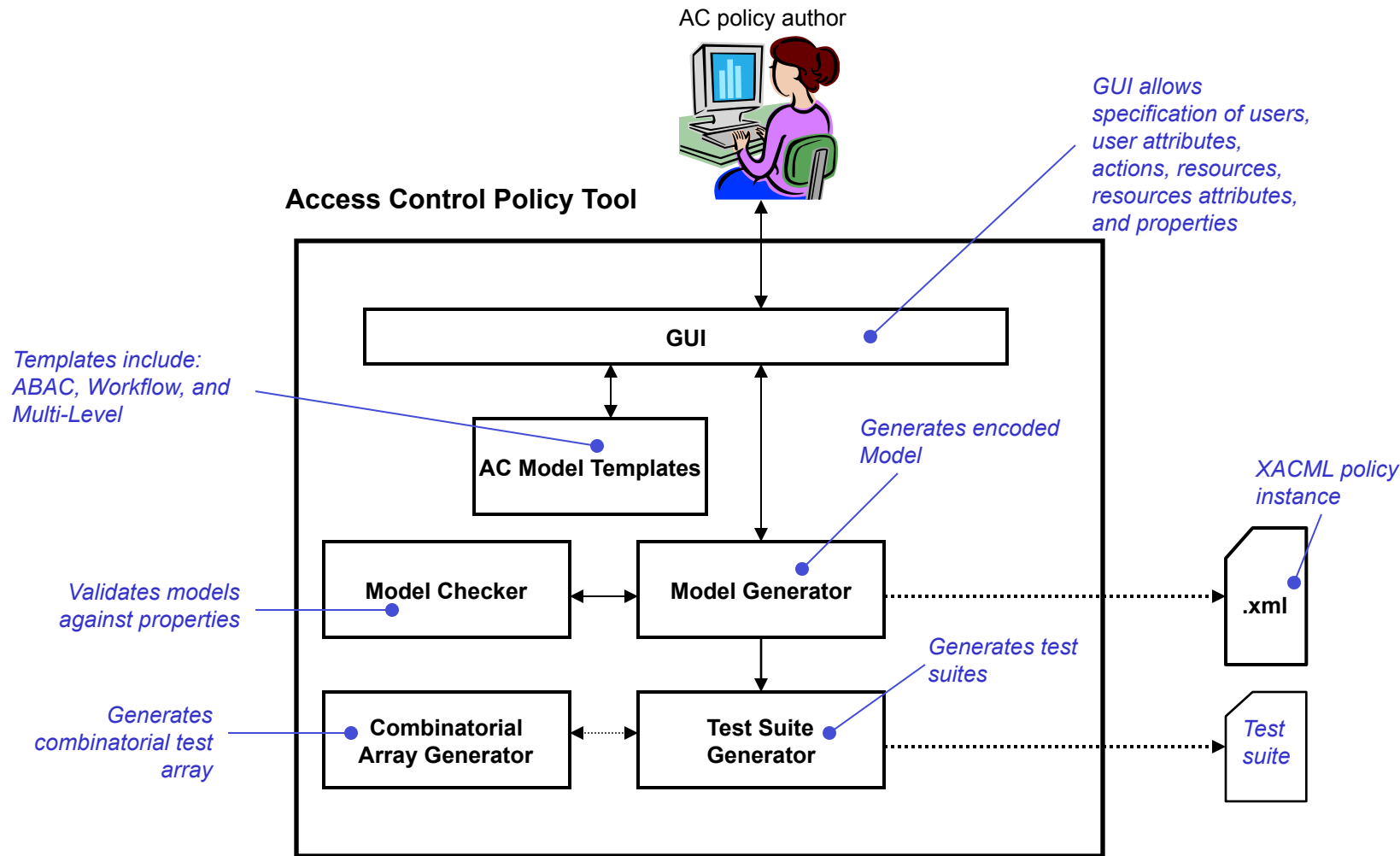
### Testing

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

- 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.
- 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 toolbar with tabs for 'Subject', 'Resource', 'Action', 'ABAC', 'MultiLevel', 'WorkFlow', 'Property', 'Combination', 'NuSMV', 'Test', and 'XACML'. The main area is titled 'Properties' and is divided into two panels: 'Attributes' and 'Attribute Values'. The 'Attributes' panel contains a list of attributes: 'Government\_Category;String', 'CFR\_Part\_23\_Training;String', 'Assurance\_Level;String', and 'Remote\_Access;Boolean'. The 'Attribute Values' panel contains a list of values: 'Federal' and 'State'. Below each panel are 'Add', 'Update', and 'Delete' buttons. At the bottom of the window is a 'Log' window with the following text: 'Param:Government\_Category;String value:1', 'Param:Assurance\_Level;String value:3', 'Param:read;Boolean value:1', and 'DONE !'. The log window has a scrollbar and navigation arrows.



## Approaches: Property Verification

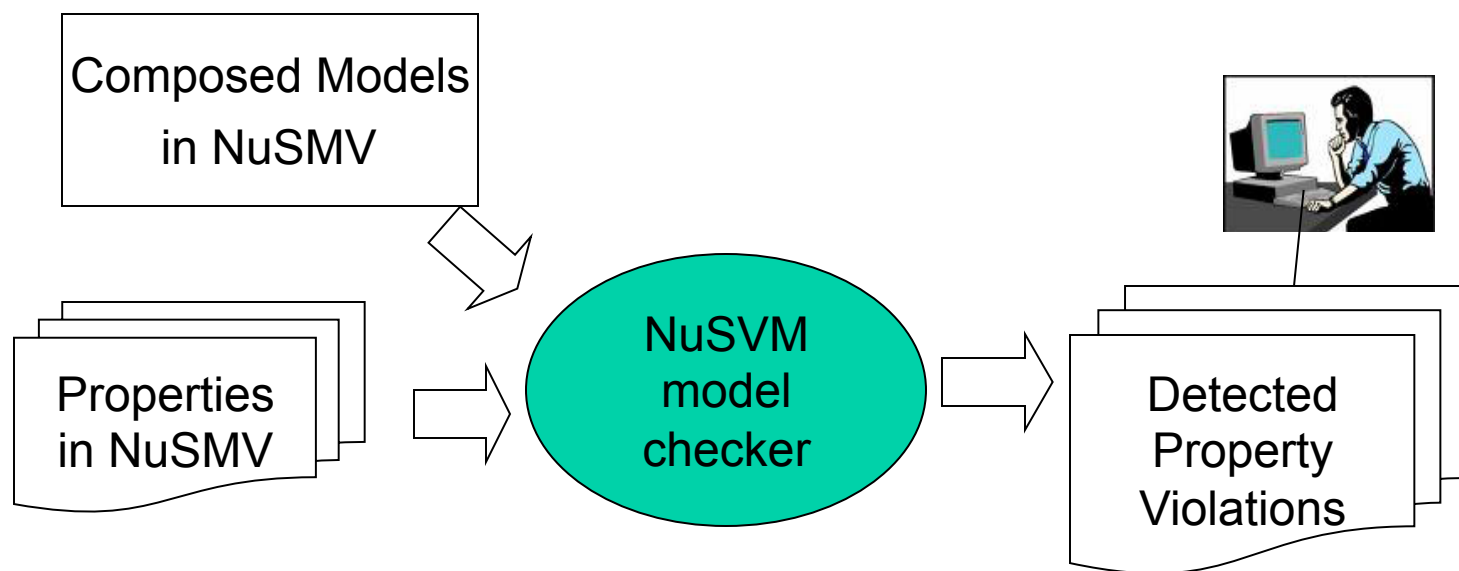
Conflicts among policy entities and their complexity may leak unauthorized or prohibit authorized access privileges.

Convert composed models and user-specified properties to input models and properties for the model checker - 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

The screenshot shows a software application window titled "ACPT - DNI\_demo.xml". The window has a menu bar with "File" and "Edit". Below the menu bar is a tabbed interface with several tabs: "Subject", "Resource", "Action", "ABAC", "MultiLevel", "WorkFlow", "Property", "Combination", "NuSMV", "Test", and "XACML". The "Property" tab is currently selected and highlighted. The main content area of the "Property" tab is titled "PROPERTY properties" and contains a section labeled "List of Specified Properties". This section displays a text box with the following property specification: `SPEC (Government_Category = Federal) & (CFR_Part_23_Training = Current) & (Assurance_Level = 1) & (Remot`. Below the text box is a horizontal scrollbar. At the bottom of the "List of Specified Properties" section are three buttons: "Add", "Update", and "Remove". Below the main content area is a "Log" section, which is currently empty.

# 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 interface. The main window has a menu bar with 'File' and 'Edit'. Below the menu bar is a tabbed interface with tabs for 'Subject', 'Resource', 'Action', 'ABAC', 'MultiLevel', 'WorkFlow', 'Property', 'Combination', 'NuSMV', 'Test', and 'XACML'. The 'Combination' tab is active, showing 'COMBINATION properties'. This area is divided into two panes: 'Policy Models' on the left and 'Selected Policy Models (first-applicable combining)' on the right. The 'Policy Models' pane contains a list with 'ABAC#Policy A' and 'ABAC#Policy B'. The 'Selected Policy Models' pane contains a list with 'ABAC#Policy B' and 'ABAC#Policy A'. A 'Select-->' button is positioned between the two panes. Below the panes are three buttons: 'Remove', 'Up', and 'Down'. At the bottom of the application is a 'Log' window with a scrollable text area containing 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 'NuSMV' tab is active, displaying a 'Commands' section with two policy selection options: 'ABAC#Policy A' (unchecked) and 'ABAC#Policy B' (checked). A 'Results' window is overlaid on top, titled 'File: results\nu-out--1516904739.txt'. The results text includes version information for NuSMV 2.4.3 and a MiniSat SAT solver. The property being tested is: `-- 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`. The result of the verification is `is true`, which is circled in red. A 'Log' window at the bottom left shows the execution steps: 'run NuSMV verification...', 'NuSMV file is created', 'NuSMV file : E:\V work\Data\project\access control\AC...', and 'run NuSMV verification...'.

# Approaches: Property Verification – Example cont.

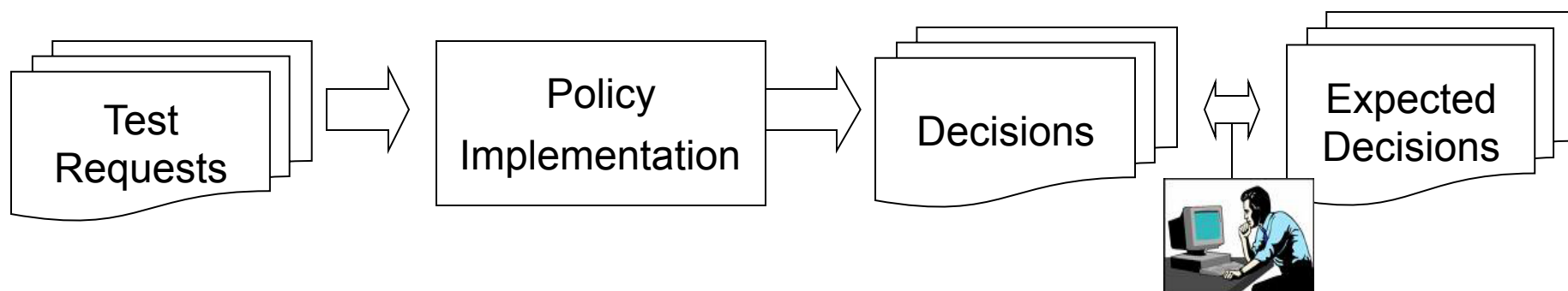
Test the property against Policy A, the result return *false* with counterexample.

The screenshot shows a software interface for policy verification. The main window is titled "ACPT - DNI\_demo.xml" and has a menu bar with "File" and "Edit". Below the menu bar are several tabs: "Subject", "Resource", "Action", "ABAC", "MultiLevel", "WorkFlow", "Property", "Combination", "NuSMV", "Test", and "XACML". The "Test" tab is active. In the "Commands" section, there are two checkboxes: "ABAC#Policy A" (checked) and "ABAC#Policy B" (unchecked). Below these is a button labeled "NUSMV Verification for Merged Policies". In the "Results" section, there is a text area containing "Combined Policies (Precedence based on first-applicable combination)". Below this is a checkbox labeled "Default deny rule for each combined policy" (unchecked) and a button labeled "NUSMV Verification for Combined Policies". At the bottom left, there is a "Log" section with a text area.

A smaller window titled "File: results\nu-out--984463566.txt" is overlaid on the main window. It displays the following text:

```
-- specification &G (((((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 <-
```

## 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 t-way covering array generation tool, NIST ACTS.

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, with 34 on-off switches, we have  $2^{34} = 17$  billion possible inputs:



## Approaches: Policy Testing - Combinatorial Test cont.

- Combinatorial tests based on 2-way interactions: 11 tests
- Combinatorial tests based on 3-way interactions: 33 tests
- ... 4-way interactions: 85 tests
- 5-way interactions: 211 tests
- 6-way interactions: 522 tests

Rationale: empirically-derived Interaction Rule, that failures involve only a small number of conditions interacting ( $\leq 6$  in real-world reports)

## Approaches: Policy Testing - Combinatorial Test cont.

Access control rules 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 displays configuration options for test case generation, including a dropdown for 'Select t Combinations (t-way):' set to 4, and a checked checkbox for 'Default deny rule for each combined policy'. A 'Test Generation' button is visible. Below the configuration, a 'Log' window shows the execution of NuSMV verification.

The 'Log' window output is as follows:

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

An overlaid 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 =
```

## 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 main 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'. A 'Log' window at the bottom left shows the following output:

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

The second 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>
```

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

# 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>
          </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

Commercial AC policy management tools do 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

- 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.
- More flexible user interfaces for composing AC rules and properties.
- AC property sets for specified AC requirements.

# Questions?

<http://csrc.nist.gov/groups/SNS/acpt/index.html>

[vhu@nist.gov](mailto:vhu@nist.gov)

[d.kuhn@nist.gov](mailto:d.kuhn@nist.gov)