eXtensible Access Control Markup Language (XACML) I

Nicola Zannone
Homework

Let $pol$ be an EPAL policy defined over a vocabulary $Voc$ where $Voc$ consists of the user, data, purpose and action hierarchies below.

(a) User Hierarchy

(b) Data Hierarchy

(c) Purpose Hierarchy

(d) Action Hierarchy

$$pol = \begin{cases} 
\langle (u_1, d_2, p_1, a_0)(\circ, \text{true}, o_1) \rangle \\
\langle (u_1, d_1, p_2, a_2)(+ , \text{true}, o_2) \rangle \\
\langle (u_1, d_0, p_0, a_2)(\circ, \text{true}, o_3) \rangle \\
\langle (u_4, d_2, p_4, a_4)(-, \text{true}, o_3) \rangle \\
\langle (u_2, d_1, p_2, a_4)(+, \text{true}, o_5) \rangle \\
\end{cases}$$

Default ruling: $-$

Default obligations: $\{o_6\}$

Evaluate the following access requests against $pol$:

- $req_1 = (u_3, d_5, p_1, a_2)$
- $req_2 = (u_4, d_3, p_6, a_3)$
- $req_3 = (u_2, d_6, p_2, a_4)$
- $req_4 = (u_2, d_3, p_6, a_5)$
Evaluate the following access requests against pol:

\[ \text{req}_1 = (u_3, d_5, p_1, a_2) \]
\[ \text{req}_2 = (u_4, d_3, p_6, a_3) \]
\[ \text{req}_3 = (u_2, d_6, p_2, a_4) \]
\[ \text{req}_4 = (u_2, d_3, p_6, a_5) \]
Evaluate the following access requests against pol:

- $req_1 = (u_3, d_5, p_1, a_2) \ (\neg, \{o_1, o_3, o_6\}) \ R_1, R_3$
- $req_2 = (u_4, d_3, p_6, a_3)$
- $req_3 = (u_2, d_6, p_2, a_4)$
- $req_4 = (u_2, d_3, p_6, a_5)$

### pol

\[
\begin{align*}
pol = \{ & \langle (u_1, d_2, p_1, a_0)(\circ, true, o_1) \rangle, \\
& \langle (u_1, d_1, p_2, a_2)(+ ,true, o_2) \rangle, \\
& \langle (u_1, d_0, p_0, a_2)(\circ, true, o_3) \rangle, \\
& \langle (u_4, d_2, p_4, a_4)(- ,true, o_4) \rangle, \\
& \langle (u_2, d_1, p_2, a_2)(+ ,true, o_5) \rangle \}\end{align*}
\]

Default ruling: $\neg$
Default obligations: $\{o_6\}$
Evaluate the following access requests against \( \text{pol} \):

\[
\begin{align*}
\text{req}_1 &= (u_3, d_5, p_1, a_2) \quad (−, \{o_1, o_3, o_6\}) \quad R_1, R_3 \\
\text{req}_2 &= (u_4, d_3, p_6, a_3) \quad (−, \{o_6\}) \quad \text{No rule applies} \\
\text{req}_3 &= (u_2, d_6, p_2, a_4) \\
\text{req}_4 &= (u_2, d_3, p_6, a_5)
\end{align*}
\]
Homework: Solution

Evaluate the following access requests against \( \text{pol} \):

\[ \begin{align*}
\text{req}_1 &= (u_3, d_5, p_1, a_2) & (\neg, \{o_1, o_3, o_6\}) & R_1, R_3 \\
\text{req}_2 &= (u_4, d_3, p_6, a_3) & (\neg, \{o_6\}) & \text{No rule applies} \\
\text{req}_3 &= (u_2, d_6, p_2, a_4) & (\text{scope_error}, \emptyset) \\
\text{req}_4 &= (u_2, d_3, p_6, a_5)
\end{align*} \]
Homework: Solution

\[ \text{pol} = \begin{cases} \langle (u_1, d_2, p_1, a_0)(\circ, \text{true}, o_1) \rangle \\
\langle (u_1, d_1, p_2, a_2)(+, \text{true}, o_2) \rangle \\
\langle (u_1, d_0, p_0, a_2)(\circ, \text{true}, o_3) \rangle \\
\langle (u_4, d_2, p_4, a_4)(-, \text{true}, o_4) \rangle \\
\langle (u_2, d_1, p_2, a_2)(+, \text{true}, o_5) \rangle \end{cases} \]

Default ruling: -
Default obligations: \{o_6\}

Evaluate the following access requests against pol:

\[ \text{req}_1 = (u_3, d_5, p_1, a_2) \quad (-, \{o_1, o_3, o_6\}) R_1, R_3 \]
\[ \text{req}_2 = (u_4, d_3, p_6, a_3) \quad (-, \{o_6\}) \text{ No rule applies} \]
\[ \text{req}_3 = (u_2, d_6, p_2, a_4) \quad (\text{scope_error}, \emptyset) \]
\[ \text{req}_4 = (u_2, d_3, p_6, a_5) \quad (+, \{o_5\}) R_5 \]
What is XACML

- XACML stands for eXtensible Access Control Markup Language
- OASIS standard
- XACML defines
  - Language to express access control policies, access requests and responses
    - Attribute-based access control policy language
    - Implemented in XML
    - Default attributes and functions
  - Access decision process
    - how to evaluate a request against a policy
  - High-level architecture (PEP, PDP, PIP, PAP, etc.)
History

- OASIS Standard
- XACML 1.0 (February 2003)
- XACML 2.0 (February 2005)
- XACML 3.0 (January 2013)
  - XACML Core
  - Privacy Policy Profile
  - Role Based Access Control
  - OASIS Security Assertions Markup Language (SAML)
  - ...

- Implementations
  - Sun Implementation (http://sunxacml.sourceforge.net/)
  - Xengine (http://xacmlpdp.sourceforge.net/)
  - SAFAX (http://security1.win.tue.nl/safax/)
  - Many more at http://www.oasis-open.org/
XACML Data Flow

**Policy administration point (PAP)**
creates a policy or policy set

**Context handler** constructs an
XACML request context and sends it
to the PDP

**Policy decision point (PDP)**
evaluates applicable policy and
renders an authorization decision

**Policy enforcement point (PEP)**
performs access control, by making
decision requests and enforcing
authorization decisions

**Policy information point (PIP)**
that acts as a source of attribute values
Outline

- Policy Language
- Access Request
- Evaluation
- Response
- Exercise
How to specify a policy?
Attribute-Based Access Control (ABAC)

- Attributes are (name, value) pairs
  - values can be complex data structures
- Associated with
  - subjects
  - actions
  - objects
  - environment (context)
- No fundamental difference between subjects, objects, actions and other contextual information
XACML Policy Language Model
XACML Policies (simplified)

- **Target**: identifies which requests are applicable
- **Effect**: Permit or Deny
- **Condition**: defines additional constraints on the applicability of requests
- **Rule**: returns an effect if the target matches
- **Policy**: composes several rules together
- **Obligation**: mandatory requirements to be satisfied
XACML Policy Language

- ⟨Target⟩ contains a conjunctive sequence of ⟨AnyOf⟩ elements
- ⟨AnyOf⟩ contains a disjunctive sequence of ⟨AllOf⟩ elements
- ⟨AllOf⟩ contains a conjunctive sequence of ⟨Match⟩ elements
- ⟨Match⟩ defines an atomic access constraint
  - Determine which attributes and attribute values should be provided in a request in order for the request to be applicable
Attributes, Data Types, Functions

Attributes

- urn:oasis:names:tc:xacml:1.0:subject:authn-locality:ip-address
- urn:oasis:names:tc:xacml:1.0:subject:authentication-method
- urn:oasis:names:tc:xacml:1.0:subject:authentication-time
- urn:oasis:names:tc:xacml:1.0:subject:request-time
- urn:oasis:names:tc:xacml:1.0:subject:subject-id
- urn:oasis:names:tc:xacml:2.0:example:attribute:role
- urn:oasis:names:tc:xacml:1.0:resource:resource-location
- urn:oasis:names:tc:xacml:1.0:resource:resource-id
- urn:oasis:names:tc:xacml:1.0:action:action-id

Data types

- http://www.w3.org/2001/XMLSchema#string
- http://www.w3.org/2001/XMLSchema#boolean
- http://www.w3.org/2001/XMLSchema#time
- http://www.w3.org/2001/XMLSchema#date

Functions

- urn:oasis:names:tc:xacml:1.0:function:string-equal
- urn:oasis:names:tc:xacml:1.0:function:boolean-equal
- urn:oasis:names:tc:xacml:1.0:function:date-equal
- urn:oasis:names:tc:xacml:1.0:function:time-equal
Example

- In a company, the staff can access the printer.

- Attributes:
  - User ID: Alice, Bob, ... 
  - Group: staff, intern, ...
  - Resources: printer, scanner, ...
Example: ⟨Match⟩

The ⟨Match⟩ element identifies a particular value for an attribute:

- **Group is staff**
  
  ```xml
  <Match MatchId="string-equal">
  <AttributeValue>staff</AttributeValue>
  <AttributeDesignator AttributeId="group"/>
  </Match>
  ```

  **Simplified Syntax**
  
  Match:
  - group = staff

- **Resource is printer**
  
  ```xml
  <Match MatchId="string-equal">
  <AttributeValue>printer</AttributeValue>
  <AttributeDesignator AttributeId="resource"/>
  </Match>
  ```

  **Simplified Syntax**
  
  Match:
  - resource = printer
The `<Target>` element defines which requests are applicable:

```xml
<Target>
  <AnyOf>
    <AllOf>
      <Match MatchId="string-equal">
        <AttributeValue>staff</AttributeValue>
        <AttributeDesignator AttributeId="group">
      </Match>
      <Match MatchId="string-equal">
        <AttributeValue>printer</AttributeValue>
        <AttributeDesignator AttributeId="resource">
      </Match>
    </AllOf>
  </AnyOf>
</Target>
```

**Simplified Syntax**

Target:
- Match: all
  - group = staff
  - resource = printer
<Rules>

- RuleId [Required]
  - A string identifying this rule.
- Effect [Required]
  - Rule effect can be either Permit or Deny
- ⟨Description⟩ [Optional]
  - A free-form description of the rule
- ⟨Target⟩ [Optional]
  - Define the applicability of the rule
- ⟨Condition⟩ [Optional]
  - A predicate that MUST be satisfied
- ⟨ObligationExpressions⟩ [Optional]
  - A conjunctive sequence of obligation expressions which MUST be evaluated into obligations by the PDP.
- ⟨AdviceExpressions⟩ [Optional]
  - A conjunctive sequence of advice expressions which MUST be evaluated into advice by the PDP.
Smallest entity that SHALL be presented to the PDP for evaluation

- **PolicyId [Required]**
  - Policy identifier

- **RuleCombiningAlgId [Required]**
  - Algorithm used to combine the rules forming the policy

- **Description** [Optional]**
  - A free-form description of the policy

- **Target** [Required]
  - Define applicability of the policy

- **Rule** [Any Number]
  - Sequence of rules forming the policy
  - Combined according to combining algorithm

- **ObligationExpressions** [Optional]
  - A conjunctive sequence of obligation expressions which MUST be evaluated into obligations by the PDP.

- **AdviceExpressions** [Optional]
  - A conjunctive sequence of advice expressions which MUST be evaluated into advices by the PDP.
〈PolicySet〉

- PolicySetId [Required]
  - Policy set identifier
- PolicyCombiningAlgId [Required]
  - Define how policies forming the policy set are combined
- 〈Description〉 [Optional]
  - A free-form description of the policy set
- 〈Target〉 [Required]
  - Define the applicability of the policy set
- 〈PolicySet〉 [Any Number]
  - A policy set that is included in this policy set
- 〈Policy〉 [Any Number]
  - A policy that is included in this policy set
- 〈PolicyIdReference〉 [Any Number]
  - A reference to a policy that MUST be included in this policy set
- 〈ObligationExpressions〉 [Optional]
  - A conjunctive sequence of obligation expressions which MUST evaluated into obligations by the PDP
- 〈AdviceExpressions〉 [Optional]
  - A conjunctive sequence of advice expressions which MUST evaluated into advices by the PDP
Example: Policy

Access is permitted to the printer for members of the staff.

```
<Policy PolicyID="P1"
   RuleCombiningAlgId="deny-overrides">
   <Rule Effect="Permit">
      <Target>
         <AnyOf>
            <AllOf>
               <Match MatchId="string-equal">
                  <AttributeValue>staff</AttributeValue>
                  <AttributeDesignator AttributeId="group">
               </Match>
               <Match MatchId="string-equal">
                  <AttributeValue>printer</AttributeValue>
                  <AttributeDesignator AttributeId="resource">
               </Match>
            </AllOf>
         </AnyOf>
      </Target>
   </Rule>
</Policy>
```

Simplified Syntax

- policy: P1
- rule: permit
  - match: all
    - group = staff
    - resource = printer
Policy Hierarchy
How to specify access request?
ReturnPolicyIdList [Required]
- used to request that the PDP return a list of all fully applicable policies and policy sets which were used in the decision as a part of the decision response.

Attributes [One to Many]
- Specifies information about attributes of the request context by listing a sequence of Attribute elements associated with an attribute category. One or more Attributes elements are allowed. Different Attributes elements with different categories are used to represent information about the subject, resource, action, environment or other categories of the access request.
Example: Request

Alice, a staff member, wants to access the printer.

```xml
<Request>
  <Attributes>
    <Attribute AttributeId="user">
      <AttributeValue>Alice</AttributeValue>
    </Attribute>
    <Attribute AttributeId="group">
      <AttributeValue>staff</AttributeValue>
    </Attribute>
    <Attribute AttributeId="resource">
      <AttributeValue>printer</AttributeValue>
    </Attribute>
  </Attributes>
</Request>
```

Simplified Syntax

Request:
- user: alice
- group: staff
- resource: printer
Policy evaluation
Policy Applicability

- A Rule/Policy/PolicySet has a target
  - Attributes of subject, object, action, environment
- Access Requests specify a context
  - Attributes of subject, object, action, environment
- Rule/Policy/PolicySet is applicable if the attribute values in the access request “match” the attribute values in the target

REMARK: Attributes in the target have attribute \textit{MustBePresent}. In case that no matching attribute is present in the request, then the attribute is considered missing. If the attribute is missing, then \textit{MustBePresent} governs the applicability of the rule/policy/policyset

- If \textit{MustBePresent} is “False” (default value), then a missing attribute makes the rule/policy/policyset not applicable
- If \textit{MustBePresent} is “True”, then a missing attribute results in an error (Indeterminate)
## Target evaluation

<table>
<thead>
<tr>
<th><code>&lt;AnyOf&gt;</code> values</th>
<th>Target value</th>
</tr>
</thead>
<tbody>
<tr>
<td>All “Match”</td>
<td>“Match”</td>
</tr>
<tr>
<td>At least one “No Match”</td>
<td>“No Match”</td>
</tr>
<tr>
<td>Otherwise</td>
<td>“Indeterminate”</td>
</tr>
</tbody>
</table>

The target value is “Match” if all `<AnyOf>` specified in the target match values in the request context.

<table>
<thead>
<tr>
<th><code>&lt;AllOf&gt;</code> values</th>
<th><code>&lt;AnyOf&gt;</code> Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>At least one “Match”</td>
<td>“Match”</td>
</tr>
<tr>
<td>None matches and at least one “Indeterminate”</td>
<td>“Indeterminate”</td>
</tr>
<tr>
<td>All “No match”</td>
<td>“No match”</td>
</tr>
</tbody>
</table>

`<AnyOf>` match values in the request context if at least one of their `<AllOf>` elements matches a value in the request context.

<table>
<thead>
<tr>
<th><code>&lt;Match&gt;</code> values</th>
<th><code>&lt;AllOf&gt;</code> Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>All “True”</td>
<td>“Match”</td>
</tr>
<tr>
<td>No “False” and at least one “Indeterminate”</td>
<td>“Indeterminate”</td>
</tr>
<tr>
<td>At least one “False”</td>
<td>“No match”</td>
</tr>
</tbody>
</table>

`<AllOf>` matches a value in the request context if the value of all its `<Match>` elements is “True”.
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff
    - resource: scan
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff
    - resource: scan

False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff
    - resource: scan

True
False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff
    - resource: scan

No match
True
False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
    True
  - AllOf
    - group: staff
    - resource: scan
    No match
    True
    False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff
    - resource: scan

True
True
No match
True
False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
    Match True True
  - AllOf
    - group: staff
    - resource: scan
    No match True False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff
    - resource: scan

Match True False
Example: Target Evaluation

Request:
- user: alice
- group: staff
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
    - AllOf
      - group: staff
      - resource: scan

Match
Match
Match
True
True
No match
True
False
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

(M) Must be present
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

(M) Must be present
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

(M) Must be present

Indeterminate
False
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
      - resource: printer
    - AllOf
      - group: staff (M)
      - resource: scan

(M) Must be present

No match
Indeterminate
False
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

True
No match
Indeterminate
False

(M) Must be present
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

(M) Must be present

- False
- True
- No match
- Indeterminate
- False
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

(M) Must be present
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

(M) Must be present

No match  No match  False  True  No match  Indeterminate  False
Example: Target Evaluation

Request:
- user: alice
- resource: printer

Target:
- AnyOf
  - AllOf
    - group: staff
    - resource: printer
  - AllOf
    - group: staff (M)
    - resource: scan

No match
No match
No match
False
True
Indeterminate
False

(M) Must be present
Access request → PEP → PDP → PAP → PS3

PS1 → P1 → R1 (Deny) → R2 (Permit) → P2 → R3 (Deny) → R4 (Deny)

PS2 → P3 → R5 (Permit) → R6 (Deny) → R7 (Deny) → P4 → R8 (Permit) → R9 (Permit)

Access decision?
Access Decision Set

- Permit (P)
  - requested access is permitted

- Deny (D)
  - requested access is denied

- Indeterminate (I)
  - PDP is unable to evaluate the request
  - missing attributes, network errors while retrieving policies, division by zero during policy evaluation, syntax errors in the decision request or in the policy, etc.

- NotApplicable (NA)
  - PDP does not have any policy that applies to the request
Extended *Indeterminate* Set

- Record potential effect value when errors occur
  - Indeterminate\{P\} (I\{P\})
  - Indeterminate\{D\} (I\{D\})
  - Indeterminate\{PD\} (I\{PD\})

- Used for
  - rule evaluation
  - some combining algorithms
Decision Set Projection

\[ D_4 = \{ P, D, NA, I \} \]
\[ D_6 = \{ P, D, NA, I\{ P \}, I\{ D \}, I\{ PD \} \} \]

\[ D_6 \rightarrow D_4 \quad D_4 \rightarrow D_6 \]

\[ I\{ P \} \rightarrow I \]
\[ I\{ D \} \rightarrow I \]
\[ I\{ PD \} \rightarrow I \]
\[ I \rightarrow I\{ PD \} \]
# Rule evaluation

<table>
<thead>
<tr>
<th>Target</th>
<th>Condition</th>
<th>Rule value</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Match” or no target</td>
<td>“True”</td>
<td>Effect “NotApplicable”</td>
</tr>
<tr>
<td>“Match” or no target</td>
<td>“False”</td>
<td>“Indeterminate{P}” if the Effect is Permit, or “Indeterminate{D}” if the Effect is Deny</td>
</tr>
<tr>
<td>“Match” or no target</td>
<td>“Indeterminate”</td>
<td>“Indeterminate{P}” if the Effect is Permit, or “Indeterminate{D}” if the Effect is Deny</td>
</tr>
<tr>
<td>“No-match”</td>
<td>Don’t care</td>
<td>“NotApplicable”</td>
</tr>
<tr>
<td>“Indeterminate”</td>
<td>Don’t care</td>
<td>“Indeterminate{P}” if the Effect is Permit, or “Indeterminate{D}” if the Effect is Deny</td>
</tr>
</tbody>
</table>
Example: Rule Evaluation (1)

Request:
- user: alice
- group: staff
- resource: printer

rule: permit
- target:
  - AnyOf
    - AllOf
      - group: staff
      - resource: printer
      - AllOf
        - group: staff
        - resource: scan

Match
Match
Match
True
True
No match
True
False
## Example: Rule Evaluation (1)

**Request:**
- user: alice
- group: staff
- resource: printer

<table>
<thead>
<tr>
<th>rule</th>
<th>permit</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td></td>
</tr>
<tr>
<td>- AnyOf</td>
<td>Match</td>
</tr>
<tr>
<td>- AllOf</td>
<td>Match</td>
</tr>
<tr>
<td>- group: staff</td>
<td>True</td>
</tr>
<tr>
<td>- resource: printer</td>
<td>True</td>
</tr>
<tr>
<td>- AllOf</td>
<td>No match</td>
</tr>
<tr>
<td>- group: staff</td>
<td>True</td>
</tr>
<tr>
<td>- resource: scan</td>
<td>False</td>
</tr>
</tbody>
</table>
Example: Rule Evaluation (2)

Request:
- user: alice
- resource: printer

rule: permit
- target:
  - AnyOf
    - AllOf
      - group: staff
      - resource: printer
      - AllOf
      - group: staff (M)
      - resource: scan

(M) Must be present

No match
No match
No match
False
True
No match
Indeterminate
False
Example: Rule Evaluation (2)

Request:
- user: alice
- resource: printer

rule: permit
- target:
  - AnyOf
    - AllOf
      - group: staff
      - resource: printer
    - AllOf
      - group: staff (M)
      - resource: scan

NotApplicable
No match
No match
False
True
No match
Indeterminate
False

(M) Must be present
Different rules and policies can be applicable!!
Combining Algorithms

- Deny-overrides
- Ordered-deny-overrides
- Permit-overrides
- Ordered-permit-overrides
- Deny-unless-permit
- Permit-unless-deny
- First-applicable
- Only-one-applicable (only PolicyCombiningAlgorithm)
Deny Overrides (Defined over $D_6$)

```java
Decision denyOverridesCombiningAlgorithm(Node[] children)
    Boolean atLeastOneErrorD = false;
    Boolean atLeastOneErrorP = false;
    Boolean atLeastOneErrorDP = false;
    Boolean atLeastOnePermit = false;
    for (i = 0; i < lengthOf(children); i++)
        Decision decision = children[i].evaluate();
        if (decision == Deny)
            return Deny;
        if (decision == Permit)
            atLeastOnePermit = true;
            continue;
        if (decision == NotApplicable)
            continue;
        if (decision == IndeterminateD)
            atLeastOneErrorD = true;
            continue;
        if (decision == IndeterminateP)
            atLeastOneErrorP = true;
            continue;
        if (decision == IndeterminateDP)
            atLeastOneErrorDP = true;
            continue;
        if (atLeastOneErrorDP)
            return IndeterminateDP;
        if (atLeastOneErrorD && (atLeastOneErrorP || atLeastOnePermit))
            return IndeterminateDP;
        if (atLeastOneErrorD)
            return IndeterminateD;
        if (atLeastOnePermit)
            return Permit;
        if (atLeastOneErrorP)
            return IndeterminateP;
    return NotApplicable;
```
Permit Overrides (Defined over $\mathcal{D}_6$)

```java
Decision permitOverridesCombiningAlgorithm(Node[] children)
    Boolean atLeastOneErrorD = false;
    Boolean atLeastOneErrorP = false;
    Boolean atLeastOneErrorDP = false;
    Boolean atLeastOneDeny = false;
    for( i=0 ; i < lengthOf(children) ; i++ )
        Decision decision = children[i].evaluate();
        if (decision == Deny)
            atLeastOneDeny = true;
            continue;
        if (decision == Permit)
            return Permit;
        if (decision == NotApplicable)
            continue;
        if (decision == IndeterminateD)
            atLeastOneErrorD = true;
            continue;
        if (decision == IndeterminateP)
            atLeastOneErrorP = true;
            continue;
        if (decision == IndeterminateDP)
            atLeastOneErrorDP = true;
            continue;
        if (atLeastOneErrorDP)
            return IndeterminateDP;
        if (atLeastOneErrorP && (atLeastOneErrorD || atLeastOneDeny))
            return IndeterminateDP;
        if (atLeastOneErrorP)
            return IndeterminateP;
        if (atLeastOneDeny)
            return Deny;
        if (atLeastOneErrorD)
            return IndeterminateD;
    return NotApplicable;
```
First Applicable (Defined over $\mathcal{D}_4$)

Decision firstApplicableEffectRuleCombiningAlgorithm(Node[] children)
  for( i = 0 ; i < lengthOf(children) ; i++ )
    Decision decision = evaluate(children[i]);
    if (decision == Deny)
      return Deny;
    if (decision == Permit)
      return Permit;
    if (decision == NotApplicable)
      continue;
    if (decision == Indeterminate)
      return Indeterminate;
  return NotApplicable;
Only One Applicable (Defined over $D_4$) only for Policies

Decision onlyOneApplicablePolicyPolicyCombiningAlgorithm(Policy policy[])
    Boolean atLeastOne = false;
    Policy selectedPolicy = null;
    ApplicableResult appResult;
    for ( i = 0; i < lengthOf(policy) ; i++ )
        appResult = isApplicable(policy[i]);
        if ( appResult == Indeterminate )
            return Indeterminate;
        if( appResult == Applicable )
            if ( atLeastOne )
                return Indeterminate;
            else
                atLeastOne = true;
                selectedPolicy = policy[i];
        if ( appResult == NotApplicable )
            continue;
    if ( atLeastOne )
        return evaluate(selectedPolicy);
    else
        return NotApplicable;
Other combining algorithms

- Ordered-deny-overrides
  - Identical to Deny-overrides combining algorithms
  - with one exception...
  - Order in which rules are evaluated SHALL match the order as listed in the policy

- Ordered-permit overrides

- Deny-unless-permit

- Permit-unless-deny

- To be studied
  - XACML specification (pages 135-147)
Combining Algorithms: Example

PS3
First applicable

PS1
Only one applicable

PS2
Permit overrides

P1
First applicable

P2
Deny overrides

P3
Permit overrides

P4
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
Only one applicable

PS2
Permit overrides

P1
First applicable

P2
Deny overrides

P3
Permit overrides

P4
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
Applicable
Only one applicable

PS2
Permit overrides

P1
First applicable

P2
Deny overrides

P3
Permit overrides

P4
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
Applicable
Only one applicable

PS2
Permit overrides

P2
Deny overrides

P1
N/A
First applicable

P4
Deny overrides

P3
Permit overrides

P5
First applicable

R1
Deny

P2
Deny overrides

R2
Permit

P3
Permit overrides

R3
Permit

P4
Deny overrides

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example

- **R1**: Deny
- **R2**: Permit
- **R3**: Permit
- **R4**: Deny
- **R5**: Permit
- **R6**: Deny
- **R7**: Deny
- **R8**: Permit
- **R9**: Permit

- **P1**: N/A
- **P2**: Applicable
- **P3**: Applicable
- **P4**: Applicable
- **P5**: Applicable

- **PS1**: Only one applicable
- **PS2**: Permit overrides
- **PS3**: First applicable

-XACML

P1

R1

R2

R3

R4

R5

R6

R7

R8

R9

P2

R1

R2

R3

R4

R5

R6

R7

R8

R9

P3

R1

R2

R3

R4

R5

R6

R7

R8

R9

P4

R1

R2

R3

R4

R5

R6

R7

R8

R9

P5

R1

R2

R3

R4

R5

R6

R7

R8

R9

Applicable

First applicable

Deny overrides

Permit overrides

Deny overrides

First applicable
Combining Algorithms: Example

- **PS3**
  - Applicable
  - First applicable

- **PS1**
  - Applicable
  - Only one applicable

- **PS2**
  - Permit overrides

- **P1**
  - N/A
  - First applicable

- **P2**
  - Applicable
  - Deny overrides

- **P3**
  - Permit overrides

- **P4**
  - Deny overrides

- **P5**
  - First applicable

- **R1**
  - Deny

- **R2**
  - Permit

- **R3**
  - Permit

- **R4**
  - Deny

- **R5**
  - Permit

- **R6**
  - Deny

- **R7**
  - Deny

- **R8**
  - Permit

- **R9**
  - Permit
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
Applicable
Only one applicable

PS2
Permit overrides

P5
First applicable

P1
N/A
First applicable

P2
Applicable
Deny overrides

P3
Permit overrides

P4
Deny overrides

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit

P2
Applicable
Deny overrides
Combining Algorithms: Example

XACML
Policy Evaluation

PS3
Applicable
First applicable

PS1
Applicable
Only one applicable

PS2
Permit overrides

P1
N/A
First applicable

P2
N/A
Deny overrides

P3
Permit overrides

P4
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit

P1
N/A
First applicable

P2
N/A
Deny overrides

P3
Permit overrides

P4
Deny overrides

P5
First applicable
Combining Algorithms: Example

- **R1** Deny
- **R2** Permit
- **R3** Permit
- **R4** Deny
- **R5** Permit
- **R6** Deny
- **R7** Deny
- **R8** Permit
- **R9** Permit

**Policy Evaluation**

- **PS1** N/A
  - Only one applicable
- **PS2** N/A
  - Permit overrides
- **PS3** N/A
  - First applicable

**Applicable Rules**

- **P1** N/A
  - First applicable
- **P2** N/A
  - Deny overrides
- **P3** N/A
  - Permit overrides
- **P4** N/A
  - Deny overrides
- **P5** N/A
  - First applicable

**N/A**

- **R1** Deny
- **R2** Permit
- **R3** Permit
- **R4** Deny
- **R5** Permit
- **R6** Deny
- **R7** Deny
- **R8** Permit
- **R9** Permit
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
N/A
Only one applicable

PS2
Applicable
Permit overrides

P1
N/A
First applicable

P2
N/A
Deny overrides

P3
Permit overrides

P4
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
N/A
Only one applicable

PS2
Applicable
Permit overrides

P1
N/A
First applicable

P2
N/A
Deny overrides

P3
Applicable
Permit overrides

P4
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit

P1
N/A

P2
N/A

P3

P4

P5

PS2

PS1

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A

N/A
Combining Algorithms: Example

- **R1**: Deny
- **R2**: Permit
- **R3**: Permit
- **R4**: Deny
- **R5**: Permit
- **R6**: Deny
- **R7**: Deny
- **R8**: Permit
- **R9**: Permit

**Policy Evaluation**

- **R1** and **R2** are evaluated first.
- **R3** and **R4** are evaluated after **R1** and **R2**.
- **R5** and **R6** are evaluated after **R3** and **R4**.
- **R7** and **R8** are evaluated after **R5** and **R6**.
- **R9** is evaluated last.

**Applicability**

- **P1**: First applicable
- **P2**: Deny overrides
- **P3**: Permit overrides
- **P4**: Deny overrides
- **P5**: First applicable

**Override Examples**

- **P1** and **P3** are mutually exclusive.
- **P2** overrides **P1**.
- **P3** overrides **P4**.

**Combining Algorithms**

- **PS1**: Only one applicable
- **PS2**: Permit overrides
- **PS3**: First applicable

**N/A**

- **N/A** indicates no applicable policy.

---

XACML

Policy Evaluation
Combining Algorithms: Example
Combining Algorithms: Example

PS1
Only one applicable

P1
First applicable

R1
Deny

R2
Permit

P2
Deny overrides

N/A

R3
Permit

R4
Deny

P3
Deny overrides

R5
Permit

R6
Deny

P4
Deny overrides

R7
Deny

R8
Permit

P5
First applicable

R9
Permit

PS2
Applicable
Permit overrides

PS3
Applicable
First applicable

PS1
N/A

PS2
N/A

PS3
Only one applicable

PS1
Deny overrides

PS2
Permit overrides

PS3
First applicable

N/A

N/A

N/A

N/A

N/A

N/A

N/A

Deny

N/A

Deny

N/A

Deny

N/A
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
N/A
Only one applicable

P1
N/A
First applicable

P2
N/A
Deny overrides

P3
Deny
Permit overrides

P4
Applicable
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example

- **R1**: Deny
- **R2**: Permit
- **R3**: Permit
- **R4**: Deny
- **R5**: Permit
- **R6**: Deny
- **R7**: Deny
- **R8**: Permit
- **R9**: Permit

**Policy Evaluation**

**PS1**
- N/A
- Only one applicable

**PS2**
- Applicable
- Permit overrides

**PS3**
- Applicable
- First applicable

**P1**
- N/A
- First applicable

**P2**
- N/A
- Deny overrides

**P3**
- Deny
- Permit overrides

**P4**
- Applicable
- Deny overrides

**P5**
- First applicable
Combining Algorithms: Example

PS3
Applicable
First applicable

PS1
N/A
Only one applicable

PS2
Applicable
Permit overrides

P1
N/A
First applicable

P2
N/A
Deny overrides

P3
Deny
Permit overrides

P4
Applicable
Deny overrides

P5
First applicable

R1
Deny

R2
Permit

R3
Permit

R4
Deny

R5
Permit

R6
Deny

R7
Deny

R8
Permit

R9
Permit
Combining Algorithms: Example
Combining Algorithms: Example

```
R1 Deny
R2 Permit
R3 Permit
R4 Deny
R5 Permit
R6 Deny
R7 Deny
R8 Permit
R9 Permit
```

```
P1 N/A
P2 N/A
P3 Deny
P4 Permit
P5 Permit
```

```
PS1 N/A
First applicable
Only one applicable
```

```
PS2 Permit
Permit overrides
```

```
PS3 Applicable
First applicable
```

```
N/A
Deny overrides
Permit overrides
First applicable
```

```
N/A
```

```
N/A
Deny
Permit
```

```
N/A
N/A
Permit
```

```
N/A
N/A
N/A
Permit
```

```
PS1 N/A
First applicable
Only one applicable
```

```
PS2 Permit
Permit overrides
```

```
PS3 Applicable
First applicable
```

```
N/A
Deny overrides
Permit overrides
First applicable
```

```
N/A
```

```
N/A
Deny
Permit
```

```
N/A
N/A
Permit
```

```
N/A
```

```
```
Combining Algorithms: Example

PS3
- Permit
  - First applicable

PS1
- N/A
  - Only one applicable

PS2
- Permit
  - Permit overrides

P1
- N/A
  - First applicable

P2
- N/A
  - Deny overrides

P3
- Deny
  - Permit overrides

P4
- Permit
  - Deny overrides

P5
- First applicable

R1
- Deny

R2
- Permit

R3
- Permit

R4
- Deny

R5
- Permit

R6
- Deny

R7
- Deny

R8
- Permit

R9
- Permit
Which Algorithm should I use?

- Deny-overrides
- Ordered-deny-overrides
- Permit-overrides
- Ordered-permit-overrides
- Deny-unless-permit
- Permit-unless-deny
- First-applicable
- Only-one-applicable
Evaluation Policy(set)

We have also to consider the target of a policy/policyset...

<table>
<thead>
<tr>
<th>Target</th>
<th>CA Value</th>
<th>Policy set Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match</td>
<td>( x )</td>
<td>( x )</td>
</tr>
<tr>
<td>No-match</td>
<td>Don’t care</td>
<td>NotApplicable</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>NotApplicable</td>
<td>NotApplicable</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>Permit</td>
<td>Indeterminate{P}</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>Deny</td>
<td>Indeterminate{D}</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>Indeterminate{PD}</td>
<td>Indeterminate{PD}</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>Indeterminate{P}</td>
<td>Indeterminate{P}</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>Indeterminate{D}</td>
<td>Indeterminate{D}</td>
</tr>
</tbody>
</table>
Obligations

- In XACML post obligations
- Rules, policies and policy sets may contain one or more `ObligationExpressions`
- `FulfillOn` determines when an obligation should be considered
- Obligations passed up to the next level of evaluation
- Obligations are passed up
  - if the rule, policy or policy set is evaluated
  - if decision matches `FulfillOn` attribute
Obligations: Example

- FulfillOn=Permit
- FulfillOn=Deny
- obligations to be fulfilled

PS1 o3
First applicable

P1 o1
First applicable
R1 Deny
R2 Permit
R3 Permit
R4 Deny
R5 Permit
R6 Deny
R7 Deny
R8 Permit
R9 Permit

PS2 o6
Permit overrides

P2 o2
Deny overrides

PS3 o8
First applicable

P3 o4
Permit overrides

P4 o5
Deny overrides

P5 o7
First applicable

PS1 Only one applicable
PS2 Permit overrides
PS3 Only one applicable
Obligations: Example

- o: FulfillOn=Permit
- o: FulfillOn=Deny
- o: obligations to be fulfilled

PS1 03 N/A
Only one applicable

PS2 06 Permit
Permit overrides

PS3 08 Permit
First applicable

P1 01 N/A
First applicable

P2 02 N/A
Deny overrides

P3 04 Deny
Deny overrides

P4 05 Permit
Permit overrides

P5 07 N/A
First applicable

R1 Deny
R2 Permit
R3 Permit
R4 Deny
R5 Permit
R6 Deny
R7 Deny
R8 Permit
R9 Permit
Obligations: Example

- **o**: FulfillOn=Permit
- **o**: FulfillOn=Deny
- **o**: obligations to be fulfilled

**PS3**
- **08**
- Permit
- First applicable

**PS1**
- **03**
- Only one applicable

**PS2**
- **06**
- Permit
- Permit overrides

**P1**
- **01**
- First applicable

**P2**
- **02**
- Deny overrides

**P3**
- **04**
- Deny
- Permit overrides

**P4**
- **05**
- Permit
- Deny overrides

**P5**
- **07**
- First applicable

**R1**
- Deny

**R2**
- Permit

**R3**
- Permit

**R4**
- Deny

**R5**
- Permit

**R6**
- Deny

**R7**
- Deny

**R8**
- Permit

**R9**
- Permit
Obligations: Example

- o: FulfillOn=Permit
- o: FulfillOn=Deny
- o: obligations to be fulfilled

PS1
- o3
- Only one applicable

PS2
- o6
- Permit overrides

PS3
- o8
- Permit
- First applicable

P1
- o1
- First applicable

P2
- o2
- Deny overrides

P3
- o4
- Deny

P4
- o5
- Permit overrides

P5
- o7
- First applicable

R1
- Deny

R2
- Permit

R3
- Permit

R4
- Deny

R5
- Permit

R6
- Deny

R7
- Deny

R8
- Permit

R9
- Permit
Obligations: Example

- o: FulfillOn=Permit
- o: FulfillOn=Deny
- o: obligations to be fulfilled

P1
  - o1
  - First applicable
  - N/A

P2
  - o2
  - Deny overrides
  - N/A

P3
  - o4
  - Deny
  - Permit overrides
  - N/A

P4
  - o5
  - Deny
  - Permit overrides
  - N/A

P5
  - o7
  - First applicable
  - N/A

R1
  - Deny

R2
  - Permit

R3
  - Permit

R4
  - Deny

R5
  - Permit

R6
  - Deny

R7
  - Deny

R8
  - Permit

R9
  - Permit

PS1
  - o3
  - Only one applicable
  - N/A

PS2
  - o6
  - Permit
  - Permit overrides
  - {o5}

PS3
  - o8
  - Permit
  - First applicable
  - {o5}
Obligations: Example

- o: FulfillOn=Permit
- o: FulfillOn=Deny
- o: obligations to be fulfilled

**PS1**

- N/A
- Only one applicable

**P1**

- N/A
- First applicable

**P2**

- N/A
- Deny overrides

**P3**

- N/A
- Deny
- Permit

**P4**

- N/A
- Deny
- Permit

**P5**

- N/A
- Permit

**PS2**

- N/A
- Permit overrides

**P1**

- N/A

**PS3**

- N/A
- Permit
- First applicable
- {o5,o8}

- **PS3**

- N/A

**P2**

- N/A

**P3**

- N/A

**P4**

- N/A

**P5**

- N/A

**R1**

- N/A

**R2**

- N/A

**R3**

- N/A

**R4**

- N/A

**R5**

- N/A

**R6**

- N/A

**R7**

- N/A

**R8**

- N/A

**R9**

- N/A

- {o4}

- {o5}

- {o5, o8}

- N/A

- N/A

- N/A

- N/A

- Deny

- Permit

- Deny

- Permit

- Deny

- Permit

- Deny

- Permit

- Permit

- Permit
One more thing about obligations

- only obligations associated with evaluation paths
- When nondeterminism is unacceptable, use deterministic combining algorithms
  - ordered-deny-overrides
  - ordered-permit-overrides
- Specification of policies
- Applicability of policies
- Access decision
- Combining algorithms
- Obligations

Response?
〈Result〉

Represent authorization decision

- 〈Decision〉 [Required]
  - The authorization decision: Permit, Deny, Indeterminate or NotApplicable

- 〈Status〉 [Optional]
  - Indicate whether errors occurred during evaluation

- 〈Obligations〉 [Optional]
  - list of obligations that MUST be fulfilled by the PEP

- 〈AssociatedAdvice〉 [Optional]
  - list of advice that provide supplemental information to the PEP

- 〈PolicyIdentifierList〉 [Optional]
  - If the ReturnPolicyIdList attribute in the 〈Request〉 is true, PDP MUST return a list of all policies which were found to be fully applicable. That is, all policies where both the 〈Target〉 matched and the 〈Condition〉 evaluated to true, whether or not the 〈Effect〉 was the same or different from the 〈Decision〉.
Encapsulate authorization decision made by PDP

Include a sequence of one or more results

⟨Result⟩ [One to Many]
  ▶ authorization decision
  ▶ one ⟨Result⟩ element per requested resource
Exercise (Text is in the repository)
Hints...

- Build policy hierarchy
- Annotate it with relevant information
  - Target
  - Combination algorithm
  - Obligations (and FulfillOn)
- Match target with request context
- Response = decision + set of obligations
Exercise: Request

Access request

Subject
Alice
Nurse

Resource
med:record

Action
read
Exercise: Policy

P1
Deny-overrides
A1:read
O1
O2

P2
First-applicable

P1
Permit-overrides
R1: med:record
O4
O5

R1
Permit
S1:physician
S2:nurse

R2
Deny
S1:physician

R3
Deny
S1:nurse
A1:write

R4
Permit
S1:nurse
A1:read
Exercise

Access Request
Alice
nurse
med:record
read

**P1**
Deny-overrides
A1:read
O1
O2

**P2**
First-applicable
O3

**R1**
Permit
S1:physician
S2:nurse

**R2**
Deny
S1:physician

**R3**
Deny
S1:nurse
A1:write

**R4**
Permit
S1:nurse
A1:read

**PS**
Permit-overrides
R1: med:record
O4
O5
Exercise

Access Request
Alice
nurse
med:record
read

P1
Deny-overrides
A1:read
O1
O2

P2
First-applicable
O3

R1
Permit
S1:physician
S2:nurse

R2
Deny
S1:physician

R3
Deny
S1:nurse
A1:write

R4
Permit
S1:nurse
A1:read

PS
Permit-overrides
R1: med:record
O4
O5

Applicable
Exercise

Access Request
Alice
nurse
med:record
read

Applicable

PS
Permit-overrides
R1: med:record
O4
O5

Applicable

P1
Deny-overrides
A1:read
O1
O2

P2
First-applicable
O3

R1
Permit
S1:physician
S2:nurse

R2
Deny
S1:physician

R3
Deny
S1:nurse
A1:write

R4
Permit
S1:nurse
A1:read
**Exercise**

**Access Request**
- Alice
- nurse
- med:record
- read

**Applicable**
- **P1**
  - Deny-overrides
  - A1:read
  - O1
  - O2

**Applicable**
- **P2**
  - First-applicable
  - O3

**Permit**
- **R1**
  - Permit
  - S1:physician
  - S2:nurse

**Deny**
- **R2**
  - Deny
  - S1:physician

**Deny**
- **R3**
  - S1:nurse
  - A1:write

**Permit**
- **R4**
  - S1:nurse
  - A1:read

**Permit-overrides**
- R1: med:record
- O4
- O5

**Deny-overrides**
- A1:read

**First-applicable**
Exercise

Access Request
Alice nurse med:record read

Applicable
PS
Permit-overrides
R1: med:record
O4 O5

Applicable
P1
Deny-overrides
A1:read
O1 O2

Permit
R1 Permit
S1:physician
S2:nurse

N/A
R2 Deny
S1:physician

Applicable
P2
First-applicable
O3

Deny
R3
S1:physician
A1:write

Permit
R4 Permit
S1:nurse
A1:read
Exercise

Access Request
Alice nurse med:record read

Applicable

PS Permit-overrides
R1: med:record O4 O5

P1 Deny-overrides
A1:read O1 O2

R1 Permit
S1:physician S2:nurse

R2 Deny
S1:physician

R3 Deny
S1:nurse A1:write

R4 Permit
S1:nurse A1:read

Permit {01}

N/A

P2 First-applicable O3

P1 Permit

O4

O5

Deny-overrides
A1:read

O1

O2

Deny

S1:physician

Deny

S1:nurse A1:write

Permit

Permit-overrides
R1: med:record
Exercise

Access Request
Alice
nurse
med:record
read

Permit {01}

PS
Permit-overrides
R1: med:record
O4
O5

Applicable

P1
Deny-overrides
A1:read
O1
O2

N/A

P2
First-applicable
O3

Applicable

Permit

R1
Permit
S1:physician
S2:nurse

Deny
S1:physician

R3
Deny
S1:nurse
A1:write

R4
Permit
S1:nurse
A1:read
Exercise

Access Request
Alice
nurse
med:record
read

Permit {01}
P1
Deny-overrides
A1:read
O1
O2

Permit
R1
Permit
S1:physician
S2:nurse

N/A
R2
Deny
S1:physician

N/A
R3
Deny
S1:nurse
A1:write

N/A
R4
Permit
S1:nurse
A1:read

Applicable
PS
Permit-overrides
R1: med:record
O4
O5

Applicable
P2
First-applicable
O3

Deny-overrides
P1
A1:read
O1
O2

First-applicable
P2
O3

Applicable
PS
N/A

Permit
N/A

N/A

N/A
Exercise

Access Request
Alice
nurse
med:record
read

Permit {01}
P1
Deny-overrides
A1:read
O1
O2

Permit
R1
Permit
S1:physician
S2:nurse

N/A
R2
Deny
S1:physician

N/A
R3
Deny
S1:nurse
A1:write

Permit
R4
Permit
S1:nurse
A1:read
Exercise

Access Request
Alice
nurse
med:record
read

Permit {01}
P1
Deny-overrides
A1:read
O1
O2

Permit {O3}
P2
First-applicable
O3

R1
Permit
S1:physician
S2:nurse

R2
Deny
S1:physician

R3
Deny
S1:nurse
A1:write

R4
Permit
S1:nurse
A1:read
Exercise

Access Request
Alice
nurse
med:record
read

Permit {01, 04} {03, 04}
PS
Permit-overrides
R1: med:record
O4
O5

P1
Deny-overrides
A1:read
O1
O2

Permit {01}
R1
Permit
S1:physician
S2:nurse

N/A
R2
Deny
S1:physician

N/A
R3
Deny
S1:nurse
A1:write

Permit
R4
Permit
S1:nurse
A1:read

Permit-overrides
R1: med:record
O4
O5

First-applicable
O3

Deny-overrides
A1:read
O1
O2

Deny
S1:physician

N/A
References

Homework (Deadline: 15/10/2019)

Text is in the repository