Privacy-aware Access Control II

Nicola Zannone
Access Control

- **Goal**: protect data and resources from unauthorized access
  - Confidentiality
  - Integrity
- **Policy**: define which actions a subject is allowed to perform on an object
- **Model**: define specification and enforcement of access control policies
  - DAC, MAC, RBAC, ABAC
Privacy Regulations

Impose stringent requirements on the collection, processing and disclosure of personal data

- Fair and lawful processing
- Purpose specification
- Consent
- Minimality
- Minimal disclosure

- Information quality
- Data subject’s control
- Sensitivity
- Information security

Purpose Specification

Personal data should be collected for specified, lawful and legitimate purposes and not processed in ways that are incompatible with the purposes for which data have been collected.
Beyond Access Control

- Access control focuses on the actions users perform on objects
- Privacy is more than confidentiality and integrity of data
- Privacy policies focus on the purpose for which data are used
  - “We will collect and use customer identifiable information for billing purposes and to anticipate and resolve problems with your service.”
- Notion of purpose must play a major role in access control
  - Access decisions should be made based on purpose
Privacy Policy

- **Who**: user identities or roles
- **What**: resources or data
- **How**: actions
- **Why**: the reason for which data are processed
- **Conditions**: under which the access is granted/denied
- **Obligations**: mandatory requirements to be fulfilled
Access Control

- Subject
- Object
- Action
- Purpose
- Conditions
- Obligations
Usage Control

- Subject
- Object
- Action
- **Purpose**
- Conditions
- Obligations
  - pre-obligations
  - ongoing-obligations
  - post-obligations
Privacy-Aware Access Control

- Access decisions based on purpose
  - Purpose: the reasons for data collection and access

Challenges

- Purpose Management
- Purpose Determination
- Purpose Control (Verification)
Outline

- Hippocratic Databases (Agrawal, 2002)
- Purpose-based Access Control (Byun and Li, 2008)
- Enterprise Privacy Authorization Language (IBM 2003)
  - Overview
  - Policy Specification
  - Policy Enforcement
  - Policy Management
Define the purpose hierarchy, role hierarchy, and access purpose authorizations in Purpose-based Access Control for the following scenario.

- A loan origination process consists of checking the financial information of the customer, making an offer and approving the loan.
- The loan origination process can only be performed during office hours.
- Bank employees (both managers and clerks) can verify financial information of customers.
- Any clerk can offer a loan of 1000$ or lower.
- Loans greater than 1000$ can be offered only by senior clerks.
- A senior clerk is a clerk with at least five years experience.
- Managers can approve a loan.
Solution

Define purpose hierarchy and role hierarchy along with role attributes for the scenario above.

**Purpose Hierarchy**

- General purpose
  - Loan origination
    - Financial information
      - Checking
    - Offer
    - Loan approval

**Role Hierarchy**

- ExpYear
  - Bank employee
    - Clerk
    - Manager
Solution

Purpose Hierarchy

- general purpose
  - loan origination
    - financial information
      - checking
    - offer
    - loan approval

Role Hierarchy

- ExpYear
  - bank employee
    - clerk
    - manager

Define the access purpose authorizations for the scenario.
Access purpose authorization is a pair \( \langle ap, cr \rangle \)

- \( ap \in \mathcal{P} \)
- \( cr \) is a conditional role
Solution

Purpose Hierarchy

Role Hierarchy

Access purpose authorizations:

\[\langle \text{financial information checking}, \langle \text{employee}, (\text{CurrentTime} \geq 9) \land (\text{CurrentTime} \leq 17) \rangle \rangle\]

\[\langle \text{offer}, \langle \text{clerk}, (\text{CurrentTime} \geq 9) \land (\text{CurrentTime} \leq 17) \land (\text{LoanAmount} \leq 1000) \rangle \rangle\]

\[\langle \text{offer}, \langle \text{clerk}, (\text{CurrentTime} \geq 9) \land (\text{CurrentTime} \leq 17) \land (\text{ExpYear} \geq 5) \rangle \rangle\]

\[\langle \text{loan approval}, \langle \text{manager}, (\text{CurrentTime} \geq 9) \land (\text{CurrentTime} \leq 17) \rangle \rangle\]
Outline

- Hippocratic Databases (Agrawal, 2002)
- Purpose-based Access Control (Byun and Li, 2008)
- Enterprise Privacy Authorization Language (IBM 2003)
  - Overview
  - Policy Specification
  - Policy Enforcement
  - Policy Management
EPAL: Overview
EPAL

- Enterprise Privacy Authorization Language
  - Platform for Enterprise Privacy Practices (E-P3P)

- Proposed by IBM

- Submitted to W3C for standardization (November 2003)
  - no action so far
Privacy Issues in Enterprises

- Enterprises store a large amount of personal data
- Large enterprises may not know what type of personal data are collected and where they are stored
- Enterprises may not know the consent a customer has given nor the legal regulations that apply to personal data
- Enterprises that process or store data collected by another enterprise are unable to enforce privacy consistently on behalf of the collecting enterprise
Privacy-enabled Data Management

Current situation
- Enterprises cannot give privacy guarantees
- Customers hesitate revealing personal data
- Legal problems; no business

Enterprises need to
- Adhere to legal regulations
- Obtain consent before using personal data
- Only use data for consented purposes
- Enable customers to retain control over their data
EPAL Goals (and Non-Goals)

- **Privacy Control**
  - Enable enterprise to enforce privacy promises made to customers
  - Prevent applications & employees from violating privacy

- **Privacy Violation Detection**
  - Off-line check whether privacy has been violated

- **Privacy Envelopes**
  - Transfer of policy-protected data

- **Use text or P3P instead**
  - Display privacy preferences
  - Collect consent
EPAL Framework

- **Formalize Privacy Policies**
  - Machine-readable language that can be enforced automatically
- **Formalize Policy Options**
  - Opt-in/opt-out options
- **Manage customer consent**
  - Privacy policies can be seen as a contract between customer and enterprise

- **Enforce policies**
  - Granting/denying access + enforce obligations
- **Compliance Audit**
  - Verify where data have been handled properly
EPAL: Privacy Policy Specification
Privacy Policies

- **Who**: users or roles
- **What**: resources or data
- **How**: actions
- **Why**: the reason for which data are processed
- **Conditions**: under which the access is granted/denied
- **Obligations**: actions to be taken before or after the access
The Sticky Policy Paradigm

Traditional access control

▶ “Let’s use the data for marketing!”
▶ “Wait a second, has the customer given the consent?”

Sticky policy paradigm

▶ “Check if marketing has been consented by the customer…”
▶ “If not, ask for consent first”
Enterprise Privacy Policies

- Data transmitted along with the policy regulating access to it
- Policies encompass a vocabulary and authorization rules
- Rules allow/deny privacy relevant actions, depending on purpose
EPAL Policy Model
Hierarchy

**Definition**

A hierarchy is a pair \((H, \succ)\)

- \(H\) is a finite set
- \(\succ \subseteq H \times H\) is a transitive, non-reflexive relation

\[H = \{User, Data, Purpose, Action\}\]

**User hierarchy**

- employee \(\succ\) manager
- employee \(\succ\) clerk
Obligation Model

Definition

An obligation model is a pair \((O, \to)\)

- \(O\) is a set of obligations
- \(\to \subseteq \mathcal{P}(O) \times \mathcal{P}(O)\) is a relation (imply) on the powerset of \(O\) where \(\overline{o_1} \to \overline{o_2}\) for all \(\overline{o_2} \subseteq \overline{o_1}\)

Some remarks:

- Fulfilling a set of obligations implies fulfilling all subsets
- Relation \(\to\) is transitive
EPAL Privacy Policies

- **Vocabulary** defines scope of policy:
  - Data, users, actions, and purposes as hierarchies
  - Obligations as lists

- **Ruleset** contains authorization rules:


  Example: “Email can be used for the book-of-the-month club if age is more than 13”

- default ruling: allow, deny, don’t care
- default obligation
- global condition
Vocabulary

Definition

A vocabulary is a tuple \((UH, DH, PH, AH, Var, OM)\)

- \(UH\) is a user hierarchy
- \(DH\) is a data hierarchy
- \(PH\) is a purpose hierarchy
- \(AH\) is an action hierarchy
- \(Var\) is a conditional vocabulary
- \(OM\) is an obligation model
**Ruleset**

**Definition**

A ruleset for a vocabulary $Voc$ is a subset of $U \times D \times P \times A \times \{+,-,\circ\} \times C(Var) \times \wp(O)$. A rule as the form $\langle (u, d, p, a), (r, c, \bar{o}) \rangle$ where

- $(u, d, p, a)$ is the **scope** of the rule
- $(r, c, \bar{o})$ is the **qualifier** of the rule

<table>
<thead>
<tr>
<th>privacy policy (informal)</th>
<th>A sales agent is allowed to collect a customer’s data for processing orders if the customer is older than 13 years of age. The data should be deleted after 3 years.</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>sales agent</td>
</tr>
<tr>
<td>data</td>
<td>customer record</td>
</tr>
<tr>
<td>purpose</td>
<td>order processing</td>
</tr>
<tr>
<td>action</td>
<td>collect</td>
</tr>
<tr>
<td>ruling</td>
<td>allow</td>
</tr>
<tr>
<td>condition</td>
<td>the customer is older than 13 years</td>
</tr>
<tr>
<td>obligation</td>
<td>delete data after 3 years</td>
</tr>
</tbody>
</table>
Privacy Policy

Definition

A privacy policy is a tuple \((Voc, R, gc, dr, do)\) where

- \(Voc\) is a vocabulary
- \(R\) is a ruleset defined over \(Voc\)
- \(gc \in C(Var)\) is a global condition
- \(dr \in \{+, -, \circ\}\) is default ruling
- \(do \in \wp(O)\) is a default obligation
Access request

<table>
<thead>
<tr>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A request is a tuple ((u, d, p, a))</td>
</tr>
<tr>
<td>- (u) is a user</td>
</tr>
<tr>
<td>- (d) is a data item</td>
</tr>
<tr>
<td>- (p) is a purpose</td>
</tr>
<tr>
<td>- (a) is an action</td>
</tr>
</tbody>
</table>

**Remark:** A request is **valid** for a vocabulary \(Voc\) if \(u, d, p, a \in Voc\)
EPAL: Policy Enforcement
Policy Enforcement

Policy maps any well-defined access request

\((\text{user, action, purpose, data}) + \text{variable assignment}\)

to decision \(\{\text{allow, deny, } \circ\} + \text{obligations}\)
Inheritance

- Allow inherits down along hierarchies
- Deny inherits up and down along hierarchies

Processing access request (user, data, purpose, action)

- Check whether there exists applicable rule(s)
  - that cover request directly or by inheritance
  - that satisfies condition(s)

Decision

- first applicable *allow* or *deny* rule
Matching Rules

Let $(u, d, p, a)\Box(u', d', p', a')$ iff $u \Box u'$, $d \Box d'$, $p \Box p'$, $a \Box a'$ for
\(\Box \in \{\geq, \geq\}\)

- \(\geq\) means down the hierarchies
- \(\leq\) means up and down the hierarchies

- A positive rule $\langle (u, d, p, a)(+, c, \bar{o}) \rangle$ matches a request $(u', d', p', a')$ iff $(u, d, p, a) \geq (u', d', p', a')$

- A negative rule $\langle (u, d, p, a)(-, c, \bar{o}) \rangle$ matches a request $(u', d', p', a')$ iff $(u, d, p, a) \leq (u', d', p', a')$

- An obligation rule $\langle (u, d, p, a)(\diamond, c, \bar{o}) \rangle$ matches a request $(u', d', p', a')$ iff $(u, d, p, a) \geq (u', d', p', a')$
Example

1. $\langle (u_3, d_1, p, a)(-, c_1, o_1) \rangle$ (red)
2. $\langle (u_2, d_2, p, a)(+, c_2, o_2) \rangle$ (green)
3. $\langle (u_1, d_0, p, a)(\circlearrowright, c_3, o_3) \rangle$ (empty)
Exercise: Rule Matching

1. $\langle (u_3, d_1, p, a)(-, c_1, o_1) \rangle$ (red)
2. $\langle (u_2, d_2, p, a)(+, c_2, o_2) \rangle$ (green)
3. $\langle (u_1, d_0, p, a)(\circ, c_3, o_3) \rangle$ (empty)

Which rules (if any) apply to the following authorization requests?

- $(u_1, d_1, p, a)$
- $(u_2, d_1, p, a)$
- $(u_3, d_2, p, a)$
Solution

1. \(\langle (u_3, d_1, p, a)(-, c_1, o_1) \rangle\) (red)
2. \(\langle (u_2, d_2, p, a)(+, c_2, o_2) \rangle\) (green)
3. \(\langle (u_1, d_0, p, a)(\circ, c_3, o_3) \rangle\) (empty)

Which rules (if any) apply to the following authorization requests?

- \((u_1, d_1, p, a)\)  \(R1, R3\)
- \((u_2, d_1, p, a)\)
- \((u_3, d_2, p, a)\)
Solution

1. $\langle (u_3, d_1, p, a)(-, c_1, \overline{c_1}) \rangle$ (red)
2. $\langle (u_2, d_2, p, a)(+, c_2, \overline{c_2}) \rangle$ (green)
3. $\langle (u_1, d_0, p, a)(\circ, c_3, \overline{c_3}) \rangle$ (empty)

Which rules (if any) apply to the following authorization requests?

- $(u_1, d_1, p, a)$ R1, R3
- $(u_2, d_1, p, a)$ None
- $(u_3, d_2, p, a)$
Solution

1. \(\langle (u_3, d_1, p, a)(-, c_1, \overline{o_1}) \rangle\) (red)
2. \(\langle (u_2, d_2, p, a)(+, c_2, \overline{o_2}) \rangle\) (green)
3. \(\langle (u_1, d_0, p, a)(\circ, c_3, \overline{o_3}) \rangle\) (empty)

Which rules (if any) apply to the following authorization requests?

- \((u_1, d_1, p, a)\) R1, R3
- \((u_2, d_1, p, a)\) None
- \((u_3, d_2, p, a)\) R3
Policy Evaluation

- Check rules in given order for applicability
  - rule covers request directly/by inheritance
  - condition/s are satisfied

- Decision
  - First applicable deny/allow-rule decides + rule’s obligation(s)
Policy Evaluation

- If request \( q \notin \text{Req}(Voc) \), return \( (\text{scope\_error}, \emptyset) \)
- If global condition is false, return \( (\text{policy\_error}, \emptyset) \)
- Process all rules in their order
  - If the rule is applicable
    - If the rule is an allow rule, return ruling allow and obligations in the rule together with accumulated obligations
    - If the rule is a deny rule, return ruling deny and obligations in the rule together with accumulated obligations
    - If the rule is a don’t care rule, accumulate obligations
  - If no (allow or deny) rules are applicable, return default ruling and default obligations together with accumulated obligations
Exercise: Policy Evaluation

1. \( \langle (u_3, d_1, p, a)(-, c_1, o_1) \rangle \) (red)
2. \( \langle (u_2, d_2, p, a)(+, c_2, o_2) \rangle \) (green)
3. \( \langle (u_1, d_0, p, a)(\circ, c_3, o_3) \rangle \) (empty)
4. Default ruling: +
5. Default obligation: \( o_4 \)

Evaluate the following access requests:

- \( (u_1, d_1, p, a) \)
- \( (u_2, d_1, p, a) \)
- \( (u_3, d_2, p, a) \)
Solution

1. $\langle (u_3, d_1, p, a)(-, c_1, \overline{o_1}) \rangle$ (red)
2. $\langle (u_2, d_2, p, a)(+, c_2, \overline{o_2}) \rangle$ (green)
3. $\langle (u_1, d_0, p, a)(\circ, c_3, \overline{o_3}) \rangle$ (empty)
4. Default ruling: +
5. Default obligation: $\overline{o_4}$

Evaluate the following access requests:

- $(u_1, d_1, p, a)$  R1, R3
- $(u_2, d_1, p, a)$  None
- $(u_3, d_2, p, a)$  R3
Solution

1. $\langle (u_3, d_1, p, a)(-, c_1, \overline{o_1}) \rangle$ (red)
2. $\langle (u_2, d_2, p, a)(+, c_2, \overline{o_2}) \rangle$ (green)
3. $\langle (u_1, d_0, p, a)(\circ, c_3, \overline{o_3}) \rangle$ (empty)
4. Default ruling: +
5. Default obligation: $\overline{o_4}$

Evaluate the following access requests:

- $(u_1, d_1, p, a)$ R1, R3 $-$, $\overline{o_1}$
- $(u_2, d_1, p, a)$ None
- $(u_3, d_2, p, a)$ R3
1. \(\langle (u_3, d_1, p, a)(-, c_1, \overline{o_1})\rangle\) (red)
2. \(\langle (u_2, d_2, p, a)(+, c_2, \overline{o_2})\rangle\) (green)
3. \(\langle (u_1, d_0, p, a)(\circ, c_3, \overline{o_3})\rangle\) (empty)
4. Default ruling: +
5. Default obligation: \(\overline{o_4}\)

Evaluate the following access requests:
- \((u_1, d_1, p, a)\)  \(R1, R3\)  \(\sim, \overline{o_1}\)
- \((u_2, d_1, p, a)\)  None  \(+, \overline{o_4}\)
- \((u_3, d_2, p, a)\)  \(R3\)
Solution

1. \(\langle (u_3, d_1, p, a)(-, c_1, o_1)\rangle\) (red)
2. \(\langle (u_2, d_2, p, a)(+, c_2, o_2)\rangle\) (green)
3. \(\langle (u_1, d_0, p, a)(\circ, c_3, o_3)\rangle\) (empty)
4. Default ruling: +
5. Default obligation: \(\overline{o_4}\)

Evaluate the following access requests:

- \((u_1, d_1, p, a)\) R1, R3 \(-, \overline{o_1}\)
- \((u_2, d_1, p, a)\) None \(+, \overline{o_4}\)
- \((u_3, d_2, p, a)\) R3 \(+, \overline{o_3} \cup \overline{o_4}\)
EPAL: Policy Management
Policy Management

- Policy refinement
  - A policy refines another policy if the first also satisfies the second
  - Compliance with legal regulations

- Policy composition
  - Notion of constructively combining two policies
  - Provide operators to construct policies
Policy Refinement

Refinement means adding details to an existing policy while preserving the original privacy statements:

- **Ruling**: Whenever the original policy allows (denies) a request, the refined policy also allows (denies) the request

- **Obligation**: Fulfillment of the refined obligations implies fulfillment of the original obligations for every request
Policy Refinement: Ruling

What does it mean that $r_1$ refines $r_2$ ($r_1 < r_2$)?

- If $r_2 \in \{\text{deny, allow}\}$ then $r_1 = r_2$
  - (weak form also: $r_2 = \text{allow}$ and $r_1 = \text{deny}$)

- If evaluation of $r_2$ returns $(\text{scope\_error, } \emptyset)$, i.e. the request is not defined over the policy vocabulary, then $r_1$ can be arbitrary

- If evaluation of $r_2$ returns $(\text{policy\_error, } \emptyset)$, i.e. $gc_2$ evaluates false, then evaluation of $r_1$ returns $(\text{policy\_error, } \emptyset)$

- If $r_2 = \text{don’t care}$ then $r_1 \in \{\text{deny, allow, don’t care}\}$
Policy Refinement: Obligation

- Meaning of “$o_1$ refines $o_2$” slightly more complicated
- Simply requiring $o_1 \rightarrow o_2$ not suited when policies are defined over a different vocabulary
  - $o_1 \in O_1 \setminus O_2$ and $o_2 \in O_2 \setminus O_1$
  - $o_1 \rightarrow o_2$ cannot be deduced
- “$o_1$ refines $o_2$” if there exists $o \in O_1 \cap O_2$ such that $o_1 \rightarrow o \rightarrow o_2$
  - $O1$: $o_1 =$ “delete now”, $o =$ “delete in a day” with $o_1 \rightarrow o$
  - $O2$: $o =$ “delete in a day”, $o_2 =$ “delete in a week” with $o \rightarrow o_2$
Exercise: Policy Refinement (1)

Policy 1 refines Policy 2?

Policy 1
\[
\langle (u_1, d, p, a)(+, c_1, \bar{0}) \rangle \\
\langle (u_6, d, p, a)(-, c_2, \bar{0}) \rangle \\
\langle (u_2, d, p, a)(+, c_3, \bar{0}) \rangle \\
(dr, do)
\]

Policy 2
\[
\langle (u_1, d, p, a)(+, c_1, \bar{0}) \rangle \\
\langle (u_6, d, p, a)(-, c_2, \bar{0}) \rangle \\
(dr, do)
\]
Exercise: Policy Refinement (1)

Policy 1 refines Policy 2?

**Policy 1**
\[ \langle (u_1, d, p, a)(+, c_1, \overline{o}) \rangle \]
\[ \langle (u_6, d, p, a)(-, c_2, \overline{o}) \rangle \]
\[ \langle (u_2, d, p, a)(+, c_3, \overline{o}) \rangle \]
\( (dr, do) \)

**Policy 2**
\[ \langle (u_1, d, p, a)(+, c_1, \overline{o}) \rangle \]
\[ \langle (u_6, d, p, a)(-, c_2, \overline{o}) \rangle \]
\( (dr, do) \)
Exercise: Policy Refinement (2)

Policy 1 refines Policy 2?

Policy 1
\[\langle (u_1, d, p, a)(+, c_1, \overline{o}) \rangle\]
\[\langle (u_6, d, p, a)(-, c_2, \overline{o}) \rangle\]
\[\langle (u_2, d, p, a)(+, c_3, \overline{o}) \rangle\]
\[(dr, do)\]

Policy 2
\[\langle (u_1, d, p, a)(+, c_1, \overline{o}) \rangle\]
\[\langle (u_2, d, p, a)(-, c_2, \overline{o}) \rangle\]
\[(dr, do)\]
Exercise: Policy Refinement (2)

Policy 1 refines Policy 2?

**Policy 1**

\[
\langle (u_1, d, p, a)(+, c_1, \bar{o}) \rangle \\
\langle (u_6, d, p, a)(-, c_2, \bar{o}) \rangle \\
\langle (u_2, d, p, a)(+, c_3, \bar{o}) \rangle \\
(dr, do)
\]

**Policy 2**

\[
\langle (u_1, d, p, a)(+, c_1, \bar{o}) \rangle \\
\langle (u_2, d, p, a)(-, c_2, \bar{o}) \rangle \\
(dr, do)
\]
Policy Refinement: Algorithm

- Evaluate both policies for any request and any assignment, and compare results
- Not efficient
  - several requests have the same matching rules
  - rules can have conditions that cannot be satisfied at the same times
  - several rules not applicable for particular request and assignment
Policy Refinement: Algorithm

**Goal**: transform policies to make the comparison easier

- Scope based expansion
  - ordered list of scope based rules
  - sequence of qualifiers
- Normalization of qualifier sequences
  - elimination of obligation ruling
- Comparison of qualifier sequence
- Comparison of extended rule-lists

See details and example on the paper (see reference at the end)
Scope-based expansion (1)

- Expand deny rules
  - Extend deny rules to all elements of hierarchies
  - Explicitly exclude elements that are not originally target of the rule

1. $(u_0, d_0, p, a), (dr, true, do)$
2. $(u_1, d_2, p, a), (−, c_1, o_1)$
3. $(u_2, d_1, p, a), (+, c_2, o_2)$
4. $(u_0, d_0, p, a), (−, c_1, o_1)$
Scope-based expansion (2)

- Float down obligation rules
  - No overlap with the next lower rule: the rules are swapped
  - If the scope of the target rule is contained in the scope of the next rule: the qualifier of the next rule is appended to the target rule’s qualifier
  - If the scope of the next rule is contained in the scope of the target rule: the rules are swapped and the qualifier of the target rule is appended to the qualifier sequence of the next rule.
  - If the rules partially overlap: the rules are swapped and a new rule is added to deal with the overlap

\[
\text{overlap}(\langle(u_1, d_1, p_1, a_1), \text{seq}_1\rangle, \langle(u_2, d_2, p_2, a_2), \text{seq}_2\rangle) = \langle(u, d, p, a), \text{seq}_1\rangle
\]

where \(u = \begin{cases} u_1 & \text{if } u_1 \leq u_2 \\ u_2 & \text{otherwise} \end{cases}\)

and similarly for the other dimensions.
Scope-based expansion (3)

- Float up positive rules until a rule whose scope comprises the scope of the positive rule is reached
- Scope-based rule list
  - rules matching for a small number of elements come first
  - extended rules: scope + sequence of qualifiers
    - \( \langle (u, d, p, a), (r_1, c_1, o_1) \rangle \) and \( \langle (u, d, p, a), (r_2, c_2, o_2) \rangle \) then \( \langle (u, d, p, a), \langle (r_1, c_1, o_1); (r_2, c_2, o_2) \rangle \rangle \)
- New rule set is equivalent to the original rule set
Normalization of qualifier sequences

- Remove obligation rules by shifting obligations
- Remove qualifiers “hidden” beyond a qualifier with higher precedence
- Transformation rules

\[
\frac{(\circ, c_1, o_1); (r, c_2, o_2)}{(r, c_1, o_1 \& o_2)} \quad c_1 \Rightarrow c_2, \quad r \in \{+, -\}
\]

\[
\frac{(\circ, c_1, o_1); (r, c_2, o_2)}{(r, c_1 \& c_2, o_1 \& o_2); (r, c_2, o_2); (\circ, c_1, o_1)} \quad \neg(c_1 \Rightarrow c_2), \quad r \in \{+, -\}
\]

\[
\ldots
\]
Given two normalized, scope-based rule lists $SR_1$ and $SR_2$

- For each $\langle (u_2, d_2, p_2, a_2), seq_2 \rangle \in SR_2$, process $SR_1$ in descending precedence
- Check each overlapping rule $\langle (u_1, d_1, p_1, a_1), seq_1 \rangle \in SR_1$ whether $seq_2$ and $seq_1$ constitute a refinement
- If there is no refinement, return $false$
Summary

- Access control vs. Privacy
  - Protect information from unauthorized access
  - No control on how information is intended to be used
- Privacy-Aware Access Control
  - Access decision based on purpose
- Hippocratic Databases
  - Metadata for the specification of privacy policy and privacy authorization tables
- Purpose-Based Access Control
  - Purpose management
  - Access purpose determination
- Enterprise Privacy Authorization Language
References


- EPAL 1.2 submission to the W3C 10 Nov 2003. Available at http://www.w3.org/Submission/2003/SUBM-EPAL-20031110/ (suggested)
Homework (Deadline: 10/10/2019)

- Study policy comparison algorithm
- Let \(\text{pol}\) be an EPAL policy defined over a vocabulary \(\text{Voc}\) where \(\text{Voc}\) consists of the user, data, purpose and action hierarchies below.

(a) User Hierarchy

(b) Data Hierarchy

(c) Purpose Hierarchy

(d) Action Hierarchy

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

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

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