Focus of our Research

- Client-side PETs to
  - minimize data disclosed
  - filter data received
  - keep track of data
  - control multiple identities
  - ...

- Infrastructure PETs to
  - hide relations
  - unlinkable credentials
  - ...

- What happens to the data once disclosed?
- How to enable businesses to work with pseudonyms?
- How to authenticate and authorize, relative to a pseudonym?
2b. E-P3P Applications and Non-Goals

Privacy Control:
- Preventing 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:
- Displaying Privacy Preferences
- Collecting consent

1a. 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 info
- Only use the data for consented purposes
- Enable the customers to retain control
1b. The Sticky Policy Paradigm

Today: (at most) one policy
- "Let's use these data for marketing!"
- "Wait a second, I'll update the policy..."

The future: sticky policy paradigm
- "Check the policy if marketing has been consented by this particular customer..."
- "If not, we ask for consent first"

1c. Enterprise Privacy Policies

EPAL policy
- EPAL defines policy terminology and authorization rules
- Rules allow/deny privacy relevant actions, depending on purpose
- Can be mapped to P3P, supporting consistent internal/external views
2a. EPAL by Example

Privacy promise:
- "Email can only be used for the book-of-the-month club if consent has been given and age is more than 13":

EPAL rule:
- <ALLOW
  data-user="borderless-books"
data-category="email"
purpose="book-of-the-month-club"
operation="read"
condition="/CustomerRecord/Consent/BookClub=True
&& /CustomerRecord/age>13">
From Privacy Promises to Enforcement

<table>
<thead>
<tr>
<th>Elements</th>
<th>P3P</th>
<th>EPAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories</td>
<td>list; predefined</td>
<td>hierarchy</td>
</tr>
<tr>
<td>Data-Users</td>
<td>list; predefined</td>
<td>hierarchy</td>
</tr>
<tr>
<td>Purposes</td>
<td>list; predefined</td>
<td>hierarchy</td>
</tr>
<tr>
<td>Operations</td>
<td>'use'</td>
<td>list</td>
</tr>
<tr>
<td>Conditions</td>
<td>none</td>
<td>yes</td>
</tr>
<tr>
<td>Obligations</td>
<td>'retention'</td>
<td>yes</td>
</tr>
<tr>
<td>Choices</td>
<td>+/- purpose</td>
<td>generalized</td>
</tr>
<tr>
<td>Deployment</td>
<td>none</td>
<td>cross-application</td>
</tr>
</tbody>
</table>

**Conclusion**
- Simple
- Interoperable
- Non-extensible
- Limited
- Sector-specific
- Privacy Control
- Interoperable within each Sector

2b. EPAL Syntax

Elements: EPAL element definitions define scope:
- Data users, purposes, and categories are **hierarchies**
- Operations, obligations, and conditions are **lists**

Rules: EPAL rules authorize access:
- \((d\text{-category}, d\text{-user}, \text{purpose}, \text{operation}, \text{condition}, \text{obligation})\)
2b. Example: An EPAL Rule in XML

```xml
<rule id="rule1" precedence="5" ruling="allow">
  <short-description>Models Par 8 of HIPPAA law.</short-description>
  <data-user id="sales"/>
  <data-category id="financial"/>
  <purpose id="email marketing"/>
  <action id="read"/>
  <condition id="consentToMarketing"/>
  <obligation id="retention">
    <parameter id="days">5</parameter>
  </obligation>
</rule>
```

2c. EPAL Semantics

**Inheritance:**
- Allow inherits down along hierarchies
- Deny inherits up and down (group-like)

**Processing request (d-category, d-user, purpose, operation):**
- Check whether there exists applicable rule(s)
  - that cover request directly or by inheritance
  - have satisfied condition(s)
- Decide:
  - Allow and deny rules → Ignore allow rules
  - Either Allow or deny rules → Choose one for ruling + obligations
  - No rule → Default ruling (with no obligations)
2c. EPAL Semantics

Inheritance:
- Allow inherits down along hierarchies
- Deny inherits up and down (group-like)

Processing request (d-category, d-user, purpose, operation)
- Check whether there exists applicable rule(s)
  - that cover request directly or by inheritance
  - have satisfied condition(s)
- Decide:
  - Allow and deny rules → Ignore allow rules
  - Either Allow or deny rules → Choose one for ruling + obligations
  - No rule → Default ruling (with no obligations)

Deploying Vocabularies

Application-specific Terms
- application + task
- storage location
- data
- credentials
- operation
- (stored) procedures

(Privacy) Policy Terms
- (purpose)
- data-user
- data-category
- (privacy) action
- condition
- obligation

Requirements:
- Application-independent Policies 'CPO driven'
- Enforce this policy across multiple applications

Problems:
- Applications should not know the policy
- Applications use their own terminology
- Applications know nothing about 'purposes'
References and Contact

IBM Privacy Institute

IBM Zurich Research Lab

How to reach us:
- wmi@zurich.ibm.com
- (+41) 1 724 8220

Questions?