Privacy by Design
Strategies & Patterns

Jaap-Henk Hoepman

Digital Security (DS)
Radboud University Nijmegen, the Netherlands
@xotoxot // ⭐️ jhh@cs.ru.nl // 🌐 www.cs.ru.nl/~jhh
Introduction

- Security
- Privacy
- Identity Management
- Internet of Things
Software development cycle

Concept Development

Analysis

Design

Implementation

Testing

Evaluation
Privacy by design

- Protect privacy during technology development:
  - From conception...
  - ... to realisation.

Through the full product development lifecycle
Levels of abstraction

- **Design pattern**
  - “Commonly recurring structure to solve a general design problem within a particular context”
Software development cycle

- Concept Development
- Analysis
- Design
- Implementation
- Testing
- Evaluation

Design patterns
Levels of abstraction

- **Design pattern**
  - “Commonly recurring structure to solve a general design problem within a particular context”

- **(Privacy enhancing) technology**
  - “A coherent set of ICT measures that protects privacy” – implemented using concrete technology
Software development cycle

- Concept Development
- Analysis
- Design
- Implementation
- Testing
- Evaluation

Privacy enhancing technologies

Design patterns
Levels of abstraction

- **Design strategy**
  - “A basic method to achieve a particular design goal” – *that has certain properties that allow it to be distinguished from other basic design strategies*

- **Design pattern**
  - “Commonly recurring structure to solve a general design problem within a particular context”

- **(Privacy enhancing) technology**
  - “A coherent set of ICT measures that protects privacy” – *implemented using concrete technology*
Concept development & analysis

Privacy Impact Assessment

Concept Development

Analysis

Privacy Design Strategies
Eight Privacy Design Strategies
Source #2: data protection law

Core principles

- Data minimisation
- Purpose limitation
- Proportionality
- Subsidiarity
- Data subject rights: consent, (re)view
- Adequate protection
- (Provable) Compliance
What happens if we want to apply these data protection principles to an information storage (ie database) system?
## Database tables

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individuals</td>
</tr>
</tbody>
</table>

- minimise
- separate
- aggregate
- hide
8 privacy design strategies

- **Minimise**
  - The amount of PII should be minimal

- **Separate**
  - Process PII in a distributed fashion

- **Aggregate**
  - Process PII in the least possible detail

- **Hide**
  - PII should not be stored in plain view
What did we cover

**Core principles**
- Data minimisation
- Purpose limitation
- Proportionality
- Subsidiarity
- Data subject rights: consent, (re)view
- Adequate protection
- (Provable) Compliance

**Design strategies**
- Minimise
- Separate
- Aggregate
- Hide
What did we cover

Core principles

- Data minimisation
- Purpose limitation
- Proportionality
- Subsidiarity
- Data subject rights: consent, (re)view
- Adequate protection
- (Provable) Compliance

Design strategies

- Minimise
- Separate
- Aggregate
- Hide
What did we cover

- **Core principles**
  - Data minimisation
  - Purpose limitation
  - Proportionality
  - Subsidiarity
  - Data subject rights: consent, (re)view
  - Adequate protection
  - (Provable) Compliance

- **Design strategies**
  - Minimise
  - Separate
  - Aggregate
  - Hide
  - Enforce
  - Inform
  - Control
  - Demonstrate
8 privacy design strategies

- Minimise
  - The amount of PII should be minimal

- Separate
  - Process PII in a distributed fashion

- Aggregate
  - Process PII in the least possible detail

- Hide
  - PII should not be stored in plain view

- Enforce
  - A privacy policy should be in place and be enforced

- Inform
  - Subjects should be informed when PII is processed

- Control
  - Subjects should have control over when/how PII is processed

- Demonstrate
  - Compliance to policies and legal requirements must be demonstrated
What about design patterns?

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Patterns</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimise</td>
<td>Select before you collect, anonymisation, ….</td>
<td></td>
</tr>
<tr>
<td>Separate</td>
<td>Distribute, sector-specific pseudonyms</td>
<td></td>
</tr>
<tr>
<td>Aggregate</td>
<td>Data fuzzing; coarse-grained location</td>
<td></td>
</tr>
<tr>
<td>Hide</td>
<td>Encryption, onion routing, ….</td>
<td></td>
</tr>
<tr>
<td>Enforce</td>
<td>Access control, privacy licenses</td>
<td></td>
</tr>
<tr>
<td>Inform</td>
<td>P3P (?)</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Informed consent (?)</td>
<td></td>
</tr>
<tr>
<td>Demonstrate</td>
<td>Privacy management system, logging</td>
<td></td>
</tr>
</tbody>
</table>
“Provides a scheme for refining the subsystems or components of a software system, or the relationships between them. It describes a commonly recurring structure of communicating components that solves a general design problem within a particular context.”
(Privacy) Design Pattern Template

- **Name**
- **Intent**
- **Application context**
  - Including the problem it aims to solve
- **Implementation**
  - components & relationships
- **Consequences / Forces & Concerns**
  - Results, side-effects, trade offs
  - When to apply
  - When not to apply
- **Examples**
- **Related patterns**
Contribute
- http://wiki.science.ru.nl/privacy/Main_Page

See also: www.pilab.nl