Privacy Architectures: Reasoning About Data Minimisation and Integrity

Thibaud Antignac
Daniel Le Métayer

Cappris October 2014

PRIVATICS
Inria Rhône-Alpes
CITI lab / INSA Lyon

Formal methods for a systematic approach to privacy by design

- design space for privacy-friendly systems is wide
  - ➡ designers need support tools and methods
- privacy properties are complex and may be conflicting
  - ➡ formal models can help designers to reason and choose
- protocol level contains too many details
  - ➡ architecture is the right level of abstraction

Privacy by design

Advocated by lawyers
To care from the early stage
Data minimisation

Advocated by lawyers

- **Soft law**
  - *The Future of Privacy,*
    Article 29 Working Party position, 2009
  - *Protecting Consumer Privacy in an Era of Rapid Change,*
    FTC report, 2012
  - *Guidelines governing the Protection of Privacy* […],
    15(a)(iii) memorandum, **OECD** recommendation, 2013

- **Hard law**
  - *EU General Data Protection Regulation*,
    European Commission amended proposal as voted by **European Parliament**, 2014

To care from the early stage

- **prevention** rather than cure

- embedded within the **entire life cycle**
  - **early design stage**
  - deployment
  - use
  - final disposal

- **responsibility** by the controller or processor

[*Privacy by Design: The 7 Foundational Principles*, Cavoukian, 2009]
[Article 61, *EU General Data Protection Regulation*, European Commission amended proposal as voted by European Parliament, 2014]

Data minimisation

- collection **limited to** the **minimum necessary** for the purposes
- only if the purposes **could not be fulfilled by** processing information **not involving personal data**

[Article 5, *EU General Data Protection Regulation*, European Commission amended proposal as voted by European Parliament, 2014]

Architecture

Departure from usual approaches
Architecture is the right level
Privacy architecture

Departure from usual approaches

- **many different** privacy-friendly protocol **proposals**
- **empirical process**
  - from the requirements
  - to an **ad-hoc** and **integrated** protocol
- need for an intermediary level
  - to **model properties** and **abstract** away the details
  - to **reason** about these properties
  - to cope with **conflicting requirements**
  - to **justify** choices
  - backed by **methods** and **tools**

Architecture is the right level

- number of **fundamental decisions** that **profoundly affect** the system and its development process

- **predicting** system qualities
- **early design** decisions
- defining **constraints** on an implementation
- supplying a **transferable**, **reusable model**
- incorporation of independently developed **components**

[*Software Architecture in Practice*, Bass, Clements & Kazman, 2012]
[*Software Engineering*, Sommerville, 2010]

Privacy architecture

Software Architecture = {Elements, Form, Rationale}

with Elements ∈ {**Processing** elements,
Data elements,
Connecting elements}

with Form = {**Properties**,
Relationships}

[*Foundations for the Study of Software Architecture*, Perry & Wolf, 1992]

Privacy architecture

Privacy architecture = {Elements, Form, Rationale}

with **Elements** ∈ {data subjects, controllers, and processors,
personal and non-personal data,
IT system}

with **Form** = {processing, collection, and transfer capabilities}

with **Rationale** = {process, verifications}

Reasoning system

Components
Privacy properties
Attack models and privacy risks
Inference rules
Integration within the development process

Elements

- agents can
  - collect data => **has_i (X)**
  - transfer data => **$receive_{i,j}$ (X)**
  - process data => **$compute_i$ (X = f(Y, Z))**
- agents can also
  - trust agents => **$trust_{i,j}$**
  - verify attestations => **$verif^{Attest}_i$ ($attest_j$ (X = T))**
- there are other primitives

Privacy properties

- is data
  - totally disclosed? => **has^{All}_i (X)**
  - not disclosed at all? => **has^{None}_i (X)**
  - partly disclosed? => **has^{One}_i (X)**

- is data
  - known correct? => **K_i (X = T)**
  - believed correct? => **B_i (X = T)**

Attack model and privacy risks

- agents can
  - deduce new values => $\mathbf{Dep_i\ (X, \{Y, Z\})}$ for X = f(Y, Z)
  - infer new relations => $\mathbf{\{X = Y, Y = Z\} \triangleright_i X = Z}$

[*Deductive Algorithmic Knowledge*, Pucella, 2006]

Inference rules

$$\frac{has_i\ (X)}{has^{All}_i\ (X)} \qquad \frac{receive_{i,j}\ (X)}{has^{All}_i\ (X)} \qquad \frac{compute_i\ (X = T)}{has^{All}_i\ (X)}$$

$$\frac{Dep_i\ (X, \{Y, Z\}) \quad A \vdash has^{All}_i\ (Y) \quad A \vdash has^{All}_i\ (Z)}{A \vdash has^{All}_i\ (X)}$$

Inference rules

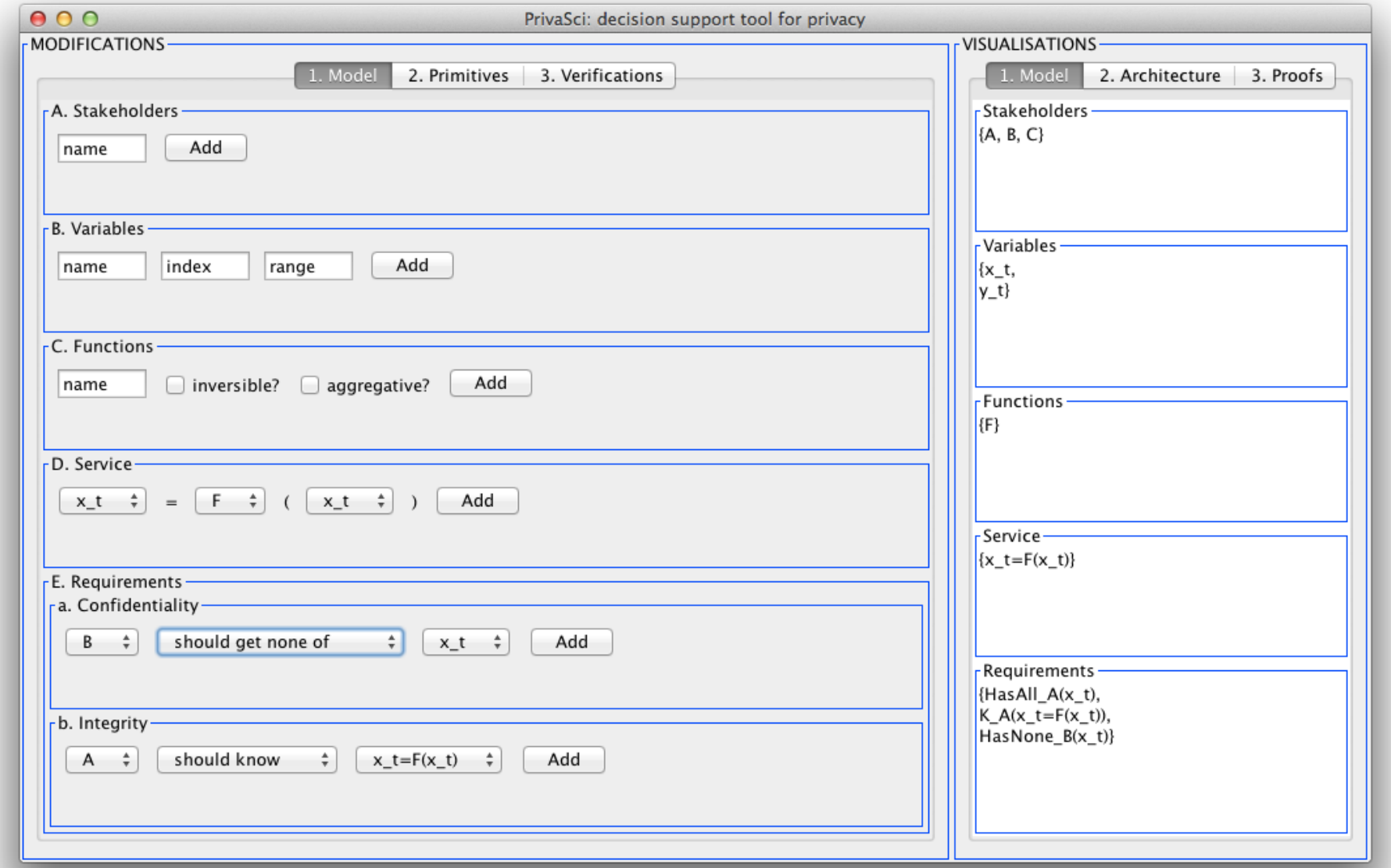
$$\frac{\text{compute}_i\ (X = T)}{K_i\ (X = T)} \qquad \frac{\text{verif}^{\text{Attest}}{}_i\ (\text{attest}_j\ (X = T)) \quad \text{trust}_{i,j}}{K_i\ (X = T)}$$

$$\frac{\{X = Y,\ Y = Z\} \triangleright_i X = Z \quad K_i\ (X = Y) \quad K_i\ (Y = Z)}{A \vdash K_i\ (X = Z)}$$

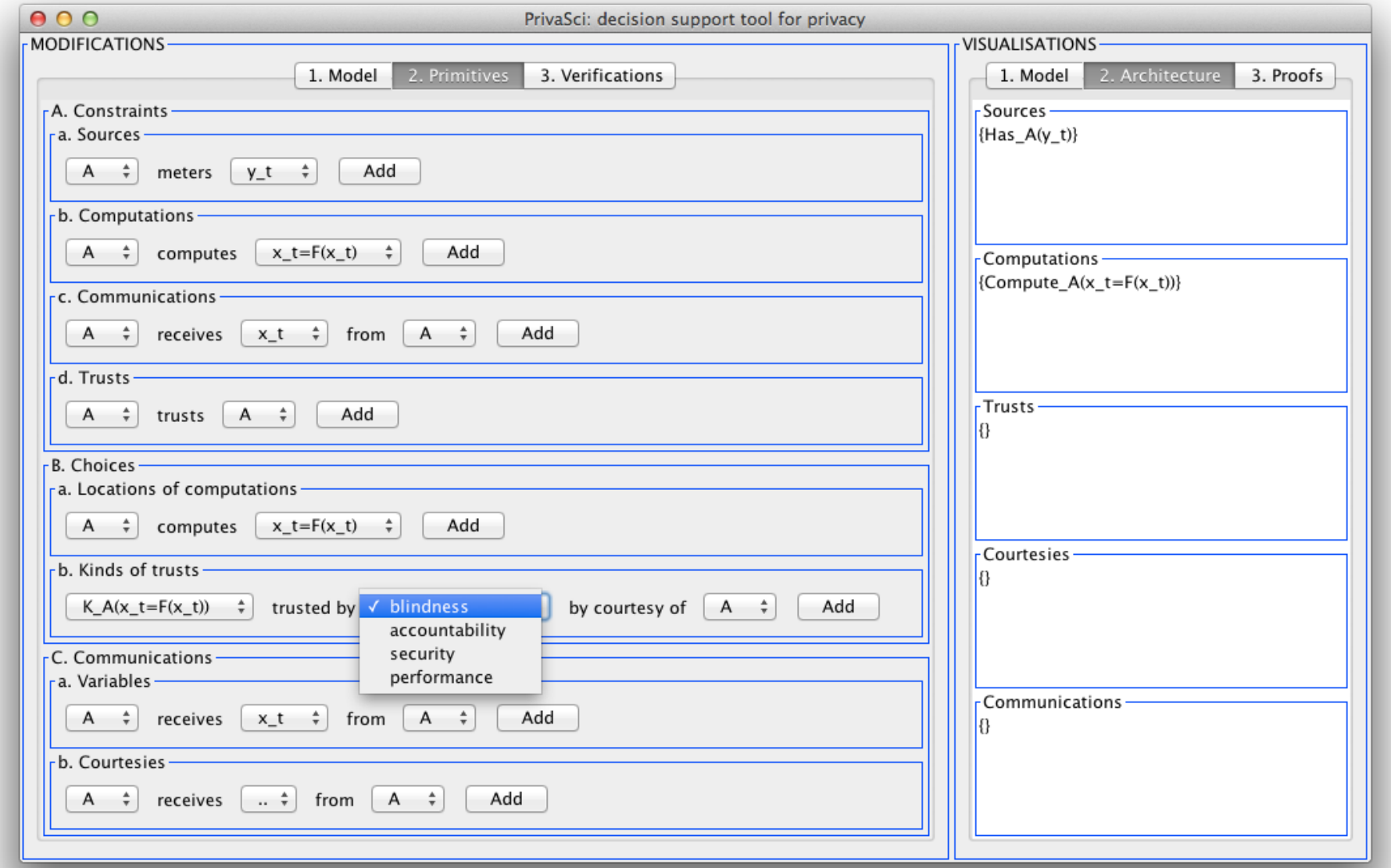
Integration within the development process

- 3 levels
  - requirements
  - architecture
  - verification
- help the designer
  - strategies Q&A to choose components
  - feedback
  - proof trees
- asset for documenting the system

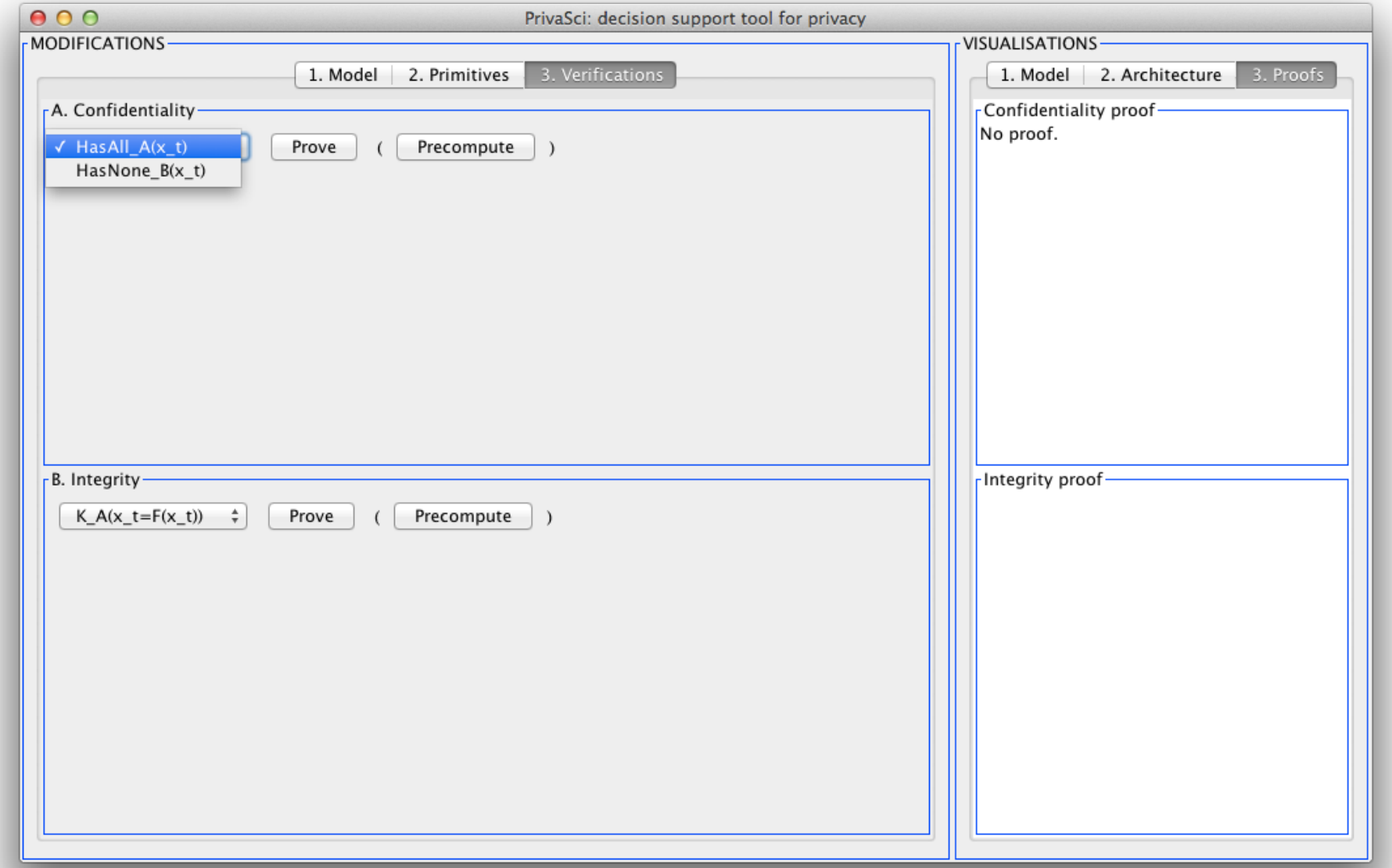
Integration within the development process



Integration within the development process



Integration within the development process



Smart metering case study

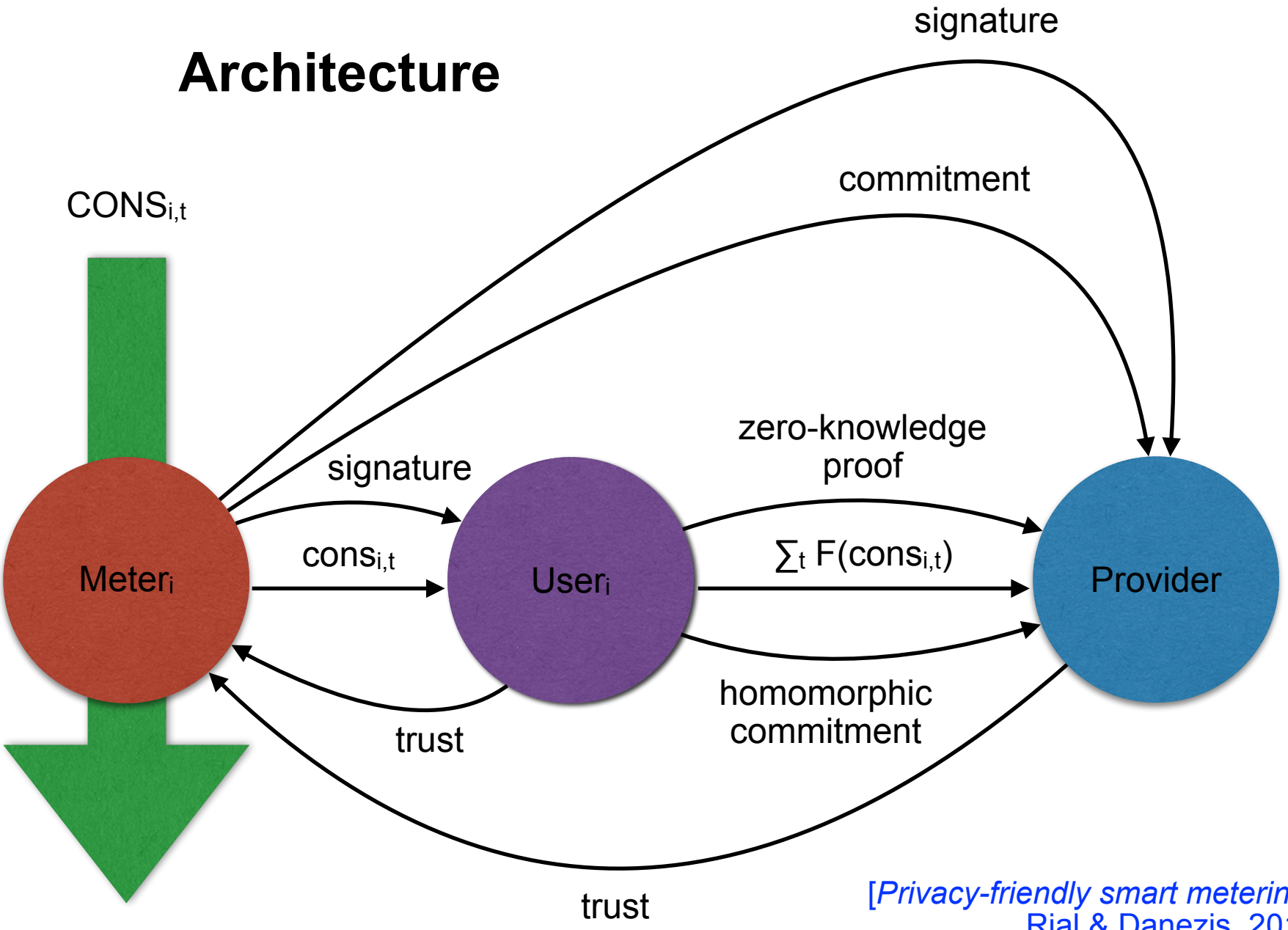
Requirements
Architecture
Verifications

Requirements

- **Confidentiality**
  - **the provider only needs the total fee for the bill period**
  - the user needs all the detailed consumption

- **Integrity**
  - **the provider must be sure the reported fee is correct**
  - the user must be sure the detailed consumption and the reported fee are correct

Architecture



[*Privacy-friendly smart metering*, Rial & Danezis, 2010]

Architecture

$\text{receive}_{\text{Provider,User}}$
$(\{\text{proof}_{\text{User}}\ (\text{attest}_{\text{Meter}}\ (\text{cons}_t = \text{CONS}_t))\}, \varnothing)$

$\text{receive}_{\text{User,Meter}}$
$(\{\text{attest}_{\text{Meter}}\ (\text{cons}_t = \text{CONS}_t)\}, \{\text{cons}_t\})$

$\text{receive}_{\text{Provider,User}}$
$(\{\text{proof}_{\text{User}}\ (\text{fee} = \sum_t F\ (\text{cons}_t)\}, \text{fee})$

$\text{has}_{\text{Meter}}\ (\text{CONS}_t)$

$\text{verif}^{\text{Attest}}{}_{\text{User}}$
$(\text{attest}_{\text{Meter}}\ (\text{cons}_t = \text{CONS}_t))$

$\text{verif}^{\text{Proof}}{}_{\text{Provider}}\ (\text{proof}_{\text{User}}$
$(\text{attest}_{\text{Meter}}\ (\text{cons}_t = \text{CONS}_t)))$

$\text{compute}_{\text{Meter}}$
$(\text{cons}_t = \text{CONS}_t)$

$\text{compute}_{\text{User}}$
$(\text{fee} = \sum_t F\ (\text{cons}_t))$

$\text{verif}^{\text{Proof}}{}_{\text{Provider}}$
$(\text{proof}_{\text{User}}\ (\text{fee} = \sum_t F\ (\text{cons}_t)))$

$\text{trust}_{\text{User,Meter}}$

$\text{trust}_{\text{Provider,Meter}}$

[*Privacy-friendly smart metering*, Rial & Danezis, 2010]

Verification: confidentiality for the provider

$$\frac{\text{receive}_{\text{Provider,User}}\ (\{\text{proof}_{\text{User}}\ (\text{fee} = \sum_t F\ (\text{cons}_t)\}, \text{fee})}{\text{has}^{\text{All}}_{\text{Provider}}\ (\text{fee})}$$

$$\frac{\text{No premises of other rules apply}}{\text{has}^{\text{None}}_{\text{Provider}}\ (\text{cons}_t)}$$

Verification: integrity for the provider

$$\frac{\mathrm{verif}^{\mathrm{Proof}}{}_{\mathrm{Provider}}\ (\mathrm{proof}_{\mathrm{User}}\ (\mathrm{attest}_{\mathrm{Meter}}\ (\mathrm{cons}_t = \mathrm{CONS}_t)))\quad \mathrm{trust}_{\mathrm{Provider,Meter}}}{K_{\mathrm{Provider}}\ (\mathrm{cons}_t = \mathrm{CONS}_t)}$$

$$\frac{\mathrm{verif}^{\mathrm{Proof}}{}_{\mathrm{Provider}}\ (\mathrm{proof}_{\mathrm{User}}\ (\mathrm{fee} = \textstyle\sum_t F\ (\mathrm{cons}_t)))}{K_{\mathrm{Provider}}\ (\mathrm{fee} = \textstyle\sum_t F\ (\mathrm{cons}_t))}$$

$$\frac{\{\mathrm{cons}_t = \mathrm{CONS}_t,\ \mathrm{fee} = \textstyle\sum_t F\ (\mathrm{cons}_t)\} \triangleright_{\mathrm{Provider}}\ \mathrm{fee} = \textstyle\sum_t F\ (\mathrm{CONS}_t) \qquad K_{\mathrm{Provider}}\ (\mathrm{cons}_t = \mathrm{CONS}_t)\quad K_{\mathrm{Provider}}\ (\mathrm{fee} = \textstyle\sum_t F\ (\mathrm{cons}_t))}{K_{\mathrm{Provider}}\ (\mathrm{fee} = \textstyle\sum_t F\ (\mathrm{CONS}_t))}$$

Future work & Conclusion

Future works

- Other privacy criteria
  - data retention
  - liability
  - data weakening
- Privacy patterns

Formal methods for a systematic approach to privacy by design

- design space for privacy-friendly systems is wide
  - ➡ designers need support tools and methods
- privacy properties are complex and may be conflicting
  - ➡ formal models can help designers to reason and choose
- protocol level contains too many details
  - ➡ architecture is the right level of abstraction

Thank you

PRIVATICS
Inria Rhône-Alpes
CITI lab / INSA Lyon
www.inria.fr

Formal system: architectural primitives

$$A ::= \{R\}$$

$$
\begin{aligned}
R ::= & \; Has_i\left(\tilde{X}\right) & \mid & \; Receive_{i,j}\left(\{S\}, \{\tilde{X}\}\right) \\
& \mid Compute_i\left(\tilde{X} = T\right) & \mid & \; Check_i\left(\{Eq\}\right) \\
& \mid Verif_i^{Proof}\left(Pro\right) & \mid & \; Verif_i^{Attest}\left(Att\right) \\
& \mid Spotcheck_{i,j}\left(X_k, Eq\right) & \mid & \; Trust_{i,j}
\end{aligned}
$$

$$S ::= Pro \mid Att$$

$$Pro ::= Proof_i\left(\{P\}\right)$$

$$P ::= Att \mid Eq$$

$$Att ::= Attest_i\left(\{Eq\}\right)$$

$$Eq ::= T_1 \; Rel \; T_2$$

$$Rel ::= \; = \mid < \mid > \mid \leq \mid \geq$$

Formal system: properties

$$\phi ::= Has_i^{all}\left(\tilde{X}\right) \mid Has_i^{none}\left(\tilde{X}\right) \mid Has_i^{one}\left(\tilde{X}\right)$$
$$\mid K_i\left(Eq\right) \quad \mid B_i\left(Eq\right) \quad \mid \phi_1 \wedge \phi_2$$
$$Eq ::= T_1\ Rel\ T_2 \mid Eq_1 \wedge Eq_2$$

Confidentiality

$$\textbf{H1}\ \frac{Has_i\left(\tilde{X}\right) \in A}{A \vdash Has_i^{all}\left(\tilde{X}\right)} \qquad \textbf{H2}\ \frac{Receive_{i,j}\left(S, E\right) \in A \quad \tilde{X} \in \{E\}}{A \vdash Has_i^{all}\left(\tilde{X}\right)}$$

$$\textbf{H3}\ \frac{Compute_i\left(\tilde{X} = T\right) \in A}{A \vdash Has_i^{all}\left(\tilde{X}\right)} \qquad \textbf{H4}\ \frac{Spotcheck_{i,j}\left(X_k, E\right) \in A}{A \vdash Has_i^{one}\left(X\right)}$$

$$\textbf{H5}\ \frac{Dep_i\left(\tilde{X}, \left\{\tilde{X}^1, \ldots \tilde{X}^n\right\}\right) \quad \text{for all } l \in [1, n], A \vdash Has_i^{all}\left(\tilde{X}^l\right)}{A \vdash Has_i^{all}\left(\tilde{X}\right)}$$

$$\textbf{H6}\ \frac{\text{None of the pre-conditions of H1, H2, H3, H4, or H5 holds for } X \text{ or any } X_k}{A \vdash Has_i^{none}\left(\tilde{X}\right)}$$

+ coherence and structural rules

Integrity

$$\mathbf{B}\ \frac{Spotcheck_{i,j}\,(X_k, E) \in A \qquad Eq \in E}{A \vdash B_i(Eq)}$$

$$\mathbf{K1}\ \frac{Compute_i\left(\tilde{X} = T\right) \in A}{A \vdash K_i(\tilde{X} = T)}$$

$$\mathbf{K3}\ \frac{Verif_i^{Proof}\left(Proof_j(E)\right) \in A \qquad Eq \in E}{A \vdash K_i(Eq)}$$

$$\mathbf{K2}\ \frac{Check_i\,(E) \in A \qquad Eq \in E}{A \vdash K_i(Eq)}$$

$$\mathbf{K4}\ \frac{Verif_i^{Proof}\left(Proof_j(E)\right) \in A \qquad Attest_k(E') \in E \qquad Trust_{i,k} \in A \qquad Eq \in E'}{A \vdash K_i(Eq)}$$

$$\mathbf{K5}\ \frac{Verif_i^{Attest}\,(Attest_j(E)) \in A \qquad Trust_{i,j} \in A \qquad Eq \in E}{A \vdash K_i(Eq)}$$

$$\mathbf{KB}\ \frac{A \vdash K_i(Eq)}{A \vdash B_i(Eq)}$$

$$\mathbf{K}\triangleright\ \frac{E \triangleright_i Eq_0 \qquad \text{for all } Eq \in E, A \vdash K_i(Eq)}{A \vdash K_i(Eq_0)}$$

$$\mathbf{B}\triangleright\ \frac{E \triangleright_i Eq_0 \qquad \text{for all } Eq \in E, A \vdash B_i(Eq)}{A \vdash B_i(Eq_0)}$$

+ coherence and structural rules

Semantics properties

Trace-based semantics

- set of compatible traces of events
- events allowed if instantiation of architectural primitives except for computations
- events modify state of knowledge of actors