Log Design for Accountability

Denis Butin, Marcos Chicote and Daniel Le Métayer
Background — The Need for Accountability

Implementing Accountability by Design with PPL

Future Work
Background — The Need for Accountability
Implementing Accountability by Design with PPL
Future Work

Background

Individuals share more & more PII (Personally Identifiable Information)

Stronger privacy guarantees, more transparency needed
Privacy Impact Assessment

- Modern analytic approach to mitigate privacy risks
- Done before deployment
- No guarantees to users about actual running system
Motivation (1/2)

- Runtime / a posteriori verifications needed!
- “Proven trust” instead of “blind trust”
- Data controllers should be accountable to data subjects
- Practical requirements?
Need to provide the means to check that policies were complied with

Approach: check PII handling event logs against policies, automatically

Duality — if PIA done right (*implies* design choices), accountability possible (*depends* on design)
What is Accountability?

- Obligation to accept responsibility for actions
- Attributability: who did what?
- Non-repudiable evidence that cannot be falsified
- Transparent use of information
Accountability does not emerge spontaneously
Feasibility of comprehensive a posteriori verification?
Depends directly on technical architecture!

Example — requirements on logs for accountability

**Timestamps needed in logs if notification to data subject within an hour required when sharing their age with a third party**
Enabling Accountability (2/2)

Need to define:

- **Obligations** to be met ➞ Policy language
- Compliance checking **evidence** ➞ Log architecture
- Compliance checking **procedure** ➞ Log analyzer
Usage Policy Languages

- Almost no one reads lengthy text-formatted privacy policies . . .
- . . . Usage policy languages allow data handling details to be standardized, set and matched!
- On both sides: data subject (preferences), data controller (policies).
- Examples: P3P, EPAL, XACML
Primelife Policy Language (PPL) (1/3)

- Access and data usage policy language, developed by SAP® (European project PrimeLife)
- Extends XACML with usage control features; uses SAML protocol language
- Symmetric architecture (data subject side / data controller side) yields Sticky Policies (agreements)
Primelife Policy Language (PPL) (2/3)

- Automated matching of
  - Data Subject (Data Handling Preferences) &
  - Data Controller (Data Handling Policies)

- Wide range of obligations possible \((\text{trigger} + \text{action})\)

- Authorizations
  - Use for a specific purpose
  - Downstream (third party) usage
Primelife Policy Language (PPL) (3/3)

- Only informal specification available until our work
- Trigger examples: At time / periodic / on PII deletion / on PII access for purpose . . .
- Action examples: Delete PII / encrypt PII / notify DS / log . . . (usually before a set deadline)
PII Event Logging

- Data Controller must provide evidence that agreements met
- Audit possible through inspection of a log against the corresponding sticky policy
- Structure of logs conditions auditability, hence accountability
- Deciding what to include in logs — not a trivial task
Architectural Overview
Contribution: Formalising PPL

- Relevant events precisely defined (syntax) / ambiguities identified
- Compliance properties described (semantics)
- Tool built for automated compliance checking — Haskell implementation
- Policy matching supported
- Reasoning over compliance can be generalised
Guidelines for Log Design

- Importance of explicitness — sufficiently detailed event information needed
- Avoid ambiguity; reflect causal relationships
- Accountability definitions shape log structure & vice versa
- Include contextual information if obligation of performance
Future Work

- Currently: Bridge between abstract and real logs (parsing) — evolving requirements
- Correctness proof for log compliance analyser (formal methods)
- Accountability-oriented, standardised log format (policy language-independent)
- Detailed case studies illustrating design guidelines