Policy Auditing over Incomplete Logs: The reduce algorithm

Anupam Datta
Carnegie Mellon University

Fall 2014
A covered entity may disclose an individual’s protected health information (phi) to law-enforcement officials for the purpose of identifying an individual if the individual made a statement admitting participating in a violent crime that the covered entity believes may have caused serious physical harm to the victim.

**Concepts in privacy policies**
- **Actions:** send(p1, p2, m)
- **Roles:** inrole(p2, law-enforcement)
- **Data attributes:** attr_in(prescription, phi)
- **Temporal constraints:** in-the-past(state(q, m))
- **Purposes:** purp_in(u, id-criminal))
- **Beliefs:** believes-crime-caused-serious-harm(p, q, m)
Detecting Privacy Violations

The Oracle

Species
Title

Computer Program
A program designed to investigate the human psyche.

Detect policy violations

Automated audit for black-and-white policy concepts

Oracles to audit for grey policy concepts

Computer-readable privacy policy

Audit

Detecting Privacy Violations

The Matrix character

Species
Title

Computer Program
A program designed to investigate the human psyche.

Detect policy violations

Automated audit for black-and-white policy concepts

Oracles to audit for grey policy concepts

Computer-readable privacy policy

Audit
Auditing Black-and-White Policy Concepts

With D. Garg (CMU → MPI-SWS) and L. Jia (CMU)

2011 ACM Conference on Computer and Communications Security
Key Challenge for Auditing

Audit Logs are Incomplete

Future: store only past and current events
Example: Timely data breach notification refers to future event

Subjective: no “grey” information
Example: May not record evidence for purposes and beliefs

Spatial: remote logs may be inaccessible
Example: Logs distributed across different departments of a hospital
Abstract Model of Incomplete Logs

Model **all** incomplete logs uniformly as 3-valued structures

\[ \mathcal{L}(P) \in \{tt, ff, uu\} \]

Define **semantics** (meanings of formulas) over 3-valued structures
reduce: The Iterative Algorithm

\[ \text{reduce} \left( \mathcal{L}, \varphi \right) = \varphi' \]
Syntax of Policy Logic

- First-order logic with restricted quantification over infinite domains (challenge for reduce)
- Can express timed temporal properties, “grey” predicates
A covered entity may disclose an individual’s protected health information (phi) to law-enforcement officials for the purpose of identifying an individual if the individual made a statement admitting participating in a violent crime that the covered entity believes may have caused serious physical harm to the victim.

\( \forall p_1, p_2, m, u, q, t. \)
\( (send(p_1, p_2, m) \land tagged(m, q, t, u) \land attr\_in(t, phi)) \)
\( \Rightarrow inrole(p_1, covered\_entity) \land inrole(p_2, law\_enforcement) \land (purp\_in(u, id\_criminal)) \land \exists m'. \diamond state(q,m') \land is\_admission\_of\_crime(m') \land believes\_crime\_caused\_serious\_harm(p_1, q, m') \)
reduce: Formal Definition

General Theorem: If initial policy passes a syntactic **mode check**, then finite substitutions can be computed.

\[
\text{reduce}(L, \forall x. \varphi)
\]

Applications: The entire HIPAA and GLBA Privacy Rules pass this check.
Example

\[
\phi = \\
\forall p_1, p_2, m, u, q, t. \\
\left(\text{send}(p_1, p_2, m) \land\right)
\left(\text{tagged}(m, q, t, u) \land\right)
\left(\text{attr_in}(t, \phi)\right)
\]

\[
\text{inrole}(p_1, \text{covered-entity}) \land \text{inrole}(p_2, \text{law-enforcement}) \\
\land \text{purp_in}(u, \text{id-criminal}) \\
\land \exists m'. \left(\text{state}(q, m')\right) \\
\land \text{is-admission-of-crime}(m') \\
\land \text{believes-crime-caused-serious-harm}(p_1, m'))
\]

Log

Aug 15, 2014
state(Bob, M1)

Sept 17, 2014
send(UPMC, allegeny-police, M2)
tagged(M2, Bob, date-of-treatment, id-bank-robber)

\[
\phi' = T \\
\land \text{purp_in}(\text{id-bank-robber, id-criminal}) \\
\land \text{is-admission-of-crime}(M1) \\
\land \text{believes-crime-caused-serious-harm}(\text{UPMC, M1})
\]
Correctness of Reduce

**Theorem 3.2** (Partial correctness of reduce). If $\text{reduce}(\mathcal{L}, \varphi) = \psi$ and $\mathcal{L} \leq \mathcal{L}'$, then (1) $\mathcal{L}' \models \varphi$ iff $\mathcal{L}' \models \psi$ and (2) $\mathcal{L}' \models \overline{\varphi}$ iff $\mathcal{L}' \models \overline{\psi}$. 
Implementation and Case Study

- Implementation and evaluation over simulated audit logs for compliance with all 84 disclosure-related clauses of HIPAA Privacy Rule

- Performance:
  - Average time for checking compliance of each disclosure of protected health information is 0.12s for a 15MB log

- Mechanical enforcement:
  - reduce can automatically check 80% of all the atomic predicates
Ongoing Transition Efforts

- Integration of reduce algorithm into Illinois Health Information Exchange prototype
  - Joint work with UIUC and Illinois HLN

- Auditing logs for policy compliance
  - Ongoing conversations with Symantec Research
Applications of Reduce

- Audit to detect violations of policy or demonstrate compliance

- Provide explanations for violations (e.g., which clause of HIPAA was violated)

- Help train employees about privacy laws (e.g., check whether a certain type of disclosure is permitted by HIPAA)
Learning Outcomes for You

- Translate privacy laws into first-order logic for use by reduce

- Use reduce tool to check logs for compliance with laws

- Use reduce to check whether certain types of disclosures are permitted by a privacy law

Homework 1 will make you work through these problems
Possible project around other privacy laws such as FERPA, COPPA
Related Work

Privacy Specification Languages

- P3P [Cranor et al.], XACML [OASIS], EPAL [Backes et al.]: Less expressive (no temporal ops,..)
- *Logic of Privacy and Utility* [Barth et al.]: Related specification logic; enforcement only for propositional fragment
Related Work

Logical Specification of Privacy Laws

Smaller fragments of laws

- *Logic of Privacy and Utility* [Barth et al.]: Example clauses from HIPAA and GLBA
- PrivacyAPIs [Gunter et al.]: HIPAA 164.506
- Datalog HIPAA [Lam et al.]: HIPAA 164.502, 164.506, 164.510
Related Work

Runtime monitoring in MFOTL

[Basin et al ’10]

- Pre-emptive enforcement
- Efficient implementation
- Assumes past-completeness of logs
- Less expressive mode checking (“safe-range check”)
- Cannot express HIPAA or GLBA
Related Work

Industry practice

Fairwarning Audit Tool
- Customized SQL queries over access logs
- Queries not tied to policy clauses
Detecting Policy Violations

- **The Oracle**
  - The Matrix character

<table>
<thead>
<tr>
<th>Species</th>
<th>Computer Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>A program designed to investigate the human psyche.</td>
</tr>
</tbody>
</table>

- Computer-readable privacy policy

- **Audit**
  - Automated audit for black-and-white policy concepts
  - Detect policy violations
  - Oracles to audit for grey policy concepts

---

21
Thanks!
More Technical Details
Definition of $\widehat{\text{sat}}$

Assume: The function $\text{sat}(L, P)$ computes all substitutions $\sigma$ for variables in $P$ such that $L \models P\sigma$, if certain argument positions in $P$ are ground.

\[
\begin{align*}
\widehat{\text{sat}}(L, p_O(t_1, \ldots, t_n)) & = \text{sat}(L, p_O(t_1, \ldots, t_n)) \\
\widehat{\text{sat}}(L, \top) & = \{\bullet\} \\
\widehat{\text{sat}}(L, \bot) & = \{\} \\
\widehat{\text{sat}}(L, c_1 \land c_2) & = \bigcup_{\sigma \in \text{sat}(L, c_1)} \sigma + \text{sat}(L, c_2\sigma) \\
\widehat{\text{sat}}(L, c_1 \lor c_2) & = \text{sat}(L, c_1) \cup \text{sat}(L, c_2) \\
\widehat{\text{sat}}(L, \exists x. c) & = \text{sat}(L, c) \setminus \{x\} \quad (x \text{ fresh})
\end{align*}
\]
Mode Analysis: Idea

- Example 1: \( \text{addless}(x, y, a) = x + y < a \)

- Key idea: If input positions are grounded, then only finite number of satisfying substitutions for output positions.

- Example 1 moding: \( \text{addless}(+, -, +) \)

- Example 2: \( \theta = \text{send}(p1, p2, m) \land \text{tagged}(m, q, t, u) \)

- \( \text{send}(-, -, -) \): all positions are output mode
- \( \text{tagged}(+, -, -, -) \): message position is input mode
Mode Analysis: Predicates

1. $\{\} \vdash \text{send}(p1, p2, m): \{p1, p2, m\}$
2. $\{p1, p2, m\} \vdash \text{tagged}(m, q, t, u): \{p1, p2, m, q, t, u\}$

$\forall k \in I(pO) \cdot \text{fv}(t_k) \subseteq \chi_I \quad \chi_O = \chi_I \cup \bigcup_{j \in O(pO)} \text{fv}(t_j)$

$\chi_I \vdash pO(t_1, \ldots, t_n): \chi_O$
Mode Analysis: Conjunction

1. \{\} |\- send(p_1, p_2, m): \{p_1, p_2, m\}

2. \{p_1, p_2, m\} |\- tagged(m, q, t, u): \{p_1, p_2, m, q, t, u\}

3. \{\} |\- send(p_1, p_2, m) \land tagged(m, q, t, u): \{p_1, p_2, m, q, t, u\}

\[\chi_I \vdash e_1 : \chi \quad \chi \vdash e_2 : \chi O\]

\[\chi_I \vdash e_1 \land e_2 : \chi O\]
Mode Analysis and \( \widehat{\text{sat}} \)

Example: \( \theta = \text{send}(p1, p2, m) \land \text{tagged}(m, q, t, u) \)

- \( \text{send}(-,-,-,-) \): all positions are output mode
- \( \text{tagged}(+,\,-,-,-) \): message position is input mode
- \( \widehat{\text{sat}}(\theta) = \text{sat}(\text{send}(p1,p2,m)) + \text{sat}(\text{tagged}(m,q,t,u) \sigma) \)

\[
\{ \begin{align*}
p1 & \rightarrow \text{UPMC}, \\
p2 & \rightarrow \text{allegeny-police}, \\
m & \rightarrow \text{M2}, \\
q & \rightarrow \text{Bob}, \\
u & \rightarrow \text{id-bank-robber}, \\
t & \rightarrow \text{date-of-treatment}
\end{align*} \}
\]

Log

Jan 1, 2011
state(Bob, M1)

Jan 5, 2011
send(UPMC, allegeny-police, M2)
tagged(M2, Bob, date-of-treatment, id-bank-robber)
Mode Analysis: Termination of

General Theorem: If initial policy passes a syntactic **mode check**, then finite substitutions can be computed

Applications: The entire HIPAA and GLBA Privacy Rules pass this check