Computational Logic and Security

EE219 C Class Presentation

Privacy and Contextual Integrity: Frameworks and Applications –
(Barth, Datta, Mitchell, Nissenbaum)

Preserving Secrecy Under Refinement –
(Pavol Černý, Rajeev Alur, Steve Zdancewic)
Paper 1:

Privacy and Contextual Integrity: Frameworks and Applications
Privacy

- **Privacy** is the ability of an individual or group to keep their lives and personal affairs out of public view, or to control the flow of information about themselves. Privacy can be seen as an aspect of security—one in which trade-offs between the interests of one group and another can become particularly clear. - Wikipedia

Privacy is an individual’s right of determining, ordinarily, to what extent his thoughts, sentiments, and emotions shall be communicated to others. - Common Law Right to Privacy (Samuel Warren and Louis Brandeis, 1890)
Contextual Integrity (CI)

- Contextual Integrity, is respected when norms of appropriateness and distribution are respected; it is violated when any of the norms are infringed.

- Norms of Appropriateness: types of information are/are not appropriate for a given context

- Norms of Distribution (Flow) determine the principles governing distribution (flow) of information from one party to another.
  - S shares information with R at S’s discretion
  - R requires S to share information
  - R may freely share information about S
  - R may not share information about S with anyone
  - R may share information about S under specified constraints
Components of information flow in CI

- Sender
- Recipient
- Subject
- Attributes
- Past
- Future
- Combination

Role Based Access Control

XACML
What this paper presents

- A background on contextual integrity
- Formalization in Linear Temporal Logic
- Policy Relations and Operations
- Example cases of privacy laws: HIPAA, GLBA, COPPA
What this paper presents

- A background on contextual integrity
- Formalization in Linear Temporal Logic
- Policy Relations and Operations
- Example cases of privacy laws: HIPAA, GLBA, COPPA
Contextual Integrity (CI)

A transfer of information is:
A(Alice) gives information to B(Bob) about C(Charlie).

There is always an associated context.
A is doctor, B is insurance agency and C is patient.
A is teacher, B is student and C is hiring firm.

Privacy (security of information) expectation depends on what it is, the agents (A,B,C) involved as well as the context.
Contextual Integrity

- Agents abstracted into roles (e.g. doctor, patient)
- Particular information abstracted into types (e.g., height, age, medical condition)
- Norms state what is allowed and what is disallowed
- Transmission principles impose past and future requirements on history of agent interaction
What this paper presents

- A background on contextual integrity
- Formalization in Linear Temporal Logic
- Policy Relations and Operations
- Example cases of privacy laws: HIPAA, GLBA, COPPA
Formalization

Modeling communicating agents

- Set of Agents $P$ (who)
- Set of attributes $T$ (what)
- Knowledge state $K = P \times P \times T$

$(p, q, t) \in K$ is $p$ knows the value of attribute $t$ of $q$. 
Data Model

Modeling attribute inference: If postal address is known, postal code is known.

- Computational rule \((T',t)\) where \(T' \subseteq T\) and \(t \in T\). We say \(t\) is derivable from \(T\)
- Learning relation on knowledge states
- \(\forall k \forall p,q \in P\) if \(\{p\} \times \{q\} \times T \subseteq k\) and \(t\) is derivable from \(T\), then \(k \rightarrow k'\) where \(k' = k \cup \{(p,q,t)\}\). The transitive closure of \(\rightarrow\) defines the new knowledge state from existing state \(k\) after adding element \((p,q,t)\).
Communication model

Modeling messages

- A message $m \subseteq P \times T$, which is closed under computation rules.
- A communication action would be $(p_1, p_2, m)$ where $p_1$ is sender, $p_2$ is receiver and $m$ is the message.
- A communication action transform knowledge states as follows:

$$\forall \kappa, \hat{\kappa}. \forall p_1, p_2 \in P. \forall m \in M.$$  

if $\kappa \xrightarrow{I} \hat{\kappa}$ and $\{p_1\} \times \text{content}(m) \subseteq \hat{\kappa}$,

then $\kappa \xrightarrow{(p_1, p_2, m)} \kappa'$,

where $\kappa' = \hat{\kappa} \cup \{p_2\} \times \text{content}(m)$. The contents of the message are first computed by the sender (at $\hat{\kappa}$) and then learned by the recipient (at $\kappa'$).
CI model

Modeling contextual integrity

- Set of Roles R (partially ordered set - specialization)
- Partition of R i.e. Set of contexts C
- A agent can have multiple roles.
- History of agent is an infinite trace: a sequence of triples \((k,r,a)\) where \(k\) is knowledge state, \(r\) is role state and \(a\) is communication action and \(\kappa_n \xrightarrow{a_{n+1}} \kappa_{n+1}\), for all \(n \in \mathbb{N}\).
Temporal Logic

Syntax of logic

$$\varphi ::= \text{send}(p_1, p_2, m) \mid \text{contains}(m, q, t) \mid \text{inrole}(p, r) \mid \text{incontext}(p, c) \mid t \in t' \mid \varphi \land \varphi \mid \neg \varphi \mid \varphi U \varphi \mid \varphi S \varphi \mid \Box \varphi \mid \exists x : \tau . \varphi$$

- $t \in t'$ means $t$ can be inferred from $t' \subseteq T$.
- Rest are familiar to us – LTL with existential quantifier.
Norms

Formula representing contextual norms

\[
\sigma \models \Box \forall p_1, p_2, q : P. \forall m : M. \forall t : T. \text{incontext}(p_1, c) \land \\
\text{send}(p_1, p_2, m) \land \text{contains}(m, q, t) \rightarrow \\
\bigvee_{\varphi^+ \in \text{norms}^+(c)} \varphi^+ \land \\
\bigwedge_{\varphi^- \in \text{norms}^-(c)} \varphi^-
\]

where \text{norm}^+ and \text{norms}^- are as follows

**positive norm:** \text{inrole}(p_1, \hat{r}_1) \land \text{inrole}(p_2, \hat{r}_2) \land \text{inrole}(q, \hat{r}) \land (t \in \hat{t}) \land \theta \land \psi

**negative norm:** \text{inrole}(p_1, \hat{r}_1) \land \text{inrole}(p_2, \hat{r}_2) \land \text{inrole}(q, \hat{r}) \land (t \in \hat{t}) \land \theta \rightarrow \psi

- \theta is an agent constraint (free of temporal operators)
- \psi represents principle of transmissions and is temporal phenomenon describing past and future actions of agents.
- Attribute closed downward for positive norm and upward for negative norm
What this paper presents

- A background on contextual integrity
- Formalization in Linear Temporal Logic
- Policy Relations and Operations
- Example cases of privacy laws: HIPAA, GLBA, COPPA
Policy Relations and Operations

- Policy Consistency -> LTL satisfiability
- Policy Refinement -> Implication
- Policy Combination -> Conjunction/Disjunction
- Strong compliance -> Satisfiability
- Weak compliance -> LTL runtime verification (efficient)

Benefits:
- Non-ambiguous representation and enforcement
- Automated standard LTL tools
Policy Relations and Operations

- Policy Consistency \(\rightarrow\) LTL satisfiability

  **Def.** A privacy policy \(\theta\) is consistent with a purpose \(\alpha\) if there exists a trace \(\sigma\) such that \(\sigma \models \theta \land \alpha\).

- Policy Refinement/Entailment \(\rightarrow\) implication

  **Def.** A privacy policy \(\theta_1\) entails a policy \(\theta_2\) if the LTL formula \(\theta_1 \rightarrow \theta_2\) is valid over traces.
Compliance modeling

- Weak compliance -> LTL runtime verification

**Def.** Given a finite past history $\sigma$, an action $a$ weakly complies with privacy policy $\theta$ if $\sigma \cdot a$ is a path in the tableau of $\theta$ that starts at an initial $\theta$-atom. The future requirements of $\sigma \cdot a$ is the LTL formula $\psi$ such that, for all traces $\sigma'$,

$$\sigma' \models \psi \text{ if, and only if, } \sigma \cdot a \cdot \sigma' \models \theta.$$ 

- Strong Compliance -> Satisfiability

**Def.** Given a finite past history $\sigma$, an action $a$ strongly complies with a privacy policy $\theta$ if there exists a trace $\sigma'$ such that $\sigma \cdot a \cdot \sigma' \models \theta$. 
What this paper presents

- A background on contextual integrity
- Formalization in Linear Temporal Logic
- Policy Relations and Operations
- Example cases of privacy laws: HIPAA, GLBA, COPPA
Example (HIPAA Privacy Rule)

- Addressed information flow: Transfer of protected health information (phi) about patients from covered entities (e.g. hospitals) to health care providers.

- Sender role: Covered entity (e.g. hospitals)

- Recipient role: Health care provider

- Subject role: Patient

- Information type: Protected health information
Example (HIPAA Privacy Rule)

Legislative statement expresses
permissible actions - positive norms
forbidden actions - negative norms

Let us look at 5 examples of norms in
HIPAA and how it is modeled.
(4 positive and 1 negative norm)
Norm 1 Positive

Any person may be given information about himself/herself until it conflicts with some forbidding norm.
Norm 1 Positive

Any person may be given information about himself/herself until it conflicts with some forbidding norm.

\[ \text{inrole}(p_1, \text{covered-entity}) \land \text{inrole}(p_2, \text{individual}) \land (q = p_2) \land (t \in \phi) \]
Norm 2 Positive

Healthcare provider is entitled to information about its patient from hospital.
Norm 2 Positive

Healthcare provider is entitled to information about its patient from hospital.

\text{inrole}(p_1, \text{covered-entity}) \land \text{inrole}(p_2, \text{provider}) \land \text{inrole}(q, \text{patient}) \land (t \in \phi)
A psychotherapy-note cannot be shown to the subject until the psychiatrist approves.

\[
\text{inrole}(p_1, \text{covered-entity}) \land \text{inrole}(p_2, \text{individual}) \land (q = p_2) \land (t \in \text{psychotherapy-notes}) \rightarrow \\
\bigdiamond \exists p : \text{inrole}(p, \text{psychiatrist}) \land \text{send}(p, p_1, \text{approve-disclose-psychotherapy-notes})
\]

- Psychiatrist sends approval to disclose notes
- Psychotherapy note shown to subject

A \rightarrow S only if A has happened in past

Norm 1 permits, norm 3 prohibits; no contradiction as norm 1 only permits doesn’t mandate.
Norm 4 Positive

Covered entity (hospital, health-centre) can release information about location and condition of any individual to anyone enquiring about him with name.

\[ \text{inrole}(p_1, \text{covered-entity}) \land \text{inrole}(p_2, \text{individual}) \land \text{inrole}(q, \text{individual}) \land (t \in \text{condition-and-location}) \land \]

\[ \Diamond \exists m': M. \text{send}(p_2, p_1, m') \land \text{contains}(m', q, \text{name}) \]

Some body (p2) sent hospital (p1) message with name of q

The condition and location of q is given to p2 by the hospital (p1)
Clergy can obtain directory information that contains (directly or transitively) individual’s name, general condition, location.

\[ \text{inrole}(p_1, \text{covered-entity}) \land \text{inrole}(p_2, \text{clergy}) \land \text{inrole}(q, \text{individual}) \land (t \in \text{directory-information}) \]
COPPA and GLBA

An exercise in specifying informal specifications in LTL.

Let’s “run over” a couple of examples from COPPA.
When a child sends information to a website the parents must have received a notice, granted permission and since not revoked permission.

\[
\text{inrole}(p_1, \text{child}) \land \text{inrole}(p_2, \text{web-site}) \land (q = p_1) \land (t \in \text{protected-info}) \rightarrow \\
\exists p : P. \text{inrole}(p, \text{parent}) \land \neg \text{send}(p, p_2, \text{revoke-consent}) S \\
(\text{send}(p, p_2, \text{grant-consent}) \land \lozenge \text{send}(p_2, p, \text{privacy-notice}))
\]
COPPA

- Website must delete information after a parent revokes permission?
  
  “Present infrastructure does not support removal of information.”

Can we model revoke as reassignment of existing attributes to “unassigned” – is there a better way to model “forgetting actions” (without actually having a stack!)
What this work does not cover?

- Anonymous information (like in HIPAA)
  Name-SID-Year in grad school-number of library visits is private BUT
  Year in grad school-number of library visits is NOT

- ‘Averaged’ information (group attribute)
  Name-Age-Telephone Bill is private BUT
  Average data Age-Telephone Bill is NOT

- Data value based policy not just type-based
  Load distribution of a network – peak hours when it is vulnerable to DOS attack could be kept confidential
Paper 2:

Preserving Secrecy Under Refinement
Motivation

- Privacy = Secrecy
- Implementation = Refinement
- Secrecy preserving refinement needed to implement privacy preserving laws

Given the HIPAA, after we have written the laws using the previous paper’s technique as LTL, how do we ensure that an Hospital Information System is consistent with the privacy laws?

Instead of model checking the entire system, can we build an abstraction which would be safe with respect to the privacy (secrecy) rules?

TO DO THIS WE NEED “SECRECY PRESERVING REFINEMENT”
Summarization

Property under consideration – P
T’ – abstract trace Ti – concrete trace

T1(P) T2(P) T3(¬P) → T’ (P is secret)
T1(P) T2(P) T3(P) → T’ (P inferred)
T1(¬P) T2(¬P) T3(¬P) → T’ (¬P inferred)
Outline

- Defining a framework for secrecy
- Comparison with existing notions of secrecy
- Non-expressibility in mu-calculus
- Secrecy preserving refinement
- Simulation based proof method
- Applications
Outline

- Defining a framework for secrecy
- Comparison with existing notions of secrecy
- Non-expressibility in mu-calculus
- Secrecy preserving refinement
- Simulation based proof method
- Applications
Defining Secrecy

3 parameters:

- Property (predicate over system variables) to be kept secret $\alpha$
  (like $\text{first\_letter(password) = s}$)

- Distinguishing power of the observer (Observation equivalence
  of runs) $\equiv$
  (like “$r1 \equiv r2$ if the respective last states are equivalent
  $\text{obs(last(r1)) = obs(last(r2))}$”)

- Executions of interest $\beta$
  (like “all runs terminating without error”)

Defining Secrecy

“α is secret in β with respect to ≡”

\[
IR(r, \alpha, \equiv) = \begin{cases} 
T \text{ iff } \forall r' \equiv r \Rightarrow r' \in \alpha \\
F \text{ iff } \forall r' \equiv r \Rightarrow r' \notin \alpha \\
M \text{ otherwise}
\end{cases}
\]

α is secret in β with respect to ≡ iff for all r in β,
\[IP(r, \alpha, \equiv) = M.\]
Example

Ξ : Capital letters denote secret information
- not available to observer

<table>
<thead>
<tr>
<th>$\alpha, \beta$</th>
<th>2 A or G A</th>
<th>Only 1 A</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G \ x = 5$</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>$G \ y = x$</td>
<td>True</td>
<td>$m$</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
a, x &= 5, \quad y = 5 \\
a, x &= 4, \quad y = 4 \\
\sim a, x &= 5, \quad y = 5 \\
\sim a, x &= 5, \quad y = 3 \\
a, x &= 7, \quad y = 7 \\
a, x &= 5, \quad y = 7
\end{align*}
\]
Outline

- Defining a framework for secrecy
- Comparison with existing notions of secrecy
- Non-expressibility in mu-calculus
- Secrecy preserving refinement
- Simulation based proof method
- Applications
Linear-time Secrecy

Special case with

- $r \approx r'$ iff the sequences of labels are the same; strong, timing-sensitive equivalence

- $r \approx wr'$ iff the sequence of labels are the same, modulo ε label

Example:

A: $x=?; y=0; z=x; \text{send } z$

B: $x=?; y=0; z=y; \text{send } z$

By looking at sent bit, both yield ttt0. A reveals $x$ was set to 0, B does not.
Noninterference

Special case with

- $r \approx r'$ iff their initial states share the values of low variables and the same holds for their final states
- $\beta$ set of all terminating runs
- noninterference w.r.t. $P$ iff for all $\alpha$ in $P$, $\alpha$ is secret in $\beta$ w.r.t. $\approx$

The above ensures that if two input states share the same values of low variables, then the behaviors of the program executed from these states are indistinguishable by the observer.”
Perfect Security Property

Special case with
- $r \approx r'$ iff their sub-sequences of low-security labels are equal.
- $P$ contains a property $\alpha_h$ for each high-security action $h$.
- $\alpha_h$ holds for a run $r$ if $h$ occurs in $r$
- $\beta$ is the set of all runs
- PSP holds iff for all $\alpha_h$ in $P$, $\alpha_h$ is secret in $\beta$ w.r.t. $\approx$

The above ensures that though the observer knows the specification (the set of all possible traces) and observes the low events, but he or she cannot deduce whether a high-security event occurred or not.
Outline

- Defining a framework for secrecy
- Comparison with existing notions of secrecy
- Non-expressibility in mu-calculus
- Secrecy preserving refinement
- Simulation based proof method
- Applications
Secrecy is not a property of a single run.
Mu-Calculus

- **Thm:** Secrecy is **not** definable in \( \mu \)-calculus.

**Proof:** It is not a regular tree language.

(We can work it on the board if required subject to limitations of time)
Outline

- Defining a framework for secrecy
- Comparison with existing notions of secrecy
- Non-expressibility in mu-calculus
- Secrecy preserving refinement
- Simulation based proof method
- Applications
Secrecy-preserving Refinement

• “If $\alpha$ is secret in $S$, then $\alpha$ is secret in $I$”, where implementation $I$ refines specification $S$?

• Notation: Let $P$ be the set of all the secret properties, then we need to define “$I$ $P$-refines $S$” which is consistent with above notion of secrecy preservation.
Standard refinement

Definition (standard refinement):

All behaviors of Imp are allowed by Spec

\( \text{Runs}(\text{Imp}) \subseteq \text{Runs}(\text{Spec}) \).

Not a sufficient condition.
Standard refinement

- Definition (standard refinement):
  All behaviors of Imp are allowed by Spec.
  \((\text{Runs}(\text{Imp}) \subseteq \text{Runs}(\text{Spec}))\)

It is a necessary condition.
Definition

Intuition: Refinement which preserves secrecy needs extending equivalence relation to the runs of the two systems.

\[ \text{Equiv} \equiv \text{is now a subset of} \]

\[ (\text{Runs in Spec U Runs in Imp}) \times (\text{Runs in Spec U Runs in Imp}) \]

Secrecy-preserving refinement
Let \( T_{\text{spec}}, T_{\text{imp}} \) be two labeled transition system, let \( \mathcal{P} \) be a set of properties and let \( \equiv \) be an equivalence relation on \( R(T_{\text{spec}}) \cup R(T_{\text{imp}}) \). \( T_{\text{imp}} \) \( \mathcal{P} \)-refines \( T_{\text{spec}} \) w.r.t. \( \equiv \) iff for all runs \( r \in R(T_{\text{imp}}) \), there exists a run \( r' \in R(T_{\text{spec}}) \) such that \( r \equiv r' \) and for all properties \( \alpha \in \mathcal{P} \), \( \text{IP}(r, \alpha, \equiv) \subseteq \text{IP}(r', \alpha, \equiv) \).
Simulation.

Thm: If \( \text{Runs}(\text{Imp}) \subseteq \text{Runs}(\text{Spec}) \) and \( \text{Imp} \) simulates \( \text{Spec} \), then \( \text{Imp} \) P-refines \( \text{Spec} \).

Suppose \( \text{Spec} \) does not leak the secret.

Contradiction with the simulation condition.

Leaks a secret
Simulation.

\[ \text{Runs}(\text{imp}) \subseteq \text{Runs}(\text{Spec}) \text{ and } \text{Imp} \text{ simulates } \text{Spec} \]

Not a necessary condition.
Outline

- Defining a framework for secrecy
- Comparison with existing notions of secrecy
- Non-expressibility in mu-calculus
- Secrecy preserving refinement
- Simulation based proof method
- Applications
Applications

- Verifying cryptographic algorithms.
- Validating refinement and implementation of protocols.
- Validating refinement and implementation of formal policy formulations as we saw in the first part.
- Malware detection: if $M \mid m$ P-refines $M$, then $m$ is not a spyware with respect to properties in $P$ about $M$. Is it so?
Thanks!

Any Questions?

Where else to look at –