Bootstrapping Privacy Compliance in Big Data Systems

Anupam Datta

Fall 2015
Privacy Compliance for Bing

Setting:

- Auditor has access to source code
The Privacy Compliance Challenge

Legal Team
Crafts Policy

Privacy Champion
Interprets Policy

Developer
Writes Code

Audit Team
Verifies Compliance

English Privacy Policy

Compliant?

Millions of Lines of Undocumented Code
A Streamlined Audit Workflow

Legal Team
- Crafts Policy
- Interprets Policy

Developer
- Code analysis

Audit Team
- Verifies Compliance

Grok
- Data inventory with policy labels

Checker
- Potential violations

Legalease
- A formal policy specification language

Encode
- Refine

Fix code
- Update Grok

Annotated Code
- Legalease Policy
A Streamlined Audit Workflow

**Workflow** for privacy compliance

**Legalease**, usable yet formal policy specification language

**Grok**, bootstrapped data inventory for big data systems

**Scalable** implementation for Bing
Privacy as Restrictions on Personal Information Flow

Direct

Restrictions

Purpose & Role based

Temporal

EPAL

XACML

*-access control

FOTLs

[Formal Contextual Integrity, Reduce audit algorithm, Basin et al.]

Grok + Legalease

Jif, FlowCaml, ...

[Hayati & Abadi]

Information Flow Experiments

Differential Privacy

Probabilistic Interference

Differential Privacy

Interference

Information Flow

Purpose → Planning

Differential Privacy
A Streamlined Audit Workflow

Legal Team
Crafts Policy

Privacy Champion
Interprets Policy

Legalease
A formal policy specification language

Grok
Data inventory with policy datatypes

Code analysis, developer annotations

Developer
Writes Code

Audit Team
Verifies Compliance

Fix code

Annotated Code

Update Grok

Legalease Policy

Potential violations

Encode

Refine
## Specification: Legalease

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Usable by lawyers and privacy champs.</td>
<td>Expressive enough for real-world policies.</td>
<td>Precise semantics for local reasoning.</td>
</tr>
</tbody>
</table>
Legalease : Syntax

Policy Clause $C$ ::= $D$ | $A$
Deny Clause $D$ ::= DENY $T_1 \cdots T_n$ EXCEPT $A_1 \cdots A_m$ | DENY $T_1 \cdots T_n$
Allow Clause $A$ ::= ALLOW $T_1 \cdots T_n$ EXCEPT $D_1 \cdots D_m$ | ALLOW $T_1 \cdots T_n$
Attribute $T$ ::= ⟨attribute-name⟩ $v_1 \cdots v_l$
Value $v$ ::= ⟨attribute-value⟩
DENY Datatype IPAddress
UseForPurpose Advertising

We will not use full IP Address for Advertising.
We will not use full IP Address for Advertising. IP Address may be used for detecting abuse. In such cases, it will not be combined with account information.
Designed for Usability

Exceptions
How legal texts are structured
One-to one correspondence

Local Reasoning
Each exception refines its immediate parent
Formally proven property

H. DeYoung, D. Garg, L. Jia, D. Kaynar, and A. Datta, “Experiences in the logical specification of the HIPAA and GLBA privacy laws”
We will **not** use full IP Address for **Advertising**. IP Address may be used for detecting **abuse**. In such cases, it will not be combined with account information.
A Lattice of Policy Labels

- If “IPAddress” use is allowed then so is everything below it
- If “IPAddress:Truncated” use is denied then so is everything above it
Designed for Precision

\[ T^G \not\subseteq T^C \]
\[ \text{ALLOW } T^C \text{ EXCEPT } D_1 \cdots D_m \text{ denies } T^G \quad (A_1) \]

\[ T^G \subseteq T^C \quad \exists_i D_i \text{ denies } T^G \]
\[ \text{ALLOW } T^C \text{ EXCEPT } D_1 \cdots D_m \text{ denies } T^G \quad (A_2) \]

\[ T^G \subseteq T^C \quad \forall_i D_i \text{ allows } T^G \]
\[ \text{ALLOW } T^C \text{ EXCEPT } D_1 \cdots D_m \text{ allows } T^G \quad (A_3) \]

\[ \bot \in T^G \cap T^C \]
\[ \text{DENY } T^C \text{ EXCEPT } A_1 \cdots A_m \text{ allows } T^G \quad (D_1) \]

\[ \bot \not\in T^G \cap T^C \quad \exists_i A_i \text{ allows } T^G \cap T^C \]
\[ \text{DENY } T^C \text{ EXCEPT } A_1 \cdots A_m \text{ allows } T^G \quad (D_2) \]

\[ \bot \not\in T^G \cap T^C \quad \forall_i A_i \text{ denies } T^G \cap T^C \]
\[ \text{DENY } T^C \text{ EXCEPT } A_1 \cdots A_m \text{ denies } T^G \quad (D_3) \]

**TABLE I**
GRAMMAR FOR LEGALEASE

**TABLE III**
INFERENGE RULES FOR LEGALEASE
ALLOW
EXCEPT
DENY DataType IPAddress:Expired
DENY DataType UniqueIdentifier:Expired
DENY DataType SearchQuery, PII InStore Store
DENY DataType UniqueIdentifier, PII InStore Store

DENY DataType BBEPData UseForPurpose Advertising

DENY DataType BBEPData, PII InStore Store

DENY DataType BBEPData:Expired

DENY DataType UserProfile, PII InStore Store

DENY DataType PII UseForPurpose Advertising
DENY DataType PII InStore AdStore

DENY DataType SearchQuery UseForPurpose Sharing
EXCEPT
ALLOW DataType SearchQuery:Scrubbed

"we remove the entirety of the IP address after 6 months"
"[we remove] cookies and other cross session identifiers, after 18 months"
"We store search terms (and the cookie IDs associated with search terms) separately from any account information that directly identifies the user, such as name, e-mail address, or phone numbers."
"we do not use any of the information collected through the Bing Bar Experience Improvement Program to identify, contact or target advertising to you"
"we take steps to store [information collected through the Bing Bar Experience Improvement Program] separately from any account information we may have that directly identifies you, such as name, e-mail address, or phone numbers"
"we delete the information collected through the Bing Bar Experience Program at eighteen months."
"we store page views, clicks and search terms used for ad targeting separately from contact information you may have provided or other data that directly identifies you (such as your name, e-mail address, etc.)."
"our advertising systems do not contain or use any information that can personally and directly identify you (such as your name, email address and phone number)."
"Before we [share some search query data], we remove all unique identifiers such as IP addresses and cookie IDs from the data."
Designed for Expressivity (Google, October 2013)

ALLOW
EXCEPT
    DENY DataType PII UseForPurpose Sharing

EXCEPT
    ALLOW DataType PII:OptIn
EXCEPT
    ALLOW AccessByRole Affiliates
EXCEPT
    ALLOW UseForPurpose Legal

DENY DataType DoubleClickData, PII
EXCEPT
    ALLOW DataType DoubleClickData, PII:OptIn

- “We do not share personal information with companies, organizations and individuals outside of Google unless one of the following circumstances apply:”
- “We require opt-in consent for the sharing of any sensitive personal information.”
- “We provide personal information to our affiliates or other trusted businesses or persons to process it for us”
- “We will share personal information [if necessary to] meet any applicable law, regulation, legal process or enforceable governmental request.”
- “We will not combine DoubleClick cookie information with personally identifiable information unless we have your opt-in consent”
Legalease Usability

Survey taken by 12 policy authors within Microsoft
Encode Bing data usage policy after a brief tutorial

Time spent
2.4 mins on the tutorial
14.3 mins on encoding policy

High overall correctness
A Streamlined Audit Workflow

Legal Team
Crafts Policy

Encode

Privacy Champion
Interprets Policy

Refine

Legalease
A formal policy specification language

Grok
Data inventory with policy labels

Code analysis
Developer annotations

Annotated Code

Checker

Update Grok

Potential violations

Audit Team
Verifies Compliance

Fix code

19
A Streamlined Audit Workflow

Legal Team
Crafts Policy

Privacy Champion
Interprets Policy

Legalease
A formal policy specification language

Grok
Data inventory with policy labels

Developer
Writes Code

Audit Team
Verifies Compliance

Encode
Refine

Annotated Code
Update Grok

Potential violations
Legalease Policy

Fix code
Map-Reduce Programming Systems

Scope, Hive, Dremel
Data in the form of Tables

Code Transforms Columns to Columns
No Shared State
Limited Hidden Flows

users =
  SELECT _name, _age
  FROM datasetAB
user_tag =
  SELECT GenerateTag(_name, _age)
  FROM users
OUTPUT user_tag TO datasetC
Grok

Purpose Labels
Annotate programs with purpose labels
**Grok**

### Purpose Labels
Annotate programs with purpose labels

### Initial Data Labels
Heuristics and Annotations
D. E. Denning. “A lattice model of secure information flow”
A Lattice of Policy Labels

- If “Profile” use is allowed then so is everything below it
- If “Name” use is denied then so is everything above it
Implicit flows

Beyond direct flows discussed in healthcare audit examples
Map

Operate on rows
in parallel
eg. filtering

Reduce

Combine groups of rows
eg. aggregation

users =
SELECT Name, Age
FROM datasetAB

users_35 =
SELECT _name, _age
FROM users
WHERE (_age > 35)

ages_35 =
SELECT _age, COUNT(_name) AS Profile
FROM users_35
GROUP BY _age

OUTPUT ages_35 TO datasetC
Combine Noisy Sources

- Carefully curated regular expressions
- Leverages developer conventions
- Significant Noise

- Expensive
- Low Noise

- Very Expensive
- Definitive
- Need very few of these

- Variable Name Analysis
- Developer Annotations
- Auditor Verification
Why Bootstrapping Grok Works

Pick the nodes which will label the most of the graph

~200 annotations label 60% of nodes

A small number of annotations is enough to get off the ground.
Scale

- 77,000 jobs run each day
  - By 7000 entities
  - 300 functional groups
- 1.1 million unique lines of code
  - 21% changes on avg, daily
- 46 million table schemas
- 32 million files
- Manual audit infeasible
- Information flow analysis takes ~30 mins daily
Nightly Compliance Process

Static code analysis

Generate report

Manual Audit

files 25M+
schemas 2M+
privacy elements* 300K+
audit candidates 10K+
teams 8
A Streamlined Audit Workflow

Legal Team
Crafts Policy

Encode

Privacy Champion
Interprets Policy

Refine

Legalease
A formal policy specification language

Grok
Data inventory with policy labels

Code analysis
Developer annotations

Annotated Code

Checker

Update Grok

Potential violations

Fix code

Audit Team
Verifies Compliance

Fix code

33
A Streamlined Audit Workflow

Workflow for privacy compliance

Legalease, usable yet formal policy specification language

Grok, bootstrapped data inventory for big data systems

Scalable implementation for Bing

Legal Team
Crafts Policy

Encode

Grok
Data inventory with policy datatypes

Code analysis, developer annotations

Developer
Writes Code

Privacy Champion
Interprets Policy

Fix code

Audit Team
Verifies Compliance

Potential violations

Update Grok

Annotated

Code

Legalease
A formal policy specification language

Workflow for privacy compliance
Reference

Policy Labels: Datatypes

Going down within a lattice:

- TypeState specifies limited temporal properties
- :Encrypted
  - Account Info:E
  - Unique ID:E
  - Location:E...
  - Name:E...
  - Account ID:E...
  - IP Address:E...

- :Truncated
  - Account Info:T
  - Unique ID:T
  - Location:T...
  - Name:T...
  - Account ID:T...
  - IP Address:T...
Policy Types: Concept Lattices

InStore Lattice  UseForPurpose Lattice  AccessByRole Lattice
Formal Semantics

\[
\begin{align*}
T^G \sqsubseteq T^C & \quad \exists_i D_i \text{ denies } T^G \\
\text{ALLOW } T^C \text{ EXCEPT } D_1 \cdots D_m & \text{ denies } T^G
\end{align*}
\]  \ (A_2)

Based on Lattice Orderings on Policy Types
Formal Semantics

Recursively check exceptions
ALLOW clauses have DENY clauses as exceptions
Top Level clause determines Blacklist/Whitelist