“Complete, Safe Information Flow with Decentralized Labels”

Andrews C. Myers and Barbara Liskov (MIT)

Presenters: Wilfred Glover-Akpey and Anuj Sawani
Background

“A Decentralized Model for Information Flow Control”, Andrew C. Myers and Barbara Liskov

- a new model for controlling information flow in systems with mutual distrust and decentralized authority.
- model allows users to share information with distrusted code
- The model improves on existing multilevel security models
Background (continued)

- Shows how static program analysis can be used to certify proper information flows in this model.
- Principals can attach flow policies to pieces of data.
- Principals can declassify labels by modifying their own policies.
- It is compatible with static checking of information flow.
CONTRIBUTION

- They extend the model to allow safe re-labeling that the previous work does not support.
- This work defines a model that enables them to know which re-labeling is legal.
- They define a rule for static checking and are also able to prove that it is both sound and complete for all safe re-labeling that it allows.
- Label checking and label inference are shown to be easy and efficient when using this newly defined rule.
Decentralized Label Model

What is it?
- Labels allow individual owners of information to express their own policies.
- Owners are principals.
- Some principals are authorized to act for other principals.
- Every value used or computed in a program execution has an associated label.

Label
- A set of components, each of which expresses a policy for a single owner.
- A set of flows; a flow is an (owner, reader) pair.

Policy – a set of readers permitted by the owner to read the data.

Principal hierarchy – the acts-for relations between principals in the system.
The Patient/Doctor Example

Principals: For Patients we have Patient_A
For Doctors we have doctor_B

HMO:
- maintains the patient’s medical history
- tracks information flow of the patient
The Patient/Doctor Example (continued)
Related Work

- “Lattice Model of Information flow”, Bell, LaPadula and Denning. – early work
- “(ORAC) model” by C. McCollum et al, – dynamically checked.
- “The Chinese wall policy”, Brewer & Nash, etc – Complex aggregate policies, & others.
- “Certification of Programs for Information flow”, by Dorothy and Peter Denning,– static analysis of security guarantees and others.
Questions ?
Melissa Virus

- Microsoft Word-based macro virus
- Circumvent poor program policy
  - Uses VB script extensions
- Spreads via Emails – 1.2 billion affected

Solution: Language-based security to prevent malicious information flow
Formal model for labels

- **L** = \{ \text{component} : \text{Readers} \}

- Label – set of components
- Each component – a policy for a single owner
Notations and Symbols

- K ∈ L – Component K in L
- R_K – set of readers
- Principal Hierarchy
  - Consists of the acts-for relations between principals in the system

\[ P \vdash x \preceq y \] - in the principal hierarchy P, x can act for y
\[ P' \supseteq P \] - P' extends P
The Label - Set of Flows

- Information flow denoted by \((o_k, r)\)

- Flow set constraints
  - Reader Constraint
    - If flow contains \((o, r)\) and \(r'\) acts-for \(r\), then set contains \((o, r')\)
  - Owner Constraint
    - If flow contains \((o, r)\) and \(o'\) acts-for \(o\), then set contains \((o', r)\)

- Label’s interpretation – Function \(X\)
  - Maps a label to a set of flows

\[
XL = \{(o, r) \mid \forall I \in L : o_I \supseteq o \rightarrow r \in RR_I\}
\]
Sound and Complete Relabeling

- A reader may be dropped from some owner’s reader set.
- A new owner may be added to the label, with an arbitrary reader set.
- A reader may be added as long as it can act for some member of the reader set.
- An owner may be replaced by an owner that acts for it.

No New Flows Added
Static Correctness condition

- Relabeling from $L_1$ to $L_2$ in principal hierarchy $P$
  - Valid only if no new flows are added

\[
X(L_1, P') \supseteq X(L_2, P') \quad \text{where} \quad P' \supseteq P
\]

- Example

DATA
$L_1 = \{\text{patient}_A : \text{doctors}\}$
Less restrictive

Relabeling

DATA
$L_2 = \{\text{HMO_records} : \text{doctor}_B\}$
More restrictive
Static vs Dynamic Checking

- Consider code
  - \( b \) more restrictive than \( x \)

```java
x=0;
if (b==1)
  x=1;
```

- Run-time Check
  - Case 1 : \( b = \) false
    - Program continues
  - Case 2 : \( b = \) true
    - Program aborts (type check fails)
      - Implicit flow

- Compile-time Check
  - Failure will not give any info about \( x \) or \( b \).
Static Checking
Definitions of Join and Meet

- Join
  - Result of a computation (such as adding two numbers)
  - Concatenation of all components
  - Least Restrictive Upper Bound

- Meet
  - Useful for automatic label inference
  - Join of all pair-wise components
  - Most Restrictive Lower Bound
Conclusion

- Describes a complete relabeling rule for the decentralized label model
- Make the model more practical and usable
- Provides flexibility
  - Safe declassification