A SOUND TYPE SYSTEM FOR SECURE FLOW ANALYSIS

Paper By: Denise Volpano, Geoffrey Smith and Cynthia Irvine

Presented By: Endri VANGJEL, Marvin TCHOULA NJIA, Mohamed AL ALI, Mohammed AL MANSOORI
Motivation

• Ensure Secure Information Flow

• Build Basis of provably-secure programming languages
Outline

- Introduction
  - Secure Flow
  - Noninterference
- Lattice Model
- Type System
  - Definition and Goal
  - Core Language
  - Inference Rules
- Type Soundness and proof
- Conclusions
Introduction

- Secure Flow
- Non-interference
Lattice Model

- Bell and Lapadula Extension
- Lattice is a couple \((SC, \leq)\)
  - Security Classes
    - Secrecy
    - Integrity

- Certification Conditions
  - Explicit Flows
  - Implicit Flows
Lattice Model

Type System
  Definition
  Goal
  Core Language
  Secure Flow Types
  Secure Flow Typing Rules
  Inference Rules

Type Soundness
Type System

- Definition
  - Formal system
  - Contains Type Inference Rules
  - Focused on Secure Info. Flow

- Goal
  - Check Program’s Correctness
  - Separate Security Policies from Algorithms
Core Language

- Block-structured language

\[
\begin{align*}
(p \text{hrases}) & \quad p ::= e \mid c \\
(expressions) & \quad e ::= x \mid l \mid n \mid e + e' \mid e - e' \mid e = e' \mid e < e' \\
(commands) & \quad c ::= e ::= e' \mid c; c' \mid \text{if } e \text{ then } c \text{ else } c' \mid \\
& \quad \text{while } e \text{ do } c \mid \text{letvar } x ::= e \text{ in } c
\end{align*}
\]

- Core Language Types

\[
\begin{align*}
(data \text{ types}) & \quad \tau ::= s \\
(phrase \text{ types}) & \quad \rho ::= \tau \mid \tau \text{ var} \mid \tau \text{ cmd}
\end{align*}
\]
Type System

- Secure Flow Typing Rules
- Example

\[
\begin{align*}
\gamma &\vdash e : \tau \text{ var}, \\
\gamma &\vdash e' : \tau \\
\hline
\gamma &\vdash e := e' : \tau \text{ cmd}
\end{align*}
\]
Inference Rules

- Rules define Type System

- Typing Judgment
  - Example:

\[ \lambda; \gamma \vdash p : \rho \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda ): location typing</td>
<td>( \gamma ): identifier</td>
<td></td>
</tr>
<tr>
<td>typing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p ): phrase</td>
<td>( \rho ): phrase type</td>
<td></td>
</tr>
</tbody>
</table>
Inference Rules

\[\begin{align*}
\text{(INT)} & \quad \lambda; \gamma \vdash n : \tau \\
\text{(VAR)} & \quad \lambda; \gamma \vdash x : \tau \text{ var} \quad \text{if } \gamma(x) = \tau \text{ var} \\
\text{(VARLOC)} & \quad \lambda; \gamma \vdash l : \tau \text{ var} \quad \text{if } \lambda(l) = \tau \\
\text{(ARITH)} & \quad \begin{array}{l}
\lambda; \gamma \vdash e : \tau, \\
\lambda; \gamma \vdash e' : \tau
\end{array} \\
\quad \quad \lambda; \gamma \vdash e + e' : \tau \\
\text{(R-VAL)} & \quad \begin{array}{l}
\lambda; \gamma \vdash e : \tau \text{ var}
\end{array} \\
\quad \quad \lambda; \gamma \vdash e : \tau
\end{align*}\]

Figure 2: Inference Rules
Type Soundness

• Important Lemmas

  • Simple Security
    • Applies to expressions

  • Confinement
    • Applies to commands
Type Soundness and proof

- Soundness means that variable in a well typed program do not interfere with variables at low security levels.

- \( \lambda \vdash c : \rho \)
  - \( \mu \vdash c \Rightarrow \mu' \)
  - \( \nu \vdash c \Rightarrow \nu' \)
  - \( \text{dom}(\mu) = \text{dom}(\nu) = \text{dom}(\lambda) \)
  - \( \nu(l) = \mu(l) \quad \forall l \text{ such that } \lambda(l) \leq \tau \)

Then \( \nu'(l) = \mu'(l) \quad \forall l \text{ such that } \lambda(l) \leq \tau \)
Idea of the proof

The soundness of the system is established with respect to natural semantic which consist of a set of evaluation rules.

**BASE** \[ \mu \vdash n \Rightarrow n \]

**CONTENTS** \[ \mu \vdash l \Rightarrow \mu(l) \quad \text{if } l \in \text{dom}(\mu) \]

**ADD** \[ \mu \vdash e \Rightarrow n, \quad \mu \vdash e' \Rightarrow n' \]
\[ \mu \vdash e + e' \Rightarrow n + n' \]

**UPDATE** \[ \mu \vdash e \Rightarrow n, \quad l \in \text{dom}(\mu) \]
\[ \mu \vdash l := e \Rightarrow \mu[l := n] \]

**SEQUENCE** \[ \mu \vdash e \Rightarrow \mu', \quad \mu' \vdash e' \Rightarrow \mu'' \]
\[ \mu \vdash e; e' \Rightarrow \mu'' \]
Conclusions

- The paper
  - Formulates Denning’s Analysis
  - Proves System Type Soundness
  - Builds Basis of provably-secure programming languages
Questions

Do you have any Questions?