CS 5430: Information Flow
Part II: Dynamic Enforcement

Fred B. Schneider
Samuel B Eckert Professor of Computer Science

Department of Computer Science
Cornell University
Ithaca, New York  14853
U.S.A.
Enforcement of FBAC

FLI imposes restrictions on each statement.
\[ v \rightarrow w \implies \Gamma(v) \subseteq \Gamma(w) \]

- **Static Enforcement**
  - Compiler ensures type-correct programs satisfy restrictions.

- **Dynamic Enforcement**
  - Run-time checks ensure program execution satisfies restrictions.
  - Changes to labels mean program execution satisfies restrictions.
Why Dynamic Enforcement?

- Static enforcement: Rejects program if any execution could violate Flow-Label invariant.
- Dynamic enforcement: Blocks after partial execution when Flow-Label invariant could be violated.

\[
\text{if } 0 = 0 \text{ then } x_L := 2 \text{ else } x_L := x_H \text{ fi}
\]
Why Dynamic Enforcement?

- Static enforcement: Rejects program if any execution could violate Flow-Label invariant.
- Dynamic enforcement: Blocks after partial execution when Flow-Label invariant could be violated.

Type error!

\[
\text{if } 0 = 0 \text{ then } x_L := 2 \text{ else } x_L := x_H \text{ fi}
\]
Why Dynamic Enforcement?

- **Static enforcement:** Rejects program if execution could violates Flow-Label invariant
- **Dynamic enforcement:** Blocks after partial execution when Flow-Label invariant could be violated.

```plaintext
if 0 = 0 then x_L := 2 else x_L := x_h fi
```

```plaintext
ctx = L
check:
ctx ⊔ Γ(2) ⊆ Γ(x_L)?
```

```plaintext
ctx ⊔ Γ(x_H) ⊆ Γ(x_L)?
```
Why Dynamic Enforcement?

- **Static enforcement:** Rejects program if execution could violates Flow-Label invariant
- **Dynamic enforcement:** Blocks after partial execution when Flow-Label invariant could be violated.

```
check: ctx ⊔ Γ(2) ⊑ Γ(x_L)?

if 0 = 0 then x_L := 2 else x_L := x_h fi

ctx = L
```
Implementing Dynamic Enforcement

Conjecture: To implement dynamic enforcement:
- Precede $x := \text{Expr}$ with check: “$\text{ctx} \sqcup \Gamma(\text{Expr}) \subseteq \Gamma(x)$?”
- Block execution if check fails.
Implementing Dynamic Enforcement

Conjecture: To implement dynamic enforcement:

- Precede $x := Expr$ with check: “$\text{ctx} \sqcup \Gamma(Expr) \sqsubseteq \Gamma(x)$?”
- Block execution if check fails

```
x_L := 0
if B then x_L := Expr
  else skip
fi
```
Implementing Dynamic Enforcement

Conjecture:
- Precede $x := \text{Expr}$ with check: “$\text{ctx} \sqcup \Gamma(\text{Expr}) \sqsubseteq \Gamma(x)$?”
- Block execution if check fails

$$x_L := 0$$
$$\text{if } B \text{ then } x_L := \text{Expr}$$
$$\text{else } \text{skip}$$
$$\text{fi}$$

But... when stop on check:
- ... $B=$true leaks!
- Result: implemented RNI (=termination insensitive) only
Solution: Hybrid Enforcement

\[ x_L := 0 \]

\[ \text{if } B \text{ then } x_L := \text{Expr} \]
\[ \quad \text{else } \text{skip} \]
\[ \text{fi} \]

- \( B \rightarrow x_L \) whether or not \( x_L := \text{Expr} \) executes.
  - For \( \Gamma(B) = H \), could exist memories \( M \) and \( M' \) with different \( H \) values causing termination with \( x_L \) having different values.
Solution: Hybrid Enforcement

\[ x_L := 0 \]

\[ \text{if } B \text{ then } x_L := \text{Expr} \]

\[ \text{else } \text{skip} \]

\[ \text{fi} \]

- \( B \rightarrow x_L \) whether or not \( x_L := \text{Expr} \) executes.
  - For \( \Gamma(B) = H \), could exist memories \( M \) and \( M' \) with different \( H \) values causing termination with \( x_L \) having different values.
  - FLI requires \( \Gamma(B) \subseteq x_L \)
    - Before if -or- Within then and within else
  - FLI also requires \( \Gamma(\text{Expr}) \subseteq \Gamma(x_L) \) before \( x_L := \text{Expr} \)
Solution: Hybrid Enforcement

- $x_L := 0$

if $B$ then $x_L := \text{Expr}$
else skip
fi

- $B \rightarrow x_L$ whether or not $x_L := \text{Expr}$ executes.
  - For $\Gamma(B) = H$, could exist memories $M$ and $M'$ with different $H$ values causing termination with $x_L$ having different values.
  - FLI requires $\Gamma(B) \sqsubseteq x_L$
    - Before $\text{if}$ -or- Within $\text{then}$ and within $\text{else}$ -or- After $\text{if}$
  - FLI also requires $\Gamma(\text{Expr}) \sqsubseteq \Gamma(x_L)$ before $x_L := \text{Expr}$

- What if $B$ is $x_H \neq x_H$? (...) enforcement is conservative.)
Hybrid Enforcement: Summary

\[ \text{if } B \text{ then } C_1 \text{ else } C_2 \text{ fi} \]

- Insert check \( \Gamma(\text{Expr}) \subsetneq \Gamma(x) \) before execution of each \( "x := \text{Expr}" \) in \( C_1 \) or \( C_2 \).

- Insert check \( \Gamma(B) \subsetneq \Gamma(x) \) within execution of both \( C_1 \) and \( C_2 \) if \( "x := ..." \) appears anywhere within \( C_1 \) or within \( C_2 \).
Flow-Sensitive Labels

A given variable might be given different **flow-sensitive** labels during execution.

Example:

\[ x := \text{Hval}; \quad x := 0; \quad x_L := x \]

Observe:

- If \( \Gamma(x) = \text{H} \) then program does not type check.
Flow-Sensitive Labels

A given variable might be given different flow-sensitive labels during execution.

Example:

\[ x := \text{Hval}; \quad x := 0; \quad x_L := x \]

red given label \( \text{H} \); green given label \( \text{L} \)

Program does type check and satisfies:

\[ v \rightarrow w \Rightarrow \Gamma(v) \subseteq \Gamma(w) \]
Flow Sensitive Labels + Dynamic?

\[ x := 0 \quad \{ \Gamma(x) = L \} \]

\[ \text{if } h > 0 \quad \text{then} \quad x := 2; \quad \{ \Gamma(x) = \Gamma(h) = H \} \]
\[ \quad \text{else} \quad \text{skip} \]
\[ \text{fi} \]

- \( h > 0 \) is true: After fi \( \Gamma(x) = H \)
- \( h > 0 \) is false: After fi \( \Gamma(x) = L \)

**Problem:** \( h \rightarrow x \) but \( \Gamma(h) \not\subseteq \Gamma(x) \)
Flow Sensitive + …

**Rule:** Block execution from entering conditional commands with high guards and lower targets.

```
x := 0
if h > 0 then x := 2
else skip
fi
```

Stop here!
Flow Sensitive + …

Rule: Update labels of target variables in untaken branches to capture implicit flow.

\begin{align*}
x &:= 0 \\
\text{if } h > 0 \text{ then } & x := 2; \quad \Gamma(x) := \Gamma(h) \\
\text{else} & \quad \text{skip}; \quad \Gamma(x) := \Gamma(h) \\
\text{fi}
\end{align*}
Leaks thru Flow-Sensitive Labels

Suppose: $\Gamma(m) = M$ and $L \subseteq M \subseteq H$

if $m > 0$ then $w := hi$ else $w := lo$ fi
Leaks thru Flow-Sensitive Labels

Suppose: $\Gamma(m) = M$ and $L \subseteq M \subseteq H$

\[
\begin{align*}
\text{false} & \quad M \\
\text{if } m > 0 \text{ then } & w := \text{hi} \quad \text{else } w := \text{lo f i}
\end{align*}
\]
Leaks thru Flow-Sensitive Labels

Suppose: \( \Gamma(m) = M \) and \( L \subseteq M \subseteq H \)

\[
\text{if } m > 0 \text{ then } w := \text{hi} \text{ else } w := \text{lo fi}
\]

- Value of \( m \) leaks to label \((M \text{ vs } H)\) of \( w \).
Avoiding Leaks thru Flow Sensitive 1

**Rule:** Use the same flow-sensitive label for an assignment target, independent of guard.

Example

```
if m > 0 then w := hi else w := lo fi
```

(Sound but conservative.)
Avoiding Leaks thru Flow Sensitive 2

**Rule:** Associate a metalabel with each label.

Example:

```
false
if m > 0 then w := hi else w := lo fi
true

⟨ M, M ⟩
⟨ H, M ⟩
```

Labels for meta-labels?
Summary

**FLI:** $v \rightarrow w \implies \Gamma(v) \subseteq \Gamma(w)$

- **Static enforcement**
  - Conservative

- **Dynamic enforcement**
  - Insert tests
    - Mind the untaken assignment!
  - Change labels
    - Static
    - Dynamic: Leaks thru labels