Hybrid Information Flow monitoring against Web tracking

Frederic Besson, Nataliia Bielova, Thomas Jensen
Inria, France

AJACS meeting
17 December 2014

Panopticlick

How Unique — and Trackable — Is Your Browser?

[Eckersley'10]

Your browser fingerprint **appears to be unique** among the 2,419,678 tested so far.

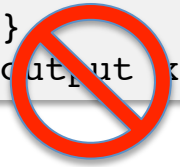
Currently, we estimate that your browser has a fingerprint that conveys **at least 21.21 bits of identifying information.**

- Information needed to **uniquely identify a browser**
  - n – number of connected devices: 5 000 000 000
  - $log_2 n$ – number of bits for a unique id: 33 bits
- **Idea: distinguish** users **by browser fingerprints**:
  - HTTP headers
  - Browser and OS features: language, plugins, fonts, screen, …

The most identifying features (via JavaScript and Flash)

Some scripts are useful

```
var x = 0;
if (name == "FireFox") {
     x = 1;
}
output x;
```

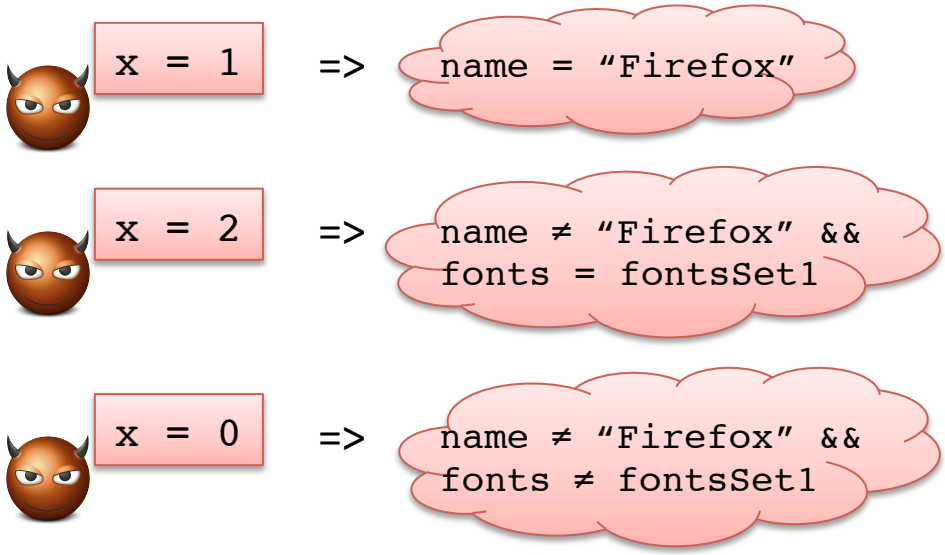


`name:` browser name
`output x:` request containing `x` sent

Non-interference is too restrictive:
`x` depends on name

What does tracker learn?

```
var x = 0;
if (name == "Firefox") {
    x = 1;
}
else {
    if (fonts == fontsSet1) {
        x = 2;
    }
}
output x;
```



Depending on user's browser, **different executions** of this script **leak different quantity** of information!

Quantification of leakage

- **Self-information, or "surprisal"**
  - "amount of information about the identity" [Eckersley'10]
  - = beliefs for deterministic programs [Clarkson, Myers, Schneider'07]

$$Leak(A) = -\log_2 P(A)$$

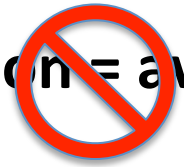
```
var x = 0;
if (name == "FireFox"){
    x = 1;
}
output x;
```

Popularity of "FireFox" is 21%

$$Leak\ (name = \text{"FireFox"}) = -\log_2 0.21 = 2.25 \text{ bits}$$

$$Leak\ (name \neq \text{"FireFox"}) = -\log_2 0.79 = 0.34 \text{ bits}$$

- **Entropy-based definition = average leakage for all browsers!**

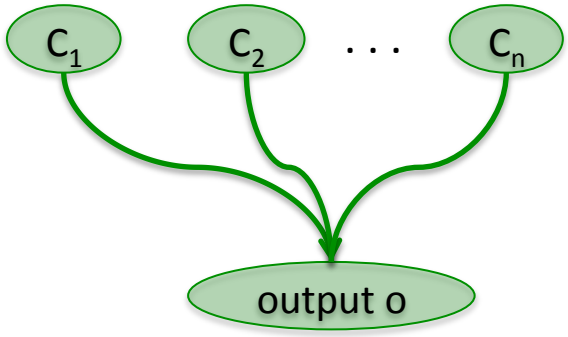


$$H(name) - H(name \mid x) = 0.74 \text{ bits}$$

Knowledge of tracker: configurations

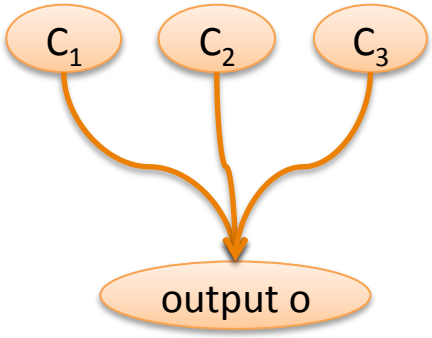
- Browser configuration $C : Features \rightarrow Val$
- $Features = \{name, fonts, \ldots\}$ and $C(name) = \text{“FireFox”}$
- Leakage by **self-information**: $Leak(A) = -\log_2 P(A)$

Noninterference
All configurations



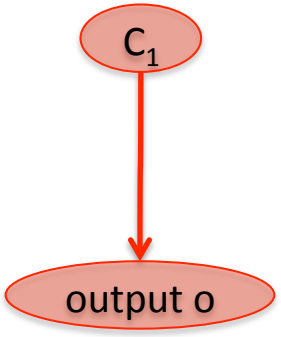
$-\log_2(P(C_1) + \ldots + P(C_n)) = -\log_2 1 = 0$ bits

Partial leakage
Some configurations



$-\log_2(P(C_1) + P(C_2) + P(C_3))$ bits

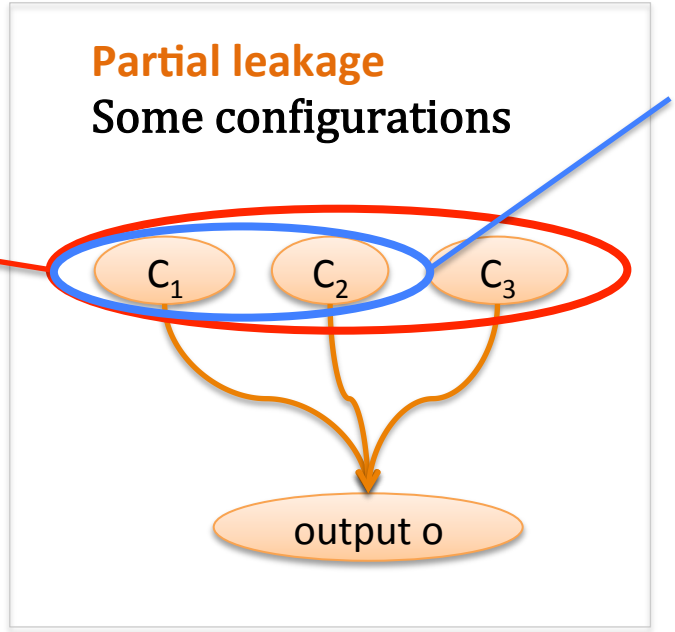
Complete leakage
One configuration



$-\log_2 P(C_1)$ bits

Knowledge of tracker: configurations

Actual knowledge of tracker is a set of equivalent configurations $Eq(P,C)$



We over-approximate knowledge by a set of configurations

Smaller set induces a bigger leakage:

$S_1 \subseteq S_2 \Rightarrow Leak(S_1) \geq Leak(S_2)$

Knowledge of tracker: formula

- Set of configurations represented by a formula

$$B ::= tt \mid ff \mid f = v \mid f \neq v \mid B \wedge B \mid B \vee B$$

f: browser feature
v: value

Noninterference
All configurations

$\{C_1, C_2, \ldots, C_n\}$

tt

Partial leakage
Some configurations

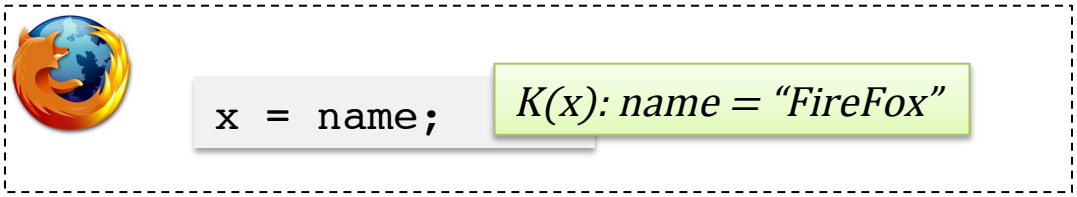
$\{C_i \mid C_i(\text{name}) = \text{“FireFox”} \wedge C_i(\text{fonts}) \neq \text{fontsSet}\}$

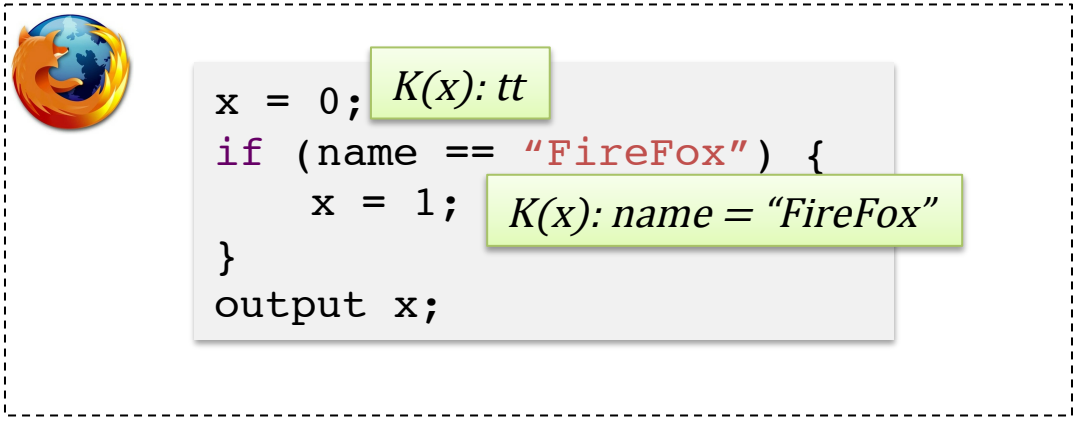
$name = \text{“FireFox”} \wedge fonts \neq fontsSet$

Stronger formula induces a bigger leakage:
$(a_1 \Rightarrow a_2) \Rightarrow Leak(a_1) \geq Leak(a_2)$

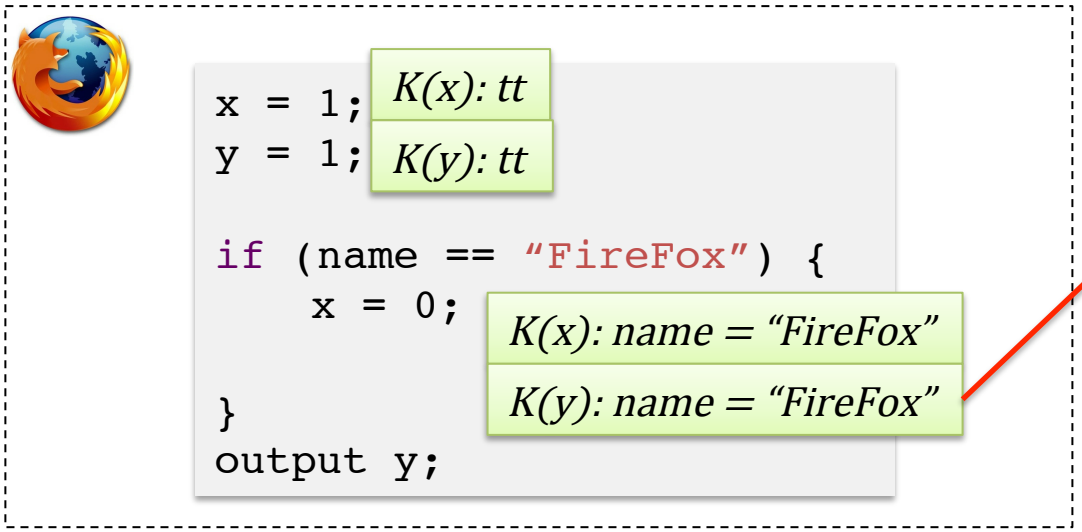
Dynamic knowledge propagation

- Dynamic labeling $K: Vars \rightarrow Formula$
  - for browser features: $K(name): name = \text{“FireFox”}$





Dynamic knowledge propagation



Crucial for soundness:
***y* could be updated in the other branch**

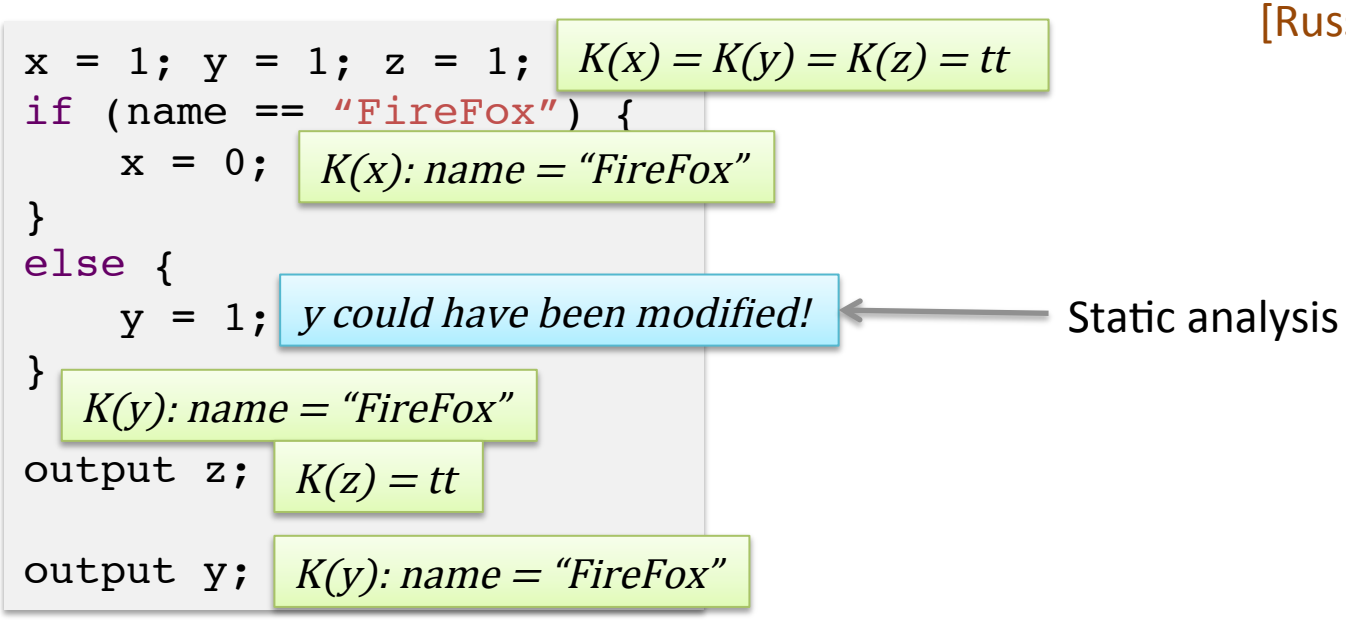
Resembles
No-sensitive upgrade

Dynamic analysis **is not very precise!**

Let's statically analyze non-executed branches!

Hybrid monitor + syntactic checks

[Le Guernic et al.'07]
[Russo and Sabelfeld'10]



We can gain precision by tracking the values!

HM + constant propagation and dependency

- Dynamic analysis: $env: Var \rightarrow Val$
- Static analysis: $env: Var \rightarrow Val \cup \{T\}$

```
var x = 1;
var y = fonts;

if (name == "FireFox") {
    x = 1;
}
else {
    if (y != fontsSet) {
        x = 2;
    }
}
output x;
```

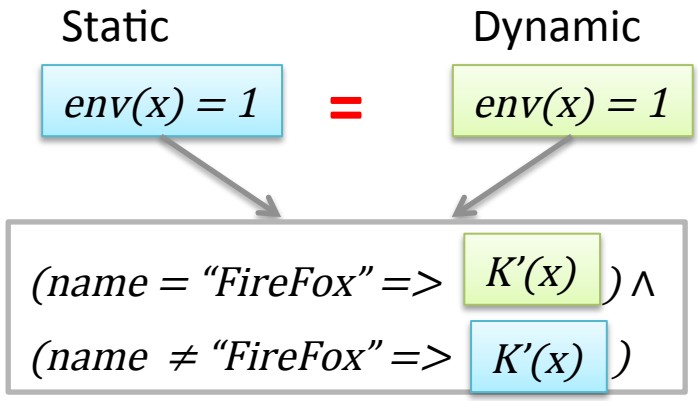
$env(x) = 1$

$K(y)$: $fonts = fontsSet$

$env(x) = 1$ $K'(x)$: tt

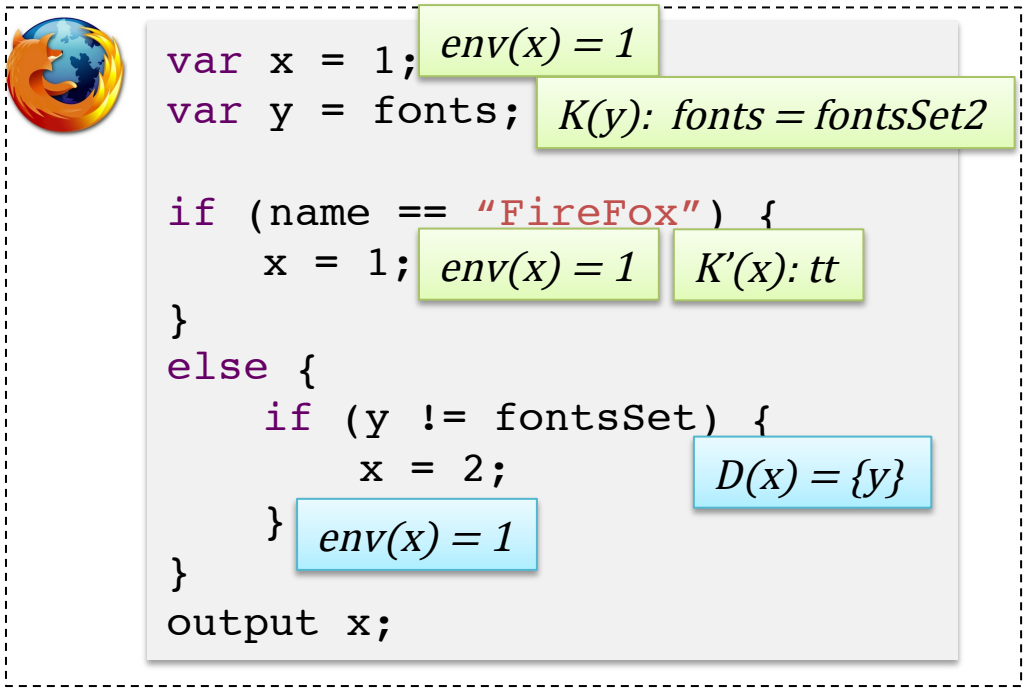
$env(x) = 1$

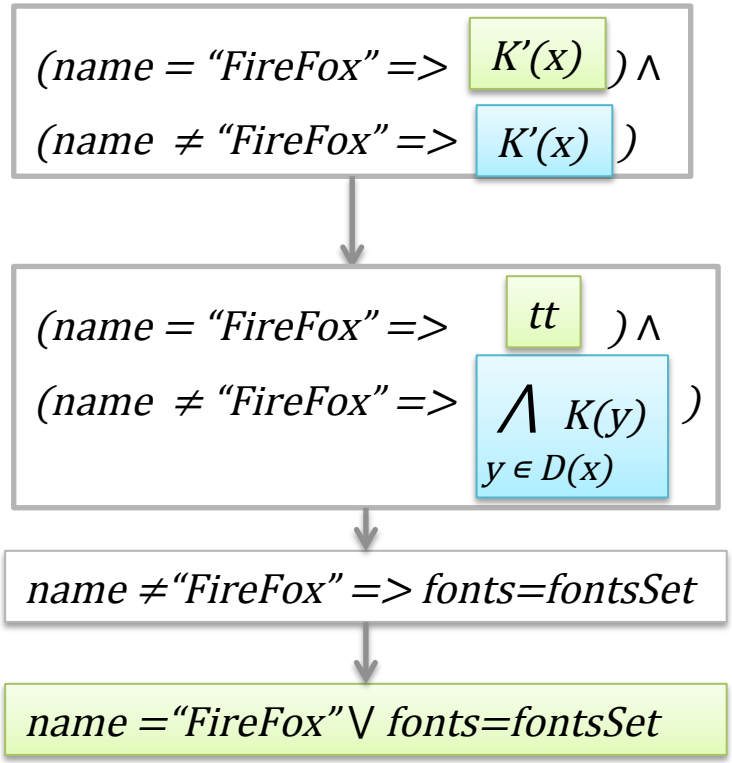
Combination of knowledge in $K(x)$



HM + constant propagation and dependency

- Dependency analysis $D: Var \rightarrow 2^{Var}$





HM + constant propagation and dependency

One rule of hybrid monitor:

Static analysis uses the same environment!

$$[\text{IFTHEN}]\ \frac{[\![B]\!]^{\rho}_{C} \qquad (S_1,(\rho,K)) \Downarrow_C s' \qquad (S_2,\rho) \Downarrow^{\sharp} s^{\sharp}}{(\text{if } B \text{ then } S_1 \text{ else } S_2,(\rho,K)) \Downarrow_C \{\!|B,K,s',s^{\sharp}|\!\}_{\rho}}$$

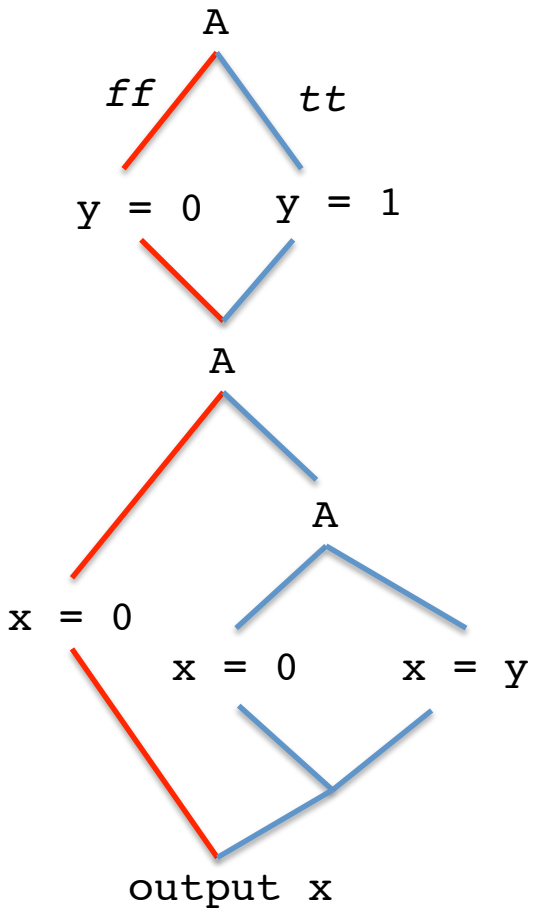
$$\{\!|B,K,(\rho',K'),(\rho^{\sharp},D)|\!\}_{\rho} = (\rho',K''), \text{ where}$$

$$K''(x) = \begin{cases} \left(\neg\delta(B)^{K}_{\rho} \Rightarrow \bigwedge_{y\in D(x)} K(y)\right) \wedge \left(\neg\bar{\delta}(B)^{K}_{\rho} \Rightarrow K'(x)\right) & \text{if } \rho^{\sharp}(x) = \rho'(x) \\ \delta(B)^{K}_{\rho} \wedge K'(x) & \text{otherwise} \end{cases}$$

Resembles weakest precondition:

$$wp(\text{if } B \text{ then } S_1 \text{ else } S_2) = \bigwedge \begin{pmatrix} \neg B & \Rightarrow & wp(S_2) \\ B & \Rightarrow & wp(S_1) \end{pmatrix}$$

Example

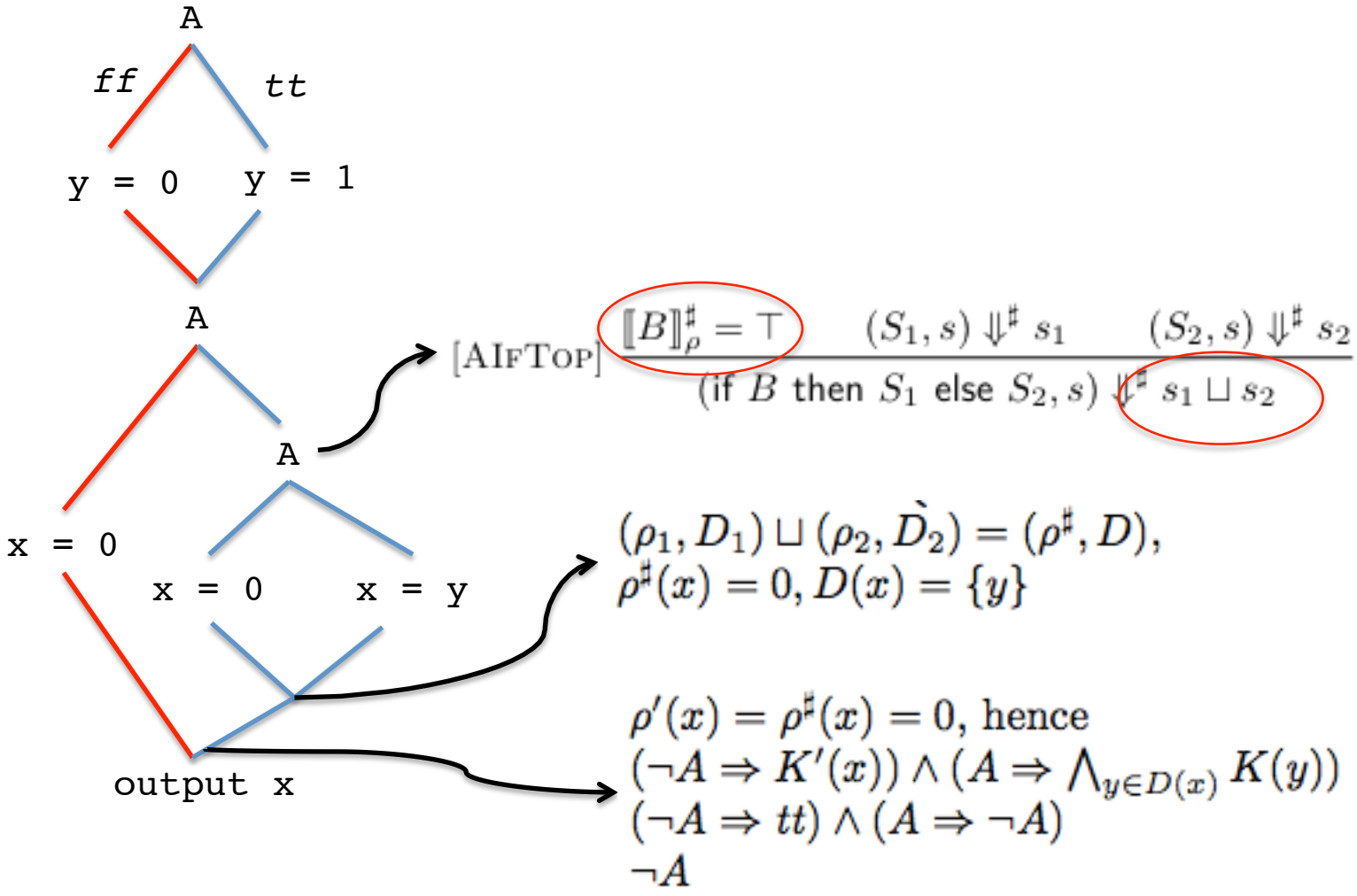


A === (name == “FireFox”)

```
var x = 0;
var y = 0;

if (A) { y = 1;}
else {y = 0;}
if (A) {
    if (A) { x = 0;}
    else { x = y;}
}
else { x = 0;}
output x;
```

Example


A
ff
tt
y = 0
y = 1
A
A
x = 0
x = 0
x = y
output x
[AIfTop]
$\llbracket B \rrbracket^\sharp_\rho = \top \qquad (S_1, s) \Downarrow^\sharp s_1 \qquad (S_2, s) \Downarrow^\sharp s_2$
$(\text{if } B \text{ then } S_1 \text{ else } S_2, s) \Downarrow^\sharp s_1 \sqcup s_2$
$(\rho_1, D_1) \sqcup (\rho_2, D_2) = (\rho^\sharp, D),$
$\rho^\sharp(x) = 0, D(x) = \{y\}$
$\rho'(x) = \rho^\sharp(x) = 0$, hence
$(\neg A \Rightarrow K'(x)) \wedge (A \Rightarrow \bigwedge_{y \in D(x)} K(y))$
$(\neg A \Rightarrow tt) \wedge (A \Rightarrow \neg A)$
$\neg A$

Soundness and Precision

Actual knowledge of tracker is a set of equivalent configurations $Eq(P,C)$

Definition (Soundness)

A hybrid monitor is **sound** if for all variables x, $K(x)$ over-approximates the knowledge of the tracker

$$Models(K(x)) \subseteq Eq(P,C)$$

Theorem (Soundness)

A **sound** static analysis induces a **sound** hybrid monitor.

All the theorems are proven in Coq: http://www.irisa.fr/celtique/ext/QIF/

Soundness and Precision

Definition (Precision)

A hybrid monitor A is **more precise than** a hybrid monitor B, if for all variables x:

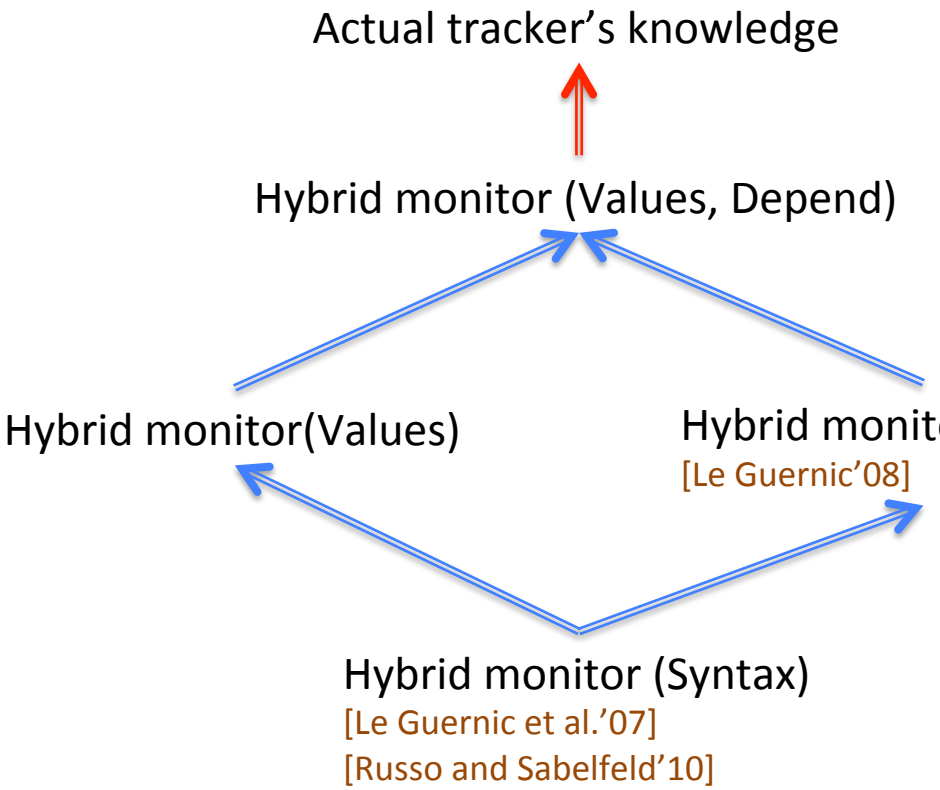
$$Models(K_B(x)) \subseteq Models(K_A(x))$$

Theorem (Precision)

A **more precise** static analysis induces a **more precise** monitor.

All the theorems are proven in Coq: http://www.irisa.fr/celtique/ext/QIF/

Hierarchy of hybrid monitors parameterized by static analysis



All monitors over-approximate tracker's knowledge

Our monitor approximates tracker's knowledge more precisely that other monitors

All the relations are proven in Coq: http://www.irisa.fr/celtique/ext/QIF/

Our results

- Hybrid information flow monitoring
  - Labeling with knowledge
  - Knowledge => quantitative leakage
  - Parameterization by static analysis
- Soundness and precision
  - Requirements for static analysis
  - Easy comparison of hybrid monitors
- Hierarchy of hybrid monitors ordered by precision
  - Constant propagation + dependency analysis => more precise monitor