Hybrid Information Flow monitoring against Web tracking

Frederic Besson, Nataliia Bielova, Thomas Jensen
Inria, France

26th IEEE Computer Security Foundations Symposium
June 26-28, 2013
Alexa-top 10,000 sites [Nikiforakis et al. 12]
• 88.45% of sites have at least one remote JavaScript
• per site: up to 295 remote JavaScript

Nataliia Bielova, CSF'13
How can they track me?

• **Stateful tracking: well-known and getting addressed**
  - Third-party cookies blocking
  - Non-interference for JavaScript
  - EU e-Privacy directive

  [Austin, Flanagan 12]
  [De Groef et al. 12]
  [Hedin, Sabelfeld 12]

• **Stateless tracking: not addressed**
  - IP address tracking
  - Web browser fingerprinting

Nataliia Bielova, CSF'13
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, ...

---

[Nataliia Bielova, CSF'13]
Some scripts are useful

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

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

name: browser name
output $x$: request containing $x$ sent
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!
Quantitative information flow

- Traditional model:
  - Decrease in uncertainty: entropy-based [Smith’09]
  - Increase in accuracy: belief-based [Clarkson, Myers, Schneider’07]

- Traditional analysis:
  - Static analysis for all program executions
    [Clark, Hunt, Malacaria’07] [Mardziel, Magill, Hicks, Srivatsa’11]

- Our approach:
  - Monitor one program execution and quantify leakage
Quantification of leakage

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

\[
I(A) = -\log_2 P(A)
\]

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

Popularity of “FireFox” is 21%

\[
I(name = "FireFox") = -\log_2 0.21 = 2.25 \text{ bits}
\]

\[
I(name \neq "FireFox") = -\log_2 0.79 = 0.34 \text{ bits}
\]

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

\[
H(name) - H(name \mid x) = 0.74 \text{ bits}
\]
The rest of this talk

- **Hybrid monitoring** for quantitative information flow
  - Knowledge representation
  - Labeling propagation

- **Soundness and precision**

- **Hierarchy** of hybrid monitors ordered by precision
Knowledge of tracker: configurations

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

Noninterference
All configurations

Partial leakage
Some configurations

Complete leakage
One configuration

- $-\log_2 (P(C_1) + \ldots + P(C_n)) = -\log_2 1 = 0 \text{ bits}$
- $-\log_2 (P(C_1) + P(C_2) + P(C_3)) \text{ bits}$
- $-\log_2 P(C_1) \text{ bits}$
Knowledge of tracker: configurations

Actual knowledge of tracker is a set of equivalent configurations \( \text{Eq}(P,C) \)

Partial leakage
Some configurations

We over-approximate knowledge by a set of configurations

Smaller set induces a bigger leakage:
\[
-\log_2(P(C_1) + P(C_2) + P(C_3)) \leq -\log_2(P(C_1) + P(C_2))
\]
Knowledge of tracker: formula

- Set of configurations represented by a formula

\[ B ::= tt \mid ff \mid f = v \mid f \neq v \mid B \land B \mid B \lor B \]

$f$: browser feature
$v$: value

**Noninterference**
All configurations
\{C_1, C_2, ..., C_n\}

**Partial leakage**
Some configurations
\{C_i \mid C_i(name) = \text{“FireFox”} \land C_i(fonts) \neq fontsSet\}

\[ name = \text{“Fire Fox”} \land fonts \neq fontsSet \]
Dynamic knowledge propagation

- Dynamic labeling
  - for browser features:

\[ K(name): \text{name} = "FireFox" \]
Dynamic knowledge propagation

\[ x = 1; \]
\[ K(x): tt \]
\[ if \ (name == \ "FireFox") \ { \]
\[ x = 1; \]
\[ K(x): name = \"FireFox\" \]
\[ } \]
\[ output \ x; \]

Dynamic analysis is not very precise!

Let’s statically analyze non-executed branches!
Hybrid Monitoring

- Dynamic analysis:
  - Static analysis:

\[
\begin{align*}
env: & \text{Var} \rightarrow \text{Val} \\
env: & \text{Var} \rightarrow \text{Val} \cup \{T\}
\end{align*}
\]

\begin{verbatim}
var x = 1;
var y = fonts;
if (name == "FireFox") {
    x = 1;
}
else {
    if (y != fontsSet) {
        x = 2;
    }
}
output x;
\end{verbatim}

Combination of knowledge in \( K(x) \)

Static
- \( env(x) = 1 \)
- \( K(y): \text{fonts} = \text{fontsSet} \)

Dynamic
- \( env(x) = 1 \)
- \( (name = "FireFox" \Rightarrow K'(x)) \land \\
  (name \neq "FireFox" \Rightarrow K'(x)) \)

name = “FireFox” OR fonts = fontsSet
Static analysis

• Dependency analysis

\[ D: \text{Var} \rightarrow 2^{\text{Var}} \]

\begin{verbatim}
var x = 1;   \hspace{1cm} \text{env}(x) = 1
var y = fonts; \hspace{1cm} K(y): \text{fonts} = \text{fontsSet2}

if (name == "FireFox") {
    x = 1;   \hspace{1cm} \text{env}(x) = 1 \hspace{1cm} K'(x): tt
}
else {
    if (y != fontsSet) {
        x = 2;   \hspace{1cm} D(x) = \{y\}
    }
    \hspace{1cm} \text{env}(x) = 1
}
output x;
\end{verbatim}

(name = “FireFox” \Rightarrow K’(x)) \land 
(name \neq “FireFox” \Rightarrow K’(x))

(name = “FireFox” \Rightarrow tt) \land 
(name \neq “FireFox” \Rightarrow \bigwedge K(y) \hspace{1cm} y \in D(x))

name \neq “FireFox” \Rightarrow fonts = fontsSet

name = “FireFox” \lor fonts = fontsSet
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

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

**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/](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:

\[ \text{Models}(K_B(x)) \subseteq \text{Models}(K_A(x)) \]

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/](http://www.irisa.fr/celtique/ext/QIF/)
Hierarchy of hybrid monitors parameterized by static analysis

Actual tracker’s knowledge

Hybrid monitor (Values, Depend)

Hybrid monitor (Values)

Hybrid monitor (Syntax)

Hybrid monitor (Depend)

[Le Guernic’08]

[Le Guernic et al.’07]

[Russo and Sabelfeld’10]

All monitors over-approximate tracker’s knowledge

Our monitor approximates tracker’s knowledge more precisely than that other monitors

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

Nataliia Bielova, CSF’13
Future work

• Support for enforcement
  ▪ threshold-based enforcement
  ▪ possible leakage due to enforcement action

• Extension to Java-like language
  ▪ and, eventually, to JavaScript-like language
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